Active Deep Learning of Representations for Similarity Search

Aktives tiefes Lernen von Repräsentationen für die Ähnlichkeitssuche

Der Technischen Fakultät
der Friedrich-Alexander-Universität
Erlangen-Nürnberg

zur
Erlangung des Doktorgrades Dr.-Ing.

vorgelegt von

Christoffer Löffler, M.Sc.

aus Nürnberg

Als Dissertation genehmigt
von der Technischen Fakultät
der Friedrich-Alexander-Universität Erlangen-Nürnberg

Tag der mündlichen Prüfung:	23.10.2023
Gutachter:	Prof. Dr. Björn Eskofier
	Prof. Dr. Ute Schmid

Acknowledgements

This thesis would not have been possible without the support of many people, for which I am immensely grateful.

I thank my colleagues at Fraunhofer Institute for Integrated Circuits (IIS), and especially Christopher Mutschler who was always a demanding and motivating mentor, ever since my Bachelor's thesis. I'd also like to thank Nicolas Witt for being a great head of my unit, and for always taking care of my interests as a PhD student. Furthermore, thank you to Tobias Feigl, Georgios Kontes and Jann Goschenhofer as a small sample of representatives of the many colleagues at Fraunhofer who I've shared the best experiences with over the years. You've all provided an excellent environment for research and friendships alike. A special shout out goes to Tonia Christ for jointly tending our wonderful office garden that demanded the admiration of many visitors.

Thank you Prof. Björn Eskofier, not only for hosting me at the MaD Lab, but also for your patient mentoring. I appreciate your guidance in the academic world, the lab's great atmosphere that you've shaped, and that you convey such a positive aspiration for academic research. Thanks also goes to Dario Zanca for the collaboration and thoughtful sharing of experience in the Applied ML group, and to all my lab colleagues for the enjoyable time that I was fortunate enough to have with you. To the students Anes Redžepagić, Christopher Kraus, Felix Ott, Karthik Ayyalasomayajula, Lorenz Gorse, Minhao Qiu, Rosanna Dietrich-Sußner, Sascha Riechel and Wei-Cheng Lai, who put their trust in me as their thesis supervisor, thanks for the humbling experience.

To Prof. Christopher Rozell I'm extending my thanks not only for hosting me at Georgia Tech, which was not an easy setup at all, but also for providing me with new perspectives on conducting research and on teaching. Thank you to Kion Fallah, Stefano Fenu, Matt O'Shaugnessy, Kyle Johnson and all the people at the SIPLab and CODA building for welcoming me so warmly in Atlanta.

My funding agencies allowed me to conduct the research for this thesis and also to contribute to exciting projects. I'd like to thank the Fraunhofer IIS, the German Academic Exchange Service (DAAD), with the program for International Research Visits for Computer Scientists (IFI), and the Bavarian Ministry of Economic Affairs, Infrastructure, Energy and Technology, with the project Leistungszentrum Elektroniksysteme (LZE) and the Center for Analytics-Data-Applications (ADA-Center) within the framework of "BAYERN DIGITAL II".

To my family and friends, thanks for always being there and being interested. Besides listening to me, I was always happy to be distracted from work by things that really matter in life. Thank you to my wonderful wife Andrea Martínez. Your love and support encouraged me to even start on this journey. This thesis would not exist without you.

Abstract

In recent years Deep Learning has revolutionized many fields in computer science such as Computer Vision (CV), Natural Language Processing (NLP), and Information Retrieval (IR). For example, modern DL is at the core of systems that process large amounts of complex data, such as images, video or text, and then retrieve information or even generate semantically meaningful responses to queries within fractions of a second.

However, the (historic) data that is available for many potential applications brings new problems that require human attention. The cleaning, preparing and annotation of data can evolve into a similar issue as the search of a needle in a haystack: take too much time, and increase costs. Some data may even never be annotated in sufficient detail and thus require alternative solutions. In this cumulative dissertation, I address gaps in the literature at three levels of the Machine Learning process, that enable modeling of complex data and reduce cost of annotations.

The first objective considers issues with the retrieval of complex, unstructured and sparsely annotated data from large (historic) databases. We proposed a metric learner [P1] that learns a lower-dimensional representation of the data and thus enables efficient Information Retrieval. It jointly estimates a structure of the unstructured data and learns pairwise similarity, such that a previously prohibitive distance metric can be calculated orders of magnitudes faster.

The second objective consists of a unified Deep Active Learning (DAL) policy that reduces the annotation costs in Deep Learning via the use of Active Learning. We propose Imitating Active Learner Ensembles (IALE) [P2], an Imitation Learning approach to DAL that leverages a learner Deep Neural Network (DNN)'s state and uses different signals to learn how to learn actively from multiple heuristics. Our method then acquires more informative samples than any of these baseline heuristic.

The third objective considers cost-efficient learning of individual similarity functions, in cases where Machine Learning models for Information Retrieval suffer from a semantic gap in the problem domain. In [P3], we propose to learn similarity from few annotated samples by combining fine-tuning of large pre-trained models with Active Learning sampling methods to reduce cost. We present a user study that demonstrates the strong benefits of the sampling method w.r.t. both cost and query difficulty.

To summarize, we contribute to Metric Learning, Deep Active Learning and Information Retrieval, and show adaptive similarity search for unstructured data. Hence, our work facilitates the efficient use of annotators' time and the collection of high quality annotations, even on highly complex data.

Zusammenfassung

In den letzten Jahren haben tiefe lernende Verfahren die Informatik revolutioniert. So bilden tiefe lernende Verfahren den Kern von Systemen, die große Mengen komplexer Daten wie bspw. Bilder, Videos oder Texte verarbeiten, und innerhalb von Sekundenbruchteilen semantisch aussagekräftige Antworten auf Anfragen finden oder sogar synthetisieren können.

Die (historischen) Daten, die für viele potenzielle Anwendungen zur Verfügung stehen, bringen jedoch neue Probleme mit sich, die menschliche Aufmerksamkeit erfordern. Die Bereinigung, Aufbereitung und Annotation dieser Daten entwickelt sich zu einem ähnlichen Problem wie die Suche nach der metaphorischen Nadel im Heuhaufen: Die Arbeiten dauert lange und treiben die Kosten in die Höhe. Manches kann sogar nie ausreichend annotiert werden und erfordert daher alternative Lösungen. In dieser kumulativen Dissertation befasse ich mich mit den Lücken in der Literatur auf drei Ebenen des maschinellen Lernens. Meine Forschungsziele sind die Modellierung komplexer Daten per tiefer lernender Verfahren sowie die Senkung der Kosten für Annotationen.

Das erste Ziel befasst sich mit der schnellen Ähnlichkeitssuche für komplexe, unstrukturierte und spärlich annotierte Daten innerhalb von großen (historischen) Datenbeständen. Wir stellen einen Ansatz des metrischen Lernens vor [P1], der eine niedrigdimensionale Repräsentation der Daten erlernt und so ein effizientes Durchsuchen der Daten ermöglicht. Das System lernt sowohl eine Struktur der unstrukturierten Daten und als auch ein Maß einer paarweise Ähnlichkeit, so dass eine zuvor zu teure Distanzmetrik um Größenordnungen schneller berechnet werden kann.

Das zweite Ziel besteht in der Entwicklung einer besseren Strategie für aktives Lernen für tiefe lernende Verfahren, wodurch Annotationskosten reduziert werden. Wir stellen mit Imitating Active Learner Ensembles [P2] ein Verfahren für aktives Lernen vor, das den Zustand eines tiefen neuronalen Netzes nutzt, um selbst zu lernen, welche Strategie des aktiven Lernens für das Zielnetz zu welchem Zeitpunkt einzusetzen ist. Unsere Methode wählt informativere Daten zum Annotieren aus, als andere Basisheuristiken.

Das dritte Ziel ist das kosteneffiziente Lernen individueller Ähnlichkeitsfunktionen in Fällen, in denen Modelle des maschinellen Lernens für die Ähnlichkeitssuche aufgrund der semantischen Lücke in der Problemdomäne suboptimal sind. In [P3] stellen wir ein Verfahren hierfür vor, welches eine Ähnlichkeitsmetrik von nur wenigen annotierten Daten erlernt. Dazu kombinieren wir zur Kostenreduktion ein vortrainiertes Modell mit einer optimierten Methodik des aktiven Lernens. Im Zuge einer Nutzerstudie zeigen wir die Vorteile der Optimierung sowohl in Bezug auf die Kosten als auch auf die Schwierigkeit der Annotationsaufgabe vor.

Zusammenfassend tragen wir zu den Themen des metrischen Lernens, des aktiven Lernens von tiefen Verfahren, und des Informationsabrufs bei, und zeigen eine adaptive Ähnlichkeitssuche für unstrukturierte Daten. Unsere Arbeit ermöglicht hierdurch eine effizienter Nutzung der Zeit von Annotatoren und die Sammlung höherwertiger Annotationen, selbst bei der Verarbeitung von hochkomplexen Daten.

Contents

List of Symbols and Abbreviations

List of Abbreviations

Abbreviation	Description
AI	Artificial Intelligence
AL	Active Learning
ALBL	Active Learning by Learning
ANN	Artificial Neural Network
AR	Augmented Reality
AUC	Area Under Curve
AutoML	Automatic Machine Learning
BADGE	Batch Active learning by Diverse Gradient Embeddings
BALD	Bayesian Active Learning by Disagreement
BNN	Bayesian neural network
cGAN	conditional Generative Adversarial Network
CNN	Convolutional Neural Network
COMB	combination algorithm
CV	Computer Vision
DAL	Deep Active Learning
DB	database
DFD	Discrete Fréchet Distance
DL	Deep Learning
DML	Deep Metric Learning
DNN	Deep Neural Network
DTW	Dynamic Time Warping
EMD	Earth Movers Distance
FAU	Friedrich-Alexander-Universät Erlangen-Nürnberg
GAN	Generative Adversarial Network
GDPR	General Data Protection Regulation
IALE	Imitating Active Learner Ensembles
IF	Information Filtering
IL	Imitation Learning
IR	Information Retrieval
L2T	Learning to teach

Abbreviation	Description
LCSS	Longest Common Subsequence
LIME	Local Interpretable Model-agnostic Explanations
LSA	Linear Strategy Aggregation
LSTM	Long Short-Term Memory
MAB	Multi-Armed Bandit
MDS	Multidimensional Scaling
ML	Machine Learning
MLP	Multilayer Perceptron
NAS	Neural Architecture Search
NER	Named Entity Recognition
NLP	Natural Language Processing
OOD	Out of Distribution
OT	Optimal Transport
RF	Radio Frequency
RL	Reinforcement Learning
RNN	Recurrent Neural Network
SGD	Stochastic Gradient Descent
SGNS	Skip-Gram with Negative Sampling
SHAP	SHapley Additive exPlanations
SSIM	Structural Similarity Index Measure
t-SNE	t-Distributed Stochastic Neighbor Embedding
TA-VAAL	Task-Aware Variational Adversarial Learning
TCAV	Testing with Concept Activation Vectors
tSTE	t-Stochastic Triplet Embedding
VAAL	Variational Adversarial Active Learning
VAE	Variational Autoencoder
XAI	eXplainable Artificial Intelligence

List of Symbols

Symbol	Unit	Description
a		Action
B		Batch
d		Distance

Symbol	Unit	Description
f		Function representing a neural network
$\mathcal{L}$		Loss function
l_2		Euclidean distance
l_∞		Chebyshev distance
$\mathcal{O}$		Landau's symbol describing asymptotic behavior of complexity
p^-		Negative sample in a triplet
p^+		Positive sample in a triplet
p		Anchor sample in a triplet
π		Policy function
Q		Q function computing expected reward for an action given a state
R		Reward in Reinforcement Learning
S		Similarity function
S		State space
s		State of a policy
t		Triplet
T		Set of triplets
θ		Parameters of a neural network
x		Sample
$\mathcal{X}$		Dataset X
y		Annotation or label

List of Figures

Part I

Introduction

1 Motivation

In recent years Deep Learning (DL) has revolutionized many fields in computer science such as Computer Vision (CV), Natural Language Processing (NLP), and Information Retrieval (IR). The adoption of deep learning came naturally to many real-world applications. For many potential applications, (historic) data is available which can be used for learning. However, the analogy of the needle in the haystack applies: even if the data can be cleaned and prepared for learning, it may often still be too unstructured and lacking annotations, that are mandatory for many of the most successful learning approaches.

First, such missing structure may be even more severe than simply missing labels, as the fundamental comparability of two samples is not given, if the structure of the sample is not provided. One example of this problem is the missing ordering of data of a sample of multi-agent trajectory tracking like in the team sport football, where no global ordering exists. There is a need for a robust method to estimate such ordering, especially in tasks such as IR. A form of this task is similarity search, that relies on comparability of pairs of samples to retrieve similar information from large databases.

Second, annotations may take the form of labels. The generation of high quality labels is typically a task that involves human labor. However, such human participation may take time and be costly. The optimization of this cost is an active field of research called Active Learning (AL), that puts humans into the training loop and queries them for the most informative annotations. The choice of a suitable algorithm for Deep Active Learning (DAL) is difficult, because it depends on the models and the dataset at hand, which motivates further research.

Third, annotations may take the form of relative similarity between samples, which adds another dimension to the learning task, as there are not necessarily clear class boundaries but rather complex semantics at play. This creates a semantic gap that learning from humans may bridge by imitating the humans' innate similarity metric. One way to performing such active metric learning cost-effectively may be Active Learning.

1.1 Objectives of the thesis

This thesis focuses on three consecutive aspects of Information Retrieval with only few labeled data. First, we enable fast, approximate Information Retrieval on unstructured data using Deep Metric Learning. Then, we enhance Deep Active Learning by learning a unified Active Learning policy. And finally, we propose a novel use of AL for Deep Metric Learning for fast, high-quality similarity search on unstructured data.

Information Retrieval. First, we investigate how we can leverage learned representations of unstructured data in order to accelerate similarity search, and apply it to scenes of football play. The IR operation functions as similarity search, which means it searches the nearest neighbors to a query sample. The representation learned by a DNN would enable retrievals of similar samples by orders of magnitude faster than alternative approaches [1].

Deep Active Learning. Second, deep models require large amounts of labeled training data in supervised learning. Creating labeled datasets can be an expensive task. Traditional

Active Learning [2] uses heuristics or leverages knowledge of the model and data distribution to select what labels are most informative. This reduces the costs of labeling data by querying an oracle only with the most valuable samples for a target model. Deep Active Learning [3] investigates the issues of AL specific to DNNs, such as the composition of batches or balancing diverse and uncertain acquisition functions. Learning a unified, balanced strategy of multiple AL heuristics for batch AL is an objective of this work.

Active Metric Learning. Third, many real-world tasks are beyond the semantic gap and traditional learning methods are not able to predict like human beings, even using state of the art feature detectors [4]. Generally, this may be tackled by learning humans' innate classification or similarity functions directly from from annotations [5]. Note that relative similarity is richer than classification [6] and can be used to generate an ordinal embedding, e.g., encoding inter-class similarities and the variances of a class. However, this would theoretically require many more labels to build a dataset of full pairwise comparisons [4, 5, 6]. Hence, the goal is to annotate a minimal set of annotations necessary, and to estimate the closest possible approximation with as low an annotation cost as possible. The development of an Active Learning method for learning a metric, and conducting a user study on football trajectory data to evaluate the learned embedding are objectives of this work.

1.2 Contributions

This section summarizes the main contributions of this work with respect to the objectives of the doctoral project. For the first objective, Information Retrieval, we contribute Deep Siamese Metric Learning [P1] that learns lower-dimensional representations of unordered, complex spatiotemporal trajectory data. We address the second objective with our novel method IALE [P2], that learns a unified AL strategy for DNNs from experts. The third objective on AL for learning a metric from annotations contributes a DML method and user study evaluation [P3].

- Trajectory retrieval systems are currently limited by the complexity of the high dimensional data and do not estimate an optimal assignment of multiple, unstructured trajectories [1, 7, 8]. Due to this, they are slow and do not scale well. In the journal paper "Deep Siamese Metric Learning: A Highly Scalable Approach to Searching Unordered Sets of Trajectories" [P1], published in *ACM Transactions on Intelligent Systems and Technology* (ACM TIST) in the *Special Issue on Intelligent Trajectory Data Analytics*, we presented a unified approach for IR. Our method uses a Siamese Network to jointly learn how to estimate the optimal assignment between two ensembles of trajectories, e.g., two teams of football players, and also to learn a lower-dimensional embedding of the complex trajectory data, that preserves the distance metric between samples of the dataset. It supports dense or sparse inputs and uses "Gated Temporal Convolutions" as a masking mechanism for sparse inputs. This combination learns vector representations of trajectory ensembles, that accelerate similarity search by orders of magnitude compared to prior approaches.
- In DAL, a recent problem is balancing multiple criteria in one batch acquisition [3]. This can be addressed, for example, by heuristics, that acquire highly informative samples in a batch [9], or by learning an acquisition function using Reinforcement Learning (RL) [10, 11, 12, 13, 14]. We propose an alternative that combines existing heuristics with learning to learn in the journal paper "IALE: Imitating Active Learner

Ensembles" [P2], published in the *Journal of Machine Learning Research* (JMLR) and presented[1] at the *Conference on Neural Information Processing Systems* (NeurIPS), IALE proposes to use imitation learning to learn a policy for AL from an ensemble of a diverse set of DAL algorithms in the batch-mode pool-based settings. The policy leverages a learner DNN's state via "introspection", and uses different signals, e.g., gradients or predictive uncertainty, to learn a suitable acquisition function from multiple heuristics, depending on the state of the AL process. IALE acquires more informative samples than any baseline in our experiments with well-known image classification datasets and is transferable between datasets and even classifier architectures.

- In Metric Learning, bridging the semantic gap while simultaneously requiring as few annotations as possible [4, 6], and learning a predictive model for out-of-sample data [15] is challenging. We propose a method that adapts InfoTuple [5] and triplet mining techniques [16] in the journal paper "Active Learning of Ordinal Embeddings: A User Study on Football Data" [P3], published[2] in the *Transactions of Machine Learning Research* (TMLR). We propose to use the information gain-based AL method Infotuple [5] and increase its efficiency (wrt. label cost and time spent) by constructing queries from a sub-set of the pool using an adapted triplet mining technique [16]. This decreases the computational complexity considerably. Furthermore, we conduct an AL user study to compare different variants of AL (Information Gain, Nearest Neighbor) and of triplet mining while fine-tuning a DNN for learning notions of similarity of a complex football dataset [P1]. We show that AL is superior to passive learning, and that user-specific similarity functions can form relatively consistent groups.

In addition to the main contributions in the journal publications listed above, I have contributed to two more publications on adaptive or interactive ML as the first or shared first author.

- In "A Sense of Quality for Augmented Reality Assisted Process Guidance" [P4], presented at the *IEEE International Symposium on Mixed and Augmented Reality* (ISMAR) as a poster[3], we proposed an Augmented Reality (AR) system for process guidance systems, that uses ML to predict quality metrics. It uses inertial sensors mounted on work tools and an AR headset to classify work steps and guide workers in an assembly process. In this work, I implemented the ML algorithm, discussed the literature and experiments, and wrote the paper.
- In "Automated Quality Assurance for Hand-held Tools via Embedded Classification and AutoML" [P5], presented at the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD) as a demonstration[4], we propose an end-to-end Automatic Machine Learning (AutoML) method and a custom hardware platform for the recording of labeled datasets, the development and the deployment of time series segmentation and classification models on embedded hardware. In this work, I implemented the DNN algorithms, performed experiments, and wrote the paper.

[1] See the presentation at https://neurips.cc/virtual/2022/poster/56122
[2] See the presentation at https://openreview.net/forum?id=0q3tx5kinu
[3] See the fast-forward presentation at https://youtu.be/x9jL5V5i5SM
[4] See the presentation at https://slideslive.com/38932435

Furthermore, I have collaborated with researchers on publications adjacent to my research and contributed to the following publications

- In "Recipes for Post-training Quantization of Deep Neural Networks" [P6], presented[5] on the *EMC^2: Workshop on Energy Efficient Machine Learning and Cognitive Computing*, we evaluated post-training quantization and showed the benefits of greedily selecting an optimal global bit-width. In this work, I contributed parts of the evaluation pipeline, dataset, DNN model, and textual description, and reviewed the paper.
- In "ViPR: Visual-Odometry-aided Pose Regression for 6DoF Camera Localization" [P7], presented[6] at the *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (CVPR) on the *Joint Workshop on Long-Term Visual Localization, Visual Odometry and Geometric and Learning-based SLAM*, we presented an approach to fusing an absolute pose estimation of six degrees of freedom with an optical flow from a camera, e.g., using FlowNet [17], to improve a mobile agent's positioning estimation. In this work, I co-supervised the prior thesis project and reviewed the paper.
- In "Localization Limitations of ARCore, ARKit, and Hololens in Dynamic Large-scale Industry Environments" [P8], presented at the *International Conference on Computer Graphics Theory and Applications* (GRAPP), we conducted an evaluation study of the state-of-the-art AR systems. Our study focused on dynamic, large-scale industrial environments, that pose different real-world challenges than used for entertainment. In this work, I contributed as project manager and reviewed the paper.

1.3 Overview of the thesis

The remainder of the thesis is structured as follows. Part II discusses relevant fundamentals and the state-of-the-art. First, Chapter 2 introduces fundamentals on Information Retrieval (Sec. 2.1), Active Learning (Sec. 2.3) and Active Metric Learning (Sec. 2.4). Next, Chapter 3 presents the current state-of-the-art relevant to this thesis's objectives and is structured analogously to the fundamentals. It presents recent work on Information Retrieval (Sec. 3.1), Active Learning (Sec. 3.2) and Active Metric Learning (Sec. 3.3). Part III outlines the contributed publications that make up the core of this work. Lastly, Part IV discusses the contributions in Chapter 7 and concludes with an outlook in Chapter 8.

[5] See the presentation at https://youtu.be/Gmxh9QgA9Hc?t=14547
[6] See the presentation at https://youtu.be/ZCTxPJPCfFo?t=20577

Part II

Fundamentals and State of the Art

2 Fundamentals

This chapter introduces the necessary fundamentals on Information Retrieval (Sec. 2.1), Deep Metric Learning (Sec. 2.2), Active Learning (Sec. 2.3) and Active Metric Learning (Sec. 2.4) that enable readers to easily follow the state of the art in the Part 3.

2.1 Information Retrieval with Unstructured Trajectories

Information Retrieval from large databases is not only popular in NLP, where it helps for tasks like the building of knowledge bases and extracting data from documents, but also finds attention in many other fields such as sports, e.g., for querying similar plays [1], or transportation, e.g., for querying recurrent convoys of vehicles [18].

Querying for similar samples is coined similarity search [19]. It matches a query sample to other samples from a database (DB) to retrieve the top-n most similar results. If there are enough annotations or metadata available for each sample, then the matching algorithm can take these as a shortcut and perform classical DB look-ups. However, metadata may be available only sparsely or not at all. In similarity search, data is represented as dense vectors instead, that are compared pairwise, using an operation defined on the sample itself, such as the Euclidean distance [1]. However, for some data types, such as multi-agent tracking in team sports, pairwise comparisons are not well defined because agents' pairwise assignments follow no clear convention. Thus they require an expensive additional optimization before similarity can even be computed.

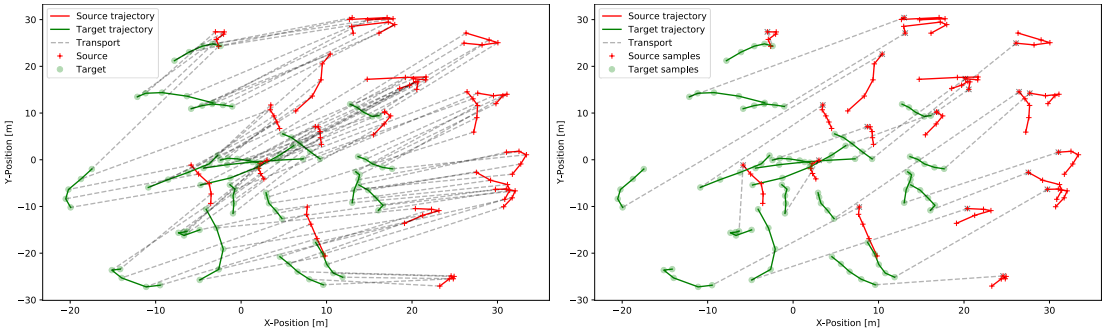
This section first introduces Optimal Transport (OT) as a general approach for the assignment problem in Sec. 2.1.1. Then, Sec. 2.1.2 gives an overview of relevant similarity metrics for IR. Finally, Sec. 2.2 presents Deep Metric Learning as a way of learning low-dimensional, distance-preserving representations for similarity search.

2.1.1 Transport Problem

For domains that involve multiple agents, like the tracking of positions in team sports, it can be impossible to define a stable pairwise distance metric. This is due to the undefined relationships between the tracked entities. For example, in football, the players' roles can change over time as they adapt their strategies to deal with the opposing team [20]. Given two samples of multi-agent tracking data, it is possible to assign each agent to its optimal counterpart in the other sample based on a distance metric.

This problem is located within the more general OT problem, which was initially stated by Monge [21] as finding the least expensive solution for moving dirt from one place to another. One metric to use for solving such problems is the Earth Movers Distance (EMD) or Wasserstein metric [22]. In its general form, Monge stated an optimization problem for finding an optimal mapping T from a source distribution μ_s to a target distribution μ_t while minimizing the cost function $c : \Omega_s \longrightarrow \Omega_t$.

Note that the Wasserstein metric is for general probability distributions, whereas the transport problem for multi-agent trajectory data is a constrained problem that does not allow the division of trajectories into point measures. The players' trajectories, and thus



(a) Point wise Wasserstein transport.

(b) Hungarian solution for summary statistics.

Figure 1: Both plots show positional tracking in football of 20 field players each, for two sets of trajectories (red, green). Each of the total 40 trajectories consists of 5 samples (about 0.2 s). The Wasserstein transport (a) finds an optimal point-wise transport but breaks up semantic trajectories. The Hungarian algorithm over summary statistics (b) instead transports trajectories as semantic units.

their identities, cannot be randomly split up over time. Fig. 1a shows the optimal solution for two different multi-agent trajectory sets, one in red and the other in green. Here the OT is a point-wise one-to-one transport. To preserve the semantics of the football domain, an alternative solution in Fig. 1b finds correspondence between summary statistics of trajectories instead. Then the Hungarian algorithm [23] can be used as an efficient optimal solver.

2.1.2 Similarity Functions

Determining the similarity between two vectors can be implemented via distance calculations. For vectors representing trajectory data, complex distance metrics were proposed, such temporal-aware similarity or segment-based methods [24]. For application scenarios with multiple moving objects, e.g., multiple pedestrians, athletes, vehicles, animals or others, we extend the problem to sets of vectors. For determining similarity of sets of vectors, an assignment between these two sets has to be calculated first, see Sec. 2.1.1. Next, a suitable distance metric for pairs of vectors may be applied.

This work is motivated by the problem of similarity metrics for sets of complex unstructured data. Without loss of generality, we study multi-agent trajectory data from football. With respect to the choice of such a suitable distance metric for this team sport, we follow Sha et al. [1], who have conducted a qualitative analysis of five different distance metrics. They compare their accuracy for 38 different classes of football play. The Euclidean distance compares favorably to more complex distance metrics, such as Dynamic Time Warping (DTW) [25].

The Euclidean distance (l_2 distance) between two vectors p, q can be written as a Mahalanobis distance [26]

$$d_{\text{Mahalanobis}}(p,q) = \sqrt{(p_1 - q_2)^T M (p_1 - q_2)} \tag{1}$$

$$= \sqrt{\sum_{i=1}^{n} q_i - p_i} \tag{2}$$

where the matrix M is the identity matrix. This distance function may generally suffer from the curse of dimensionality [27], in that the high dimensionality of data causes distances to increasingly become uniform [28]. However, by using football trajectory data in this work, this problem of the Euclidean distance may not be as severe according to Sha et al. [1]. This may be due to the typical sampling rate of 25Hz of the measurement equipment, that generates a dimensionality that may not cause large negative effects of this type. Our work studies these effects deeper [P1]. For other application domains, we refer to Wang et al. [24]. Still, computing a pairwise distance for larger trajectory DBs is prohibitively expensive due to the quadratic growth with the number of samples. While only calculating distances between a query and millions of other elements may still be reasonable, the necessary transport step (e.g., Hungarian Algorithm or OT) incurs additional costs before each pairwise comparison [P1].

2.2 Deep Metric Learning

Learning lower dimensional representations from raw data accelerates the search immensely. Siamese Networks [29] can be used to learn semantic similarity in NLP or for face verification [30]. Alternatively, a novel metric can be directly learned from human annotations using a triplet network [31]. These approaches all aim to learn an embedding or target space, that is of a smaller dimension than the original sample space, but that preserves some notion of similarity. Ideally, the learned similarity is equal to a human notion of similarity and semantically meaningful.

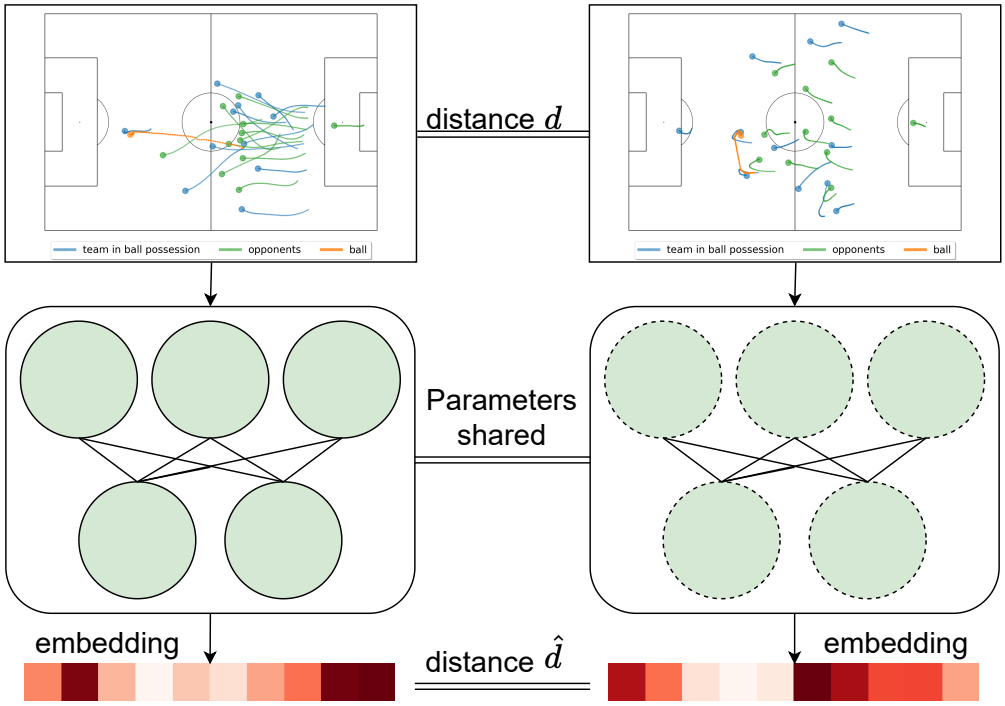


Figure 2: Siamese Networks learn a distance preserving embedding for pairs of samples so that the difference between distance d and the distance $\hat{d}$ between the samples in the embedding is small.

2.2.1 Siamese Network

Fig. 2 visualizes a Siamese Network similar to one that Chopra et al. describe in [30]. This example uses a Convolutional Neural Network (CNN) f_θ to process exemplary data from a football dataset that consists of images for simplicity. The distance d is defined between two images p and q, and may be the l_2 distance (see Eq. 2). The distance $\hat{d}$ can be another distance, and it is calculated between the two embedding vectors. The exemplary embedding in Fig. 2 is of length 10 and we visualize the encoding of a sample as shades of color. Siamese Networks f_θ use the loss

$$\mathcal{L}(p,q) = (||f_\theta(p) - f_\theta(q)||_2 - d(p,q))^2. \tag{3}$$

Training minimizes the difference of Euclidean distance $\hat{d}$ in the learned embedding f_θ and the raw data, hence, it learns a distance d preserving representation. Due to the use of the Euclidean distance in the Siamese loss, it is sensitive to noise. Furthermore, large distances have a large impact on the network's gradients during training. Due to this, the embedding captures coarse, large similarity structures better than fine, small neighborhoods, as these produce only small gradient signals. Hence, this approach is well suited to a first restriction of the search space [P1].

2.2.2 Triplet Network

Triplets [32, 33] of an anchor p, a positive sample p^+ and a negative sample p^- constitute an alternative learning paradigm besides the pairwise comparisons that is used in Siamese Networks. Annotated triplets represent relative similarity to the anchor sample p, and thus are useful for weak supervision that is class-independent and does not require clear labels. This can be written as the condition that the relative similarity $r_{ij}(p_i, p_j)$ of two samples p_i and p_j, that a human oracle annotates, fulfills the inequality $r(p,p^+) > r(p,p^-)$. From these, it is possible to learn a similarity function S such that $S(p,p^+) > S(p,p^-)$.

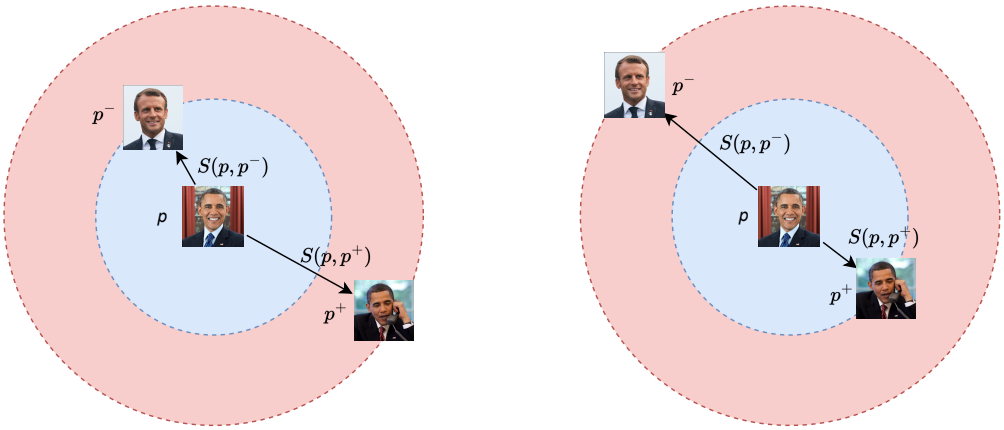


Figure 3: Triplets consist of an anchor (or query) sample, a positive sample that is similar to the anchor, and a negative sample that is dissimilar. Learning with triplets minimizes the distance between the anchor and positive sample $S(p,p^+)$ and maximizes the distance between the anchor and negative sample $S(p,p^-)$.

Wang et al. [34] initially proposed training a deep model for learning image similarity metric S from triplets, and Hoffer et al. [31] subsequently show that the *triplet network*, is

applicable to a wider variety of domains, challenging the Siamese network. Given a set of triplets and a deep model f_θ such as a CNN, the triplet loss is defined as follows

$$\mathcal{L}(p, p^+, p^-) = \max(||f(p) - f(p^+)||_2 - ||f(p) - f(p^-)||_2 + 1, 0). \quad (4)$$

The loss maximizes pairwise distances between the triple's elements such that the model learns an embedding in that the distance between the anchor and the positive sample is minimized, and the distance between the anchor and the negative sample is maximized, see Fig. 3 for a visualization.

2.3 Active Learning

Active Learning (AL) is a set of ML methods that selects and annotates the most informative samples for learning [2] and is based on the assumption that there is a subset of samples, that obtains a similar performance than training on the whole dataset. Furthermore, it assumes that there is a sample selection method that finds this subset faster than sampling at random would. The methods choose the training data of models based on heuristics a-priori and use insights on the learning task or model.

The main motivation for applying AL is to save on the cost and time of generating annotations. Cost may be driven by the complexity of the annotation task itself, and can be further complicated if annotators require a certain skill such as specialized medical expertise [35]. Time spent annotating typically scales the financial effort invested into the dataset creation which can be a bottleneck in the progress of ML projects. AL aims to overcome this issue.

At the center of AL are four components [2]. First, a human-in-the-loop, who annotates data, second, a (large) pool of unlabeled data to select samples from, third, a (smaller) pool of annotated data, and finally a ML model that learns the task, see Fig. 4. In the AL process, the annotator (i.e., the domain expert or "oracle") creates an initial set of labeled data that is selected at random. Based on this set, the loop starts and a first model is trained on the labeled data. Then, given an acquisition function that may use information about the data and the model, the function selects the next sample(s) from the unlabeled pool. The acquisition function is at the center of any AL strategy and determines whether the model can learn faster from selected data compared to random sampling. Finally, the oracle is queried for annotations and once annotated, the now labeled samples are moved from the unlabeled pool to the labeled pool, on which the model is trained. This loop continues until a budget is exhausted, a prediction quality is achieved or any other stopping condition is triggered [2].

Literature distinguishes pool-based and stream-based AL [2] scenarios. The pool-based scenario is centered around a database of unlabeled samples and the acquisition function may select queries from it. The stream-based scenario is instead centered around a (continuous) stream of unlabeled samples that is fed into the AL loop incrementally. Apart from these two categories the size of a query further subdivides AL algorithms into different families. Queries with only one sample may select only the most informative sample, but are costly w.r.t. model training, because models are trained more often, i.e., after each acquired sample, instead of only after a larger set of samples. Thus, larger queries (i.e., batch-mode AL) are more compute and time efficient as models may fit to faster increasing dataset sizes, but the batches may introduce sampling bias, e.g., by oversampling one class

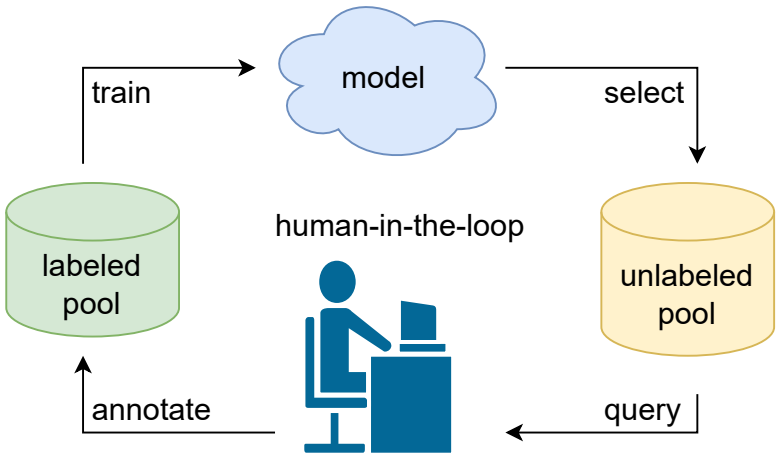


Figure 4: The human-in-the-loop Active Learning procedure consists of four components. An "oracle" that annotates unlabeled samples, a Machine Learning model that learns a task, the labeled and unlabeled pools of data, and an acquisition function to select previously unlabeled samples to query the oracle.

that is least-certain [9]. Still, this mode is considered important when using DL, because retraining is especially costly and an incremented dataset size of only one sample tends to show unstable performance differences per loop iteration [36].

Acquisition functions are at the center of the AL loop. Because they select the samples to query the oracle with, the functions is is responsible for the effectiveness of the approach. An exemplary acquisition function for classification tasks is selecting the unlabeled samples for which the model's prediction is the least confident [37]. Here, the predicted likelihood for an unlabeled sample x of a set $\mathcal{X}$ of a model is $\hat{x} = \text{argmax}_{x \in \mathcal{X}}(1 - P(\hat{y}|x))$. For each sample x we choose the prediction $\hat{y} = f_\theta(x)$ with the highest likelihood. This way the acquisition function constructs queries with samples the model f_θ is least confident about.

Besides, a long line of research has resulted in a large body of work on different acquisition functions for many ML models [2]. The concepts of these are primarily based on uncertainty-sampling [37, 38], querying-by-committee [35, 39], expected error reduction or model change [36, 40], sample density [41], or reduction of variance, i.e., maximizing information gain [42]. With the wide adoption of DL the research on this topic has experienced new attention to adopt the basic principles to deep network architectures. However, many ideas are similar, hence the following sections explain uncertainty-based AL (see Sec. 2.3.2), density-based AL (see Sec. 2.3.3) and hybrid approaches (see Sec. 2.3.4). On this basis, Sec. 3.2 presents the current state-of-the-art of the field.

2.3.1 Deep Active Learning

Large amounts of training data are required for Deep Learning models in order to generalize well to new, unseen data [3]. This is problematic for training strategies that rely on annotated data, such as supervised learning. While employing AL to reduce the associated costs of creating these large corpora of annotated samples, DAL has to take the specifics of DNNs into account. Traditional AL methods [2] used in conjunction with query sample sizes of only one are inadequate for use with DNNs [41]. This is due to two main reasons [43], first, the training of common DL models is costly and increases annotation cost by introducing

waiting times, potentially by days. Second, increasing the labeled dataset by only one sample may not even have any significant impact on the model.

This section provides an overview over the fundamental methods of AL with special attention to DNNs. Methods for DAL [3] were developed along similar ideas as traditional AL, but we tailored towards DNNs. These ideas are outlined in the following sections, while Sec. 3.2 reviews the state-of-the-art literature.

2.3.2 Uncertainty Sampling

Uncertainty sampling [37] uses the model's own doubts about its knowledge to improve it. Intuitively, samples that the model is unsure about should provide more information about the dataset than samples that are already modeled with very low variance and high precision. This idea traditionally is feasible for probabilistic models [2]. However, the predictive uncertainty of DNNs may not be available for any task. Regression models typically predict one or more continuous variables (e.g., they may predict the yield in chemical processes from data on temperature and pressure), whereas classification may provide a distribution over class likelihoods as part of their softmax output [44]. Furthermore, the softmax uncertainty measure may be insufficiently calibrated [35], i.e., training with softmax can result in overconfident DNNs [45], which leads to sampling bias in AL[35]. Hence, it is crucial to obtain a better calibrated uncertainty measure to select the most informative samples.

We distinguish uncertainty in a reducible and an irreducible part. The irreducible part of uncertainty is referred to as aleatoric and is the inherently random effect of a process, e..g, a coin flip has a stochastic component that cannot be reduced further and models will predict two equally likely outcomes [46]. The reducible uncertainty is also known as the epistemic uncertainty and is caused by the lack of knowledge [46]. Hence, the aim of AL is to reduce this epistemic uncertainty and not the inherent aleatoric uncertainty that may be present in the data itself. For uncertainty-based AL, we aim to efficiently reduce the "ignorance" [46] of a model by sampling informative samples.

To address the need for an improved measure of epistemic uncertainty, Gal et al. [38] proposed a method to estimate a CNN's uncertainty via casting it as a Bayesian Convolutional Neural Network. This allows for the predictive uncertainty to be decomposed into the aleatoric and epistemic components. Following the discussions of Valdenegro-Toro and Saromo Mori [47], the mean of the variance of a prediction corresponds to the aleatoric component and the variance of the mean corresponds to the epistemic component. An important difference to traditional Bayesian Neural Networks is that Gal et al. proposed a computationally efficient trick that is based on Monte-Carlo simulation and follows an ensemble- or vote-agreement scheme [48]. Their scheme randomly drops connections of the DNN and performs multiple inferences for one sample. The resulting distribution of the inferences' result may be interpreted as if it originates from separate models in an ensemble. Another refinement of uncertainty measures for DNNs was proposed by Beluch et al. [35], who use the "power of ensembles" for AL. The authors show that an uncertainty measure derived from an ensemble is better calibrated compared to using a single model. While their observation was validated only for few data scenarios, an active learner that uses an ensemble may also perform better w.r.t. accuracy in more complex data domains than using a single model softmax for measuring predictive uncertainty.

Given such reliable measures of uncertainty [35, 38], we can employ different acquisition functions that select the most appropriate samples for querying. The Max Entropy criterion is directly based on Shannon's information theory [49] and maximizes the predictive entropy $H[y|x, D_{\text{train}}]$ for an ensemble of T elements [35] as

$$H[y|x, D_{\text{train}}] = -\sum_c \left(\frac{1}{T} \sum_t P(y = c|x, \theta_t) \right) \tag{5}$$

$$\log \left(\frac{1}{T} \sum_t P(y = c|x, \theta_t) \right) \tag{6}$$

where c is the class, y is the prediction and θ_t are the weights of the forward pass t. To employ this for ensembles [35] or Monte-Carlo simulations [38], we can simply sum the probabilities $p(\cdot|\cdot)$ up and average them over the number of passes T. Alternatively, the Variation Ratio acquisition function selects samples with the most dispersed probability [38] (see Eq. 7) or similarly, that have the highest disagreement in the ensemble [35] (see. Eq. 8)

$$\text{variation-ratio}(x) = 1 - max_y(P(y|x, \theta_t) \tag{7}$$

$$= 1 - \frac{m}{T}, \tag{8}$$

where m is the number of predictions falling into the class c category over the number of forward passes T [35].

2.3.3 Diversity Sampling

Fitting a model to a training dataset that is representative for the whole dataset is a popular rational in AL. Such a subset may also be more diverse than sampling an equal amount of least certain samples, and thus not overfit to only a small part of a task, such as the dataset's outliers, as may happen when focusing only on these least certain samples. Furthermore, as DAL is computationally more efficient when trained with batches of data instead of single samples, the composition of a batch may benefit from an acquisition of more diverse samples as well, because it avoids biased training. Random sampling can be viewed as a simple version of a diverse sampling method. However, it may also overfit, e.g., if the dataset is imbalanced. Sener and Sevarese [41] proposed to select the core-set of a dataset, that covers the dataset optimally, is unbiased, and works with larger dataset sizes than uncertainty-based sampling, as it is less affected by outliers and supports less biased, more representative sample batches.

The approach is based on previous work in optimization, where core-sets were used for, e.g., k-center clustering [50], or even for AL with Support Vector Machines [51]. The crucial difference is to build the core-set not in the data domain, but in the learned manifold (i.e., the DNN's embedding) of the pool data [41]. The methods developed in previous work are then used to solve the k-center task and to construct the query.

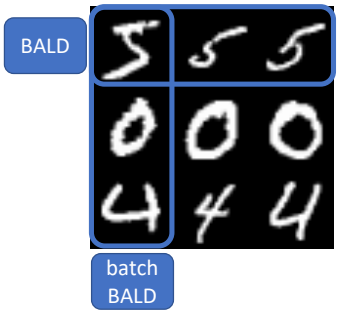


Figure 5: The idealized acquisition of BatchBALD [9] selects a more diverse query compared to BALD [42], that selects the most informative samples, even if they repeat.

2.3.4 Balanced Criteria

Balancing the selection of representative samples on the one hand with the selection of the most informative ones on the other is especially relevant for DAL due to its benefits when used in combination with the typical mini-batch training of DNNs. The balanced construction of batches is the objective of many recent strategies, among others the work by Kirsch et al. [9] that is based on previous ideas of Houlsby et al. [42].

The seminal work by Houlsby et al. [42] utilizes the mutual information of the DNN's parameters and its predictions to determine whether the annotation of a sample would provide new information to the model. In their method, that is known as Bayesian Active Learning by Disagreement (BALD), the mutual information of the predicted label y and the parameters θ is calculated as follows [35]:

$$I(y;\theta|x,D_{\text{train}}) = H(y|x,D_{\text{train}}) \tag{9}$$

$$- E_{P(\theta|D_{\text{train}})}[H(y|x,\theta,D_{\text{train}})] \tag{10}$$

Here, the first term measures the entropy over predictions, that is high for uncertain predictions, and the second term measures the expectation of the prediction, given the model and its parameters θ, that is low for certain predictions. Next, maximizing I helps finding samples for which the predictions are uncertain, but at the same time, a low uncertainty of the DNN's parameters.

BALD selects only one sample at a time and does not measure any redundancy of information, if it is used to select additional samples, and as such performs worse with larger batch sizes. Kirsch et al. [9] argue that it overestimates the joint mutual information of a batch, as it would count the information of each pair (y,x) separately even if they are similar and their information overlaps. Hence, Kirsch et al. extend it such that the maximization optimizes the whole batch B of samples $x_1,...,x_b$ and estimates the joint of the samples. Notice that Eq. 10 and 11 are the same for batch size 1.

$$\begin{aligned} I(y_1,...,y_b;\theta|x_1,...,x_b,D_{\text{train}}) = &H(y|x_{1:b},D_{\text{train}}) \\ &- E_{p(\theta|D_{\text{train}})}[H(y|x_{1:b},\theta,D_{\text{train}})]. \end{aligned} \tag{11}$$

Here, Kirsch et al. elaborate that I measures the mutual information between a set of samples ad takes the intersection of the information content of the variables 1, ..., b into account, and optimizes directly for B's diversity. Fig. 5 visualizes the key difference between BALD, that strictly optimizes information sample-wise, e.g., by selecting only the most informative samples even if they are from the same class and may have overlapping information content, and BatchBALD, that instead samples more diversely, e.g., from different classes, and avoids probably redundant samples. This highlights that the Entropy H of a set of samples $x_{1:b}$ lead to less repetition in a batch and more sample efficient batch-mode AL.

2.4 Active Metric Learning

This section summarizes the fundamental concepts of Active Metric Learning. First, it defines the terms and presents the motivation, Sec. 2.4.1. Next, Sec 2.4.2 explains essential algorithms such as InfoGain and Multidimensional Scaling (MDS). Finally, Sec. 2.4.3 concludes with a summary of good user study design.

2.4.1 Semantic Gap

Active metric learning aims to learn a metric, or a similarity matrix, for all pairs in a dataset [4]. This is different from classification tasks, where inter-class similarities and intra-class variance are typically ignored [6].

Employing human annotators to learn similarity in human-in-the-loop systems has one important reason, that concerns the "similarity function" [4]: there exists a semantic gap between humans' innate understanding of similarity and the state of the art feature detectors. For a wide range of topics, humans "know" similarity whereas implementing similarity metrics is difficult. For example, Tamuz et al. [4] cite the question of whether a joke is funny or not as a task that is best learned from human annotators. The New Yorker uses AL implemented by Tamuz et al. in the NEXT system [52, 53] to rank comics by their humorous quality[1].

Perceptual metric learning has the goal to learn human-perceived inter-object similarity [6], i.e., a "continuous" measure or a "degree" of similarity. Due to the combinatory complexity of pairwise comparisons, such learning is most often performed using some form of AL to select only useful queries [4], with more efficient batched queries [6] or with tuples larger than two [5] (see Sec. 3.3.2).

2.4.2 Essential Algorithms

Multidimensional Scaling (MDS)[2] can be understood as a dimensionality reduction method that respects distances in the higher-dimensional space, and that is typically constructed from a matrix with pairwise distances of the given dataset [54]. Interestingly, it stems from the field of psychophysics and processing sensory information, and was initially also used with non-metric similarities [54] (that do not adhere to the triangle equation).

Following Mead [54], we define a dataset with n elements, and a triangular matrix ($n \times n$) that contains pairwise dissimilarities ("distances" d), where row i and column j contains the

[1] Participate here: https://www.newyorker.com/cartoons/vote
[2] Implemented as part of scikit-learn: https://scikit-learn.org/stable/modules/manifold.html

distance $d_{i,j}$ between samples i and j. These distances are a metric, if the triangle equation is satisfied: $d_{i,k} + d_{k,j} \geq d_{i,j}$. Otherwise, they are of an ordinal, rank ordered, non-metric nature, that may match dissimilarities (observed by humans) more closely.

Metric MDS minimizes the difference between distance $d_{i,j}$ in the higher-dimensional space and $\hat{d}_{i,j}$ in the lower-dimensional space as

$$\sum_{i=2}^{n} \sum_{j=1}^{i-1} [d_{i,j}^2, -\hat{d}_{i,j}^2]. \tag{12}$$

The metric variant of MDS requires three steps as outlined by Mead [54]: i) obtain pairwise comparisons, ii) convert relative comparisons into absolute distances, and iii) determine the required (but lower) dimensionality to represent this. The optimization finds $\hat{d}_{i,j}$ that are similar to $d_{i,j}$.

The non-metric variant of MDS deals with dissimilarities that are ranked [55], and for that the distance between samples is not known or not relevant. The optimization seeks $f(d_{i,j})$ that is similar to $d_{i,j}$, but $f(d_{i,j})$ preserves only the order between samples i, j, k, l as

$$d_{i,j} < d_{k,l} \Leftrightarrow f(d_{i,j}) \leq f(d_{k,l}) \tag{13}$$

$$\Leftrightarrow d_{i,j}^* \leq d_{k,l}^* \tag{14}$$

and the distances $f(d_{i,j}) = d_{i,j}^*$ are called disparities (i.e., preserve only relative order, not distance). Approaches such as Kruskal's non-metric MDS minimize the so-called stress between pairs of $\hat{d}$ and d^* [55].

In the probabilistic case of the MDS [56, 57, 58], each point is represented by a vector, whose components are characterized by independent normal distributions. The probabilistic modeling of the dissimilarity between samples in the embedding can be used to inform entropy-based AL methods, e.g, Tamuz et al. [4], see Sec. 3.3.2.

The state of the art in active metric learning [4, 5] formulates the information contained in a query based on fundamental information theory. As described in Sec. 2.3.4, Houlsby et al. [42] propose the informativeness of a sample in Eq. 10 as the subtraction of the uncertainty of the oracle to label this sample from the uncertainty of a predicted label for a sample (given the embedding). Optimizing the result of this subtraction avoids redundant queries (for that a model is certain) and uncertain responses (when the oracle is uncertain). This concept was adapted to metric learning [4, 59]. This way, methods can build an annotated set of highly informative relative comparisons from few annotations. Hence, this section introduces the basic concepts of information theory.

Both Goodfellow et al. [60] and Cover and Thomas [61] introduce Information Theory as quantifying the information of a signal, e.g., for finding the limit of data compression or transmission. Intuitively, an unlikely signal (or sample for Machine Learning), is more informative than a repeated one. Similarly, independent samples provide additional information.

The self-information $I(\cdot)$ of sample x for binary systems is measured in bits required to encode the signal

$$I(x) = -\log_2 P(x) \tag{15}$$

In Machine Learning, measuring information of probability distribution uses Shannon Entropy $H(\cdot)$ as

$$H(x) = \mathbb{E}_{x\sim P}[I(x)] \tag{16}$$
$$= -\mathbb{E}_{x\sim P}[\log P(x)] \tag{17}$$

where P is the probability mass function. Here, we calculate the number of bits required to encode the whole distribution P over all the samples X that we draw from it [60]. This measures the expected information $\mathbb{E}_{x\sim P}[I(x)]$, given the distribution P of sample x. The encoding $I(x)$ can be small if the distribution P of random variable X is only a few values, but larger if its more uncertain. Then, it has a higher entropy $H(x)$ and contains more information. We can write this also as $H(X) = -\sum_{x\in X} P(x)\log P(x)$ [61]. The visualization in Fig. 6 is inspired by Cover and Thomas [61] and shows the deconstruction of joint entropy $H(Y|X)$ into its components of entropy and mutual information as rectangles.

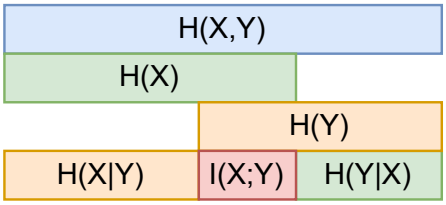


Figure 6: The joint entropy $H(X,Y)$ measure two random variables X and Y that describe the total entropy. Then, $H(X)$ and $H(Y)$ describe the entropy of each variable. $H(Y|X)$ is the information about Y that cannot be explained by X, and $H(X|Y)$ vice versa. Finally, $I(X;Y)$ is the mutual information of both X and Y.

The conditional entropy $H(Y|X)$ measures how much of the random variable Y cannot already be explained by X [61]

$$H(Y|X) = \sum_{x\in X} P(x)H(Y|X = x) \tag{18}$$

For one sample x, we can write this equation as follows

$$H(Y|X = x) = -\sum_{y} P(y|x)\log P(y|x) \tag{19}$$

The mutual information $I(X;Y)$ is the "reduction of uncertainty" [61] that Active Learning also seeks. AL aims to select those unlabeled samples that are most informative. Methods estimate the informativeness based on a set of labeled and unlabeled samples from the same distribution. $I(X;Y)$ is the information that observing one random variable (from the unlabeled pool) provides in addition to another random variable (the labeled training data). Information theory defines this as the information gain, i.e., the mutual information, between X and Y.

$$I(X;Y) = H(X) - H(X|Y) \tag{20}$$
$$= \sum_{x\in X}\sum_{y\in Y} P(x,y)\log\frac{P(x,y)}{P(x)P(y)} \tag{21}$$

Cover and Thomas [61] explain that $I(X;Y)$ measures the reduction of uncertainty about X that observing Y brings. Methods in AL often maximize the information that annotating unlabeled samples from the pool dataset Y gains for the labeled dataset X used in model training.

The definition of the joint entropy [61] $H(X,Y)$ of two random variables X and Y follows from the entropy $H(X)$ as

$$H(X,Y) = -\sum_{x\in X}\sum_{x\in Y} P(x,y)\log P(x,y). \tag{22}$$

2.4.3 User Study Design

The recent book by Robert (Munro) Monarch [62] on human-in-the-loop machine learning is an exhaustive resource on the design principles and practical considerations for conducting Active Learning studies with human participants. The considerations that we deem most relevant to this work are on the quality control of annotators, the aggregation of annotations from different annotators, transfer learning and user interface choices.

2.4.3.1 Intra-/Inter-Annotator Agreement

When a group of human annotators create labeled ground-truth "golden" datasets, they can make mistakes. Especially for complex tasks such as speech recognition [62] or translation [63], performance on the same level or above a single human (expert) annotator can be achieved by ML methods. For user studies, this means that the collection of data requires a quality metric for annotators, such as the inter- or intra-annotator agreement [62]. Conceptually, inter-annotator agreement requires that a set of queries is repeated for each participant and an agreement score is derived from dis-/similar answers. For intra-annotator agreement the same set of queries is repeated to reach a score of the annotators consistency. From these measures, its possible to learn, for example, whether the labels are trustworthy, whether annotators need to be better instructed in the annotation task or even filtered out [62].

2.4.3.2 Aggregating Annotations

Once a set of annotations is collected from an ensemble of annotators, the labels could be directly combined into one large training dataset. However, there may be mistakes, e.g., a disagreement among the labels or annotators may report additional confidences, that requires additional steps before deciding what label to trust [62], e.g., calculating the confidence or the entropy of the annotations. In practice, Monarch lists three options if we cannot decide onto a label based on the agreements the annotators have achieved:

- annotate a problematic sample once more
- assign an expert annotator to label the sample
- exclude the sample

2.4.3.3 Transfer Learning

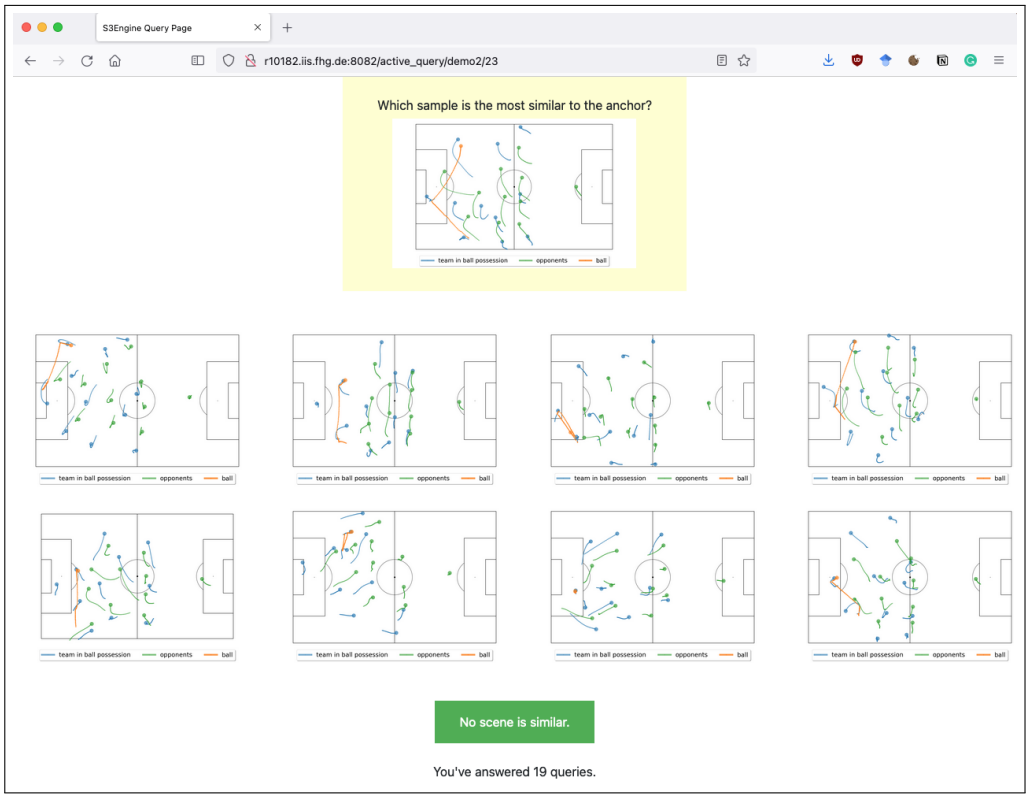
User studies may utilize transfer learning from one dataset to another task to gain an initial boost with labeling, or to implement AL strategies [62]. However, in the end pre-trained

weights may only provide extracted features or learned representations but the learning task still requires new annotations [62]. Especially recent advances in NLP, with its large language models [64, 65], and CV to a lesser extend [62], promote the use of pre-trained models in AL studies. A downside with such training is that pre-training on a source dataset may transfer biases to the novel dataset as well and has to be considered [62].

2.4.3.4 User Interfaces

The three basic design principles for user interfaces for collecting annotations [62] are the "affordance" of the design (e.g., an object behaves like we expect), the "feedback" provided to a user's action (e.g., click registered in the system), and the perceived "agency" of the annotators. Both the affordance and the feedback together make a user interface intuitive, and agency should give participants a sense of "power" or "ownership", e.g., an AL algorithm should improve with the number of annotated samples, or the interface lets them submit all relevant information.

Concretely, the web-based active metric learning platform NEXT [52, 53] provides an intuitive and minimalist design language for annotating similarity queries. Similarly to our own design in Fig. 7a, NEXT shows the query's anchor centered at the top and the choices in rows at the bottom. We add highlighting of the anchor, as well as a large and obvious radio button. The number of choices is variable. Finally, our web-based annotation tool provides direct feedback after an action in the form of an animated spinner (see Fig. 7b). Since AL methods may run for several seconds, such direct feedback is even more important [62], since it helps avoid annotator fatiguing for longer.



(a) Multiple choice interface.

(b) Spinner

Figure 7: a) The user interface is a very simply multiple choice interface with attention to ”affordance” (point-and-click). b) A spinner animation provides direct feedback after an annotation action.

3 State of the art

This part consists of three parts. First, Sec. 3.1 presents the recent work on Information Retrieval with trajectory data, specifically from the sports domain. Next, Sec. 3.2 reviews the trends and developments in (Deep) Active Learning. Finally, Sec. 3.3 concludes with a discussion of the state of the art in Active Learning of Similarity Metrics.

3.1 Information Retrieval

This Section presents the different components and research streams, that currently compose Information Retrieval systems in the domain of trajectory data mining, and those that are highly relevant for its future development. Sec. 3.1.1 shows the state of the art of frameworks for Sports Scene Retrieval. Next, Sec. 3.1.2 explains metrics that measure similarity of trajectories. Then, Sec. 3.1.3 presents the recent work on learning similarity for trajectory data and presents permutation invariant networks. Finally, Sec. 3.1.4 concludes with the current limitations.

3.1.1 Sports Scene Retrieval Systems

The application of finding similar samples in a large dataset in sports is complex, and several technically different approaches were proposed to solve it. These Sports Scene Retrieval systems solve two specific problems with various techniques and varying success: first, assignments of trajectories between pairs of trajectory sets, and second, a data representation that is both accurate but also enables dataset queries at interactive speeds. Each of these sets is a temporal slice of trajectories of fixed length that is called a "scene", hence the name "Sports Scene Retrieval".

The following methods represent the state of the art and show how this thesis extends it. Sports Scene Retrieval systems exist in three variants. The first one constructs data structures, such as hierarchical clusters based on formation templates to limit the search space, and may extend them with Machine Learning [1, 7, 8]. The second variant uses deep representation learning to replace the efficient data structures [66]. The most recent work uses RL to search similar scenes and combines it with deep metric learning [67].

Sha et al. [1] presented the query paradigm for basketball called Chalkboarding in 2016, and seeded a series of follow-up work on Sports Play retrieval [P1, P3, 7, 8, 66, 67]. Chalkboarding allows users to sketch a query scene and the system then retrieves a similar play from a database. To do so, it estimates a solution to the assignment problem based on roles and formation templates (per player, per match, and per team), that it learns from data similarly to the expectation maximization approach by Bialkowski et al. [20]. The authors solve the combinatorial complexity using hierarchical and semantic clustering of scenes in the database based on distances of ball trajectories. Both accelerate retrieval considerably but introduce an unquantified error. Furthermore, generalizing this approach to other domains such as soccer is problematic, because the role assignments are not static within matches or teams. Finally, this thesis evaluates the retrieval speeds and finds that the computational complexity of large datasets is a limitation [P1]. The authors followed up [7] with an improved solution to the assignment problem, that uses hierarchical templates, that

they construct from all data. However, the computational demands determine the hierarchy depth of the templates instead of semantics. Due to this limitation, for each leaf, they use a sub-optimal assignment to compute distance on the raw, high-dimensional trajectory data. Di et al. [8] address this high dimensionality by learning a ranking function from users. This partially extends the Chalkboarding method, but still relies on the clustering of the database.

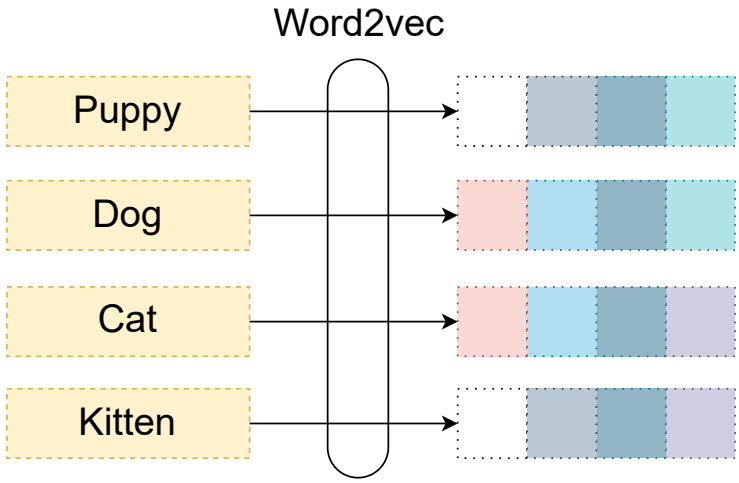


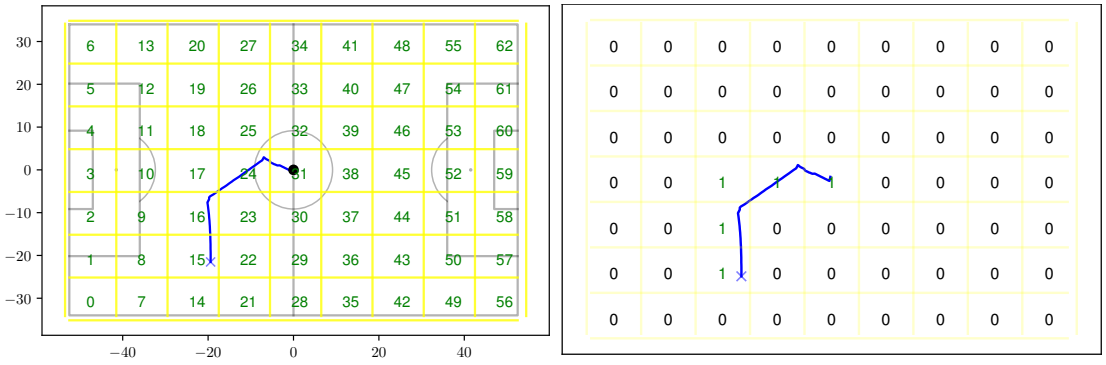
Figure 8: The embedding of words of word2vec [68, 69], or trajectories in the case of play2vec [66], learns vector representations, that have a low (cosine) distance, if they are conceptually more similar. Here, "puppy" is similar to "dog", and "kitten" is similar to "cat". Wang et al. apply this embedding conceptually analogously to segmented trajectories.

Wang et al. [66] follow an alternative approach and propose play2vec. They first segment games into semantic parts (ball possession) and then treat the segments analogously to sentences in NLP: they propose to learn relations and similarities similarly to word2vec [68, 69] for languages. Fig. 8 shows an example of embedded words, that Wang et al. analogously applies to trajectories. Mikolov et al. [68, 69] explain, that performing a simple arithmetic offset operation, for example vector(puppy) − vector(dog) + vector(cat) as in Fig. 8, results in a vector representation of the equivalent vector(kitten).

Wang et al. [66] observe that trajectories are made up of repeating segments. Such segments from the same contexts also appear in similar sports scenes. To create segments, the authors first map trajectories to a spatial grid of $5x7$ elements to avoid assignment issues, see Fig. 9a for a larger example of our own data, and then segment the trajectories into tokens, analogously to words in NLP. In a sequence of segments, the contexts before and after a segment are predictive for it, and thus Skip-Gram with Negative Sampling (SGNS) [68, 69] can be applied in training. Fig. 9b shows a segment matrix generated from a simple play segment for our football data.

Learning the play2vec embedding has its benefits, as their user-study shows: its retrieval quality appears superior to Chalkboarding. However, it also has some limitations. It does not allow weighing trajectories with different importance during model training, and the learned embedding preserves only a relative similarity.

The most recent work of Wang et al. [67] builds onto their play2vec method for learning similarity, and adds intra- and inter-game search components to their framework. For



(a) A grid overlaid onto the playing field.

(b) The segment matrix for the playing field.

Figure 9: (a) shows one example trajectory for a scene and a large grid overlaid. (b) shows the accompanying segment matrix.

intra-game retrieval, they first present two simple algorithms to split the game's search space and only compare select samples with a query scene: first, the exact and incremental ExactS, which enumerates all possible scenes, and second, the approximate SizeS, that only considers scenes with similar length. Their contribution, however, is the use of Reinforcement Learning to learn a strategy that splits the search space until it finds the most similar sample, and thus, is most efficient compared to their baselines. For inter-game search, the authors propose a novel idea for selecting what games to search in. They do not create one global embedding of all scenes, as our work enables [P1]. Instead, they use deep metric learning to learn a ranking of games, that are ordered similarly to a query scene, in order to search only in those similar games. They train a triplet network on randomly sampled triplets (see Sec. 2.2.2), where the anchor sample is the shortest and the positive sample has the highest similarity in the play2vec representation. The triplet network then is trained on play2vec representations with the triplet as a target as in Eq. 4.

3.1.2 Trajectory Similarity Metrics

Similarity metrics are an essential component of Information Retrieval systems. The similarity between trajectories can be measured using distance functions. Wang et al. [24] provide a categorization for this type of data: metrics can capture the true ordering of positions (trajectory-trajectory) or not (point-trajectory) and agree with the differentiation of Tao et al. [70] on temporal, spatial and spatiotemporal data. In tracking team-based sports, trajectories can typically be expected to be temporally aligned, and spatially registered in a fixed semantic frame (playing field). Sha et al. [1] showed in an qualitative study on sports trajectory data, that the differences between popular distance metrics tend to be negligible, e.g., Euclidean distance, Dynamic Time Warping, Fréchet distance, or l_∞ distance do not differ much. However, the current state of the art may improve upon this and is discussed here.

The three papers [1, 24, 70] discuss and evaluate, among others, the following popular metrics, that serve as baselines. Discrete Fréchet Distance (DFD) [71] is a curve-based distance, that takes the ordering of points into account. Dynamic Time Warping (DTW) [72] measures distances by warping the sequences. Longest Common Subsequence (LCSS) [73] computes an edit distance by matching sub-sequences that are similar. However, these order-preserving distances are no true metrics, as they do not obey the triangle inequality,

which can become a limitation. The triangle inequality for three points x, y and z requires that $d(x,z) \leq d(x,y) + d(y,z)$. If this fails, then distances do not have to be meaningfully comparable and transitive, losing these benefits of metric spaces. For example, DTW warps the sequence to align features, which distorts the original points of trajectory pairs [74], and thus we cannot use it to construct a metric space and it does not lend itself optimally for indexing, clustering [70] or to learn an embedding.

Temporal-aware similarity metrics also respect temporal information but require the sample rate to be well-calibrated. The benefit of this type of metric, e.g., by Frentzos et al. [75], is to integrate the time w.r.t. Euclidean distance, and thus handles spatiotemporal information. This approach is useful for comparing the temporal dimension of trajectories, like in comparing different modes of transport like buses and cars, and finding temporally aligned trajectories, e.g., to adapt time tables. In sports scenes, we can align the temporal dimension and are provided with a well-calibrated sampling rate.

More complex, longer trajectories may be converted into segments, which also reduces issues with sampling rate calibration [24]. Following this idea, Chen et al. [74] recently proposed a deep representation learning-based method, that computes embeddings for trajectories and preserves the similarity between them to accelerate similarity computation. The authors propose to first divide trajectories into segments and encode these sequences using a memory-augmented Recurrent Neural Network (RNN) model, that captures the sequential order information that is relevant for trajectory data [24]. In a second step, the attention-based learning to rank method learns similarity for the training set, based on the learned representation in the final state of the RNN model and Euclidean distance.

3.1.3 Metric Learning for Sets

As discussed in Sec. 3.1.1, one key component of information retrieval systems is the similarity metric between samples. Finding a low-dimensional representation of high-dimensional data can accelerate the necessary pairwise comparisons. Here, the goal is to represent complex data in a meaningful vector space [76] like an embedding. An additional difficulty for Metric Learning with trajectory data is the permutation invariant nature of a set [P1], i.e., unordered sets of trajectories, that may also be found in similar data such as pointsets [76].

3.1.3.1 Permutation Invariant Networks

Zaheer et al. [77] proposed DeepSet initially in 2017 as a deep model, that can process permutation invariant data. Arsomngern et al. [76] later integrated the DeepSet model into their Deep Metric Learner to leverage its permutation invariance and thus avoid typical networks' issues with order-sensitivity with set data, as explained in Sec. 3.1.3.2. DeepSets work essentially by processing each set member individually into an individual representation, "[adding] up all representations and then [applying] nonlinear transformations" [77], thus using its commutative property to learn permutation invariant representations.

The authors evaluate DeepSets on various applications, with text context retrieval being the most relevant that explains the idea well. The explanatory example task is the retrieval of the most similar words to the given query set (tiger, lion, cheetah). Each of these words is processed into a representation, and all of these are subsequently added up. Then the most similar words to the common concept of "big cats", that forms a cluster of learned representations, could be (jaguar, puma). Transferring this idea to trajectory data would

require a similar additive processing step, that considers pairwise distances for regressing similarity rather than class labels.

3.1.3.2 Metric Learning with Unordered Sets

This section covers the learning of similarity for samples that are composed of unordered data or datasets. The methods go beyond traditional supervised deep metric learning with labeled samples and a pairwise loss and investigate auxiliary losses or self-supervised learning from triplets [76, 78].

Zhang et al. [78] extend learning trajectory similarity with a triplet loss, which includes terms for auxiliary supervision and for optimal matching. They do not learn similarity for sets of trajectories, but instead focus on longer trajectories, that they split into sub-trajectories. Hence, their first extension is an auxiliary supervision loss, that learns similarity from sub-trajectory similarity instead of for temporally aligned trajectories as available in sports tracking. The distance between sub-trajectories can provide richer information, similar to what a single player's trajectory in sports may provide for the similarity between scenes. The authors encode data with RNN-based encoder, that maps into a similarity space in a residual network, and then randomly sample sub-trajectories as supervision signals to define the sub-trajectory loss. The distance for each pair of sub-trajectory L_{near}^{sim} is then added as a term in the triplet loss (see Sec. 2.2). The second element is learning an optimal matching of trajectory points from two different trajectories. This matching may be based on a very complicated distance metric, and learning to estimate this relationship reduces computational complexity. The authors use an optimal matching between pairs of trajectories as targets. An Long Short-Term Memory (LSTM) then learns a vector representation. The next step randomly samples triplets, such that the positive sample optimally matches the anchor, and the negative sample is mismatched. The learned embedding space then approximates a matching between trajectories. Their work may serve as an extension to Siamese Metric Learning, which learns an estimation of optimal assignments [P1], by using auxiliary losses for finer similarity of player's trajectories, and also explicitly learns an optimal matching.

Arsomngern et al. [76] propose a self-supervised Deep Metric Learning solution for generic pointsets in order to leverage unlabeled data, akin to pseudo labeling. To this end, they transfer ideas from CV or NLP to pointset data. However, the pointsets feature has an order invariance that is incompatible with typical networks' order-sensitive input, similar to the assignment problem in sports tracking. Hence, the authors use DeepSets [77] that are permutation invariant and combine it with features extracted by a self-attention mechanism from transformers [79]. The learning objective is also based on a triplet loss, but it differs to [78] in several aspects. Due to the unstructured format of point set data, the authors use the Earth Movers Distance, that is described in Sec. 2.1.1, to measure similarity between samples. These distances are then used to construct triplets like Zhang et al. [78]: samples with lower distances serve as positives and farther away as negative. In sports tracking a player's trajectory forms one closed semantic unit, hence EMD for whole sports scenes is less suitable as it breaks logical constraints. Arsomngern et al. propose another extension to triplet loss, that specifically considers the similarity of the anchor and negative samples. These are weighted higher in their proposed weighted self-supervised EMD triplet loss, as semi-hard negative sampling can enhance learning. For example, Schroff et al. [80] experimented with selecting hard negatives from a mini-batch and found it to be more stable and quicker to converge. The learned embedding preserves the more complex similarity metric as Euclidean distance between points in the embedding

3.1.4 Limitations of current Information Retrieval methods

Current approaches to information retrieval from datasets of unstructured trajectory are limited in several aspects. Even though tracking is ubiquitous today, IR is still largely based on manual annotations by experts. Other approaches may use computationally expensive data structures that do not scale well, estimate trajectory assignments only for subsets of the data and limit search space, or do not even support multi-agent trajectory data. Hence, indexing and IR remains an expensive, slow and imprecise operation. We address these limitations in the publication [P1] in Appendix A.

3.2 Deep Active Learning

This chapter focuses on the state of the art in Deep Active Learning and highlights methods, that learn to (actively) learn. First, data distributions can be learned with generative models and exploited for AL [81, 82, 83, 84], Sec. 3.2.1. Learning AL can be understood as an optimization problem and solved using Reinforcement Learning [10, 12, 13, 14, 85, 86, 87, 88] or Neural Architecture Search (NAS) [89], see Sec. 3.2.2, by imitating experts [90, 91], see Sec. 3.2.3, or by selecting the most suitable strategy [92, 93, 94], see Sec. 3.2.4. Meta Learning aims to solve a related task to Active Learning, namely to enables deep models to learn quickly with few samples [95, 96, 97, 98, 99, 100, 101, 102], see Sec. 3.2.5. Finally, Sec. 3.2.6 concludes with a discussion of limitations.

3.2.1 Generative Active Learning

Generative models can learn latent spaces of dataset distributions and synthesize new samples. AL can then exploit the latent space's nature to select diverse and representative samples either for an oracle to annotate or to add informative synthetic samples to the training dataset.

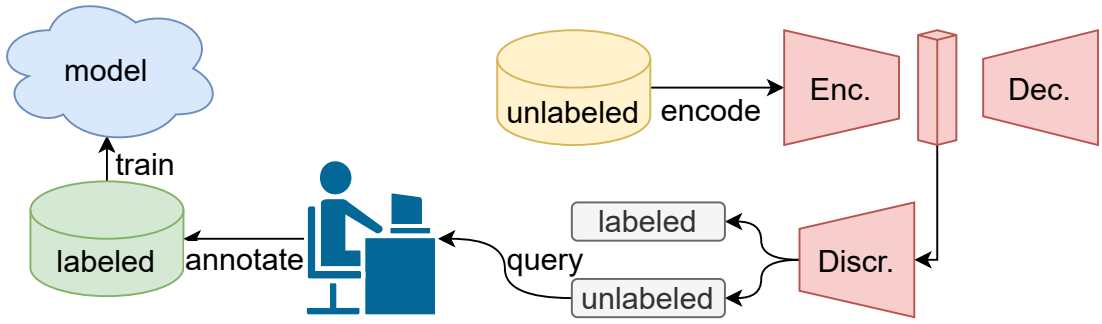


Figure 10: Generative methods, such as Variational Adversarial Active Learning, use an encoder-decoder network to learn the data distribution's latent space and jointly train a discriminator to distinguish between labeled and unlabeled samples. This serves as the selection function for active learning.

Mahapatra et al. [82] propose to generate new data samples with high informativeness to supplement smaller training datasets. They use a conditional Generative Adversarial Network (cGAN) [103] to learn the data distribution and generate new samples. The conditioning enables directing the generation process with auxiliary information, e.g., class labels. Next, a Bayesian neural network (BNN) [104] actively selects generated samples with

high aleatoric uncertainty (statistical, uncertainty in the data) and epistemic uncertainty (uncertainty in model parameters). The evaluation of such an augmented training dataset required considerably fewer annotated samples.

Sinha et al. [81] propose Variational Adversarial Active Learning (VAAL) that learns this latent space using a Variational Autoencoder (VAE) [105], see Fig. 10. Similarly to training a Generative Adversarial Network (GAN), VAAL first generates new samples and then challenges an adversarial discriminator network to differentiate the generated from unlabeled samples. Their discriminator is a Multilayer Perceptron (MLP) that estimates how representative a sample is for either the labeled or the unlabeled pool. Both the VAE and the discriminator are trained jointly, in an adversarial fashion. Then, the authors use the discriminator's predictions' probability to select those samples with low confidence from a batch.

As is common in Active Learning, follow-up work often develops hybrid approaches of uncertainty and diversity sampling. The two more recent approaches by Shui et al. [83] and by Kim et al. [84] explore balancing a hybridization of query strategies built on VAAL. Wasserstein Adversarial Active Learning [83] balances uncertainty and diversity explicitly. Uncertainty sampling reduces the empirical risk but can lead to sampling bias, and diverse sampling may benefit the exploration but become inefficient for smaller batch sizes. Task-Aware Variational Adversarial Learning (TA-VAAL) [84] extend VAAL with "task awareness", i.e., balancing sampling based on uncertainty with sampling based on the data distribution. Tasks are encoded into VAAL as the loss information from a ranking conditional adversarial network. This ranking estimates the errors of the predictions, i.e., the losses. This way, Kim et al. combine VAAL's influential sample selection with loss information for selecting difficult samples.

3.2.2 Reinforcement Learning

Reinforcement learning typically models the AL cycle as sequential decision-making problem. Alternatively, training a (deep) model to perform selections can be cast slightly differently, as meta-learning. Two methods [87, 88] combine versions of meta-learning with the REINFORCE gradient [106] as follows.

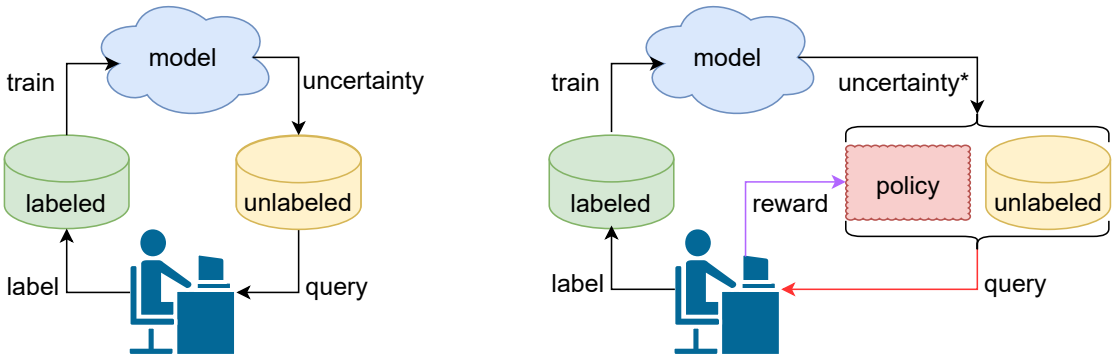


Figure 11: Reinforced Active Learning inserts a policy into the closed-loop AL system, that selects queries and receives a reward. A bulk filter, based on a heuristic, e.g., uncertainty-based sample selection, can reduce complexity (here: uncertainty*).

Ravi and Larochelle [88] propose a meta-learning approach to teach an MLP a diverse and uncertain active learning strategy. They train a prototypical network [107] similarly to zero-shot learning. They also adapt the training process. The training episodes also contain batches for N-class and K-shot problems with a support set S_{support}. However, the difference for active learning is the selection of a subset $A \subseteq U$ from the unlabeled pool U to construct larger support set $S'_{\text{support}} = S_{\text{support}} \cup A$ with the help of an oracle. To this end, the authors compute prototypes $\{c_k\}_{k=1}^{K}$ for all K classes and calculate statistics for each unlabeled sample to any prototype. Then, two MLPs learn quality and diversity statistics for all of these unlabeled samples, where quality refers to how useful a sample is to the classifier, and diversity to the dissimilarity of unlabeled samples and already selected samples in A in their feature-vector representation. The MLPs are trained with REINFORCE gradient [106] and, together with the prototypical network, can be used as an AL strategy. Bachman et al. [85] previously proposed a similar method to Ravi and Larochelle [88], that uses matching networks [108] for meta-learning instead of prototypical networks, together with reinforcement learning to learn AL (cast as sequential decision making problem). According to [88], the main differences between both meta-learning papers are the different settings, e.g., batch-mode and few-shot classification.

Learning to teach (L2T) is a student-teacher approach proposed by Fan et al. [87], that goes even further than Active Learning. While the student model can be any traditional or deep machine learning method, the teacher model predicts a training set and a loss function. Active Learning is a special case, coined "data teaching". An interesting element of L2T is the introspection of the teacher into the student. The authors use "data features", i.e., metadata like labels or histograms, but also "student model features" like training metrics such as historic training loss. Sample importance is estimated by using the samples' predicted probabilities, loss values, and margin values, effectively turning the method into a hybrid of uncertainty- and expected model change strategies.

Several authors consider active learning in a stream-based setting. Woodward and Finn [10] consider active learning in the online setting. As in a classical RL scenario, an agent, or policy, is tasked with a sequence of samples and has to decide whether to request an annotation from an oracle or whether it predicts a label. The authors use an LSTM model for the policy, and define the rewards R for a correct label prediction as $R_{cor} = +1$, for an incorrect as $R_{inc} = -1$ and for requesting annotations from the oracle with $R_{req} = -0.05$, effectively discouraging the policy. Modifying R_{inc} trades off accuracy for label requests. Fang et al. [11] apply Deep Q-learning to stream-based scenarios, and learn a policy function $Q^{\pi}(s, a)$ for policy π, that determines the utility of action a from state s. State and action are specifically tailored to the active learning problem in the NLP domain, and their evaluation shows that the selection policy learns a strategy that is transferable for cross-lingual named entity recognition. Interesting ideas for the state's composition are the direct use of the model's predictive marginal distributions and its confidence, as the policy π's available state representations. Fang et al. also shape the reward to compensate for the reward delay by rewarding changes on a held-out performance.

Ren et al. [3] summarize the more recent focus on automated design of deep active learning either by RL [12, 13] or NAS [89] as saving costs in designing heuristics manually.

Hausmann et al. [12] extend the ideas of Fang et al. [11] to a general model that learns an acquisition function. The authors train a Bayesian network as a policy network that learns in a reinforcement feedback loop in every labeling round. They bootstrap the network by

relying on pre-filtering using existing heuristics. Figure 11 visualizes such an extension of the classic AL loop. Liu et al. [13] also propose to learn an acquisition function for active learning, but specifically for the task of person re-identification using a CNN. A policy selects a sample to query the oracle with. The reward formulation uses an interesting measure of uncertainty: they use a triplet loss to measure the heteroscedastic uncertainty of the data (uncertainty in the input data, inherent in the data). The reward for a label is the difference between the farthest positive and the nearest negative sample in a batch. Hence, highly rewarded samples are those that are harder to distinguish.

Geifman and El-Yaniv [89] propose to combine active learning with Neural Architecture Search. Their motivation is that the optimization of model architecture in the active learning loop addresses the issue of "hindsight knowledge", and that model architectures have to be already known for a problem for active learning to be truly effective. Thus, the authors propose to alternate the optimization of the architecture and the active learning step.

In contrast to RL approaches, Konyushkova et al. [86] propose to label samples according to the expected error reduction and learn a greedy regression model for this task. However, Casanova et al. [14] then combine their ideas, specifically representing the state space S with a smaller, with a RL agent and Q-Learning. The policy is trained for semantic image segmentation and selects relevant regions in images for the oracle to annotate.

3.2.3 Imitation Learning

IL is closely related to RL, but while an agent actively explores an environment in RL, a policy learns from imitating experts in IL frameworks, such as DAgger [90].

Relevant for this review are IL policies, that learn AL from "experts". These experts themselves can be implementations of AL strategies. Liu et al. [91] propose to teach a policy the imitation of an "algorithmic expert", and use the DAgger [90] framework. Their expert essentially randomly sub-samples a small dataset from the unlabeled pool and retrains a classifier on each sample individually. Then, they derive a sample-wise "preference score" based on the measured change of the model's prediction quality for each retrained network. This derived target is similar to AL based on expected error reduction. In summary, their method follows one leader but requires re-training per each sample. This design decision is computationally expensive, and hence limiting, as well as sub-optimal due to the random sub-sampling itself.

Liu et al. implement the policy's state similar to other works by Fang et al. [11] or Fan et al. [87]. However, instead of the classifier's predictive marginal distribution and confidence [11], or a hybrid of uncertainty- and expected model change ("data features", "student model features") [87], they construct the state primarily from dataset representations and (predicted) labels. It consists of their prediction model's representations of the sample under consideration, of the labeled pool, and of the unlabeled pool, as well as the predicted label of the sample and all labels of the labeled pool. In our publication [P2], we ablate different state elements to shed some necessary light on this area of research and experiment with gradient signals as proposed by Ash et al. [36].

A more recent work by Kong et al. [109], called NimbleLearn, also uses Imitation Learning for batch-mode AL. However, the authors propose to learn a policy network that predicts

an "ideal sample" for each batch. The authors train a policy network π on a fully labeled source dataset and apply this to unlabeled datasets in an "active transfer", similar to Liu et al. [91]. Their policy network does not select the sample from the set of all unlabeled samples but instead predicts a feature vector P_r of the ideal sample at round r. π uses the base model's state s, which includes samples' feature vectors of labeled, unlabeled, and selected samples, as well as more "general features" like the percentage of each class in the labeled sample pool, model coefficients and entropy. The sample selection action a then selects the unlabeled sample that is closest to the predicted feature vector as follows

$$a_r = \text{argmin}_{x \in D_{unlabeled}} \text{dist}(x, P_r) \tag{23}$$

where dist is the cosine distance. Kong et al. train their policy in simulation using DAgger. The imitation learning loop starts with a random policy, that selects Q samples per round and fills a batch, selecting ideal samples, based on their validation score, similar to [91]. The evaluation of text classification with an LSTM classifier as the base model f shows three types of "active transfer". In the direct transfer of f in a transfer learning fashion, the cold transfer applies a pre-trained policy π but randomly initializes the parameters θ of f, and the warm transfer transfers both π and f from simulation to the target dataset. The authors show that the knowledge transfer of the policy π is better than transferring trained base models f.

3.2.4 Multi-Armed Bandit

A branch of research casts learning an Active Learner as a Multi-Armed Bandit (MAB) problem [92, 93, 94].

Initially, Baram et al. [92] considered the task of online active learning with an ensemble of AL heuristics and propose a combination algorithm (COMB). Using an ensemble of "experts" instead of following only one leader like [91] may perform more consistently over the learning cycle than only one heuristic [92]. Here, the choice of which one expert to select from the ensemble is the MAB problem to solve. The authors propose a semi-supervised maximum entropy performance measure over the unlabeled pool to score each expert individually. For classification, this is the hand-crafted Classification Entropy Maximization criterion. Then, a MAB algorithm has to balance "exploration" of the potential performance of any available expert with the "exploitation" of a well-performing expert, to maximize the total "reward" (gain in accuracy) over time. It chooses the highest ranked expert to select samples to query the oracle with.

Hsu and Lin [93] propose Active Learning by Learning (ALBL), that extend the performance measure by using an importance-weighted test accuracy, that can estimate the target performance more directly than CEM. In addition, the authors treat the experts as the available choices instead of a changing set of available samples like COMB [92]. In a follow-up paper, Chu and Lin [94] investigate whether the learned strategy or "experience", can be transferred to other datasets. The authors propose Linear Strategy Aggregation (LSA) to continuously update the learned model with the goal of maximizing AL performance.

Recently, Kim and Yoo [110] tackled active learning in a data imbalanced setting with a MAB. Their method blends query strategies with a novel minority preferential query in a similarly adaptive way as previous work [92, 93]. Given acquisition functions, e.g., BALD or variations ratio, their novel minority preferential strategy uses an uncertainty measure conditioned

on the minority class, adapting the "strength" of the preference over the whole duration of the active learning process. The three arms of the bandit are the conventional, the P-minority, and (P-MIN) the E-minority query strategies (E-MIN). Here, P-MIN sampling identifies minority classes by sampling the most uncertain instances from the unlabeled pool, measured by the acquisition function. Analogously, E-MIN identifies the minority classes in the labeled pool. The probability to select a possible arm is updated through regret minimization.

3.2.5 Meta Learning

Methods from Meta Learning are "learning to learn" [111].They often operate in the few-shot learning setting. Hochreiter et al. [111] initially used a meta-learner LSTM to learn the optimization of a task-specific learner with very fast convergence compared to Stochastic Gradient Descent.

As such, some work explores more variants of learning to learn [95, 96, 97]. Chen et al. [96] use a recurrent network to optimize black-box function, Finn et al. [97] learn an optimal weight initialization for diverse tasks, and Ravi and Larochelle's LSTM Meta Learner [95] goes one step further and uses an LSTM to directly predict the deep network's weight updates. Despite their interesting approaches, Jamal et al. [99] found that for hard tasks, the meta-learners have a tendency to overfit. The meta-learners are also computationally expensive [98], fail to converge [100] or have a limited temporal horizon [96, 101].

A more direct approach to learning to actively learn was recently proposed by Li et al. [102]. It is based on the sampling strategy of the expected model change. The authors propose to predict the loss of a target deep model, from its extracted features and combine it with a ranking loss. Hence, they use a loss prediction module to cast deep active learning as a ranking problem and name the method learning to rank for active learning.

3.2.6 Limitations of current Deep Active Learning methods

Current approaches to Deep Active Learning, and more specifically IL, are limited in several aspects. First, the methods lack support for selecting full batches to query the oracle with. Furthermore, methods may be slow to train due to the unrolling of small subsets of possible choices to estimate an optimal selection. In addition, the state formulations are weaker and lack introspection into the task learner model, e.g., a CNN to classify images. From this follows the lack of adaption to changing sample needs of the task learner model during the lifetime of the AL learning loop. Finally, once learned policies may not be transferable to other network architectures or even other or larger datasets. Hence, Deep Active Learning remains an important field of research. We address these limitations in the publication [P2] in Appendix B.

3.3 Active Learning of Similarity Metrics

This section focuses on the state of the art in active learning of similarity metrics. Hereby it focuses on triplet mining [16, 31], Sec. 3.3.1, on Active Metric Learning from triplets [5, 15], Sec. 3.3.2, using probabilistic models [4, 5, 15, 112, 113], Sec. 3.3.2.1, or constructing query

batches [6, 114], Sec. 3.3.2.2, and on learning similarity (of sport scenes) in user studies [8], Sec. 3.3.3. Sec. 3.2.6 finally discusses current limitations.

3.3.1 Triplet Mining

The triplet network (see Sec. 2.2.2) learns an embedding where similar samples are encoded in more similar vectors, according to a learned metric, and dissimilar samples are mapped further apart [33].

The methods for selecting an anchor p, a positive sample p^+ and a negative sample p^- for training a triplet network is called triplet mining. This is conceptually related to active learning, since the selection of suitable training data may lead to faster convergence during training [16]. There are two contexts: offline and online mining. This is similar to the difference in batch-mode or streaming-mode Active Learning, in that in offline triplet mining, a method selects p^+ and p^- before batch construction (from a pool of data). In online triplet mining the triplets are constructed from a smaller batch, similar to acquisitions in streaming-mode AL. Furthermore, Xuan et al. [16] distinguish four fundamental mining concepts, see Fig. 12:

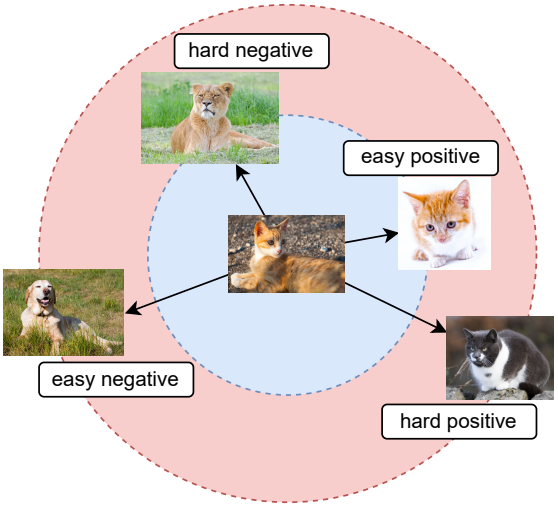


Figure 12: Given an anchor sample p, the other samples are defined through a distance metric and their similarity. This figure shows the hard and easy negative (class) samples, as well as the hard and easy positive (class) samples.

- **Hard positive mining**. These samples have a high distance to the anchor, but are of the same class (or of high similarity) [16].
- **Hard negative mining**. Such samples are negative, but they have a low proximity to positive samples in the embedding. These samples are supposedly beneficial for the triplet learning task [16], but together with hard positive mining, this may lead to noisy gradients and bad convergence [115].
- **Easy negative mining**. In contrast, easy negative samples have a (very) high distance from positive samples. Due to this, their gradients are not as informative [16].
- **Easy positive mining**. Similarly, easy positive samples are more similar to the anchor, and thus also have a lower distance to it. Still, these samples may help maintain the intra-class variance and reduce the over-clustering problem [16]. Intuitively, the

lowest distance similar samples are pulled together, while the farther distant similar samples are not, maintaining the intra-class variance.

The recent benchmark study by Musgrave et al. [115] furthermore summarizes mixed concepts. A variety of complexity is supposed to further improve learning, such as different types of negatives (easy, semi hard, hard). The adaptive selection of hard samples in an online batch is also reported as beneficial, if they are harder than the hardest positive sample. However, Musgrave et al. critique the possible improvements since the 2006 introduction of contrastive [116] and triplet [32] losses as artifacts of inconsistent evaluation settings.

3.3.2 Active Metric Learning

(Deep) Active Metric Learning aims to learn a model incrementally. It queries an oracle for annotations such that the training converges quickly, i.e., such that the annotation costs are reduced [5, 6, 15]. The learned model may then be used for classification, clustering or ranking. Interestingly, a perceptual metric may instead capture human-perceived similarities [5, 6], that can be compared to multi-label learning or hierarchically structured knowledge [117].

As previously discussed, there exist "semantic gaps" [118] between human annotations and classical feature extraction (see Sec. 2.4.1). It follows that, depending on the width of the semantic gap, the learned metric may either be defined on learned features [6, 114], on probabilistic models [4, 5, 112, 113] or on a combination [P3, 15].

3.3.2.1 Probabilistic Models

Jamieson and Nowak [112] phrase the problem as fitting non-metric data, such as humans' innate similarity functions, into a low-dimensional ordinal embedding. Their method collects annotated tuples of relative comparisons (triplets, see Sec. 2.2.2). They select only those queries for labeling, for which it holds that previously collected information is ambiguous for the embedding model, that is MDS).

Tamuz et al. [4] propose to learn similarity of data that is above the semantic gap purely from a crowd of human annotators. The similarity function between pairs of samples is called the "kernel". Hence, their "Crowd Kernel" method fits the collected annotations to an Euclidean embedding like MDS. To reduce the required full set of comparisons for triplets of $\mathcal{O}(n^3)$ for n samples, they propose an adapted, iterative sampling with complexity $\mathcal{O}(n \log l)$ where l is the lower number of compared samples. The idea is that all samples (count of n) are only compared to a subset with l elements. Tamuz et al. propose to select these queries by maximizing information gain, relative to the previously learned probability distribution. For example, they randomly choose an anchor sample p and then maximize the information gain of the choice of the other two samples b and c of the triplet (p, b, c). The information gain is the difference of the entropy of the posterior distribution $H(\tau)$ minus the probability P of either sample b or c of a triplet being closer to a: $H(\tau) - PH(\tau_b) - (1 - P)H(\tau_c)$. Now, the algorithm maximizes this for different instances of b and c.

Heim et al. [113] extend these works by incorporating auxiliary information in the form of feature vectors for each sample. These vectors may be meta-information, like properties of taste (salty, sweet, etc.) in the Yummly Food Dataset dataset that the authors use to validate their method. This information is used to reduce the model's uncertainty.

Canal et al. [5] build on the previous work on active triplet learning [4, 31, 42, 112, 119] and generalize tuples to larger rank orderings of more than three samples. This increases the context that human experts may use to resolve ambiguities and decide on a similarity metric more easily. The authors then introduce a method to select such larger tuples that is based on maximizing mutual information and that uses a probabilistic MDS. They also present simplifying assumptions to render the method tractable, such as statistical independence of sequential queries, that is commonly assumed in AL [59].

In a recent paper, Nadagouda et al. [15] unify the tasks metric learning and classification in one common framework. They use queries of arbitrary tuple sizes, but replace the MDS with a probabilistic DNN. Furthermore, they reformulate the query objective. It does not query for a complete ranking of a tuple $(t^1, t^2, .., t^n)$ of arbitrary size relative to the anchor sample, but simply queries for the one most similar sample from the n available samples. This is then dubbed the nearest neighbor to the anchor, and the method is hence named Info-NN. The probabilistic inference on the DNN uses Monte Carlo sampling [120] to estimate the probability distribution of each sample (see Sec. 2.3.2). This is required for the computation of the mutual information between query and the DNN's embedding. The authors follow Houlsby et al. [42] and define informativeness of new queries as i) a measure of the uncertain samples given the current embedding, and ii) a measure of the certainty of the oracles. The first property avoids redundant queries, and the second property avoids uncertain query responses.

3.3.2.2 Batch Queries

Constructing batches of queries [6, 114] may decrease the training cost and thus increase efficiency of human annotators. It is a natural extension to querying larger tuple sizes [15, 59]. The recent work [6, 114] is similar to pool-based AL for classification (see Sec. 2.3), in that it aims to decorrelate [114] informativeness and diversity or to direct select the least biased subset [6]. Compared to AL methods for classification tasks like BatchBald [9] (see Sec. 2.3.4), the methods presented in this chapter are applicable to relative similarity queries such as triplets. Both methods learn an embedding from triplets using a DNN, and present efficient methods for batch-mode Deep Active Metric Learning.

Kumari et al. [114] approach the selection of whole batches with a two-step approach. Their aim is to select non-redundant batches triplets, that is similar to online triplet mining (see Sec. 3.3.1) but within an AL framework. First, they select a sub-set of possible queries with high entropy. However, they find that such selection has a strong correlation and thus performs worse than sequentially selecting the same number of queries (i.e., one at a time). Thus, their second step decorrelates these informative samples to reduce the redundancy between queries. Similarity to Core-Set [41], the authors propose to estimate a covering set over the "overcomplete" set of informative triplets that were sub-sampled from the pool. They use the greedy farthest-point sampling [121] heuristic to solve the NP-hard covering set problem. They propose four distance measures that are conditioned on the fitted DNN model:

- Gradient distance, as proposed by Ash et al. [36] for batch-mode DAL, measures the expected model change of the DNN represented by normalized gradient vectors per triplet.
- Euclidean distance measures the average of l_2 distances between possible orderings of a triplet's components' embeddings.

- Centroidal distance instead averages the triplet's embeddings but still calculates the l_2 distance between these averaged embeddings of triplets.
- Orientation distance is a centroidal distance extended by the triplet's orientation in the embedding space. This idea is similar to defining a clustering distance over orientation and magnitude of gradient vectors proposed by Ash et al. [36].

The more recent work by K. Priyadarshini et al. [6] postulates that the online selection of triplets (like in [114]) leads to highly correlated (redundant) selections with low utility, e.g., because the ad-hoc heuristics listed above do perform inconsistently and the informativeness of samples is only a point estimate. This estimate may be sub-optimal and then lead to compounded error in subsequent query selections. K. Priyadarshini et al. propose a principled maximum entropy method that is promising to be less biased. Compared to the two-step approach of Kumari et al., this information-theoretic method estimates the whole probability distribution and then selects batches with the maximum joint entropy over the whole pool. This way, it reduces the redundancy and maximizes informativeness and diversity. Hereby, diversity is not defined over distance metrics in the embedding, but instead over the entropy of the model's uncertainty measures, similarly to BatchBald (Kirsch et al. [9], see Sec. 2.3.4).

The authors propose several contributions to make their method tractable. First, they use the Monte-Carlo Dropout method [122] to efficiently estimate the unlabeled triplets' mean and covariance. From these, they derive a multivariate Gaussian distribution over the mean and variance of the triplets. Importantly, this captures the inter-triplet correlation. In a second step, a greedy heuristic selects those triplets that maximize the informativeness of a selected batch. It sequentially selects the triplet t_k at step k from the available set of unlabeled triplets T_U with maximum conditional entropy given the batch of previously selected samples B_{k-1}

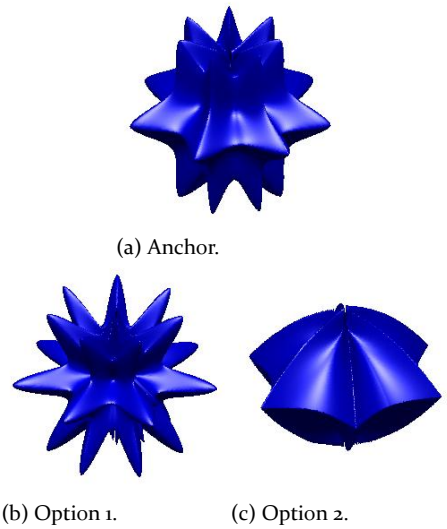
$$t_k = \text{argmax}_{t \in T_U \setminus B_{k-1}} H(B_{k-1} \cup \{t\}) - H(B_{k-1}). \tag{24}$$

3.3.3 Conducting User Studies

This section describes the recent work on user studies for active machine learning in sports [8, 66, 67] and on experimentation platforms [52, 53] to run AL studies.

Di et al. [8] propose to learn ranking from relative pairwise comparisons of sports scenes, e.g., soccer, ice hockey, or American football. The goal is to learn user-specific interests and improve an embedding that uses the Euclidean distance. They use a linear rankSVM with features extracted from player trajectories by a convolutional autoencoder. However, the user-specific interests are not collected from users. Instead, the authors use hand-crafted simulations of seven models of user preferences. Di et al. assume that users are interested in the types of shots (e.g., three-point shots) or the similarity in the ball trajectory, that are all computed from position data. Then, they run a simulation to train the rankSVM for basketball data. Afterwards, they validate the rankSVM with a real user study with 3 participants. The study is designed as a clickthrough experiment, and uses a rank quality metric for evaluation: participants query the system with one scene, and then view several pages with responses of similar scenes. They click on similar scenes in the order that ranks the search results by similarity. In summary, the participants were satisfied with the responses in 90% of the time, after being asked a binary yes/no question. The ranking

metrics suggested high quantitative quality as well. An example of a user study as the means to evaluate the quality of a Sports Scene Retrieval system was presented by Wang et al. [66, 67]. The authors conduct a user study solely to evaluate their proposed "play2vec" system in direct competition with Chalkboarding [1]. In the study seven domain experts first familiarize themselves with the visualization of the trajectory data in a warm-up phase. Then the participants are queried with an anchor scene and two retrieved top-1 scene from Chalkboarding and play2vec, for a total for 10 randomly chosen query scenes per participant. The participants then respond which of the two samples is more relevant, i.e., similar, to the anchor scene, without knowing which algorithm retrieved which samples. The relative comparison between two retrieval systems resulted in more reliable estimate of relative retrieval quality than the smaller study by Di et al.



(a) Anchor.

(b) Option 1. (c) Option 2.

Figure 13: (a) shows the anchor of a query, and (b) and (c) the options to select. Participants visit a website that shows these or similar images, and asks them to select the most similar option with respect to the anchor.

Jamieson et al. [52] and Sievert et al. [53] present the experimentation platform "NEXT" for conducting real-world AL user studies. It is designed for real-world, scalable and also reproducible experiments in mind, and can be accessed online by simply using a web browser. Participants open a prepared query website, and interact with it to submit their response to the system. For example, see Fig. 13 for the accompanying task for annotating triplets. Then, the NEXT platform can run AL algorithms, update models, or log information. The authors stress that real experiments have additional properties that influence the performance of AL algorithms that are impossible to model in a simulation, because they are complex and unforeseeable. The algorithm and network response times are cumulative and can cause fatigue in human participants. Furthermore, annotators may annotate with different levels of quality, or respond in a non i.i.d. manner (see Sec. 2.4.3). For example, humans notice delays of about 400 ms or longer. After subtracting the network delays, an AL method has to run in about 50 to 100 ms. The authors then describe that this can be challenging for contextual bandits that are computed in real time. Furthermore, there is a trade-off between updating the model and running AL methods for longer. Updating models with collected annotations more frequently avoids collecting "stale" responses that do not contain novel information. However, increasing run time of AL methods can lead to more informative queries. Both options use the same budget of computational time resources.

3.3.4 Limitations of current Active Similarity Learning methods

Current IR systems for unstructured data and studies on the active learning of ordinal embeddings are limited in several aspects. First, sports scene retrieval systems still rely on the Euclidean distance. This distances captures global ordinal structure better than local one due to the effects of the high dimensionality [P1, 28], despite the findings in previous smaller-scale studies [1]. Still, no exhaustive studies exist that aim to efficiently learn similarity metrics for trajectory data from human annotators. Second, modern active learners for ordinal embeddings only randomly sub-sample the pool of available samples to construct queries in order to achieve run-time, that are sensible for studies with human participants. Furthermore, they employ very simple embedding methods that do not generalize to new samples. In addition, only few studies on AL consider the effects of sample composition on query complexity for humans. This complexity factor may result in skipped responses and fatigue, which leads to more error prone labeling. Some authors even call for a deeper study of the influence of queries in terms of similar psychometric properties. Lastly, current experimentation frameworks for AL user studies lack support for modern DL and thus limit the field from advancing in this promising direction. We address these limitations in the publication [P3] in Appendix C.

Part III

Contributed Papers

4 Deep Siamese Metric Learning: A Highly Scalable Approach to Searching Unordered Sets of Trajectories

Löffler, Christoffer and Reeb, L., Dzibela, D., Marzilger, R., Witt, N., Eskofier, B., Mutschler, C.: Deep Siamese Metric Learning: A Highly Scalable Approach to Searching Unordered Sets of Trajectories. In: ACM Transactions on Intelligent Systems and Technology, Volume 13, Issue 1, Article N.: 6, pp 1-23, (2022). See Appendix C for the full paper.

Abstract This work proposes metric learning for fast similarity-based scene retrieval of unstructured ensembles of trajectory data from large databases. We present a novel representation learning approach using Siamese Metric Learning that approximates a distance preserving low-dimensional representation and that learns to estimate reasonable solutions to the assignment problem. To this end, we employ a Temporal Convolutional Network architecture that we extend with a gating mechanism to enable learning from sparse data, leading to solutions to the assignment problem exhibiting varying degrees of sparsity. Our experimental results on professional soccer tracking data provide insights on learned features and embeddings, as well as on generalization, sensitivity, and network architectural considerations. Our low approximation errors for learned representations and the interactive performance with retrieval times several magnitudes smaller show that we outperform the previous state of the art.

Author contributions C.L. wrote and edited the paper, reviewed related work, and interpreted and discussed the results. C.L. and L.R. conceptualized the methodology, wrote code and performed experiments. C.L. supervised the research project. N.W., D.D., R.M. and C.M. contributed with ideas and discussions. C.L. designed the experiments and ablation studies. L.R., D.D., R.M., N.W., B.E. and C.M. reviewed and edited the paper. N.W., B.E. and C.M. supervised the work.

5 IALE: Imitating Active Learner Ensembles

Löffler, C., and Mutschler, C.: IALE: Imitating Active Learner Ensembles. In: Journal of Machine Learning Research, Volume 23, pp. 1-29 (2022). See Appendix B for the full paper.

Abstract Active learning prioritizes the labeling of the most informative data samples. However, the performance of active learning heuristics depends on both the structure of the underlying model architecture and the data. We propose IALE[1], an imitation learning scheme that imitates the selection of the best-performing expert heuristic at each stage of the learning cycle in a batch-mode pool-based setting. We use DAGGER to train a transferable policy on a dataset and later apply it to different datasets and deep classifier architectures. The policy reflects on the best choices from multiple expert heuristics given the current state of the active learning process and learns to select samples in a complementary way that unifies the expert strategies. Our experiments on well-known image datasets show that we outperform state-of-the-art imitation learners and heuristics.

Author contributions C.L. conceptualized the methodology and reviewed the literature, implemented the algorithms and experiments, and discussed the results. C.L. wrote the initial draft of the paper and C.L. and C.M. reviewed and edited the paper. C.M. contributed with ideas and discussions, and supervised the work.

[1] `IALE` is pronounced /eɪl/.

6 Active Learning of Ordinal Embeddings: A User Study on Football Data

Löffler, C., Fallah, K., Fenu, S., Zanca, D., Eskofier, B., Rozell, C., Mutschler, C.: Active Learning of Ordinal Embeddings: A User Study on Football Data. In: Transactions on Machine Learning Research, pp. 1-26 (2022). See Appendix C for the full paper.

Abstract Humans innately measure distance between instances in an unlabeled dataset using an unknown similarity function. Distance metrics can only serve as proxy for similarity in information retrieval of similar instances. Learning a good similarity function from human annotations improves the quality of retrievals. This work uses deep metric learning to learn these user-defined similarity functions from few annotations for a large football trajectory dataset. We adapt an entropy-based active learning method with recent work from triplet mining to collect easy-to-answer but still informative annotations from human participants and use them to train a deep convolutional network that generalizes to unseen samples. Our user study shows that our approach improves the quality of the information retrieval compared to a previous deep metric learning approach that relies on a Siamese network. Specifically, we shed light on the strengths and weaknesses of passive sampling heuristics and active learners alike by analyzing the participants' response efficacy. To this end, we collect accuracy, algorithmic time complexity, the participants' fatigue and time-to-response, qualitative self-assessment and statements, as well as the effects of mixed-expertise annotators and their consistency on model performance and transfer-learning.

Author contributions C.L. implemented algorithms, designed and conducted experiments, wrote and revised the manuscript. K.F. and S.F. contributed to the theoretical framework and experimental design, offering insights. K.F. and D.Z. provided manuscript feedback. C.R. guided the research, including experimental and algorithmic design. C.M. supervised and helped refine the experiments and methodology, and provided manuscript feedback. B.E. supervised the project and provided manuscript feedback.

Part IV

Perspectives

7 Discussion

This Chapter discusses the future perspectives that are relevant to the objectives of this thesis. Sec. 7.1 discusses research avenues for working with sports scene search and other trajectory data, and Sec. 7.2 addresses the same for Active Learning.

7.1 Sports Scene Search

We proposed to learn a lower-dimensional embedding of sports scenes [P1]. This representation preserves a target metric and estimates the assignment problem of the unstructured trajectory ensemble.

Learning a vector representation, such as Word2Vec [68, 69] in NLP, is an important component of many ML approaches, and also for applications such as IR [123]. It should become a crucial part of sports scene retrieval systems like Chalkboarding [1] as well. A lower-dimensional representation of complex data is important, because it simplifies the problem by mitigating the curse of dimensionality [27, 28]. The vector representation effectively acts as a learned dimensionality reduction algorithm. Hence, the goal of learning a lower-dimensional representation is to preserve distances in the learned embedding. A sub-goal is to estimate the optimal assignment of ensembles of trajectories as part of the learned embedding function because the exact calculation is computationally infeasible with $\mathcal{O}(n^3)$ for large datasets using the Hungarian algorithm [23].

7.1.1 Addressed Literature Gaps in Deep Metric Learning

We presented a solution for jointly learning a distance preserving embedding and estimation of the optimal assignment problem [P1]. Prior work processed trajectory data in Play2Vec [66, 67], but did not handle ensembles of unstructured data that is common for team sports trajectories, or built data structures for retrieval [1, 7, 20] instead of reducing dimensionality. We contributed a DNN-based method that jointly estimates the assignment and learns a distance preserving embedding of complex trajectory ensembles. Our Siamese network handles sparse inputs using "Gated Temporal Convolutions". The benefits of our learned vector representation of the highly complex data are manifold. First, we break the curse of dimensionality effectively. The learned embedding reduces the dimensionality for samples of 5 s from 5.750 to between $64 - 1024$, depending on the required fidelity. Furthermore, we proposed to estimate the assignment problem via different templates, and further reduce computational complexity at inference. All these contributions lead to an accelerated similarity search, that is faster by orders of magnitude compared to the state of the art [1].

7.1.2 Metric Learning Perspectives

This section discusses the possible perspectives of the contribution [P1].

Trajectory Data Applications. Using our proposed Deep Siamese Metric Learning at their core, other sports scene search applications [1, 7, 20] can benefit from the lower

dimensional representation and estimation of the assignment problem. Other research groups may focus on adjacent team sports, like basketball or ice hockey, that feature a similar property of unordered trajectories. Furthermore, other applications with trajectory data, such as in commerce (tracking of customer movements [124, 125]), medicine (walking gait analysis from trajectory data[1]) or industry (Radio Frequency (RF) positioning that searches for similar channel-frequency or sender-receiver assignments), could benefit from smaller representations.

Recommendation Systems. Our learned embedding lends itself to further refinement by using recommendation systems from the field of Information Filtering (IF) [126] or by active embedding search [59]. IF is a specialized sub-field of IR, that Belkin et al. [126] considers as being "two sides of the same coin". If a search engine would be an IR system, then a recommender system would be an IF system. On one hand, personalized recommendations and rankings can be learned from users by systems such as Di et al. [8]. On the other hand, active embedding search [59] searches for a preferred point within a Euclidean space, and would directly benefit from a lower-dimensional distance preserving embedding with an estimated solution to the assignment problem.

Complex Similarity Metrics. A natural extension of our metric learner is the use of other, more complex similarity metrics than the Euclidean distance, that handle spatiotemporal information (see Sec. 3.1.2). Future research may conduct larger studies with more participants, other (team) sports or longer scenes than Sha et al. [1], and experiment with other metrics, such as OT (see Sec. 1), and find alternative distance metrics to the Euclidean distance. The use of any distance metric is possible because our Siamese network approach is independent of their choice. The distance between input samples can even be different from the learned embedding distance. The benefit of using our method is greater, the more complex the distance metric is, that our method approximates. Querying users can serve as the base to learn Mahalanobis metrics [127, 128, 129], users' preferences [130], or to jointly learn both a metric and preferences [131].

7.1.3 Limitations of the Contributed Work

Our proposed Deep Siamese Metric Learning approach has several limitations, that we partially address in later work [P3]. Recall that we estimate the pairwise distance and the assignment of two sets of trajectories. This leads to two problems.

First, the Euclidean distance that we use to estimate distances is susceptible to noise if used with high-dimensional data such as trajectory data. While this does not overly impact the global structure of the learned embedding, the effect is noticeable in the smaller structures. We see this in the evaluation of the rank correlation of the top-100 nearest neighbors [P1]. It shows that the ranking of the original data and the learned embedding diverge slightly. However, this issue can be easily solved by introducing a two-step system, that computes an optimal assignment and distance on the reduced (e.g., top-100) nearest neighbors directly.

Second, we use templates to assign trajectories to channels in order to estimate a solution to the assignment problem. We show that these templates may vary in quality as part of

[1] See for example project https://www.mad.tf.fau.de/research/projects/personalization-of-muscoskeletal-models/

our evaluation [P1]. Future work may propose enhanced templates similar to the adaptive elliptical assignment that we propose in a later work [P3].

Apart from algorithmic limitations, a final limitation is that we process only football data. Hence, future work may apply our method to other applications in sports, such as American football, basketball or baseball, or even transfer the methodology to other domains.

Our overriding idea for Deep Siamese Metric Learning is to jointly estimate the distance and assignment of high-dimensional trajectory data. The learned embedding not only enables new use-cases in the sports domain, such as interactive similarity search, but may also spark future work, e.g., into active search or IF [59, 126].

7.2 Deep Active Learning

To reduce the costs of annotating unlabeled sets of data, it is essential to focus on the parts of the data pool that create the most value, i.e., that are the most informative samples for a machine learning model. Asking annotators to label batches at a time is not only more efficient for the participant [2] because the slow fitting of models can use more data at once, and annotating can be done in parallel. The predominant training paradigm Stochastic Gradient Descent (SGD) for DNNs uses (mini) batches [60]. Furthermore, training DNNs is an expensive optimization. It is more efficient to re-train on more than one sample at a time to reduce the annotators' waiting time between queries. Hence, one objective of DAL is to construct queries of more than one sample for the annotating oracles [3]. Good batches are more beneficial for learning and reduce the required amount of labeled samples. This typically requires acquisition functions that combine diverse and uncertain criteria [3, 9]. Limitations of other work are to set a good balance between different criteria over the AL cycle as proposed by [91, 93, 110]. For example, Batch Active learning by Diverse Gradient Embeddings (BADGE) [36] acquires diverse samples based on the directions of gradient signals, but requires relatively large acquisition sizes to beat mono criteria AL [P2].

7.2.1 Addressed Literature Gaps in Deep Metric Learning

We presented the method IALE [P2] that learns an acquisition function (policy π) that is suitable for DAL. During policy training, π imitates the best acquisition function from a set of functions, given the internal state of a task model f and a policy training dataset. We address the limitations of prior work in multiple aspects. First, we construct batches of variable sizes. Second, we unify different acquisition functions of an arbitrary kind. And third, we use the task model's state to generate queries. This allows the policy to adapt the acquisition function to the model's state during the AL cycle, as our evaluation shows.

7.2.2 Deep Active Learning Perspectives

This section discusses the possible perspectives of the contribution [P2].

Applying IALE. Other work, that can directly benefit from IALE, may apply it in domains typically addressed by DAL. As surveyed by Ren et al. [3], some examples from computer vision are tumor classification in medicine, plankton classification or counting of cells, and other examples from NLP are listed as Named Entity Recognition (NER) or sentiment

classification, among others. Any application that requires annotations but is also cost-sensitive, may generally benefit from AL, and specifically from IALE.

Extending Imitation Learning. Our work on IALE is in line with prior work by Liu et al. [91] and parallel research by Kong et al. [109], that both use IL to learn acquisition functions. Future research may incorporate aspects of IALE, such as the formulation of its state, or the learning objective proposed by Kong et al. [109].

7.2.3 Limitations of the Contributed Work

Future work may address current limitations, such as the sub-sampling used in both methods, that may be handled with adjacent ideas like Learning to Rank [102]. This may further increase the methods' performance by considering the full pool distribution. Extending the methods and their evaluation to other problems such as annotating imbalanced datasets, or to consider noisy oracles, are other interesting research avenues, as are larger experiments with more experts.

Lastly, IALE's ability to transfer the trained policy π between different classifier architectures and datasets is an intriguing property, that necessitates further research. The conceptual connection to the "learning to learn" [111] paradigm, like Ravi and Larochelle's LSTM Meta Learner [95], that predicts a target network's parameters, outlines an interesting parallel to IALE's learned acquisition function, that predicts the most informative samples for a target network.

7.3 Active Metric Learning

Learning humans' innate similarity function estimates a metric from as few annotated samples (e.g., triplets) as necessary. This is important because it can help bridge the semantic gap, which may open up new use-cases to prediction models that use learned features based on "human kernels" as Tamuz et al. put it figuratively [4, 6]. The primary goal of Active Metric Learning is to find a lower-dimensional embedding of high-dimensional data, that encodes a distance metric between samples which is of interest to humans [4, 5] [2]. This representation may be learned from human annotations, where the "active" learning component aims to reduce the cost associated with labeling data. Using DNNs similarly as us or, e.g., Nadagouda et al. [15], focuses on learning features from high-dimensional data. Problems are manifold, and are founded in both Active Learning and Metric Learning alike. The challenges of AL are the selection of suitable queries to participants that reduce the annotation costs [4, 5, 6, 15, 112, 113, 114]. A major limitation is the efficiency of querying annotators. This is a central objective of research, e.g., it may be increased with larger query- or batch sizes [6, 59, 114]. Another limitation is the embedding function, that may be learned only for a closed set due to the common use of MDS. Here, recent work addresses this by instead training an Artificial Neural Network (ANN) as model [15].

[2] See also the relevant mission statement at https://siplab.gatech.edu/

7.3.1 Addressed Literature Gaps in Deep Metric Learning

In our publication Active Learning of ordinal embeddings [P3], we propose to adapt an entropy-based AL method [5] with a triplet mining heuristic [16]. First, the mining effectively sub-samples the pool to reduce the computational complexity of the query selection step while simultaneously providing primarily useful samples to select from. Second, the reduced pool leads to less skipped responses by human annotators, and leads to less fatigue and lower likelihood of errors due to reduced concentration. And third, the method guides the training of a DNN that generalizes to new samples. We perform a user study to validate the method both qualitatively and quantitatively. It sheds light on the strengths and weaknesses of our method when applied in a real-world experiment with human participants.

7.3.2 Active Metric Learning Perspectives

This section discusses the possible perspectives of the contribution [P3].

Other work that addresses learning semantic similarity from few annotations given high-dimensional input data can benefit from our method. Typical data domains are images or complex time series like positions, and the issues to solve are cost-related (time cost, fatiguing). Example applications may be learning the semantic similarity between images in photo databases [132] or medical image processing [133] that usually requires expert annotations.

Our work learns a metric for sports scenes, hence similar systems for other sports, such as ice hockey [8], American football [8, 66, 67] or basketball [1], may benefit from fine-tuning their similarity metric from annotations.

7.3.3 Limitations of the Contributed Work

Our work builds on prior work [5, 16]. Future research may incorporate ideas from our work, such as the adaptive sub-sampling of the pool, the training of a generalizing prediction model, or the optimization of user studies towards reduced fatiguing (skipped queries) besides pure improvement of quantitative metrics like accuracy. It may further address current limitations.

First, modern DAL relies on batched queries to an oracle, and recent publications [6, 114] propose suitable acquisition functions that select uncertain but also diverse queries. However, the benefit of a larger training batches of more than only one sample is mainly efficiency. This can be constructed from larger tuples from which we generate many pairs or triplets to train with.

Second, it may be beneficial to directly use a probabilistic DNN [15] instead of MDS or t-Stochastic Triplet Embedding (tSTE) by incorporating the Monte-Carlo method by Gal et al. [38].

Third, our proposed triplet mining approach may be further improved and should be diligently analyzed. The work by Musgrave et al. [115] shows minimal differences for different loss functions, and a critical follow-up study on triplet mining could lead to similar insights.

Finally, performing larger user studies using thousands of annotators, e.g., by employing Amazon Mechanical Turk[3] like Canal et al. [5], on additional datasets may empirically show the benefits (or uncover weaknesses) of our proposed method.

Our overriding idea for Deep Active Metric Learning is to combine the predictive power of Deep Neural Networks for unobserved samples with entropy-based Active Learning to learn similarity from few annotations. This may find application as a Deep Metric Learning [133] method to bridge semantic gaps [4].

[3] https://www.mturk.com

8 Conclusion and Outlook

8.1 Findings and Contributions

The findings and contributions of this dissertation can be summarized as follows:

- In [P1], we presented an approach based on a Deep Siamese Network that learns to jointly estimate an optimal assignment between two unstructured ensembles of trajectories of a football dataset, and to preserve their distance metric while learning a lower dimensional embedding. The learned vector representation vastly accelerates similarity search compared to prior approaches [1].
- In [P2], we presented an Active Learning strategy, that uses Imitation Learning to learn a unified AL policy (i.e., an acquisition function) from a diverse set of baseline heuristics. It proposes an expressive state definition that includes, e.g., gradient signals and describes predictive uncertainty, and supports batch-mode pool-based AL. We experimentally show that our policy acquires more informative samples than any baseline, and that it is transferable between datasets and classifier architectures.
- In [P3], we presented a method to increase the computational efficiency of the information gain-based Infotuple via triplet mining, that leads to reduced labeling cost and time spent annotating data. We propose to use the embedding space of an ANN for triplet mining, and thus optimize the query acquisition. We conduct an AL user study on a football trajectory dataset to demonstrate the effectiveness of our method compared to both active and passive baselines.

In summary, this thesis presents methods to learn small but accurate representations of football trajectory data, to improve Active Learning for Deep Learning models, and finally to efficiently use AL for learning similarity metrics to fine-tune and personalize the trajectory representations.

Compared to previous methods, the learned representations of the trajectory data do not require additional side information in training, such as roles or other meta information. Instead we only employ a simple distance metric like the Euclidean distance [P1]. Similarly, while the learned embedding is small, it retains sufficient information without the need for additional structure to reduce the search space, such as hierarchical tree structures or clustering that prior work relied upon. Moreover, we propose to directly fine-tune the embedding using Active Learning to learn the experts' innate similarity function in an interactive user study [P3]. This way we can retain all achieved benefits of [P1] while personalizing the underlying similarity metric. Still, our contribution may be integrated with side information to strengthen such systems. Recent work [134] from the field of Few-shot classification discusses our work in that new context, and proposes a type of class hierarchy to enhance transfer learning.

Towards the enhancement of AL, we propose to learn an acquisition function from demonstration, forgoing the need for hand-crafted heuristics. Instead our policy learns directly from data and the Deep Learning model's state [P2]. Our method may be directly used in applications, or be integrated into algorithm selection frameworks, such as TAILOR [135], who propose to integrate "Learning Active Learning" methods like ours and transfer them

to unseen datasets. In addition, our method's introspection component may be used in other approaches easily.

In addition to this work's main contributions, the works on augmenting human labor with ML [P4, P5] have sparked discussion. Our ideas motivated further research into related applications for Human Activity Recognition [136] or quality assurance [137], and served as examples for ML applications on the edge [138].

8.2 Outlook

Employing Deep Learning effectively for Information Retrieval, teaching DL models cost-efficiently using Active Learning and learning similarity directly from humans to bridge semantic gaps will continue to pose challenges to practitioners of ML. In this context, we have proposed novel methods or frameworks for trajectory-based Information Retrieval, Deep Active Learning, and Deep Active Metric Learning, that addressed and solved previously open problems. Still, many problems remain open, and many applications still search for their methodology.

Regarding the future of Artificial Intelligence in sports, Tuyls et al. [139] provide a broad discussion. Concerning trajectory data, the authors see open research in the recommendation of generated trajectories using prediction models that may have far reaching consequences. It may serve to propose novel solutions for constrained set pieces, help find short-term plays (with subsets of players) or even discover new long-term strategies for entire teams. Our work [P1, P3] may efficiently measure the similarity of generated solutions and thus serve as a selection function. In addition, may incorporate expert knowledge into similarity metrics to guide the selection and find better proposals in less time.

In the visionary publication on Deep Active Learning by Ren et al. [3], the authors list many open issues and research questions. Our work [P2] is compatible with many of their ideas and may serve as a basis. First, the authors call for more reliable and comparable benchmark studies, that unify currently inconsistent evaluations of methods, baselines, and strategies. By publishing a code supplement for [P2], and by replicating previous evaluations, we follow this spirit and enable future work to easily evaluate `IALE`. A concrete proposal to extend our work [P2] is a generalized training on generated datasets, similarly to the idea proposed by Ren et al. This would rely on our method's ability to be transferred between datasets. Furthermore, a hybrid strategy that combines `IALE` with semi- or unsupervised methods is feasible, e.g., through a label propagation component that relies on the ANN. Finally, the literature [3, 135] increasingly focuses on more task-independent AL, e.g., through adaptive algorithm selection [135], which may also benefit from `IALE` transferable policy. To summarize, our work touches upon several of these directions and may serve as a basis for, e.g., hybrid, task-independent AL strategies learned from data.

Learning similarity from annotators requires many computationally expensive steps, such as the selection of samples or the re-training of models. Further simplifications on the query selection method [5] or using auxiliary tasks [15] may reduce these costs. Concretely, the side-information available in different application domains, e.g., the sparse labels in football data, may serve as auxiliary tasks for finer clustering of embeddings and pre-selection of query samples. This may become relevant for transferring our methods [P1, P3] to novel datasets with denser or hierarchical annotations, or to other tasks like classification. Recent work leverages such hierarchical knowledge structures. Yin et al. elicit information from

experts [117] and Zhang et al. perform Few-shot classification [134]. Both embed data such that distances are semantically meaningful and can be organized hierarchically, such that fewer queries or shots are required. These types of approaches may directly benefit from more efficient representations [P1], from more meaningful distance metrics [P3], or from querying experts less often [P3]. An additional application besides similarity search is active search [59]. Once a ordinal embedding has converged and is adapted to a specific similarity metric, it may enhance this type of IR. Concretely, recommendation systems or search engines, that use vector databases or directly embeddings, e.g., sports scene retrieval systems, may not query for similar samples but instead search for yet unknown samples directly. Such search methodologies [59] may use the previously unavailable fine-tuned embedding generated by our methods [P1, P3] for trajectory-based applications as a basis to accelerate the iterative active search.

To conclude this work, we put our contributions into perspective. As of today, many ML applications face a scarcity of annotated data [140], and will most likely continue to do so. This is especially relevant with the proliferation of ML to novel applications and data sources. At the same time, the trend of generating more data of higher complexity accelerates [141]. These developments clash with the need for high quality annotations, especially for high risk applications [1]. Hence, the key enabler for ML systems remains the human-in-the-loop, who plays a crucial role in providing expertise and high quality annotations for models' initial training and even their operating life cycle [142]. We hope that our contributions facilitate a more reliable and efficient use of annotator time, even on highly complex data, and the generation of error free and high quality annotations for an ultimately better and more trustworthy use of Machine Learning.

[1] EU proposal of Artificial Intelligence Act from 2021 even mandates high annotation quality in Art. 10, 3. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex%3A52021PC0206

Appendix

A Deep Siamese Metric Learning: A Highly Scalable Approach to Searching Unordered Sets of Trajectories

Deep Siamese Metric Learning: A Highly Scalable Approach to Searching Unordered Sets of Trajectories

CHRISTOFFER LÖFFLER* and LUCA REEB*, Fraunhofer IIS, Germany and Friedrich-Alexander-University Erlangen-Nürnberg (FAU), Germany
DANIEL DZIBELA, Fraunhofer IIS, Germany
ROBERT MARZILGER, Fraunhofer IIS, Germany
NICOLAS WITT, Fraunhofer IIS, Germany
BJÖRN M. ESKOFIER, Friedrich-Alexander-University Erlangen-Nürnberg (FAU), Germany
CHRISTOPHER MUTSCHLER, Fraunhofer IIS, Germany

This work proposes metric learning for fast similarity-based scene retrieval of unstructured ensembles of trajectory data from large databases. We present a novel representation learning approach using Siamese Metric Learning that approximates a distance preserving low-dimensional representation and that learns to estimate reasonable solutions to the assignment problem. To this end, we employ a Temporal Convolutional Network architecture that we extend with a gating mechanism to enable learning from sparse data, leading to solutions to the assignment problem exhibiting varying degrees of sparsity.

Our experimental results on professional soccer tracking data provides insights on learned features and embeddings, as well as on generalization, sensitivity and network architectural considerations. Our low approximation errors for learned representations and the interactive performance with retrieval times several magnitudes smaller shows that we outperform previous state of the art.

CCS Concepts: • **Computing methodologies** → **Machine learning**; • **Information systems** → **Information retrieval**.

ACM Reference Format:
Christoffer Löffler, Luca Reeb, Daniel Dzibela, Robert Marzilger, Nicolas Witt, Björn M. Eskofier, and Christopher Mutschler. 2022. Deep Siamese Metric Learning: A Highly Scalable Approach to Searching Unordered Sets of Trajectories. *ACM Trans. Intell. Syst. Technol.* 13, 1, Article 6 (February 2022), 22 pages. https://doi.org/10.1145/3465057

1 INTRODUCTION

The wide spread of devices and systems that generate positioning data allows for trajectory mining in a variety of applications. This has the potential to improve and revolutionize how we are currently working with unstructured multi-agent trajectory data, e.g., in sports [31], in mobility monitoring [33] or when we optimize transportation [11, 21]. However, as such trajectory data is often unstructured and unlabeled, their useful exploitation is still limited.

*Both authors contributed equally to this research.

Authors' addresses: Christoffer Löffler, christoffer.loeffler@iis.fraunhofer.de; Luca Reeb, reebla@iis.fraunhofer.de, Fraunhofer IIS, Nordostpark 84, Nuremberg, Germany, 90411 and Friedrich-Alexander-University Erlangen-Nürnberg (FAU), Erlangen, Germany; Daniel Dzibela, Fraunhofer IIS, Nuremberg, Germany, daniel.dzibela@iis.fraunhofer.de; Robert Marzilger, Fraunhofer IIS, Nuremberg, Germany, robert.marzilger@iis.fraunhofer.de; Nicolas Witt, Fraunhofer IIS, Nuremberg, Germany, nicolas.witt@iis.fraunhofer.de; Björn M. Eskofier, Friedrich-Alexander-University Erlangen-Nürnberg (FAU), Erlangen, Germany, bjoern.eskofier@fau.de; Christopher Mutschler, Fraunhofer IIS, Nuremberg, Germany, christopher.mutschler@iis.fraunhofer.de.

For instance, in sports applications, a central problem is the search and retrieval of similar occurrences in trajectory databases [31]. Such occurrences, i.e., *scenes*, can be formally defined as ensembles of trajectories, i.e., trajectories of the movements of multiple agents. Before we can compare (i.e., derive the similarity between) two scenes, we need to find an optimal assignment (*assignment problem*) for the particular trajectories, i.e., we need to find the correct match between the corresponding actors/players. Next, we can calculate a suitable distance metric between scenes. An incorrect assignment in the first step induces a large error in the estimated distance, i.e., the similarity between the two scenes. However, alignment and pair-wise comparison quickly becomes infeasible for larger databases (such as those that are available in the sports industry).

Previous work that deals with the infeasible computational complexity that comes with large databases either filters the data [10, 31, 41], learns approximations of the optimal assignment algorithm [40], or constrains the possible assignments themselves [31, 32] and thereby introduces approximation errors. While these approaches do in fact accelerate search and retrieval, their primary goal is to reduce the search space. However, this not only results in less optimal assignments and quality but also yields sub-optimal retrievals due to the limited amount of searchable data.

Fig. 1 illustrates two scenes from a trajectory database of soccer players of the German *Bundesliga*. Each scene shows variations of a reoccurring pattern that can be found throughout the season. Each single trajectory follows an inherent strategy and role, that may differ between teams and even for each player over the course of the game. Our primary goal is to find such similar scenes without constraining neither trajectory assignments nor the total search space over all samples in the database. A fast search scheme for high-dimensional trajectory sets is beneficial to sports-related applications and especially team sports (such as football, soccer, basketball, etc.), but it is not limited to that: it is also beneficial for work that is as diverse as searches similar trajectories in cellular positioning data for human mobility analysis [33], mining of frequent patterns in urban transportation [21], clustering air traffic trajectories for optimization [11], and even wildlife management in ecological behaviour analysis [8].

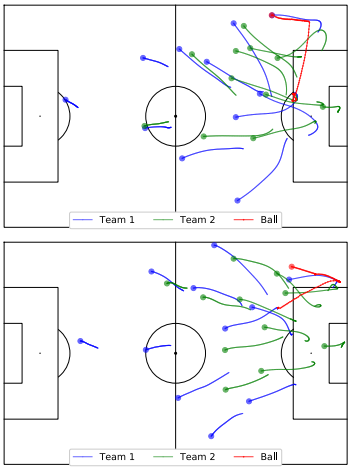


Fig. 1. Two scenes from a soccer database.

Current scene retrieval systems (especially in sports) are mainly based on manual annotations by analysts such as [20], or other automatic classification attempts that are equally expensive, e.g., highlight extraction using video data [12]. Still, tracking data becomes ubiquitously available and should become more usable despite it being unstructured. However, among the main challenges are the indexing and retrieval within these large trajectory datasets [44], given (i) multiple moving agents and (ii) the quantity of data. First, multiple trajectories of different roles are present, e.g., player formations in adversarial team sports. To calculate the distance between two scenes correctly, a correct assignment between trajectories is key. Simplifying this task by solving assignments with available meta-information only works for few applications such as player roles [31] in sports. However, this neither generalizes to other applications nor does it work in all sports. E.g., in soccer alone at least ten different formations exist [2] and players randomly break formations [2]. On the other hand, methods to optimally solve the problem such as combinatorial optimization [31] do not scale well. Second, given a manually annotated *query scene*, finding similar scenes in an entire sports season worth of data is computationally expensive. Suitable trajectory similarity functions

use metrics such as Dynamic Time Warping [29], Longest Common Sub-Sequence [37], or the Euclidean distance [31] that all need pair-wise comparisons.

This article proposes an approach which presents several solutions to the assignment problem, and which approximates the optimal distance between ensembles of trajectories, speeding up the computation needed for scene retrieval. We train a *Siamese Neural Network* (SNN) which learns a lower dimensional embedding for a given dataset, and which preserves distances between trajectories independent from use case specific distance metrics. This accelerates pair-wise comparisons while keeping the approximation error low (as we show for the *Euclidean distance* metric for a whole season of trajectories from games played in the German *Bundesliga*). This in turn enables novel interactive applications that have previously been intractable. Our contributions are as follows:

- We propose sparse and dense estimations to the assignment problem suitable for learning with deep neural networks. To this end, we extend the learning model with *Gated Temporal Convolutions* as a masking mechanism.
- We show that the learned embedding of a Siamese Neural Network, using these Gated Temporal Convolutions, approximates trajectory distances with a low approximation error.
- We show and discuss results for the *scene similarity* on a large soccer trajectory dataset. Our study includes an analysis of the embedding neighborhood and considers real wall-clock time compared to prior state of the art. We furthermore evaluate the generalizability of our method and perform detailed ablation studies.

The rest of the article is structured as follows. Sec. 2 discusses related work on trajectory similarity, sports scene retrieval systems, and approaches which solve the assignment problem. Sec. 3 formalizes the problem and Sec. 4 presents the details of our method. We show our experimental setup in Sec. 5 and discuss the results in Sec. 6. Sec. 7 concludes.

2 RELATED WORK

We discuss related work on the assignment problem (Sec 2.1), learning distance metrics (Sec. 2.2), and the search and retrieval of trajectories (Sec. 2.3), as well as the search and retrieval's special cases with event data (Sec. 2.3.1) or with tracking data (Sec. 2.3.2).

As we primarily target sports scene analysis, we give a brief overview of relevant work from that area. Usually, methods applied here do not retrieve similar scenes but analyze sports event- and position-data to predict different properties, e.g., classification and clustering. Recently, Wenninger et al. [39] evaluated modern machine learning models, e.g., convolutional or recurrent networks, to predict tactical behavior in beach volleyball such as play direction and their success. An older branch of research, among others [13, 26, 30], investigated the use of self-organizing maps that use the learned map for downstream tasks such as classification. The specific problem of classifying ensembles of trajectories was discussed in [34], however, the assignment problem was solved by design (the data was provided through skeletal tracking). The analysis of spatio-temporal trajectories of multiple players was proposed in [6, 7]. Trajectories are approximated with splines that are filtered and normalized. Several similarity measures are compared in terms of classification accuracy using the reduced representations. However, as assignments must be provided manually, the applicability is limited.

2.1 The Assignment Problem

The Hungarian algorithm [18] is an optimal solver for the assignment problem. It is used in object detection [5] to match proposed and actual object bounding boxes, in object tracking [43] to associate the identity of objects over time, and in identification [40] to match words based on their semantic similarity. Approximations [19] use neural networks based on similarity or cost matrices. As neural networks are also capable of processing unaligned data, the pre-processing

step can be skipped, either as part of the network architecture [19, 40] or as part of the learning objective [5, 43]. In our work we choose to learn it as part of the learning objective and jointly optimize a distance-preserving lower dimensional representation of the data that accelerates pairwise comparisons. To this end, we extend the model architecture with masks to support sparse data, i.e., with Gated Temporal Convolutions, to learn assignment methods from the data.

2.2 Distance Metric Learning

In metric learning, relevant approaches either learn a better *approximation* of the distance function or instead a *lower dimensional* transformation of the data that preserves a distance. Approximations as proposed in [17] are motivated by the fact that exact metrics are either not smooth enough (over small perturbations in the inputs) or too unreliable for the application. They formulate metric learning as a regression problem that is solved with Siamese Neural Networks [4] that learn the similarity. Similar to our work, a distance acts as a regression target to the network. However, the proposed model only emits a distance given two inputs, i.e., the Siamese network must be applied to a pair of data. In contrast, our approach also allows to process single inputs, so that we can construct their embedding offline, before using them in search. This allows us to quickly search for similar examples. `Sentence-BERT` [27] approximates similarity for natural language processing, specifically for sentence-pair regression, using ideas similar to dimensionality reduction. The authors propose an attention-based transformer network together with Siamese or Triplet Networks to learn a semantically meaningful embedding based on a pair-wise similarity metric as regression target (i.e., they estimate semantic textual similarity). This approach aims to learn a lower dimensional embedding for pairs of sentences. The authors do not focus on ensembles of sentences and have no assignment problem to deal with. However, we build upon the basic methodology and extend it appropriately for learning an ensemble similarity.

2.3 Trajectory retrieval

Most trajectory retrieval systems make special assumptions such as temporal bounds of interesting search spaces or focus on other data-inherent properties that cannot be generalized. For instance, Yadamjav et al. [41] propose a framework for query retrieval of similarly co-moving trajectories, e.g., convoys of objects. An indexing structure filters the dataset based on temporal constraints (i.e., discards old samples to improve computational performance) and is fast due to an in-memory table lookup. Similarity metrics such as a point-wise max-min distance, i.e., Hausdorff distance, are extended to work on subsets of cluster points. In contrast, our approach builds a distance-preserving approximated representation of the complete dataset and thus allows scaling to larger problems, while at the same time accelerating lookup speed.

2.3.1 Event-Data Based Sports Play Retrieval. Instead of predicting similarities directly from trajectory data, the methods that search in event-data develop *query formulations* on annotated event-data. The work closest to ours is presented by Richly et al. [28]. They aim to retrieve scenes using a graphical query language composed of action sequences in user-specified areas. Another approach by Decroos et al. [9] uses an SQL-like query language instead, which also allows the OR-operator to relax search constraints over characteristic, user-defined functions.

However, annotated event data is both expensive to acquire and relatively sparse. Moreover, as such search approaches operate over event-data, they are inherently limited in their performance. Queries defined on tracking-data on the other hand allow to search for very specific situations. More importantly though, the need for event-data makes these approaches only applicable when (expensive) event-data annotations of sufficiently high quality are available.

2.3.2 Tracking-Data Based Sports Play Retrieval. There is also work that uses trajectory ensembles directly as queries for similarity-based retrieval. Such methods must handle the assignment problem and the expensive distance computation.

`Chalkboarding` [31] approximates the assignment problem using player roles and a match- and team-specific role/formation template, which are learned from data as proposed in [3]. While this introduces an unquantified error, it speeds up retrievals considerably. However, the approach does not generalize well to other domains such as soccer, where the role assignments are not static within matches and teams. Sha et al. [32] address the assignment problem with hierarchical templates which are iteratively aligned over all matches. This relaxes the fixed role assignment, making it applicable to more applications. However, the hierarchy depth is not determined by semantic properties of the data, but with respect to computation speed, e.g., cluster size. For each cluster, a suboptimal assignment template is then used for the expensive distance computations on data of (unchanged) high dimensionality. To compare distant elements, even less optimal assignment templates are used. In contrast, we propose a data-driven approach to the assignment problem that uses optimal solutions as learning targets instead of finding arbitrary approximations.

An acceleration of distance computations can be achieved by a pre-clustering of the search space [10, 32]. Sha et al. [32] propose a k-means clustering that strongly depends on the Euclidean distance as its similarity measure to cluster the scenes. An interesting way to refine search results was proposed by Di et al. [10], who extended `Chalkboarding` by using a ranking function that is sensitive to user preferences, which are determined from simulated click-through data. In contrast to clustering approaches we learn a low-dimensional representation and solve scene retrieval without a pre-clustering of the scenes, making our approach agnostic to the underlying similarity metric.

An alternative to `Chalkboarding` is proposed by Wang et al. [38] who matches segments into coherent parts, i.e., ball possession phases, which they process analogously to sentences in natural language processing. Their network learns relative similarities and relations between these segments. While a qualitative user-study proves it to be superior to `Chalkboarding`, it lacks flexibility. As user-drawn query-sketches cannot be used as queries, their framework cannot handle sparsity in queries. Furthermore, single trajectories from an ensemble are indiscernible to the model, e.g., users cannot give higher weight to the ball's trajectory if that is not learned during model training. The embedding only represents a relative ordering that is specific to a similarity metric that includes relevance features for the whole ensemble.

3 PROBLEM FORMULATION

In this work, we investigate the more general case of trajectories ensemble retrieval (i.e., scene retrieval) where the ordering between trajectories is unknown. We consider trajectory ensembles as unordered, fix-sized sets of trajectories. Trajectories are temporally ordered sets of data such as spatial positions or other arbitrary types of information, and we assume that the trajectories are spaced equidistant. We represent trajectories as $S \times T$ matrices, where S is the spatial dimension of the trajectory, as follows:

$$\vec{x} = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1T} \\ x_{21} & x_{22} & \dots & x_{2T} \\ \vdots & \vdots & \ddots & \vdots \\ x_{S1} & x_{S2} & \dots & x_{ST} \end{bmatrix}$$

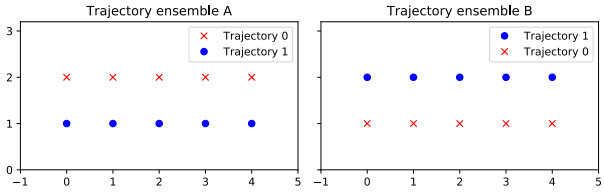


Fig. 2. Two trajectory ensembles with time increasing from left to right (-1 to 5). In the left ensemble A, the trajectories are ordered top-down, in the right ensemble B, they are ordered bottom-up. Aside from the different indexing, the data are identical.

For 3D trajectories we set $S = 3$ and for 2D trajectories $S = 2$, respectively. The distance d between two trajectories $\vec{x}$ and $\vec{x'}$ is given by the average Euclidean distance at each point in time:

$$d(\vec{x}, \vec{x'}) = \frac{1}{T} \sum_{t=1}^{T} ||\vec{x}_{:,t} - \vec{x'}_{:,t}||_2 \tag{1}$$

We chose the Euclidean distance based on the evaluation of various distance functions for the sports use case [31]. However, our method does not depend on it, as we use Siamese Networks as general function approximators.

Trajectory ensembles are given by $X = \{\vec{x}^{(1)}, \vec{x}^{(2)}, ..., \vec{x}^{(N)}\}$ where $\vec{x}^{(1)}, .., \vec{x}^{(N)}$ are trajectories. Analogous to trajectories, we represent trajectory ensembles as tensors of shape $N \times S \times T$. Note that we indexed the trajectories containing X with ascending integers. This ordering between trajectories is arbitrary and differs from the ordering of data-points within trajectories: within a trajectory the order has physical meaning, i.e., the time stamp. This has no equivalent for trajectories in an ensemble. Swapping two differently-valued and indexed data-points in a trajectory destroys its identity. For a trajectory ensemble, this is not the case: $\{\vec{x}^{(1)}, \vec{x}^{(2)}\} = \{\vec{x}^{(2)}, \vec{x}^{(1)}\}$. This arbitrary indexing also represents one of the core problems, formally known as the *assignment problem*.

Fig. 2 illustrates the assignment problem. The optimal assignment of ensemble A and B results in a distance of 0, because they match perfectly, i.e., trajectory 0 from A with trajectory 1 from B. However, there exist more possible permutations for which the distance is much greater than 0. With every new ensemble, this optimal assignment has to be calculated again, before further algorithms like distance calculations or clustering can be applied. In this work, we propose a learning approach to this problem, which approximates the assignment optimization and preserves distances, so that distance calculations or clustering remain possible. This example shows that defining the distance between trajectory ensembles is more complex than for pairs of trajectories. To compute their distances, a permutational optimization of their component's order is necessary for all pairs of ensembles X and X', that minimizes the sum over all distances between component trajectories. The Hungarian algorithm [18] solves this in polynomial time of $O(n^3)$ by iterating over the cost adjacency matrix, but is still too slow for large databases.

Let $P(X, X')$ be the permutation function found by the Hungarian algorithm which *aligns* X to X'. When $P(X, X')$ is applied to X, the indices that give the ordering are re-assigned in such a way that the sum of distances between trajectories with equal indices in X and X' is globally minimal:

$$d(X, X') = \sum_{i=1}^{N} f(X_{P(X,X')(i)}, X'_i), \tag{2}$$

with $P(X, X') = \min_{\sum_{i=1}^{N} f(X_{g(i)}, X'_i)} \{g : g \in \{1, \ldots, N\} \rightarrow \{1, \ldots, N\}\}$ and g being invertible.

Until now, we considered trajectory ensembles as unordered sets and treated trajectories equally. However, in special cases, such as team sports, we can apply some obvious simplifications without loss of generality. For soccer, we treat both teams and the ball differently, i.e., we always align attacking to attacking and defending to defending players, the ball trajectory to the ball trajectory, goalkeepers to goalkeepers. To compute the overall distance between two scenes, we sum up all individual trajectory ensemble distances, e.g., for soccer the distance between the attacking or defending team and the ball. These simplifications reduce the problem size of the trajectories that have to be assigned, in the case of soccer from 23 to 10.

We draw an example based on the two scenes from Fig. 1. Each trajectory ensemble has a color, i.e., the attacking team in blue, the defending in green and the ball in red. We compute the distance between the two scenes as follows: (1) we compute the optimal assignment of the first team's trajectories, (2) we compute d_0 by adding up the trajectory distances between all matched trajectories, (3) we repeat step 1 and 2 for the second team (yielding d_1), (4) we add up all trajectory distances between the ball trajectories and both goal-keeper trajectories (yielding d_2), and (5) we finally sum up all distances: $d_0 + d_1 + d_2$ which gives the total distance d.

4 DEEP SIAMESE METRIC LEARNING

With *Deep Siamese Metric Learning*, we propose a principled approach to approximate the exact similarity between trajectory ensembles. Instead of a hand-crafted approximation, we directly learn the exact similarity while also improving retrieval speed. The similarity metrics' accuracy directly depends on how the assignment problem is approximated, hence we propose four approaches that aim to alleviate previous limitations.

As traditional methods do not scale well, their search scope is limited to small scene datasets. This is due to the high time complexity, as described previously. For (optimal) trajectory assignments, the time complexity is $O(s \cdot n^3)$ using the optimized Hungarian algorithm for s scenes. In our approach, we simplify the search space using metric learning and different assignment methods, reducing the cost to $O(s \cdot M)$, where M is the embedding size of the metric learner.

First, for metric learning, we follow a similar idea as `Sentence-BERT` [27], that uses a Siamese Neural Network to map natural language sentences into an embedding in which the cosine distance corresponds to the semantics' similarity in a supervised regression setting. In our work we transfer and extend that idea to trajectory ensemble similarity. We map scenes into an embedding space in which the distance can be computed more efficiently (while being close to the original distance). To select the best distance for the sports use case we rely on previous work [31], where the authors evaluated a set of commonly used trajectory similarity metrics with regard to their suitability to classify sports plays into 38 distinct categories chosen by experts. Their results showed the suitability of the L_2 similarity. However, our approach is not limited to this choice as we can resort to any other similarity measure as well.

Second, we deal with the assignment problem in our framework, as well. As calculating the optimal assignment is too expensive to compute for datasets of practically relevant size, approximations were used previously, e.g., based on roles and formations in sports [31]. However, these fixed assignments of roles do not generalize well [32], not even to variations in the same sport, and neither to other use cases with more trajectories and formations [2]. Moreover, this fundamental assignment problem is intrinsic to sports, as creative changes to fixed strategies are especially beneficial for success [25]. These circumstances motivate alternative solutions to the assignment problem, i.e., forms of sparse encoding, location-based grid assignments or even random assignments, that we propose in Sec. 4.3.

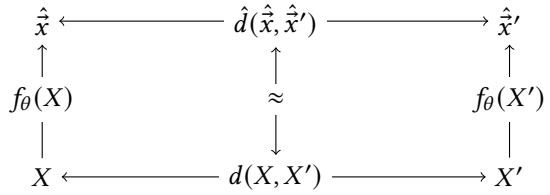


Fig. 3. Schematic of the approach that projects the high dimensional data to a low dimensional representation. The trajectory ensembles X and X' are passed through a *Siamese* network f_θ producing the embedded scenes $\hat{\vec{x}}$ and $\hat{\vec{x}}'$ respectively. The network is then trained such that the distance in the embedding $\hat{d}$ matches the ground-truth d.

4.1 Objective and Metric Learning Scheme

We consider a non-linear function approximator f_θ with parameters θ that maps inputs $X \in \mathcal{S} = \mathbb{R}^{N \times 2 \times T}$ (e.g., a trajectory ensemble) into an embedding $\hat{\mathcal{S}} = \mathbb{R}^M$ such that the distance $d : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$ over the input space is preserved in $\hat{\mathcal{S}}$. More specifically, the distance function $\hat{d} : \hat{\mathcal{S}} \times \hat{\mathcal{S}} \to \mathbb{R}$ over the embedding preserves the property $\hat{d}(\hat{\vec{x}}, \hat{\vec{x}}') \approx d(X, X')$ with $\hat{\vec{x}} = f_\theta(X)$ for all $X \in \mathcal{S}$. Fig. 3 illustrates the approach. If d and $\hat{d}$ are equal for every scene pair X and X' we can evaluate $\hat{d}$ instead of calculating the computationally expensive ground-truth distance d. This speeds up the retrieval time considerably.

We use the same distance function, i.e., the Euclidean distance $\hat{d}(\hat{\vec{x}}, \hat{\vec{x}}') = ||\hat{\vec{x}} - \hat{\vec{x}}'||_2$, as the distance function $\hat{d}$ between embedded inputs. This choice enables a wide range of machine learning algorithms to the embedding for further processing, e.g., to restrict the search space via clustering [32]. For f_θ, we use a neural network composed of several residual temporal convolution layers for the feature extraction together with fully-connected layers to map these features to the embedding space. The observations are hence pairs of trajectory ensembles, i.e., (X, X').

The dataset size is quadratic in the number of trajectory ensembles available and training on all permutations hence quickly becomes infeasible. Instead, we sample pairs uniformly at random and minimize the *empirical risk*, i.e., $\min_\theta \mathbb{E}_{x,y \sim \mathcal{D}}[L(f_\theta(x), y)]$ where $\mathcal{D}$ is the data generating distribution. We use the following loss function:

$$L(X, X') = (||f_\theta(X) - f_\theta(X')||_2 - d(X, X'))^2, \tag{3}$$

i.e., the Mean Squared Error (MSE) and include two regularization terms:

$$||f_\theta(X)||_2 + ||f_\theta(X')||_2 + ||\theta||_2. \tag{4}$$

The first term penalizes embedded points far from the origin and the second is a standard L_2 weight regularization term that penalizes large weights in the parameters of the neural network. Note that the first term is required for convergence as the norm of the difference between two points is shift-invariant, i.e., as there are infinite optimal solutions if no order is imposed.

We use two identical Siamese Neural Networks (SNN) [4] that process distinct inputs that are joined by a metric function, i.e., the distance function. Each twin projects the input into the embedding space, on which we compute the loss. In contrast to the conventional use, we never join the outputs of the twins in a joint layer. Instead, the Euclidean distance of their outputs is directly compared to the ground-truth distance. This is due to the different problem formulation, as conventional SNNs learn a similarity metric based on class labels, whereas we use the networks to directly learn an existing metric [27]. Note that the structure of this solution is well tailored to the problem setting: f_θ is deterministic, therefore $d(\vec{X}, \vec{X}) = \hat{d}(\hat{\vec{x}}, \hat{\vec{x}}) = 0$ (identity is indiscernible)

and $\hat{d}(\hat{\vec{x}}, \hat{\vec{x}}') = \hat{d}(\hat{\vec{x}}', \hat{\vec{x}})$ (symmetry) is given by construction. In essence, the SNN learns a distance-preserving dimensionality reduction.

4.2 Siamese Network Architecture

Our SNN processes sparse temporal data. While historically recurrent networks such as Long Short-Term Memory (LSTM) [15] have been the dominant approach to process time-series data, more recent variants of convolutional networks such as the Temporal Convolutional Network (TCN) are more successful [35, 36]. Bai et al. [1] apply TCN to a wide variety of datasets with en par or better results than LSTMs while being faster to train.

The core difference of TCNs compared to classical convolutional networks is the left-padded same-convolution. Intuitively, this means that the history is padded while the present is not. This mechanism implicitly assigns weight to data based on its recency, as values in the past are replaced by zero-padding. A large benefit is the performance: TCNs are inherently parallelizable as they compute responses locally and stationary by using the last k time-steps. TCNs are hence well suited for data with a maximum dependency length, such as the trajectory ensembles in this work, for which we require a common length by construction. In addition, the assignment methods may produce sparse input data (see Sec. 4.3) as, for instance, there are at least 47 roles in soccer with only 23 trajectories including the ball. Hence, we require an architecture that can easily handle missing inputs. Prior work on TCN [35, 36, 42] provides methods that explicitly handle sparsity successfully in the domains of image and audio processing.

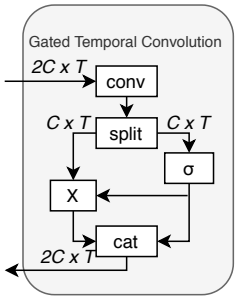


Fig. 4. Schematic of the gated temporal convolution operation.

We adapt the gating mechanism from [42] where the masks are passed to convolution layers alongside the data. The neural network continuously updates the masks and uses their information to extract features. Following a convolution operation, the masks are applied to the data in a *differentiable* way, i.e., by multiplying the data with the sigmoid of the masks. This is visualized in Fig. 4. The input $\mathrm{x} \in 2C \times T$ is composed of $C \times T$ values and an equally shaped mask. The temporal convolution layer extracts features in x and updates the mask. The mask channels are then passed through the sigmoid function and multiplied element-wise to the value channels. In the end, value and mask channels are concatenated and returned. We adapt this gating mechanism to 1-dimensional time-series data for TCNs and refer to it as *Gated Temporal Convolution*.

The network architecture itself is similar to the one in [1] and is based on a Residual Network (ResNet) [14]. While we selected this architecture due to its previous use with gating mechanisms (although on data from a different domain), its general popularity, and typically good performance, our proposed method is not limited to the use of this specific architecture. We adapt the temporal convolution block from [1] by replacing the regular convolutions with Gated Temporal Convolutions. The network is composed of a series of residual Gated Temporal Convolution blocks, see Fig. 5 with a fully-connected layer as an embedding space at the end. The ResNet's typical skip-connections add the input of the residual path to the output to enable deeper networks, see Fig. 5. The Rectified Linear Unit (ReLU) activations [23] after each convolution do not change values in the mask channels, as their range is $[0, 1]$. The channel size C is constant throughout the network but the dilation rate is increased with depth.

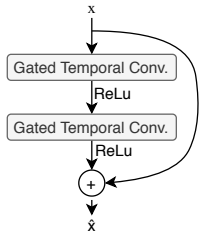


Fig. 5. Residual Gated Temporal Convolution.

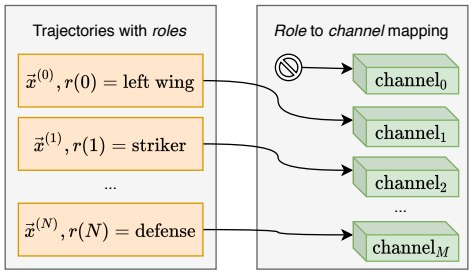


Fig. 6. Channel assignment by role: the trajectories on the left are assigned to channels based on their role r. Each channel corresponds to only one role, e.g., channel_1 always contains left wingers or remains empty (such as channel_0.)

4.3 The Assignment Problem

To estimate the similarity between two trajectory ensembles we need to have a reliable assignment of trajectories between the two ensembles with each trajectory itself being of dimension $C \times T$. This is different from convolutional network learning on image data where we assume a stable ordering of color channels. Our idea is to still consider trajectory ensembles like images when we learn convolutional filters. We map each trajectory to a channel, so that the kernel learns on the C-dimensional space with C as the temporal information for the multiple *intensity values* of T.

We require a stable ordering of channels (as it is the default case for color channels in image processing). A channel encodes information that convolution kernels learn, and neural networks presumably even extract an abstract *meaning* from these. Hence, we also construct assignment schemes that aim for relatively stable orderings. However, we also investigate if a random assignment (that breaks with the assumption) is also a suitable *channel assignment method*.

(1) Random Assignment. For each game, the trajectory assignment to a channel is stable, but then random in between games. This introduces randomness as the assignments preserve no roles. We hypothesize that this unbiased assignment looses valuable additional meta-information, e.g., player roles or positional grids, that the more sophisticated schemes can capture.

(2) By Role. For each trajectory, we may know meta information on its specific *role*, e.g., in soccer, there exist a multitude of formations. We can construct an assignment that assigns each *role* a separate channel. Given trajectory $\vec{x^{(i)}}$ with a *role* r, we assign $\vec{x^{(i)}}$ to the channel $c(r)$, where c is a bijective map from the set of roles to channel indices. The *roles* are unique in each scene. The twins of the Siamese Network can learn two trajectory ensembles where both have c_1 populated, but one twin has channel c_0 masked out in favor of another channel, see Fig. 6 for an illustration.

This assignment method leads to sparse input data when the roles in two compared scenes do not match up. For the soccer use case, a total of 23 roles are defined but only a subset is populated with data. With two teams, this results in a total of 47 channels, i.e., 23 per team and the ball, and a high degree of sparsity. We fill empty channels with zeros and use masks during the training procedure to deal with these missing values.

(3) Inferred from data. Alternatively to using meta information, we infer assignments from trajectory data. We construct these assignments either as *role positions*, using a fixed set of artificial *template* trajectories similar to [32], or as *grid positions*, that are uniformly spatially distributed. Both approaches are illustrated in Fig. 7. We calculate *role positions* using the Hungarian algorithm [18] and align the trajectory ensembles to the template. Each position p in the template is mapped to channel $c(p)$ bijectively. If trajectory $\vec{x^{(i)}}$ is assigned to p_j, $\vec{x^{(i)}}$ will be inserted into $c(j)$. Each cross (position) in Fig. 7a corresponds to a channel-trajectory pair and symbolizes a *role*. Hence,

Attribute	Range	Meaning
Identity	$\mathbb{N}$	The data-generating entity, e.g., player-id.
Role	$\{0, \ldots, 21\}$	Encoding of player role
Team	$\{0, 1\}$	Encoding of the team
Period	$\{0, 1\}$	Game period
X	$[-52.5, 52.5]$	Position in meters
Y	$[-34, 34]$	
T	$\mathbb{N}$	Time of recording
Ball possession	$\{0, 1\}$	Encoding of team in ball possession
Game activity	$\{0, 1\}$	Indicates if game was paused or active
T	$\mathbb{N}$	Time of recording

Table 1. Trajectory data and meta information in the dataset.

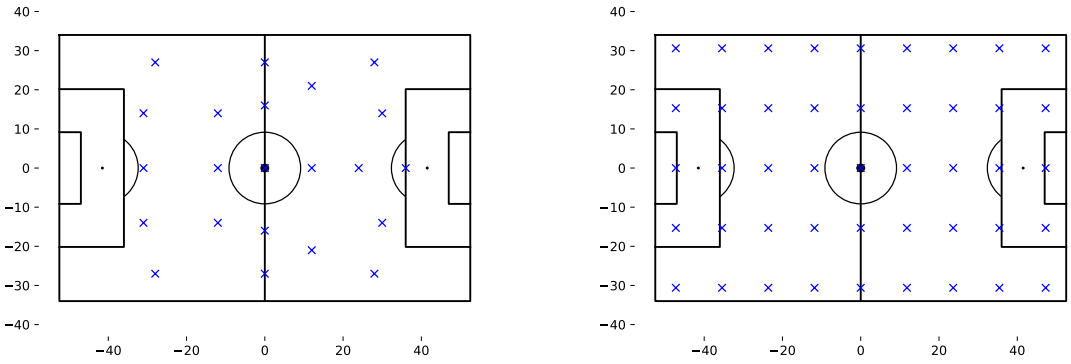
there are 22 positions in the template for the soccer use case, transforming the data into a sparse representation.

To calculate *grid positions* we use an unbiased template with increased spatial resolution. The network thus learns a set of roles from data instead of relying on pre-defined roles. This increases sparsity but also computational cost. Similarly to role positions, we again use the Hungarian algorithm to align the trajectory ensembles (during training). The Siamese networks rely on the gating mechanism to handle sparse inputs that result from some assignment methods, an architectural property that generalizes beyond the soccer domain. Section 5 shows a performance analysis and ablation study which discuss the benefits of our proposed solution.

5 EXPERIMENTAL SETUP

5.1 Dataset

The dataset we used for our experiments consists of 306 soccer matches from the *1. Bundesliga* from season 2014/15. We discarded two of these games due to incorrect or missing annotation. For each match, positions for each player and the ball were extracted from multi-perspective video feeds at a sampling rate of 25 Hz. The trajectory data structure is given in Tab. 1, and includes meta-information such as roles, ball possession or whether the game was active or not.



(a) Role position template.

(b) Grid position template.

Fig. 7. Channel assignment templates: each cross indicates the constant position of a trajectory in the template over time. Fig. 7a shows an assignment template based on the expected role position. Fig. 7b shows an assignment template based on a grid of trajectories. It is composed of nine different positions in the horizontal axis and five in the vertical axis. The difference between trajectory counts for each axis was chosen such that the positions are approximately spaced out equidistantly.

Pre-processing. We describe the pre-processing steps to convert the dataset into an appropriate scene representation for Siamese Neural Networks with Gated Temporal Convolutions. We extract game scenes of fixed length to generate trajectory ensembles. In our experiments, we set the length to either 5 s (as suggested in [10, 31, 32]) or 20 s (which was suggested by a number of sports scouts we interviewed). For our experiments, we simplify the implementation by applying several constraints to the extracted data[1]:

- One team has significantly more ball possession than the other (this circumvents an assignment over teams).
- The attacking team plays from left to right in order to use an absolute coordinate system. This ensures that differences if the playing direction in two otherwise identical scenes do not complicate the distance calculations.
- The game is active (*not paused*) for most of the time during the scene. This filters irrelevant parts of the game.
- No players are missing during the scene (even though our method could handle sparse data).

Normalization. Common normalization schemes such as mean subtraction and division by standard deviation (assuming a Gaussian distribution) is impractical for distance functions as maximum-likelihood estimation quickly becomes intractable on large datasets. The number of distances is quadratic in the number of scenes, so we approximate the normalization, while the statistics for the inputs $\vec{X}$ can be estimated directly. For the approximation, we use a running-average, i.e., similar to batch normalization, using the following (iterative) update rules:

$$\mu_{i+1}^{(x)} = (1-\beta)\mu_i^{(x)} + \beta \mathrm{E}[x_i] \tag{5}$$

$$\sigma_{i+1}^{(x)} = (1-\beta)\sigma_i^{(x)} + \beta \sqrt{\mathrm{Var}[x_i]} \tag{6}$$

with the momentum parameter $\beta \in [0, 1]$ determining the *mass* of the moving average and a series of values x_n, stopping after 100,000 steps. The inputs $\vec{X}$ are then normalized as:

$$\vec{X} \leftarrow \frac{(\vec{X} - \mu^{(\vec{X})})}{\sigma^{(\vec{X})}}. \tag{7}$$

For the distance d however, the mean-subtraction must be omitted, as otherwise it may take negative values, which the Siamese network cannot produce. Hence, we downscale the distances to preserve ordering between samples via

$$d(\vec{X}, \vec{X}') \leftarrow \frac{d(\vec{X}, \vec{X}')}{\sigma^{(d)}}. \tag{8}$$

5.2 Configuration

For our experiments, we extract in total (after pre-processing) about 1,200,000 scenes of 5 s and 400,000 scenes of 20 s from 304 games. We split the scenes extracted from the matches into three sets for training, validation and test at 80/10/10% ratio, e.g., for 20 s long scenes this yields 309,665/44,424/44,536 samples per dataset. However, due to its combinatorical nature, we cannot process all possible combinations and their distances. Hence, we construct large subsets for 5 s and 20 s by randomly sampling scene pairs and computing their ground-truth distances, resulting in (per scene length) 10 million pairs for the train dataset, and 1 million each for validation and test dataset.

For all experiments we use PyTorch [24] and the Adam optimizer [16] with an initial learning rate of $\eta = 1e^{-3}$ and $\beta_1 = 0.9$ and $\beta_2 = 0.99$ for the momentum terms, the decay factor $s_\eta = 1e^{-1}$

[1]These are only meant to simplify our implementation but do not pose any limitations on our method in general.

every $N_\eta = 5$ epochs, for 15 epochs in total. We set the L_2 weight penalty λ_1 to 0.001. Preliminary experiments showed no convergence issues due to drifting, hence we set the embedding L_2 penalty $\lambda_2 = 0$. All inferences are computed using single-threaded code on an AMD Ryzen 7 2700 processor with 16GB RAM and a GeForce RTX 2070 GPU while we use NVidia Tesla V100 GPUs for training.

5.3 Evaluation Metrics

The *evaluation metrics* measure the errors introduced by the learned representations. For retrieval, we are interested in (1) the structural correspondence, (2) the ordering and the neighborhood structure, and (3) the retrieval set comparison.

(1) Structural correspondence. First, we measure the *structural correspondence* between the distance of pairs in the original and the learned representation using the MSE and the Mean Absolute Percentage Error (MAPE). MAPE is the expected absolute error relative to the ground-truth in percent, e.g., a MAPE of 10% means that on average, the prediction is off by 10 percent:

$$\text{MAPE}_{\mathcal{D}} = \frac{1}{|\mathcal{D}|} \sum_{i=1}^{|\mathcal{D}|} \left| \frac{d(\vec{X}_i, \vec{X}_i') - \hat{d}(\hat{\vec{x}}_i, \hat{\vec{x}}_i')}{d(\vec{X}_i, \vec{X}_i')} \right|. \tag{9}$$

(2) Ordering and neighborhood. The ordering and neighborhood information of scenes is crucial. However, evaluating the performance of approximations of the nearest neighbor search on the full dataset is prohibitively expensive. Hence, we only work on the smaller subset of the validation dataset $\mathcal{M}_{sub}$, containing 5025 scenes with all $5025 \cdot 5024/2$ ground truth distance pairs. We use the top-N Mean Spearman Rank Correlation Coefficient (MSRCC) as a metric on $\mathcal{M}_{sub}$ to analyze the structure of the embedding, i.e., the scene ordering between the ground-truth and the Euclidean distance in the embedding. We use rank correlation to quantify the correspondence between the scene ordering in the ground-truth and in the embedding as approximation errors that change the ordering between samples are more relevant.

We calculate the MSRCC as follows. For a query scene $\vec{X}_{query,i}$ we find the N nearest neighbors $\{\vec{X}_0, ..., \vec{X}_N\}$ and the distances to them $A_i = \{d(\vec{X}_{query,i}, \vec{X}_0), ..., d(\vec{X}_{query,i}, \vec{X}_N)\}$ using the ground-truth distances. We compute their distances in the embedding $\hat{A}_i = \{\hat{d}(f_\theta(\vec{X}_{query,i}), f_\theta(\vec{X}_0)), ..., \hat{d}(f_\theta(\vec{X}_{query,i}), f_\theta(\vec{X}_N))\}$ and the *Spearman rank correlation coefficient* $r_i = \text{r}(A_i, \hat{A}_i)$ between distances in embedding and ground-truth. The MSRCC for $\mathcal{D}$ is then the average over every query scene:

$$\text{MSRCC}_{\mathcal{D}} = \frac{1}{|\mathcal{D}|} \sum_{i=0}^{|\mathcal{D}|} r_i. \tag{10}$$

We evaluate the top-N MSRCC for $N = 100$, as the nearest neighbors are the most relevant samples for the data retrieval use case, and for the whole dataset in order to assess the overall structure.

(3) Retrieval set comparison. To quantify how many of the closest N scenes are retrieved via the embedding, we calculate the top-N Mean Intersection over Union (MIoU). This is the intersection between the N closest scenes in the embedding and in the ground-truth divided by their union, averaged over every query scene. We can therefore relate it to an adaption of the accuracy metric that measures the ratio between the number of scenes that are correctly retrieved and the number of scenes that are incorrectly retrieved in the top-N. In Fig. 8, the MIoU is given as a function of the accuracy, e.g., the

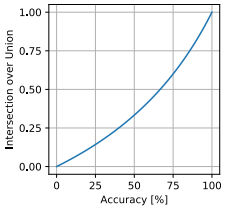


Fig. 8. Relation between the IoU of same-sized sets and the accuracy.

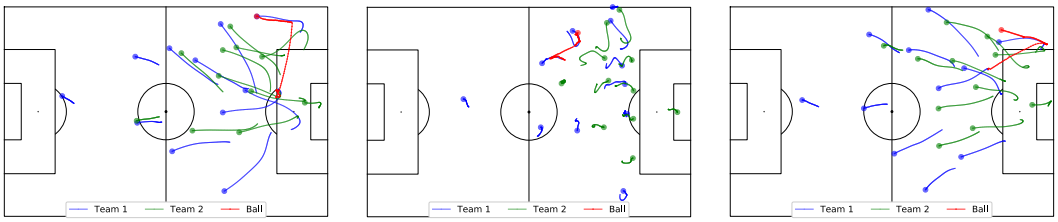
top-100 MIoU of 0.5 means that about 67 of the 100 closest scenes were also in the closest 100 scenes in the embedding.

6 EVALUATION

We first evaluate the quality of the retrievals by analyzing the scenes that our method returns (Sec. 6.1) and by analytically comparing the embedding neighborhood structure with ground-truth retrievals (Sec. 6.2). Next, we compare the retrieval speed of our approach against a naïve baseline and `Chalkboarding` (Sec. 6.3), and assess the system's generalization on a hold-out test dataset (Sec. 6.4). We conduct an expansive ablation study (Sec. 6.5) that performs a sensitivity analysis on important hyper-parameters (Sec. 6.5.1), ablates our proposed gating mechanism (Sec. 6.5.2) and learns distances for longer scene lengths (Sec. 6.5.3).

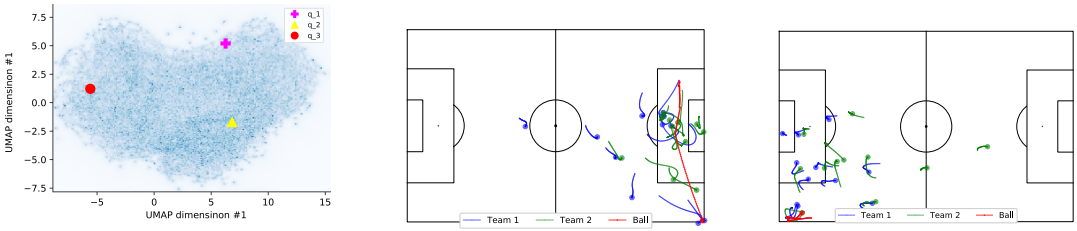
6.1 Scene Retrieval

We begin by showing exemplary retrieval queries and results for two different configurations of our method, i.e., $M = 2$ and $M = 64$. The query scene q_1 in Fig. 9a resembles a cross from the left flank. Its nearest neighbors (retrieved by our method) in Fig. 9b (for $M = 2$) and in Fig. 9c (for $M = 64$) make it very obvious that our method is capable of learning at very different levels of detail. In the query q_1 all players rush towards the defending team's goal, i.e., the scene is highly dynamic. In contrast, the query result for $M = 2$ shows a much more static game situation. The positional layout is also different: the query result for $M = 2$ is clinched relative to q_1 in the horizontal axis. The



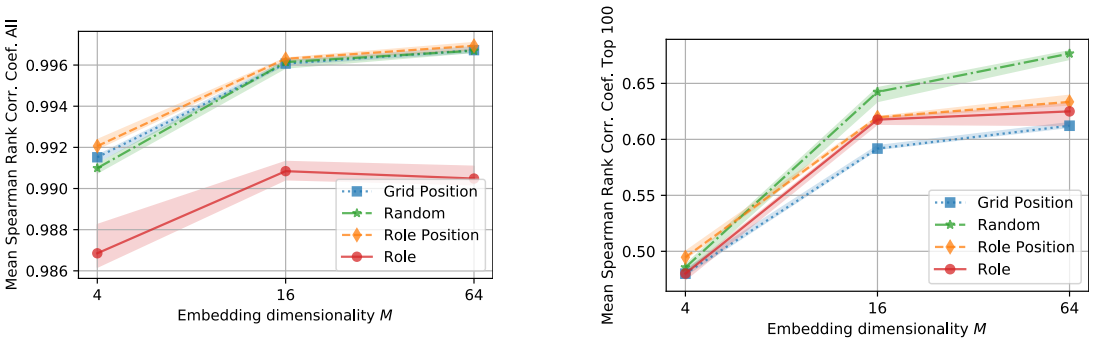
(a) Query scene q_1. (b) Neighbor of q_1 in $M = 2$. (c) Neighbor of q_1 in $M = 64$.

Fig. 9. Different scenes. The start of each trajectory is indicated by a circle. The attacking team is shown in blue, the defending in green, and the ball in red. Fig. 9a shows the query scene with a corner kick used for retrieval of the nearest neighbors in different embedding sizes M. Fig. 9b and 9c show the query results for q_1 retrieved via a 2- and 64-dimensional embedding, respectively.



(a) 2D UMAP of 64D embeddings. (b) Query scene q_2. (c) Query scene q_3.

Fig. 10. The 64D embedding neighborhood that was fitted to a 2D manifold seemingly shaped like a curved plane in Fig. 10a (using UMAP [22]), and we highlight the queries q_1, q_2 and q_3. The scenes q_2 in 10b and q_3 in 10c show vastly different plays compared to q_1, especially in location. This is also visible on the manifold.



(a) Top-$|\mathcal{M}_{sub}|$ MSRCC.

(b) Top-100 MSRCC.

Fig. 11. Left: MSRCC for the subset $\mathcal{M}_{sub}$ over all scenes; Right: MSRCC for the top-100 with increasing embedding dimensionality M for each channel assignment method.

ball trajectory is also misplaced. The learned embedding resembles an approximation of the center of mass of a scene as $M = 2$ does not allow for a rich representation of the scene. However, with an increased size of $M = 64$ the model learns a very exact and powerful representation of scenes. The resulting scene in Fig. 9c exhibits similar game dynamics, i.e., both teams rush towards the defending team's goal. This is a representative qualitative analysis which shows that our method can retrieve subjectively similar scenes if the embedding dimension is appropriately set to capture the actual information hidden in a scene.

Next, we analyze the learned embedding for $M = 64$ in order to get a clearer understanding of what the network has learned. To this end, we visualize the validation dataset as a 2D UMAP [22] heat map in Fig. 10a, and highlight the representation for q_1 and for two additional query scenes scenes, q_2 in Fig. 10b and q_3 in Fig. 10c. These scenes q_2 and q_3 show very different games and take place at different corners. We see their positional arrangement on the learned manifold (here fitted to 2D), that has the shape of a curved plane. We flatten out the manifold to resemble the game pitch and see that the queries q_1 and q_2 are relatively close to where their games take place. Similarly, the embedding of q_3 mirrors this, as it takes place in the opposite corner. Note that while the 2D UMAP simplifies the learned representations to their main, positional features, we saw in Fig. 9 that additional properties such as game dynamics are also modeled.

We saw that the 64D embedding contains additional learned features, like game dynamics, and perform better than the 2D embedding. However, Fig. 10a can only give intuitive insights. Hence, we provide further performance metrics and more in-depth analysis in the remainder of the evaluation.

6.2 Analysis of Embedding Neighborhood

We analyze the structure of the embedding space and measure how well our method preserves inter-sample distances and hence the ordering of the retrieval results. For this, we compare the original ordering of scenes with their embedded form on two different scales: on a fine scale, i.e., relative to the 100 nearest neighbors, and on a coarse scale, i.e., the position of a scene in the embedding relative to *all others* sampled from the $\mathcal{M}_{sub}$ subset. The fine scale is important for evaluating how well the ordering of the most relevant retrievals is maintained between the learned representations, i.e., it measures the precision of our approach which is an important metric for the task of scene retrieval. The coarse scale evaluates the general feasibility of the approach, e.g., whether it can be used as a method for clustering as pre-processor.

Fig. 11a shows the coarse structural score for the whole $\mathcal{M}_{sub}$, i.e., the top-$|\mathcal{M}_{sub}|$ score. In this experiment, `Role Position` shows the best performance, closely followed by `Grid Position` and `Random` assignments. `Role` is left far behind as it has a strong inductive bias due to role specific assignments that are suboptimal on more diverse scenes. Nevertheless, all assignment methods perform better than the coefficient of 0.986, and hence are suitable for clustering at least. Furthermore, all assignment methods show better performance with increasing embedding dimensionality M until $M = 16$, from where they settle into a plateau. Overall, all assignment methods for any embedding dimensionality, even for $M = 2$, show high MSRCC, indicating that the coarse structure is captured very well.

Fig. 11b shows the fine scale Spearman rank correlation coefficient averaged over the nearest 100 neighboring scenes. This directly influences the retrieval quality. Here, the ordering of `Random` assignments is best according to MSRCC, followed by `Role`, `Role Position` and `Grid Position`. Even though `Role Position` was marginally ahead for the overall ordering, its ordering of the nearest neighbors is worse than `Random` and `Role`. Again, all assignment methods improve with increasing M until $M = 16$, after which they plateau. In contrast to the overall ordering of scenes, the ordering of the nearest neighbors changes strongly with differing assignment methods and sizes of embedding dimensionality. Even with relatively high MAPE, the overall ordering is captured well (Fig. 11a), but when observing the ordering at a smaller scale the errors become visible (Fig. 11b). The better fine scale performance of `Role` compared to its coarse scale results can be explained by a greater benefit of the inductive bias of explicit role assignments for very similar scenes, in contrast to detrimental effects when scenes, and thus the formation of players, are further apart.

As the high MSRCC indicates, the overall ordering is preserved well (which enables clustering of the search space [32]), while the fine scale ranking is not perfectly preserved. To alleviate the implications of this sub-optimal top-100 MSRCC, we may instead evaluate the ground truth distance to rank these 100 nearest results better, or even train a specific ranking algorithm for top-100 similarly to [10].

We furthermore show the differences in MAPE between the larger validation set and the smaller $\mathcal{M}_{sub}$ in Fig. 12. Differences are only marginal for `Role Position` and `Grid Position`, which perform nearly identical on both datasets. However, both `Random` and `Role` exhibit a lower MAPE in the validation set. These two assignment methods do not generalize as well as methods that adapt the channel assignment to the actual positional layouts of trajectories in a scene. In larger datasets, players may often switch actual roles depending on the game's situation [3]. Models without positional channel information have no additional indication of the dynamic role each player fills in a scene.

The high divergence of the $\mathcal{M}_{sub}$ subset from the validation set indicates that `Random` and `Role` only rely on static player roles, which do not necessarily match well with dynamic roles in scenes, whereas the methods `Grid Position` and `Role Position` generalize better. We consider the small differences for $M \in [2, 4]$ to be of less importance, because models learn little to no dynamics as the previous experiments show. This overly simplistic representation results in a lower generalization error, but ultimately is not really usable either. Overall, `Role Position` performs strong in the fine and coarse settings, and also generalizes very well to larger problems.

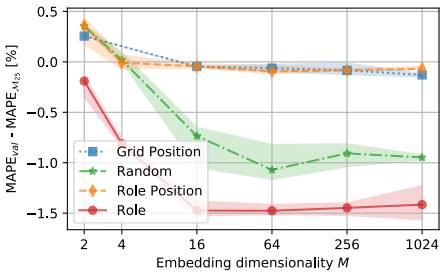


Fig. 12. Difference between the MAPE on the validation set and the subset $\mathcal{M}_{sub}$ for increasing embedding size M and channel assignment methods. Negative values indicate that the MAPE for $\mathcal{M}_{sub}$ was greater than for the validation set.

6.3 Retrieval Speed

Our main goal is to enable real-time scene search
on very large datasets of trajectory ensembles. Hence, we evaluate the required retrieval time for searches on 10K and 1M scenes and compare it with the previous state-of-the-art. For our experiments, we use scenes of 5 s duration, with 11 trajectories per team and both teams accounted for in the distance computation. The experiment is designed to be maximally fair, and we average over 100 repetitions. All database scenes are in memory, no clustering is performed to limit the search space, and the query is aligned to a template, i.e., both teams are aligned to their own template that contains 11 trajectories.

To measure the retrieval time, we compute the distance between query and each sample in the database. For our method, the retrieval speed is the time of one forward pass through the neural network, followed by the computation of the Euclidean distance from the query's embedding vector to other embedded scenes. For all methods, we search only for the 10 scenes with smallest distance, and do not sort the results.

We show the retrieval speed for varying sizes of embedding dimension in Tab. 2. As a `baseline` we implemented a naïve (brute force) distance computation and also include `Chalkboarding` [31] retrieval. With 10,000 scenes in the database, our method takes only about 10 ms for $M \leq 64$, while `Chalkboarding` takes 346ms and `Baseline` even takes 65 s, which is prohibitively long. At embedding sizes of $M \in [2, 4, 16, 64]$, searches are not limited by the distance function computations. With $M \in [256, 1024]$ it begins to affect retrieval times with 17 ms to 45 ms. With smaller M, the cost of computing forward passes through the network is nearly constant as only the last fully-connected layer varies with M. The largest part of the $9 - 11ms$ run time is data transfer overhead, e.g., GPU to CPU transfer and vice versa, while the distance computation time itself is minimal.

There are mainly two reasons why our method outperforms `Chalkboarding`. First, `Chalkboarding` expensively aligns queries to the learned template, and second, it computes the ranking based on 23 trajectories, hence one distance computation operates over a $23 \times 2 \times 125$ dimensional tensor in contrast to only M values in the embedding.

The larger experiment with 1M scenes corresponds to almost an entire season (305 games). As the data does not fit into main memory we could not compute results for `Baseline`, `Chalkboarding`, and our method with $M = 1024$. However, from the available experiments it is apparent that their

Method	Retrieval time 10k scenes	Retrieval time 1M scenes
`Baseline`	68.887s	> 1 h
`Chalkboarding`	0.346s	> 30 s
$M = 2$	0.009s	0.052s
$M = 4$	0.009s	0.054s
$M = 16$	0.010s	0.100s
$M = 64$	0.011s	0.287s
$M = 256$	0.017s	0.998s
$M = 1024$	0.045s	-

Table 2. Evaluation of the retrieval time using differently sized embeddings, the `Chalkboarding` [31] retrieval system and the baseline. All retrieval speed tests except for the baseline are averaged over 100 runs.

retrieval times are not competitive, i.e., >1 h and >30 s. Instead, our method yields an interactive retrieval time. Between $M = 16$ or $M = 64$ it offers an interesting trade-off. $M = 16$ requires only about 35% of the time of $M = 64$ but increases MAPE by about 18%. The lower error rate for $M = 64$ should be preferred if the resulting set of nearest neighbors is not further refined or ranked, i.e., similar to `Chalkboarding`'s two-step cluster-then-refine approach.

6.4 Generalization

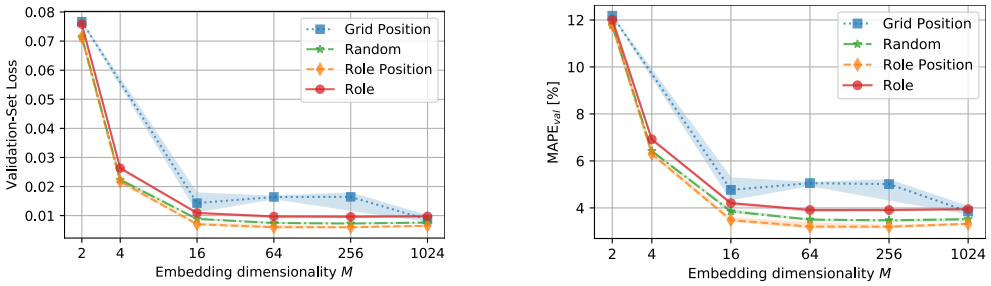
We also assess generalization on a hold-out dataset that contains 30 matches. We design our experiment as follows. We sample 1M pairs from this unseen test set and compute the MAPE between the pair-distances and their representation in the embedding. The difference between the MAPE computed on the test and validation set allows us to evaluate how our approach generalizes to unseen data.

We use the best performing channel assignment method `Role Position` and set the embedding size $M = 64$. This offers a speedy retrieval with relatively low error on the validation set with a MAPE of 2.66%. On the test set, this optimal model achieves a MAPE of 2.68%. This minimal generalization error of about 0.02% shows that our method learns a general representation which is not only a fit to the training data, but also works on previously unseen scenes.

6.5 Ablation Study

6.5.1 Sensitivity Analysis. We study the sensitivity of our approach to different sizes of the *embedding dimension* $M \in [2, 4, 16, 64, 256, 1024]$ and the channel assignment methods `Grid Position`, `Random`, `Role Position` and `Role` on the validation set, in order to measure their impact on the approximation error.

Fig. 13 shows the MSE to the left and the MAPE to the right. The experiments were performed the validation-set over three runs. Here, we report the evaluation for the network when the validation loss is minimal. The MSE on the left shows errors irrespective of the *location* of the ground-truth trajectory. However, a low MAPE does not necessarily follow from a low MSE. The MAPE is a measure of how relevant the scenes are ranked on average, by comparing the nearest neighbors in the embedding. `Role Position` shows best performance followed closely by `Random` and `Role`. We see low variability over repeated runs for each of the assignment methods. For all of them, performance increases rapidly until $M = 16$, and then plateaus from $M = 64$ to $M = 1024$. In contrast, the channel assignment by `Grid Position` shows slightly worse performance with a



(a) MSE over the validation set.

(b) MAPE over the validation set.

Fig. 13. Validation-set MSE loss (left) and MAPE (right) computed on predetermined set of pairs with increasing embedding dimensionality M for each channel assignment method. The points indicate the median over three runs. The colored areas correspond to the standard deviation centered around the mean. Note that the x-axis has a logarithmic scaling.

higher variability over repeated runs. However, with the larger capacity of $M = 1024$ it catches up to the other assignment methods (at the cost of compute time).

The decrease in error with increasing embedding dimension M is expected. With a larger M, more scene features can be embedded into the representation. Interestingly, the performance of most models improves only up to $M = 64$. We hypothesize that this is due to a lack of model capacity in the feature extractor. The maximum width of channels during the feature extraction is 128, the networks learn solely based on these features. Accordingly, embedding sizes greater than $M = 128$ use an under-determined projection matrix, i.e., some of the embedding axes linearly depend on each other and are redundant. Hence, scaling the width of the models could allow models with large M to improve further.

The channel assignment methods exhibit varying degrees of sparsity and of positional encoding. The evaluation results for `Grid Position`, `Role Position` and `Role` suggest that sparsity impacts performance negatively. We next examine the positive impact of our gating mechanism onto sparse data.

6.5.2 Ablating Gating Mechanism. In this work we introduced a novel gating mechanism for TCNs for sparse data to address the assignment problem. Here, we investigate the performance benefits in an ablation study. We show differences for sparse (channel assignment) inputs through the masking operation.

In Fig. 14 we show the differences of MAPE values between the full and the ablated TCN architecture without gating mechanism. The sparsest method `Role` shows clear performance degradation compared to the more dense methods `Grid Position`, `Random` and `Role Position`. The changes range within 0.3% MAPE, i.e., the expected worsening of performance due to missing information is more subtle. The absolute MAPE for `Random`, `Role Position` and `Grid Position` is slightly below 3% and `Role` around 3.75%. In essence, omitting the gating mechanism clearly leads to a decreased predictive performance, with the realistic real world configuration of $M = 64$ showing the largest difference.

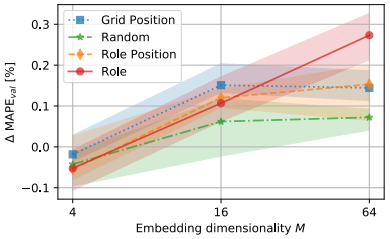


Fig. 14. ΔMAPE wo/ gating over different embedding dimensionalities $M \in [4, 16, 64]$.

Overall, the effect on all sparse channel assignment methods is larger than for the dense `Random` assignment, indicating that sparse assignment methods benefit from information in the masks. We explain the minimal change for `Random` due to the complete lack of additional information that masks encode for it, i.e., every mask value is one. Hence, the performance improvements can be partially accounted to the increased model capacity from additional mask channels. Interestingly, the degree of sparsity does not affect the effect size, as `Role` benefits to a larger extend than does `Grid Position`. To summarize, our gating mechanism improves performance in most (useful) scenarios, likely due to the information encoded through the masks and the increased model capacity, while the sparse channel assignment methods improve most.

6.5.3 Scene Length. Besides sport scenes of 5 s length for smaller scale analysis, sports scientists and scouts are also interested in larger strategic movements over longer periods of time. Generally, for other applications, the complexity of the data varies greatly, e.g., the amount of data per scene between bird foraging and urban transportation patterns. For these reasons we evaluate our learning method for longer scenes of 20 s length, which exhibit a larger feature complexity. In our experiment, we make use of all four assignment methods and use the previously best performing network architecture but extend the input width and receptive field size accordingly. We used a training set of 10 million scenes.

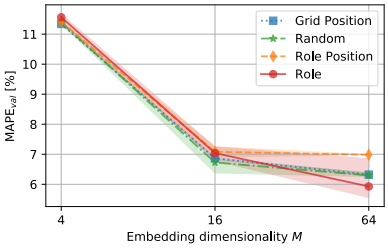


Fig. 15. MAPE for 20 s scene length.

The experimental results in Fig. 15 for $\mathcal{M} = 64$ show an increased absolute approximation error of a MAPE of 6.3% with `Random` assignments, and 5.8% for `Role Position`. This error increased compared to the previous 3.47% at 5 s scene length. With respect to the local neighborhood ranking for 20 s scene length, we find that the error did not increase similarly to MAPE, with very good performance of a top-$|\mathcal{M}_{sub}|$ MSRCC of still 0.97, and the fine granular neighborhood top-100 ranking MSRCC of 0.69. Also the top-100 ranking score MIoU is high (0.57). This means that the overall retrieval accuracy remains almost unchanged. For the longer scenes of 20 s duration, their pair-wise distances grow larger, and following from that, their neighborhood structure thins out, with neighbors farther apart from each other. Hence, the absolute approximation error MAPE has less impact on (local) ranking because absolute pairwise distances are also larger. This shows that our approach is especially well-suited for more complex data due to its typically larger distances and also due to the massive savings of computation time.

Moreover, using an estimate of the embedding density we could derive an optimal cluster size to restrict the search space in a principled way. Instead of directly using the embedding for scene ranking, computing the exact ground truth distance in a cluster is feasible. Here, MAPE serves as the expected radius in which most relevant scenes lie.

7 CONCLUSION

We proposed a novel approach to similarity-based scene retrieval of trajectory ensembles that uses approximations to the assignment problem at much lower computational costs than state of the art. Using Siamese Networks at its core we learn a low-dimensional representation that preserves the distance between sample pairs and thus accelerates the search by several orders of magnitude. The low approximation error allows fast search and subsequent ranking of the closest neighbors, while leaving the global neighborhood structure almost unchanged. We propose and evaluate four channel assignment methods, both application agnostic or biased for role-based (sports) trajectories, and found the hybrid `Role Position` to work best for the evaluated sports tracking application.

Our experimental results prove that our method learns non-trivial trajectory features like game play dynamics, and users can select an optimal trade-off between estimation accuracy and search speed, depending on the application. Furthermore, the proposed gating mechanism increases performance for sparse channel encoding.

In conclusion, our approach enables a highly accurate and interactive retrieval of similar scenes. A trivial extension with a two-step ranking system could additionally incorporate a refined second ranking step of the top-100 retrieved samples, using the exact distance computation on original sample representations. Furthermore, adapting our framework to applications in similar team sports like basketball or ice hockey is obvious, especially in datasets of unordered trajectory ensembles. The method may help the analysis of data from in-store tracking of customer movement by finding similar movement patterns. In the medical domain, learning the similarity of parameters of walking gait only from movement trajectories (specifying a human motion simulator) could benefit. In radio-frequency positioning, interference minimization in a changing environment is a hard problem, that could profit from accelerated search of historically similar channel/frequency and sender/receiver assignments.

ACKNOWLEDGMENTS

This work was supported by the Bavarian Ministry of Economic Affairs, Infrastructure, Energy and Technology as part of the Bavarian project Leistungszentrum Elektroniksysteme (LZE) and through the Center for Analytics-Data-Applications (ADA-Center) within the framework of "BAYERN DIGITAL II".

REFERENCES

[1] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. 2018. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. arXiv preprint arXiv:1803.01271 (2018).
[2] Alina Bialkowski, Patrick Lucey, Peter Carr, Yisong Yue, Sridha Sridharan, and Iain Matthews. 2014. Identifying team style in soccer using formations learned from spatiotemporal tracking data. In International Conference on Data Mining Workshop. IEEE, 9–14.
[3] Alina Bialkowski, Patrick Lucey, Peter Carr, Yisong Yue, Sridha Sridharan, and Iain Matthews. 2014. Large-scale analysis of soccer matches using spatiotemporal tracking data. In International Conference on Data Mining. IEEE, IEEE Computer Society, Los Alamitos, CA, USA, 725–730.
[4] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. 1994. Signature verification using a" siamese" time delay neural network. In Advances in neural information processing systems. 737–744.
[5] Nicolas Carion, F. Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. 2020. End-to-End Object Detection with Transformers. In European Conference on Computer Vision.
[6] Christopher Mutschler. 2010. Online Data–Mining of Interactive Trajectories in Realtime Location Systems. Master's thesis. Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU).
[7] Christopher Mutschler, Gabriella Kókai, and Thorsten Edelhäußer. 2011. Online Data Stream Mining on Interactive Trajectories in Soccer Games. In Proceedings of the 2nd International Conference on Positioning and Context-Awareness (Brussels). 15–22.
[8] Ian R Cleasby, Ewan D Wakefield, Barbara J Morrissey, Thomas W Bodey, Steven C Votier, Stuart Bearhop, and Keith C Hamer. 2019. Using time-series similarity measures to compare animal movement trajectories in ecology. Behavioral Ecology and Sociobiology 73, 11 (2019), 151.
[9] Tom Decroos, Jan Van Haaren, and Jesse Davis. 2018. Automatic discovery of tactics in spatio-temporal soccer match data. In Proceedings of the International Conference on Knowledge Discovery & Data Mining. 223–232.
[10] Mingyang Di, Diego Klabjan, Long Sha, and Patrick Lucey. 2018. Large-Scale Adversarial Sports Play Retrieval with Learning to Rank. ACM Transactions on Knowledge Discovery from Data (TKDD) 12, 6 (2018), 1–18.
[11] Esther Calvo Fernández, José Manuel Cordero, George Vouros, Nikos Pelekis, Theocharis Kravaris, Harris Georgiou, Georg Fuchs, Natalya Andrienko, Gennady Andrienko, Enrique Casado, et al. 2017. DART: a machine-learning approach to trajectory prediction and demand-capacity balancing. SESAR Innovation Days, Belgrade (2017), 28–30.
[12] X. Gao, X. Liu, T. Yang, G. Deng, H. Peng, Q. Zhang, H. Li, and J. Liu. 2020. Automatic Key Moment Extraction and Highlights Generation Based on Comprehensive Soccer Video Understanding. In International Conference on Multimedia Expo Workshops (ICMEW). IEEE, 1–6.
[13] Andreas Grunz, Daniel Memmert, and Jürgen Perl. 2012. Tactical pattern recognition in soccer games by means of special self-organizing maps. Human movement science 31, 2 (2012), 334–343.
[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In Proceedings of the Conference on Computer Vision and Pattern Recognition. 770–778.
[15] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. Neural computation 9, 8 (1997), 1735–1780.
[16] Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In 3rd International Conference on Learning Representations, San Diego, CA, USA.
[17] Georg Kohl, Kiwon Um, and Nils Thuerey. 2020. Learning Similarity Metrics for Numerical Simulations. In International Conference on Machine Learning. PMLR, 5349–5360.
[18] Harold W Kuhn. 1955. The Hungarian method for the assignment problem. Naval research logistics quarterly 2, 1-2 (1955), 83–97.
[19] Mengyuan Lee, Yuanhao Xiong, Guanding Yu, and Geoffrey Ye Li. 2018. Deep neural networks for linear sum assignment problems. IEEE Wireless Communications Letters 7, 6 (2018), 962–965.
[20] D. Link. 2014. A toolset for beach volleyball game analysis based on object tracking. International Journal of Computer Science in Sport 13 (01 2014), 24–35.
[21] Yingchi Mao, Haishi Zhong, Xianjian Xiao, and Xiaofang Li. 2017. A Segment-Based Trajectory Similarity Measure in the Urban Transportation Systems. Sensors 17, 3 (Mar 2017), 524.

[22] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Großberger. 2018. UMAP: Uniform Manifold Approximation and Projection. Journal of Open Source Software 3, 29 (2018), 861.
[23] Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In Proceedings of the 27th International Conference on International Conference on Machine Learning (Haifa, Israel) (ICML'10). 807–814.
[24] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Advances in Neural Information Processing Systems 32. Montréal, Canada, 8024–8035.
[25] Jürgen Perl. 2018. Formation-based modelling and simulation of success in soccer. International Journal of Computer Science in Sport 17, 2 (2018), 204–215.
[26] Jürgen Perl and Daniel Memmert. 2011. Net-Based Game Analysis by Means of the Software Tool SOCCER. International Journal of Computer Science in Sport (01 2011).
[27] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). 3973–3983.
[28] Keven Richly. 2018. Leveraging spatio-temporal soccer data to define a graphical query language for game recordings. In 2018 IEEE International Conference on Big Data (Big Data). IEEE, 3456–3463.
[29] Pavel Senin. 2008. Dynamic time warping algorithm review. Information and Computer Science Department University of Hawaii at Manoa Honolulu, USA 855, 1-23 (2008), 40.
[30] B. Serrien, M. Goossens, and J-P. Baeyens. 01 Jul. 2017. Issues in Using Self-Organizing Maps in Human Movement and Sport Science. International Journal of Computer Science in Sport 16, 1 (01 Jul. 2017), 1 – 17.
[31] Long Sha, Patrick Lucey, Yisong Yue, Peter Carr, Charlie Rohlf, and Iain Matthews. 2016. Chalkboarding: A new spatiotemporal query paradigm for sports play retrieval. In Proceedings of the 21st International Conference on Intelligent User Interfaces. 336–347.
[32] Long Sha, Patrick Lucey, Stephan Zheng, Taehwan Kim, Yisong Yue, and Sridha Sridharan. 2017. Fine-grained retrieval of sports plays using tree-based alignment of trajectories. arXiv preprint arXiv:1710.02255 (2017).
[33] Zhi-Hao Shen, W. Du, X. Zhao, and Jianhua Zou. 2019. Retrieving Similar Trajectories from Cellular Data at City Scale. ArXiv abs/1907.12371 (2019).
[34] Huong Yong Ting, Kok-Swee Sim, and Fazly Salleh Abas. 2015. Kinect-based badminton movement recognition and analysis system. International Journal of Computer Science in Sport 14, 2 (2015), 25–41.
[35] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. 2016. WaveNet: A Generative Model for Raw Audio. In 9th ISCA Speech Synthesis Workshop. 125–125.
[36] Aäron Van Den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. 2016. Pixel recurrent neural networks. In Proceedings of the 33rd International Conference on International Conference on Machine Learning-Volume 48. 1747–1756.
[37] Michail Vlachos, George Kollios, and Dimitrios Gunopulos. 2002. Discovering similar multidimensional trajectories. In Proceedings 18th international conference on data engineering. IEEE, 673–684.
[38] Zheng Wang, Cheng Long, Gao Cong, and Ce Ju. 2019. Effective and efficient sports play retrieval with deep representation learning. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 499–509.
[39] Sebastian Wenninger, Daniel Link, and Martin Lames. 01 Jul. 2020. Performance of machine learning models in application to beach volleyball data. International Journal of Computer Science in Sport 19, 1 (01 Jul. 2020), 24 – 36.
[40] Han Xiao. 2020. Hungarian layer: A novel interpretable neural layer for paraphrase identification. Neural Networks 131 (2020), 172–184.
[41] Munkh-Erdene Yadamjav, Zhifeng Bao, Baihua Zheng, Farhana M. Choudhury, and Hanan Samet. 2020. Querying Recurrent Convoys over Trajectory Data. ACM Trans. Intell. Syst. Technol. 11, 5, Article 59 (Aug. 2020), 24 pages.
[42] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. 2019. Free-form image inpainting with gated convolution. In Proceedings of the IEEE International Conference on Computer Vision. 4471–4480.
[43] Yu Zhao, Quan Chen, Wengang Cao, Jie Yang, Jian Xiong, and Guan Gui. 2019. Deep learning for risk detection and trajectory tracking at construction sites. IEEE Access 7 (2019), 30905–30912.
[44] Yu Zheng. 2015. Trajectory data mining: an overview. ACM Transactions on Intelligent Systems and Technology (TIST) 6, 3 (2015), 1–41.

B IALE: Imitating Active Learner Ensembles

Journal of Machine Learning Research 23 (2022) 1-29 Submitted 4/21; Revised 12/21; Published 2/22

IALE: Imitating Active Learner Ensembles

Christoffer Löffler CHRISTOFFER.LOEFFLER@FAU.DE
Machine Learning and Data Analytics Lab
Friedrich-Alexander University Erlangen-Nürnberg (FAU)
Carl-Thiersch-Straße 2b, 91052, Erlangen, Germany

Christopher Mutschler CHRISTOPHER.MUTSCHLER@IIS.FRAUNHOFER.DE
Fraunhofer IIS
Fraunhofer Institute for Integrated Circuits IIS
Nuremberg, Germany

Editor: Andreas Krause

Abstract

Active learning prioritizes the labeling of the most informative data samples. However, the performance of active learning heuristics depends on both the structure of the underlying model architecture and the data. We propose IALE[1], an imitation learning scheme that imitates the selection of the best-performing expert heuristic at each stage of the learning cycle in a batch-mode pool-based setting. We use DAgger to train a transferable policy on a dataset and later apply it to different datasets and deep classifier architectures. The policy reflects on the best choices from multiple expert heuristics given the current state of the active learning process, and learns to select samples in a complementary way that unifies the expert strategies. Our experiments on well-known image datasets show that we outperform state of the art imitation learners and heuristics.

Keywords: active learning, deep neural networks, imitation learning, dataset aggregation, transferable policy

1. Introduction

The high performance of deep learning on various tasks from computer vision (Voulodimos et al., 2018) to natural language processing (NLP) (Barrault et al., 2019) also comes with a few disadvantages. One of the major drawbacks is the large amount of labeled training data they require. Obtaining such data is expensive and time-consuming and often requires domain expertise (Löffler et al., 2020).

Active Learning (AL) is an iterative process where during every iteration an oracle (e.g., a human) is asked to label the most informative unlabeled data sample(s). In *pool-based* AL all data samples are available (while most of them are unlabeled). In *batch-mode* pool-based AL, we select unlabeled data samples from the pool in acquisition batches greater than 1. Batch-mode AL decreases the number of AL iterations required and makes it easier for an oracle to label the data samples (Settles, 2009). As a selection criteria we usually need to quantify how informative a label for a particular sample is. Well-known criteria include heuristics such as model uncertainty (Gal et al., 2017; Roth and Small, 2006; Wang

1. IALE is pronounced /eɪl/.

and Shang, 2014; Ash et al., 2020), data diversity (Sener and Savarese, 2018), query-by-committee (Beluch et al., 2018), and expected model change (Settles et al., 2008). As ideally we label the most informative data samples at each iteration, the performance of a machine learning model trained on a labeled subset of the available data selected by an AL strategy is better than that of a model that is trained on a randomly sampled subset of the data.

Besides the above mentioned, in the recent past several other data-driven AL approaches emerged. Some are modelling the data distributions (Mahapatra et al., 2018; Sinha et al., 2019; Tonnaer, 2017; Hossain et al., 2018) as a pre-processing step, or similarly use metric-based meta-learning (Ravi and Larochelle, 2018; Contardo et al., 2017) as a clustering algorithm. Others focus on the heuristics and predict the best suitable one using a multi-armed bandits approach (Hsu and Lin, 2015). Recent approaches that use reinforcement learning (RL) directly learn strategies from data (Woodward and Finn, 2016; Bachman et al., 2017; Fang et al., 2017). Instead of pre-processing data or dealing with the selection of a suitable heuristic they aim to learn an optimal selection sequence on a given task.

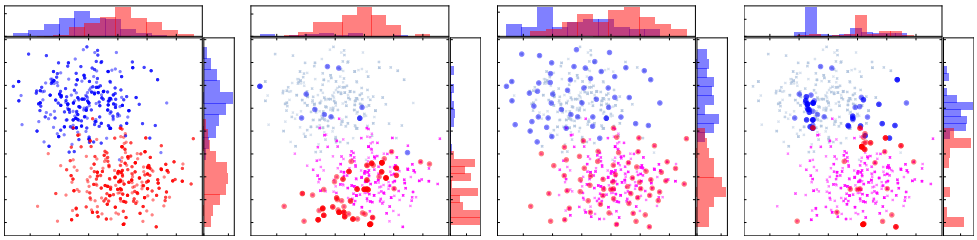
However, the RL approaches not only require a huge amount of samples they also do not resort to existing knowledge, such as potentially available AL heuristics. Moreover, training the RL agents is usually time-consuming as they are trained from scratch. Hence, when only few labeled training data and a potent algorithmic expert are available imitation learning (IL) helps. IL trains, i.e., *clones*, a policy to transfer the expert to the related few data problem. While IL mitigates some of the aforementioned issues, previous approaches are still limited (including that of Liu et al. (2018)), e.g., by their limited expressiveness of the *state* representations, their computational efficiencies, their non-arbitrary acquisition sizes, and their lack of complementary experts. They were also so far only evaluated on NLP tasks.

We propose `IALE`, that is based on imitation learning and that makes use of a diverse set of experts from different heuristic families, i.e., uncertainty, diversity, expected model-change, and query-by-committee, in a batch-mode AL setting with *arbitrary* acquisition sizes. Our policy extends previous work (see Section 2) by learning at which stage of the AL cycle which of the available strategies performs best, based on a more expressive state, that allows a powerful introspective view into the classifier model to better assess its confidence. We use Dataset Aggregation (DAgger) to train a robust and transferable policy and apply it to other problems from similar domains (see Section 3). We show that we can (1) train a policy on image datasets such as MNIST, Fashion-MNIST, Kuzushiji-MNIST, Extended MNIST, CIFAR and SVHN, (2) transfer the policy between them, and (3) even transfer the policy between different classifier architectures (see Section 4).

2. Related Work

Next to the AL approaches for traditional ML models (Settles, 2009) also ones applicable to deep learning have been proposed (Gal et al., 2017; Sener and Savarese, 2018; Beluch et al., 2018; Settles et al., 2008; Ash et al., 2020). Below we discuss AL strategies that are trained on data.

Generative Models. Explicitly modeled data distributions capture the *informativeness* that can be used to select samples based on diversity. VAAL (Sinha et al., 2019) is a pool-based semi-supervised AL method, where a discriminator discriminates between labeled and unlabeled samples using the latent representations of a variational autoencoder. The



(a) Data distribution (b) *Entropy* (c) *CoreSet* (d) *Our method*

Figure 1: Active learning on two normal distributions in a 2D feature space. Fig. (1a) shows data and histograms. The remaining plots show labelled and unlabelled data, and histograms of labelled samples for different acquisition strategies. We show the state after training an MLP (two layers with 16 units and ReLU activation) via AL (20 initially labelled samples, acquiring 10 labels per iteration, 10 times). We present (b) *Entropy* sampling, (c) *CoreSet* sampling, and (d) `IALE`, that imitates the other two, see Section. 3 for details.

representations are used to pick the most diverse and representative data points (Tonnaer, 2017). Mirza and Osindero (2014) use a conditional generative adversarial network to generate samples with different characteristics from which the most informative are selected using the uncertainty measured by a Bayesian neural network (Kendall and Gal, 2017; Mahapatra et al., 2018). Such approaches are similar to ours (as they capture dataset properties) but instead we model the dataset implicitly and infer a selection heuristic via imitation.

Metric Learning. Metric learners such as the one proposed by Ravi and Larochelle (2018) use a set of statistics calculated from the clusters of un-/labeled samples in a Prototypical Network's (Snell et al., 2017) embedding space, or learn to rank (Li et al., 2020) large batches. Such statistics use distances (e.g., Euclidean distance) or are otherwise converted into class probabilities. Two MLPs predict either a quality or diversity query selection using backpropagation and the REINFORCE gradient (Mnih and Rezende, 2016). While they rely on statistics over the classifier's embedding and explicitly learn two strategies (quality and diversity) we use a richer state and are not constrained to specific strategies.

Meta learning. A similar field is *learning-to-learn*. Especially optimization-based meta learning, that aims to improve or discover learning algorithms (Hochreiter et al., 2001), potentially leads to alternative and fast converging learning algorithms for deep learning in few data (or few-shot) settings. Methods such as the LSTM Meta Learner (Ravi and Larochelle, 2017), which uses an LSTM to predict a network's parameter updates, the recurrent neural network-based approach by Chen et al. (2017), which learns to optimize black-box functions within a fixed horizon, and model-agnostic meta-learning (MAML) (Finn et al., 2017), which produces a well performing parameter initialization for the (task-specific) differentiating fine-tuning stage, are exemplary approaches. However, not only are these methods often computationally expensive to train, e.g., due to MAML's meta-gradient updates (Nichol et al., 2018). They also to tend to overfit on hard tasks (Jamal and Qi, 2019) or even fail to converge for ambiguous small tasks (Finn et al., 2018). Moreover, we do not require back-propagation through time (as Ravi and Larochelle (2017) do) with its

limiting time horizon (Mishra et al., 2018; Chen et al., 2017), but instead rely on gradient descent in combination with a high capacity state for our learned policy. This allows us to generalize to nearly arbitrary horizons including larger ones than seen during policy training.

Reinforcement Learning (RL). The AL cycle can be modeled as a sequential decision making problem. Woodward and Finn (2016) propose a stream-based AL agent based on memory-augmented neural networks where an LSTM-based agent learns to decide whether to predict a class label or to query the oracle. Matching Networks (Bachman et al., 2017) extensions allow for pool-based AL. Fang et al. (2017) use Deep Q-Learning in a stream-based AL scenario for sentence segmentation. In contrast to them we consider batch-mode AL with acquisition sizes ≥ 1 and work on a pool-setting instead of a stream-setting. While Bachman et al. (2017) propose a strategy to extend the RL-based approaches to a pool setting, they still do not work on batches. Instead, we allow batches of arbitrary acquisition sizes. Konyushkova et al. (2017) formulate AL as a regression task for a greedy label acquisition, that predicts the expected reduction of the classification error for each sample, and that is trained on either synthetic or real data. They use a Random Forest classifier, with features like predicted probability and forest variance, for binary classification and batch-sizes of one. Follow-up work (Konyushkova et al., 2018) replaces the greedy approach with a Q-Learning-based RL agent. Casanova et al. (2020) propose a DQN-based extension of Konyushkova et al. (2018)'s method, that learns to sample image regions for a semantic image segmentation task, focusing on classes that are underrepresented in the training dataset. Their work is specifically aimed at selecting relevant regions in images to optimize the classes' mean intersection of union. Our work focuses on different problems for learning AL, as we investigate useful sample relevance features for AL especially from deep neural networks, learn a unified AL heuristic from existing experts, and investigate the transfer of the policy between real datasets of multi-class problems with larger batch-sizes, on more complex datasets and transfers between classifier architectures. Fan et al. (2018) propose a meta-learning approach that trains a student-teacher pair via RL. The teacher optimizes *data teaching* by selecting labeled samples that let the student learn faster. In contrast, our method learns to selects samples from an unlabeled pool, i.e., in a missing target scenario. The teacher-student analogy is similar to our approach, however, the objective, method and available (meta-)data to learn a good teacher (policy) are considerably different.

Multi-armed Bandit (MAB). Baram et al. (2004) treat the online selection of AL heuristics from an ensemble as the choice in a multi-armed bandit problem. COMB uses the known EXP4 algorithm to solve it, and ranks AL heuristics according to a semi-supervised maximum entropy criterion (Classification Entropy Maximization) over the samples in the pool. Building on this, Hsu and Lin (2015) learn to select an AL strategy for an SVM-classifier and use importance-weighted accuracy extension to EXP4 that better estimates the performance of each AL heuristic improvement as an unbiased estimator for the test accuracy. Furthermore, they reformulate the MAB setting so that the heuristics are treated as the bandits where the algorithm selects the one with the largest performance improvement (in contrast to COMB's formulation where the unlabeled samples are treated as bandits). Chu and Lin (2016) extend Hsu and Lin (2015) to a setting where the selection of AL heuristics is based on a linear weighting, aggregating *experience* over multiple datasets. They adapt the semi-supervised reward scheme from Hsu and Lin (2015) to work with their deterministic queries. Instead of selecting from a set of available heuristics, we propose the learning of a

unified AL policy. This allows our policy model to learn an *interpolation* between batches of samples proposed by single heuristics also exploiting the deep network classifier's internal state.

Imitation Learning (IL). Imitation learning methods such as DAgger (Ross et al., 2011) can also be used to train an AL policy. For instance, Liu et al. (2018) propose a follow-the-leader approach that selects samples that improve classifier accuracy. During policy training they *roll out* a few possible acquisitions (using a small random pool subset) and retrain the classifier on each sample independently to infer *preference scores*. However, this not only leads to sub-optimal selections, it also requires an expensive re-training per sample in the roll-out. In contrast to them, we explore an alternative way for using IL, as `IALE` imitates an ensemble of a wide variety of AL heuristics to learn a unified AL strategy. Our *state* consists of novel features for *introspection*, such as gradients inferred from proxy-labels, as similarily proposed by Ash et al. (2020). Hence, `IALE` is well suited for deep learning (efficient inference, fast convergence), even compared to classical AL baselines.

3. IALE: Imitating Active Learner Ensembles

`IALE` learns an AL sampling strategy from *multiple experts* in a *pool-based* setting by imitating their behavior. We train a policy with data consisting of states (that encode, e.g., labeled and unlabeled sample distributions, uncertainty, and gradient signals) and best expert actions (i.e., samples selected for labeling) collected over the AL cycles. Hence, our policy learns with options, where each expert's (potentially sub-optimal) selection is an option it may choose to learn, according to their rank. Analogously, our approach is similar to a distillation of the experts. The policy is then applied on a different task. To discover states that are unlikely to be produced by the experts, DAgger (Ross et al., 2011) balances exploration (via the current policy) and exploitation (via the AL experts) to collect a large set of states and actions. We train the policy network on all the previous states and actions after each AL iteration.

3.1 Background

In pool-based AL we train a model M on a dataset $\mathcal{D}$ by iteratively labeling data samples. Initially, M is trained on a small amount of labeled data $\mathcal{D}_{\text{lab}}$ randomly sampled from the dataset. The rest of the data is considered as the unlabeled data pool $\mathcal{D}_{\text{pool}}$, i.e., $\mathcal{D} = \mathcal{D}_{\text{lab}} \cup \mathcal{D}_{\text{pool}}$. From that point onwards during the AL iterations a subset of $\mathcal{D}_{\text{sel}}$ is selected from $\mathcal{D}_{\text{pool}}$ by using an acquisition function $a(M, \mathcal{D}_{\text{pool}})$. The data is labeled and then removed from $\mathcal{D}_{\text{pool}}$ and added to $\mathcal{D}_{\text{lab}}$. The size of $\mathcal{D}_{\text{sel}}$ is based on the acquisition size acq (>1 for batch-mode AL). The AL cycle continues until a labeling budget of $\mathcal{B}$ is reached. M is retrained after each acquisition to evaluate the performance boost with respect to the increased labeled dataset only (and not the additional training time).

The acquisition function a uses heuristics on the trained model M to select the most informative data samples from $\mathcal{D}_{\text{pool}}$. For deep AL those include uncertainty-based *MC-Dropout* (Gal et al., 2017), query-by-committee-based *Ensembles* (Beluch et al., 2018), data diversity-based *CoreSet* (Sener and Savarese, 2018), gradient-based *BADGE* (Ash et al., 2020), and soft-max-based *Confidence-* or *Entropy*-sampling (Wang and Shang, 2014).

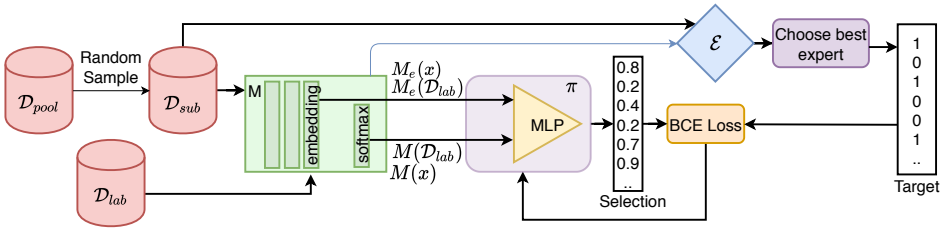


Figure 2: Training π to imitate experts $\mathcal{E}$: (1) we pass samples from $\mathcal{D}_{\text{sub}}$ and $\mathcal{D}_{\text{lab}}$ through the current classifier M; (2) the embeddings and the predictions are input to π, whose output vector is compared with the target vector predicted by the best expert; (3) we calculate a loss and back-propagate the error through π; (4) we extend the labeled pool data $\mathcal{D}_{\text{lab}}$ by $\mathcal{D}_{\text{sel}}$ and retrain M.

MC-Dropout uses a Monte-Carlo inference scheme based on a *dropout* layer to approximate the model's predictive uncertainty (Gal and Ghahramani, 2016) and then uses these values to select the most uncertain samples (Gal et al., 2017). *Ensembles* (Beluch et al., 2018) model predictive uncertainty using a committee of N classifiers, initialized with different random seeds. However, while at inference time we need to run only N forward-passes per sample (compared to *MC-Dropout* performing two dozens or more Monte-Carlo passes), the training of N-1 additional deep models can become prohibitively expensive in many use-cases. *CoreSet* (Sener and Savarese, 2018) aims to select diverse samples by solving the k-centers problem on the classifier's embeddings. This involves minimizing the distance between each of the unlabeled data samples to its nearest labeled samples. *BADGE* determines the magnitudes of the gradients in a batch using proxy-labels and selects samples by uncertainty and diversity. Soft-max-based heuristics (*Confidence*- and *Entropy*-sampling) use predictive uncertainty and are computationally lightweight at lower AL performance (Gal and Ghahramani, 2016; Ash et al., 2020). *Confidence* selects the samples with the lowest class probability and *Entropy* the ones with largest entropy of their probability distribution.

3.2 Learning Multiple Experts

Instead of using specific heuristics we propose to learn the acquisition function using a policy network. Once the policy is trained on a source dataset, it has learned a unified active learning heuristic, and can be applied to different target datasets.

Figure 1 illustrates the approach with two-dimensional Gaussian distributions. After ten acquisitions of ten new samples each strategy has sampled a distinct set of labeled data. While uncertainty-based *Entropy* mostly samples from only one class, the diversity-based *CoreSet* covers a more representative set over the data and selects data from the whole distribution. Our approach was trained to imitate both methods. Figure 1d shows that it acquired diverse samples (in this example at a ratio of 60 to 40), and samples especially along the decision boundary, reaching higher accuracy than any of the single baselines alone. The learned strategy hence combines the advantages of both imitated methods.

Figure 2 sketches the imitation learning framework. The policy network π is a Multi-Layer Perceptron (MLP) trained to predict the *usefulness* of labeling samples from the

unlabeled data pool $\mathcal{D}_{\text{pool}}$ for training the model M, similar to an AL acquisition function. As input the policy network takes the current *state*, that encodes, e.g., M's information on learned representations $M_e(x)$ and $M_e(\mathcal{D}_{\text{lab}})$, as well as its predictive uncertainty derived from $M(x)$, $M(\mathcal{D}_{\text{lab}})$, and label information from $\mathcal{D}_{\text{lab}}$. We use the predictions $M(x)$ to enrich the state with *pseudo-label* gradient signals to guide the policy's decisions later on[2]. Our state representation significantly extends previous work (Contardo et al., 2017; Konyushkova et al., 2017; Liu et al., 2018; Casanova et al., 2020) and adds novel features that allow for model introspection. The policy π then outputs the *action* to be taken at each step. *Action* here refers to an AL acquisition, i.e., which of the unlabeled data samples should be labeled and added to the training data. π learns the best *actions* from a set of experts $\mathcal{E}$ which predict the best actions for a given AL state, and thus learns its own *complementary* strategy. A subset of the pool dataset $\mathcal{D}_{\text{sub}}$ with size n is used instead of the whole pool dataset at each active learning iteration for training the policy.

States. As π uses the state information to make decisions, a state s should be maximally compact but still unique, i.e., different *situations* should have a different state encoding, and they have to allow to predict M's expected improvement. Our state encoding uses three types of information: (1) M's learned representations, (2) M's predictive uncertainty, and (3) M's gradient signals for unlabeled samples. We distinguish the state's parameters as either derived from the labeled or from the unlabeled pool. Together, these parameters form a minimal but comprehensive description of a model's state at each step of the AL-cycle.

For **labeled samples** the following parameters encode M's *learned representations* and *predictive uncertainty*:

- The embedded samples' mean $\mu\big(M_e(\mathcal{D}_{\text{lab}})\big)$: the embedding M_e of a sample by M is the output of the final layer (i.e., the layer before the soft-max layer in case of classification), see Figure 2. The size of this representation is independent of the (growing) size of $\mathcal{D}_{\text{lab}}$ and thus will not become a computational bottle-neck.
- The ground-truth empirical distribution of class labels

$$\vec{e}_{\mathcal{D}_{\text{lab}}} = \left(\frac{\sum_{y \in \mathcal{D}_{\text{lab}}} 1[y==0]}{|\mathcal{D}_{\text{lab}}|}, .., \frac{\sum_{y \in \mathcal{D}_{\text{lab}}} 1[y==i]}{|\mathcal{D}_{\text{lab}}|}\right),$$

  which is a normalized vector of length i, i.e., the number of classes, with percentage of occurrence per class using the labels of the already acquired data samples.
- M's predicted distribution of class labels for the labeled data $\vec{e}_{M(\mathcal{D}_{\text{lab}})}$, i.e., a normalized vector as before but with predicted class labels instead of ground-truth.

We encode the predictive uncertainty by including both the ground truth and the predicted empirical distribution. The policy can base its decisions on the model M's prediction errors, e.g., by detecting wrong predictions of already labeled samples and decide to acquire more similar samples.

For **unlabeled samples** (n data samples in $\mathcal{D}_{\text{sub}}$), we encode the information that is necessary to help π acquire more relevant samples. First, we calculate M's representation for each data sample $x_i \in \mathcal{D}_{\text{sub}}$, i.e., its embedding $M_e(x_i)$ in the same embedding space as

2. see Eq. 1 for an exact definition of the state.

the already labeled samples. Second, we also predict each sample's label $M(x_i)$. Finally, we encode gradient signals as the gradients of unlabeled data provide a powerful view due to its effect on the classifier model (as they encode M's expected change directed towards the steepest learning steps). Although their calculation usually requires labeled samples we can still approximate them using proxy-labels (Ash et al., 2020). We define a proxy-label $\hat{y}$ (i.e., the one that has the highest class probability for $M(x_i)$). Then, the gradient's magnitude and direction at the embedding layer describes the model's uncertainty and its expected change. We thus capture gradient information at the embedding layer as $g(M_e(x_i))$ and encode it as part of the state.

In summary, the state enables the policy to learn to select samples (1) where the model is uncertain (i.e., where it predicts the wrong labels), (2) where the model might gain most information (i.e., the gradient's magnitude is large), and (3) to learn to select samples that increase the labeled pool's diversity (i.e., to acquire less well-represented samples, using the label statistics and the learned representations). Hence, we describe a state s as follows:

$$s := \left[\mu(M_e(\mathcal{D}_{\text{lab}})), \vec{e}_{\mathcal{D}_{\text{lab}}}, \vec{e}_{M(\mathcal{D}_{\text{lab}})}, \begin{bmatrix} M_e(x_0) \\ \vdots \\ M_e(x_n) \end{bmatrix}, \begin{bmatrix} M(x_0) \\ \vdots \\ M(x_n) \end{bmatrix}, \begin{bmatrix} g(M_e(x_0)) \\ \vdots \\ g(M_e(x_n)) \end{bmatrix}\right] \tag{1}$$

Actions. In our approach, actions are essentially the resulting selections from acquisition functions, that π learns to imitate. The ground truth actions (selections) provided by the experts are binary vectors of length k, where a 1 at index i means that x_i should be selected for labeling. We may think of the experts are policies themselves that only have a fixed acquisition function. `IALE`'s acquisition function, on the other hand, uses its neural network's prediction to select samples. The output of the MLP is analogously a vector with a *desirability score* ρ_i for each unlabeled sample from $\mathcal{D}_{\text{sub}}$, i.e., $\rho_i := \pi(s_i)$, from which we choose the highest ranked samples. This results in a binary selection vector $\vec{v} = (\rho_0, \cdots, \rho_n)$ with $\sum_{i=0}^{n} \vec{v}_i = \text{acq}$. We use a *binary cross entropy loss* to update π's weights:

$$\mathcal{L}(\rho, \vec{t}) = -\sum_{i=0}^{n} \vec{t}_i \log(\rho_i) - (1 - \vec{t}_i) \log(1 - \rho_i), \tag{2}$$

where $\vec{t}$ is the target vector provided by the *best* expert (similar to a greedy multi-armed bandit approach (Hsu and Lin, 2015)), guiding π towards the best expert's suggestion.

Our IL-based approach uses the experts to turn AL into a supervised learning problem, i.e., the action of the best expert becomes the label for the current state s. From all the experts $\mathcal{E}$ we determine the best one by letting all of them select samples for labeling, and then rank their performance using temporary models trained for each expert. Our choice of AL heuristics for the set of experts $\mathcal{E}$ includes particular types but is arbitrarily extendable. Using *MC-Dropout*, *Ensemble*, *CoreSet*, *BADGE*, *Confidence* or *Entropy* allows us to only minimally modify the classifier model M, e.g., we add dropout at inference to use *MC-Dropout*. π aims to learn certain derived properties from s, such as model uncertainty.

Our hypothesis is that π imitates the best suitable heuristic for each phase of the AL cycle, i.e., starting with relying on one type of heuristics for selections of samples in the beginning and later using a different one for *fine-tuning* M. (see also Section 4.3). This is in

Algorithm 1 Imitating Active Learner Ensembles

1: data $\mathcal{D}$, labeled validation data $\mathcal{D}_{\text{val}}$, classifier M, budget $\mathcal{B}$, experts $\mathcal{E}$, acquisition size acq, subset size n, probability p, states $\mathcal{S}$, actions $\mathcal{A}$, random policy π (acq ≥ 1, $n = 100$).
2: **for** e $= 1 \ldots$ episodes$_{\max}$ **do**
3: $\quad \mathcal{D}_{\text{lab}}, \mathcal{D}_{\text{pool}} \leftarrow \text{split}(\mathcal{D})$
4: $\quad$ **repeat**
5: $\quad\quad M \leftarrow \text{initAndTrain}(M, \mathcal{D}_{\text{lab}})$
6: $\quad\quad \mathcal{D}_{\text{sub}} \leftarrow \text{sample}(\mathcal{D}_{\text{pool}}, n)$
7: $\quad\quad e^* \leftarrow \text{bestExpert}(\mathcal{E}, M, \mathcal{D}_{\text{sub}}, \mathcal{D}_{\text{val}})$
8: $\quad\quad \mathcal{D}_{\text{sel}} \leftarrow e^*.\text{SelectQuery}(M, \mathcal{D}_{\text{sub}}, \text{acq})$
9: $\quad\quad \mathcal{S}, \mathcal{A} \leftarrow \text{toState}(M, \mathcal{D}_{\text{sub}}, \mathcal{D}_{\text{lab}}), \text{toAction}(\mathcal{D}_{\text{sel}})$
10: $\quad\quad$ **if** $\text{Rnd}(0, 1) \geq p$ **then**
11: $\quad\quad\quad$ // We may choose π's selection
12: $\quad\quad\quad \mathcal{D}_{\text{sel}} \leftarrow \pi.\text{SelectQuery}(M, \mathcal{D}_{\text{sub}}, \text{acq})$
13: $\quad\quad$ **end if**
14: $\quad\quad \mathcal{D}_{\text{lab}} \leftarrow \mathcal{D}_{\text{lab}} \cup \mathcal{D}_{\text{sel}}$
15: $\quad\quad \mathcal{D}_{\text{pool}} \leftarrow \mathcal{D}_{\text{pool}} \setminus \mathcal{D}_{\text{sel}}$
16: $\quad\quad$ Update policy using $\{\mathcal{S}, \mathcal{A}\}$
17: $\quad$ **until** $|\mathcal{D}_{\text{lab}}| > \mathcal{B}$
18: **end for**

line with previous research that combines uncertainty- and density-based heuristics and that learns an adaptive combination framework that weights them over the training course (Li and Guo, 2013). π learns a more suitable selection for the classifier's learning stage through introspection into the classifier's state. Note that this is more adaptive to new problems than e.g. encoding time directly (for instance as a function of the number of acquisitions).

3.3 Policy Training

Our policy training builds on the intuition behind DAgger, which is a well-known algorithm for IL that aims to train a policy by iteratively growing a dataset for supervised learning. The key idea is that the dataset includes the states that are likely to be visited over the course of solving a problem (in other words, those state and action encodings that would have been visited if we would follow a hard-coded AL strategy). To this end, it is common when using DAgger to determine a policy's next state by either *following* the current policy or an available expert (Ross et al., 2011). We thus grow a list of state and action pairs, and randomly either choose expert or policy selections as the action.

Each episode of the IL cycle lasts until the AL labeling budget is reached for $episodes_{max}$ iterations. We aggregate the *states* and *actions* over all episodes, and continually train the policy on the pairs. We use DAgger to further randomize the exploration of $\mathcal{D}$. Instead of always following the best expert's advice, we randomly follow the policy's prediction, and thus enrich the possible states.

Our IL approach for training π is given in Algorithm 1. At each AL cycle, we randomly sample a subset $\mathcal{D}_{\text{sub}}$ of n samples from the unlabeled pool $\mathcal{D}_{\text{pool}}$ (line 3). We find the best

expert e^* from a set of experts $\mathcal{E}$ (line 7) by extending the training dataset by the expert selections (from $\mathcal{D}_{\text{sub}}$) and train a classifier each. This means that each expert constructs one batch according to its heuristic, e.g., a batch composition could maximize model-change, and queries the oracle for labels. We choose the best expert by comparing the resulting classifiers' accuracies on the labeled validation dataset. We next set its acquisition as this iteration's chosen target and store *state* and *action* for the policy training (line 9). Depending on the probability p (line 10) we then either use the policy or the best expert to increase $\mathcal{D}_{\text{lab}}$ for the next iteration (line 14). After each episode we retrain π on the *state* and *action* pairs (line 16).

4. Experiments

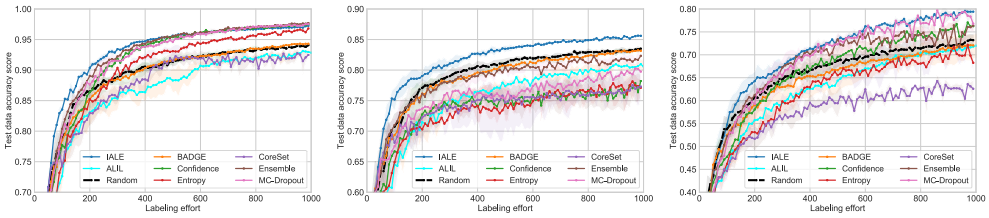
We first describe our experimental setup (Section 4.1). Next, we describe how we trained our policy on MNIST (Section 4.2) and evaluate our approach by transferring it to test datasets, i.e., to FMNIST and KMNIST (Section 4.3), and Extended MNIST, SVHN and CIFAR-10/-100 (Section 4.4). We end with a discussion of ablation studies and the limitations of our approach (Section 4.5). The source code is available at `https://github.com/crispchris/IALE` and can be used to reproduce our experimental results.

4.1 Experimental Setup

Datasets. We use the image classification datasets MNIST (LeCun et al., 1998), Fashion-MNIST (FMNIST) (Xiao et al., 2017), Kuzushiji-MNIST (KMNIST) (Clanuwat et al., 2018), Extended MNIST (Cohen et al., 2017), CIFAR-10/-100 (Krizhevsky, 2009), and SVHN (Netzer et al., 2011) for our evaluation. The MNIST-variants consist of $70,000$ grey-scale images (28×28px) in total for 10 classes. MNIST contains the handwritten digits $0-9$, FMNIST contains images of clothing (i.e., bags, shoes, etc.), and KMNIST consists of Hiragana characters. Extended MNIST contains, among others, a 26-class split of handwritten letters (28×28px) with 145,600 samples. SVHN, CIFAR-10 and CIFAR-100 are higher dimensional image classification datasets (32×32 pixels, 3 color channels) with 10, or 100 classes. The CIFAR-variants contain 60.000 images of objects and animals. SVHN contains 600.000 images of house numbers.

To evaluate `IALE` we train a policy π, run it on unseen datasets along with the baselines, and average the results (over 3 iterations). We denote the number of labeled samples in the experiments as *labeling effort* until a budget is reached, to be able to compare different acquisition sizes for the same total budget. The similarity between FMNIST and MNIST (that has previously been shown (Nalisnick et al., 2019)) and the difficulty of FMNIST (it has been shown to be a demanding dataset for AL methods (Hahn et al., 2019)) make these datasets a perfect combination to evaluate `IALE`. We show broader generalization on Extended MNIST and the higher dimensional SVHN and CIFAR datasets (5 repetitions). Appendix A.3.2 presents additional results for transferring π.

Architectures of classifier M. We use the same CNN architecture that has been employed in previous research (Gal and Ghahramani, 2016). Our model has two convolutional layers, followed by a max pooling and dense layer. We add dropout layers after the convolution and dense layers and use ReLU activations. A soft-max layer allows for classification. We



(a) CNN/MNIST (b) CNN/Fashion-MNIST (c) CNN/Kuzushiji-MNIST

Figure 3: Active learning performance of the trained policy (trained on MNIST), compared to the baseline approaches including *ALIL* (Liu et al., 2018), validated on MNIST and evaluated on FMNIST and KMNIST.

also provide results for using π on a simple MLP and on a more complex ResNet-18 (He et al., 2016), and supplemental results in Appendix A.3.1.

Architecture of policy π. Our policy model π uses an MLP with three dense layers with 128 neurons each. The first two dense layers are followed by a ReLU activation layer, whereas the final layer has only one neuron and the output of this layer is passed onto a sigmoid function to constrain the outputs to the range $[0, 1]$. and to further process it into an aggregating top-k operation.

Baselines. We compare our method with different well-known AL approaches from literature: *ALIL* (which we adapted from Liu et al. (2018) to work with image classification tasks), *MC-Dropout*, *Ensemble*, *CoreSet*, *BADGE*, *Confidence*-sampling, *Entropy*-sampling and a random sampling. Appendix A.1.1 provides a more details on the baselines.

Notation. We denote the unlabeled dataset as $\mathcal{D}_{\text{pool}}$, the already labeled data as $\mathcal{D}_{\text{lab}}$ and a labeled validation data $\mathcal{D}_{\text{val}}$. We randomly sub-sample $\mathcal{D}_{\text{sub}}$ of size n from $\mathcal{D}_{\text{pool}}$. We use a budget $\mathcal{B}$ and acquisition size acq to select $\mathcal{D}_{\text{sel}}$ from $\mathcal{D}_{\text{sub}}$. We derive the state $\mathcal{S}$ from a classifier M, e.g., a CNN or ResNet, to train `IALE`'s policy network π, i.e., an MLP with two hidden layers. In policy training, experts $\mathcal{E}$ propose actions $\mathcal{A}$. DAgger's hyper parameter p is the probability for following either the best expert or the policy π itself.

4.2 Policy Training and Validation

We use the `MNIST` dataset as our source dataset on which we train our policy for 100 episodes, with each episode containing data from an AL cycle. The initial amount of labeled training data is 20 samples (class-balanced). At each step of the active learning process, 10 samples are labeled and added to the training data until a labeling budget $\mathcal{B}$ of $1,000$ is reached. We use the AL heuristics *MC-Dropout*, *Ensemble*, *CoreSet*, *BADGE*, *Confidence* and *Entropy* as experts, and use $\mathcal{D}_{\text{val}}$ with 100 labeled samples to score the acquisitions of the experts. The pool dataset is sampled with $n = 100$ at each AL iteration. We choose $p = 0.5$ for means of comparison with the baselines (based on preliminary experiments, see Appendix A.2.1 on *Exploration-Exploitation*). We train the policy's MLP on the growing list of state and action pairs using the binary cross entropy loss from Equation 2 and use the Adam optimizer (Kingma and Ba, 2015) for 30 epochs with a learning rate of 10^{-3}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, ϵ=10^{-8}, without any weight decay.

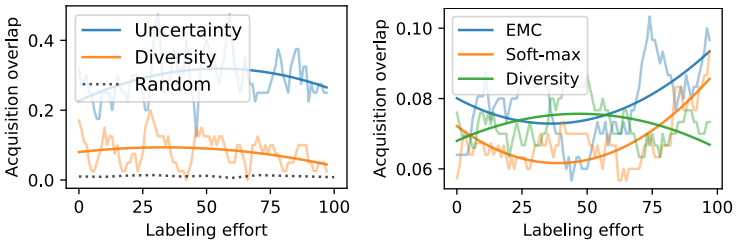
Figure 3a shows the results of our method in comparison to all the baseline approaches on MNIST, on which the policy was trained on. Our method consistently outperforms or is at least *en par* (towards the end, when enough representatives samples are labeled) with all the other methods. This finding on the policy-training dataset is not surprising, however, `IALE` performs better acquisitions than, e.g., *Ensemble* and *MC-Dropout*, for the important first half of the labeling budget, where it matters the most. In this experimental setting, *Confidence*-sampling performs similarly to the two more complex methods, even though it uses only the simple soft-max probabilities. While *Entropy* beats random sampling, it is still not competitive. *BADGE* performs similar to random sampling, which is due to the small acquisition size of 10 (the better performance of *BADGE* was reported with much larger acquisition sizes of 100 to 10,000 in Ash et al. (2020) as its mix of uncertainty and diversity heuristic benefits from these). The same applies to *CoreSet*, however, here it performs worst on average over all experiments. This finding is in line with previous research (Sinha et al., 2019; Hahn et al., 2019) and can be attributed to a weakness of the used p-norm distance metric regarding high-dimensional data, called the *distance concentration phenomenon*. The accuracy of *ALIL* on MNIST is similarly low as *CoreSet*. Moreover, *ALIL* is designed to add only one sample to the training data at a time (no batch-mode).

A general finding regarding computational efficiency in active learning is that `IALE` is faster than most baselines. While *MC-Dropout* requires 20 forward passes to decide which samples it acquires, and *Ensembles* $N = 5$ forward passes, one for each model, our approach requires only 2 inferences (for $\mathcal{D}_{\text{sub}}$ and $\mathcal{D}_{\text{lab}}$). The support for batch-mode (instead of selecting single samples) and using expert heuristics' batch acquisitions (instead of rolling out training of random samples from a small subset), accelerates the training of `IALE` compared to *ALIL* by several orders of magnitude (6 minutes versus 215 minutes per epoch on one NVidia Tesla V100 GPU). In a quantitative evaluation (with a labeling budget of 10.000 samples, an acquisition size of 10, training a ResNet-18 for 100 epochs, same GPU as before) the run time for `IALE` is 10:17:31 (hh:mm:ss) vs. 9:45:12 for random sampling. Compared to 49:15:23 for *Ensembles*, 14:23:18 for *MC-Dropout* and 11:58:47 for *BADGE*, this shows that `IALE` is faster. Only two baselines *Conf* (10:12:01) and *Entropy* (10:05:21) run faster, but they perform worse than `IALE` and even worse than random sampling.

4.3 Policy Transfer and Testing

To evaluate π's performance, we have to run it on a different dataset than the one that it has been trained on. Hence, we train π on the source dataset MNIST as in Section 4.2 and use it for the AL problem on FMNIST and KMNIST. We use an initial class-balanced labeled training dataset of 20 samples and add 10 samples per AL acquisition cycle until we reach a labeling budget of 1,000 samples. All the baselines are evaluated along with our method for comparison.

Figures 3b and 3c show the performance of `IALE` along with the baselines on FMNIST and KMNIST. Still, `IALE` consistently outperforms the baselines on both datasets. We can see that it learns a combined and improved policy that outperforms the individual experts consistently and (sometimes) even with large margins. On FMNIST `IALE` is the only method that actually beats a random sampling (similar findings have previously been reported by Hahn et al. (2019)). `IALE` is consistently $1 - 3\%$ better than random sampling



(a) Uncertain and diverse. (b) Types of uncertainty.

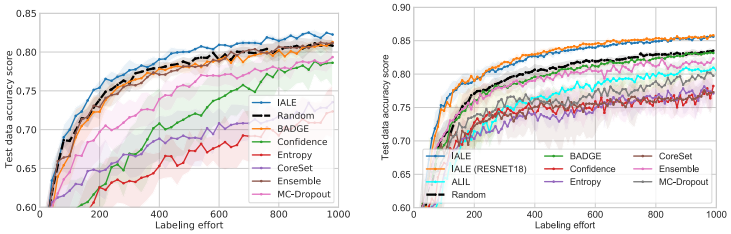
Figure 4: Complementary acquisition.

on FMNIST, on the harder KMNIST dataset `IALE` is even $7-9\%$ ahead. The baselines give a mixed picture. *ALIL*'s performance is not competitive on any task and actually never beats a random sampling strategy. We also see unstable performance for *MC-Dropout* and *Ensemble*, that generally perform similarly well. The simple soft-max heuristics *Entropy* and *Confidence* fail on FMNIST. *CoreSet* lags far behind, especially on KMNIST. *BADGE* always performs like random sampling, due to the aforementioned problematic acquisition size.

Sample composition of acquisition batches. We compare `IALE`'s chosen samples with the ones chosen by the experts, to gain insights on what `IALE` imitates and how the composition changes over the AL cycles. We evaluate all baseline experts and our method 5 times for 100 AL cycles on FMNIST ($|\mathcal{D}_{\text{sub}}| = 100$ and acq-size$= 10$), and report the results as well as fitted polynomials to highlight trends. In addition, we report the intersection of two i.i.d. randomly selected sets as the *Random* baseline with 1% overlap. For an acquisition size of 50, such a random overlap increases to 25%. Since the acquisitions between imitated baselines overlap, we are especially interested in their complementary acquisitions, e.g., samples that were only selected by a specific heuristic. Hence, we first separate AL into the families of *uncertainty*- and *diversity*-based methods. We group ensemble model combinations (EMCs) (Lakshminarayanan et al., 2017) (*Ensemble*, *MC-Dropout*) with single model soft-max methods (*Confidence*, *Entropy*), and compare with methods with diversity (*BADGE* and *CoreSet*). The results in Figure 4a show that the policy predominantly overlaps with uncertainty-based baselines. As Figure 4b shows, the exclusively by *EMCs* selected samples form the larger set. `IALE` may outperform any single heuristic due to its complementary strategy. The overlap between π and *any* other heuristic (intersection), decreases from about 80% to 60% over time. π selects samples that none of the experts choose, see Appendix A.2.2, where we also show π's overlap with single heuristics.

4.4 Policy Generalization

`IALE` learns a transferable AL policy. It unifies heuristics and generalizes over tasks and model architectures, because the state retains its formulation between tasks and architectures.



(a) MLP/Fashion-MNIST. (b) CNN/Fashion-MNIST.

Figure 5: (a) π (trained on CNN and MNIST) applied to MLP on FMNIST (b) π (trained on ResNet-18 and MNIST) applied to CNN on FMNIST.

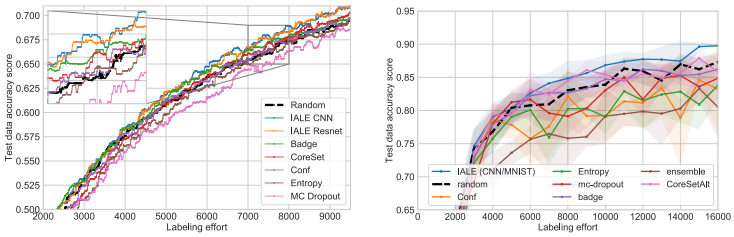
In this section, we evaluate the extend and limitations of the policy for transfers between MLP, CNN and ResNet classifiers, and on increasingly complex image datasets.[3]

Architecture transfer. Here, we explore the transfer of our approach to different architectures. We apply a policy, trained on CNN and MNIST, to an MLP (two hidden layers, 128 units, ReLU activation) on FMNIST and show the results in Figure 5a. Even with 5 random seeds and an increase batch size of 20 `IALE`, *BADGE* and *Ensemble* train classifiers stably, with `IALE` being on top. Next, we train a second policy on ResNet-18 and MNIST (`IALE ResNet`). We apply it to a CNN model on the FMNIST dataset and compare also with the previous policy (`IALE CNN` on MNIST). The results in Figure 5b show that both variants perform similarly despite their different policy training contexts. A potential explanation is that the policy learned similar decision strategies for both types (and sizes) of convolutional networks. This also shows that the state s and policy π are well-matched. First, the state s is rich enough for the policy π to learn and decode relevant information for generalizing the AL task. Second, the policy has a high enough capacity for transferring the policy to different networks and tasks, even though the embedding size itself is fixed.[4]

Both transfer experiments show `IALE`'s general ability to learn a *model-agnostic* AL strategy, of course within this experiment's scope. Experiments with deeper networks or an explanatory analysis of the policy's decision rules and state remain as future work.

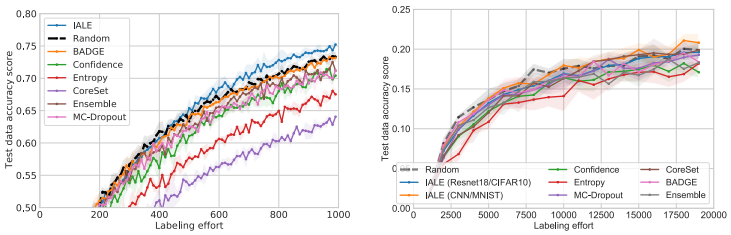
Higher-dimensional data. Next, we evaluate `IALE` on the higher-dimensional CIFAR-10 and SVHN. For the latter, we increase the batch- or acquisition size to $1,000$ so that training converges. We re-use the two policies (CNN/MNIST and ResNet/MNIST) from the previous experiment, but train exclusively ResNet-18 classifiers, because the simpler classifier models (MLP, CNN) did not yield satisfying results for any AL strategy. In summary, we use $2,000$ initial labels, an acq-size of 10 and $\mathcal{B} = 10,000$ for CIFAR-10 and $1,000$ initial labels, an acq-size of $1,000$ and $\mathcal{B} = 16,000$ for SVHN. Figure 6a shows all results on CIFAR-10

3. The transfer of the two different π's works because the shape of the classifier's embedding is invariant to architecture and data.
4. While the size of the embedding of the training and test architectures need to match exactly, it is only one aspect of the proposed approach besides the full state and the policy's network. Specifically, given an embedding of sufficient size we can construct an adequate state for good generalization from that embedding and other information, i.e., predictive uncertainty and gradient information. Our approach does not only rely on the embedding and the constructed state. Instead, the policy network itself has a large capacity and decodes the state.



(a) ResNet/CIFAR-10. (b) ResNet/SVHN.

Figure 6: (a) π (trained on CNN (`IALE CNN`) or ResNet-18 (`IALE ResNet`) on MNIST) applied to ResNet-18 on CIFAR-10 (filtered). (b) π (trained on CNN and MNIST) applied to ResNet-18 on SVHN (filtered).

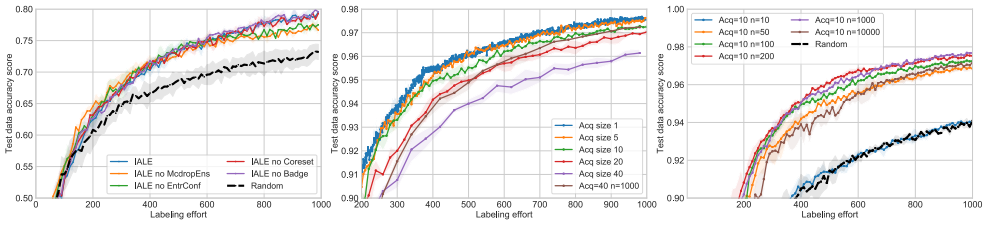


(a) CNN/Extended MNIST (letters). (b) ResNet/CIFAR-100.

Figure 7: (a) π (trained on CNN and MNIST) applied to CNN on Extended MNIST (letters). (b) similarly on CIFAR-100 (filtered).

with different policies: `IALE` is at least on par or better than a random sampling while the other experts are on par or worse than random sampling, some of them considerably. On SVHN, the performance gap opens wider with a larger batch-size (Figure 6b). Here, `IALE` performs best and is the only AL method that consistently beats a random sampling, with a relatively large margin. Furthermore, we want to emphasize that `IALE` generalizes beyond its budget of $1,000$ during policy training to longer time horizons of budgets like $10,000$ or even $16,000$. This is a benefit of our policy's introspective, state-based learning over alternatives like optimization-based meta-learning (Ravi and Larochelle, 2017; Chen et al., 2017), that struggles with longer time horizons as shown by Mishra et al. (2018) and Chen et al. (2017). Finally, these results show, that our framework learns a *task-agnostic* AL strategy for the presented image datasets. See Appendix A.3.2 for raw results.

Arbitrary class count. We perform experiments on Extended MNIST (26 classes; letters) and CIFAR-100 (100 classes) to increase the complexity of the classification task. To do this we must remove the fixed-length vectors *prediction* and *empirical class distribution* from the policy's state. Even though this may lead to a slightly poorer performance (see Appendix A.4.3) this allows transfers to tasks with an arbitrary numbers of classes. To start, we train two new policies with the reduced state, one with CNN on MNIST and another



(a) Experts. (b) Varying acquisition sizes. (c) Varying $|\mathcal{D}_{\text{sub}}|$.

Figure 8: Ablation studies on `IALE` for (a) different expert sets (on KMNIST), (b) acq-sizes and (c) n=$|\mathcal{D}_{\text{sub}}|$ (on MNIST).

with a ResNet-18 on CIFAR-10. Then, we apply `IALE` to a CNN on the Extended MNIST dataset.

Figure 7a shows the results from which we can draw two important observations: (1) `IALE` can be applied to problems with arbitrary class count, and (2) `IALE` still performs well compared to baselines. Next, we explore the limits by applying both policies to ResNet-18 on CIFAR-100 ($1,000$ initial labels, an acq-size of $1,000$ and $\mathcal{B} = 19,000$). Figure 7b shows that while `IALE` is still on par or better than the baselines, random sampling works surprisingly well, and shows the limitations of current AL heuristics, leaving space for future research.

4.5 Ablation Studies

Varying experts. To investigate the influence of experts, we leave out some types of experts: We categorize them into 4 simple groups, i.e., EMCs (*McdropEns*), soft-max uncertainty (*EntrConf*), diversity (*Coreset*) and hybrid (*Badge*), and leave one subset out. We fully train each method on MNIST with $\mathcal{B} = 1,000$ and an acquisition size of 10, and present the results of the evaluation on KMNIST in Figure 8a (more results including the ablation of state elements can be found in Appendixes A.4.2 and A.4.3). We see that most combinations perform well compared to the baselines. However, leaving out uncertainty-based heuristics can decrease performance, as they contribute the largest fraction to `IALE`'s selection composition (see Section 4.3). Even though training time is longer with *MC-Dropout*, the gains in performance can be worth it. In contrast, the soft-max uncertainty-based heuristics are computationally cheap and yield well-performing policies.

Hyperparameters. Two important parameters are the acquisition size *acq* and the size of $\mathcal{D}_{\text{sub}}$. Machine learning engineers may specifically be interested in modifying the acquisition size according to practical constraints. Hence, we show that arbitrary values are possible when applying π. Figures 8b and 8c show results for applying the policy for acq of 1 to 40 and size of $\mathcal{D}_{\text{sub}}$ between n=10 and n=$10,000$. During policy training we fixed acq= 10 and size of $\mathcal{D}_{\text{sub}}$ to n=100. This section also addresses the question of how `IALE` learns to sample diverse sets of points. Our empirical study shows that it does not sample non-diverse sets, which could be a failure state. Varying the acquisition size and $\mathcal{D}_{\text{sub}}$ produces the following insights. As expected, `IALE` performs best at acq= 1 and worst at acq= 40, if n is unchanged (because n limits the available choices), e.g., bad samples are chosen. Increasing n to $1,000$ alleviates this. However, there is an upper limit to the

size of $\mathcal{D}_{\text{sub}}$ after which performance deteriorates again, see Figure 8c. We believe that random sub-sampling simplifies the selection of diverse, uncertain samples. The performance decrease for small and large n supports this hypothesis. An optimal size of the sub-sampled data could be determined for different active learners, similarly to how some acquisition sizes are more suitable than others (see diversity vs. uncertainty in Appendix A.4). However, this additional interesting finding is not investigated further within this paper. The lower limit becomes apparent again when n is smaller than 10 times acq, with n =acq essentially being a random sampling. From our observations, n should be $10 - 100$ times acq (for 10 classes). From the small differences within this value range, it is suggested that our method is suitable for larger acquisition sizes for batch-mode AL, as its performance is not affected much.

5. Conclusion

We proposed a novel imitation learning approach for active learning. Our method learns to imitate the behavior of different active learners, such as uncertainty-, diversity-, model change- and query-by-committee-based heuristics, on one initial dataset and model, and transfers the obtained knowledge to work on other datasets and models (that share an embedding space). Our policy π is a simple MLP that learns a unified strategy from the experts based on a state with high capacity that contains gradient signals, embeddings and statistics of the data. Our experiments on different image datasets (four MNIST variants, CIFAR-10/100, SVHN) and model architectures (MLP, CNN, ResNet) show that `IALE` outperforms the state of the art and learned a complementary strategy. An ablation study and analysis of the influence of certain hyper-parameters also shows the limitations of our approach.

Future work investigates alternatives to the sampling step, as it may lead to sub-optimal choices from very large or imbalanced datasets. This would require a different loss than cross-entropy, in order to retain the ordering information, and could lead to a reformulation as a learning-to-rank problem of (compatible) experts' choices instead. Finally, an analysis of how π's *state* enables a transfer of its active learning strategy between classifier architectures and datasets may lead to some level of explanation of the principles of deep active learning.

Acknowledgments

We would like to acknowledge support for this project from the Bavarian Ministry of Economic Affairs, Infrastructure, Energy and Technology as part of the Bavarian project Leistungszentrum Elektroniksysteme (LZE) and the Center for Analytics-Data-Applications (ADA-Center) within the framework of "BAYERN DIGITAL II".

Appendix A.

In this section we provide an extension of the experiments section (Section 4) and feature additional results that support a more complete evaluation of `IALE`. We adhere to the same section structure.

A.1 Experimental Setup

A.1.1 Baselines

In the following is a short explanation of the baselines and experts that we used in our experiments:

1. *Random Sampling* randomly samples data points from the unlabeled pool.

2. *MC-Dropout* (Gal et al., 2017) approximates the sample uncertainty of the model by repeatedly computing inferences of the sample, i.e., 20 times, with dropout enabled in the classification model.

3. *Ensemble* (Beluch et al., 2018) trains an ensemble of 5 classifiers with different weight initializations. The uncertainty of the samples is quantified by the disagreement between the model predictions.

4. *CoreSet* (Sener and Savarese, 2018) solves the k-center problem using the pool-embeddings of the last dense layer (128 neurons) before the soft-max output to pick samples for labeling.

5. *BADGE* (Ash et al., 2020) uses the gradient of the loss (given pseudo labels), both its magnitude and direction, for k-means++ clustering, to select uncertain and diverse samples from a batch.

6. *Confidence*-sampling (Wang and Shang, 2014) selects samples with the lowest class probability of the soft-max predictions.

7. *Entropy*-sampling (Wang and Shang, 2014) calculates the soft-max class probabilities' entropy and then selects samples with the largest entropy, i.e., where the model is least certain.

8. *ALIL* (Liu et al., 2018): we modify *ALIL*'s implementation (that is initially intended for NLP tasks) to work on image classification task. Due to the high runtime costs of running *ALIL* (as the acquisition size is 1), we perform the training of *ALIL* for 20 episodes. We trained the *ALIL* policy network with a labeling budget $\mathcal{B}$ of $1,000$ and an up-scaled policy network comparable to that of our method along with a similar M as we use to evaluate the other AL approaches. We left the coin-toss parameter p at 0.5, and the k parameter for sequential selections from a random subset of $\mathcal{D}_{\text{pool}}$ at 10.

We use the variation ratio metric (Gal et al., 2017) to quantify and select the data samples for labeling from the uncertainty obtained from *MC-Dropout* and *Ensemble* heuristics. The

variation ratio metric is given by its Bayesian definition (Gal et al., 2017) for a data sample $x \in \mathcal{D}_{\text{pool}}$ in Equation 3 and for an ensemble expert (Beluch et al., 2018) in Equation 4:

$$\text{variation-ratio}(x) = 1 - max_y p(y|x, D) \tag{3}$$

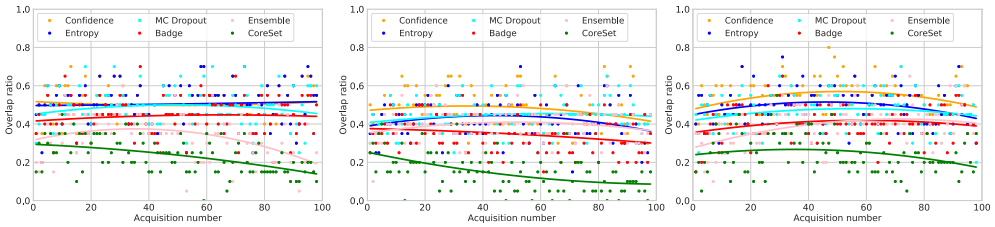
$$= 1 - \frac{m}{N}, \tag{4}$$

where m is number of occurrences of the mode and N is the number of forward passes or number of models in the ensemble.

A.1.2 Datasets

We show samples of the three datasets MNIST, Fashion-MNIST and Kuzushiji-MNIST in Figure 9 to illustrate their similarity. Extended MNIST consists of handwritten letters, while the other image datasets used in the evaluation (CIFAR and SVHN) differ greatly and have color information.

Figure 9: Examples for the three datasets MNIST, Fashion-MNIST, and Kuzushiji-MNIST.



(a) MNIST (b) Fashion-MNIST (c) Kuzushiji-MNIST

Figure 10: The overlap plots for all datasets MNIST, FMNIST and KMNIST datasets.

A.2 Policy Training

A.2.1 Exploration-Exploitation in DAgger

DAgger uses a hyper-parameter p that determines how likely π predicts the next action, and thereby setting the next state, instead of using the best expert from $\mathcal{E}$. In this preliminary study we compare the influence it has to either fix p to 0.5 or to use an exponential decay parameterized by the number of the current episode *epi*: $1 - 0.9^{epi}$. We train the policy on MNIST for 100 episodes with a labeling budget of 1,000 and an acquisition size of 10 (as before). Our result is that the *fixed* policy outperforms the *exponential* one by a small margin for the transfer of the policy to another dataset than the trained one, which is in line with previous findings (Liu et al., 2018).

A balanced (i.e., fixed) ratio does not emphasize one over the other, whereas an exponentially decay quickly relies on the policy for selecting new states of the dataset, and thus it trains on too few optimal states over the AL cycle.

A.2.2 Overlap ratios

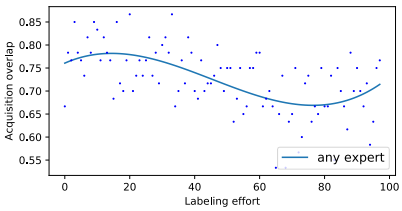


Figure 11: Overlap with any expert on Fashion-MNIST.

In addition to analyzing complementary compositions `IALE`'s acquisitions, we show overlaps with each baselines independently. This detailed view shows which heuristic π imitates the most. The overlap is given in percent in relation to the baselines (see Figure 10) for different datasets. We plot second-order polynomials, fit to the percentages (given as dots) over 100 acquisitions of size 10. Interestingly, the overall overlap is lower on FMNIST, where our method is the only one that beats a random sampling. We confirm again that π mostly imitates uncertainty-based heuristics, i.e., soft-max heuristics and *MC-Dropout*, and the uncertainty-/diversity-heuristic *BADGE* (close behind). *Ensemble* is overlapping mostly at the beginning. *CoreSet* has the lowest overlap. Interestingly, the policy chooses about half of the samples differently from any single baseline. On the other hand, the overlap with *any* expert is relatively high but decreases over the AL cycle (see Figure 11). In other words, the policy selects a portion of samples that none of the experts selected. Note that `IALE`'s acquisitions are build from combinations of the heuristics (instead of single votes), as we show in Section 4.3. Here, the percentages do not sum up to 1 as the overlap ratios between baseline and `IALE` are independent and may also overlap with each other.

A.3 Policy Transfer

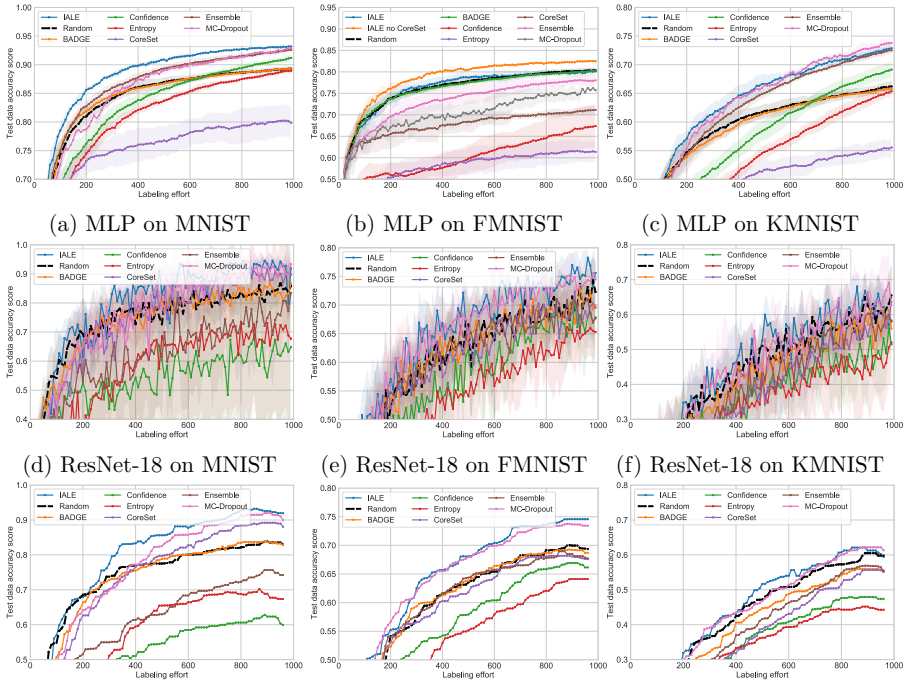
In this section we provide additional results on our studies on how our method performs in regard to applying it to unseen scenarios. These include that we use different datasets and classifier models in training and application of the policy. We show that π learns a task-agnostic AL strategy, that outperforms the baselines.

A.3.1 Classifier Architecture

In the evaluation Section 4, we show that our method is not bound to a specific classifier architecture. Here, we add results for the MLP architecture and give raw curves for a ResNet-18 classifier. To train `IALE`, we train policies on MNIST and apply them to all MNIST variants. All results for the experiments are given in Figure 12. We see the robustness of π over fundamentally different classifier architectures (2 to 18 layers). The deviations

for ResNet-18 are very large due to the very deep architecture and the modest amount of training data. We use median filtering in Figures 12g, 12h, and 12i.

These experiments show that π can learn AL strategies for both very small and very deep architectures and still outperform baselines. Even though the strongest baselines, i.e., *CoreSet* and *MC-Dropout*, come close to our method in accuracy, they are less versatile and require more computational resources, that is especially noticeable on deeper architectures.



(a) MLP on MNIST (b) MLP on FMNIST (c) MLP on KMNIST

(d) ResNet-18 on MNIST (e) ResNet-18 on FMNIST (f) ResNet-18 on KMNIST

(g) ResNet-18 on MNIST (h) ResNet-18 on FMNIST (i) ResNet-18 on KMNIST

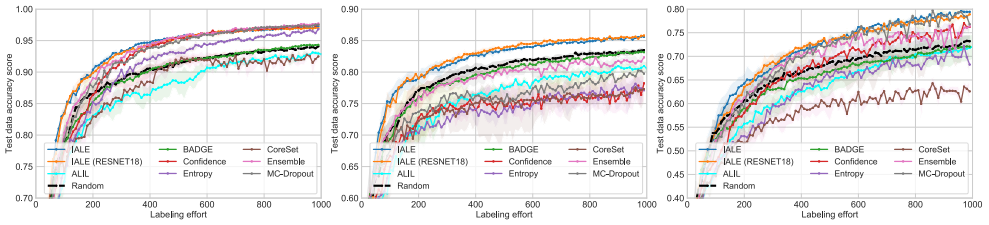
Figure 12: MLP and ResNet-18 classifiers, data averaged or median filtered. Active learning performance of the trained policy in comparison with the baseline approaches on MNIST, FMNIST and KMNIST datasets.

A.3.2 Classifier Architecture and Dataset

In Section 4.4, we show that π learns active learning independent from dataset and classifier. Here, we show additional results that mix both the source datasets *and* the classifiers.

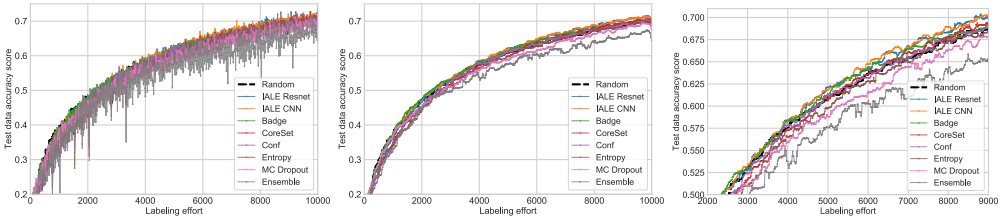
We report the results for applying π (trained on ResNet-18 and MNIST) to a CNN and all MNIST variants in Figure 13. `IALE` is always performing at the top, showing that it learns a *model- and task-agnostic* active learning strategy that transfers well.

CIFAR-10. We show additional results for applying π to a ResNet-18 classifier on `CIFAR-10`. To reiterate, we use two different π: π_1 was trained using ResNet-18 and MNIST (`IALE ResNet`) and π_2 was trained using CNN and MNIST (`IALE CNN`). The complete results



(a) MNIST (b) Fashion-MNIST (c) Kuzushiji-MNIST

Figure 13: Applying a policy trained using a ResNet-18 classifier (trained on MNIST) to a CNN-based classifier (on MNIST, FMNIST and KMNIST).



(a) full (raw) (b) full (filtered) (c) detail (filtered)

Figure 14: Full and enlarged segments of learning curve: Applying π trained on a CNN (IALE CNN) or ResNet-18 (IALE ResNet), trained on MNIST, to a ResNet-18 classifier on CIFAR10.

in Figure 14 are noisy due to the acquisition size of 10, and we report the raw learning curves (Figure 14a) and median filtered learning curves (Figure 14b). The most interesting segment of the learning curve is in Figures 14c in more detail and filtered. The results generally show the feasibility of transferring π to both different classifiers and datasets. `IALE` is on par or better than random sampling, and the other baselines are either on par or worse than random sampling (some of them considerably).

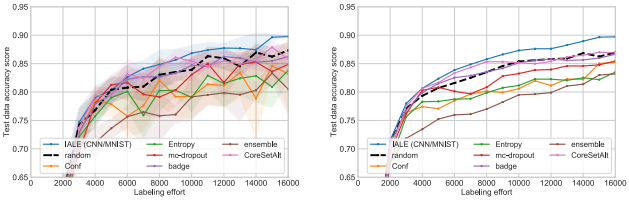
SVHN. We show the averaged learning curves for `SVHN` in Fig. 15a besides the smoothed averages for improved visibility in Fig. 15b. While the results exhibit some variance, we can clearly see that `IALE` performs best (and is the only AL methods that is consistently able to beat a random sampling.

While more experiments are certainly required to further emphasize these initial claims of generalizability to more diverse tasks, these findings are already very promising.

A.4 Ablation Studies

A.4.1 Hyperparameters.

We report fine-granular steps of acquisition sizes (see Figure 16a) with values between 1 and 10, plus 20 and 40, for $|\mathcal{D}_{\text{sub}}|$ of 100. Overall, a clear difference is not visible below 10 samples. For enhanced readability, we show a magnified section of the varied acquisition

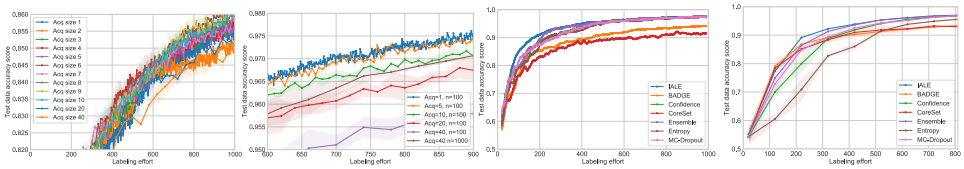


(a) SVHN. (b) SVHN.

Figure 15: The more complex dataset SVHN requiress more samples than MNIST variants. Learning curves as (a) averages and (b) smoothed plots).

sizes and $|\mathcal{D}_{\text{sub}}|$ in Figure 16b, that clearly shows the benefits of tuning $|\mathcal{D}_{\text{sub}}|$ to a suitable value for the acquisition size.

Acquisition sizes including baselines: We additionally compare the baseline active learning methods with our approach, as these exhibit different performance at different acquisition sizes, see Figure 16. We have included comparisons with acquisition sizes of either 1 or 100 (1 or 3 repetitions). For our method, for an acquisition size of 1 we chose $|\mathcal{D}_{\text{sub}}| = 100$ and for acquisition size of 100 we chose $|\mathcal{D}_{\text{sub}}| = 2,000$. While the results show that `IALE` outperforms the baselines they also highlight the large effect that the acquisition size has on some of the baseline methods. For instance, *CoreSet* constructs better set covers with larger batches, and *BADGE* increases its accuracy by constructing a representative sampling as well. At the same time the uncertainty-based methods, apart from *Entropy*, remain unaffected.

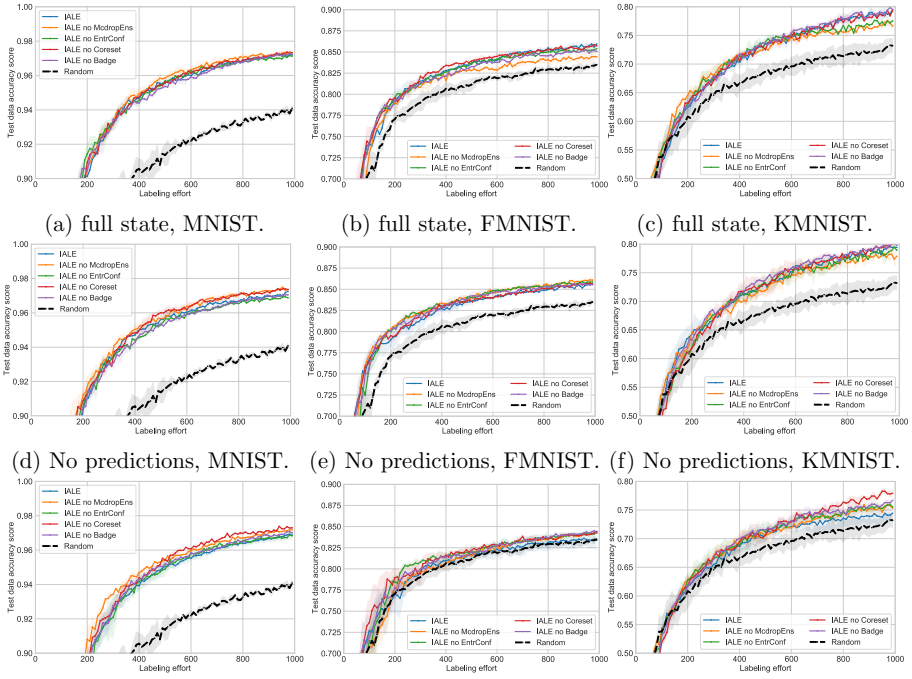


(a) Varying acq-sizes. (b) Varying acq-sizes (c) acq-size 1. (d) acq-size 100.

Figure 16: (a) Evaluating the acquisition sizes from 1 to 10 on FMNIST, and (b) varying different sub-pool sizes on MNIST. Diversity vs. uncertainty: Some experts are more suitable to other acquisition sizes, see (c) and (d), both evaluated on MNIST.

A.4.2 Varying Experts

We present more results for variations of sets of experts in Figure 17, and train the policies with the unchanged hyper-parameters and the CNN classifier on MNIST. The results for all three datasets show that the generally high performance of `IALE` holds for the leave-one-out sets of experts, with the full set of experts being consistently among the best performing policies.



(a) full state, MNIST. (b) full state, FMNIST. (c) full state, KMNIST.

(d) No predictions, MNIST. (e) No predictions, FMNIST. (f) No predictions, KMNIST.

(g) No gradients, MNIST. (h) No gradients, FMNIST. (i) No gradients, KMNIST.

Figure 17: The active learning performance for each (leave-one-out) set of experts. For partial state (without predictions, without gradients), we plot active learning performance for each (leave-one-out) set of experts.

A.4.3 Varying State Elements

Next, we study the state more closely. For unlabeled samples, the state contains two types of representations for predictive uncertainty: the statistics on predicted labels $M(x_n)$ and the gradient representations $g(M_e(x_n))$. In this study, we focus on leaving out one or the other. To get the full picture, we again train sets of experts for reduced states.

In Figure 17 we see that dropping gradients generally decreases performance (bottom row), while dropping predicted labels $M(x_n)$ affects performance very little (top row). However, the influence of different sets of experts is more important. We cannot see that a particular set of states and experts generally outperforms others consistently (while the negative effect of leaving out $g(M_e(x_n))$ is consistently visible). Overall, we find that using as many experts as available, combined with a full state both performs well and works reliably. Even though training a policy this way does not guarantee the best performance, it always performs among with the group of best policies.

References

Jordan T. Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. In *International Conference on Learning Representations (ICLR)*, Virtual Conference, Formerly Addis Ababa Ethiopia, 2020.

Philip Bachman, Alessandro Sordoni, and Adam Trischler. Learning algorithms for active learning. In *34th International Conference on Machine Learning (ICML)*, pages 301–310, Sydney, Australia, 2017.

Yoram Baram, Ran El-Yaniv, and Kobi Luz. Online choice of active learning algorithms. *Journal of Machine Learning Research*, (5):255–291, 2004.

Loïc Barrault, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post, and Marcos Zampieri. Findings of the 2019 conference on machine translation. In *4th Conference on Machine Translation*, pages 1–61, Florence, Italy, 2019.

William H Beluch, Tim Genewein, Andreas Nürnberger, and Jan M Köhler. The power of ensembles for active learning in image classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9368–9377, Salt Lake City, UT, 2018.

Arantxa Casanova, Pedro O. Pinheiro, Negar Rostamzadeh, and Christopher J. Pal. Reinforced active learning for image segmentation. In *International Conference on Learning Representations (ICLR)*, Virtual Conference, Formerly Addis Ababa Ethiopia, 2020.

Yutian Chen, Matthew W. Hoffman, Sergio Gómez Colmenarejo, Misha Denil, Timothy P. Lillicrap, Matt Botvinick, and Nando De Freitas. Learning to learn without gradient descent by gradient descent. In *International Conference on Machine Learning (ICML)*, volume 2, pages 1252–1260, Sydney, Australia, 2017.

Hong-Min Chu and Hsuan-Tien Lin. Can active learning experience be transferred? In *International Conference on Data Mining (ICDM)*, pages 841–846, Barcelona, Spain, 2016.

Tarin Clanuwat, Mikel Bober-Irizar, Asanobu Kitamoto, Alex Lamb, Kazuaki Yamamoto, and David Ha. Deep learning for classical japanese literature. In *NeurIPS Workshop on Machine Learning for Creativity and Design*, Montreal, Canada, 2018.

Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. Emnist: Extending mnist to handwritten letters. In *International Joint Conference on Neural Networks (IJCNN)*, pages 2921–2926, Anchorage, AK, 2017.

Gabriella Contardo, Ludovic Denoyer, and Thierry Artières. A Meta-Learning Approach to One-Step Active-Learning. In *International Workshop on Automatic Selection, Configuration and Composition of Machine Learning Algorithms*, pages 28–40, Skopje, Macedonia, 2017.

Yang Fan, Fei Tian, Tao Qin, Xiang-Yang Li, and Tie-Yan Liu. Learning to teach. *arXiv preprint arXiv:1805.03643*, 2018.

Meng Fang, Yuan Li, and Trevor Cohn. Learning how to active learn: A deep reinforcement learning approach. In *Conference on Empirical Methods in Natural Language Processing*, pages 595–605, Copenhagen, Denmark, 2017.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135. PMLR, 2017.

Chelsea Finn, Kelvin Xu, and Sergey Levine. Probabilistic model-agnostic meta-learning. *arXiv preprint arXiv:1806.02817*, 2018.

Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *33rd International Conference on Machine Learning (ICML)*, New York, NY, 2016.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. In *34th International Conference on Machine Learning (ICML)*, Sydney, Australia, 2017.

Lukas Hahn, Lutz Roese-Koerner, Peet Cremer, Urs Zimmermann, Ori Maoz, and Anton Kummert. On the robustness of active learning. In *5th Global Conference on Artificial Intelligence*, volume 65 of *EPiC Series in Computing*, pages 152–162, Bolzano, Italy, 2019.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, Las Vegas, NV, 2016.

Sepp Hochreiter, A. Steven Younger, and Peter R. Conwell. Learning to learn using gradient descent. In Georg Dorffner, Horst Bischof, and Kurt Hornik, editors, *Artificial Neural Networks — ICANN 2001*, pages 87–94, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg.

HM Sajjad Hossain, MD Abdullah Al Haiz Khan, and Nirmalya Roy. Deactive: Scaling activity recognition with active deep learning. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, 2:66:1–66:23, 2018.

Wei-Ning Hsu and Hsuan-Tien Lin. Active learning by learning. In *29th AAAI Conference on Artificial Intelligence (AAAI)*, page 2659–2665, Austin, Texas, 2015.

Muhammad Abdullah Jamal and Guo-Jun Qi. Task agnostic meta-learning for few-shot learning. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11719–11727, Long Beach, CA, 2019.

Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5574–5584, Long Beach, CA, 2017.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, San Diego, CA, 2015.

Ksenia Konyushkova, Raphael Sznitman, and Pascal Fua. Learning active learning from data. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4225–4235, Long Beach, CA, 2017.

Ksenia Konyushkova, Raphael Sznitman, and Pascal Fua. Discovering general-purpose active learning strategies. *arXiv preprint arXiv:1810.04114*, 2018.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 6402–6413, Long Beach, CA, 2017.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Minghan Li, Xialei Liu, Joost van de Weijer, and Bogdan Raducanu. Learning to rank for active learning: A listwise approach. In *International Conference on Pattern Recognition (ICPR)*, Milano, Italy, 2020.

Xin Li and Yuhong Guo. Adaptive active learning for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Portland, OR, 2013.

Ming Liu, Wray Buntine, and Gholamreza Haffari. Learning how to actively learn: A deep imitation learning approach. In *56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1874–1883, 2018.

Christoffer Löffler, Christian Nickel, Christopher Sobel, Daniel Dzibel, Jonathan Braat, Benjamin Gruhler, Philipp Woller, Nicolas Witt, and Christopher Mutschler. Automated quality assurance for hand-held tools via embedded classification and automl. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML-PKDD)*, Ghent, Belgium, 2020.

Dwarikanath Mahapatra, Behzad Bozorgtabar, Jean-Philippe Thiran, and Mauricio Reyes. Efficient active learning for image classification and segmentation using a sample selection and conditional generative adversarial network. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 580–588, Granada, Spain, 2018.

Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

N Mishra, M Rohaninejad, X Chen, and P Abbeel. A Simple Neural Attentive Meta-Learner. In *International Conference on Learning Representations (ICLR)*, pages 1–17, Vancouver, Canada, 2018.

Andriy Mnih and Danilo J. Rezende. Variational inference for monte carlo objectives. In *33rd International Conference on Machine Learning (ICML)*, page 2188–2196, New York, NY, 2016.

Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, and Balaji Lakshminarayanan. Do deep generative models know what they don't know? In *International Conference on Learning Representations (ICLR)*, New Orleans, LA, 2019.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NeurIPS Workshop on Deep Learning and Unsupervised Feature Learning*, Granada, Spain, 2011.

Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *International Conference on Learning Representations (ICLR)*, pages 1–11, Toulon, France, 2017.

Sachin Ravi and Hugo Larochelle. Meta-learning for batch mode active learning. In *International Conference on Learning Representations (ICLR), Workshop Track Proc.*, Vancouver, Canaday, 2018.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 627–635, Ft. Lauderdale, FL, 2011.

Dan Roth and Kevin Small. Margin-based active learning for structured output spaces. In Johannes Fürnkranz, Tobias Scheffer, and Myra Spiliopoulou, editors, *Machine Learning: ECML 2006*, pages 413–424, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations (ICLR)*, Vancouver, Canada, 2018.

Burr Settles. Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2009.

Burr Settles, Mark Craven, and Soumya Ray. Multiple-instance active learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1289–1296, Vancouver, Canada, 2008.

Samarth Sinha, Sayna Ebrahimi, and Trevor Darrell. Variational adversarial active learning. In *International Conference on Computer Vision (ICCV)*, pages 5972–5981, Seoul, South Korea, 2019.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4077–4087, Long Beach, CA, 2017.

LMA Tonnaer. Active learning in vae latent space. *Eindhoven University of Technology*, 2017.

Athanasios Voulodimos, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis. Deep learning for computer vision: A brief review. *Comp. Intelligence and Neuroscience*, 2018.

D. Wang and Y. Shang. A new active labeling method for deep learning. In *International Joint Conference on Neural Networks (IJCNN)*, pages 112–119, Beijing, China, 2014.

Mark Woodward and Chelsea Finn. Active one-shot learning. In *NeurIPS Deep Reinforcement Learning Workshop*, Barcelona, Spain, 2016.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

C Active Learning of Ordinal Embeddings: A User Study on Football Data

Active Learning of Ordinal Embeddings: A User Study on Football Data

Christoffer Löffler[1,2] *christoffer.loeffler@fau.de*
Dario Zanca[2] *dario.zanca@fau.de*
Björn Eskofier[2] *bjoern.eskofier@fau.de*
[2] *Friedrich-Alexander-Universität Erlangen-Nürnberg*
Machine Learning and Data Analytics (MaD) Lab,
Carl-Thiersch-Straße 2b, 91052 Erlangen, Germany

Kion Fallah[3] *kion@gatech.edu*
Stefano Fenu[4] *sfenu3@gatech.edu*
Christopher Rozell[3] *crozell@gatech.edu*
[3] *School of Electrical and Computer Engineering*
[4] *School of Interactive Computing*
Georgia Institute of Technology, Atlanta, GA

Christopher Mutschler[1] *christopher.mutschler@iis.fraunhofer.de*
[1] *Precise Positioning and Analytics*
Fraunhofer Institute for Integrated Circuits (IIS)
Nuremberg, Germany

Reviewed on OpenReview: *https://openreview.net/forum?id=oq3tx5kinu*

Abstract

Humans innately measure the distance between instances in an unlabeled dataset using an unknown similarity function. Distance metrics can only serve as a proxy for similarity in information retrieval of similar instances. Learning a good similarity function from human annotations improves the quality of retrievals. This work uses deep metric learning to learn these user-defined similarity functions from few annotations for a large football trajectory dataset. We adapt an entropy-based active learning method with recent work from triplet mining to collect easy-to-answer but still informative annotations from human participants and use them to train a deep convolutional network that generalizes to unseen samples. Our user study shows that our approach improves the quality of the information retrieval compared to a previous deep metric learning approach that relies on a Siamese network. Specifically, we shed light on the strengths and weaknesses of passive sampling heuristics and active learners alike by analyzing the participants' response efficacy. To this end, we collect accuracy, algorithmic time complexity, the participants' fatigue, time-to-response, qualitative self-assessment and statements, as well as the effects of mixed-expertise annotators and their consistency on model performance and transfer learning.

1 Introduction

Position tracking of persons, vehicles or objects is ubiquitous and enables various trajectory data mining tasks (Zheng, 2015), such as match or performance analysis in popular sports like football (Löffler et al., 2021), hockey (Chen et al., 2005) and basketball (Sha et al., 2016), or (public) transport planing and mobility analysis (Fernández et al., 2017; Shen et al., 2019; Yadamjav et al., 2020). With each additional moving agent, the data become increasingly complex due to its often unstructured and high-dimensional nature. For example, football has 23 trajectories (players and ball) that are typically tracked via multi-camera video systems Löffler et al. (2021). This affects tasks such as similarity-based information retrieval, which

queries similar occurrences to a given query scene (Sha et al., 2016). Here, a scene is an ensemble of agents' trajectories in a window of time.

In such large and complex datasets, information retrieval requires two expensive steps. First, the unstructured trajectories are optimally assigned, e.g., using the Hungarian algorithm Kuhn (1955). Second, the pairwise distance between the matched pairs of trajectories is computed on the raw trajectory data. Already by themselves, the two steps do not scale well to more realistic datasets that can have both a large number of samples and high dimensionality.

Recently, convolutional Siamese networks were leveraged by Löffler et al. (2021) to learn approximations of both the trajectory assignment and distance metric. The resulting lower-dimensional representation enables the scaling up of Euclidean distance-based information retrieval. However, there are two limitations. First, the Euclidean distance in itself is limiting, e.g., it does not weigh subjectively more important trajectories higher, and is affected by the data's high dimensionality (Canessa et al., 2020). Second, following directly from estimating the Euclidean distance of high dimensional data, the embedding captures global ordinal structures better than local ones (Löffler et al., 2021).

This work proposes to actively learn a distance function from human annotations. This learned similarity function is independent of the data's dimensionality and leads to a lower-dimensional ordinal embedding that more closely matches human perception. We address the associated costs of the human-in-the-loop by adapting an Active Learning (AL) sampler. We hypothesize that human annotations preserve more relevant information than the Euclidean distance and that a neural network can learn this perceived similarity from few annotations. We pay special attention to mixed-expertise annotators.

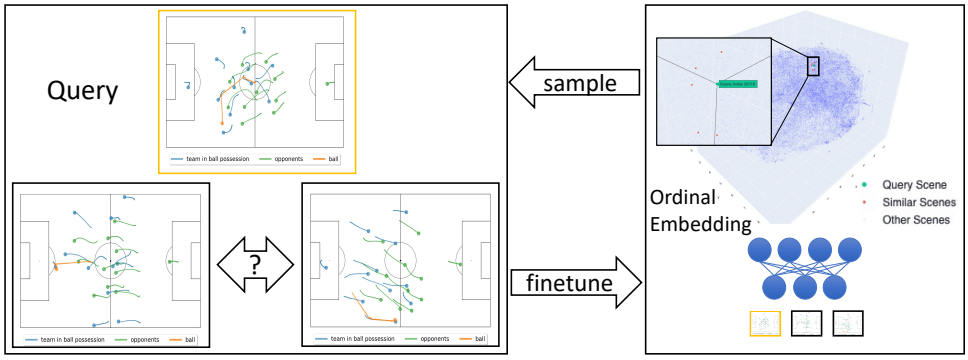
Our method learns rank-ordering from relative comparisons of a tuple of data instances using an active query selection method. In our experiments, we choose InfoTuple (Canal et al., 2020) over Crowd Kernel Learning (Tamuz et al., 2011) as it generalizes triplet queries to arbitrary tuple size and queries the oracle with more informative tuples. This has clear benefits: due to the tuples' larger size, they provide more context and are easier to annotate, and are also more sample efficient. We conduct a user study to evaluate the Active Learning sampling and use a well-suited football trajectory dataset for it. The study is centered around challenging relative comparison queries to human annotators. The query tuple composition impacts the efficiency of annotators, who may skip hard queries (see pre-study in Appendix A.1). Furthermore, while participants may have different similarity functions in mind, we can use a proxy similarity function for pre-training (Sha et al., 2016; Löffler et al., 2021). Crucially, the study allows a broader evaluation than accuracy per labeled sample. It also evaluates practically important questions such as the impact of skipped responses for different tuple composition algorithms, and it reports the annotators' intra-rater consistency and inter-rater reliability. We use a questionnaire for qualitative self-assessment of expertise and perceived similarity. These properties make our study suitable for the evaluation.

We pose three broader research questions. $\mathbf{RQ}_1$: How consistent and reliable are the participants? $\mathbf{RQ}_2$: What is the best baseline heuristic to generate tuple queries? $\mathbf{RQ}_3$: Does active sampling perform better than non-active sampling? Our experiments then also answer which methods are the most suitable with respect to pure improvement in effective accuracy, relative time efficiency (i.e., user response time), or sampling efficiency (i.e., number of tuples skipped). We conclude with a study on how users' response consistency affects the learned metric and the capacity for transfer learning. We summarize our contributions as follows

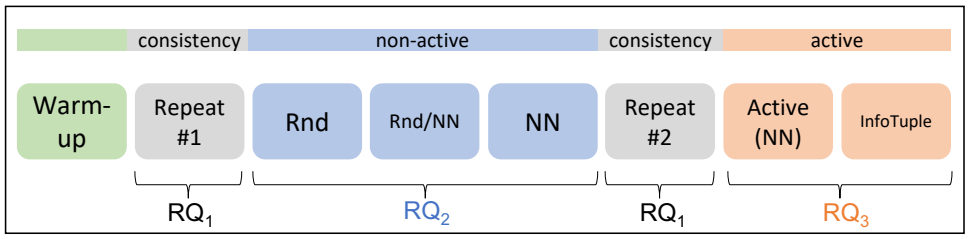
- We reduce the computational complexity of the InfoTuple active learner.
- We experimentally analyze the efficiency of our method with a user study on active learning methods on real-world data, analyzing strengths and limitations.
- We answer the research questions and show the adapted InfoTuple active sampling leads to higher triplet accuracy than (non) active sampling methods but falls behind slightly in sampling efficiency. In addition, participants can form relatively consistent groups of user-specific similarity functions.
- We release a simple but effective web app for active learning experiments[1].

[1]Code available at `https://github.com/crispchris/Active-Learning-of-Ordinal-Embeddings`

The rest of this article is structured as follows. Section 2 formulates the problem and presents AL methods and adaptations. Section 3 explains our experimental design, including the procedure, metrics, participants, and dataset. We present the results in Section 4 and discuss them in Section 5. Section 6 concludes.



(a) Left: We collect participants' responses to relative similarity queries. In our study, we use a tuple size of nine. Right: We finetune a neural network to learn an ordinal embedding. We use different (active) sampling methods to generate queries. We show a 3D UMAP (McInnes et al., 2018) plot of the learned embedding purely for visualization.



(b) We ask participants to annotate InfoTuples in sequential phases. We start with a warm-up, then measure consistency in two repeated phases ($\mathbf{RQ}_1$). The main research questions are answered in a non-active ($\mathbf{RQ}_2$) and in an active phase ($\mathbf{RQ}_3$). The query tuple composition heuristics in $\mathbf{RQ}_2$ are Random (Rnd), Nearest Neighbor (NN), and their combination (Rnd/NN).

Figure 1: The study consists of several different annotation phases. (a) shows the procedure to collect annotations and to finetune and sample from the learned embedding. (b) shows the multi-phase study design that compares different sampling strategies and evaluates model performance with respect to efficiency and effectiveness.

2 Method

Tuple composition and sample selection are important for learning an ordinal embedding specific to a human's similarity function. We formulate the problem first as pairwise similarity learning using a Siamese network like Löffler et al. (2021), which initially approximates assignments through metric learning (Section 2.1). This delivers a meaningful embedding to warm-start tuple selection strategies. Next, Section 2.2 extends the objective to triplet-based learning to generate an ordinal embedding. Section 2.3 then explains optimizations of InfoTuple, and our adaption to select the most informative samples from a meaningful candidate set.

2.1 Problem Formulation

We first investigate calculating pairwise distances of spatio-temporal matrices $\vec{x}$ of dimensionality $S \times T$:

$$\vec{x} = \begin{bmatrix} x_{1,1} & x_{1,2} & ... & x_{1,T} \\ x_{2,1} & x_{2,2} & ... & x_{2,T} \\ \vdots & \vdots & \ddots & \vdots \\ x_{S,1} & x_{S,2} & ... & x_{S,T} \end{bmatrix},$$

where the number of the dimensions of the trajectory S is the spatial dimension ($S = 2$ for 2D positional tracking), and the number of time steps T is the temporal dimension ($T = 125$ for 5 s tracking at 25 Hz). A trajectory may correspond to an agent like a football player. A row vector $\boldsymbol{x}_{s,1:T}$ represents a single dimension s over time T. Positional tracking in sports may be calculated from multi-perspective video feeds Löffler et al. (2021). For 5 s scenes of one player sampled at 25 Hz, this produces trajectories $\vec{x}$ of the dimensionality 2×125.

In multi-agent tracking, a scene $\boldsymbol{X}$ consists of N different trajectories such that $\boldsymbol{X} = \{\vec{x}^{(1)}, \vec{x}^{(2)}, .., \vec{x}^{(N)}\}$ (Löffler et al., 2021). See Fig. 1a for three scenes. Hence, $\boldsymbol{X}$ is of dimensionality $N \times S \times T$. For a scene with all 22 players and 1 ball, this results in a size of $23 \times 2 \times 125$. However, the ordering of the dimension N is unknown, because agents may follow different strategies in each scene and can adapt their role in a game as needed Bialkowski et al. (2014). This is problematic for calculating the distance between matrices (Löffler et al., 2021). Specifically, given a pair of matrices $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$, the respective pairwise assignment of vectors $\vec{x}^{(i)}$ and $\vec{x}'^{(j)}$ from each matrix is unknown.

Hence, to calculate the matrices' pairwise distance, we first compute the optimal row-wise assignment in the dimension N of $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ using the Hungarian algorithm (Kuhn, 1955), that minimizes the sum of pairwise distances between the matrices in $\mathcal{O}(n^3)$. For this we calculate the distance d between two trajectories $\vec{x}$ and $\vec{x'}$ as the average Euclidean distance over the spatial dimensions at each point in time (Löffler et al., 2021):

$$d(\vec{x}, \vec{x'}) = \frac{1}{T} \sum_{t=1}^{T} ||\vec{x}_{:,t} - \vec{x'}_{:,t}||_2 \tag{1}$$

The distance between ensembles of trajectories $d_{ens}(\boldsymbol{X}_1, \boldsymbol{X}_2)$ is simply the sum of distances between trajectories (as in Eq.1) of matrices $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$, that are optimally assigned along dimension N (using the Hungarian algorithm). In terms of sports tracking data, this procedure maps players from one scene optimally onto players of another scene such that the distance between scenes is minimal.

We then further follow Löffler et al. (2021) and use Deep Siamese Metric Learning with a Temporal Convolutional Network (Lea et al., 2016) with a Resnet architecture (He et al., 2016) f to learn an approximate assignment and lower dimensional, distance-preserving embedding. The observations are pairs of matrices $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ sampled uniformly at random from all possible permutations of the dataset. Our goal is to find an embedding that preserves the distance $d_{ens}(\boldsymbol{X}_1, \boldsymbol{X}_2) \approx \hat{d}_{ens}(\widehat{\boldsymbol{X}}_1, \widehat{\boldsymbol{X}}_2)$ where $\hat{d}_{ens}$ is the Euclidean distance between the learned lower dimensional representations $\mathrm{f}(\boldsymbol{X}_1) = \widehat{\boldsymbol{X}}_1$ and $\mathrm{f}(\boldsymbol{X}_2) = \widehat{\boldsymbol{X}}_2$. Hence, the learning objective can be formulated as

$$\mathcal{L}(\boldsymbol{X}_1, \boldsymbol{X}_2) = \big(||\,\mathrm{f}(\boldsymbol{X}_1) - \mathrm{f}(\boldsymbol{X}_2)||_2 - d_{ens}(\boldsymbol{X}_1, \boldsymbol{X}_2)\big)^2. \tag{2}$$

We use two L_2 regularization terms. The first $||\,\mathrm{f}(\boldsymbol{X}_1)||_2 + ||\,\mathrm{f}(\boldsymbol{X}_2)||_2$ centers learned representations and the second $||\theta||_2$ performs weight regularization on the network f's parameters θ. $\hat{d}_{ens}$ is the Euclidean distance. It is simple to implement but still measures play similarity just as well as, e.g, Dynamic Time Warping, Fréchet distance or l_∞ distance (Sha et al., 2016).

$\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ contain trajectories of multiple *agents* which leads to a computationally expensive assignment problem (Sha et al., 2017) between two sets of trajectories describing two different scenes. This is because role

assignments are not fixed and often even disjoint between teams. Sha et al. (2017) build a tree structure to perform the assignment, that others subsequently use to dynamically set role-based assignments independent of a priori positions (Di et al., 2018). In contrast, we estimate these assignments similar to Löffler et al. (2021) (based on trajectory data and agnostic of roles) who compare three approaches: random assignments, exploiting additional meta-information such as roles, or inferring spatial proximity from data. The last variant uses a fixed grid template and assigns rows to channels, but this introduces sparsity in these inputs, which become overdetermined, as there are fewer players than entries in the grid template. Instead, we propose an improved template matching algorithm: we center a circular template with as many available positions as rows over the spatial center of $\boldsymbol{X}$. Then, we fit the template's variance in each dimension S to match $\boldsymbol{X}$. For a pair $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$, the template fitting and matching produces representations, with that we can estimate pairwise distances with lower error than Löffler et al. (2021), but without introducing sparsity in the network's inputs. See Appendix A.4 for the experimental comparison of the variants.

In summary, this approximates and reduces the computational complexity for s scenes from $\mathcal{O}(s \cdot n^3)$ down to $\mathcal{O}(s \cdot m)$ where m is the size of the network's embedding.

2.2 Active Metric Learning

Learning a metric from human annotators is costly. Active Learning helps reduce the amount of labeled relative comparisons by querying the oracle with informative samples (Settles, 2009; Houlsby et al., 2011). Active Learners use a model state, an acquisition function, and a query format.

Initial model state. Performing active sampling on a non-random model state is called warm-starting. Since there are no ground-truth labels available, we can use the learned Euclidean embedding to warm start AL methods, as inspired by Simo-Serra et al. (2015). The Euclidean embedding is independent of the participants' similarity function and can serve as a more general early estimate of similarity. Following that, we may then fine-tune this pre-trained model to an initial set of annotations before actively selecting queries.

Acquisition function. For learning from relative comparisons, the acquisition function needs to construct the query Q from the most suitable candidate samples from the dataset. Following the notation from Canal et al. (2020), a tuplewise query Q_n at time step n of the AL procedure has a "head" object a_n and an un-ordered "body" of objects $B^n = (b_1^n, b_2^n, ..., b_{k-1}^n)$. Now $Q_n = (a_n, B^n)$ denotes the n^{th} tuple query, and a participant's ranking response $R(Q_n) = (R_1(Q_n), ..., R_{k-1}(Q_n))$ which is a permutation of B^n rank-ordered by similarity such that $R_i(Q_n) \prec R_j(Q_n)$, if $i < j$. This indicates that the oracle ranks the object $R_i(Q_n)$ more similar to the anchor a_n than the object $R_j(Q_n)$ (Canal et al., 2020).

Next, we follow a principled approach for an initial Active Learning heuristic. Given the head $\boldsymbol{X}_a$, we categorize closer samples in the Euclidean embedding as either positive $\boldsymbol{X}_p$ or negative $\boldsymbol{X}_n$. Xuan et al. (2020) further break down neighbors based on their distance to $\boldsymbol{X}_a$. Hard negatives are close to the head but dissimilar, whereas easy negatives are far away from it and least similar. Of these two types, the hard samples are most beneficial for learning, whereas easy ones produce no useful gradient (Xuan et al., 2020). Conversely, hard positives are highly similar samples, that are far away from $\boldsymbol{X}_a$, and thus mining these for tuples is difficult. Easy positives on the other hand are naturally available in an appropriately pre-trained embedding. Xuan et al. (2020) show that selecting easy positives keeps intra-class variance and helps to avoid over-clustering of the embedding.

We adapt the triplet mining concepts to Active Learning. Query tuples Q_n composed of $\boldsymbol{X}_a$'s nearest neighbors can contain both easy positives and hard negatives. The other samples with higher distance from $\boldsymbol{X}_a$ are hard positives and easy negatives and are least useful when learning similarity. Nadagouda et al. (2022) recently proposed a similar NN acquisition function to collect both similarity and classification responses, see Sec. 2.4.

Query format. Generally, an ordinal embedding can be learned from tuples of size $s \geq 3$. A triplet loss (Chechik et al., 2010) over the three samples $\boldsymbol{X}_a, \boldsymbol{X}_p, \boldsymbol{X}_n$ is defined as follows:

$$\mathcal{L}(\boldsymbol{X}_a, \boldsymbol{X}_p, \boldsymbol{X}_n) = \max(\| \mathrm{f}(\boldsymbol{X}_a) - \mathrm{f}(\boldsymbol{X}_p) \|_2 - \| \mathrm{f}(\boldsymbol{X}_a) - \mathrm{f}(\boldsymbol{X}_n) \|_2 + 1, 0) \tag{3}$$

Canal et al. (2020) show that human annotators can benefit from larger tuple sizes than three. Query tuples Q_n or arbitrary size greater than three can provide more context for considering the similarity and can be more sample-efficient. We can simply decompose a response $R(Q_n)$ into triplets $t = \{\boldsymbol{X}_a, \boldsymbol{X}_p, \boldsymbol{X}_n\}$ and use triplet loss. In this work, we determine the tuple size experimentally via a pre-study.

To summarize, we establish *Active (NN)* as an Active Learning heuristic, that constructs queries of arbitrary size from the nearest neighborhood of an anchor sample in an embedding.

2.3 Informative Queries

The selection of easy positives and hard negatives may not reliably construct the most informative queries. InfoTuple (Canal et al., 2020) improves upon this by maximizing the information gain of new queries. Given a probabilistic embedding, and previous and candidate queries, InfoTuple then selects that query that maximizes the mutual information that a response provides. It then uses a d-dimensional probabilistic embedding, such as t-Stochastic Triplet Embedding (tSTE) (Van Der Maaten & Weinberger, 2012) or probabilistic Multi-Dimensional Scaling (MDS) (Tamuz et al., 2011), to embed the dataset $\mathcal{X}$ of size N. Next, Canal et al. (2020) use the datasets's embedding $\boldsymbol{M} \in \mathbb{R}^{d \times N}$ to define a similarity matrix $\boldsymbol{K} = \boldsymbol{M}^T\boldsymbol{M}$. From this similarity, the authors calculate a $N \times N$ pairwise distance matrix $\boldsymbol{D}$ for $\mathcal{X}$. Given the queries Q_1, Q_2, ..., Q_{n-1} and their responses $R(Q_1)$, $R(Q_2)$, ..., $R(Q_{n-1})$, they select the next query from a set of possible queries using the conditional entropy $H(\cdot|\cdot)$ of the next possible reply $R(Q_n)$:

$$\arg\min_{Q_n} H(R(Q_n)|R(Q_{n-1})) - H(R(Q_n)|\boldsymbol{K}, R(Q_{n-1})) \tag{4}$$

The equation trades of two terms: the first term selects for uncertain queries provided previous responses $R(Q_{n-1})$ while the second term selects for unambiguous responses $R(Q_n)$ that can be encoded in the similarity matrix $\boldsymbol{K}$ by responses $R(Q_{n-1})$. Balancing these two measures selects queries with high entropy, but that can be consistently annotated by the oracle.

Next, Canal et al. (2020) develop simplifying assumptions on the joint statistics of query and embedding to enable a tractable estimation via Monte Carlo sampling. Importantly, they assume a Gaussian distribution on inter-object distances, and only need to sample the distribution for the current query Q_n, as proposed by Lohaus et al. (2019). With these simplifications Eq. 5 and Eq. 6 depend only on the distance matrix $\boldsymbol{D}$:

$$H(R(Q_n)|R(Q_{n-1})) = H\left(\mathop{\mathbb{E}}_{D_{Q_n} \sim \mathcal{N}_{Q_n}^{n-1}}[p(R(Q_n)|\boldsymbol{D}_{Q_n})]\right) \tag{5}$$

$$H(R(Q_n)|K, R(Q_{n-1})) = \mathop{\mathbb{E}}_{D_{Q_n} \sim \mathcal{N}_{Q_n}^{n-1}}[H(p(R(Q_n)|\boldsymbol{D}_{Q_n}))] \tag{6}$$

They slightly abuse the notation by using $H(X) = H(p(X))$ for a probability mass function p of random variable X, and $\mathcal{N}_{Q_n}^{n-1}$ to represent the Gaussian distribution $\mathcal{N}(D_{Q_n}^{n-1}, \sigma_{n-1}^2)$. We refer to Canal et al. (2020) for further details.

2.4 Adaption

In this work, we use a tandem of a generalizing neural network for learning similarity and InfoTuple's probabilistic model for selecting queries via Active Learning. We adapt InfoTuple to increase its effectiveness, leading to less ambiguous queries, improved sampling efficiency, and a reduction of the required time to select informative tuples.

Effectiveness. Following the assumptions in Sec. 2.2, we use the neural network's embedding to generate sample candidates to form the body B from the neighborhood of a head a. This differs from the approach of Nadagouda et al. (2022), which selects the most informative nearest neighbor query from all possible queries. We do this because our problem domain suffers from sparse similarity. For instance, successful goals are

typically rare but passes in midfield are much more common, leading to a long-tailed, imbalanced distribution of sample similarity with sparsity in the tail (Wang et al., 2010). Thus, queries may often provide no positive and informative candidates. This leads to a high rate of skipped queries, which in turn leads to fatigue of the human annotators. We discuss this phenomenon in Sec. 5. This pre-selection from the Neural Network's embedding space increases the likelihood of finding similar candidates and reduces the subjective difficulty of the annotation task. Based upon a sampling of m candidates, InfoTuple then selects the most informative query Q_n. The set of m candidates is larger than the tuple size of Q_n and is a hyperparameter (see Sec. 3.5).

Efficiency. The temporal complexity of the selection algorithm is primarily bound in the Monte Carlo style computation of Eq. 5 and Eq. 6. First, reducing the available candidate samples for greater effectiveness also reduces computational complexity. It leads to a smaller set of candidate samples available for constructing queries from. Second, we downsample the remaining m samples like Canal et al. (2020), which further reduces the number of constructed queries Q that have to be evaluated.

3 Experimental Design

In this section, we design a user study to answer the initial research questions.

3.1 Procedure

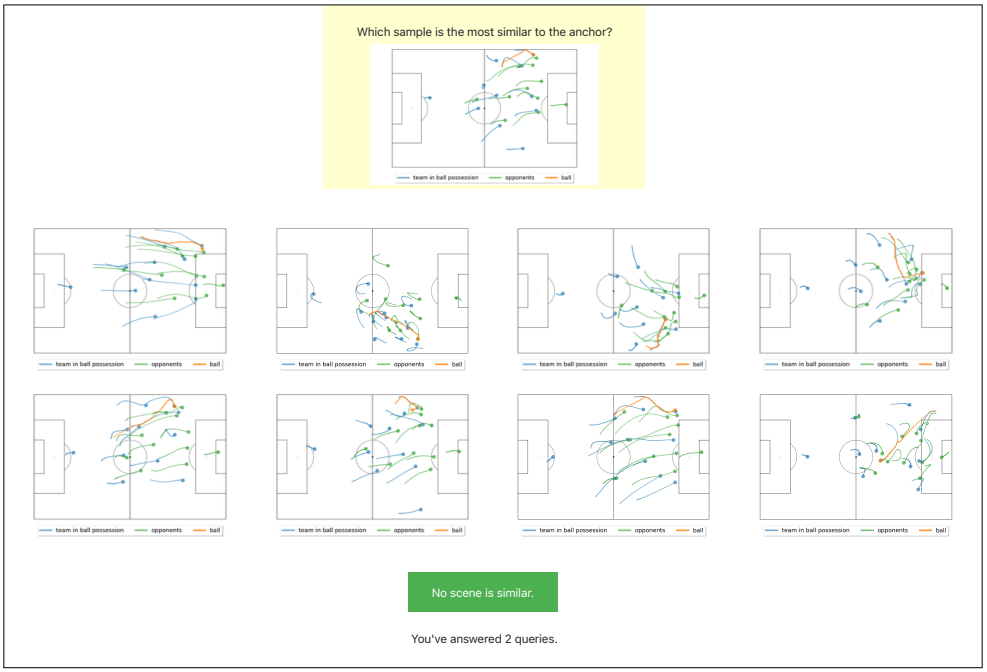


Figure 2: The web-based query page asks annotators to compare eight samples with an anchor and to either select the most similar sample or skip the query. Each sample shows the team in ball possession (blue), the defending team (green), and the ball (orange).

The study is designed in several phases and continuously collects annotations from 18 participants, see Fig. 1b. A smaller pre-study with 7 participants provided preliminary insights. With its help, we set the tuple size of InfoTuple to one anchor and eight samples for its body. The number of required samples for learning a meaningful ordinal embedding was about 250 to 300 triplets. Hence, the user study collects an equivalent number of tuples. Furthermore, we chose to train user-specific models, as the pre-study showed

that learned similarity is highly individual and does not generalize well. We include an evaluation of this notion in Sec. 4.4.

Task description. The task is to annotate an InfoTuple such as in Fig. 2. Participants compare an anchor with eight other samples, and then either choose the most similar sample, based on their own function of similarity or skip the query.

Phases. The user study is segmented into five parts. The first is an introduction with a description of the data and the mechanics of the annotation process. The second is a short warm-up of five queries, that familiarizes participants with the mechanics of the annotation website. Next, the split phase $\mathbf{RQ}_1$ occurs before and after $\mathbf{RQ}_2$. Finally, the active learning $\mathbf{RQ}_3$ concludes. $\mathbf{RQ}_1$ addresses intra-/inter-rater metrics. We repeat a fixed set of InfoTuple as Fig. 1b shows. We denote these sets as *repeated set* and highlight the phases as $\mathbf{RQ}_1$ in the figure. The repetition takes place after other phases, and the location of samples on the website is randomized, in order to break up any potentially lingering memories of the spatial layout of repeated queries. This way, we collect at least 20 InfoTuple but increase the set for each skipped sample. Hence, the repeated set may also grow in size as it also repeats skipped samples. Next, for the second research question, we collect annotations with different compositions of InfoTuple. The figure highlights the three segments as $\mathbf{RQ}_2$: *Rnd*, *Random/NN* and *NN*. We collect 20 InfoTuple to answer the research question and 10 more for testing. Finally, the last phase labeled $\mathbf{RQ}_3$ executes two different active learning strategies and trains neural networks with the annotations collected from participants. The first method is the Nearest Neighbor active learner, and the second is the adapted InfoTuple algorithm. We collect 20 InfoTuple with an acquisition size of one.

Warm start. We use a warm-start strategy for active learners. Starting from a pre-trained Siamese network, we finetune a model for each participant with their replies to the repeat #1 phase.

Annotation tool. The annotation tool is implemented in Python as a Flask web app. It uses the PyTorch framework for deep learning and integrates InfoTuple [2]. The user study was performed on an AMD Ryzen 7 3800X (64GB RAM) and an NVidia Geforce RTX 2070 Super (8GB VRAM), and participants used their clients to access a web frontend. Each participant had the same computing time and resources available.

3.2 Survey

We ask participants to self-assess in order to experiment with extensions from individual models toward generalized models of user-defined similarity. This serves as a qualitative, parallel approach to the intra- and inter-rater metrics.

We provide a simple questionnaire to inquire about the participants' self-assessment of their expertise and a qualitative description of their individual similarity function. We specifically ask them about their definition of similarity before and after the participation. This allows for qualitatively estimating the clustering of participants into groups that employ similar notions.

The questions were the following:

1. What is your expertise in football on a scale of 1 (novice) to 6 (expert)?
2. What will you look for when you compare football scenes' similarity?
3. Did your definition of similarity change?
4. What did you look for when you compare football scenes' similarity?

The first two questions were answered when looking at the tutorial. To capture any changes, the last two questions were answered after finishing all queries of the procedure. See Appendix A.2 for an analysis.

[2] https://github.com/siplab-gt/infotuple

3.3 Metrics

Triplet accuracy is defined as the agreement between two triplets (a, b_1, b_2) and (a', b'_1, b'_2) for a set of annotated triplets. In this study, we generate triplets from annotated tuples of arbitrary size.

Intra-rater consistency C for a participant P_i measures the agreement of annotations and skipped samples between the two phases Repeat #1 and Repeat #2:

$$C(P_i) = \frac{\text{ratings}_{\text{agreement}}(P_i)}{\text{ratings}_{\text{total}}(P_i)}. \tag{7}$$

Pairwise inter-rater reliability R for two participants P_i and P_j includes numbers of all annotations and skips. It is the fraction of ratings in agreement over the total number of ratings:

$$R_{P_i,P_j} = \frac{\text{ratings}_{\text{agreement}}(P_i, P_j)}{\text{ratings}_{\text{total}}(P_i, P_j)}. \tag{8}$$

Response effectiveness (Bernard et al., 2018) E is the measure of accuracy per time spend on a response. More effective sampling methods have a higher ratio relative to others.

$$E = \frac{\text{triplet accuracy}}{\text{response time}}. \tag{9}$$

We additionally evaluate the total effectiveness TE of AL samplers, which includes the time spent sampling and training a model. While this depends on the choice of model and available resources for sampling and training, we consider it a relevant performance factor

$$TE = \frac{\text{triplet accuracy}}{\text{response time} + \text{computation time}}. \tag{10}$$

A high number of skipped responses may lead to participants' fatigue. Hence, we track the methods' label effectiveness LE. The accuracy relative to the number of skipped annotations provides a measure that extends the concept of efficiency (Bernard et al., 2018), which only measures the number of labeled instances over time.

$$LE = \frac{\text{triplet accuracy}}{\text{skipped responses}}. \tag{11}$$

3.4 Dataset

We use a dataset from the German Bundesliga from season 2014/15, that consists of trajectories sampled at 25 Hz of 304 games, extracted from multi-perspective video feeds.

For our user study, we choose one game and extract scenes of 5 s fixed length with 50% overlap of 2.5 s. We further pre-process the data to simplify the implementation: we transform the data so that the team in ball possession players from left to right, we only use active (not paused) scenes, where one team has more ball possession than the other, and we require that no players are missing. This yields $1,005$ samples for the game. We consider the experiment size to be minimal but still representative. It allows us to control side effects so that we can see and compare the actual performance of the different methods. This helps to answer the research questions and still tests the proposed method.

We pre-train the baseline Siamese network similarly to Löffler et al. (2021). For this, we extract about 1,200,000 scenes from 304 games with a smaller overlap of 1 s, and use the same pre-processing. Then, we split the data into train/validation/test splits of 80/10/10%. As we cannot process all possible combinations of scene pairs, we instead randomly sample 10 million pairs for training and 1 million otherwise. Then, we train using the regularized Siamese loss (see Eq. 2) and the ellipse assignment method (see Sec. 2.1). We refer to Löffler et al. (2021) for additional details on the baseline's training and optimizer configuration.

Test dataset. There is no ground-truth dataset for user-specific similarity functions. A full set of comparisons would be unfeasible, given the combinatorial complexity, further motivating the use of Active Learning. Hence, for the test dataset, we collect additional hold-out sets of each participant's annotations. The inconsistency of annotations then impacts the possible accuracy score and introduces a baseline test error. Especially a low intra-rater consistency poses an upper limit of achievable performance.

Additionally, the composition of tuples itself may have a large impact on the test scores, as they represent different embedding neighborhoods. Annotations sampled at random for testing would provide information on the ordinal embedding's global order and are less biased. Test tuples from the anchor's neighborhood are based on the pre-trained Siamese embedding and thus are biased. However, they may provide some insights into the embedding's fine structure.

3.5 Experimental Setup

We use a Temporal Convolutional Network (Lea et al., 2016) with a Resnet architecture (He et al., 2016) like Bai et al. (2018) and Löffler et al. (2021). For its training, we use the triplet loss described earlier with an Adam optimizer with a learning rate of 0.001 and batch size of 32 for 10 epochs after each acquisition. We implement the experiments in PyTorch.

The InfoTuple method uses the tSTE to fit a probabilistic embedding. Furthermore, we select the hyperparameters for InfoTuple as follows. We sub-sample the candidates to 100 neighboring samples, that is about 10% of the whole dataset, and generate 10 random permutations as possible queries Q_n. We set the number of Monte Carlo passes to 10 and sub-sample the factorial of the 8-tuples with the factor 0.1.

4 Experiments

This section first presents the fundamental intra- and inter-rater metrics in Sec. 4.1. With this context, we give the sampling efficiency of the evaluated methods in Sec. 4.2 and we specifically show how the quality of annotators' replies affects the learning of their similarity functions in Sec. 4.2.1 Next, we analyze response times and the amounts of skipping in Sec. 4.3, before we conclude with a study on clustering of highly agreeing annotators in Sec. 4.4. We asked 18 participants (aged 20 to 35, one female) to take part in the user study.

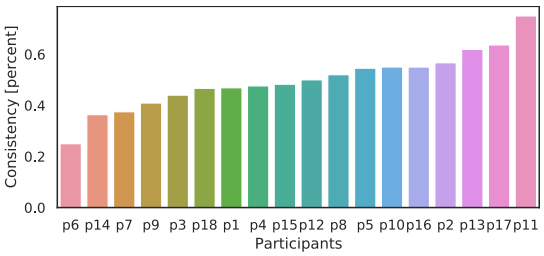
4.1 Intra- and Inter-Rater Metrics

As a first step, which will allow us to better understand the results, we evaluate the annotators' intra-rater consistency and their inter-rater reliability. This allows us to analyze similarity functions based on the annotator's noise as well as discover possible groups of similarity functions.

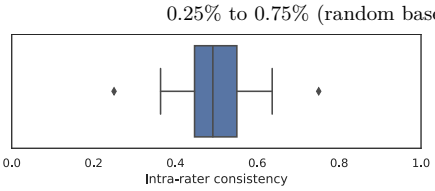
Intra-rater reliability. We determine participants' reply consistency so that we can consider the noise level of their replies in later analysis.

The procedure contains two identical sets of queries (Repeat #1 and Repeat #2) that we compare to calculate consistency as defined in Eq. 7. Repeat #1 consists of at least 20 queries, with additional queries for each skipped reply. Repeat #2 then repeats the exact same queries per user. We shuffle the samples in the UI to avoid memory effects and repeat the set only after other annotation tasks.

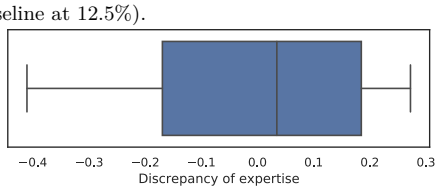
Fig. 3a shows each participant's consistency from a low of 0.25 to a high of 0.75. The typical reply consistency clusters of 0.498 ($\pm$0.11), see Fig. 3b. We see two outliers, one to the low (p6) and to the high end (p11). The consistency agrees with the participants' self-assessment of their expertise as 1 for p6 and 6 for p11 on a scale of 1 (novice) to 6 (expert). Next, we normalize the self-assessment between 0 and 1, and show the difference to measured consistency in Fig. 3c. The self-assessed expertise typically matches consistency within 1 step up or down on the scale. This validates the quantitatively measured consistency. Furthermore, this score may help decide whether to collect annotations from a participant, or whether the generic baseline model is the better-performing alternative (see Sec. 4.2.1).



(a) Intra-rater consistency on repeated queries is from 0.25% to 0.75% (random baseline at 12.5%).



(b) The intra-rater consistency centers around 49.8% (±0.11) with two outliers at the extremes.



(c) Survey self-assessment minus their measured consistency shows the small difference between qualitative and quantitative expertise.

Figure 3: The intra-rater consistency in (a) shows how reliable participants determine similarity in a fixed and repeated set of queries, and we show their distribution in b. (c) shows that participants' self-assessed expertise in the survey differs from measured consistency by 20% or 1.2 points on the scale of 6.

These results show that the participants answer queries inconsistently, as the mean consistency to find the same similar samples from eight in total is 0.49(±0.12). Additionally, the consistency measures identify unreliable oracles which help understand algorithmic performance better.

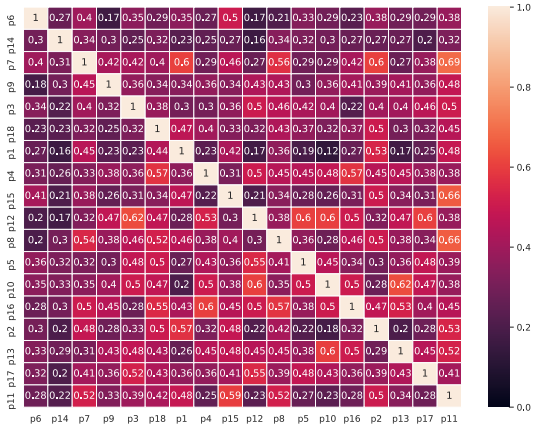


Figure 4: The inter-rater reliability on a fixed, repeated dataset shows whether participants form clusters of similarity functions, or whether they have orthogonal notions of similarity.

Inter-rater reliability. The subjective similarity functions may be part of clusters. Similar metrics could be learned from annotations if participants' concepts of similarity are close enough. The identification of such groups may be beneficial for analysis and fine-tuning.

The repeated sets of queries (Repeat #1) are identical for all participants. Only the actual number of replies may differ depending on skipped replies. Hence, it allows us to calculate the inter-rater reliability as

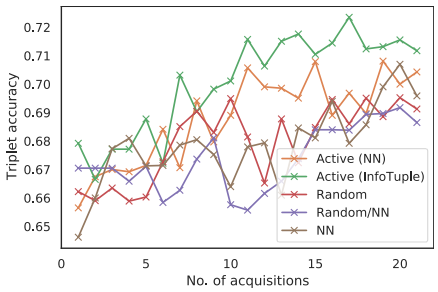


Figure 6: Triplet accuracy averaged over all users on the test set. We initialize the training data with the first *Fixed* dataset and then perform 20 acquisitions of size one. We train user-individual models and show the averages.

defined by Eq. 8. This way, we can identify sets of participants whose similarity functions overlap sufficiently to be considered as cluster members. We then use the reliability as the pairwise distance for hierarchical bottom-up clustering with maximum linkage.

We show a heat map of the pairwise reliability of the raters in Fig. 4 and a dendrogram of hierarchical clusters in Fig. 5. In the heatmap, we sort participants by their consistency from low to high. Higher inter-rater reliability is highlighted with brighter colors. Participants with lower consistency also show lower inter-rater reliability.

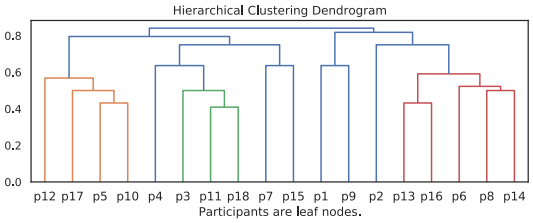


Figure 5: Hierarchial clustering dendrogram based on inter-rater reliability as pairwise distance.

The intra- and inter-rater scores appear to be similarly ranged distributions. That means that the replies of some participants may be similar enough to cluster them together, allowing us to treat the entire cluster as one individual. The dendrogram shows hierarchical clusters of agreeing participants. We search for clusters of inter-rater reliability that have similar linkage as the intra-rater consistency, and highlight clusters with a linking distance equivalent to 0.37 or above, e.g., the four participants [p5, p10, p12, p17] form one such cluster.

We can determine clusters of similarity functions from these results. Still, the noise of replies can be in a similar magnitude as the reliability between participants, which corroborates the need for an analysis of transfer learning within such clusters. We will conduct this in Sec. 4.4.

4.2 Triplet Accuracy

The gains in predictive accuracy per annotated sample demonstrate whether a tuple composition is informative for the model. The accuracy represents the usefulness of the model for the application. We hypothesize that an active learning sampling of tuples is most efficient in increasing accuracy.

For this, we calculate the triplet accuracy on user-specific test datasets over 20 acquisition steps. For each participant, we first fine-tune the pre-trained network on 20 annotated InfoTuples from their Repeat #1 phases (equivalent to 140 triplets). We evaluate three non-active tuple composition methods as baselines (Rnd, Random/NN, NN) in comparison to two active learning methods (Active-NN, InfoTuple). The baselines are constructed from replies to fixed queries, the active learners were not fixed but interactively adapted to users. The test sets are user-specific sets of 10 annotated InfoTuple that were sampled randomly.

The test in Fig. 6 sees the active learners ahead, with InfoTuple on top. Purely randomly composed tuples compare well, especially in the first half of the acquisition phase, but level out around 69% triplet accuracy.

The remaining composition methods, which were constructed at least partially from the pre-trained network's embedding neighborhood stagnate for most of the duration of the experiment. These results for triplet accuracy are for 40 InfoTuples (280 triplets) in total. We further investigate the ceiling of the triplet accuracy, which is likely due to the noisy annotations and inconsistent replies, by training and testing user-individual models with all 120 InfoTuples (or 840 triplets) from the different annotation phases. The average triplet accuracy then reaches 73.7% (±0.01), which is only marginally higher than previously. The low gains with three times as many annotations show the diminishing returns of larger data collections.

Given the limited consistency of replies and the lack of ground truth data, there is an upper limit to the model's ability to learn. However, the relatively higher scores of InfoTuple are consistent and lead to increased model accuracy, whereas the NN active learner comes in second but similarly to the baselines. In terms of efficiency, InfoTuple is clearly preferable.

4.2.1 Noisy Oracles

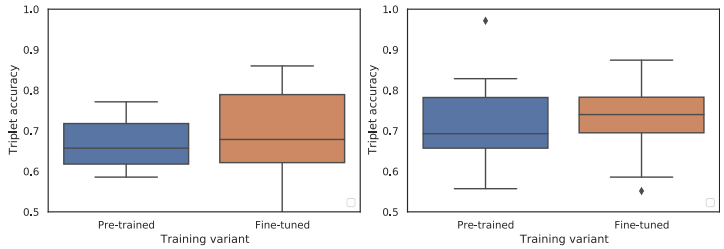
Consistency of replies affects triplet accuracy and influences the network's ability to learn user-specific similarity functions. A lower consistency is harder to train with, and a pre-trained network performs better on test data than a fine-tuned one. More consistent annotations, however, enable fine-tuning. The InfoTuple active learner benefits from more consistent annotations.

We compare the triplet accuracy for the pre-trained model with all fine-tuned models for each user. We again use the users' self-annotated test sets *Rnd* and compare the accuracy. Additionally, we split the participants into groups of lower and higher consistency. For each group, we compare the pre-trained model with one trained using the InfoTuple active learner to test fine-tuning with more consistent annotations.

We show the effect of consistency on the InfoTuple algorithm in Fig. 7. For annotators with low consistency, the pre-trained network often performs similarly or even better than the finetuned variant. The pre-trained model is a strong baseline for inconsistent annotators. Fig. 7b shows that consistent annotations allow the active learner to learn a similar or better model than the pre-trained baseline, with 74% for the finetuned case vs 69.2% pre-trained on the test set. Appendix A.3 reports dis-aggregated triplet accuracy per user, supporting the results.

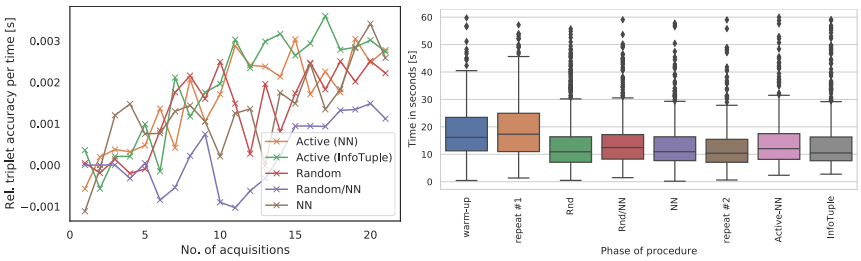
Generally, if the pre-trained network is already performing well on a test set, the fine-tuning with inconsistent data tends to decrease test triplet accuracy. It depends on the consistency of the similarity function used for annotations if fine-tuning is beneficial. For consistent participants, that also tend to be self-assessed experts, better user-specific similarity functions can be learned.

Furthermore, with InfoTuple as the AL sampler, we see strong increases in triplet accuracy over the pre-trained baseline (see Fig. 7). The most consistent $\frac{2}{3}$ participants lead to more stable learning and higher triplet accuracy, which is likely due to their more consistent replies or clearer notion of similarity (see Fig. 7b).



(a) Least consistent $\frac{1}{3}$ on Rnd. (b) Most consistent $\frac{2}{3}$ on Rnd.

Figure 7: This figure shows the effectiveness of fine-tuning via InfoTuple active learning in comparison to the pre-trained baseline for different levels of consistency. Participants are grouped according to their consistency, into a least consistent third and the most consistent two-thirds. Relative gains are greater for more consistent replies.



(a) Response Effectiveness over the number of acquisitions.

(b) The response times per phase, here as box plots over all users, show a relatively stable mean around 10 s for each phase of the procedure.

Figure 8: We present the triplet accuracy after applying Eq. 9 in (a) and show response times in (b).

4.3 Considering Time and Sampling Efficiency

We consider three different perspectives that shed some light on the relationship between triplet accuracy, time, and skipping. These metrics are relevant for determining the efficiency of the time spent, as well as the potential fatigue of participants.

4.3.1 Response Effectiveness

The relative return in accuracy on the invested amount of response time is a relevant criterion when choosing a sampling method. The time that users spend analyzing samples and forming decisions may also be indicative of how difficult a query is to reply to. This excludes the time that methods require to propose queries.

We evaluate the response time for all users by dividing the relative gains in predictive accuracy by the participant's average response time. This yields a relative improvement in accuracy per time spend on a response.

We normalize triplet accuracy using Eq. 9 and report the results in Fig. 9. There, we see the highest increase in triplet accuracy for InfoTuple, followed by the NN active learner. The Random baseline follows closely. Interestingly, the strongest method InfoTuple is also the one that is replied to quickest. The response times of the 5 methods in our study are: Random with 9.7 s, Random/NN with 10.98 s, NN with 10.35 s, and the active samplers NN with 11.19 s and InfoTuple with 9.48 s, see Fig. 8b for an overview and Fig 14 in Appendix A.3 for user-specific details.

The relative return in accuracy should not be bought with exorbitant amounts of time. Our results show that both active learners perform comparably well due to their higher relative gains in accuracy compared to the baselines, which speaks for them as tuple composition methods.

4.3.2 Total Effectiveness

While the response time itself is important for choosing a suitable sampling method, the overall time spent may also include an active method's computational overhead. The analysis is similar to Sec. 4.3.1 but also includes the time of methods to compute queries on top.

The algorithm execution times cause minor decreases in total effectiveness for the two Active Learning methods. The Nearest Neighbor active learner computes for 2.8 s ($\pm$ 0.59 s) and the InfoTuple active learner needs 6.11 s ($\pm$ 0.56 s). However, this depends mainly on the computational power and implementation efficiency. In this work, we optimized InfoTuple to decrease its complexity while still providing benefits over others.

Computation, such as the training of the neural network or the calculations performed by InfoTuple, impacts the accuracy-per-time efficiency. However, while we consider the cognitive load for participants to be taxing,

the waiting time between queries may be less demanding. In our user study, the additional waiting time of the active methods was still reported as *tough* by several participants and should be considered when selecting a method.

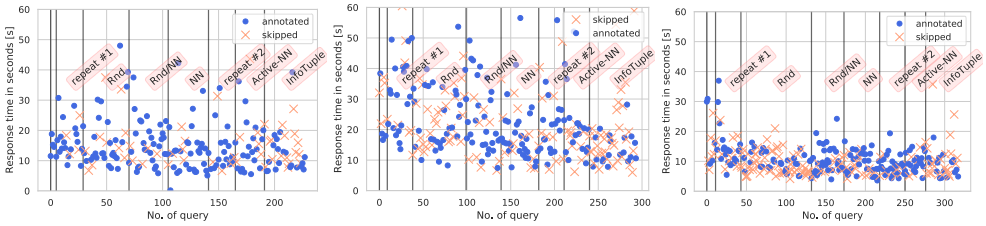
4.3.3 Label Effectiveness

We consider the number of skips during the labeling process as a potential source of participants' fatigue, because it takes longer to reach the set number of annotations. Hence, participants are queried repeatedly. Higher label effectiveness is to be preferred, see Eq. 11.

We present the least, the median, and the most consistent participants' timeline of skipped and annotated queries in Fig. 9 to highlight the prolonging effect, that a large number of skipped queries has on the duration of the experiment, and thus also on the participants' fatigue. In addition, we use Eq. 11 to calculate the label effectiveness of the different sampling methods. We report the details in Fig. 14 in Appendix A.3 and discuss the findings here.

The NN active learner combines an apparently easier-to-answer query composition with the benefits of active learning and increases accuracy (by the number of skipped queries) with $LE = 1.95\%$, followed by non-active with $LE = 1.39\%$. InfoTuple follows closely with $LE = 1.26\%$ due to its raw gains of accuracy. Even though more queries were skipped, responses were more informative overall. Random and Random/NN sampling score worst due to the high amount of skips or lower gains in accuracy with $LE = 0.75\%$ and $LE = 0.35\%$ respectively. Furthermore, we may infer that skipping unsuitable queries may be necessary to annotate more consistency, see $p6$ with $p15$ or $p11$ in Fig. 9. The mean percentage of skipped samples is 22.22% for NN queries, closely followed by Random/NN sampling (22.36%) and the NN active learner (24.53%). Both methods compose queries from close neighbors of the query anchor, i.e., likely similar samples. InfoTuple queries are skipped 50.1% of the time and random tuples form the rear with 53.97%.

The methods that primarily focus on NN sampling perform best due to the higher likelihood of positive samples in a query. InfoTuple samples from a larger pool of most informative samples, which may cause participants to skip more queries. However, its queries tend to be more informative so that InfoTuple keeps up with the other methods.



(a) Skips vs responses p6. (b) Skips vs responses p15. (c) Skips vs responses p11.

Figure 9: All response times for the least, the median, and the most consistent participant. We highlight the annotation phases.

4.4 Data Augmentation with Annotations

Participants' similarity functions may align well enough to profit from combining their annotations in order to augment their training dataset. This would increase sampling efficiency, as a larger collection of annotations would be available for warm-starting the training. Due to the comparable levels of intra-rater consistency and inter-rater reliability, such *clusters* seem realistic.

Our analysis selects the pair of participants with the highest inter-rater reliability. We exclude skipped samples and we select the pair with more consistent participants. we then initialize the training dataset with the combined set of both participants' Repeat #1 replies, fine-tune one model for each participant on their annotations selected by InfoTuple, and test on the participants' own test sets. Recall that since the queries,

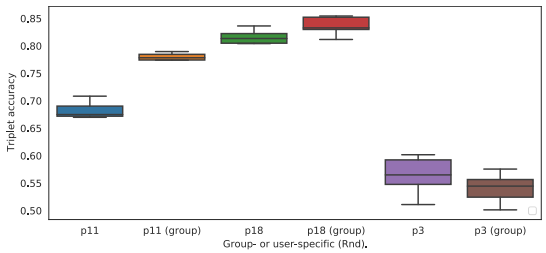


Figure 10: The effectiveness of grouping the training data from participants with similar consistency and notions of similarity. It can be beneficial to combine their warm-up training datasets to increase triplet accuracy.

that make up the Repeat #1 data are identical, the combined annotations augment the training with replies to skipped queries and can even contain different replies.

We select the participants $p3, p11$, and $p18$ according to the hierarchical clustering in Fig. 5, because $p11$ is the most consistent participant overall. The results in Fig. 10 show the respective participant's differences on the test set. Two of the participants benefit from additional annotations, the third sees a further decrease in accuracy.

Augmenting the warm-start dataset with the annotations of similar annotators can be beneficial. Our results for a triplet of participants, which also includes the highest inter-rater reliability, confirm it. This suggests that similarity functions may be clustered and used for transfer learning. However, our additional experiments with larger clusters yield diminishing returns and confirm our pre-study's findings, likely due to lower inter-rater reliability.

4.5 Discussion

RQ$_1$: How consistent and reliable are the participants?
Our evaluation shows that the participants are able to respond to the queries with relatively high quality, as their intra-rater consistency is in the range of 0.498 ($\pm$0.11) on average. Their self-assessment in the range of 1 to 6 is relatively consistent with a divergence of 20%, which we show indicates whether a user-specific data collection is beneficial. Furthermore, we found that noisy oracles affect the network's ability to learn user-specific similarity functions. However, even though the more consistent group benefited more from fine-tuning, the tendency does also points towards a smaller increase for less consistent annotators.

RQ$_2$: What is the best baseline heuristic to generate tuple queries?
The composition of tuple queries has implications beyond the learned model's triplet accuracy. Participants may take longer to respond or even find no response at all, which ultimately leads to fatigue. We show in Fig. 6 that the heuristics Random and NN yield a very similar triplet accuracy with Random/NN as a close third. However, once we account for the average response time, the gap between NN and Random on the one hand, and Random/NN on the other increases. We see two possible explanations. First, the Nearest Neighbor sampling generates queries from the relatively sparse Euclidean embedding, that are more likely to contain easy positives that participants select more quickly. Conversely, queries generated by the Random heuristic contain easy negative samples and are often quickly skipped, see Fig. 9. The skipping renders Random sampling less efficient, as the procedure's sampling phases require more queries to collect the same number of annotations, which leads to higher fatigue. Finally, the combination of Random/NN mixes easy positives and negatives and seems to require closer inspection by participants and is less sample efficient because of this, see Fig. 8b.

Overall, the NN heuristic balances the three metrics triplet accuracy, time spent on responses, and the number of skips and fatiguing of participants better than the other baselines Random and Random/NN.

RQ$_3$: Does an active sampling method perform better than baselines?
We show that annotations can be more valuable due to Active Learning, as the differences between ActiveNN and the baseline NN heuristics show. Furthermore, selecting informative samples with InfoTuple increases sample efficiency compared to ActiveNN, because a purely NN-query may not be maximally informative due to the sparse similarity of our problem domain. Response times are comparable to the heuristic baselines. Interestingly, the increases in accuracy per skipped sample are in favor of the ActiveNN sampler compared to InfoTuple, while the absolute triplet accuracy of models trained using InfoTuple is higher. Additionally, we show that the gained accuracy per time justifies the computational complexity of AL. Overall, Active Learning and specifically InfoTuple perform better than heuristic baselines.

Quality of Annotations
The quality of annotations is an important factor in machine learning. Specifically for Active Learning, applications typically deal with expensive domain experts, that may perform more reliably, but whose time is more expensive. Alternatively, non-expert annotators may seem more cost-effective but are likely to be less reliable (Wu et al., 2020). In our study, we address the scenario of mixed expertise and evaluate the deep metric learning objective with respect to quantitative and self-assessment. We show that self-assessment can be predictive of the usefulness of Active Learning compared to a strong baseline metric learner. We show that transfer learning can be applied successfully using more consistent, and potentially also semantically similar annotations.

Broader Applicability of the Proposed Method
The ideas proposed in this paper are twofold. First, the proposed deep active learning method may be applied to other tasks. Learning similarity of trajectory data is a common problem, e.g., in other team sports like basketball (Sha et al., 2016), hockey (Chen et al., 2005) or Australian football (Alexander et al., 2022), in animal farming (Chen et al., 2005), or in personal mobility, traffic or public transportation (Shen et al., 2019; Yadamjav et al., 2020; Fernández et al., 2017). Still, the propose method uses a flexible learner that may be adapted for other data than trajectories, e.g., learning similarity in image datasets such as Food73 (Wilber et al.) or ranking the age of faces (Liu et al.) and learning similarity between texts (Neculoiu et al.). Generally, the proposed method may be considered with the aim to train (or fine-tune) a generalizing Metric Learner from few relative comparisons of large tuple size with the help of a human-in-the-loop.

Broader Evaluation of Active Learning
The second idea that we propose is our call for a broader evaluation of new active learning algorithms. In Active Learning, the predominant performance metric seems to be accuracy per labeled sample. There are, however, additional practically important considerations besides accuracy. Experimental platforms, such as NEXT (Jamieson et al., 2015), capture detailed statistics on, e.g., network response time, timing data that helps to estimate human fatigue, and also the quality of annotations, i.e., via intra-rater agreement. More recent AL studies Canal et al. (2020) additionally analyze the tasks' difficulty and response time. Hence, in our work, we do not only rank methods by accuracy. Our analysis of effective accuracy over (total) time spent on responding, and the efficiency of accuracy by skipped queries support the results. The adapted InfoTuple method, which evaluates the Nearest Neighbors sampled from the neural network embedding, shows to be less fatiguing as well.

5 Related Work

Many approaches for deep metric learning use triplets to learn generalizing representations (Hoffer & Ailon, 2015; Xuan et al., 2020). Recently, Xuan et al. (2020) introduced the idea of sampling triplets with easy positive examples. This addresses the issue of over-clustering of the learned embedding, and it is tolerant of high-class variance. This is highly relevant to our study because it motivates the composition of tuple queries. Our study evaluates the effect of easy positive and negative samples with a human-in-the-loop.

Active learning uses insights into the data distribution or model views to select the sample(s) that best improves the model's fit of the underlying data distribution. We can group approaches for deep learning into different categories. Some estimate model uncertainty to select the most informative (Houlsby et al., 2011; Kirsch et al., 2019) or uncertain samples (Gal et al., 2017; Beluch et al., 2018), others select representative samples using a covering set (Sener & Savarese, 2018) or combine diverse and uncertain selections (Ash

et al., 2020). Nadagouda et al. (2022) use information theory to perform active metric learning and active classification on a deep probabilistic model that uses Monte Carlo dropout sampling (Gal & Ghahramani, 2016) as an approximation of uncertainty. They sample the most informative NN queries from all possible ones. Similarly, Canal et al. (2020) learn an ordinal embedding by maximizing the mutual information of queries to an oracle. Their work is based on a probabilistic model, fitted to relative comparisons, that learns the ordinal embedding. Both Nadagouda et al. (2022) and Canal et al. (2020) use larger tuple sizes than three, which improves sample efficiency and also provides more context for easier annotations.

Information retrieval from multi-agent trajectory datasets has to first solve the assignment problem of these agents' trajectories (Sha et al., 2017; 2016; Löffler et al., 2020), and second, provide a quickly searchable representation (Sha et al., 2016; 2017; Di et al., 2018; Löffler et al., 2020). Di et al. (2018) additionally learn ranking from humans. They estimate the assignment of agents (players) within a tree-structure (Sha et al., 2017) and then train a convolutional autoencoder on 2D trajectory plots to extract features. In contrast, we train one neural network Löffler et al. (2020), that learns to solve the assignment problem and produces a lower dimensional embedding, such that additional tree structures are redundant and search is accelerated. Di et al. (2018) learn ranking from participants of a click-through study, that collects only pairwise comparisons, and trains a linear rankSVM on these tuples and the extracted features of the autoencoder. This approach differs from this paper, as we optimize the network's embedding directly. However, the embedding may be used for on-top learning to rank approaches (Di et al., 2018) or even active embedding search (Canal et al., 2019).

6 Conclusion

In this work, we studied how to learn unknown similarity functions, that humans innately use to measure similarity between instances in an unlabeled, unstructured dataset. To learn these metrics from few annotations, we adapted mining easy positive triplets from a query sample's neighborhood in a neural network's embedding space and applied InfoGain to construct the most meaningful queries from the sparse similarity of a football trajectory dataset. Our user study shows the benefits of this method and provides a nuanced evaluation with respect to accuracy and the practical considerations of the effectiveness of Active Learning with a human-in-the-loop: response, total and label effectiveness. An accompanying survey sheds light on the mixed expertise of participants, their diverse notions of similarity, and their relation to a strong metric learning baseline. Future work may leverage the user-specific ordinal embedding to perform an iterative active search, that can further improve information retrieval.

Author Contributions

The main author Christoffer Löffler implemented algorithms, designed and conducted experiments, wrote and revised the manuscript. Kion Fallah and Stefano Fenu contributed to the theoretical framework and experimental design, offering insights. Kion Fallah and Dario Zanca provided manuscript feedback. Christopher Rozell guided the research, including experimental and algorithmic design. Christopher Mutschler supervised and helped refine the experiments and methodology, and provided manuscript feedback. Björn Eskofier supervised the project and provided manuscript feedback.

Acknowledgments

We would like to acknowledge support for this project from the IFI programme of the German Academic Exchange Service (DAAD), the Bavarian Ministry of Economic Affairs, Infrastructure, Energy and Technology as part of the Bavarian project Leistungszentrum Elektroniksysteme (LZE) and the Center for Analytics-Data-Applications (ADA-Center) within the framework of "BAYERN DIGITAL II". We thank the study participants for their valuable contributions. Special thanks to Nicolas Witt and Robert Marzilger for their support in organizing the study.

References

Jeremy P. Alexander, Karl B. Jackson, Timothy Bedin, Matthew A. Gloster, and Sam Robertson. Quantifying congestion with player tracking data in Australian football. 17(8):e0272657, 2022. ISSN 1932-6203. doi: 10.1371/journal.pone.0272657.

Jordan T. Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. In *International Conference on Learning Representations (ICLR)*, Virtual Conference, Formerly Addis Ababa Ethiopia, 2020.

Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

William H Beluch, Tim Genewein, Andreas Nürnberger, and Jan M Köhler. The power of ensembles for active learning in image classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9368–9377, Salt Lake City, UT, 2018.

Jurgen Bernard, Marco Hutter, Matthias Zeppelzauer, Dieter Fellner, and Michael Sedlmair. Comparing Visual-Interactive Labeling with Active Learning: An Experimental Study. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):298–308, January 2018. ISSN 1077-2626, 1941-0506, 2160-9306. doi: 10.1109/TVCG.2017.2744818.

Alina Bialkowski, Patrick Lucey, Peter Carr, Yisong Yue, Sridha Sridharan, and Iain Matthews. Large-scale analysis of soccer matches using spatiotemporal tracking data. In *International Conference on Data Mining*, pp. 725–730, Los Alamitos, CA, USA, 2014. IEEE, IEEE Computer Society.

Gregory Canal, Andy Massimino, Mark Davenport, and Christopher Rozell. Active embedding search via noisy paired comparisons. In *International Conference on Machine Learning*, pp. 902–911. PMLR, 2019.

Gregory Canal, Stefano Fenu, and Christopher Rozell. Active ordinal querying for tuplewise similarity learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):3332–3340, Apr. 2020. doi: 10.1609/aaai.v34i04.5734.

Enrique Canessa, Sergio E Chaigneau, Sebastián Moreno, and Rodrigo Lagos. Informational content of cosine and other similarities calculated from high-dimensional conceptual property norm data. *Cognitive Processing*, 21(4):601–614, 2020.

Gal Chechik, Varun Sharma, Uri Shalit, and Samy Bengio. Large scale online learning of image similarity through ranking. *Journal of Machine Learning Research*, 11(3), 2010.

Lei Chen, M. Tamer Özsu, and Vincent Oria. Robust and fast similarity search for moving object trajectories. In *Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data - SIGMOD '05*, pp. 491. ACM Press, 2005. ISBN 978-1-59593-060-6. doi: 10.1145/1066157.1066213.

Mingyang Di, Diego Klabjan, Long Sha, and Patrick Lucey. Large-scale adversarial sports play retrieval with learning to rank. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 12(6):1–18, 2018.

Esther Calvo Fernández, José Manuel Cordero, George Vouros, Nikos Pelekis, Theocharis Kravaris, Harris Georgiou, Georg Fuchs, Natalya Andrienko, Gennady Andrienko, Enrique Casado, et al. Dart: a machine-learning approach to trajectory prediction and demand-capacity balancing. *SESAR Innovation Days, Belgrade*, pp. 28–30, 2017.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In Maria Florina Balcan and Kilian Q. Weinberger (eds.), *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 1050–1059, New York, New York, USA, 20–22 Jun 2016. PMLR.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. In *34th International Conference on Machine Learning (ICML)*, Sydney, Australia, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. In *International workshop on similarity-based pattern recognition*, pp. 84–92. Springer, 2015.

Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745*, 2011.

Kevin G Jamieson, Lalit Jain, Chris Fernandez, Nicholas J Glattard, and Rob Nowak. Next: A system for real-world development, evaluation, and application of active learning. *Advances in neural information processing systems*, 28, 2015.

Andreas Kirsch, Joost van Amersfoort, and Yarin Gal. Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 7026–7037. Vancouver, Canada, 2019.

Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2 (1-2):83–97, 1955.

Colin Lea, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal convolutional networks: A unified approach to action segmentation. In *European conference on computer vision*, pp. 47–54. Springer, 2016.

Hao Liu, Jiwen Lu, Jianjiang Feng, and Jie Zhou. Ordinal Deep Feature Learning for Facial Age Estimation. In *2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, pp. 157–164. IEEE. ISBN 978-1-5090-4023-0. doi: 10.1109/FG.2017.28.

Christoffer Löffler, Christian Nickel, Christopher Sobel, Daniel Dzibel, Jonathan Braat, Benjamin Gruhler, Philipp Woller, Nicolas Witt, and Christopher Mutschler. Automated quality assurance for hand-held tools via embedded classification and automl. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML-PKDD)*, Ghent, Belgium, 2020.

Christoffer Löffler, Luca Reeb, Daniel Dzibela, Robert Marzilger, Nicolas Witt, Björn M. Eskofier, and Christopher Mutschler. Deep siamese metric learning: A highly scalable approach to searching unordered sets of trajectories. *ACM Trans. Intell. Syst. Technol.*, 13(1), nov 2021. ISSN 2157-6904. doi: 10.1145/3465057.

Michael Lohaus, Philipp Hennig, and Ulrike von Luxburg. Uncertainty estimates for ordinal embeddings. *arXiv preprint arXiv:1906.11655*, 2019.

Leland McInnes, John Healy, Nathaniel Saul, and Lukas Grossberger. Umap: Uniform manifold approximation and projection. *The Journal of Open Source Software*, 3(29):861, 2018.

Namrata Nadagouda, Austin Xu, and Mark A. Davenport. Active metric learning and classification using similarity queries. *arxiv preprint arXiv:2202.01953*, 2022. doi: 10.48550/ARXIV.2202.01953.

Paul Neculoiu, Maarten Versteegh, and Mihai Rotaru. Learning Text Similarity with Siamese Recurrent Networks. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pp. 148–157. Association for Computational Linguistics. doi: 10.18653/v1/W16-1617.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations (ICLR)*, Vancouver, Canada, 2018.

Burr Settles. Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2009.

Long Sha, Patrick Lucey, Yisong Yue, Peter Carr, Charlie Rohlf, and Iain Matthews. Chalkboarding: A new spatiotemporal query paradigm for sports play retrieval. In *Proceedings of the 21st International Conference on Intelligent User Interfaces*, pp. 336–347, 2016.

Long Sha, Patrick Lucey, Stephan Zheng, Taehwan Kim, Yisong Yue, and Sridha Sridharan. Fine-grained retrieval of sports plays using tree-based alignment of trajectories. *arXiv preprint arXiv:1710.02255*, 2017.

Zhi-Hao Shen, W. Du, X. Zhao, and Jianhua Zou. Retrieving similar trajectories from cellular data at city scale. *ArXiv*, abs/1907.12371, 2019.

Edgar Simo-Serra, Eduard Trulls, Luis Ferraz, Iasonas Kokkinos, Pascal Fua, and Francesc Moreno-Noguer. Discriminative Learning of Deep Convolutional Feature Point Descriptors. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 118–126, Santiago, Chile, December 2015. IEEE. ISBN 978-1-4673-8391-2. doi: 10.1109/ICCV.2015.22.

Omer Tamuz, Ce Liu, Serge Belongie, Ohad Shamir, and Adam Tauman Kalai. Adaptively learning the crowd kernel. *arXiv preprint arXiv:1105.1033*, 2011.

Laurens Van Der Maaten and Kilian Weinberger. Stochastic triplet embedding. In *2012 IEEE International Workshop on Machine Learning for Signal Processing*, pp. 1–6. IEEE, 2012.

Gang Wang, David Forsyth, and Derek Hoiem. Comparative object similarity for improved recognition with few or no examples. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3525–3532. IEEE, 2010.

Michael J Wilber, Iljung S Kwak, and Serge J Belongie. Cost-Effective HITs for Relative Similarity Comparisons.

Jian Wu, Victor S Sheng, Jing Zhang, Hua Li, Tetiana Dadakova, Christine Leon Swisher, Zhiming Cui, and Pengpeng Zhao. Multi-label active learning algorithms for image classification: Overview and future promise. *ACM Computing Surveys (CSUR)*, 53(2):1–35, 2020.

Hong Xuan, Abby Stylianou, and Robert Pless. Improved Embeddings with Easy Positive Triplet Mining. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 2463–2471, Snowmass Village, CO, USA, March 2020. IEEE. ISBN 978-1-72816-553-0. doi: 10.1109/WACV45572.2020.9093432.

Munkh-Erdene Yadamjav, Zhifeng Bao, Baihua Zheng, Farhana M. Choudhury, and Hanan Samet. Querying recurrent convoys over trajectory data. *ACM Trans. Intell. Syst. Technol.*, 11(5), August 2020. ISSN 2157-6904.

Yu Zheng. Trajectory Data Mining: An Overview. 6(3):1–41, 2015. ISSN 2157-6904, 2157-6912. doi: 10.1145/2743025.

A Appendix

A.1 Pre-Study

We conducted a pre-study to set the size of InfoTuple and also the number of required annotations for user-individual model training. We furthermore have discovered that a model would not generalize well from the complete, diverse group to one hold-out user. We also collected feedback on the introduction (tutorial) and query format, which helped to improve it and to avoid confusion among participants.

Size of InfoTuple. The pre-study was conducted with 7 participants, that were split into two groups A with 4 participants and B with 3 participants. Group A was queried with tuples of size 7 (1 head, 6 body), and group B with smaller tuples of size 4 (1 head, 3 body). We monitored response times and found that both groups responded within a similar amount of time. Furthermore, participants of group B reported that they often could not select a similar sample. Hence, we determined that a larger tuple size would not negatively impact response times, but would provide additional context that would help respond to the query, consistent with Canal et al. Canal et al. (2020).

Count of InfoTuple. Second, we set the required number of annotated triplets by, first, collecting 500 annotated triplets per participant, and second, fine-tuning a model for each in increasing sub-sets sizes, that

group	notion	example
semantic	team movement	direction (attack or defense); dynamic or static
	ball movement	pass or shot; short or long; at rest
	semantic category	standards (kickoff); situations (pressuring, passing, shot); consistency of category
	team formation	compact; shifting; offensive or defensive
	area	which third; which flank
proximity	player and ball	Are there players near the ball?
	directions	direction of ball trajectory
	length of trajectory	
	distances	spatial proximity similar in candidate scenes

Table 1: Qualitative clustering of responses to questionnaire into *semantic* and *proximity*.

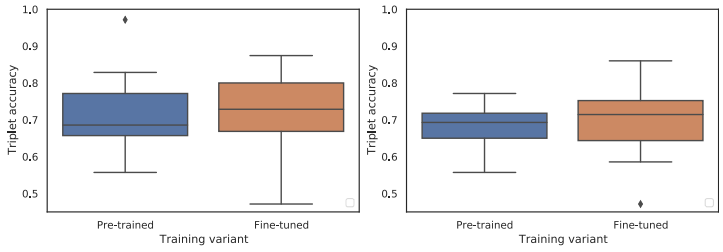
we randomly sampled from the total pool. The increase in accuracy converged over the increasing number of training data. Hence, we chose to collect 20 InfoTuple of size 8 (140 triplets) per method, as more showed diminishing returns.

Individual or general model. Lastly, we observed that we were not able to train one model by combining the training data from all participants into one large pool. We used leave-one-out cross-validation to train on all but one participant and then tested on the remaining participant as a hold-out test dataset. The *general* model did not achieve as good results as training an individual model. Hence, we designed the user study such that individual models were trained.

A.2 Notions of Similarity

We hypothesize that different notions of similarity may either already be captured by the Euclidean distance metric, or can be learned better from annotations. Recall that we have collected additional metadata on participants' notion of similarity using a questionnaire, see Sec. 3.2. This may help distinguish participant groups.

We test this hypothesis by analyzing the participants' responses to the questionnaire and extract two sets of rules, that we categorize as either *semantic* or *proximity*. Semantic rules describe the scenes' semantics whereas proximity rules largely ignore such interpretations. They instead pay closer attention to proximity or only to subsets of trajectories. To reduce bias, we ask two raters to assign participants to either group and then have the raters resolve their disagreements through an analytical discussion. Here, we compare the triplet accuracy of the agreed-upon group compositions.



(a) *proximity* notion on Rnd. (b) *semantic* notion on Rnd.

Figure 11: This figure distinguishes two types of similarity and shows the effectiveness of fine-tuning via InfoTuple active learning in comparison to the pre-trained baseline. Participants are grouped by their qualitative description of similarity, that is either proximity or semantic.

We show the extracted groups in Table 1 and each group's triplet accuracy in Fig. 11. We assign participants to either group if they describe a specific notion or provide examples that match it. The resulting *proximity*

group contains 8 participants and the *semantic* group 8, too. We excluded a participant that did not describe their notion and another that did not fit into any. We observe that high self-assessed expertise correlates with an assignment to the semantic group. Furthermore, we see an increase in triplet accuracy from pre-trained models to fine-tuned models for both groups in Fig. 11.

The increase in triplet accuracy is visible for both the semantic and the proximity groups. Hence, we can not show a difference from qualitative clustering of notions of similarity. However, fine-tuning can outperform the Siamese baseline in both cases, because the Euclidean similarity metric does not adequately capture either group's similarity function. In future work, alternative interpretations of qualitative and quantitative data may find a more suitable (sub) group segmentation.

A.3 Additional Results

Fig. 12 shows the effectiveness of fine-tuning compared to the Siamese baseline per participant. Fig. 13 shows every participant's response times for each phase. Finally, Fig. 14 shows the adapted triplet accuracy for three variants of effectiveness: response, total and label effectiveness.

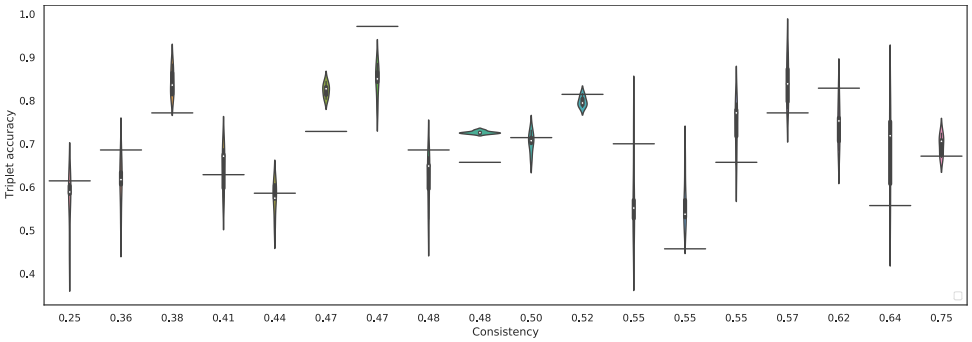


Figure 12: The effectiveness of all fine-tuning variants (violin plot) in comparison to the pre-trained Siamese network (horizontal bar) shows the impact of inconsistent oracles and a performance ceiling for the relatively noisy train and test data.

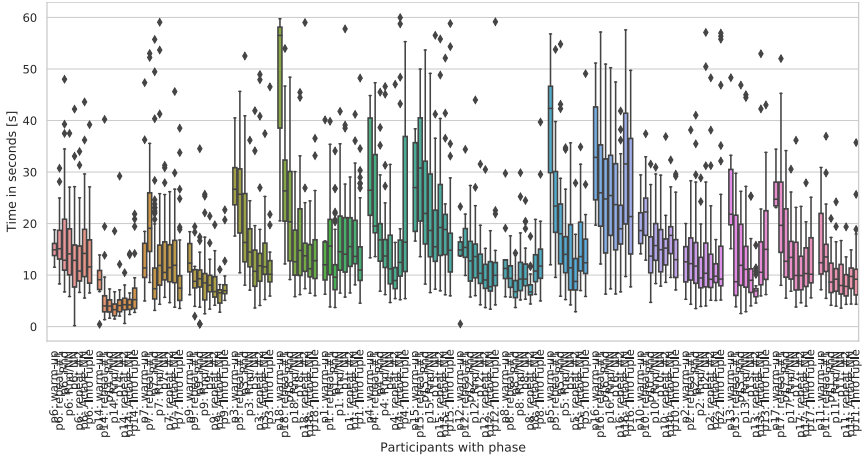
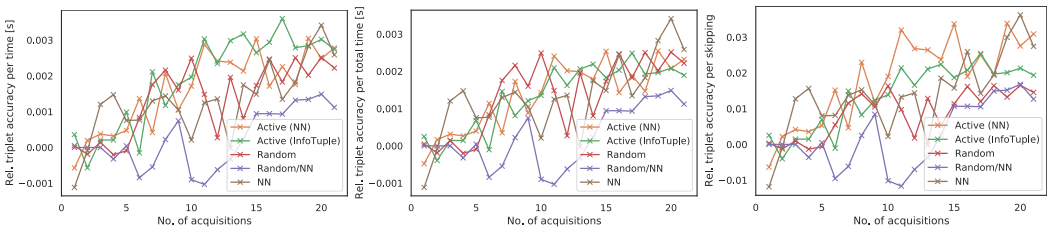


Figure 13: We show all users' response times for each phase ordered by consistency.



(a) Accuracy per response time. (b) Accuracy per total time. (c) Accuracy per skipped samples.

Figure 14: We average the results for all users on the test dataset. (a) shows the triplet accuracy per response time spent. (b) shows the triplet accuracy per time including compute time, which depends on compute resources and may limit the experiment's scale. (c) shows the triplet accuracy over the number of skips, which can cause fatigue.

A.4 Evaluation of Ellipse Template

We present the experimental comparison of our proposed ellipse template matching algorithm from Sec. 2 with the assignment methods proposed by Löffler et al. (2021). Their best-performing method uses an over-determined role position assignment, that introduces sparsity in the inputs of the Siamese network because there are more positions in the template than trajectories in the scene. Our proposed ellipse template has the same number of positions as there are trajectories, and we fit the template to each scene, see Sec. 2 for details.

We reproduce the results of Löffler et al. (2021) and compare them with our proposed template matching algorithm. For that, we use a Siamese network with an embedding size of 64, pre-train it on 304 games (see Section 3.4), and perform an evaluation of ours and their method on the same subset of the dataset that we use in our main study. We follow their evaluation scheme and calculate the structural correspondence between the original data and the learned representations. They define the Mean Absolute Percentage Error (MAPE) as the expected absolute error relative to the ground truth, as

$$\text{MAPE}_{\mathcal{D}} = \frac{1}{|\mathcal{D}|} \sum_{i=1}^{|\mathcal{D}|} \left| \frac{d_{ens}(\boldsymbol{X}_{i,1}, \boldsymbol{X}_{i,2}) - \hat{d}_{ens}(f(\boldsymbol{X}_{i,1}), f(\boldsymbol{X}_{i,2}))}{d_{ens}(\boldsymbol{X}_{i,1}, \boldsymbol{X}_{i,2})} \right|. \tag{12}$$

where they sample and compare pairs of matrices $\boldsymbol{X}_{i,1}$ and $\boldsymbol{X}_{i,2}$ from a dataset of matrix pairs $\mathcal{D}$. We adapt our notation from Sec. 2 slightly, such that $\boldsymbol{X}_{i,1}$ refers to the first matrix of the ith pair and $\boldsymbol{X}_{i,2}$ to the second matrix. Likewise, $f(\boldsymbol{X}_{i,1})$ refers to the learned representation of $\boldsymbol{X}_{i,1}$.

For their best performing method role position, Löffler et al. (2021) report a MAPE of 2.66% on their validation and 2.68% on their test set. Paired with the embedding size of 64, this also achieves speedy retrieval due to the lower dimensionality. We reproduce a similar MAPE of 2.669% on our dataset, which is in line with their results. In comparison, our ellipse template achieved a MAPE of 2.31% with the same hyperparameters. This clearly shows the benefits of our proposed method, as its error is 0.359% lower, and at the same time, it eliminates input sparsity.

A.5 Instructions

The user study uses a web-based annotation tool, that begins with a landing page with a short introduction. Participants are first presented with an introduction that familiarizes them with the data, the task, and the mechanics of the annotation tool. Fig. 15 shows the explanation of the data and description of details. Fig. 16 then describes the participant's task in this study. Finally, Fig. 17 shows an example of a query and once more explains the mechanics of the annotation process. After this participants begin annotating the warm-up samples with a button press.

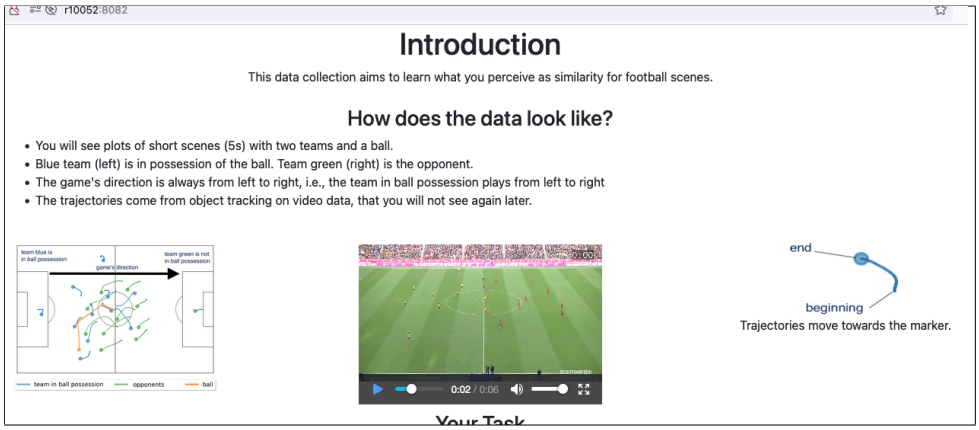


Figure 15: The landing page begins with an introduction to the dataset.

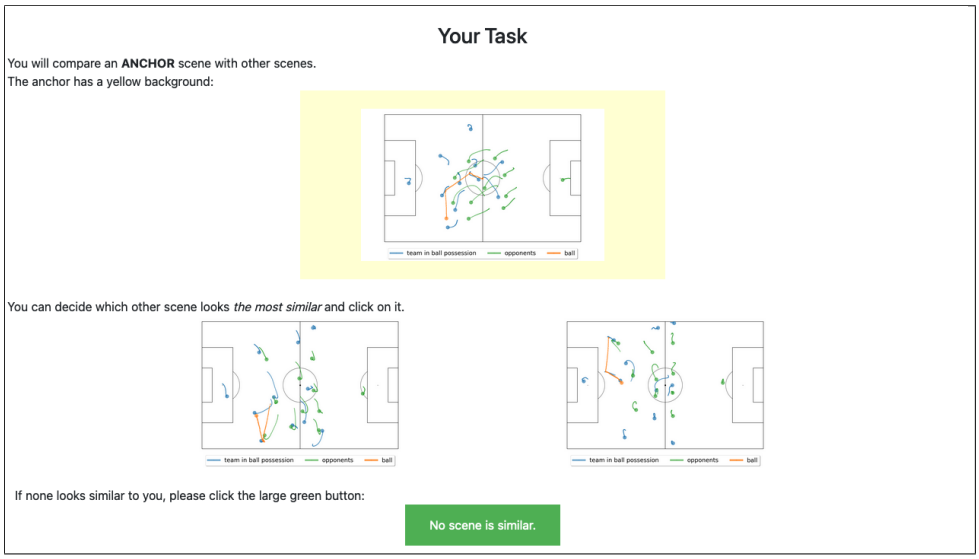


Figure 16: The next part describes the task and familiarizes participants with their options.

Figure 17: An example concludes the explanation and previews the annotation task that follows.

Bibliography

[1] Sha, L.; Lucey, P.; Yue, Y.; Carr, P.; Rohlf, C.; Matthews, I.: *Chalkboarding: A New Spatiotemporal Query Paradigm for Sports Play Retrieval*. In: *Proceedings of the 21st International Conference on Intelligent User Interfaces*. Sonoma, CA: ACM, 2016, 336–347.

[2] Settles, B.: *Active Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Cham: Springer International Publishing, 2012.

[3] Ren, P.; Xiao, Y.; Chang, X.; Huang, P.-Y.; Li, Z.; Gupta, B. B.; Chen, X.; Wang, X.: A Survey of Deep Active Learning. In: *ACM Computing Surveys* 54.9 (2021), 180:1–180:40.

[4] Tamuz, O.; Liu, C.; Belongie, S.; Shamir, O.; Kalai, A. T.: Adaptively Learning the Crowd Kernel. In: *Proceedings of the 28th International Conference on Machine Learning* (2011), 673–680.

[5] Canal, G.; Fenu, S.; Rozell, C.: Active Ordinal Querying for Tuplewise Similarity Learning. In: *34th AAAI Conference on Artificial Intelligence* (2020), 3332–3340.

[6] Priyadarshini, K.; Chaudhuri, S.; Borkar, V.; Chaudhuri, S.: *A Unified Batch Selection Policy for Active Metric Learning*. In: *Machine Learning and Knowledge Discovery in Databases. Research Track*. Ed. by Oliver, N.; Pérez-Cruz, F.; Kramer, S.; Read, J.; Lozano, J. A. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2021, 599–616.

[7] Sha, L.; Lucey, P.; Zheng, S.; Kim, T.; Yue, Y.; Sridharan, S.: *Fine-Grained Retrieval of Sports Plays Using Tree-Based Alignment of Trajectories*. In: *11th ACM International Conference on Web Search and Data Mining*. Marina Del Rey, CA, 2018.

[8] Di, M.; Klabjan, D.; Sha, L.; Lucey, P.: Large-Scale Adversarial Sports Play Retrieval with Learning to Rank. In: *ACM Transactions on Knowledge Discovery from Data* 12.6 (2018), 1–18.

[9] Kirsch, A.; van Amersfoort, J.; Gal, Y.: *BatchBALD: Efficient and Diverse Batch Acquisition for Deep Bayesian Active Learning*. In: *Advances in Neural Information Processing Systems*. Vol. 32. Curran Associates, Inc., 2019.

[10] Woodward, M.; Finn, C.: *Active One-shot Learning*. In: *Conference on Neural Information Processing System, Workshop on Deep Reinforcement Learning*. Barcelona, Spain, 2016.

[11] Fang, M.; Li, Y.; Cohn, T.: *Learning How to Active Learn: A Deep Reinforcement Learning Approach*. In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics, 2017, 595–605.

[12] Haußmann, M.; Hamprecht, F.; Kandemir, M.: *Deep Active Learning with Adaptive Acquisition*. In: *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. Macao, China: AAAI Press, 2019, 2470–2476.

[13] Liu, Z.; Wang, J.; Gong, S.; Tao, D.; Lu, H.: *Deep Reinforcement Active Learning for Human-in-the-Loop Person Re-Identification*. In: *2019 IEEE/CVF International Conference on Computer Vision*. Seoul, South Korea: IEEE, 2019, 6121–6130.

[14] Casanova, A.; Pinheiro, P. O.; Rostamzadeh, N.; Pal, C. J.: *Reinforced Active Learning for Image Segmentation*. In: *International Conference on Learning Representations*. New Orleans, LA, 2019.

[15] Nadagouda, N.; Xu, A.; Davenport, M. A.: *Active Metric Learning and Classification Using Similarity Queries*. In: *Human in the Loop Learning (HiLL) Workshop at NeurIPS 2022*. New Orleans, LA, 2022.

[16] Xuan, H.; Stylianou, A.; Pless, R.: *Improved Embeddings with Easy Positive Triplet Mining*. In: *IEEE Winter Conference on Applications of Computer Vision*. Snowmass Village, CO: IEEE, 2020, 2463–2471.

[17] Ilg, E.; Mayer, N.; Saikia, T.; Keuper, M.; Dosovitskiy, A.; Brox, T.: *FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks*. In: *IEEE Conference on Computer Vision and Pattern Recognition*. Honolulu, HI: IEEE, 2017, 1647–1655.

[18] Yadamjav, M. E.; Bao, Z.; Zheng, B.; Choudhury, F. M.; Samet, H.: Querying Recurrent Convoys over Trajectory Data. In: *ACM Transactions on Intelligent Systems and Technology* 11.5 (2020).

[19] Johnson, J.; Douze, M.; Jégou, H.: Billion-Scale Similarity Search with GPUs. In: *IEEE Transactions on Big Data* 7.3 (2021), 535–547.

[20] Bialkowski, A.; Lucey, P.; Carr, P.; Yue, Y.; Sridharan, S.; Matthews, I.: *Large-Scale Analysis of Soccer Matches Using Spatiotemporal Tracking Data*. In: *IEEE International Conference on Data Mining*. Shenzhen, China: IEEE, 2014, 725–730.

[21] Monge, G.: Mémoire Sur La Théorie Des Déblais et Des Remblais. In: *Mem. Math. Phys. Acad. Royale Sci.* (1781), 666–704.

[22] Vaserštein, L. N.: Markov processes over denumerable products of spaces, describing large systems of automata. In: *Problemy Peredači Informacii.* 5.4 (1969), 64–72.

[23] Kuhn, H. W.: The Hungarian Method for the Assignment Problem. In: *Naval Research Logistics Quarterly*. Vol. 2. 1955, 83–97.

[24] Wang, S.; Bao, Z.; Culpepper, J. S.; Cong, G.: A Survey on Trajectory Data Management, Analytics, and Learning. In: *ACM Computing Surveys* 54.2 (2022), 1–36.

[25] Senin, P.: *Dynamic Time Warping Algorithm Review*. Honolulu, HI: Information and Computer Science Department, University of Hawaii at Manoa, 2008.

[26] Mahalanobis, P. C.: On the Generalised Distance in Statistics. In: *Proceedings of the National Institute of Sciences of India* 2.1 (1936), 49–55.

[27] Bellman, R.: Adaptive Control Processes. A Guided Tour. In: vol. 255. XVI. Princeton University Press, 1961.

[28] Canessa, E.; Chaigneau, S. E.; Moreno, S.; Lagos, R.: Informational Content of Cosine and Other Similarities Calculated from High-Dimensional Conceptual Property Norm Data. In: *Cognitive Processing* 21.4 (2020), 601–614.

[29] Das, A.; Yenala, H.; Chinnakotla, M.; Shrivastava, M.: *Together We Stand: Siamese Networks for Similar Question Retrieval*. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, 2016, 378–387.

[30] Chopra, S.; Hadsell, R.; LeCun, Y.: *Learning a Similarity Metric Discriminatively, with Application to Face Verification*. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Vol. 1. 2005, 539–546 vol. 1.

[31] Hoffer, E.; Ailon, N.: *Deep Metric Learning Using Triplet Network*. In: *Similarity-Based Pattern Recognition*. Ed. by Feragen, A.; Pelillo, M.; Loog, M. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2015, 84–92.

[32] Weinberger, K. Q.; Saul, L. K.: Distance Metric Learning for Large Margin Nearest Neighbor Classification. In: *Journal of Machine Learning Research* 10.9 (2009), 207–244.

[33] Chechik, G.; Sharma, V.; Shalit, U.; Bengio, S.: Large Scale Online Learning of Image Similarity through Ranking. In: *Pattern Recognition and Image Analysis*. Ed. by Araujo, H.; Mendonça, A. M.; Pinho, A. J.; Torres, M. I. Vol. 5524. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, 11–14.

[34] Wang, J.; Song, Y.; Leung, T.; Rosenberg, C.; Wang, J.; Philbin, J.; Chen, B.; Wu, Y.: *Learning Fine-Grained Image Similarity with Deep Ranking*. In: *IEEE Conference on Computer Vision and Pattern Recognition*. Columbus, OH: IEEE, 2014, 1386–1393.

[35] Beluch, W. H.; Genewein, T.; Nürnberger, A.; Köhler, J. M.: The Power of Ensembles for Active Learning in Image Classification. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (2018), 9368–9377.

[36] Ash, J. T.; Zhang, C.; Krishnamurthy, A.; Langford, J.; Agarwal, A.: *Deep Batch Active Learning by Diverse, Uncertain Gradient Lower Bounds*. In: *International Conference on Learning Representations*. 2019.

[37] Lewis, D. D.: A Sequential Algorithm for Training Text Classifiers: Corrigendum and Additional Data. In: *ACM SIGIR Forum* 29.2 (1995), 13–19.

[38] Gal, Y.; Islam, R.; Ghahramani, Z.: *Deep Bayesian Active Learning with Image Data*. In: *Proceedings of the 34th International Conference on Machine Learning*. International Conference on Machine Learning. PMLR, 2017, 1183–1192.

[39] Seung, H. S.; Opper, M.; Sompolinsky, H.: *Query by Committee*. In: *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*. Pittsburgh, PA: ACM, 1992, 287–294.

[40] Settles, B.; Craven, M.; Ray, S.: *Multiple-Instance Active Learning*. In: *Advances in Neural Information Processing Systems*. Vol. 20. Curran Associates, Inc., 2007.

[41] Sener, O.; Savarese, S.: *Active Learning for Convolutional Neural Networks: A Core-Set Approach*. In: *International Conference on Learning Representations*. Vancouver, Canada, 2018.

[42] Houlsby, N.; Huszár, F.; Ghahramani, Z.; Lengyel, M.: Bayesian Active Learning for Classification and Preference Learning. 2011. URL: http://arxiv.org/abs/1112.5745 (visited on 05/18/2023). preprint.

[43] Zhdanov, F.: Diverse Mini-Batch Active Learning. 2019. URL: http://arxiv.org/abs/1901.05954 (visited on 04/11/2023). preprint.

[44] Bishop, C. M.: *Pattern Recognition and Machine Learning*. Information Science and Statistics. New York: Springer, 2006. 738 pp.

[45] Thulasidasan, S.; Chennupati, G.; Bilmes, J. A.; Bhattacharya, T.; Michalak, S.: *On Mixup Training: Improved Calibration and Predictive Uncertainty for Deep Neural Networks*. In: *Advances in Neural Information Processing Systems*. Vol. 32. Curran Associates, Inc., 2019.

[46] Hüllermeier, E.; Waegeman, W.: Aleatoric and Epistemic Uncertainty in Machine Learning: An Introduction to Concepts and Methods. In: *Machine Learning* 110.3 (2021), 457–506.

[47] Valdenegro-Toro, M.; Mori, D. S.: *A Deeper Look into Aleatoric and Epistemic Uncertainty Disentanglement*. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 2022, 1508–1516.

[48] Lakshminarayanan, B.; Pritzel, A.; Blundell, C.: *Simple and Scalable Predictive Uncertainty Estimation Using Deep Ensembles*. In: *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc., 2017.

[49] Shannon, C. E.: A Mathematical Theory of Communication. In: *The Bell System Technical Journal* 27 (1948), 379–423.

[50] Agarwal, P. K.; Har-Peled, S.; Varadarajan, K. R.: Geometric Approximation via Coresets. In: *Combinatorial and Computational Geometry* MSRI Publications.52 (2005).

[51] Tsang, I. W.; Kwok, J. T.; Cheung, P.-M.; Hk, C. U.; Hk, C. U.; Hk, C. U.: Core Vector Machines: Fast SVM Training on Very Large Data Sets. In: *Journal of Machine Learning Research* 6.13 (2005), 29.

[52] Jamieson, K.; Jain, L.; Fernandez, C.; Glattard, N.; Nowak, R.: NEXT: A System for Real-World Development, Evaluation, and Application of Active Learning. In: *Advances in Neural Information Processing Systems* (2015), 2656–2664.

[53] Sievert, S.; Ross, D.; Jain, L.; Jamieson, K.; Nowak, R.; Mankoff, R.: *NEXT: A System to Easily Connect Crowdsourcing and Adaptive Data Collection*. In: *Proceedings of the 16th Python in Science Conference*. 2017, 113–119.

[54] Mead, A.: Review of the Development of Multidimensional Scaling Methods. In: *The Statistician* 41.1 (1992).

[55] Jung, S.: Lecture 8: Multidimensional Scaling. Advanced Applied Multivariate Analysis. Lecture (University of Pittsburgh). 2013.

[56] Hefner JR., R. A.: Extensions of the Law of Comparative Judgment to Discriminable and Multidimensional Stimuli. Ann Arbor, MI: University of Michigan, 1959.

[57] Zinnes, J. L.; MacKay, D. B.: Probabilistic Multidimensional Scaling: Complete and Incomplete Data. In: *Psychometrika* 48.1 (1983), 27–48.

[58] Mackay, D. B.: Multidimensional Scaling: Model for Distance Judgments. In: *Journal of Mathematical Psychology* 33 (1989), 187–205.

[59] Canal, G. H.; Massimino, A. K.; Davenport, M. A.; Rozell, C. J.: Active Embedding Search via Noisy Paired Comparisons. In: *36th International Conference on Machine Learning* (2019), 1493–1512.

[60] Goodfellow, I.; Bengio, Y.; Courville, A.: *Deep Learning*. MIT press.

[61] Cover, T. M.; Thomas, J. A.: Elements of Information Theory. In: *Internet Access* (2006), 774.

[62] Monarch, R.: *Human-in-the-Loop Machine Learning: Active Learning and Annotation for Human-Centered AI*. Simon and Schuster, 2021.

[63] Toral, A.: Reassessing Claims of Human Parity and Super-Human Performance in Machine Translation at WMT 2019. 2020. URL: http://arxiv.org/abs/2005.05738 (visited on 11/07/2022). preprint.

[64] Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K.: *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Minneapolis, MN, 2019.

[65] Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V.: RoBERTa: A Robustly Optimized BERT Pretraining Approach. 2019. URL: http://arxiv.org/abs/1907.11692 (visited on 07/08/2022). preprint.

[66] Wang, Z.; Cong, G.; Long, C.; Ju, C.: *Effective and Efficient Sports Play Retrieval with Deep Representation Learning*. In: *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2019, 499–509.

[67] Wang, Z.; Long, C.; Cong, G.: Similar Sports Play Retrieval With Deep Reinforcement Learning. In: *IEEE Transactions on Knowledge and Data Engineering* 35.4 (2023), 4253–4266.

[68] Mikolov, T.; Chen, K.; Corrado, G.; Dean, J.: Efficient Estimation of Word Representations in Vector Space. 2013. URL: http://arxiv.org/abs/1301.3781 (visited on 05/18/2023). preprint.

[69] Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; Dean, J.: *Distributed Representations of Words and Phrases and Their Compositionality*. In: *Advances in Neural Information Processing Systems*. Vol. 26. Curran Associates, Inc., 2013.

[70] Tao, Y.; Both, A.; Silveira, R. I.; Buchin, K.; Sijben, S.; Purves, R. S.; Laube, P.; Peng, D.; Toohey, K.; Duckham, M.: A Comparative Analysis of Trajectory Similarity Measures. In: *GIScience & Remote Sensing* 58.5 (2021), 643–669.

[71] Eiter, T.; Mannila, H.: *Computing Discrete Fréchet Distance*. Austria: Christian Doppler Laboratory for Expert Systems, TU Wien, 1994.

[72] Byoung-Kee Yi; Jagadish, H.; Faloutsos, C.: *Efficient Retrieval of Similar Time Sequences under Time Warping*. In: *Proceedings 14th International Conference on Data Engineering*. Orlando, FL, USA: IEEE Comput. Soc, 1998, 201–208.

[73] Vlachos, M.; Kollios, G.; Gunopulos, D.: *Discovering Similar Multidimensional Trajectories*. In: *Proceedings 18th International Conference on Data Engineering*. San Jose, CA, USA: IEEE Comput. Soc, 2002, 673–684.

[74] Chen, Y.; Yu, P.; Chen, W.; Zheng, Z.; Guo, M.: Embedding-Based Similarity Computation for Massive Vehicle Trajectory Data. In: *IEEE Internet of Things Journal* 9.6 (2022), 4650–4660.

[75] Frentzos, E.; Gratsias, K.; Theodoridis, Y.: *Index-Based Most Similar Trajectory Search*. In: *IEEE 23rd International Conference on Data Engineering*. Istanbul, Turkey: IEEE, 2007, 816–825.

[76] Arsomngern, P.; Long, C.; Suwajanakorn, S.; Nutanong, S.: *Self-Supervised Deep Metric Learning for Pointsets*. In: *IEEE 37th International Conference on Data Engineering*. Chania, Greece: IEEE, 2021, 2171–2176.

[77] Zaheer, M.; Kottur, S.; Ravanbakhsh, S.; Poczos, B.; Salakhutdinov, R. R.; Smola, A. J.: *Deep Sets*. In: *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc., 2017.

[78] Zhang, H.; Zhang, X.; Jiang, Q.; Zheng, B.; Sun, Z.; Sun, W.; Wang, C.: *Trajectory Similarity Learning with Auxiliary Supervision and Optimal Matching*. In: *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*. Yokohama, Japan, 2020, 3209–3215.

[79] Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; Polosukhin, I.: *Attention Is All You Need*. In: *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc., 2017.

[80] Schroff, F.; Kalenichenko, D.; Philbin, J.: *FaceNet: A Unified Embedding for Face Recognition and Clustering*. In: *IEEE Conference on Computer Vision and Pattern Recognition*. Boston, MA: IEEE, 2015, 815–823.

[81] Sinha, S.; Ebrahimi, S.; Darrell, T.: *Variational Adversarial Active Learning*. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. Long Beach, CA, 2019, 5972–5981.

[82] Mahapatra, D.; Bozorgtabar, B.; Thiran, J.-P.; Reyes, M.: Efficient Active Learning for Image Classification and Segmentation Using a Sample Selection and Conditional Generative Adversarial Network. In: *Medical Image Computing and Computer Assisted Intervention*. Vol. 11071. Cham: Springer International Publishing, 2018, 580–588.

[83] Shui, C.; Zhou, F.; Gagne, C.; Wang, B.: *Deep Active Learning: Unified and Principled Method for Query and Training*. In: *International Conference on Artificial Intelligence and Statistics*. Palermo, Italy, 2020.

[84] Kim, K.; Park, D.; Kim, K. I.; Chun, S. Y.: *Task-Aware Variational Adversarial Active Learning*. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Nashville, TN: IEEE, 2021, 8162–8171.

[85] Bachman, P.; Sordoni, A.; Trischler, A.: Learning Algorithms for Active Learning. In: *Proceedings of the 34th International Conference on Machine Learning* (2017).

[86] Konyushkova, K.; Sznitman, R.; Fua, P.: *Learning Active Learning from Data*. In: *Proceedings of the 31st Conference on Neural Information Processing Systems*. 2017.

[87] Fan, Y.; Tian, F.; Qin, T.; Li, X.-Y.; Liu, T.-Y.: *Learning to Teach*. In: *International Conference on Learning Representations*. 2018.

[88] Ravi, S.; Larochelle, H.: Meta-Learning for Batch Mode Active Learning. In: *International Conference on Learning Representations Workshop* (2018), 1–6.

[89] Geifman, Y.; El-Yaniv, R.: *Deep Active Learning with a Neural Architecture Search*. In: *Conference on Neural Information Processing Systems*. Vancouver, Canada, 2019, 1–11.

[90] Ross, S.; Gordon, G.; Bagnell, D.: *A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning*. In: *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. JMLR Workshop and Conference Proceedings, 2011, 627–635.

[91] Liu, M.; Buntine, W.; Haffari, G.: *Learning How to Actively Learn: A Deep Imitation Learning Approach*. In: *56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*. Vol. 1. Association for Computational Linguistics, 2018, 1874–1883.

[92] Baram, Y.; El-Yaniv, R.; Luz, K.: Online Choice of Active Learning Algorithms. In: *Journal of Machine Learning Research* 5 (2004), 255–291.

[93] Hsu, W. N.; Lin, H. T.: Active Learning by Learning. In: *Proceedings of the National Conference on Artificial Intelligence* 4 (2015), 2659–2665.

[94] Chu, H. M.; Lin, H. T.: Can Active Learning Experience Be Transferred? In: *Proceedings - IEEE International Conference on Data Mining* (2017), 841–846.

[95] Ravi, S.; Larochelle, H.: Optimization as a Model for Few-Shot Learning. In: *International Conference on Learning Representations* (2017).

[96] Chen, Y.; Hoffman, M. W.; Colmenarejo, S. G.; Denil, M.; Lillicrap, T. P.; Botvinick, M.; De Freitas, N.: *Learning to Learn without Gradient Descent by Gradient Descent*. In: *34th International Conference on Machine Learning*. Vol. 2. 2017, 1252–1260.

[97] Finn, C.; Abbeel, P.; Levine, S.: *Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks*. In: *Proceedings of the 34th International Conference on Machine Learning*. PMLR, 2017, 1126–1135.

[98] Nichol, A.; Achiam, J.; Schulman, J.: On First-Order Meta-Learning Algorithms. 2018. URL: http://arxiv.org/abs/1803.02999 (visited on 05/18/2023). preprint.

[99] Jamal, M. A.; Qi, G.-J.: *Task Agnostic Meta-Learning for Few-Shot Learning*. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, 11719–11727.

[100] Finn, C.; Xu, K.; Levine, S.: *Probabilistic Model-Agnostic Meta-Learning*. In: *Advances in Neural Information Processing Systems*. Vol. 31. Curran Associates, Inc., 2018.

[101] Mishra, N.; Rohaninejad, M.; Chen, X.; Abbeel, P.: *A Simple Neural Attentive Meta-Learner*. In: *International Conference on Learning Representations*. 2018, 1–17.

[102] Li, M.; Liu, X.; van de Weijer, J.; Raducanu, B.: *Learning to Rank for Active Learning: A Listwise Approach*. In: *25th International Conference on Pattern Recognition*. 2021, 5587–5594.

[103] Mirza, M.; Osindero, S.: Conditional Generative Adversarial Nets. 2014. URL: http://arxiv.org/abs/1411.1784 (visited on 07/22/2022). preprint.

[104] Kendall, A.; Gal, Y.: *What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?* In: *31st Conference on Neural Information Processing Systems*. Long Beach, CA, 2017.

[105] Kingma, D. P.; Welling, M.: An Introduction to Variational Autoencoders. In: *Foundations and Trends in Machine Learning* 12.4 (2019), 307–392.

[106] Mnih, A.; Rezende, D.: *Variational Inference for Monte Carlo Objectives*. In: *Proceedings of The 33rd International Conference on Machine Learning*. PMLR, 2016, 2188–2196.

[107] Snell, J.; Swersky, K.; Zemel, R.: *Prototypical Networks for Few-shot Learning*. In: *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc., 2017.

[108] Vinyals, O.; Blundell, C.; Lillicrap, T.; kavukcuoglu, k.; Wierstra, D.: *Matching Networks for One Shot Learning*. In: *Advances in Neural Information Processing Systems*. Vol. 29. Curran Associates, Inc., 2016.

[109] Kong, R.; Qiu, Z.; Liu, Y.; Zhao, Q.: *NimbleLearn: A Scalable and Fast Batch-mode Active Learning Approach*. In: *International Conference on Data Mining Workshops*. Auckland, New Zealand: IEEE, 2021, 350–359.

[110] Kim, G.; Yoo, C. D.: Blending Query Strategy of Active Learning for Imbalanced Data. In: *IEEE Access* 10 (2022), 79526–79542.

[111] Hochreiter, S.; Younger, A. S.; Conwell, P. R.: Learning To Learn Using Gradient Descent. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 2130 (2001).

[112] Jamieson, K. G.; Nowak, R. D.: *Low-Dimensional Embedding Using Adaptively Selected Ordinal Data*. In: *49th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. Monticello, IL: IEEE, 2011, 1077–1084.

[113] Heim, E.; Berger, M.; Seversky, L.; Hauskrecht, M.: Active Perceptual Similarity Modeling with Auxiliary Information. 2015. URL: http://arxiv.org/abs/1511.02254 (visited on 05/18/2023). preprint.

[114] Kumari, P.; Goru, R.; Chaudhuri, S.; Chaudhuri, S.: *Batch Decorrelation for Active Metric Learning*. In: *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*. Yokohama, Japan, 2020, 2255–2261.

[115] Musgrave, K.; Belongie, S.; Lim, S.-N.: *A Metric Learning Reality Check*. In: *Computer Vision – European Converence on Computer Vision*. Ed. by Vedaldi, A.; Bischof, H.; Brox, T.; Frahm, J.-M. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2020, 681–699.

[116] Hadsell, R.; Chopra, S.; LeCun, Y.: *Dimensionality Reduction by Learning an Invariant Mapping*. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Vol. 2. New York, NY: IEEE, 2006, 1735–1742.

[117] Yin, L.; Menkovski, V.; Pechenizkiy, M.: *Knowledge Elicitation Using Deep Metric Learning and Psychometric Testing*. In: *Machine Learning and Knowledge Discovery in Databases*. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2021, 154–169.

[118] Li, X.; Guo, Y.: Adaptive Active Learning for Image Classification. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2013), 859–866.

[119] Van Der Maaten, L.; Weinberger, K.: *Stochastic Triplet Embedding*. In: *IEEE International Workshop on Machine Learning for Signal Processing*. 2012.

[120] Gal, Y.: Uncertainty in Deep Learning. Cambridge, England: University of Cambridge, 2016.

[121] Eldar, Y.; Lindenbaum, M.; Porat, M.; Zeevi, Y.: The Farthest Point Strategy for Progressive Image Sampling. In: *IEEE Transactions on Image Processing* 6.9 (1997), 1305–1315.

[122] Gal, Y.; Ghahramani, Z.: *Dropout As a Bayesian Approximation: Representing Model Uncertainty in Deep Learning*. In: *Proceedings of The 33rd International Conference on Machine Learning*. Vol. 48. 2016, 1050–1059.

[123] Van Nguyen, T.; Nguyen, A. T.; Phan, H. D.; Nguyen, T. D.; Nguyen, T. N.: *Combining Word2Vec with Revised Vector Space Model for Better Code Retrieval*. In: *IEEE/ACM 39th International Conference on Software Engineering Companion*. Buenos Aires, Argentina: IEEE, 2017, 183–185.

[124] Karamshuk, D.; Noulas, A.; Scellato, S.; Nicosia, V.; Mascolo, C.: *Geo-Spotting: Mining Online Location-Based Services for Optimal Retail Store Placement*. In: *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Chicago, IL: ACM, 2013, 793–801.

[125] Kim, J.; Hwangbo, H.; Kim, S. J.; Kim, S.: Location-Based Tracking Data and Customer Movement Pattern Analysis Using for Sustainable Fashion Business. In: *Sustainability* 11.22 (2019), 6209.

[126] Belkin, N. J.; Croft, W. B.: Information Filtering and Information Retrieval: Two Sides of the Same Coin? In: *Communications of the ACM* (1992).

[127] McFee, B.; Lanckriet, G.: *Metric Learning to Rank*. In: *Proceedings of the 27th International Conference on Machine Learning*. Madison, WI: Omnipress, 2010, 775–782.

[128] Lim, D.; Lanckriet, G.: *Efficient Learning of Mahalanobis Metrics for Ranking*. In: *Proceedings of the 31st International Conference on Machine Learning*. International Conference on Machine Learning. PMLR, 2014, 1980–1988.

[129] Jose, C.; Fleuret, F.: Scalable Metric Learning via Weighted Approximate Rank Component Analysis. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 9909 LNCS (2016), 875–890.

[130] Zhao, T.; McAuley, J.; King, I.: *Improving Latent Factor Models via Personalized Feature Projection for One Class Recommendation*. In: *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. Melbourne, Australia: ACM, 2015, 821–830.

[131] Xu, A.; Davenport, M.: *Simultaneous Preference and Metric Learning from Paired Comparisons*. In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., 2020, 454–465.

[132] Kleiman, Y.; Goldberg, G.; Amsterdamer, Y.; Cohen-Or, D.: Toward Semantic Image Similarity from Crowdsourced Clustering. In: *The Visual Computer* 32.6-8 (2016), 1045–1055.

[133] Kaya, M.; Bilge, H. Ş.: Deep Metric Learning: A Survey. In: *Symmetry* 11.9 (2019).

[134] Zhang, Z.; Wu, Z.; Zhao, H.; Hu, M.: Knowledge Transfer Based Hierarchical Few-Shot Learning via Tree-Structured Knowledge Graph. In: *International Journal of Machine Learning and Cybernetics* 14.1 (2023), 281–294.

[135] Zhang, J.; Shao, S.; Verma, S.; Nowak, R.: Algorithm Selection for Deep Active Learning with Imbalanced Datasets. 2023. URL: http://arxiv.org/abs/2302.07317 (visited on 05/17/2023). preprint.

[136] Dörr, M.; Spoden, F.; Matthiesen, S.; Gwosch, T.: Activity Recognition With Machine Learning in Manual Grinding. In: *Journal of Manufacturing Science and Engineering* 144.094504 (2022).

[137] Scislo, L.: Single-Point and Surface Quality Assessment Algorithm in Continuous Production with the Use of 3D Laser Doppler Scanning Vibrometry System. In: *Sensors* 23.3 (3 2023).

[138] Cao, J.; Lam, K.-Y.; Lee, L.-H.; Liu, X.; Hui, P.; Su, X.: Mobile Augmented Reality: User Interfaces, Frameworks, and Intelligence. In: *ACM Computing Surveys* 55.9 (2023), 1–36.

[139] Tuyls, K. et al.: Game Plan: What AI Can Do for Football, and What Football Can Do for AI. In: *Journal of Artificial Intelligence Research* 71 (2021), 41–88.

[140] Wang, X.; Chen, H.; Gan, C.; Lin, H.; Dou, Q.; Tsougenis, E.; Huang, Q.; Cai, M.; Heng, P.-A.: Weakly Supervised Deep Learning for Whole Slide Lung Cancer Image Analysis. In: *IEEE Transactions on Cybernetics* 50.9 (2020), 3950–3962.

[141] Munirathinam, S.: Chapter Six - Industry 4.0: Industrial Internet of Things (IIOT). In: *Advances in Computers*. Ed. by Raj, P.; Evangeline, P. Vol. 117. 1 vols. The Digital Twin Paradigm for Smarter Systems and Environments: The Industry Use Cases. Elsevier, 2020, 129–164.

[142] Grønsund, T.; Aanestad, M.: Augmenting the Algorithm: Emerging Human-in-the-Loop Work Configurations. In: *The Journal of Strategic Information Systems*. Strategic Perspectives on Digital Work and Organizational Transformation 29.2 (2020), 101614.

Own publications referring to this work

[P1] Löffler, C.; Reeb, L.; Dzibela, D.; Marzilger, R.; Witt, N.; Eskofier, B. M.; Mutschler, C.: Deep Siamese Metric Learning: A Highly Scalable Approach to Searching Unordered Sets of Trajectories. In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 13.1 (2021), 1–23.

[P2] Löffler, C.; Mutschler, C.: IALE: Imitating Active Learner Ensembles. In: *Journal of Machine Learning Research (JMLR)* 23.107 (2022), 1–29.

[P3] Loeffler, C.; Fallah, K.; Fenu, S.; Zanca, D.; Eskofier, B.; Rozell, C. J.; Mutschler, C.: Active Learning of Ordinal Embeddings: A User Study on Football Data. In: *Transactions on Machine Learning Research (TMLR)* (2023), 1–26.

[P4] Redzepagic, A.; Loffler, C.; Feigl, T.; Mutschler, C.: A Sense of Quality for Augmented Reality Assisted Process Guidance. In: *Adjunct Proceedings of the 2020 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)* November (2020), 129–134.

[P5] Löffler, C.; Nickel, C.; Sobel, C.; Dzibela, D.; Braat, J.; Gruhler, B.; Woller, P.; Witt, N.; Mutschler, C.: *Automated Quality Assurance for Hand-Held Tools via Embedded Classification and AutoML*. In: *Applied Data Science and Demo Track: European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD)*. Vol. 2. 2020, 532–535.

[P6] Mishra, A.; Löffler, C.; Plinge, A.: *Recipes for Post-training Quantization of Deep Neural Networks*. In: *Workshop on Energy Efficient Machine Learning and Cognitive Computing (EMC^2)*. 2020.

[P7] Ott, F.; Feigl, T.; Loffler, C.; Mutschler, C.: *ViPR: Visual-odometry-aided Pose Regression for 6DoF Camera Localization*. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPR)*. Vol. 2020-June. 2020, 187–198.

[P8] Feigl, T.; Porada, A.; Steiner, S.; Löffler, C.; Mutschler, C.; Philippsen, M.: *Localization Limitations of ARCore, ARKit, and Hololens in Dynamic Large-scale Industry Environments.* In: *International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP)*. 2020, 307–318.

[P9] Löffler, C.; Lai, W.-C.; Eskofier, B.; Zanca, D.; Schmidt, L.; Mutschler, C.: Don't Get Me Wrong: How to apply Deep Visual Interpretations to Time Series. In: *arXiv preprint arXiv:2203.07861* (2022).