Neural Processing Letters (2022) 54:3749–3769
https://doi.org/10.1007/s11063-022-10784-y

Deep Distributional Sequence Embeddings Based on a Wasserstein Loss

Ahmed Abdelwahab[1,2] · **Niels Landwehr**[1,2,3]

Accepted: 21 February 2022 / Published online: 18 March 2022

Abstract
Deep metric learning employs deep neural networks to embed instances into a metric space such that distances between instances of the same class are small and distances between instances from different classes are large. In most existing deep metric learning techniques, the embedding of an instance is given by a feature vector produced by a deep neural network and Euclidean distance or cosine similarity defines distances between these vectors. This paper studies deep distributional embeddings of sequences, where the embedding of a sequence is given by the distribution of learned deep features across the sequence. The motivation for this is to better capture statistical information about the distribution of patterns within the sequence in the embedding. When embeddings are distributions rather than vectors, measuring distances between embeddings involves comparing their respective distributions. The paper therefore proposes a distance metric based on Wasserstein distances between the distributions and a corresponding loss function for metric learning, which leads to a novel end-to-end trainable embedding model. We empirically observe that distributional embeddings outperform standard vector embeddings and that training with the proposed Wasserstein metric outperforms training with other distance functions.

Keywords Metric learning · Sequence embeddings · Deep learning

✉ Ahmed Abdelwahab
ahm.abdelwahab@gmail.com

Niels Landwehr
Landwehr@uni-hildesheim.de

[1] Institute for Computer Science, University of Potsdam , August-Bebel-Str. 89, 14482 Potsdam, Germany

[2] Leibniz Institute for Agricultural Engineering and Bioeconomy (ATB), Max-Eyth-Allee 100, 14469 Potsdam, Germany

[3] Department of Computer Science Universitätspl. 1, University of Hildesheim, 31141 Hildesheim, Germany

1 Introduction

Metric learning is concerned with learning a representation or *embedding* in which distances between instances of the same class are small and distances between instances of different classes are large. Deep metric learning approaches, in which the learned embedding is given by a deep neural network, have achieved state-of-the-art results in many tasks, including face verification and recognition [24], fine-grained image classification [21], zero-shot classification [5], speech-to-text problems [10], and speaker identification [14]. An advantage of metric learning is that the resulting representation directly generalizes to unseen classes, so the model does not need to be retrained every time a new class is introduced. This is, for example, a typical requirement in biometric applications, where it should be possible to register new subjects without retraining a model. Biometric systems also have to handle imposters, that is, subjects who are not registered in the database, which is not straightforward in standard classification settings.

In this paper, we study deep metric learning for sequence data, with a specific focus on biometric problems. Building on earlier work on *quantile layers* [1], the paper specifically studies how the distribution of learned deep features across a sequence can be represented in the learned embedding. Quantile layers are statistical aggregation layers that characterize the distribution of patterns within a sequence by approximating the quantile function of the activations of the learned filters across the sequence. Characterizing this distribution has been shown to be advantageous for biometric identification based on eye movement patterns [1]. The main contribution of this paper is to develop a deep metric learning approach for distributional embeddings based on quantile layers. Quantile layers return an estimate of the distribution of values for each learned filter across the sequence. Instead of a fixed-length vector representation of an instance, in our approach, the embedding of an instance is given by these sets of distributions. When embeddings are distributions rather than simple vectors, measuring distances between the embeddings involves comparing their respective distributions. The paper proposes a distance metric in the embedding space that is based on Wasserstein distances between the respective distributions. Compared to other distance functions such as Kulback–Leibler or Jensen–Shannon divergence, the advantage of using Wasserstein distance is that it takes into account the metric on the space in which the random variable of interest is defined. In our case, this means that distributions in which similar magnitudes of filter activations receive similar amounts of probability mass will be considered close. The paper further shows how such embeddings can be trained end-to-end on labeled training data using metric learning techniques.

Empirically, the proposed approach is studied in biometric identification problems involving eye movement, accelerometer, and EEG data. Empirical results show that the proposed distributional sequence embeddings outperform standard vector embeddings and that training with the Wasserstein metric outperforms training with other distance functions.

The rest of the paper is organized as follows. Section 2 discusses related work. Section 3 reviews quantile layers and develop a distributional embedding architecture based on these layers. Section 4 introduces a Wasserstein-based distance metric for the proposed embedding model and from this derives a novel loss function for metric learning. Section 5 empirically studies the proposed method and baselines.

2 Related Work

The work in this paper is motivated by the goal of capturing information about the distribution of patterns within a sequence in its embedding, where the patterns are defined in terms of learned features of a deep neural network. It is related to other work in deep learning that aims to capture distributions of learned features using statistical aggregation layers. Wang et al. [31] proposed end-to-end learnable *histogram layers* that approximate the distribution of learned features by a histogram. Their work uses linear apprsedighi2017histogramoximations to smoothen the sharp edges in a traditional histogram function and enable gradient flow. Sedighi and Fridrich [25] proposed a similar histogram-based aggregation layer, but use Gaussian kernels as a soft, differentiable approximation to histogram bins. Abdelwahab and Landwehr [1] introduced *quantile layers* to capture the distribution of learned features based on an approximation of the quantile function, and empirically showed that this outperforms aggregation using histograms. The contribution of our paper is to exploit quantile layers in metric learning, by defining distributional embeddings based on approximations of quantile functions and deriving loss functions for metric learning based on comparing the resulting distributions.

There is a large body of work on deep metric learning that studies different network architectures and loss functions. For example, [11] introduced a loss for a siamese network architecture that is based on all possible pairs of instances in the training data, and its objective is to minimize distances between positive pairs (same class) while maximizing the distances between negative pairs (different classes). More recently, [24] introduced the triplet loss, with links positive and negative pairs by an anchor instance. This idea has later been extended by [20,27] by providing several negative pairs linked to one positive pair to the loss function. The loss function introduced by [27] has shown superior performance in several studies [27,32,35]. Our method builds on these established deep metric learning techniques, but extends them by replacing vector embeddings with distributional embeddings, which requires corresponding changes in distance calculations and the loss function.

Distributional embeddings have been recently studied in biometric face recognition by Shi and Jain [26]. In this work, an instance (face image) is mapped to a Gaussian distribution over possible feature vectors, represented by a mean vector and a diagonal covariance matrix, where mean and covariance vectors are generated from the input instance by a deep neural network. The similarity of two inputs in embedding space can then be computed from their two distribution. The motivation for these distributional embeddings is somewhat different from our motivation in this paper: while the distribution in our model results from the inner structure of the instance being mapped (distribution of patterns within a sequence), the distribution in the model by [26] captures remaining uncertainty and is inferred from pairs of instances during training. Their work also differs from ours in that they make strong parametric assumptions about the distribution (Gaussian) and use different loss functions and different distance function in the embedding space. Similar distributional embeddings based on Gaussian distributions have also been studied by Yu et al. [34] for person re-identification and by Wang et al. [30] for implicit semantic data augmentation. We provide an empirical comparison to the work of Shi and Jain [26], which unlike the other two approaches is also directly aimed at biometric settings, in Sect. 5.

Distributional embeddings have also been studied in natural language processing in the context of word embeddings. Traditional word embedding models such as *word2vec* represent words as vectors in a metric space such that semantically similar words are mapped to similar vectors [16]. Vilnis and McCallum [28] extend this idea by mapping each word to a Gaussian

distribution (with diagonal covariance), which naturally characterizes uncertainty about the embedding. Athiwaratkun and Wilson [3] further extend this model by replacing the Gaussian distribution with a mixture of Gaussians, where the multimodal mixture can capture multiple meanings of the same word. Again, the motivation for these distributional embeddings does not result from the inner structure of the instance being mapped as in our approach, but rather captures remaining uncertainty. Another difference in the work by [28] is that their model is trained in an unsupervised fashion, while we study supervised metric learning. An approach similar to that of [28] has also been taken by [4] in order to map nodes of an attributed graph onto Gaussian distributions that function as an embedding representation. This is again an unsupervised approach, and specific to the task of node embedding.

More generally, deep metric learning models have been recently used in different application domains featuring sequential data, including natural language processing [18,19], computer vision [15,33] and speaker identification [7,14], but these approaches are based on vector embeddings rather than distributional embeddings.

3 Quantile Layers and Distributional Sequence Embeddings

This section reviews *quantile layers* as introduced by [1] and discusses how they can be used to define distributional embeddings of variable-length sequences.

In this paper, we focus on variable-length sequences and deep convolutional neural network architectures that produce embeddings of such sequences. Typically, network architectures for such sequences would employ stacked convolution layers to extract informative features from the sequence, and in the last layer use some form of global pooling to transform the remaining variable-length representation into a fixed-length vector representation. Global pooling achieves this transformation by performing a simple aggregate operation such as taking the maximum or average over the filter activations across the sequence. This has the potential disadvantage that most information about the distribution of the filter activations is lost, which might be informative for the task at hand. In contrast, quantile layers try to preserve as much information as possible about the distribution of filter activations along the sequence by approximating the quantile function of this distribution. Earlier work has shown that this information can be informative for sequence classification, substantially increasing predictive accuracy [1].

This paper proposes to use quantile layers for defining distributional embeddings of sequences. It is assumed that instances are given by variable-length sequences of the form $\mathbf{s} = (\mathbf{x}_1, ..., \mathbf{x}_T)$ where $\mathbf{x}_t \in \mathbb{R}^D$ is a vector of attributes that describes the sequence element at position t. The space of all such sequences with D attributes will be denoted by by $\mathcal{S}_D = \bigcup_{T=1}^{\infty} \mathbb{R}^{T \times D}$. When a sequence is processed by a convolutional deep neural network architecture Γ, which is taken to be the network without any final global aggregation layers, the result is a variable-length representation of the instance over K filters. This mapping will be denoted by $\Gamma : \mathcal{S}_D \rightarrow \mathcal{S}_K$. Details of the deep convolutional architectures employed are given in Sect. 5. For $\mathbf{s} \in \mathcal{S}_D$ and $k \in \{1, ..., K\}$, $\Gamma_k(\mathbf{s})$ is used to denote the variable-length sequence of activations of filter k produced by the network for sequence $\mathbf{s}$.

As in [1], this paper uses quantile functions in order to characterize the distribution of filter activations across the sequence $\Gamma_k(\mathbf{s})$. Let $x \in \mathbb{R}$ be a real-valued random variable, let $p(x)$ denote its density and $F(x)$ its cumulative distribution function. The quantile function for x is defined by

$$Q(r) = \inf\{x \in \mathbb{R} : F(x) \geq r\}$$

where inf denotes the infimum. If F is continuous and strictly monotonically increasing, Q is simply the inverse of F. Let $\mathcal{X} = \{x_1, ..., x_N\}$ be a sample of the random variable x, that is, $x_n \sim p(x)$ for $n \in \{1, ..., N\}$. The empirical quantile function $\hat{Q}_{\mathcal{X}} : (0, 1] \to \mathbb{R}$ is a non-parametric estimator of the quantile function Q. It is defined by

$$\hat{Q}_{\mathcal{X}}(r) = \inf\{x \in \mathbb{R} : r \leq \hat{F}_{\mathcal{X}}(x)\} \tag{1}$$

where $\hat{F}_{\mathcal{X}}(x) = \frac{1}{N}\sum_{i=1}^{N} I(x_i \leq x)$ is the empirical cumulative distribution function and $I(x_i \leq x) \in \{0, 1\}$ is an indicator. $\hat{Q}_{\mathcal{X}}(r)$ is a piecewise constant function that is essentially obtained by sorting the samples in $\mathcal{X}$. More formally, let π be a permutation that sorts the x_i, that is, $x_{\pi(i)} \leq x_{\pi(i+1)}$ for $1 \leq i \leq N-1$. Then $\hat{Q}_{\mathcal{X}}(r) = x_{\pi(\lceil rN \rceil)}$, where $\lceil x \rceil$ denotes the smallest integer larger or equal to x. The empirical quantile function $\hat{Q}_{\mathcal{X}}$ faithfully approximates the quantile function Q in the sense that $|\hat{Q}_{\mathcal{X}}(r) - Q(r)|$ converges almost surely to zero if $N \to \infty$ and Q is continuous at r [22].

To enable gradient flow in end-to-end learning, we will work with a piecewise linear interpolation of the piecewise constant function $\hat{Q}_{\mathcal{X}}(r)$. For $i \in \{1, ..., N\}$ and $r \in [\frac{n-1}{N}, \frac{n}{N}]$ let

$$\tilde{Q}_{\mathcal{X}}(r) = N(x_{\pi(n+1)} - x_{\pi(n)})r + nx_{\pi(n)} + (1-n)x_{\pi(n+1)} \quad \left(r \in \left[\frac{n-1}{N}, \frac{n}{N}\right]\right)$$

define a linear approximation, where $x_{\pi(N+1)} = x_{\pi(N)}$ is defined to handle the right interval border. Combining the linear approximations over the different n, for $r \in [0, 1]$ the following piecewise linear approximation is obtained:

$$\tilde{Q}_{\mathcal{X}}(r) = \sum_{n=1}^{N} \tilde{\delta}(r, n)\left(N(x_{\pi(n+1)} - x_{\pi(n)})r + nx_{\pi(n)} + (1-n)x_{\pi(n+1)}\right)$$

where $\tilde{\delta}(r, n)$ is an indicator function that is defined as one if $r \in [\frac{n-1}{N}, \frac{n}{N}]$ and zero otherwise. The piecewise linear approximation $\tilde{Q}_{\mathcal{X}}(r)$ of the quantile function depends on the sample size N, because there are N linear segments. To arrive at an approximation of the quantile function that is independent of the number of samples, we define a further piecewise linear approximation of $\tilde{Q}_{\mathcal{X}}(r)$ using M sampling points $\sigma(\alpha_1)$, ..., $\sigma(\alpha_M)$, where $\sigma(\alpha) = (1 + exp(-\alpha))^{-1}$ is the sigmoid function and $\alpha_i \in \mathbb{R}$ are parameters with $\alpha_i \leq \alpha_{i+1}$. Formally, let

$$\bar{Q}_{\mathcal{X}}(r) = \sum_{i=0}^{M} \bar{\delta}(r, i)(a_{\mathcal{X},i}r + b_{\mathcal{X},i}) \tag{2}$$

where

$$a_{\mathcal{X},i} = \frac{\tilde{Q}_{\mathcal{X}}(\sigma(\alpha_{i+1})) - \tilde{Q}_{\mathcal{X}}(\sigma(\alpha_i))}{\sigma(\alpha_{i+1}) - \sigma(\alpha_i)} \tag{3}$$

$$b_{\mathcal{X},i} = \tilde{Q}_{\mathcal{X}}(\sigma(\alpha_i)) - \sigma(\alpha_i)\frac{\tilde{Q}_{\mathcal{X}}(\sigma(\alpha_{i+1})) - \tilde{Q}_{\mathcal{X}}(\sigma(\alpha_i))}{\sigma(\alpha_{i+1}) - \sigma(\alpha_i)}, \tag{4}$$

$\bar{\delta}(r, i)$ is an indicator function that is one if $r \in [\sigma(\alpha_i), \sigma(\alpha_{i+1})]$ and zero otherwise, and we have introduced $\alpha_0 = -\infty$ and $\alpha_{M+1} = \infty$ to handle border cases. The function $\bar{Q}_{\mathcal{X}}(r)$ provides a piecewise linear approximation of the quantile function using $M+1$ line segments, independently of the sample size N. The parameters α_i are learnable model parameters in the deep neural network architectures that we study in Sect. 5.

We are now ready to define the distributional embedding for an instance, which is obtained by passing the instance through the neural network Γ and for each filter in the output of Γ approximating the quantile function of the filter activations by the piecewise linear function $\bar{Q}$.

Definition 1 (*Distributional embedding of sequence*) Let $\mathbf{s} \in \mathcal{S}_D$ and let Γ denote a convolutional neural network structure. The distributional embedding of sequence $\mathbf{s}$ is given by the vector of piecewise linear functions

$$\Psi_\Gamma(\mathbf{s}) = \left(\bar{Q}_{\Gamma_1(\mathbf{s})}, ..., \bar{Q}_{\Gamma_K(\mathbf{s})}\right) \tag{5}$$

where $\bar{Q}_{\Gamma_k(\mathbf{s})}$ is defined by Eq. 2 using $\mathcal{X} = \Gamma_k(\mathbf{s})$. Here, the notation is slightly generalized by identifying the sequence of observations $\Gamma_k(\mathbf{s})$ with the corresponding set of observations.

It should be noted that due to the piecewise linear approximations, gradients can flow through the entire embedding architecture, both to parameters α_m and the weights in the deep neural network structure Γ. This includes the sorting operation, where gradients can be passed through by reordering the gradient backpropagated from the layer above according to the sorting indices π.

4 A Wasserstein Loss for Distributional Embeddings

For training the embedding model, a deep metric learning approach will be used which trains model parameters such that instances of the same class are close and instances of different classes are far apart in the embedding space. In order to apply such approaches, a distance metric needs to be defined on the embedding space.

4.1 Distances Between Distributional Embeddings

As discussed in Sect. 3, in the setting discussed in this paper embeddings of instances are given by distributions. Measuring the distance between two embeddings thus means comparing their respective distributions. Different approaches to measure distances between probability distributions have been discussed in the literature. One of the most widely used distance functions between distributions is the Kullback–Leibler divergence. However, this measure is asymmetric and can result in infinite distances, and is therefore not a metric. A metric based on the Kullback–Leibler divergence is the square root of the Jensen–Shannon divergence, which is symmetric, bounded between zero and $\sqrt{log(2)}$, and satisfies the triangle inequality. However, this metric does not yield useful gradients in case the distributions being compared have disjoint support, which in our case would occur if two sequences with non-overlapping ranges of filter values are compared. To illustrate, let q_1 and q_2 denote densities with disjoint support A_1 and A_2, and let $m(x) = \frac{q_1(x)+q_2(x)}{2}$. Then the Jensen–Shannon divergence J of q_1 and q_2 is

$$\begin{aligned} J(q_1, q_2) &= \tfrac{1}{2} \int_{A_1 \cup A_2} q_1(x) log\left(\frac{q_1(x)}{m(x)}\right) dx + \frac{1}{2} \int_{A_1 \cup A_2} q_2(x) log\left(\frac{q_2(x)}{m(x)}\right) dx \\ &= \tfrac{1}{2} \int_{A_1} q_1(x) log\left(2\frac{q_1(x)}{q_1(x)}\right) dx + \frac{1}{2} \int_{A_2} q_2(x) log\left(2\frac{q_2(x)}{q_2(x)}\right) dx \\ &= log(2) \end{aligned}$$

independently of the distance between A_1 and A_2, resulting in a gradient of zero.

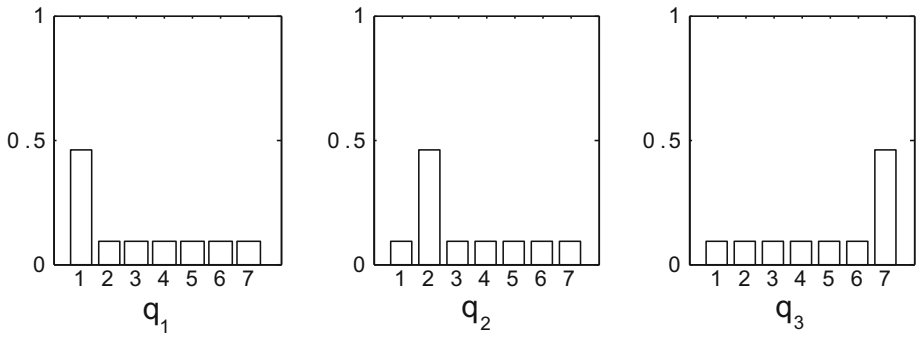

Fig. 1 According to the Wasserstein metric, distributions q_1 and q_2 are closer than q_1 and q_3, while distances would be identical under the Jensen–Shannon measure

A different class of distance functions which are increasingly being studied in machine learning [2,8,9] are Wasserstein distances. Wasserstein distances are based on the idea of optimal transport plans. They do not suffer from the zero-gradient problem exhibited by the Jensen–Shannon divergence, because they take into account the metric of the underlying space. They also guarantee continuity under mild assumptions, which is not the case for the Jensen–Shannon divergence as illustrated by [2]. In the general case, the p-Wasserstein distance (for $p \in \mathbb{N}$) between two probability measures ρ_1 and ρ_2 over a space $\mathcal{M}$ with metric d can be defined as

$$W_p(\rho_1, \rho_2) = \left(\inf_{\pi \in \mathcal{J}(\rho_1, \rho_2)} \int_{\mathcal{M} \times \mathcal{M}} d(x, y)^p d\pi(x, y) \right)^{\frac{1}{p}} \tag{6}$$

where $\mathcal{J}(\rho_1, \rho_2)$ denotes the set of all joint measures on $\mathcal{M} \times \mathcal{M}$ with marginals ρ_1 and ρ_2. For the purpose of this paper, the random variables are assumed to be real-valued. If $q_1(x_1)$ and $q_2(x_2)$ are two densities defining distributions over real-valued random variables, $x_i \in \mathbb{R}$, the p-Wasserstein distance between q_1 and q_2 is given by

$$W_p(q_1, q_2) = \left(\inf_{q \in \mathcal{J}(q_1, q_2)} \iint |x_1 - x_2|^p q(x_1, x_2) dx_1 dx_2 \right)^{\frac{1}{p}} \tag{7}$$

where $\mathcal{J}(q_1, q_2)$ defines the set of all joint distributions over x_1, x_2 which have marginals q_1 and q_2. A joint distribution $q \in \mathcal{J}(q_1, q_2)$ can be seen as a *transport plan*, that is, a way of moving probability mass from density q_1 such that the resulting density is q_2, in the sense that $q(x_1, x_2)$ indicates how much mass is moved from $q_1(x_1)$ to $q_2(x_2)$. The quantity $\iint |x_1 - x_2|^p q(x_1, x_2) dx_1 dx_2$ is the cost of the transport plan, which depends on the amount of probability mass moved, $q(x_1, x_2)$, and the distance by which the mass has been moved, $|x_1 - x_2|^p$. The infimum over the set $\mathcal{J}(q_1, q_2)$ means that the distance between the distributions is given by the optimal transport plan, which intuitively characterizes the minimum changes that need to be made to q_1 in order to transform it into q_2. For $p = 1$ the distance is therefore also called the *Earth Mover Distance*. The advantage of this measure is that it takes into account the metric in the underlying space, as can be seen from Fig. 1. Here, q_1 is closer to q_2 than it is to q_3 in the sense that the probability mass needs to be moved less far. Thus, $W_p(q_1, q_2) < W_p(q_1, q_3)$, while the Jensen–Shannon distances between the two pairs of distributions would be identical.

Because Wasserstein distances are defined in terms of optimal transport plans, computing them in general requires solving non-trivial optimization problems. However, for the case of real-valued random variables $x_i \in \mathbb{R}$, there is a simple closed-form solution to the infimum in Eq. 7. Let $x_1 \sim q_1$, $x_2 \sim q_2$ with $x_i \in \mathbb{R}$. According to [6], the function $K(x_1, x_2) = |x_1 - x_2|^p$ for $p \geq 1$ is quasi-antitone and therefore the infimum of the expectation of this function over the set of all joint distributions, $\inf_{q \in \mathcal{J}(q_1,q_2)} E[K(x_1, x_2)]$, is given by $\int_0^1 K(Q_1(r), Q_2(r))dr$, where $Q_i(r) = \inf\{t : q_i(x_i \leq t) \geq r\}$ is the quantile function to the density q_i. Equation 7 can thus be rewritten as

$$W_p(q_1, q_2) = \left(\int_0^1 |Q_1(r) - Q_2(r)|^p dr \right)^{\frac{1}{p}}. \tag{8}$$

The distance between two embeddings $\Psi_\Gamma(\mathbf{s})$ and $\Psi_\Gamma(\mathbf{s}')$ can now be defined as the Wasserstein distance between the approximate representation of the quantile functions in the embedding as defined by Definition 1, summed over the different filters k.

Definition 2 Let $\mathbf{s}, \mathbf{s}' \in \mathcal{S}_D$, let Γ denote a convolutional neural network architecture, and let $\Psi_\Gamma(\mathbf{s})$ and $\Psi_\Gamma(\mathbf{s}')$ denote the distributional embeddings of $\mathbf{s}, \mathbf{s}'$ as defined by Definition 1. Then the distance between the embeddings can be defined as

$$d_p(\Psi_\Gamma(\mathbf{s}), \Psi_\Gamma(\mathbf{s}')) = \sum_{k=1}^{K} \left(\int_0^1 |\bar{Q}_{\Gamma_k(\mathbf{s})}(r) - \bar{Q}_{\Gamma_k(\mathbf{s}')}(r)|^p dr \right)^{\frac{1}{p}} \tag{9}$$

The next proposition gives a closed-form result for computing $d_p(\Psi_\Gamma(\mathbf{s}), \Psi_\Gamma(\mathbf{s}'))$.

Proposition 1 *Let $\mathbf{s}, \mathbf{s}' \in \mathcal{S}_D$, let Γ denote a convolutional neural network architecture, let $\Psi_\Gamma(\mathbf{s})$ and $\Psi_\Gamma(\mathbf{s}')$ denote the distributional embeddings of $\mathbf{s}, \mathbf{s}'$, and let $d_p(\Psi_\Gamma(\mathbf{s}), \Psi_\Gamma(\mathbf{s}'))$ denote their distance as defined by Definition 2. Then*

$$\begin{aligned} d_p(\Psi_\Gamma(\mathbf{s}), \Psi_\Gamma(\mathbf{s}')) = \sum_{k=1}^{K} \Bigg(\sum_{i=0}^{M} & \frac{(\bar{a}_{i,k}\sigma(\alpha_{i+1}) + \bar{b}_{i,k})|\bar{b}_{i,k}\sigma(\alpha_{i+1}) + \bar{b}_{i,k}|^p}{\bar{a}_{i,k}(p+1)} \\ & - \frac{(\bar{a}_{i,k}\sigma(\alpha_i) + \bar{b}_{i,k})|\bar{a}_{i,k}\sigma(\alpha_i) + \bar{b}_{i,k}|^p}{\bar{a}_{i,k}(p+1)} \Bigg)^{\frac{1}{p}} \end{aligned} \tag{10}$$

with

$$\begin{aligned} \bar{a}_{i,k} &= a_{\Gamma_k(\mathbf{s}),i} - a_{\Gamma_k(\mathbf{s}'),i} \\ \bar{b}_{i,k} &= b_{\Gamma_k(\mathbf{s}),i} - b_{\Gamma_k(\mathbf{s}'),i} \end{aligned}$$

where $a_{\mathcal{X},i}$ and $b_{\mathcal{X},i}$ for $\mathcal{X} \in \{\Gamma_k(\mathbf{s}), \Gamma_k(\mathbf{s}')\}$ are defined by Eqs. 3 and 4, σ is the sigmoid function, and as above we have introduced $\alpha_0 = -\infty$ and $\alpha_{M+1} = \infty$ to handle border cases.

Proof (Proposition 1) Starting from Definition 2 and plugging in $\bar{Q}_{\Gamma_k(\mathbf{s})}$ as defined by Eq. 2, it can be seen that

$$\begin{aligned} & \int_0^1 |\bar{Q}_{\Gamma_k(\mathbf{s})}(r) - \bar{Q}_{\Gamma_k(\mathbf{s}')}(r)|^p dr \\ & = \int_0^1 |\sum_{i=0}^{M} \bar{\delta}(r, i) \left((a_{\Gamma_k(\mathbf{s}),i} - a_{\Gamma_k(\mathbf{s}'),i})r + b_{\Gamma_k(\mathbf{s}),i} - b_{\Gamma_k(\mathbf{s}'),i} \right) |^p dr \end{aligned}$$

$$= \sum_{i=0}^{M} \int_{\sigma(\alpha_i)}^{\sigma(\alpha_{i+1})} |\bar{a}_{i,k} r + \bar{b}_{i,k}|^p dr \tag{11}$$

$$= \sum_{i=0}^{M} \frac{(\bar{a}_{i,k} r + \bar{b}_{i,k})|\bar{a}_{i,k} r + \bar{b}_{i,k}|^p}{\bar{a}_{i,k}(p+1)} \Bigg|_{\sigma(\alpha_i)}^{\sigma(\alpha_{i+1})} \tag{12}$$

□

where in Eq. 12 the notation $G(r)|_a^b = G(b) - G(a)$ is used. In Eq. 11, the integral is over subintervals $[\sigma(\alpha_i), \sigma(\alpha_{i+1})]$ of the interval $[0, 1]$, and therefore the indicator function $\bar{\delta}(r, i)$ can be removed. In Eq. 12 the integral is solved, exploiting that according to product and chain rules

$$\begin{aligned}
&\frac{\partial}{\partial r} \frac{(\bar{a}_{i,k} r + \bar{b}_{i,k})|\bar{a}_{i,k} r + \bar{b}_{i,k}|^p}{\bar{a}_{i,k}(p+1)} \\
&= \frac{\bar{a}_{i,k}|\bar{a}_{i,k} r + \bar{b}_{i,k}|^p + (\bar{a}_{i,k} r + \bar{b}_{i,k}) p |\bar{a}_{i,k} r + \bar{b}_{i,k}|^{p-1} \text{sign}(\bar{a}_{i,k} r + \bar{b}_{i,k}) \bar{a}_{i,k}}{\bar{a}_{i,k}(p+1)} \\
&= |\bar{a}_{i,k} r + \bar{b}_{i,k}|^p.
\end{aligned}$$

The claim directly follows from Eq. 12. □

An important note with respect to the distance function $d_p(\Psi_\Gamma(\mathbf{s}), \Psi_\Gamma(\mathbf{s}'))$ is that its closed-form computation given by Proposition 1 allows gradients to be propagated through distance computations (as well as through embedding computations as discussed in Sect. 3) to the parameters of the model Γ defining the embedding. Moreover, all computations can be expressed using standard building blocks available in common deep learning frameworks, such that all gradients are available through automatic differentiation.

4.2 Loss Function

Deep metric learning trains models with loss functions that drive the model towards minimizing distances between pairs of instances from the same class (positive pairs) while maximizing distances between pairs of instances from different classes (negative pairs). Existing approaches differ in the way negative and positive pairs are selected and the exact formulation of the loss. For example, triplet-based losses as introduced by [24] compare the distance between an anchor instance and another instance from the same class (positive pair) to the distance between the anchor instance and an instance from a different class (negative pair). However, comparing a positive pair with only a single negative pair does not take into account the distance to other classes and can thereby lead to suboptimal gradients; more recent approaches therefore often consider several negative pairs for each positive pair [20,27]. Inspired by these approaches, several negative pairs are considered for each positive pair, leading to a loss function of the form

$$\mathcal{L} = \sum_{(\mathbf{s_1}, \mathbf{s_2}) \in \mathcal{P}} \sum_{\substack{(\mathbf{s}_3, \mathbf{s}_4) \in \mathcal{N} \\ \mathbf{s}_3 \in \{\mathbf{s}_1, \mathbf{s}_2\}}} \ell(\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3, \mathbf{s}_4)$$

where $\mathcal{P} \subset \mathcal{S}_D \times \mathcal{S}_D$ is a set of positive pairs and $\mathcal{N} \subset \mathcal{S}_D \times \mathcal{S}_D$ is a set of negative pairs of instances, and $\ell(\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3, \mathbf{s}_4)$ is a loss function that penalizes cases in which a negative pair $(\mathbf{s}_3, \mathbf{s}_4)$ has smaller distance than a positive pair $(\mathbf{s}_1, \mathbf{s}_2)$. A straightforward linear formulation of the loss would be $\ell(\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3, \mathbf{s}_4) = d_p(\Psi_\Gamma(\mathbf{s}_1), \Psi_\Gamma(\mathbf{s}_2)) - d_p(\Psi_\Gamma(\mathbf{s}_3), \Psi_\Gamma(\mathbf{s}_4))$. However,

only pairs of pairs that violate the distance criterion should contribute to the loss, leading to $\ell(\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3, \mathbf{s}_4) = \max(0, d_p(\Psi_\Gamma(\mathbf{s}_1), \Psi_\Gamma(\mathbf{s}_2)) - d_p(\Psi_\Gamma(\mathbf{s}_3), \Psi_\Gamma(\mathbf{s}_4)))$. This loss is further replaced by a smooth upper bound using log-sum-exp, leading to the final Wasserstein-based loss function

$$\mathcal{L} = \sum_{(\mathbf{s_1},\mathbf{s_2})\in\mathcal{P}} \sum_{\substack{(\mathbf{s}_3,\mathbf{s}_4)\in\mathcal{N} \\ \mathbf{s}_3\in\{\mathbf{s}_1,\mathbf{s}_2\}}} log\left(1 + exp^{d_p(\Psi_\Gamma(\mathbf{s}_1),\Psi_\Gamma(\mathbf{s}_2)) - d_p(\Psi_\Gamma(\mathbf{s}_3),\Psi_\Gamma(\mathbf{s}_4))}\right). \tag{13}$$

Equation 13 is of similar structure as other losses used in the literature, including the angular triplet loss [29], the lifted structured loss [20], and the N-pair loss [27].

It remains to specify how positive pairs $\mathcal{P}$ and negative pairs $\mathcal{N}$ are sampled for each stochastic gradient descent step. In this paper, we use the approach of [27] for generating $\mathcal{P}$ and $\mathcal{N}$, which has been shown to give state-of-the-art performance in several studies [27,32,35], in particular outperforming triplet-based sampling [24] and lifted structure sampling [20]. The approach constructs a batch of size $2N$ (where N is an adjustable parameter) by sampling from the training data N pairs of instances $\mathcal{P} = \{(\mathbf{s}_i, \mathbf{s}_i^+)\}_{i=1}^N$ from N different classes, such that each pair $(\mathbf{s}_i, \mathbf{s}_i^+)$ is a positive pair from a different class.[1] From the sampled batch, a set of $N(N-1)$ negative pairs is constructed by setting $\mathcal{N} = \{(\mathbf{s}_i, \mathbf{s}_j^+)\}_{\substack{i,j=1 \\ j\neq i}}^N$. Note that Eq. 13 can be computed by first computing the embeddings of the $2N$ instances in the batch, and then computing the overall loss. Thus, although the computation is quadratic in N, the number of evaluations of the deep neural network model Γ is linear in the batch size.

5 Empirical Study

The proposed method is empirically studied in three biometric identification domains involving human eye movements, accelerometer-based observation of human gait, and EEG recordings. As an ablation study, this section specifically evaluates which impact the different components of the proposed method—the metric learning approach, the use of quantile layers to fit the distribution of activations of filters across a sequence, and the Wasserstein-based distance function—have on overall performance.

The methods are also compared against state of the art methods in the domain of biometric identification from eye movements [1,23].

5.1 Data Sets

The empirical evaluation studies biometric identification based on eye movements, the gait, or the EEG signal of a subject. In all domains, the data consist of sequential observations of the corresponding low-level sensor signal—gaze position from an eye tracker, accelerometer measurements, or EEG measurements—for different subjects. The task is to identify the subject based on the observed sensor measurements.

The *Dynamic Images and Eye Movements* (DIEM) dataset [17] contains eye movement data of 210 subjects each viewing a subset of 84 video clips. The video clips are of varying length with an average of 95 seconds and contain different visual content, such as excerpts from sport matches, documentary videos, movie trailers, or recordings of street scenes. The data contain the gaze position on the screen for the left and the right eye, as well as a

[1] Source code can be found at https://github.com/abdelwahab/QuantileAggregation.

measurement of the pupil dilation, at a temporal resolution of 30 Hz. The eye movement data of a particular individual on a particular video clip is thus given by a sequence of six-dimensional vectors (horizontal and vertical gaze coordinate for left and right eye plus left and right pupil dilation), that is, $D = 6$ in the notation of Sect. 3. The average sequence length is 2850 and there are 5381 sequences overall.

The gait data comes from a study by [12] who collected the daily movement activity of 71 subjects for a period of 3 consecutive days. The recorded data consists of time series of 3D accelerometer measurements recorded at a sampling rate of 100 Hz. For each point in time, the measurement is a $D = 6$ dimensional vector consisting of the acceleration and velocity in x, y, and z direction. In the original data set, a continuous measurement for 3 days has been carried out for each individual. These long measurements contain different activities, but also long idle periods (for example, during sleep). We concentrate on subsequences showing high activity, by dividing the entire recording for each subject into intervals of length one minute, and then selecting for each subject the 30 subsequences that had the largest standard deviation in the 6-dimensional observations. This resulted in 2130 sequences overall (30 for each of the 71 subjects), with a length of $T = 6000$ per sequence.

The EEG data comes from a study by [36] who conducted EEG recording sessions with 121 subjects, measuring the signal from 64 electrodes placed on the scalp at a temporal resolution of 256 Hz of the subjects while viewing an image stimulus. The original aim of the study was to find a correlation between EEG observations and genetic predisposition to alcoholism, but as subject identifiers are available for all recordings the data can also be used in a biometric setting. Each subject completed between 40 and 120 trials with 1 second of recorded data per trial. The resulting data therefore consist of sequences of $D = 64$ dimensional vectors with a sequence length of 256 (one trial for one subject).

5.2 Problem Setting

As usual in metric learning, a setting is studied in which there are distinct sets of subjects at training and test time. The embedding model is first trained on a set of training subjects. On a separate and disjoint set of test subjects, we then evaluate to what degree the learned embedding assigns small distances to pairs of test sequences from the same subject, and large distances to pairs of sequences from different subjects. This reflects an application setting in which new subjects are registered in a database without retraining the embedding model. It also naturally allows the identification of imposters, that is, subjects who have never been observed (neither during training nor in the database of registered subjects) and try to gain access to the system.

In all three domains, we therefore first split the data into training and test data, such that there is no overlap in subjects between the two. For training the embedding model, we use data of 105 of the 210 subjects (eye movements), 36 of 71 subjects (gait data), or 61 of 121 subjects (EEG data). For the eye movement domain, we additionally ensure that there is no overlap in visual stimulus (video clips) between training and test data by splitting the set of all videos into training and test videos and only keeping the respective sequences in the training and test data. During training, each sequence constitutes an instance and the subject its class, and we train either embedding models using metric learning as discussed in Sect. 4 or, as a baseline, multiclass classification models (see Sect. 5.3 for details). We also set apart the data of 20% of the training individuals as validation data to tune model hyperparameters.

At test time, a biometric application setting is simulated by first sampling, for each test subject, a random subset of the sequences available for that subject as instances that are put

in an enrollment database. We then simulate that we observe additional sequences from a subject which are compared to the sequences of all subjects in the enrollment database. An embedding is good if the distance between these additional sequences and the enrollment sequences of the same subject is low, compared to the distance to the enrollment sequences of other subjects. More precisely, for each subject we use all except five of the sequences available for that subject as enrollment sequences. We then study how well the subject can be identified based on observing n of the remaining sequences, for $n \in \{1, .., 5\}$. Given observed sequences $\mathbf{s}_1, ..., \mathbf{s}_n$ (representing a subject that is unknown at test time), we compute distances to all subjects j as $d_j = \frac{1}{n} \sum_{i=1}^{n} d(\mathbf{s}_i, \mathbf{s}_{ij})$ where $\mathbf{s}_{ij}$ is the sequence of subject j in the enrollment database with minimal distance to $\mathbf{s}_i$. Here, the definition of the distance function d is method-specific (see below for details).

First a *verification* scenario is studied. This is the binary problem of deciding if the observed sequences $\mathbf{s}_1, ..., \mathbf{s}_n$ match a particular subject j, by comparing the computed distance d_j to a threshold value. Varying the threshold trades of false-positive versus false-negative classifications, yielding a ROC curve and AUC score. The ROC curve plots the true positive rate (fraction of cases in which a matching sequence was recognized as such) as a function of the false-positive rate (fraction of cases in which the sequence did not match the individual but was classified as matching) when varying the threshold value. AUC denotes the area under this curve. Note that an AUC of one would correspond to perfect predictions (all matching sequences classified as such and no false-positives), while an AUC of 0.5 corresponds to random prediction performance. Note that this verification scenario also covers the setting in which in imposter is trying to get access to a system as a particular user; the false-positive rate is the rate at which such imposters would be accepted.

Then a *multiclass identification* scenario is studied, where the model is used to assign the observed sequences $\mathbf{s}_1, ..., \mathbf{s}_n$ to a subject enrolled in the database (the subject $j^* = \arg\min_j d_j$). This constitutes a multiclass classification problem for which (multiclass) accuracy is measured. In this experiment, number of subjects under study is also varied, by randomly sampling a subset of subjects which are enrolled in the database; the same subset of subjects is observed at test time. The identification problem becomes more difficult as the number of subjects increases.

Finally, the robustness of the model to imposters in the multiclass identification scenario is studied, an experiment we denote as *multiclass imposters*. This reflects applications in which access to a system does not require a user name, because the system tries to automatically identify who is trying to gain access. In this experiment, half of the test subjects play the role of imposters who are not registered in the enrollment database. As in the multiclass identification setting, observed sequences are matched to the enrolled subject with minimum distance. This minimum distance is then compared to a threshold value; if the threshold is exceeded, the match is rejected and the observed sequences are classified as belonging to an imposter. Varying the threshold trades off false-positives (match of imposter accepted) versus false-negatives (match of a subject enrolled in the database rejected), yielding a ROC curve and AUC. Correctly rejecting imposters is harder in this setting because it suffices for an imposter to successfully impersonate any enrolled subject. In this experiment we also vary the number of subjects enrolled in the database.

In all three scenarios, the split of sequences into enrollment and observed sequences is repeated 10 times to obtain standard deviations of results. Moreover, accuracies and AUCs will increase with increasing n, as identification becomes easier the more data of an unknown subject is available.

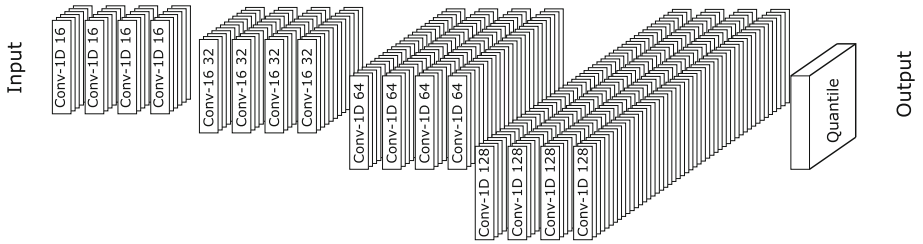

Fig. 2 The architecture used in the empirical study with an input sequence and an output as an embedding describing the input sequence. Each Convolution layer is a 1D convolution with a PReLU activation. The sequential architecture endings with a feature aggregation layer

5.3 Methods Under Study

Generally, the deep convolutional architecture proposed by [1] for biometric identification is studied as in Fig. 2, which consists of 16 stacked 1D-convolution layers with PReLU activation functions. In the experiments, the aggregation operation, loss function, and training algorithm are varied in order to evaluate the impact of these components on overall performance.

QP-WL: The method proposed in this paper, combining the quantile embeddings of Sect. 3 with the Wasserstein-based loss function and metric learning algorithm of Sect. 4. In all experiments, we set the parameter p of the distance function (see Definition 2) to one, that is, we use the Earth Mover Distance variant of the Wasserstein distance. The convolutional neural network architecture Γ of Sect. 3 is given by 16 stacked convolution layers with parametric RELU activations as defined by [1]. The number of sampling points for the quantile function is $M = 16$. At test time, distance between instances is given by the distance function from Definition 2.

QP-NPL: This method uses the same network architecture and quantile embedding as QP-WL. However, the resulting quantile embedding is then flattened into an $K \cdot M$ vector embedding, with entries $\bar{Q}_{\Gamma_k(\mathbf{s})}(\sigma(\alpha_m))$ for $k \in \{1, ..., K\}$ and $m \in \{1, ..., M\}$. Then the standard N-pair loss, which is based on cosine similarities between embedding vectors [27], is used for training. At test time, the distance between instances is given by negative cosine similarity. This method utilizes quantile-based aggregation and metric learning, but does not employ our Wasserstein-based loss function.

MP-NPL: This method uses the same basic network architecture as QP-NPL, but uses standard max-pooling instead of a quantile layer for global aggregation. This results in a K-dimensional embedding vector. As for QP-NPL, the model is trained using metric learning with the N-pair loss. At test time, distance is given by negative cosine similarity. This baseline uses metric learning, but neither quantile layers nor the Wasserstein-based loss function.

QP-CLS: This baseline uses the same network architecture and flattened quantile embedding as QP-NPL, but feeds the flattened embedding vector into a dense classification layer with softmax activation. The models is trained in a classification setting using multiclass crossentropy. Distance at test time is given by negative cosine similarity. This model is identical to the model presented in [1], except that we remove the final classification layer at test time to generate embeddings for novel subjects.

Table 1 Area under the ROC curve with standard error for all methods and domains in the verification setting for varying number $n \in \{1, 2, 3, 4, 5\}$ of observed sequences

Eye data	1 Video	2 Videos	3 Videos	4 Videos	5 Videos
QP-WL	**0.9466** ± 0.0032	**0.9716** ± 0.0020	**0.9799** ± 0.0013	**0.9837** ± 0.0008	**0.9860** ± 0.0005
QP-NPL	0.9345 ± 0.0033	0.9584 ± 0.0027	0.9667 ± 0.0020	0.9705 ± 0.0014	0.9738 ± 0.0010
MP-NPL	0.8890 ± 0.0035	0.9232 ± 0.0028	0.9334 ± 0.0017	0.9392 ± 0.0014	0.9437 ± 0.0016
QP-CLS	0.9007 ± 0.0053	0.9318 ± 0.0029	0.9424 ± 0.0025	0.9503 ± 0.0025	0.9538 ± 0.0026
Rigas et al. (2016)	0.7872 ± 0.0046	0.8649 ± 0.0054	0.8997 ± 0.0030	0.9190 ± 0.0031	0.9319 ± 0.0029
Eye data 2 features	27 Seconds	54 Seconds	81 Seconds	108 Seconds	135 Seconds
QP-WL	**0.8971** ± 0.0072	**0.8988** ± 0.0056	**0.9206** ± 0.0010	**0.9548** ± 0.0010	**0.9667** ± 0.0009
Jager et al. (2019)	0.7627 ± 0.0083	0.8326 ± 0.0057	0.8629 ± 0.0032	0.8833 ± 0.0026	0.8988 ± 0.0030
Shi and Jain (2019)	0.7670 ± 0.0062	0.8042 ± 0.0039	0.8272 ± 0.0022	0.8425 ± 0.0017	0.8512 ± 0.0019
Gait data	1 Minute	2 Minutes	3 Minutes	4 Minutes	5 Minutes
QP-WL	**0.9923** ± 0.0008	**0.9963** ± 0.0003	**0.9971** ± 0.0003	**0.9974** ± 0.0002	**0.9978** ± 0.0001
QP-NPL	0.9889 ± 0.0009	0.9932 ± 0.0004	0.9943 ± 0.0003	0.9947 ± 0.0002	0.9951 ± 0.0002
MP-NPL	0.9459 ± 0.0027	0.9624 ± 0.0027	0.9690 ± 0.0021	0.9735 ± 0.0016	0.9757 ± 0.0012
QP-CLS	0.9579 ± 0.0040	0.9756 ± 0.0018	0.9812 ± 0.0016	0.9856 ± 0.0011	0.9878 ± 0.0008
Shi and Jain (2019)	0.9611 ± 0.0026	0.9710 ± 0.0027	0.9756 ± 0.0014	0.9769 ± 0.0010	0.9786 ± 0.0010
EEG data	1 Second	2 Seconds	3 Seconds	4 Seconds	5 Seconds
QP-WL	**0.9968** ± 0.0006	**0.9985** ± 0.0001	**0.9988** ± 0.0001	**0.9991** ± 0.0000	**0.9992** ± 0.0000
QP-NPL	0.9927 ± 0.0005	0.9941 ± 0.0005	0.9953 ± 0.0003	0.9955 ± 0.0002	0.9959 ± 0.0001
MP-NPL	0.9611 ± 0.0012	0.9687 ± 0.0005	0.9713 ± 0.0005	0.9722 ± 0.0005	0.9732 ± 0.0005
QP-CLS	0.9796 ± 0.0017	0.9868 ± 0.0009	0.9901 ± 0.0010	0.9920 ± 0.0006	0.9923 ± 0.0007
Shi and Jain (2019)	0.8940 ± 0.0019	0.9184 ± 0.0021	0.9272 ± 0.0017	0.9314 ± 0.0011	0.9368 ± 0.0008

The bold values in the table indicate the highest recorded AUC value for each data set and observation setting

Shi and Jain (2019): The distributional embedding method studied by Shi and Jain [26]. This method represents the distribution over learned features by Gaussian distributions represented by means and variances, and uses a loss function that aims to maximize the mutual likelihood score for genuine pairs (see [26] for details). As embedding network, we use the same architecture as for the other four methods, see Fig. 2.

For all methods, training is carried out using the Adam optimizer with learning rate 0.0001 for 50,000 iterations, and the regularizer of the PReLU activation function is tuned as a hyperparameter on the validation set as in [1].

5.4 Results

The empirical results for the different domains are presented and discussed in turn.

5.4.1 Eye Movements

Table 1, upper third, shows area under the ROC curve for all methods and varying number n of observed sequences in the eye movement domain. As expected, AUC increases with the

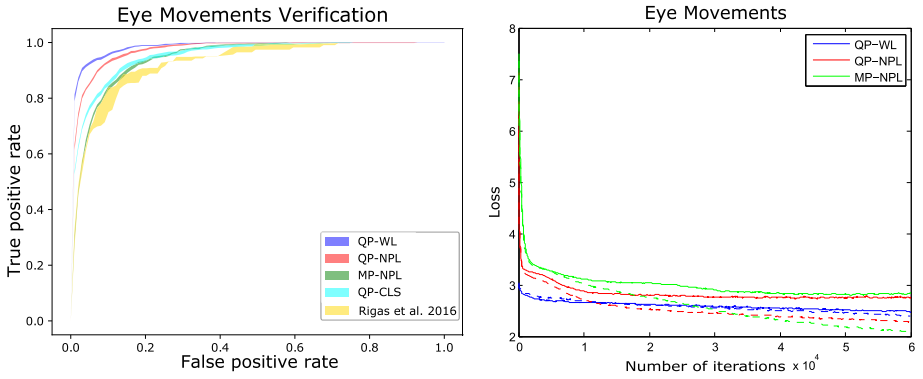

Fig. 3 Left: ROC curves in the eye movement domain for all methods using $n = 5$ observed sequences. Shaded region in ROC curves indicates standard error. Right: Training (dashed lines) and test (solid lines) loss during training as a function of the number of training iterations

number n of sequences observed at test time. Comparing QP-WL and QP-NPL, we observe that the Wasserstein-based loss introduced in Sect. 4, which works on the distributional embedding given by the piecewise linear approximations of the quantile functions, clearly outperforms flattening the distributional embedding and using N-pair loss. Comparing MP-NPL with QP-NPL and QP-WL shows that using quantile layers improves accuracy compared to max-pooling even if the quantile embedding is flattened (and more so if Wasserstein-based loss is used). Classification training (QP-CLS) reduces accuracy compared to metric learning (QP-NPL). The difference between the simplest method QP-CLS and the proposed model QP-WL is substantial: at $n = 1$ AUC increases from 0.9007 to 0.9466, an almost two-fold reduction in AUC error. Figure 3 (left) shows ROC curves in the verification setting for $n = 5$, again showing significant improvement for using the Wasserstein-based loss, metric learning, and quantile representation. Figure 3 (right) shows training (dashed lines) and test (solid lines) loss during training as a function of the number of training iterations.

Figure 4 (left) shows multiclass identification accuracy for $n = 5$ observed sequences as a function of the fraction of the 105 subjects who are enrolled. Relative results for the different methods are similar as in the verification setting. Accuracy decreases slightly when more subjects are enrolled, as the multiclass problem becomes more difficult. Figure 4 (right) shows the robustness of the model to multiclass imposters as a function of the fraction of the 105 subjects who are enrolled (up to 50%, as half of the subjects are imposters). We observe that QP-WL is much more robust to imposters than the baseline methods.

In the eye movement domain, we also compare against the state-of-the-art model by [13], denoted Jäger et al. (2019). Jäger et al. (2019) uses angular gaze velocities averaged over left and right eye as input, which we compute from our raw data. The setting of [13] is replicated by training the model using multiclass classification and using the last layer before the classification layer as the embedding at test time. The Jäger et al. (2019) architecture cannot deal with variable-length sequences, we therefore split the variable-length sequences in our data into shorter sequences of fixed length, namely the length of the shortest sequence (27 s). For a fair comparison, we also reduce the information given to our model in this experiment: using only the average gaze point rather than left and right gaze point separately, removing pupil dilation, and using the same fixed-length sequences.

Table 1, in section "Eye data 2 features", shows area under the ROC curve for our method QP-WL and Jäger et al. (2019) for varying number n of observed sequences on the simplified

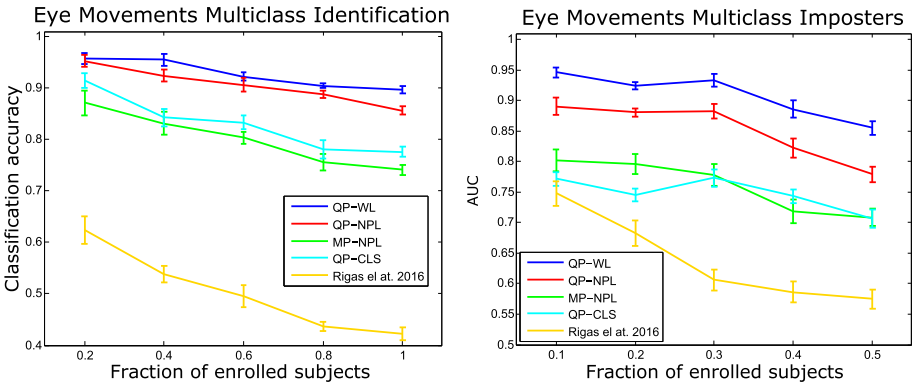

Fig. 4 Left: Identification accuracy in the multiclass identification scenario for the eye movement domain and $n = 5$ observed test instances as a function of the fraction of subjects that are enrolled. Right: area under the ROC curve for multiclass imposters as a function of the fraction of subjects enrolled. In the imposter scenario, 50% of subjects are imposters and therefore never enrolled. Error bars indicate the standard error

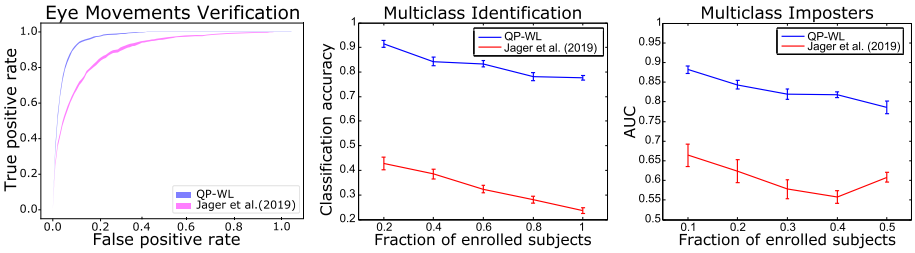

Fig. 5 Comparison between QP-WL and Jäger et al. (2019) in the eye movement domain: area under ROC curve in verification scenario (left), identification accuracy in multiclass identification scenario (center), and robustness of model to multiclass imposters (right). In this experiment, the data is simplified for both methods to match the requirements of Jäger et al. (2019), see text for details. Results of QP-WL therefore differ from results presented in Figs. 3 and 4. Error bars indicate the standard error

data. It can be observed that QP-WL achieves much higher AUC values. Comparing to the results on the full eye movement data, it can be observed that accuracies are reduced for our model, but the model outperforms Jäger et al. (2019) by a wide margin. Figure 5 shows ROC curves for the verification scenario (left) and identification accuracy (center) as well as AUC in the imposter scenario for QP-WL and Jäger et al. (2019). Again, comparing to Figs. 3 and 4 we observe a reduction in accuracy, but QP-WL strongly outperforms Jäger et al. (2019). We note that the model of [13] is focused on microsaccades, which are likely not detectable in our data due to the low temporal resolution (30 Hz compared to 1000 Hz in the study by [13]), which might explain the relatively poor performance of the model on our data.

We finally compare against the model of Shi and Jain (2019) in the eye movement domain. Because this model is also formulated for fixed-length vectors rather than variable-length sequences, we again carry out the comparison on the simplified eye movement data. Table 1, in section "Eye data 2 features", shows area under the ROC curve for Shi and Jain (2019) in comparison to QP-WL and Jäger et al. (2019). QP-WL yields higher AUC for all numbers n of observed sequences.

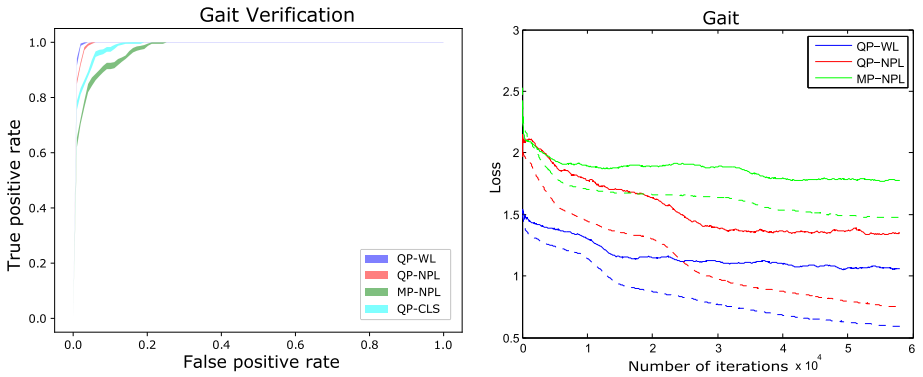

Fig. 6 Left: ROC curves in the gait domain for all methods using $n = 5$ observed sequences. Shaded region in ROC curves indicates standard error. Right: Training (dashed lines) and test (solid lines) loss during training as a function of the number of training iterations

5.4.2 Gait

Table 1, center third, shows area under the ROC curve for all methods and varying number n of observed sequences in the gait domain. Generally, it can be observed that individuals can be identified very accurately based on the accelerometer observations, with the area under the ROC curve close to the optimal value of one. As expected, predictive performance increases with increasing amounts of sensor data available to make a decision (going from one minute to five minutes of data). In terms of relative performance between the different methods, clear benefits can be observed when using the proposed loss function based on Wasserstein distance (QP-WL vs. QP-NPL), when using quantile layers instead of max-pooling aggregation (QP-WL and QP-NPL vs. MP-NPL), and when using metric learning rather than classification-based training (QP-NPL vs. QP-CLS). Compared to the simplest model QP-CLS, the proposed method increases the AUC from 0.9579 to 0.9923, a reduction in AUC error of more than a factor of five. Figure 6 (left) shows ROC curves for verification at $n = 5$ in the gait domain. Again, ROC curves show clear benefits for the Wasserstein-based loss function, metric learning approach, and quantile-based representation. Figure 6 (right) shows training (dashed lines) and test (solid lines) loss during training as a function of the number of training iterations in the gait domain.

Figure 7 (left) shows identification accuracy as a function of the fraction of subjects enrolled in the gait domain; in this setup the ordering of methods in terms of performance is the same but the difference between QP-WL and QP-NPL less pronounced. Figure 7 (right) shows robustness to multiclass imposters, with again a clear advantage of QP-WL over the baselines.

Table 1, center third, also shows results for the model of Shi and Jain (2019) in the Gait domain. In this domain, the Shi and Jain model is competitive with MP-NPL and QP-CLS, but does not reach the AUC of QP-WL and QP-NPL, showing that the combination of quantile embeddings and Wasserstein-based loss function is again superior to distributional embeddings based on Gaussian distributions.

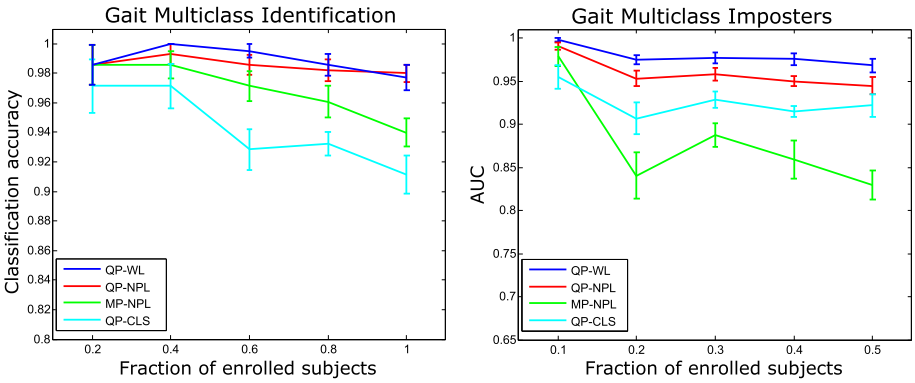

Fig. 7 Left: Identification accuracy in the multiclass identification scenario for the gait domain and $n = 5$ observed test instances as a function of the fraction of subjects that are enrolled. Right: area under the ROC curve for multiclass imposters as a function of the fraction of subjects enrolled. In the imposter scenario, 50% of subjects are imposters and therefore never enrolled. Error bars indicate the standard error

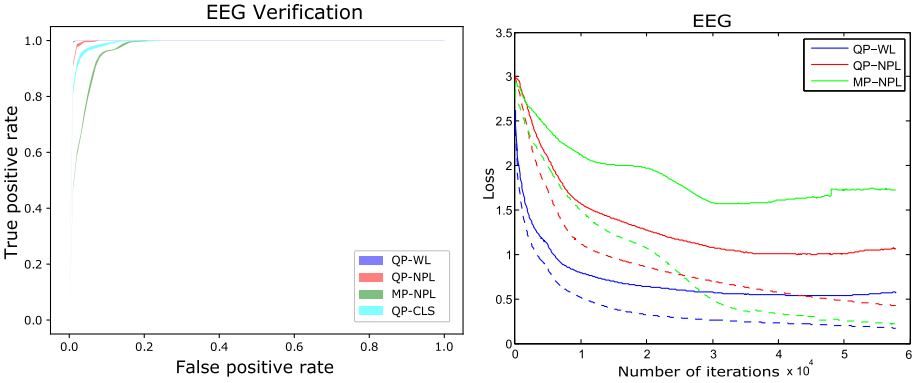

Fig. 8 Left: ROC curves in the EEG domain for all methods using $n = 5$ observed sequences. Shaded region in ROC curves indicates standard error. Right: Training (dashed lines) and test (solid lines) loss during training as a function of the number of training iterations

5.4.3 EEG

Table 1, bottom third, shows area under the ROC curve for all methods and varying number n of observed test sequences in the EEG domain. Relative performance of methods is generally similar as in the other two domains. QP-WL clearly outperforms the closest baseline, reducing 1-AUC by between 56% ($n = 1$) and 80% ($n = 5$). Figure 8 (left) shows ROC curves in the verification setting. Figure 8 (right) shows training (dashed lines) and test (solid lines) loss during training as a function of the number of training iterations in the EEG domain.

Figure 9 (left) and 9 (right) show identification accuracy as a function of the fraction of subjects enrolled and robustness of the models to multiclass imposters. As in the gait domain, differences are more pronounced in the latter setting.

Table 1, bottom third, also shows results for the model of Shi and Jain (2019) in the EEG domain. In this domain, the Shi and Jain model does not yield competitive results.

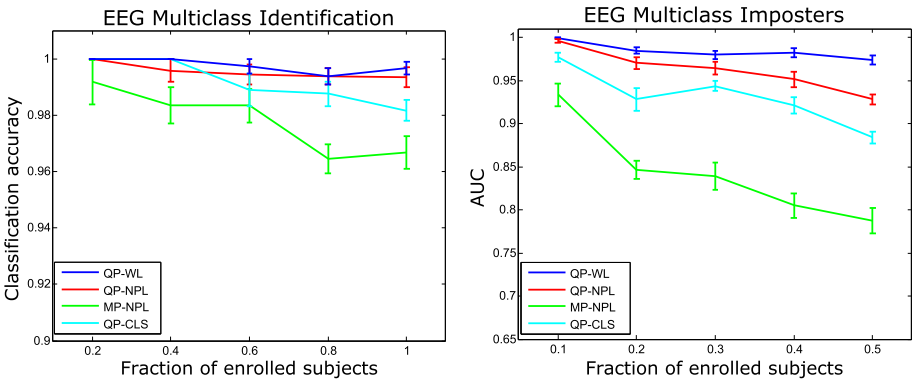

Fig. 9 Left: Identification accuracy in the multiclass identification scenario for the EEG domain and $n = 5$ observed test instances as a function of the fraction of subjects that are enrolled. Right: area under the ROC curve for multiclass imposters as a function of the fraction of subjects enrolled. In the imposter scenario, 50% of subjects are imposters and therefore never enrolled. Error bars indicate the standard error

6 Conclusions and Discussion

We developed a model for distributional embeddings of variable-length sequences using deep neural networks. Building on existing work on quantile layers, the model represents an instance by the distribution of the learned deep features across the sequence. We developed a distance function for these distributional embeddings based on the Wasserstein distance between the corresponding distributions, and from this distance function a loss function for performing metric learning with the proposed model. A key point about the model is end-to-end learnability: by using piecewise linear approximations of the quantile functions, and based on those providing a closed-form solution for the Wasserstein distance, gradients can be traced through the embedding and loss calculations.

In our empirical study, distributional embeddings outperformed standard vector embeddings by a large margin on three data sets for biometric identification based on eye movements, gait, and EEG measurements. Key empirical results that show this advantage are presented in Table 1 and the ROC curves shown in Fig. 3 for eye movement data, Fig. 6 for gait data, and Fig. 8 for EEG data, each time comparing QP-WL and QP-NPL. From a theoretical perspective, these gains can be explained by the intuition that distributional embeddings are better able to capture the distribution of local, short-term pattern in the sequences, which are a key signal for distinguishing subjects in the domains we study.

In principle, the method is generally applicable to any sequence classification problem were the goal is to obtain embeddings of sequences, as for example in most biometric settings. The particular strength of the probabilistic embedding and loss function proposed in this paper lies in being able to capture well the distribution of local patterns appearing in the sequences, which is particularly relevant for the biometric identification problems we have studied. It will likely also work well on other biometric problems where similar low-level sensor data is used for identification. However, the model will be likely less useful for sequence data were more large-scale or long-range patterns are important.

Acknowledgements This work was partially funded by the German Research Foundation under grant LA3270/1-1

Funding Open Access funding enabled and organized by Projekt DEAL.

References

1. Abdelwahab A, Landwehr N (2019) Quantile layers: statistical aggregation in deep neural networks for eye movement biometrics. In: Proceedings of the 30th European conference on machine learning
2. Arjovsky M, Chintala S, Bottou L (2017) Wasserstein generative adversarial networks. In: International conference on machine learning, pp 214–223
3. Athiwaratkun B, Wilson A (2017) Multimodal word distributions. In: Proceedings of the 55th annual meeting of the association for computational linguistics (volume 1: long papers), pp 1645–1656
4. Bojchevski A, Günnemann S (2018) Deep Gaussian embedding of graphs: Unsupervised inductive learning via ranking. In: International conference on learning representations, pp 1–13
5. Bucher M, Herbin S, Jurie F (2016) Improving semantic embedding consistency by metric learning for zero-shot classification. In: European conference on computer vision. Springer, pp 730–746
6. Cambanis S, Simons G, Stout W (1976) Inequalities for ek (x, y) when the marginals are fixed. Z Wahrscheinlichkeitstheorie und verwandte Gebiete 36(4):285–294
7. Chung JS, Nagrani A, Zisserman A (2018) Voxceleb2: deep speaker recognition. Proc Interspeech 2018:1086–1090
8. Frogner C, Zhang C, Mobahi H, Araya M, Poggio TA (2015) Learning with a Wasserstein loss. In: Advances in neural information processing systems, pp 2053–2061
9. Gao R, Kleywegt AJ (2016) Distributionally robust stochastic optimization with Wasserstein distance. arXiv preprint arXiv:1604.02199
10. Gibiansky A, Arik S, Diamos G, Miller J, Peng K, Ping W, Raiman J, Zhou Y (2017) Deep voice 2: multi-speaker neural text-to-speech. In: Advances in neural information processing systems, pp 2962–2970
11. Hadsell R, Chopra S, LeCun Y (2006) Dimensionality reduction by learning an invariant mapping. In: 2006 IEEE computer society conference on computer vision and pattern recognition, vol 2. IEEE, pp 1735–1742
12. Ihlen EA, Weiss A, Helbostad JL, Hausdorff JM (2015) The discriminant value of phase-dependent local dynamic stability of daily life walking in older adult community-dwelling fallers and nonfallers. BioMed Res Int
13. Jäger L, Makowski S, Prasse P, Liehr S, Seidler M, Scheffer T (2019) Deep eyedentification: Biometric identification using micro-movements of the eye. In: Proceedings of the 30th European conference on machine learning
14. Li C, Ma X, Jiang B, Li X, Zhang X, Liu X, Cao Y, Kannan A, Zhu Z (2017) Deep speaker: an end-to-end neural speaker embedding system. arXiv preprint arXiv:1705.02304
15. McLaughlin N, Martinez del Rincon J, Miller P (2016) Recurrent convolutional network for video-based person re-identification. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1325–1334
16. Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J (2013) Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems, pp 3111–3119
17. Mital PK, Smith TJ, Hill RL, Henderson JM (2011) Clustering of gaze during dynamic scene viewing is predicted by motion. Cogn Comput 3(1):5–24
18. Mueller J, Thyagarajan A (2016) Siamese recurrent architectures for learning sentence similarity. In: Thirtieth AAAI conference on artificial intelligence
19. Neculoiu P, Versteegh M, Rotaru M (2016) Learning text similarity with SIAMESE recurrent networks. In: Proceedings of the 1st workshop on representation learning for NLP, pp 148–157
20. Oh Song H, Xiang Y, Jegelka S, Savarese S (2016) Deep metric learning via lifted structured feature embedding. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4004–4012
21. Reed S, Akata Z, Lee H, Schiele B (2016) Learning deep representations of fine-grained visual descriptions. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 49–58

22. Resnick SI (2013) Extreme values, regular variation and point processes. Springer, New York
23. Rigas I, Komogortsev O, Shadmehr R (2016) Biometric recognition via eye movements: saccadic vigor and acceleration cues. ACM Trans Appl Percept (TAP) 13(2):1–21
24. Schroff F, Kalenichenko D, Philbin J (2015) Facenet: A unified embedding for face recognition and clustering. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 815–823
25. Sedighi V, Fridrich J (2017) Histogram layer, moving convolutional neural networks towards feature-based steganalysis. Electron Imaging 7:50–55
26. Shi Y, Jain AK (2019) Probabilistic face embeddings. In: Proceedings of the IEEE international conference on computer vision, pp 6902–6911
27. Sohn K (2016) Improved deep metric learning with multi-class n-pair loss objective. In: Advances in neural information processing systems, pp 1857–1865
28. Vilnis L, McCallum A (2015) Word representations via Gaussian embedding. In: International conference on learning representations (ICLR)
29. Wang J, Zhou F, Wen S, Liu X, Lin Y (2017) Deep metric learning with angular loss. In: Proceedings of the IEEE international conference on computer vision, pp 2593–2601
30. Wang Y, Pan X, Song S, Zhang H, Huang G, Wu C (2019) Implicit semantic data augmentation for deep networks. Adv Neural Inf Process Syst 32:12635–12644
31. Wang Z, Li H, Ouyang W, Wang X (2016) Learnable histogram: statistical context features for deep neural networks. In: European conference on computer vision. Springer, pp 246–262
32. Wu CY, Manmatha R, Smola AJ, Krahenbuhl P (2017) Sampling matters in deep embedding learning. In: Proceedings of the IEEE international conference on computer vision, pp 2840–2848
33. Wu L, Wang Y, Gao J, Li X (2018) Where-and-when to look: deep SIAMESE attention networks for video-based person re-identification. IEEE Trans Multimed 21(6):1412–1424
34. Yu T, Li D, Yang Y, Hospedales TM, Xiang T (2019) Robust person re-identification by modelling feature uncertainty. In: Proceedings of the IEEE international conference on computer vision, pp 552–561
35. Yuan Y, Yang K, Zhang C (2017) Hard-aware deeply cascaded embedding. In: Proceedings of the IEEE international conference on computer vision, pp 814–823
36. Zhang XL, Begleiter H, Porjesz B, Wang W, Litke A (1995) Event related potentials during object recognition tasks. Brain Res Bull 38(6):531–538