IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received May 8, 2020, accepted May 20, 2020, date of publication June 1, 2020, date of current version June 12, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2999055

Learning Acoustic Word Embeddings With Dynamic Time Warping Triplet Networks

DENIS SHITOV[1], ELENA PIROGOVA[1], TADEUSZ A. WYSOCKI[2,3], (Senior Member, IEEE), AND MARGARET LECH[1], (Member, IEEE)

[1]School of Engineering, RMIT University, Melbourne, VIC 3000, Australia
[2]College of Electrical and Computer Engineering, University of Nebraska-Lincoln, Lincoln, NE 68588, USA
[3]Faculty of Telecommunications, Computer Science and Electrical Engineering, UTP University of Science and Technology, 85-796 Bydgoszcz, Poland

Corresponding author: Denis Shitov (3628075@student.rmit.edu.au)

This work was supported by the RMIT University Postgraduate Scholarship.

ABSTRACT In the last years, acoustic word embeddings (AWEs) have gained significant interest in the research community. It applies specifically to the application of acoustic embeddings in the Query-by-Example Spoken Term Detection (QbE-STD) search and related word discrimination tasks. It has been shown that AWEs learned for the word or phone classification in one or several languages can outperform approaches that use dynamic time warping (DTW). In this paper, a new method of learning AWEs in the DTW framework is proposed. It employs a multitask triplet neural network to generate the AWEs. The triplet network learns acoustic representations of words through a comparison of DTW distances. In addition, a multitask objective, including a conventional word classification component, and a triplet loss component is proposed. The triplet loss component applies the DTW distance for the word discrimination task. The multitask objective ensures that the embeddings can be used with DTW directly. Experimental validation shows that the proposed approach is well-suited, but not necessarily restricted to the QbE-STD search. A comparison with several baseline methods shows that the new method leads to a significant improvement of the results on the word discrimination task. An evaluation of the word clustering in the learned embedding space is presented.

INDEX TERMS Acoustic word embedding, dynamic time warping, triplet network, query-by-example.

I. INTRODUCTION

The Dynamic Time Warping (DTW) algorithm, introduced more than two decades ago [1], finds the optimal alignment between points of two time-series. By using a nonlinear mapping of samples from one time-series into another one, this method achieves an effective alignment despite possible local temporal or phase distortions issues. The DTW and its variants have been applied in speech recognition [1], Query-by-Example (QbE) search [2], bioinformatics [3], and post-stroke rehabilitation [4].

In speech recognition and classification tasks, the DTW is used along with Mel-frequency cepstral coefficients (MFCCs) as a feature representation of acoustic time-series. However, it has been shown that direct application of MFCCs into DTW often becomes a limiting factor affecting the overall system performance since the same speech units can be pronounced differently by different speakers. These differences are natural results of high variability in the physical anatomy of the human vocal tract depending on factors such as the speaker's sex, age, accent, cognitive load, or emotional state.

In the Query-by-Example Spoken Term Detection (QbE-STD), several speech segment representations such as phonetic and Gaussian posteriorgrams [5]–[7] have been proposed. When combined with DTW, they have shown encouraging results. One of the main advantages of these methods is their applicability in low-resource scenarios. However, some aspects of speech, such as pronunciation variability, negatively affect the performance of sub-word models and remain a research challenge.

Meanwhile, the recent advent of deep learning techniques created the potentials for new powerful approaches in the speech processing field. In particular, acoustic word embeddings (AWEs) based on various types of deep learning frameworks are attracting increasing interest in the research community [8]–[11], [12], [13]. AWEs are acoustic feature

The associate editor coordinating the review of this manuscript and approving it for publication was Alberto Cano.

representations that can be learned directly from speech data. The aim is to create "less distant" time series embeddings for the same words and "more distant" for different words. The notion of embedding space and the distance between embeddings can vary. The model, proposed in [14], for example, focuses on semantic similarity. It has an encoder-decoder structure. The encoder takes acoustic input and maps it into a fixed-sized vector representation. The decoder, in turn, uses the vector representation to predict neighboring acoustic segments within a specified range of the data corpus. In [15], a new loss function is proposed and applied to directly train acoustic embeddings that retain the phonetic distance. It was achieved by minimizing the difference between embedding distance and phonetic edit distance between words. In [16], orthographic representations of words were included in their acoustic embeddings using a multi-view representation learning setting.

In general, existing approaches solely rely on the acoustic representation of words when learning their embeddings. The majority of them incorporate the training of Siamese neural networks. A Siamese network consists of a pair of coupled networks that take two speech segments as inputs and produce their embeddings, trained to minimize a hinge loss that separates the same-word pairs from different-word pairs by a given margin [8], [10]–[12]. An alternative method proposed in these studies was a single Convolutional Neural Network (CNN) trained as a word classifier, which was found to perform similarly to the Siamese CNN networks. In [8], the best Average Precision (AP) on the word discrimination task was reported to achieve a value of 0.549.

It was shown in a recent study [10] that a Siamese network consisting of two Long Short-Term Memory (LSTM) networks [17] outperformed earlier benchmarks achieving AP of 0.671. Embeddings generated by these methods had a fixed length regardless of the length of the input sequences. A comparison between the fixed-length embeddings can be made using the cosine similarity measure. However, the fixed-length nature of the embeddings makes them not suitable for the DTW framework.

Another recent study [18] proposed to use AWEs learned by feed-forward neural networks as inputs to the DTW algorithm. This approach has shown state-of-the-art performance on the QbE-STD task. However, the learning of these embeddings was not optimized for their use within the DTW algorithm. In other words, there was a mismatch between how the AWEs were trained and how they were employed later within the DTW algorithm.

In summary, there has been limited research on the applicability of AWEs in the canonical DTW framework. To address this limitation we investigate how different learning objectives of AWEs affect the performance of the DTW algorithm when embeddings are applied as input to the DTW. We evaluate the DTW performance on a word discrimination task, which is related to the QbE-STD task. As a significant contribution of this study, we propose a novel loss function for learning AWEs. In contrast to previous related work [10], [15], [16], we use DTW distance as a pairwise distance measure in the triplet loss, which allows learning acoustic word embeddings in a way that similar words have a small DTW distance and remain close to each other in the embedding space, while non-similar words occur far apart from each other and have a high DTW distance.

Additionally, we investigate whether combining a traditional word recognition objective and a triplet loss leads to an improvement in AWEs learning. We show that, if applicable, this training scenario gives significantly better results compared to corresponding single-objective settings. As shown in Section IV, AWEs learned with the proposed objective improve the discrimination performance of the DTW algorithm, in comparison with the AWEs obtained through training a word recognition neural network or a Siamese network trained to discriminate different words [10]. We also perform an analysis of the embedding space and visualize the clustering of the AWEs in that space. The code for training and experimental results of the proposed model is available for the community on Github.[1]

II. METHOD

In this section, we present the methods and concepts employed in this study and discuss our approach in detail.

A. SOFT-DTW

Soft-DTW is a variant of DTW introduced in [19]. It provides a smoothed formulation of DTW, which calculates the soft-minimum of all alignment costs as opposed to finding the minimal-cost alignment proposed in the original DTW formulation. Soft-DTW holds a valuable property that makes it particularly well-suited for application in the Deep Learning framework. As shown in [19], it is a differentiable loss function. Both its value and gradient can be calculated with $\mathcal{O}(nm)$ time/space complexity, where n and m denote the respective lengths of the two sequences.

The objective of soft-DTW can be defined as follows:

$$dtw_{\gamma}(\mathbf{x}, \mathbf{y}) := min^{\gamma}\{\langle A, \Delta(\mathbf{x}, \mathbf{y})\rangle, A \in \mathcal{A}_{n,m}\}, \tag{1}$$

where $\Delta(\mathbf{x}, \mathbf{y}) := [\delta(x_i, y_j)]_{ij} \in \mathbb{R}^{n\times m}$ is the cost matrix of pairwise distances between $\mathbf{x}$ and $\mathbf{y}$ time-series of lengths n and m, respectively. The inner product $\langle A, \Delta(x, y)\rangle$ of that matrix with an alignment matrix A gives the score of this A alignment. The generalized *min* operator is defined as:

$$min^{\gamma}(a_1, \ldots, a_n) = \begin{cases} min_{i\leq n} a_i, & \gamma = 0 \\ -\gamma \log \sum_{i=1}^{n} e^{-a_i/\gamma}, & \gamma > 0 \end{cases} \tag{2}$$

By varying the smoothing parameter γ, we can control the impact of non-optimal alignments on the resulting score of the DTW. When $\gamma = 0$, we recover the original $\mathrm{DTW}(\mathbf{x}, \mathbf{y}) := \min_{A\in\mathcal{A}_{n,m}} \langle A, \Delta(\mathbf{x}, \mathbf{y})\rangle$, which only considers the optimal alignment. Further details such as gradient computation via algorithmic differentiation and an analysis of the smoothing parameter γ can be found in [19]. An open-source

[1] codebase: `https://github.com/qdenisq/Soft-DTW-AWE`

implementation of Soft-DTW, written for PyTorch [20], was used in this study.

B. THE TRIPLET NETWORK

As described in [21], a *triplet network* is comprised of 3 instances of the same feed-forward network with shared parameters. When fed with three input samples, one of which is the *reference* (or *anchor*) sample, the network outputs two intermediate values denoting the δ distances between the embeddings generated by the reference input and the embeddings generated by the two other input samples. We denote the triplet of inputs as $\mathbf{x}, \mathbf{x}^+$, and $\mathbf{x}^-$. Where, $\mathbf{x}$ denotes an *anchor* input sample, $\mathbf{x}^+$ is an input sample of the same class as the anchor $\mathbf{x}$, whereas $\mathbf{x}^-$ is a sample that belongs to a different class. Using $Net(\mathbf{x})$, $Net(\mathbf{x}^+)$ and $Net(\mathbf{x}^-)$ to denote outputs from the shared embedding sub-networks, we can express the output of the triplet network as follows:

$$TripletNet\left(\mathbf{x}, \mathbf{x}^-, \mathbf{x}^+\right) = \begin{bmatrix} \delta\left[Net(\mathbf{x}), N\ et\left(\mathbf{x}^-\right)\right] \\ \delta\left[Net(\mathbf{x}), N\ et\left(\mathbf{x}^+\right)\right] \end{bmatrix} \in \mathbb{R}^2_+ \quad (3)$$

It is important to note that in the approach proposed here, the output of the embedding model $Net(\mathbf{x})$ is a stack of the frame-level embeddings $Net(x_t)$. This is in contrast with [10], where only an embedding of the last frame $Net(x_T)$ was used as an output.

A schematic view of the triplet network is shown in Fig. 1. Each of the input samples $(\mathbf{x}, \mathbf{x}^+, \mathbf{x}^-)$ is fed independently into the embedding sub-network Net, which could be given either as a Feedforward Neural Network, Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), or any other type of a non-linear function approximator. Embeddings are then used to compute pairwise distances $\delta\left[Net(\mathbf{x}), Net\left(\mathbf{x}^+\right)\right]$ between the outcomes given for the anchor $Net(\mathbf{x})$ and for the positive sample $Net(\mathbf{x}^+)$ (positive pair), and $\delta\left[Net(\mathbf{x}), Net\left(\mathbf{x}^-\right)\right]$ between an the outcome for the anchor $Net(\mathbf{x})$ and for the negative sample $Net(\mathbf{x}^-)$ (negative pair). In [21], the Euclidean distance was used as a distance measure between samples. Given that the proposed approach has no constraints with regards to the choice of the distance measure aside from it being differentiable, as we show in the following sections, the Euclidean distance can be substituted by the Soft-DTW-distance. The two distances (one for the positive and one for the negative pair of embeddings), are fed directly into a comparator.

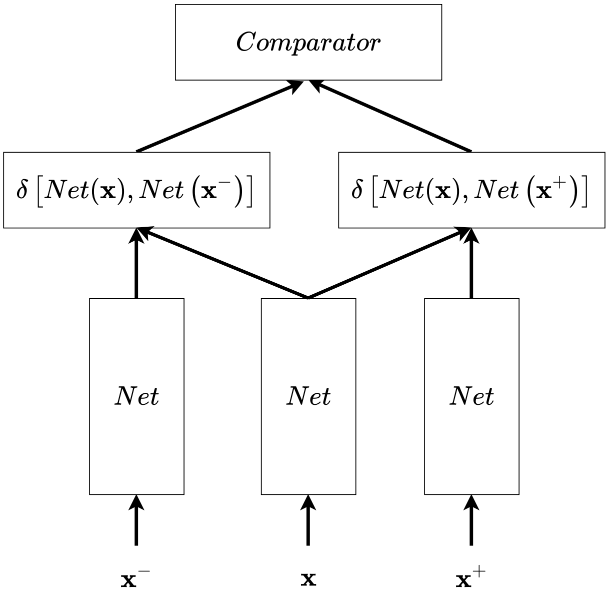

Comparator
$\delta\left[Net(\mathbf{x}), Net\left(\mathbf{x}^-\right)\right]$
$\delta\left[Net(\mathbf{x}), Net\left(\mathbf{x}^+\right)\right]$
Net
Net
Net
$\mathbf{x}^-$
$\mathbf{x}$
$\mathbf{x}^+$

FIGURE 1. Triplet network structure (adapted from [21]).

The loss function used to train the triplet network is given as:

$$\mathcal{L}(\mathbf{x}, \mathbf{x}^+, \mathbf{x}^-) = \max(\delta\left[Net(\mathbf{x}), Net\left(\mathbf{x}^+\right)\right] \\ -\delta\left[Net(\mathbf{x}), Net\left(\mathbf{x}^-\right)\right] + \alpha, 0), \quad (4)$$

where the distance from anchor to the positive sample is minimized, while the distance from anchor to the negative sample is maximized. By introducing the margin parameter a, we can define three categories of triplets that could be sampled during training:

- **easy triplets**: triplets which have a loss of 0, because $\delta\left[Net(\mathbf{x}), N\ et\left(\mathbf{x}^+\right)\right] + \alpha < \delta\left[Net(\mathbf{x}), Net\left(\mathbf{x}^-\right)\right]$
- **hard triplets**: triplets where the negative sample is closer to the anchor than the positive sample, $\delta\left[Net(\mathbf{x}), Net\left(\mathbf{x}^+\right)\right] > \delta\left[Net(\mathbf{x}), Net\left(\mathbf{x}^-\right)\right]$
- **semi-hard triplets**: triplets where the positive sample is closer to the anchor than the negative sample, but which still results in a positive loss, $\delta\left[Net(\mathbf{x}), Net\left(\mathbf{x}^+\right)\right] < \delta\left[Net(\mathbf{x}), Net\left(\mathbf{x}^-\right)\right] < \delta\left[Net(\mathbf{x}), Net\left(\mathbf{x}^+\right)\right] + \alpha$

As shown in [22], [23], triplet selection for training triplet network is equally important as the choice of the loss function. According to [23], different sampling strategies can lead to drastically different solutions for the same loss function. On the other hand, many different loss functions can perform similarly under a given sampling strategy.

As discussed in [23], selecting the hardest negative data sampling strategy leads to a bad local minimum early on in training and often results in a collapsed model when all embeddings converge into the same value irrespective of the input (i.e. $Net(\mathbf{x}) = 0$). Thus, based on the results from [22], we have chosen the online semi-hard triplet mining strategy. Online mining means that we select positive/negative samples from within a minibatch instead of generating triplets on the whole training set. Semi-hard mining means that for each anchor and positive sample we randomly select a negative sample that satisfies the criteria of semi-hard triplet: $\delta\left[Net(\mathbf{x}), Net\left(\mathbf{x}^+\right)\right] < \delta\left[Net(\mathbf{x}), Net\left(\mathbf{x}^-\right)\right] < \delta\left[Net(\mathbf{x}), Net\left(\mathbf{x}^+\right)\right] + \alpha$. In other words, we select negative samples such that they lie within a margin α, but it is still hard to learn as the distance from the anchor to negative is close to the distance from the anchor to the positive sample.

C. PROPOSED LEARNING OBJECTIVE

As discussed in Section II-B, triplet loss given in (4) does not have any particular constraints regarding the distance measure between embeddings generated by the input samples.

Thus, we employ soft-DTW distance (1) in the margin-based triplet loss as follows:

$$\mathcal{L}_{triplet}(\mathbf{x}, \mathbf{x}^+, \mathbf{x}^-) = \max(dtw_\gamma\left[f_\theta(\mathbf{x}), f_\theta\left(\mathbf{x}^+\right)\right] - dtw_\gamma\left[f_\theta(\mathbf{x}), f_\theta\left(\mathbf{x}^-\right)\right] + \alpha, 0), \quad (5)$$

where f_θ is an embedding model with parameters θ. As discussed in detail in Section II-D, we use a neural network as an embedding model, therefore parameters θ are simply weights and biases of the neural network. As mentioned before, the triplet loss function is designed to learn embeddings that will be spanned closely in the embedding space for similar samples and far apart for non-similar ones. Unlike in previous studies [10], [24] where embeddings were given as fixed-length multidimensional vectors, our embedding model f_θ outputs a sequence of embeddings with the length corresponding to the length of the input sequence $\mathbf{x}$. We employ the property of soft-DTW that allows effective calculation of the distance between time-series of different lengths; therefore, the network is designed to generate embedding outputs of varying lengths. This approach allows for capturing additional temporal information that might be missing when fixed-length embeddings are used.

Another advantage of using soft-DTW in triplet loss is that it eliminates a potential mismatch between the learning objective and the application of the embedding model in the DTW algorithm (e.g. QbE-STD task when DTW is used to detect the occurrence of the query in the reference set). We conducted a set of experiments to test the hypothesis that to achieve the best DWT performance, the embedding model should be trained in a way consistent with the application conditions. To check this hypothesis, the same embedding model was trained in several different ways, and its applicability in the DTW algorithm was evaluated. The experimental details and results are presented in Section III. To facilitate faster and more stable convergence of the embedding model training procedure, we are proposing a new multi-objective loss function combining the classification loss and the triplet loss. The proposed multi-objective loss function is defined as follows:

$$\mathcal{L} = (1-\lambda)\mathcal{L}_{CE} + \lambda\mathcal{L}_{triplet}, \quad (6)$$

$$\mathcal{L}_{CE} = -\sum_{n=1}^{3N}\sum_{i=1}^{C} I\left\{y_n = i\right\} \log \frac{e^{\mathbf{W}_i^T f_\theta(\mathbf{x}_n)+b_i}}{\sum_{j=1}^{C} e^{\mathbf{W}_j^T f_\theta(\mathbf{x}_n)+b_j}}, \quad (7)$$

where $\mathcal{L}_{triplet}$ is the triplet loss function, $\mathcal{L}_{CE}$ is a cross-entropy loss obtained from the softmax with C classes, N is a number of triplets within a minibatch, $3N$ refers to three word examples of words per triplet, $f_\theta(\mathbf{x}_n)$ is the embedding of the n-th sample, $\mathbf{W}$ and b are the weights and the bias of the linear layer attached to the last layer of the embedding model respectively, and λ is a time-varying hyper-parameter controlling the trade-off between the two terms of the loss function.

We define the time-varying hyper-parameter λ as follows:

$$\lambda(i) = \frac{a}{1+e^{-k(i-b)}}, \quad (8)$$

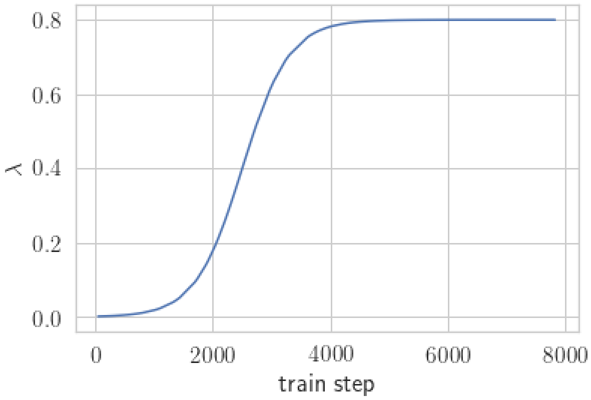


FIGURE 2. Time-varying hyper-parameter λ controlling the trade-off between the two terms of the loss function.

where i is a training step, a, k and b are constant parameters experimentally chosen to be 0.8, 0.0025, and 5000 respectively. As shown in Fig. 2, the value of λ starts at 0 and monotonically increases to the maximum of 0.8 at the 5000-th training step. This means that at the beginning of the training, the triplet term $\mathcal{L}_{triplet}$ in the loss function (6) is ignored, and the model learns according to the classification loss $\mathcal{L}_{CE}$. Over the training course, the trade-off between $\mathcal{L}_{triplet}$ and $\mathcal{L}_{CE}$ gradually shifts towards the triplet term $\mathcal{L}_{triplet}$ finally reaching the ratio of 0.8/0.2 between triplet $\mathcal{L}_{triplet}$ and classification $\mathcal{L}_{CE}$ loss terms. We have tested different values of parameters with a varying between 0.5 and 1 with a step of 0.1, b between 1000 and 5000 with step of 1000, and k between 0.0025 and 0.01 with step of 0.0025. It was found that in all scenarios, the resulting models gave very similar outcomes; however, the convergence rates were different. The values of $a = 0.8$, $k = 0.0025$ led to the fastest convergence of the network training procedure; therefore this set of values was used in the experiments described here.

It is important to note that the proposed multi-objective loss (6) is applicable only to the closed-set scenarios with fully labeled data. Classification loss $\mathcal{L}_{CE}$ assumes that the number of unique words is fixed. This assumption significantly limits the application of the proposed multi-objective approach, especially for the QbE-STD task, where the number of unique words is typically unknown. To address this issue, we have compared the performance of the embedded model trained with the multi-objective loss (6) with a model trained solely with the triplet loss function(4). Note that when using the triplet loss (4) solely, only weak supervision is implemented during training making this approach applicable in an open-set word discrimination task.

D. NETWORK ARCHITECTURE

The proposed acoustic word embedding model is based on the Long-Short-Term Memory (LSTM) network [17] architecture. This choice was motivated by many previous successful applications of LSTM in various speech recognition tasks [25]–[27]. As illustrated in Fig. 3, the model consists of two stacked LSTM layers with 512 units each, followed

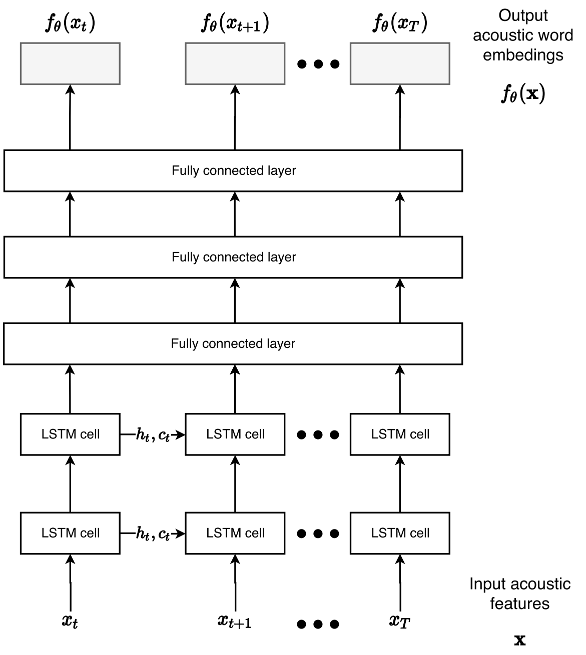


FIGURE 3. **LSTM-based acoustic word embedding model. It consists of recurrent LSTM cells and consecutive fully connected layers. The model produces an embedding $f_\theta(x_t)$ for each input segment x_t, and the resulting embedding $f_\theta(\mathbf{x})$ is a sequence of fixed-length embeddings $f_\theta(x_t)$.**

by three consecutive fully connected layers with 256 units each. The output of the last fully connected layer is the embedding. In Fig. 3, $\mathbf{x}$ (without a subscript) denotes the entire input sequence representing a word. It can be expressed as $\mathbf{x} = \{x_t\}_{t=1}^{T}$, where, x_t is a vector of frame-level acoustic features extracted from a given speech frame. For each input feature vector x_t, corresponding to a single frame, the model generates an embedding output $f_\theta(x_t)$. When Soft-DTW distance was used in $\mathcal{L}_{triplet}$, the consecutive embeddings were stacked to form the final embedding $f_\theta(\mathbf{x})$. Otherwise, only the embedding $f_\theta(x_T)$ of the last frame was used. The same embedding model structure was used in all experiments.

The acoustic word embeddings were passed to two different branches, according to (6).

The first branch consists of a single fully connected layer with 35 units, and a softmax applied to the output of this layer. The size of the output layer is equal to the number of classes (i.e. unique words). The output of the first classification branch gives the class probability distribution for the input sequence. It is used in the classification loss component $\mathcal{L}_{CE}$ of the multi-objective loss (6).

The second branch is consistent with the triplet network structure shown in Fig. 1. It calculates distances between two pairs of embeddings ($\mathbf{x}$, $\mathbf{x}^-$) and ($\mathbf{x}$, $\mathbf{x}^+$). These distances are used to compute the triplet loss $\mathcal{L}_{triplet}$ component of the multi-objective loss. It should be noted that for each input sample from the triplet $\{\mathbf{x}^-, \mathbf{x}, \mathbf{x}^+\}$, the same embedding model f_θ is used.

In cases when the model was trained with the triplet loss $\mathcal{L}_{triplet}$ solely, the first classification branch (with the $\mathcal{L}_{CE}$ objective component) was removed from the network structure. Batch-normalization and dropout were applied to the input of the network with a dropout rate $p = 0.2$. In all experiments, an Adam optimizer [28] with a learning rate of $lr = 0.0005$ was used during the training procedure. All models were implemented using PyTorch [29].

III. EXPERIMENTS

A. A SPEECH DATASET

The data used in all experiments was drawn from the Speech Commands dataset [30]. This dataset has been released under the Creative Commons BY 4.0 license [31], which means that no registration or specific permission to use it for research purposes was required. The dataset consists of 105,829 audio recordings of 35 different English words spoken by 2,618 English speakers representing a general population sample of speakers having different accents and speaking styles. Given that the word recognition was not the primary goal of this study, non-keyword or silent samples were not included in the selected data. The wide range of different speakers given in the Speech Commands dataset increased the chances of high model-generalization independent of individual speakers, genders, accents, and speaking styles. Since the recordings were captured using a laptop computer or a mobile phone, a moderate level of background noise was present. No processing was applied to remove the noise, thus creating an additional requirement for the model to be noise-robust. Each sample in the dataset was encoded as linear 16-bit single-channel PCM values, at 16 kHz sampling rate (i.e. 8 kHz signal bandwidth) and stored in wave audio-format.

In contrast to other commonly used datasets such as Switchboard [32], ZeroSpeech [33], and Spoken Web Search (SWS) 2013 [34] the Speech Commands dataset has a large number of samples representing a relatively small number of 35 unique words. Switchboard, for example, has a vocabulary of 1061 unique words [8]. This feature makes Speech Commands considerably easy yet suitable dataset for a word discrimination task.

B. TRAINING SCENARIOS

The entire speech dataset was split into 80% of training data, and 20% for evaluation. The training was performed on speech features extracted on a frame-by-frame basis. Each speech sample was split into a time sequence of frames with the frame-length of 20 ms and 10ms overlap between subsequent frames. For each frame, an acoustic feature vector x_t containing mel frequency cepstral coefficients (*MFCCs*) was calculated.

The following three word discrimination scenarios were considered:

- Strong supervision scenario - the network was trained to recognize words (i.e. the word labels were known). Models were trained using a multi-objective loss given in (6), which includes both the classification loss $\mathcal{L}_{CE}$ and the triplet loss $\mathcal{L}_{triplet}$. This approach had several limitations. Models trained with strong supervision do not guarantee good generalization of word embeddings beyond the training vocabulary. In addition, training under strong supervision requires a lot of examples per class (word). It limits the applicability of strong supervision in a low-resource scenario, when each word is represented by a small number of samples.

- Weak supervision scenario - the network was trained to recognize if a pair of words represents the same or different words (i.e. each pair of words was labeled as "the same" or "different"). Having no information about the individual word labels, doesn't allow using the classification loss $\mathcal{L}_{CE}$ for training. Therefore, the network was trained solely using the triplet loss $\mathcal{L}_{triplet}$. Since a given set of words can be used to generate a larger set of pairs labeled as "the same" or "different", this approach allows increasing the training data size without increasing the number of speech samples. It increases the applicability of weak supervision in a low-resource scenario when each word is represented by a small number of samples. It is typically the case in the QbE-STD with low-resource databases such as ZeroSpeech [33], Spoken Web Search (SWS) 2013 [34] or Query by Example Search on Speech Task (QUESST) 2014 [35].
- No supervision scenario - in this case, no embeddings were generated. The word discrimination was performed directly on MFCC vectors.

C. VALIDATION PROCEDURE AND METRICS

For each pair of words from the evaluation data set, the word discrimination system decided whether the pair contains the same or different words. The system response labels and the actual "ground-truth" labels were used to estimate the average precision (AP) as an overall system performance measure. To determine the AP value, for each pair of words from the evaluation set, the distance between acoustic embeddings of both words was calculated, and a threshold was applied to determine if the pair represents the same or different words. By sweeping the threshold value, a precision-recall curve was obtained from which the AP was estimated.

Additionally, we have compared distances between word embeddings within positive (both samples belong to the same word) and negative (different words) pairs on the batch of 2000 triplets. For each model 4000 distances were calculated (2000 for positive pairs and 2000 for negative pairs) using three different distance measures Cosine, Euclidean, and Soft-DTW. Distributions of distances between embeddings for positive and negative pairs were plotted and compared by calculating Kullback-Leibler (KL) distribution divergence measure. For a well-performing model, these two distributions should be far apart, with the distribution of positive pairs having lower distance values than the negative pairs. Accordingly, the higher the KL divergence between distributions, the better the discrimination between them.

Finally, to enable a qualitative analysis of the clustering outcomes, the learned embedding space was visualized using t-SNE [36] tool, which projected the embeddings onto the two-dimensional plane.

D. EVALUATED AWEs

As explained in Section III-B, three training scenarios were considered; one with strong, one with weak and one with no supervision. In the strong supervision scenario, word labels were provided during the training process, and the models were trained with the multi-objective loss function defined in (6). The following models were trained in this scenario:

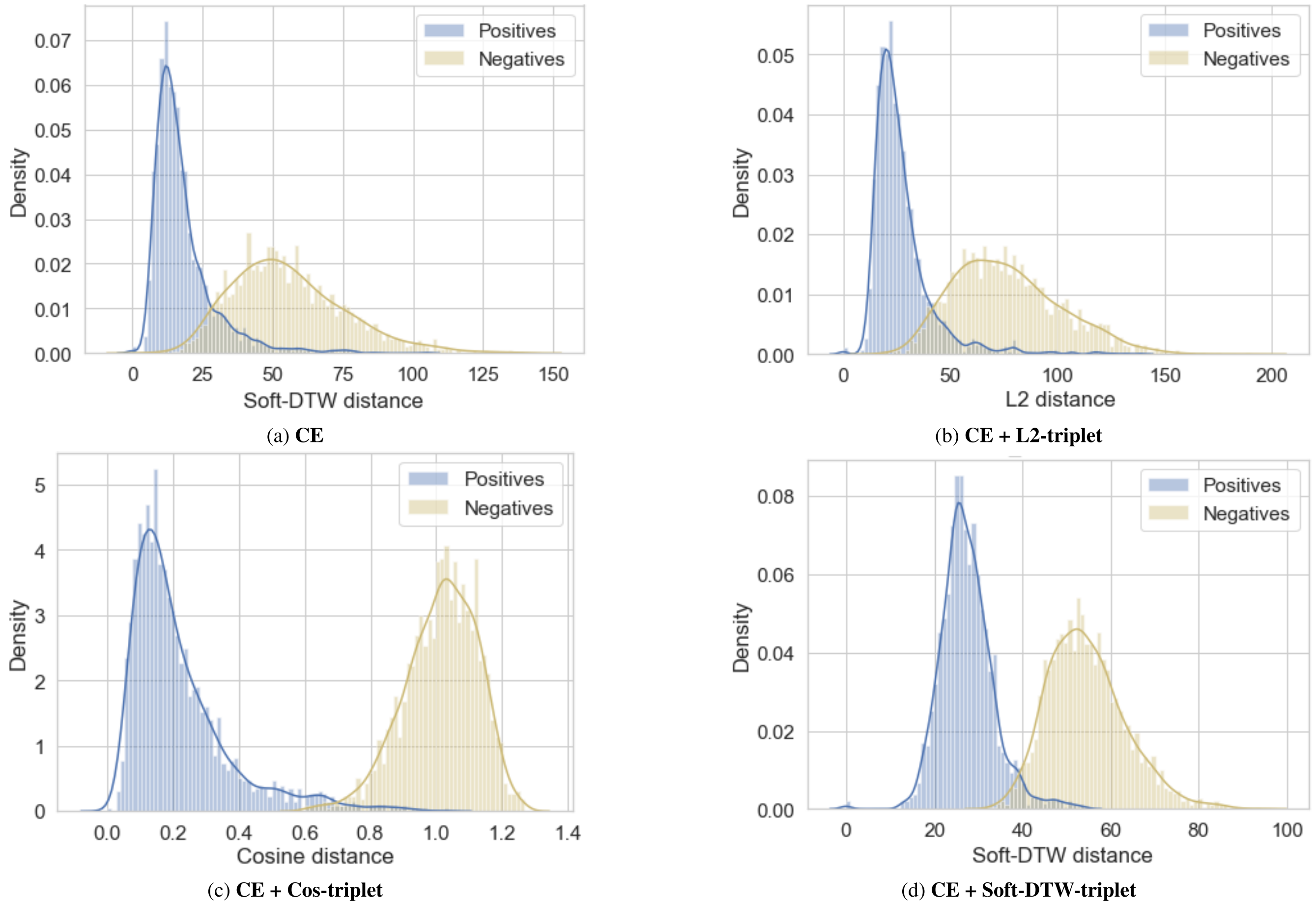
- **CE** - these AWEs were obtained via training the neural network discussed in Section II-D using the cross-entropy loss $\mathcal{L}_{CE}$ only.
- **CE + L2-triplet** - AWEs were learned using the loss function $\mathcal{L}$ similar to the one in (6). The only difference is that triplet loss $\mathcal{L}_{triplet}$ is computed with the L2 distance instead of the proposed Soft-DTW distance.
- **CE + Cos-triplet** - AWEs were learned using the loss function $\mathcal{L}$, where triplet loss $\mathcal{L}_{triplet}$ was computed with the cosine distance. This version of $\mathcal{L}_{triplet}$ was used for training the best performing model in [10].
- **CE + Soft-DTW-triplet**(our method) - AWEs were learned according to the proposed method discussed in Section II-C. In contrast to **CE + L2-triplet** and **CE + Cos-triplet**, the triplet loss $\mathcal{L}_{triplet}$ here is computed with the Soft-DTW distance $dtw_{\gamma}(\mathbf{x}, \mathbf{y})$, given in (1).

The weak supervision scenario assumes that word labels are unknown, and only information about whether the word pairs are the same or different was given. In this scenario, we evaluate models trained with the triplet loss (4) only. The following models were trained in this scenario:

- **L2-triplet** - AWEs were learned using the triplet loss $\mathcal{L}_{triplet}$,computed with the L2 distance.
- **Cos-triplet** - this model was adopted from [10], where authors call it **Siamese LSTM**. According to [10], **Siamese LSTM** outperforms all previous results on the word discrimination task and thus we compare the performance of our model against **Siamese LSTM**. To keep consistency within this work, we refer to this model as **Cos-triplet**. The major difference between learning objectives of **Siamese LSTM** and the proposed **DTW-triplet** is that the loss function uses Cosine distance rather than Soft-DTW when computing the distance between AWEs.
- **Soft-DTW-triplet** (our method) - AWEs were learned according to the proposed method discussed in the subsection II-C with the triplet loss $\mathcal{L}_{triplet}$,computed with the Soft-DTW distance $dtw_{\gamma}(\mathbf{x}, \mathbf{y})$ (1).

In the no supervision scenario, no embeddings were generated. The word discrimination was performed directly on MFCC vectors using the triplet loss $\mathcal{L}_{triplet}$ only.

Regarding the computational complexity of the training process, all models within individual training scenarios had



(a) **CE**

(b) **CE + L2-triplet**

(c) **CE + Cos-triplet**

(d) **CE + Soft-DTW-triplet**

FIGURE 4. Distribution of distances between AWEs within positive and negative pairs. Positive pair means that both samples are drawn from the same class (same word). Negative pair assumes that samples from this pair are drawn from different classes (different words). AWEs are learned with strong supervision.

identical network structures and an equal number of trainable parameters defined by the number of layers and number of units per each layer. What made them computationally different was the computational complexity of the loss function and the complexity of computing its gradient. The computational complexity of the Soft-DTW algorithm was $\mathcal{O}(nm)$ where n and m were the respective lengths of the analyzed sequences. It was a contrast to the constant complexity of the Cosine and Euclidean distances. In this study, the analyzed sequences had an average length of 100 samples, and a noticeably longer time was needed to train **Soft-DTW** models. Namely, for the **Soft-DTW-triplet** and **CE + Soft-DTW-triplet** models, the training time was increased by a factor of 1.5 compared to other models.

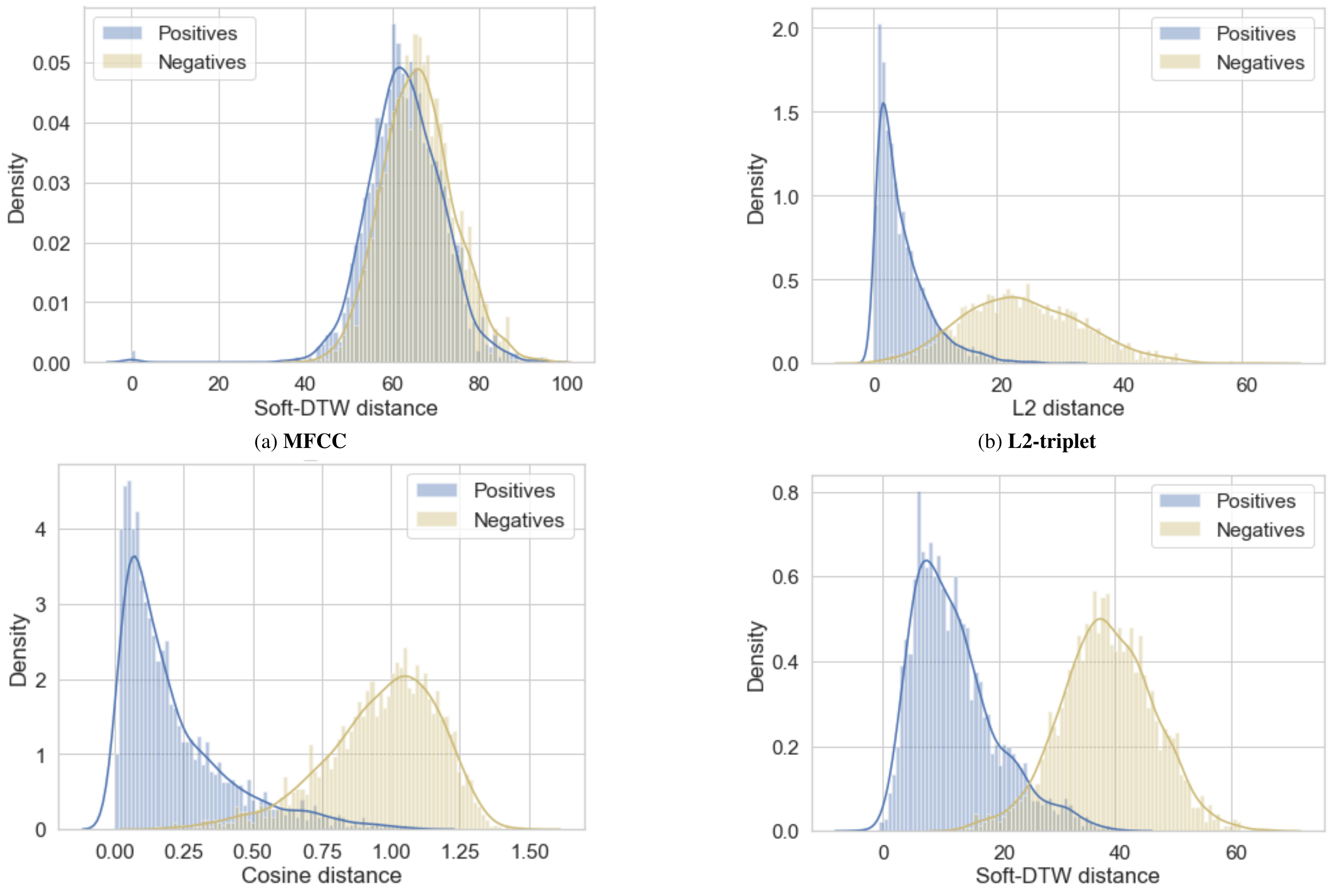
IV. RESULTS

A. DISTANCE DISTRIBUTIONS

Distances between different AWEs were calculated for a batch of 2000 triplets with corresponding 2000 similar word pairs and 2000 non-similar word pairs. For all models, a fixed set was used to avoid randomness when evaluating the performance of the resulting AWEs.

In order to validate the significance of obtained results, this procedure of distance calculation was performed on 20 randomly sampled batches of 2000 triplets each, followed by conducting the one-way analysis of variance (ANOVA). The null-hypothesis claims that there is no statistically significant difference between the means of these 20 groups. The resulting $p - value$ of ANOVA test was equal to 0.62 which didn't provide enough evidence to reject the null hypothesis.

Figures 4 and 5 depict distributions of distances between embeddings for different embedding models. Fig. 4 illustrates the distances achieved with four different models trained within the strong supervision scenario. Fig. 5 shows the distributions achieved by three different models trained within the weak supervision scenario, and the distribution for MFCCs obtained in an unsupervised way. Each sub-figure in Figures 4 and 5 corresponds to one of the different word embedding models. Within each sub-figure, one can see two distributions: 'Positives' and 'Negatives.' 'Positives' (marked in blue) correspond to the distances computed between samples of the same class (i.e. the same words), whereas 'Negatives' (marked in yellow) refer to the distances between samples representing different classes (i.e. different words). Table 1,



(a) **MFCC**

(b) **L2-triplet**

(c) **Cos-triplet**

(d) **Soft-DTW-triplet**

FIGURE 5. **Distribution of distances between AWEs within positive and negative pairs. Positive pair means that both samples are drawn from the same class (same word). Negative pair assumes that samples from this pair are drawn from different classes (different words). AWEs are learned with weak supervision.**

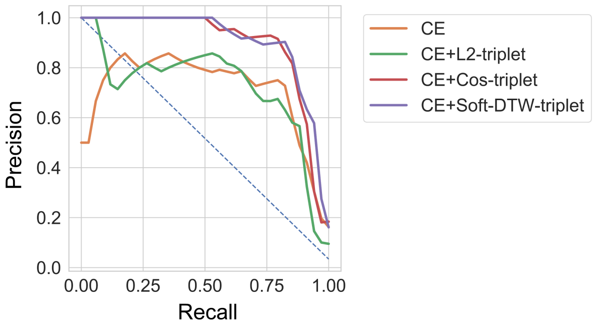


FIGURE 6. **Precision vs. recall for different AWEs in the strong supervision scenario.**

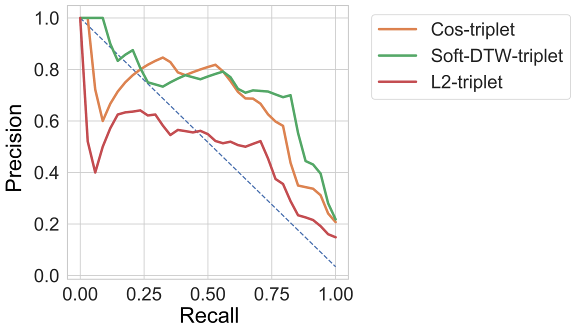


FIGURE 7. **Precision vs. recall for different AWEs in the weak supervision scenario.**

on the other hand, shows values of the KL divergence measure used to quantify distances between distribution of ‘Positives’ and ‘Negatives’ for different embedding models within all supervision scenarios.

Looking at the outcomes of the strong supervision, it can be seen in Fig.4, that **CE**, and **CE+L2-triplet** AWEs models have a similar discrepancy between ‘Positives’ and ‘Negatives’, with the largest overlap between these two distributions compared to other models trained in this scenario. This is confirmed in quantitative terms with **CE** and **CE+L2-triplet** achieving a lower KL divergence between ‘Positives’ and ‘Negatives’, equal to 4.266 and 5.279 respectively, compared to other models trained in this scenario. **CE+Cos-triplet** achieves the highest KL divergence of 7.723 with a visible distinction between ‘Positives’ and ‘Negatives’ distributions. **CE+Soft-DTW-triplet** has a slightly lower KL divergence of 7.388, which is still comparable with the best result achieved by **CE+Cos-triplet**.

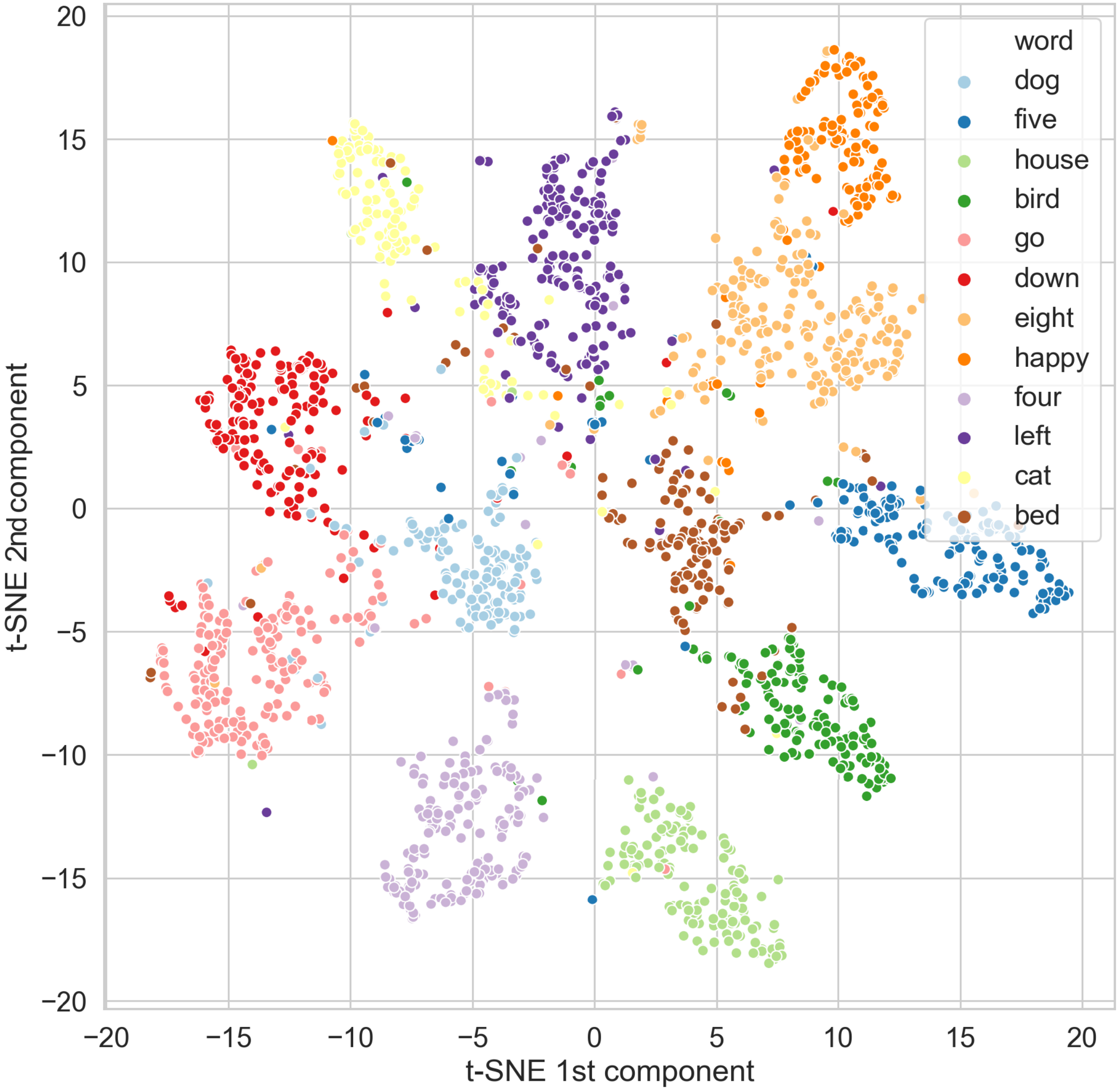


FIGURE 8. Visualization of AWEs obtained with DTW-triplet.

Considering the weak supervision scenario, Fig.5 shows that **L2-triplet** demonstrates less distinction between 'Positives' and 'Negatives' with KL divergence of 3.050 when compared against its counterpart **CE+L2-triplet** trained with strong supervision. We observed the same trend of the lower performance for other weakly supervised models relative to their fully-supervised analogs. **Cos-triplet**, for example, achieves KL divergence equal to 6.114 as opposed to **CE+Cos-triplet** KL divergence of 7.723. Our proposed model **Soft-DTW-triplet** has KL divergence of 5.885 as opposed to **CE+Soft-DTW-triplet** with AP equal to 7.388. The best performing model **Cos-triplet**, which corresponds to **Siamese LSTM** from [10], outperforms our model showing the best discrepancy between distributions and achieves KL divergence equal to 6.114.

Finally, in the case of no supervision, Fig.5 shows that there is almost no difference between two distributions for 'Positives' and 'Negatives' when MFCCs were used as AWEs. Table 1 supports this hypothesis by showing KL divergence of 1.1087 for MFCCs.

B. PRECISION VS. RECALL

Precision vs. recall curves for AWEs learned with weak and strong supervision scenarios are presented in Figures 6 and 7, respectively. In addition,Table 1 provides the AP values of word discrimination approaches based on embedding models used in all training scenarios. The AP values were calculated as the area under the precision-recall curves, characterizing the average performance of the word discrimination system across all operating points. Figures 6 and 7 show the achieved precision-recall contours in strong and weak supervision scenarios, respectively. It can be observed in Fig.6 that in the strong supervision case, **CE+Cos-triplet** and the proposed **CE+Soft-DTW-triplet** significantly outperformed

TABLE 1. Comparison of embedding models trained under different supervision scenarios.

Supervision	AWE	KL-divergence	AP
Strong	**CE**	4.266	0.720
	CE + L2-triplet	5.279	0.728
	CE + Cos-triplet	7.723	0.897
	CE + Soft-DTW-triplet	7.388	0.909
Weak	**L2-triplet**	3.050	0.497
	Cos-triplet	6.114	0.682
	Soft-DTW-triplet	5.885	0.737
No	**MFCC**	1.108	0.048

the other models **CE** and **CE+L2-triplet**. The proposed model **CE+Soft-DTW-triplet** slightly outperformed **CE+Cos-triplet**, achieving AP of 0.909 while **CE+Cos-triplet** shows AP of 0.897. It is quite noticeable that all models in this scenario produced substantially higher AP compared with results reported in [37] and [8]. This was expected due to the relatively "easy" word discrimination dataset used in this work, as discussed in Section III-A.

Results presented in Fig.7, on the other hand, are more consistent with previous word discrimination studies. **Cos-triplet**, adopted from [10], achieved AP of 0.682, which is almost identical to AP of 0.671 reported in [10] for the same model. The best performing model in the weak supervision scenario was the proposed **Soft-DTW-triplet**. It achieved AP of 0.737 outperforming the second-best **Cos-triplet** by approximately 0.05. The **L2-triplet** model showed the lowest AP equal to 0.497 among other triplet networks. This result is consistent with findings reported in [37], and shows poor performance of L2-based models.

C. EMBEDDING SPACE VISUALIZATION

To visualize the learned embedding space of the best performing **CE + Soft-DTW-triplet** model, we have created a two-dimensional plot showing embeddings of 12 different words from the dataset vocabulary using the t-SNE data visualization tool. The resulting graph is presented in Fig. 8. It can be observed in this example that the clusters for different words are well separated, showing relatively small intra and large inter-class distances. The majority of the samples were correctly clustered within the same word class, with only a small number of outliers. Moreover, the relative distance between clusters, in most cases, indicates how phonetically similar the corresponding words are. For example, words with phonetically close phonemes /æ/, /e/ reside mostly in the upper part of embedding space, whereas words with /o/ phoneme can be found in the lower left-hand corner. Although these results provide interesting insights into the distribution of the words within the embedding space, further studies are needed to investigate how phonetically correlated words are distributed in that space. It would also be interesting to analyze the effect of the model's complexity on the embedding space and its representation capability.

V. CONCLUSION

A novel method of learning acoustic word embeddings via triplet network with Soft-DTW as a distance measure between the embeddings was proposed and evaluated. It was demonstrated that the proposed model is competitive with recent deep learning benchmarks for the word discrimination task. Experimental validation results have shown that the **Soft-DTW-triplet** trained with weak supervision achieves AP equal to 0.737 on Speech Commands dataset as opposed to **Siamese LSTM** with AP of 0.682 adopted from [10]. The proposed approach is principally different from other methods using AWEs [8], [10]. Namely, it is capable of learning length-varying embeddings and determines the similarity between word embeddings with soft-DTW. To match applications of embeddings during inference and training, a triplet loss with soft-DTW was proposed as distance measure between samples. It was demonstrated that embeddings obtained via the proposed method improve word discrimination outcomes. The proposed learning objective ensures that embeddings are learned in a way that similar words have a small DTW distance and remain close to each other in the embedding space while non-similar words occur far apart from each other and have a high DTW distance. This hypothesis was validated by computing distances between similar and non-similar words and then plotting the resulted distributions of distances. A good discriminatory ability was observed meaning that similar and non-similar word pairs were far apart from each other, with a small overlap between the distributions tails.

In future studies, the proposed in this study embedding model will be evaluated on more challenging datasets such as ZeroSpeech [33], Spoken Web Search (SWS) 2013 [34], or Query by Example Search on Speech Task (QUESST) 2014 [35]. These datasets are considered as most appropriate for QbE-STD tasks. Evaluation based on these datasets will give a clear understanding of the performance of our model in more realistic QbE-STD scenarios. In addition, we will study the applicability of the proposed method in tasks of incomplete sequence matching and incomplete spoken term detection. Incomplete sequence matching is essential for systems where the decision has to be made based on incomplete temporal information. It would be interesting to see if the proposed method could be adapted to such a challenging task.

REFERENCES

[1] J. Donald Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series," in *Proc. 3rd Int. Conf. Knowl. Discovery Data Mining*, 1994, pp. 359–370.

[2] J. Proenga, A. Veiga, and F. Perdigao, "Query by example search with segmented dynamic time warping for non-exact spoken queries," in *Proc. 23rd Eur. Signal Process. Conf. (EUSIPCO)*, Aug. 2015, pp. 1661–1665.

[3] J. Aach and G. M. Church, "Aligning gene expression time series with time warping algorithms," *Bioinformatics*, vol. 17, no. 6, pp. 495–508, Jun. 2001.

[4] P. Tormene, T. Giorgino, S. Quaglini, and M. Stefanelli, "Matching incomplete time series with dynamic time warping: An algorithm and an application to post-stroke rehabilitation," *Artif. Intell. Med.*, vol. 45, no. 1, pp. 11–34, Jan. 2009.

[5] T. J. Hazen, W. Shen, and C. White, "Query-by-example spoken term detection using phonetic posteriorgram templates," in *Proc. IEEE Workshop Autom. Speech Recognit. Understand.*, Dec. 2009, pp. 421–426.

[6] Y. Zhang and J. R. Glass, "Unsupervised spoken keyword spotting via segmental DTW on Gaussian posteriorgrams," in *Proc. IEEE Workshop Autom. Speech Recognit. Understand.*, Dec. 2009, pp. 398–403.

[7] G. Aradilla, H. Bourlard, and M. Magimai-Doss, "Posterior features applied to speech recognition tasks with user-defined vocabulary," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, Apr. 2009, pp. 3809–3812.

[8] H. Kamper, W. Wang, and K. Livescu, "Deep convolutional acoustic word embeddings using word-pair side information," 2015, *arXiv:1510.01032*. [Online]. Available: http://arxiv.org/abs/1510.01032

[9] Y.-A. Chung and J. Glass, "Learning word embeddings from speech," 2017, *arXiv:1711.01515*. [Online]. Available: http://arxiv.org/abs/1711.01515

[10] S. Settle and K. Livescu, "Discriminative acoustic word embeddings: Tecurrent neural network-based approaches," in *Proc. IEEE Spoken Lang. Technol. Workshop (SLT)*, Dec. 2016, pp. 503–510.

[11] D. Yu and M. Seltzer, "Improved bottleneck features using pretrained deep neural networks," in *Proc. INTERSPEECH*, Jan. 2011, pp. 237–240.

[12] S. Settle, K. Levin, H. Kamper, and K. Livescu, "Query-by-example search with discriminative neural acoustic word embeddings," 2017, *arXiv:1706.03818*. [Online]. Available: http://arxiv.org/abs/1706.03818

[13] Y.-H. Wang, H.-Y. Lee, and L.-S. Lee, "Segmental audio Word2Vec: Representing utterances as sequences of vectors with applications in spoken term detection," 2018, *arXiv:1808.02228*. [Online]. Available: http://arxiv.org/abs/1808.02228

[14] Y.-A. Chung and J. Glass, "Speech2Vec: A sequence-to-sequence framework for learning word embeddings from speech," 2018, *arXiv:1803.08976*. [Online]. Available: https://arxiv.org/abs/1803.08976

[15] Z. Yang and J. Hirschberg, "Linguistically-informed training of acoustic word embeddings for low-resource languages," in *Proc. Interspeech*, Sep. 2019, pp. 2678–2682.

[16] W. He, W. Wang, and K. Livescu, "Multi-view recurrent neural acoustic word embeddings," 2016, *arXiv:1611.04496*. [Online]. Available: http://arxiv.org/abs/1611.04496

[17] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[18] I. Szoke, M. Skacel, L. Burget, and J. Cernocky, "Copingwith channel mismatch in Query-by-Example–But QUESST 2014," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2015, pp. 5838–5842.

[19] M. Cuturi and M. Blondel, "Soft-DTW: A differentiable loss function for time-series," 2017, *arXiv:1703.01541*. [Online]. Available: http://arxiv.org/abs/1703.01541

[20] K. Hua. (2019). *PyTorch-SoftDTW*. Accessed: Oct. 15, 2019. [Online]. Available: https://github.com/Sleepwalking/pytorch-softdtw

[21] E. Hoffer and N. Ailon, "Deep metric learning using triplet network," in *Similarity-Based Pattern Recognition*, A. Feragen, M. Pelillo, and M. Loog, Eds. Cham, Switzerland: Springer, 2015, pp. 84–92.

[22] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," 2015, *arXiv:1503.03832*. [Online]. Available: http://arxiv.org/abs/1503.03832

[23] C.-Y. Wu, R. Manmatha, A. J. Smola, and P. Krähenbähl, "Sampling matters in deep embedding learning," 2017, *arXiv:1706.07567*. [Online]. Available: http://arxiv.org/abs/1706.07567

[24] D. Ram, L. Miculicich, and H. Bourlard, "Multilingual bottleneck features for query by example spoken term detection," in *Proc. IEEE Autom. Speech Recognit. Understand. Workshop (ASRU)*, Dec. 2019, pp. 621–628.

[25] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014, *arXiv:1412.3555*. [Online]. Available: http://arxiv.org/abs/1412.3555

[26] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2013, pp. 6645–6649.

[27] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition," 2014, *arXiv:1402.1128*. [Online]. Available: http://arxiv.org/abs/1402.1128

[28] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980

[29] A. Paszke *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[30] P. Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," 2018, *arXiv:1804.03209*. [Online]. Available: http://arxiv.org/abs/1804.03209

[31] 2019. *Creative Commons*. Accessed: Oct. 15, 2019. [Online]. Available: https://creativecommons.org/licenses/by/4.0/

[32] J. J. Godfrey, E. C. Holliman, and J. McDaniel, "SWITCHBOARD: Telephone speech corpus for research and development," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Mar. 1992, pp. 517–520.

[33] E. Dunbar, X. N. Cao, J. Benjumea, J. Karadayi, M. Bernard, L. Besacier, X. Anguera, and E. Dupoux, "The zero resource speech challenge 2017," in *Proc. IEEE Autom. Speech Recognit. Understand. Workshop (ASRU)*, Dec. 2017, pp. 323–330.

[34] X. A. Miró, L. J. Rodriguez-Fuentes, I. Szöke, A. Buzo, F. Metze, and M. P. Nagarikano, "Query-by-example spoken term detection on multilingual unconstrained speech," in *Proc. INTERSPEECH*, 2014, pp. 1–8.

[35] X. Anguera, L.-J. Rodriguez-Fuentes, A. Buzo, F. Metze, I. Szoke, and M. Penagarikano, "QUESST2014: Evaluating query-by-example speech search in a zero-resource setting with real-life queries," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2015, pp. 5833–5837.

[36] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, Nov. 2008.

[37] A. Michael Carlin, S. Thomas, A. Jansen, and H. Hermansky, "Rapid evaluation of speech representations for spoken term discovery," in *Proc. INTERSPEECH*, 2011, pp. 821–824.

DENIS SHITOV received the B.S. and M.S. degrees in electronic engineering from ITMO University, Russia, in 2013 and 2015, respectively. He is currently pursuing the Ph.D. degree in electrical and electronic engineering with RMIT University, Australia. His research interests include machine learning and image and speech processing.

ELENA PIROGOVA received the B.Eng. degree (Hons.) in chemical engineering from National Technical University, Ukraine, in 1991, and the Ph.D. degree in biomedical engineering from Monash University, Australia, in 2002. She is currently a Professor of biomedical engineering with the School of Engineering. RMIT University, Australia. Her research interests include biomedical electronics and instrumentation, bioelectromagnetics, and protein modeling.

TADEUSZ A. WYSOCKI (Senior Member, IEEE) received the M.Eng.Sc. degree (Hons.) in telecommunications from the Academy of Technology and Agriculture, Bydgoszcz, Poland, in 1981, and the Ph.D. degree (summa cum laude) and the D.Sc. degree (Habilitation) in telecommunications engineering from the Warsaw University of Technology, in 1984 and 1990, respectively. In January 1992, he moved to Australia, where he worked at different universities and research centers. He currently holds a Professorship with the Department of Electrical and Computer Engineering, University of Nebraska-Lincoln. He is also with UTP University, Bydgoszcz, Poland. He is the author or coauthor of seven books, more than 250 research publications, and nine patents. His research interests include molecular communications, modeling of biological processes at nano-scale, and signal processing for communication systems.

MARGARET LECH (Member, IEEE) received the M.S. degree in physics from Maria Curie-Sklodowska University, Poland, the M.S. degree in biomedical engineering from the Warsaw University of Technology, Poland, and the Ph.D. degree in electrical engineering from the University of Melbourne, Australia. She is currently a Professor of signal processing and artificial intelligence with the School of Engineering, RMIT University, Australia. Her research interests include psychoacoustic, speech and image processing, system modeling, and optimization.