DEEP CONVOLUTIONAL ACOUSTIC WORD EMBEDDINGS USING WORD-PAIR SIDE INFORMATION

Herman Kamper[1]*, Weiran Wang*[2]*, Karen Livescu*[2]

[1]CSTR and ILCC, School of Informatics, University of Edinburgh, UK
[2]Toyota Technological Institute at Chicago, USA

h.kamper@sms.ed.ac.uk, weiranwang@ttic.edu, klivescu@ttic.edu

ABSTRACT

Recent studies have been revisiting whole words as the basic modelling unit in speech recognition and query applications, instead of phonetic units. Such whole-word segmental systems rely on a function that maps a variable-length speech segment to a vector in a fixed-dimensional space; the resulting *acoustic word embeddings* need to allow for accurate discrimination between different word types, directly in the embedding space. We compare several old and new approaches in a word discrimination task. Our best approach uses side information in the form of known word pairs to train a Siamese convolutional neural network (CNN): a pair of tied networks that take two speech segments as input and produce their embeddings, trained with a hinge loss that separates same-word pairs and different-word pairs by some margin. A word classifier CNN performs similarly, but requires much stronger supervision. Both types of CNNs yield large improvements over the best previously published results on the word discrimination task.

***Index Terms*—** Acoustic word embeddings, segmental acoustic models, fixed-dimensional representations, query-by-example search.

1. INTRODUCTION

Most current speech processing systems rely on a deep architecture to classify speech frames into subword units (often phone states). This approach still relies on frame-level independence assumptions as well as a pronunciation lexicon for breaking up words into their subword constituents. As an alternative, some researchers [1–7] have started to reconsider using whole words as the basic modelling unit.

Some of the earliest speech recognition systems were based on template-based whole-word modelling [8]. This idea has been revisited in modern template-based automatic speech recognition (ASR) systems [1, 2], as well as modern speech indexing applications such as query-by-example search [9, 10]. These systems typically use dynamic time warping (DTW) to quantify the similarity of phone or word segments of variable length. Recent work has also considered frame-level embeddings which map acoustic features to a new frame-level representation that is tailored to word discrimination when combined with DTW [11–13]. DTW, however, has known inadequacies [14] and is quadratic-time in the duration of the segments.

Levin *et al.* [3] proposed a segmental approach where an arbitrary-length speech segment is embedded in a fixed-dimensional space such that segments of the same word type have similar embeddings. Segments can then be compared by simply calculating a distance in the embedding space, a linear time operation in the embedding dimensionality. Several approaches were developed in [3], and in [15] these were successfully applied in a query-by-example search system.

This research was supported by NSF grant IIS-1321015. The opinions expressed in this work are those of the authors and do not necessarily reflect the views of the funding agency. HK is funded by a Commonwealth Scholarship.

Bengio and Heigold [4] similarly used whole-word fixed-dimensional representations in a segmental ASR lattice rescoring system. Their acoustic embeddings are obtained from a convolutional neural network (CNN), trained with a combination of a word classification and a ranking loss. When combining the hypotheses of the baseline system with the embedding-based scores, ASR performance was improved. A similar approach was followed in [5], where long short-term memory (LSTM) networks were used to obtain whole-word embeddings for a query-by-example search task. Finally, Maas *et al.* [6] trained a regression CNN that reconstructs a semantic word embedding from acoustic speech input; these features were used in a segmental conditional random field ASR system.

In this paper we compare several CNN-based approaches to each other and to the best approach of Levin *et al.* [3], on a word discrimination task. This task has been used in several other studies [11,12,16] to assess the accuracy of acoustic embedding approaches without the need to train a complete recognition or search system. Building on ideas from earlier CNN-based approaches, we propose new networks that make use of weaker supervision in the form of known word pairs. The approach is based on *Siamese networks*: tied networks that take in pairs of input vectors and minimize or maximize a distance depending on whether a pair comes from the same or different classes [17]. We show that a Siamese CNN trained with with a hinge-like contrastive loss function outperforms the best approach of Levin *et al.* [3], and performs similarly to a word classifier CNN, despite the weaker form of supervision. By reducing the Siamese CNN embedding dimensionality with a post-processing linear discriminant analysis, we also obtain a more compact embedding that maintains best performance.

2. ACOUSTIC WORD EMBEDDING APPROACHES

For speech applications using fixed-dimensional representations of whole words, it is desirable to find a mapping such that word segments of the same type are close in the embedding space while those of different types are far from each other. Formally, we use the notation $Y = \mathbf{y}_{1:T}$ to denote a vector time series, where each $\mathbf{y}_t \in \mathbb{R}^b$ is a b-dimensional frame-level feature vector (e.g. MFCCs). An acoustic word embedding approach is a function $f(Y)$ that maps arbitrary-length time series Y into a fixed-dimensional space $\mathbb{R}^d$; if Y_1 and Y_2 are two word segments, the distance between the vectors $f(Y_1)$ and $f(Y_2)$ should indicate whether they are of the same word type or not.

Typical embedding approaches use a training set of known word segments $\mathcal{Y}_{\text{train}} = \{Y_i\}_{i=1}^{N_{\text{train}}}$ to learn f. Different degrees of supervision can be assumed, ranging from unsupervised, where the only knowledge of $\mathcal{Y}_{\text{train}}$ is that it contains unidentified word segments, to supervised, where the word label for each segment is known. Below we review previous work (Sections 2.1 and 2.2) and then present our own approaches which use weak supervision in the form of known word pairs (Section 2.3), and can also additionally use word labels to find lower-dimensional but still accurate embeddings (Section 2.4).

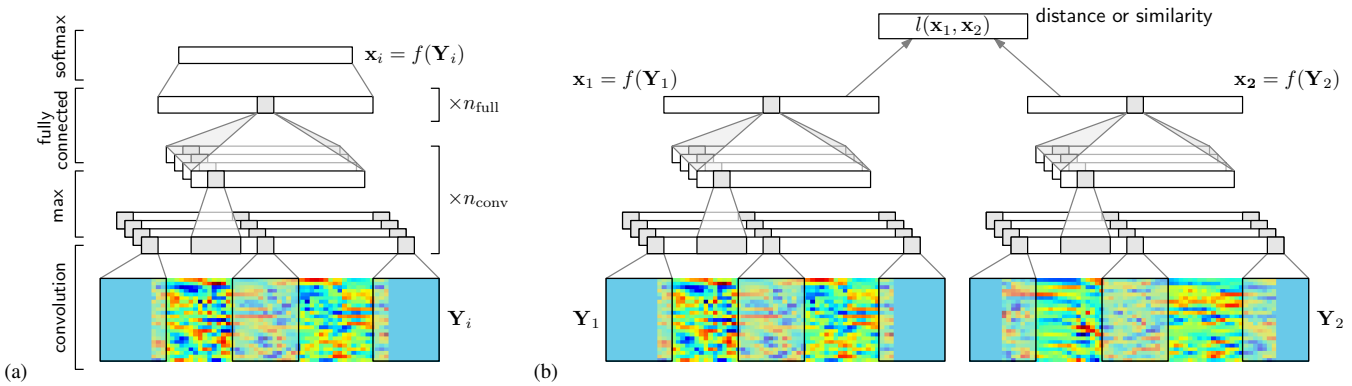

Fig. 1. (a) Word classification CNN and (b) word similarity Siamese CNN for obtaining acoustic word embeddings from padded speech input.

2.1. Reference vector methods

Several embedding approaches were proposed and compared in Levin *et al.* [3] based on the idea of using a *reference vector* to construct the mapping f. For a target speech segment, a reference vector consists of the DTW alignment cost to every exemplar in a reference set $\mathcal{Y}_{\text{ref}} \subseteq \mathcal{Y}_{\text{train}}$. Applying dimensionality reduction to the reference vector yields the desired embedding in $\mathbb{R}^d$. The intuition is that the content of a speech segment should be characterized well through its similarity to other segments. Such embeddings have subsequently been used for keyword search [15] and unsupervised term discovery [18]. One drawback is the need to compute a large number of DTW alignments. Several dimensionality reduction approaches were considered in [3], and in Section 3.3 we compare to their best overall approach which uses a combination of Laplacian eigenmaps (a non-linear graph embedding approach) and linear discriminant analysis.

2.2. Word classification CNN

For the whole-word speech recognition system in [4], Bengio and Heigold proposed that, when word labels $\mathcal{W}_{\text{train}} = \{w_i\}_{i=1}^{N_{\text{train}}}$ are available for the training segments $\mathcal{Y}_{\text{train}}$, a supervised neural network can simply be trained to predict the word class (type) given the speech as input. The softmax prediction layer of such a neural network then gives a fixed-dimensional representation in $\mathbb{R}^d$, where d here is the number of distinct word types (the vocabulary size). During testing, some inputs may correspond to unseen words, but even in these cases the softmax layer gives a fixed-dimensional distributional representation of the input in terms of seen word types.

A standard feed-forward neural network classifier, however, requires fixed-dimensional input. A simple solution was used in [4]: All word segments are padded to the same length, given by the maximum duration of a word segment in $\mathcal{Y}_{\text{train}}$. Instead of using fully-connected layers, convolutional and pooling layers are used to alleviate the effect of the padding. A convolutional neural network (CNN) such as this is shown in Figure 1(a). Our implementation uses mean-normalized MFCCs which are zero-padded to n_{pad} frames. One-dimensional convolution is performed only over time, covering a number of frames and all features, and is followed by max-pooling. These layers can be repeated. A number of fully-connected layers is used next, which feeds into the final softmax layer. Formally, the whole CNN defines a mapping function $f : \mathbf{Y}_i \in \mathbb{R}^{b \times n_{\text{pad}}} \rightarrow \mathbf{x}_i \in \mathbb{R}^d$, that takes input $\mathbf{Y}_i$, obtained by padding the variable-length b-dimensional vector time series Y_i, and produces the acoustic embedding $\mathbf{x}_i$.

Instead of using the representation from the word classification CNN directly, Bengio and Heigold [4] used a paired network with a ranking loss to map acoustic word embeddings into a common space with orthography-based word embeddings obtained by also mapping the word labels $\mathcal{W}_{\text{train}}$ into a lower-dimensional space. This was done to make it possible to use the classifier outputs in a particular lattice rescoring architecture, which requires scores for lattice arcs. The evaluation framework we use (Section 3.1) is designed to be decoupled from a recognition architecture, and we can therefore use the distributional representation from the classifier CNN directly. An investigation of whether embeddings of word labels can be additionally used to improve acoustic word embeddings is left for future work.

2.3. Word similarity Siamese CNNs

If the labels $\mathcal{W}_{\text{train}}$ for the training set $\mathcal{Y}_{\text{train}}$ are not known, a weaker form of supervision that has also been used [11–13, 19] is the knowledge that pairs of word segments in $\mathcal{Y}_{\text{train}}$ share the same unknown word type: $\mathcal{S}_{\text{train}} = \{(m, n) : (Y_m, Y_n) \text{ are of the same type}\}$. This type of side information is appealing since it is often easier to obtain in low-resource settings, for example by using an unsupervised term discovery system [20, 21] to find unidentified matching word pairs.

Such paired supervision has been used for several problems and domains, including phonetic discovery [13, 22], semantic word embeddings [23–25] and vision applications [26]. Many of these studies use *Siamese networks*, a term used since the early 1990s to describe a pair of networks with tied parameters which is trained to optimize a distance function between representations of two data instances [17]. To train these networks it is sometimes assumed that pairs not in $\mathcal{S}_{\text{train}}$ belong to different types; we also make this assumption here.

Figure 1(b) illustrates how we apply this idea to obtain acoustic word embeddings. The two sides of our Siamese network take padded inputs $\mathbf{Y}_1$ and $\mathbf{Y}_2$. For the two sides we use CNNs similar to that of the word classification CNN. But instead of terminating in a softmax layer, the final fully connected layer on each side gives the desired acoustic embedding. In initial experiments on development data, we considered several loss functions, and here we focus only on the most successful ones. We found that losses based on cosine similarity outperformed Euclidean-based losses, and in particular the coscos^2 loss from [27] gave the best performance of the losses in [17, 27]:

$$l_{\text{cos cos}^2}(\mathbf{x}_1, \mathbf{x}_2) = \begin{cases} \frac{1-\cos(\mathbf{x}_1, \mathbf{x}_2)}{2} & \text{if same} \\ \cos^2(\mathbf{x}_1, \mathbf{x}_2) & \text{if different} \end{cases} \quad (1)$$

This loss pushes the angle between embeddings of the same type to be zero, while embeddings of different types are pushed to be orthogonal.

In discrimination tasks, the decision of whether two data instances are of the same type is not based on their *absolute* distance, but rather their *relative* distance compared to other pairs. This motivates a margin-based (hinge) loss, similar to that of [24, 25]:

$$l_{\text{cos hinge}} = \max\{0, m + d_{\cos}(\mathbf{x}_1, \mathbf{x}_2) - d_{\cos}(\mathbf{x}_1, \mathbf{x}_3)\} \quad (2)$$

where $d_{\cos}(\mathbf{x}_1, \mathbf{x}_2) = \frac{1-\cos(\mathbf{x}_1,\mathbf{x}_2)}{2}$ is the cosine distance between $\mathbf{x}_1$ and $\mathbf{x}_2$, and m is a margin parameter. Here, $\mathbf{x}_1$ and $\mathbf{x}_2$ are always of the same type while $\mathbf{x}_1$ and $\mathbf{x}_3$ are of different types. This loss is therefore at a minimum when all embeddings $\mathbf{x}_1$ and $\mathbf{x}_2$ of the same type are more similar by a margin m than embeddings $\mathbf{x}_1$ and $\mathbf{x}_3$ of different types. The margin also gives some leeway to the model.

Although Siamese networks have been used widely, to our knowledge this is the first work which uses Siamese networks (in particular Siamese CNNs) to obtain acoustic word embeddings from speech.

2.4. Controlling embedding dimensionality

We aim to learn word embeddings that are both discriminative and compact (low-dimensional). The desired dimensionality may be guided by both computational and data constraints, and we may wish to be able to adjust it. For word classification networks (Section 2.2), the output dimensionality is given by the vocabulary size. In our experiments (next section), we explore adjusting the dimensionality by inserting an additional linear bottleneck layer before the final softmax, with the number of units corresponding to the desired final dimensionality. In Siamese networks (Section 2.3) the final dimensionality can be directly tuned. If we have access to word labels $\mathcal{W}_{\text{train}}$ in addition to word pairs $\mathcal{S}_{\text{train}}$, we can also perform additional dimensionlity reduction on the Siamese CNN outputs using a supervised technique; in our experiments we use linear discriminant analysis (LDA).

3. EXPERIMENTS

3.1. Evaluation and experimental setup

Ultimately we would like to evaluate the different acoustic embedding approaches for downstream speech recognition and search tasks. However, we do not want to be tied to a specific recognition architecture, and we would like to quickly compare many embedding approaches. We therefore use a word discrimination task developed for this purpose [16]; in the *same-different task*, we are given a pair of acoustic segments, each corresponding to a word, and we must decide whether the segments are examples of the same or different words.

This task can be approached in a number of ways, but typically it is done either via a DTW score between segments (when using frame-by-frame embeddings), or via a Euclidean or cosine distance between vectors (when embedding complete segments). In our evaluation, after training a model on $\mathcal{Y}_{\text{train}}$, the acoustic word embeddings of a disjoint test set $\mathcal{Y}_{\text{test}}$ are computed. For every word pair in this set, the cosine distance[1] is calculated between their embeddings. Two words can then be classified as being of the same or different type based on some threshold, and a precision-recall curve is obtained by varying the threshold. To evaluate embeddings across different operating points, the area under the precision-recall curve is calculated to yield the final evaluation metric, referred to as the average precision (AP).

We use data from the Switchboard corpus of English conversational telephone speech. Data is parameterized as Mel-frequency cepstral coefficients (MFCCs) with first and second order derivatives, yielding 39-dimensional feature vectors. Cepstral mean and variance normalization (CMVN) is applied per conversation side. For the training set $\mathcal{Y}_{\text{train}}$ we use the set of about 10k word tokens from [11, 12]; it consists of word segments of at least 5 characters and 0.5 seconds in duration extracted from a forced alignment of the transcriptions, and comprises about 105 minutes of speech. For the Siamsese CNNs, this set results in about 100k word segment pairs for $\mathcal{S}_{\text{train}}$. For testing, we use the 11k-token set $\mathcal{Y}_{\text{test}}$ from [3, 11, 12], making the results from these studies directly comparable to the results obtained here.[2] This set was extracted from a portion of Switchboard distinct from $\mathcal{Y}_{\text{train}}$. Similarly, we extracted an 11k-token development set.

As mentioned in Section 1, recent studies [11, 12] have also been using frame-level embedding approaches in combination with DTW to perform the same-different task. These approaches map the original features to a new frame-level representation that is tailored to word discrimination. We compare our results to that of [11], which uses posteriograms over a partitioned universal background model (UBM), as well as [12], which uses a correspondence autoencoder.

3.2. Network architectures

We used the Theano [28] toolkit to implement the CNN-based models of Sections 2.2 and 2.3.[3] Models are trained using ADADELTA [29], an adaptive learning rate stochastic optimization method that adapts over time based on an accumulation of past gradients; we set the momentum hyper-parameter to $\rho = 0.9$ and the precision parameter to $\epsilon = 10^{-6}$. Input speech segments are padded to $n_{\text{pad}} = 200$ frames (2 s), which corresponds to the longest word segment in $\mathcal{Y}_{\text{train}}$. The architectures of the CNNs were optimized separately on the development data for each network type, resulting in the following structures:

- *Word classifier CNN:* 1-D convolution with 96 filters over 9 frames; ReLU; max pooling over 3 units; 1-D convolution with 96 filters over 8 units; ReLU; max pooling over 3 units; 1024-unit fully-connected ReLU; softmax layer over 1061 word types.
- *Word similarity Siamese CNN:* two convolutional and max pooling layers as above; 2048-unit fully-connected ReLU; 1024-unit fully-connected linear linear; terminates in loss $l(\mathbf{x}_1, \mathbf{x}_2)$.

For the word classifier CNN, we only train on words in $\mathcal{Y}_{\text{train}}$ that occur at least three times; this gives a subset of 87% of all tokens with 1061 unique word types. This minimum count was tuned on the development set. To see the effect of the convolutions, we also train a word classifier deep neural network (DNN) using two 2048-unit fully-connected ReLU layers and a 1061-unit softmax layer. For the Siamese CNN using $l_{\cos \text{ hinge}}$, we use a margin $m = 0.15$ (tuned on the development set). If we had used ReLUs in the final layer in the Siamese CNNs, the angles between embeddings would be restricted to $[0, \pi/2]$; we therefore use a final linear layer. All weights are initialized randomly; we run all models with five different initializations and report average performance and standard deviations.

3.3. Results

Table 1 shows AP performance on the test set from previous studies (models 1 to 4) as well as our newly implemented models (5 to 11).

The first three models perform word discrimination using DTW on frame-level embeddings of word segments; model 1 works directly on acoustic features, while models 2 and 3 work on features optimized for word discrimination. Model 3 yields the best previously reported result on this task. Model 4 is the best acoustic word embedding approach from [3] (Section 2.1), representing the best previous result for an approach that produces embeddings of whole word segments.

Models 5 to 11 are the neural network-based approaches. The effect of using the convolutional layers is evident from the large improvement in AP of model 6 over model 5. Both of these models are trained on the word type labels $\mathcal{W}_{\text{train}}$, which is also the type of supervision used for model 4, making the improvement of model

[1] We also tried Euclidean distance, but as in [3,12,13], cosine worked better.

[2] In [3], a slightly different training set was used. Nevertheless, the size of their training set is comparable to the set used here.

[3] CNN code: `https://github.com/kamperh/couscous`. Complete recipe: `https://github.com/kamperh/recipe_swbd_wordembeds`.

Table 1. Average precision (AP) on the test set. Models 1 to 3: best prior DTW-based approaches; model 4: best prior acoustic word embedding approach based on reference vectors; models 5 to 11: this work. For models 1 to 3, dimensionality (dim.) is at the frame level; for models 4 to 11 it is the segment embedding dimensionality.

#	Representation	Dim.	AP
1	MFCCs with CMVN	39	0.214
2	Best partitioned UBM [11]	100	0.286
3	Correspondence autoencoder [12]	100	0.469
4	Reference vector approach [3]	50	0.365
5	Word classifier DNN	1061	0.301 ± 0.005
6	Word classifier CNN	1061	0.532 ± 0.014
7	——	50	0.474 ± 0.012
8	Siamese CNN, $l_{\text{cos cos}^2}$ loss	1024	0.342 ± 0.026
9	Siamese CNN, $l_{\text{cos hinge}}$ loss	1024	$\mathbf{0.549} \pm 0.011$
10	——	50	0.504 ± 0.011
11	LDA on model 9	100	0.545 ± 0.011

6 over model 4 noteworthy.[4] The dimensionality of the acoustic embeddings of model 6, however, is much larger than that of model 4. We therefore also trained a version of model 6 where the embedding is obtained from a linear bottleneck layer inserted just before the final softmax layer. The lower-dimensional embeddings from this approach (model 7) still improves on model 4 by a sizable margin.

Of the Siamese CNNs, the model with the $l_{\text{cos hinge}}$ loss (model 9, Section 2.3) outperforms its $l_{\text{cos cos}^2}$ counterpart, and yields a large improvement over model 4, which was the previous best acoustic embedding approach. It also gives similar performance to the word classification CNN (model 6), even though the pair-wise side information $\mathcal{S}_{\text{train}}$ used for model 9 is a weaker form of supervision than the fully labelled supervision $\mathcal{W}_{\text{train}}$ used in model 6. When reducing the embedding dimensionality to 50 (model 10), AP is still higher than any of the $d = 50$ competitors. Model 9's improvement over model 3 is also interesting since the former does not use any DTW alignment information. Finally, model 11 shows that LDA on the output of model 9 does not yield any improvement, but does produce a much smaller embedding without loss in performance. This model uses exactly the same word class supervision $\mathcal{W}_{\text{train}}$ as models 6 and 7.

3.4. Further discussion and analysis

Although the structures of models 8 and 9 are identical, the model using $l_{\text{cos hinge}}$ significantly outperforms its counterpart using $l_{\text{cos cos}^2}$. This is in line with the fact that a loss like $l_{\text{cos hinge}}$, which optimizes embeddings based on *relative* distances between positive and negative pairs, is much more closely aligned with the discrimination task than a loss like $l_{\text{cos cos}^2}$, which looks at distances of word pairs in isolation (without regard to their distances relative to other pairs). The $l_{\text{cos hinge}}$ loss also allows more freedom in the model since it does not penalize same-word pairs $(\mathbf{x}_1, \mathbf{x}_2)$ if they are already more similar by the margin m than the corresponding different-word pairs $(\mathbf{x}_1, \mathbf{x}_3)$.

The closer match between the same-different task and the training loss $l_{\text{cos hinge}}$ could also explain the improvements over the DTW-based model 3 (both using exactly the same supervision $\mathcal{S}_{\text{train}}$); this latter model aims to learn better features at the local frame level, but does so without regard to the (relative) similarities of complete segments.

[4] For model 6, embeddings are taken from the final softmax output. We also experimented with embeddings from the softmax layer but before applying the exponential normalization; this gave worse development results.

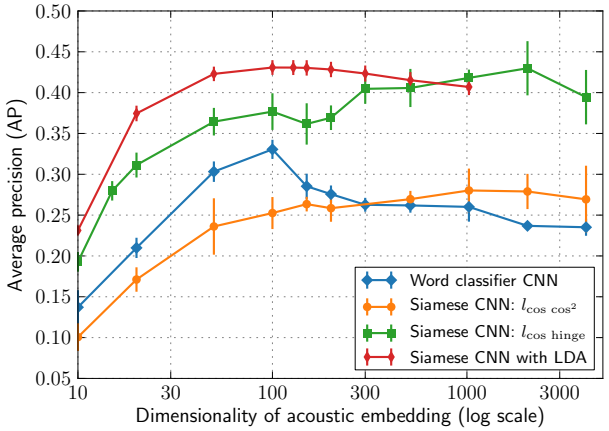

Fig. 2. Average precision (AP) on the development set for different CNN embedding approaches when varying the target dimensionality.

Figure 2 shows AP on the development set when varying the target dimensionalities of the different CNN-based approaches (see Section 2.4). The $l_{\text{cos hinge}}$ Siamese CNN outperforms all the other models (apart from the post-processed LDA model) at all operating points, and gives stable performance over a range of dimensionalities (300 and onwards). The word classifier CNN does much worse in this case (compared to the result of model 6), perhaps since embeddings here are not taken from the final layer, which is explicitly optimized for word classification, but from an intermediate layer. In contrast, for the Siamese CNNs, embeddings are always obtained directly from the layer that is optimized in the target loss. The figure also shows that when word labels $\mathcal{W}_{\text{train}}$ are available, compact embeddings can be obtained by performing LDA on top of the Siamese CNN representation, without loss in performance; this can prove to be important for downstream tasks which might require smaller embeddings.

Here, a relatively small set of labelled word examples $\mathcal{Y}_{\text{train}}$ is used to train the word classifier networks (as also done in the studies we compare to). In contrast, by using pairs of words and relative comparisons between them, a much larger set $\mathcal{S}_{\text{train}}$ is used for training Siamese networks. This type of paired supervision is ideal for generalizing to unseen word types, and is often easier to obtain in low-resource settings (see Section 2.3). While frame-level feature learning (model 3) can also use the larger pair-wise training set $\mathcal{S}_{\text{train}}$, such approaches need to be coupled with DTW, which is limiting.

4. CONCLUSION

We studied several acoustic word embedding approaches based on convolutional neural networks (CNNs); these networks take a whole-word speech segment as input and produce a fixed-dimensional vector. Our best new approach is a Siamese CNN that uses a hinge-based loss function to minimize the distance between word pairs of the same type relative to the distance between pairs of different types. On the *same-different* word discrimination task, this approach yields an average precision (AP) of 0.549, an improvement over the best previously published results on this task with whole-word embeddings (0.365 AP) and DTW with learned frame features (0.469 AP). A word classifier CNN performs similarly (0.532 AP) to the Siamese CNN, but requires much stronger labelled supervision, and performs worse at smaller dimensionalties. Future work will consider sequence models (e.g. RNNs, LSTMs), and will apply these embeddings to downstream tasks such as term discovery, speech recognition, and search.

5. REFERENCES

[1] M. De Wachter, M. Matton, K. Demuynck, P. Wambacq, R. Cools, and D. Van Compernolle, "Template-based continuous speech recognition," *IEEE Trans. Audio, Speech, Language Process.*, vol. 15, no. 4, pp. 1377–1390, 2007.

[2] G. Heigold, P. Nguyen, M. Weintraub, and V. Vanhoucke, "Investigations on exemplar-based features for speech recognition towards thousands of hours of unsupervised, noisy data," in *Proc. ICASSP*, 2012.

[3] K. Levin, K. Henry, A. Jansen, and K. Livescu, "Fixed-dimensional acoustic embeddings of variable-length segments in low-resource settings," in *Proc. ASRU*, 2013.

[4] S. Bengio and G. Heigold, "Word embeddings for speech recognition," in *Proc. Interspeech*, 2014.

[5] G. Chen, C. Parada, and T. N. Sainath, "Query-by-example keyword spotting using long short-term memory networks," in *Proc. ICASSP*, 2015.

[6] A. L. Maas, S. D. Miller, T. M. O'neil, A. Y. Ng, and P. Nguyen, "Word-level acoustic modeling with convolutional vector regression," in *Proc. ICML Workshop Representation Learn.*, 2012.

[7] O. J. Räsänen, "Generating hyperdimensional distributed representations from continuous-valued multivariate sensory input," in *Proc. CogSci*, 2015.

[8] C. S. Myers and L. R. Rabiner, "Connected digit recognition using a level-building DTW algorithm," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 29, no. 3, pp. 351–363, 1981.

[9] Y. Zhang and J. R. Glass, "Unsupervised spoken keyword spotting via segmental DTW on Gaussian posteriorgrams," in *Proc. ASRU*, 2009.

[10] Y. Zhang, R. Salakhutdinov, H.-A. Chang, and J. R. Glass, "Resource configurable spoken query detection using deep Boltzmann machines," in *Proc. ICASSP*, 2012.

[11] A. Jansen, S. Thomas, and H. Hermansky, "Weak top-down constraints for unsupervised acoustic model training," in *Proc. ICASSP*, 2013.

[12] H. Kamper, M. Elsner, A. Jansen, and S. J. Goldwater, "Unsupervised neural network based feature extraction using weak top-down constraints," in *Proc. ICASSP*, 2015.

[13] R. Thiollière, E. Dunbar, G. Synnaeve, M. Versteegh, and E. Dupoux, "A hybrid dynamic time warping-deep neural network architecture for unsupervised acoustic modeling," in *Proc. Interspeech*, 2015.

[14] L. R. Rabiner, A. E. Rosenberg, and S. E. Levinson, "Considerations in dynamic time warping algorithms for discrete word recognition," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 26, no. 6, pp. 575–582, 1978.

[15] K. Levin, A. Jansen, and B. Van Durme, "Segmental acoustic indexing for zero resource keyword search," in *Proc. ICASSP*, 2015.

[16] M. A. Carlin, S. Thomas, A. Jansen, and H. Hermansky, "Rapid evaluation of speech representations for spoken term discovery," in *Proc. Interspeech*, 2011.

[17] J. Bromley, J. W. Bentz, L. Bottou, I. Guyon, Y. LeCun, C. Moore, E. Säckinger, and R. Shah, "Signature verification using a 'Siamese' time delay neural network," *Int. J. Pattern Rec.*, vol. 7, no. 4, pp. 669–688, 1993.

[18] H. Kamper, A. Jansen, and S. Goldwater, "Fully unsupervised small-vocabulary speech recognition using a segmental Bayesian model," in *Proc. Interspeech*, 2015.

[19] D. Renshaw, H. Kamper, A. Jansen, and S. J. Goldwater, "A comparison of neural network methods for unsupervised representation learning on the Zero Resource Speech Challenge," in *Proc. Interspeech*, 2015.

[20] A. S. Park and J. R. Glass, "Unsupervised pattern discovery in speech," *IEEE Trans. Audio, Speech, Language Process.*, vol. 16, no. 1, pp. 186–197, 2008.

[21] A. Jansen and B. Van Durme, "Efficient spoken term discovery using randomized algorithms," in *Proc. ASRU*, 2011.

[22] L. Badino, C. Canevari, L. Fadiga, and G. Metta, "An auto-encoder based approach to unsupervised learning of subword units," in *Proc. ICASSP*, 2014.

[23] P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. Heck, "Learning deep structured semantic models for web search using clickthrough data," in *Proc. CIMK*, 2013.

[24] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[25] J. Wieting, M. Bansal, K. Gimpel, and K. Livescu, "From paraphrase database to compositional paraphrase model and back," *Trans. ACL*, vol. 3, pp. 345–358, 2015.

[26] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *Proc. CVPR*, 2006.

[27] G. Synnaeve, T. Schatz, and E. Dupoux, "Phonetics embedding learning with side information," in *Proc. SLT*, 2014.

[28] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, "Theano: a CPU and GPU math expression compiler," in *Proc. SciPy*, 2010.

[29] M. D. Zeiler, "ADADELTA: An adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.