UNSUPERVISED NEURAL NETWORK BASED FEATURE EXTRACTION USING WEAK TOP-DOWN CONSTRAINTS

Herman Kamper[1,2]*, Micha Elsner*[3]*, Aren Jansen*[4]*, Sharon Goldwater*[2]

[1]CSTR and [2]ILCC, School of Informatics, University of Edinburgh, UK
[3] Department of Linguistics, The Ohio State University, USA
[4] HLTCOE and CLSP, Johns Hopkins University, USA

h.kamper@sms.ed.ac.uk, melsner@ling.osu.edu, aren@jhu.edu, sgwater@inf.ed.ac.uk

ABSTRACT

Deep neural networks (DNNs) have become a standard component in supervised ASR, used in both data-driven feature extraction and acoustic modelling. Supervision is typically obtained from a forced alignment that provides phone class targets, requiring transcriptions and pronunciations. We propose a novel unsupervised DNN-based feature extractor that can be trained without these resources in zero-resource settings. Using unsupervised term discovery, we find pairs of isolated word examples of the same unknown type; these provide weak top-down supervision. For each pair, dynamic programming is used to align the feature frames of the two words. Matching frames are presented as input-output pairs to a deep autoencoder (AE) neural network. Using this AE as feature extractor in a word discrimination task, we achieve 64% relative improvement over a previous state-of-the-art system, 57% improvement relative to a bottom-up trained deep AE, and come to within 23% of a supervised system.

***Index Terms*—** Unsupervised feature extraction, deep neural networks, zero-resource speech processing, top-down constraints

1. INTRODUCTION

The use of deep neural networks (DNNs) has recently led to great advances in supervised automatic speech recognition (ASR) [1, 2]. One view of these networks is that a deep feature extractor (often initialized using unsupervised pretraining) is learnt jointly with a supervised classifier, predicting phone classes in the case of ASR [3]. Despite the resurgence of neural network (NN) research in the supervised domain, the use of NNs as feature extractors for unsupervised 'zero-resource' speech processing tasks has received little attention.

Zero-resource technology aims to solve tasks such as phonetic and lexical discovery [4, 5], spoken document retrieval [6], and query-by-example search [7, 8] by using only raw speech data. Advances in this area would enable technologies in languages where transcribed data collection is too expensive, or where it is impossible (e.g. for unwritten languages). The limited use of NNs in this domain is not surprising since, without transcriptions or dictionaries, it is impossible to obtain the phone class targets used for fine-tuning. Some [9, 10] have considered unsupervised autoencoder NNs, but not explicitly for improved feature extraction. We present a novel training algorithm for deep networks in the zero-resource setting, employing a form of weak supervision with the purpose of unsupervised feature extraction. Since the aim is a better general representation of speech, our work is relevant to any downstream zero-resource task.

The weak top-down supervision we use is obtained from an unsupervised term discovery (UTD) algorithm, which finds reoccurring word-like patterns in a speech collection [11, 12]. Other zero-resource studies have also used such top-down constraints. In [13], 'pronunciations' for UTD-discovered words were obtained after a bottom-up tokenization of the speech into subword-like units. In [14], whole-word HMMs were trained on discovered words; similar HMM states were then clustered to automatically find subword unit models. When using these top-down constraints alone, only the discovered word examples are used for model estimation, and much data is disregarded.

This was addressed in [15]. First, a Gaussian mixture model (GMM) is trained bottom-up on a speech corpus, providing a universal background model (UBM) that takes into account all data. UTD then finds reoccurring words in the corpus. For each pair of word segments of the same type, frames are aligned using dynamic time warping (DTW). Based on the idea that different realizations of the same word should have a similar underlying subword sequence, UBM components in matching frames are attributed to the same subword unit. The resulting partitioned UBM is a type of unsupervised acoustic model where every partition corresponds to a subword unit. In a multi-speaker word discrimination task, posteriorgrams calculated over the partitioned UBM significantly outperformed the original features.

As in [15] (also in much earlier work [16], and very recently [17]), the central idea of our new NN-based algorithm is that aligned frames from different instances of the same word should contain information useful for finding a better feature representation. Using layer-wise pretraining of a stacked autoencoder (AE), our approach uses a large corpus of untranscribed speech to find a suitable initialization. As in [15], word pairs discovered using UTD are then DTW-aligned to obtain frame-level constraints, which are presented as input-output pairs to the AE. We refer to this NN, trained using weak top-down constraints, as a *correspondence AE*. We use this AE as an unsupervised feature extractor by taking the encoding from a middle layer. In a word discrimination task, we compare the new feature representation to the original input features, as well as features obtained from posteriorgrams over the partitioned UBM of [15]. One shortcoming of [15] is that the UTD-step was simulated by using gold standard word pairs extracted from transcriptions; here we use a practical UTD system [12]. Our results show that NN-based feature extraction, which has proven so advantageous in supervised ASR, can also result in major improvements in the extreme zero-resource case.

2. UNSUPERVISED TRAINING ALGORITHM

We first present a concise overview of autoencoders (AEs) and how these can be used to initialize deep neural networks (DNNs). We then present the training algorithm of a correspondence AE, a neural network using weak top-down supervision in the form of word pairs obtained from an unsupervised term discovery (UTD) system.

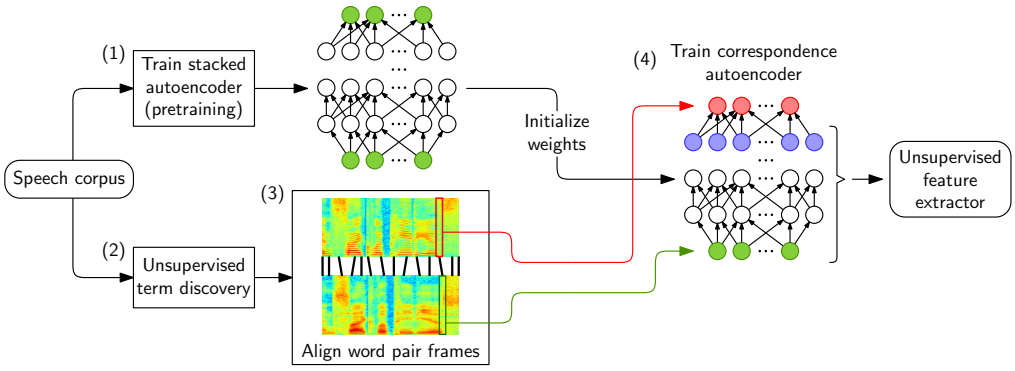

Fig. 1. Algorithm schematic for training the correspondence autoencoder for unsupervised feature extraction.

2.1. Autoencoders, pretraining and deep neural networks

An AE is a feedforward neural network where the target output of the network is equal to its input [18, §4.6]. A single-layer AE encodes its input $\mathbf{x} \in \mathbb{R}^D$ to a hidden representation $\mathbf{a} \in \mathbb{R}^{D^{(0)}}$ using $\mathbf{a} = s(\mathbf{W}^{(0)}\mathbf{x} + \mathbf{b}^{(0)})$, where $\mathbf{W}^{(0)}$ is a weight matrix, $\mathbf{b}^{(0)}$ is a bias vector, and s is a non-linear vector function ($\tanh$ in our case). The output $\mathbf{z} \in \mathbb{R}^D$ of the AE is obtained by decoding the hidden representation using $\mathbf{z} = \mathbf{W}^{(1)}\mathbf{a} + \mathbf{b}^{(1)}$. The network is trained using backpropagation to achieve a minimum reconstruction error, typically using the loss function $||\mathbf{x} - \mathbf{z}||^2$ when dealing with real-valued data. A deep network can be obtained by stacking several AEs, each AE-layer taking as input the encoding from the layer below it. This stacked AE is trained one layer at a time, each layer minimizing the loss of its output with respect to the original input $\mathbf{x}$.

AEs are often used for non-linear dimensionality reduction by having a hidden layer that is narrower than its input dimensionality [19]. Although AEs with more hidden units than the input are in principle able to learn the identity function to achieve zero reconstruction error, [20] found that in practice such networks often still learn a useful representation since early stopping provides a form of regularization. In our own experiments, we found that such AEs provide a crucial initialization for our new AE-like network; our aim here is not dimensionality reduction, but to find a better feature representation.

In a supervised setting, training a stacked AE, as explained above, is one form of *unsupervised pretraining* of a NN. This is followed by *supervised fine-tuning* where an additional output layer is added to perform some supervised prediction task, resulting in a DNN [20].

2.2. The correspondence autoencoder

Here we present the novel training algorithm for a NN which we call a *correspondence autoencoder*. While standard stacked AEs trained on speech (such as those of [9, 10]) use the same feature frame(s) as input and output, the correspondence AE uses weak top-down constraints in the form of (discovered) word pairs to have input and output frames from different instances of the same word. The algorithm follows four steps, which are illustrated in Figure 1.

Step 1: Train a stacked AE. A corpus of speech is parametrized into the set $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$, where each $\mathbf{x}_t \in \mathbb{R}^D$ is the frame-level acoustic feature representation of the signal (e.g. MFCCs). Given $\mathcal{X}$, a stacked AE is trained unsupervised directly on the acoustic features. Using this network as initialization for the correspondence AE, we are taking advantage of a large amount of untranscribed speech data to start at a point in weight space where the network provides a representation close to the acoustic features themselves.

Step 2: Spoken term discovery. A UTD system is run on the speech corpus. This produces a collection of N word segment pairs, which we use as weak top-down constraints. In [14, 15], this step was simulated by using gold standard word segment pairs extracted from transcriptions. We present experiments both when using gold standard word pairs and when using pairs obtained from UTD [12].

Step 3: Align word pair frames. In the third step of the algorithm, the N word-level constraints from UTD are converted to frame-level constraints. For each word pair, a dynamic time warping (DTW) alignment [21] is performed using cosine distance as similarity metric to find a minimum-cost frame alignment between the two words. This is done for all N word pairs, and taken together provides a set $\mathcal{F} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{F}$ of F frame-level constraints. Note that although each frame pair is unique, the time warping allowed in the alignment can result in the same frame occurring in multiple pairs.

Step 4: Train the correspondence AE. Using the stacked AE from step 1 as initialization, the correspondence AE is trained on the frame-level pairs $\mathcal{F}$. For every pair $(\mathbf{x}_i, \mathbf{y}_i)$, $\mathbf{x}_i$ is presented as input to the network while $\mathbf{y}_i$ is taken as output. The complete network is then trained using backpropagation. Although we refer to the resulting network as an autoencoder to emphasize the relationship between its input and output, it can also be described differently. Firstly, it can be seen as a type of denoising autoencoder [22], an AE were the input is corrupted by adding Gaussian noise or setting some inputs to zero; this allows more robust features to be learnt. In our case, the input $\mathbf{x}_i$ can be seen as a corrupted version of output $\mathbf{y}_i$. Secondly, our network can also be described as a standard DNN with a linear output layer, initialized using layer-wise pretraining. Normally, the term DNN is associated with a supervised prediction task, and our network can be seen as predicting $\mathbf{y}_i$ when presented with input $\mathbf{x}_i$.

Our aim is to use the correspondence AE as an unsupervised feature extractor that provides better word-discrimination properties than the original features. To use it as such, the encoding obtained from one of its middle layers is finally taken as the feature representation of new input speech, as illustrated in the right-most block of Figure 1.

3. EXPERIMENTS

3.1. Experimental setup

We use data from the Switchboard corpus of English conversational telephone speech. Using HTK [23], 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 training the stacked AE (step 1), 180 conversations are used

which corresponds to about 23 hours of speech. This same set was used for UBM training in [15]. For experiments using gold standard word pairs, we use the set used in [15] for partitioning the UBM; it consists of word segments of at least 5 characters and 0.5 seconds in duration extracted from a forced alignment of the transcriptions of the 23 hour training set. The full gold standard set consists of nearly $N = 100$k word segment pairs, comprised of about 105 minutes of speech. About 3% of these pairs are same-speaker word pairs. DTW alignment of the 100k pairs (step 3) provides a frame-level constraint set of about $F = 7$M frames, on which the correspondence AE is trained (step 4). In our truly unsupervised setup, we use word pairs discovered using the UTD system of [12]. We consider two sets. The first consists of about $N = 25$k word pairs obtained by searching the above 23 hour training set. About 17% of these pairs are produced by the same speaker. The second set consists of about 80k pairs obtained by including an additional 180 conversations in the search. About 11% of these are same-speaker pairs.

All NNs are trained with minibatch stochastic gradient descent using Pylearn2 [24]. A batch size of 256 is used, with 30 epochs of pretraining (step 1) at a learning rate of $250 \cdot 10^{-6}$ and 120 epochs of correspondence AE training (step 4) at a learning rate of $2 \cdot 10^{-3}$. Initially these parameters were set to the values given in [25], and were then adjusted based on training set loss function curves and development tests. Although it is common to use nine or eleven sliding frames as input to DNN ASR systems, we use single-frame input. This was also done in [10], and allows for fair comparison with previous work. However, multi-frame input is the focus of future work.

Our goal is to show the suitability of features from the correspondence AE in downstream zero-resource search and recognition tasks. We therefore use a multi-speaker word discrimination task developed specifically for this purpose [26]. The *same-different task* quantifies the ability of a speech representation to associate words of the same type and to discriminate between words of different types. For every word pair in a test set of pre-segmented words, the DTW distance is calculated using the feature representation under evaluation. 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 representations 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). In [26] perfect correlation was found between AP and phone error rate in a supervised setting, justifying it as an effective way to evaluate different representations of speech in unsupervised settings.

We use the same test set for the same-different task as that used in [15]. It consists of about 11k word tokens drawn from a portion of Switchboard distinct from any of the above sets. The set results in 60.7M word pairs of which 96k are from the same word type. Of these 96k pairs, only about 3% were produced by the same speaker. Additionally, we also extracted a comparable 11k-token development set, again from a disjoint portion of Switchboard. Since tuning the hyperparameters of a NN is often an art, we present performance on the development set when varying some of these parameters.

Since we share a common test setup, we can compare our feature representation directly to previous work. As a first baseline we use MFCCs directly to perform the same-different task. We then compare our model to the partitioned UBM of [15] (Section 1) and the supervised NN systems of [26]. These single-layer multistream NNs were trained to estimate phone class posterior probabilities on transcribed speech data from the Switchboard and CallHome corpora, and have a comparable number of parameters to our networks. We consider systems trained on 10 and 100 hours of speech. For the partitioned UBM and NNs, test words are parameterized by generating posteriorgrams over components/phone classes, and symmetrized Kullback-Leibler divergence is used as frame-level metric for the same-different task. For MFCCs and our AE-based features, cosine distance is used.

3.2. Choosing the network architecture

Choosing the hyperparameters of NNs is challenging. We therefore describe the optimization process followed on the development data.

To use the correspondence AE as feature extractor, the encoding from one of its middle layers is taken. We found that using features from between the fourth-last to second-last encoding layers gave robust performance. It is common practice to use a narrow bottleneck layer to force the network to learn a lower-dimensional encoding at a particular layer. We experimented with this, but found that performance was similar or slightly worse in most cases and therefore decided to only vary the number of hidden layers and units.

We experimented with correspondence AEs ranging from 3 to 21 hidden layers with 50, 100 and 150 hidden units per layer trained on the 100k gold standard word-pair set. AP performance on the development set is presented in Figure 2. On this set, all networks achieve performance greater than that of the input MFCCs. For all three hidden unit settings, performance is within 12% relative to the respective optimal settings for networks with 7 to 21 layers.

3.3. Gold standard weak top-down constraints

Table 1 shows the AP performance on the test set using the baseline MFCCs, the UBM models from [15], our AE networks, and the supervised NNs from [26]. The partitioned UBM and the correspondence AE were both trained on the gold standard 100k word-pair set. The optimal AE network on the development set (Figure 2) was used.

As reported before, although the UBM alone does not yield significant gains, the 100-component partitioned UBM results in a 34% relative improvement over the baseline MFCCs. Analogous to the UBM, the stacked AE alone also produces no improvement over the MFCCs. This contrasts with the results reported in [10], where small improvements were obtained. However, [10] used much smaller training and test sets, had a different training setup, and had the explicit aim of tokenizing speech into subword-like units rather than unsupervised feature extraction.

Without initializing the weights from the stacked AE, very poor performance is achieved by the correspondence AE (0.024 AP). However, when pretraining is used, the resulting correspondence AE

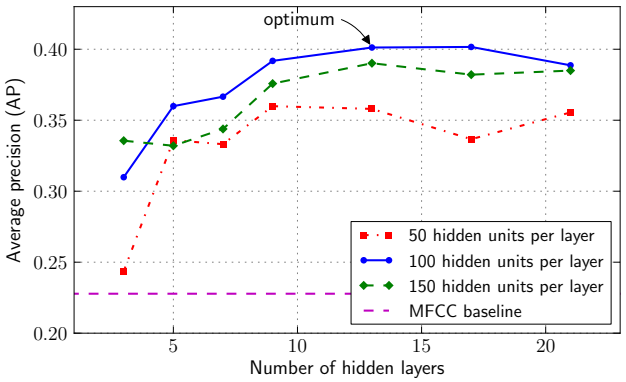

Fig. 2. Average precision (AP) on the development set for correspondence AEs with varying numbers of hidden layers and units. In each case the best hidden layer on the development set was used.

Table 1. Average precision (AP) on the test set using MFCCs, the UBM and partitioned UBM, the stacked and correspondence AEs trained on the 100k gold standard word pairs, and supervised NNs.

Features	AP
MFCC with CMVN	0.214
UBM with 1024 components [15]	0.222
1024-UBM partitioned into 100 components [15]	0.286
100-unit, 13-layer stacked AE	0.215
100-unit, 13-layer correspondence AE, no pretraining	0.024
100-unit, 13-layer correspondence AE, pretraining	**0.469**
English NN, 10 hours [26]	0.439
English NN, 100 hours [26]	0.516

Table 2. Average precision (AP) on the test set using the partitioned UBM and correspondence AEs when varying the number of gold standard word pairs N, with F the resulting number of frame pairs.

N	F	Partitioned UBM AP [15]	Correspondence AE AP
10^5	$7 \cdot 10^6$	0.286	0.469
10^4	$7 \cdot 10^5$	0.284	0.385
10^3	$7 \cdot 10^4$	0.266	0.286
10^2	$7 \cdot 10^3$	0.206	0.259

outperforms the partitioned UBM by 64% relative, and more than doubles the performance of the original MFCC features. This improvement of the correspondence AE (0.469 AP) over the partitioned UBM (0.286), both using exactly the same weak form of supervision, indicates that the NN is much better able to exploit the information gained from the top-down constraints than the GMM-based model.

The correspondence AE also outperforms the 10-hour supervised NN on this task, and comes close to the level of the 100-hour system. Since we use gold standard word pairs here comprised of only 105 minutes of speech, these results are potentially significant from a low-resource perspective. Although these improvements are surprising, the form of explicit pair-wise supervision provided to the correspondence AE is closely related to the word discrimination task. Further investigation of these observations is the focus of future work.

As in [15], to investigate dependence on the amount of supervision, we varied the number of gold standard word-pair constraints $N = 100\text{k}, 10\text{k}, 1\text{k}$ and 100 by taking random subsets of the full 100k set; consequently, the number of frame-level constraints F is varied. Results are shown in Table 2. For every set, the correspondence AE was optimized on the development data. In all cases the correspondence AE outperforms the partitioned UBM and the baseline MFCCs. With as few as 1k pairs, the correspondence AE gives the same performance as the partitioned UBM trained with all pairs.

3.4. Unsupervised term discovery weak top-down constraints

Finally, we present truly unsupervised results where an UTD system [12] is used to provide the word pairs for weak supervision. Results are shown in Table 3, with some baselines repeated from Table 1. Two UTD runs are used (Section 3.1), and Table 1 includes their pair-wise accuracies. The first produced 25k word pairs at an accuracy of 46%, while the second produced 80k pairs at 36%. Correspondence AEs were trained separately on the two sets of weak top-down constraints, with each optimized on the development data.

Both correspondence AEs significantly outperform the MFCCs and stacked AE baselines by more than 57% relative in AP, coming to within 23% of the 10-hour supervised NN baseline. Compared to the partitioned UBM trained on 100k gold standard word pairs (0.286 AP, Table 1), the completely unsupervised correspondence AEs still perform better by almost 19%, despite the much noisier form of weak supervision. Performance of the best correspondence AE from the gold standard word-pair case (0.469 AP, Table 1) relative to the best unsupervised correspondence AE (0.341 AP, Table 3), indicates that the noise introduced by the true UTD-step results in a penalty of 34%; the correspondence AE nevertheless provides a better representation than the other unsupervised baselines. It is unclear if the same will hold for the previous models [14, 15] where the truly zero-resource case was not considered.

Table 3. Average precision (AP) on the test set when using weak top-down constraints from unsupervised term discovery (UTD). The number of word pairs N and accuracy of the UTD system is also shown.

Features	N	UTD Acc.	AP
MFCC with CMVN	-	-	0.214
100-unit, 13-layer stacked AE	-	-	0.215
100-unit, 9-layer correspond. AE	25k	46%	0.339
100-unit, 13-layer correspond. AE	80k	36%	**0.341**
English NN, 10 hours [26]	-	-	0.439
English NN, 100 hours [26]	-	-	0.516

A comparison of the two correspondence AEs in Table 3 shows that, despite using significantly more pairs and allowing a deeper network to be trained, the 80k set does not provide a major improvement over the 25k set. This is attributed to the lower word-pair accuracy of the former, and shows that there is a trade-off between UTD accuracy and the number of pairs produced. Compared to the analysis in Table 2, the 25k unsupervised-obtained pairs still provide more useful supervision than 1k gold standard word pairs. A finer grained investigation of the trade-off between word pairs and accuracy, which can be varied by searching more data or by adjusting the search threshold, is the focus of future work.

4. CONCLUSIONS AND FUTURE WORK

We introduced a novel scheme for training an unsupervised autoencoder (AE) neural network feature extractor, which uses weak top-down supervision from word pairs obtained using an unsupervised term discovery (UTD) system. We evaluated this *correspondence AE* in a word discrimination task designed for comparing feature representations in zero-resource settings. In experiments where gold standard word pairs from transcriptions were used for weak supervision, we showed that our proposed AE gives a 64% relative improvement over previously reported results using the same test setup. In our own truly unsupervised setup where UTD was used to provide the weak top-down constraints, our network outperformed both baseline MFCCs and a standard stacked AE by more than 57%, coming to within 23% of a supervised system trained on 10 hours of transcribed speech. We conclude that the correspondence AE could greatly benefit downstream zero-resource tasks where transcriptions and dictionaries are not available for system development. Future work will include using the AE feature extractor to improve UTD accuracy, which in turn can improve the weak top-down constraints, and so forth. We also aim to explore the suitability of our AE feature extractor for supervised ASR.

Acknowledgements. We would like to thank Pawel Swietojanski, Liang Lu, Simon King and Daniel Renshaw for helpful discussions.

5. REFERENCES

[1] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *IEEE Trans. Audio, Speech, Language Process.*, vol. 20, no. 1, pp. 30–42, 2012.

[2] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, 2012.

[3] D. Yu, M. Seltzer, J. Li, J.-T. Huang, and F. Seide, "Feature learning in deep neural networks - studies on speech recognition," in *Proc. ICLR*, 2013.

[4] A. Jansen, K. Church, and H. Hermansky, "Towards spoken term discovery at scale with zero resources," in *Proc. Interspeech*, 2010.

[5] C. Lee and J. R. Glass, "A nonparametric Bayesian approach to acoustic model discovery," in *Proc. ACL*, 2012.

[6] H. Gish, M.-H. Siu, A. Chan, and B. Belfield, "Unsupervised training of an HMM-based speech recognizer for topic classification," in *Proc. Interspeech*, 2009.

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

[8] F. Metze, X. Anguera, E. Barnard, M. Davel, and G. Gravier, "The spoken web search task at MediaEval 2012," in *Proc. ICASSP*, 2013.

[9] M. D. Zeiler, M. Ranzato, R. Monga, M. Mao, K. Yang, Q. V. Le, P. Nguyen, A. Senior, V. Vanhoucke, J. Dean, and G. E. Hinton, "On rectified linear units for speech processing," in *Proc. ICASSP*, 2013.

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

[11] A. Park and J. R. Glass, "Unsupervised word acquisition from speech using pattern discovery," in *Proc. ICASSP*, 2006.

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

[13] O. Walter, T. Korthals, R. Haeb-Umbach, and B. Raj, "A hierarchical system for word discovery exploiting DTW-based initialization," in *Proc. ASRU*, 2013.

[14] A. Jansen and K. Church, "Towards unsupervised training of speaker independent acoustic models," in *Proc. Interspeech*, 2011.

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

[16] M. Hunt, S. M. Richardson, D. C. Bateman, and A. Piau, "An investigation of PLP and IMELDA acoustic representations and of their potential for combination," in *Proc. ICASSP*, 1991.

[17] G. Synnaeve1, T. Schatz, and E. Dupoux, "Phonetics embedding learning with side information," in *Proc. SLT*, 2014.

[18] Y. Bengio, "Learning deep architectures for AI," *Found. Trends Mach. Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[19] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[20] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Proc. NIPS*, 2007.

[21] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 26, no. 1, pp. 43–49, 1978.

[22] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proc. ICML*, 2008.

[23] S. J. Young, G. Evermann, M. J. F. Gales, T. Hain, D. Kershaw, X. Liu, G. L. Moore, J. J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. C. Woodland, *The HTK Book (for HTK Version 3.4)*, Cambridge University Engineering Department, 2009.

[24] I. J. Goodfellow, D. Warde-Farley, P. Lamblin, V. Dumoulin, M. Mirza, R. Pascanu, J. Bergstra, F. Bastien, and Y. Bengio, "Pylearn2: a machine learning research library," *arXiv:1308.4214*, 2013.

[25] C. Weng, D. Yu, S. Watanabe, and B.-H. Juang, "Recurrent deep neural networks for robust speech recognition," in *Proc. ICASSP*, 2014.

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