Word-level Acoustic Modeling with Convolutional Vector Regression

Andrew L. Maas AMAAS@CS.STANFORD.EDU
Stephen D. Miller SDMILLER@CS.STANFORD.EDU
Tyler M. O'Neil TONEIL@CS.STANFORD.EDU
Andrew Y. Ng ANG@CS.STANFORD.EDU
Stanford University, CA 94305 USA

Patrick Nguyen DRPNG@GOOGLE.COM
Google, Inc., Mountain View, CA 94043 USA

Abstract

We introduce a model that maps variable-length word utterances to a word vector space using convolutional neural networks. Convolutional networks are a rich class of architecture that, through many nonlinear layers, can model complex functions of their input. Our approach models entire word acoustics rather than short windows as in previous work. We introduce the notion of mapping these word inputs to a word vector space, rather than trying to solve the massively multi-class problem of word classification. Regressing to word vectors offers many opportunities for further work in this domain, as many techniques exist to learn word vectors for different notions of word similarity. We experiment on hundreds of hours of broadcast news, and demonstrate our model can accurately recognize spoken words. Further, we use our model to build features for the SCARF speech recognition system and achieve an improvement in large vocabulary continuous speech recognition over a baseline system.

1. Introduction

Speech recognition remains a challenging goal of artificial intelligence with countless potential applications. Modern speech recognition systems are the result of thousands of man-hours, which is reflected by their enormous complexity. Indeed, training a state-of-the-art recognizer involves several re-estimation stages, speaker normalization, and signal processing. Each of these refinements is the result of months or years of work, and typically result in small absolute improvements to the final system error rate.

Recent work has shown substantial improvements to such systems are possible by using deep neural networks to model the acoustic signal (Dahl et al., 2011; Mohamed et al., 2011; Vinyals & Ravuri, 2011). Rather than use domain expertise to engineer modifications to the already complicated existing systems, the deep learning approach instead learns a highly expressive model from large amounts of training data. Such approaches now comprise state-of-the-art systems on many computer vision tasks while employing little or no domain-specific engineering (Le et al., 2011b; Wang et al., 2010). The speech domain too has rich opportunities to replace existing domain-engineered approaches with models in the deep learning regime. Replacing such complexity facilitates further research in speech without the need for overwhelming amounts of domain-specific knowledge.

In this work, we explore using neural networks to model the acoustics of entire words rather than short, fixed-length intervals. Since words are long, variable-length acoustic elements, we use convolutional networks to collapse the variable-length inputs to a fixed-size representation. Previous work by Lee et al. (2010) uses an unsupervised convolutional model to learn features for phoneme classification. Chopra et al. (2011) use a convolution and temporal pooling architecture, also for TIMIT phone classification. We build upon previous work on convolutional neural networks for speech, but directly addresses the most challenging problem in the speech domain – large vocabulary continuous speech recognition (LVCSR). We build a con-

Appearing in *ICML 2012 Workshop on Representation Learning*, Edinburgh, Scotland, UK, 2012. Copyright 2012 by the author(s)/owner(s).

volutional network capable of classifying tens of thousands of words using long duration acoustic segments.

Our convolutional network must output information about word identity, which for the broadcast news dataset we consider requires reasoning over 75,000 unique words. A standard softmax or logistic regression classifier scales poorly to this number of classes. Instead we use *vector regression* where each word in the vocabulary is assigned a low-dimensional real-valued vector, and the network predicts the word vector given its acoustic input. This idea is related to error correcting output codes for large multi-class problems (Dietterich & Bakiri, 1995).

Much recent work from the deep learning community and others has focused on learning word vectors sensitive to different notions of similarity (Collobert & Weston, 2008; Mnih & Hinton, 2007; Turney & Pantel, 2010). Our model attempts to project an acoustic input directly into such word vector spaces. Mapping acoustics to semantics has potential applications not only in LVCSR, but also for dialogue systems such as voice search, and recognizing speaker characteristics like emotive state. Our model provides a framework for mapping acoustics to different choices of word vector space, and lays the foundation for joint models which simultaneously learn word vectors from text and acoustics.

We introduce our general convolutional network architecture which maps variable-length word acoustics to a fixed-dimensional representation, then discuss our novel vector regression output layer for speech recognition. With experiments on hundreds of hours of broadcast news we show our model can classify words in a 10,000-way decision task. Finally, we integrate our model's word predictions with the SCARF recognizers and demonstrate improvement over a baseline LVCSR system.

2. Model

We wish to interpret the acoustic signal v of an entire word w, and transform it into a vector $\phi^{(w)}$ representing the meaning of the word, or the word itself. This task corresponds to learning a function $f(v)$ such that $f(v) = \phi^{(w)}$. This function is highly complex, as acoustics depend on speaker factors such as gender and emotive state, as well as environmental factors such as background noise and microphone placement. Furthermore, since word utterances v are variable length with a substantial range in duration, the function f must accept a variable-length acoustic input.

A substantial amount of work in speech recognition focuses on engineering away distortions and speaker variation, corresponding to learning a function $g(v) = v'$. Rather than mapping v to word semantics or identity, $g(v)$ simply attempts to produce a 'clean' or canonical acoustic representation, v' (Gales & Young, 2007). Our work instead takes a deep learning approach to the problem – leveraging an abundance of data to train a model with large capacity. The goal of this approach is to fit a highly expressive model to approximate the true function $f(v)$ by leveraging an abundance of labeled examples of the form $(v, \phi^{(w)})$. This direct approach avoids the potentially challenging engineering effort of modeling away distortions in v and prescribing how the "true" v' should look.

Deep neural networks have recently demonstrated substantial success as acoustic models in HMM-based speech recognition systems (Dahl et al., 2011). These approaches reason over very short acoustic spans, typically on the order of 75 to 175 milliseconds (ms). Because the inputs are small, fixed-duration acoustic spans, the neural network directly models the entire acoustic span. Our acoustic input is instead an entire word, which are variable in length and last 312 ms on average but often last 500 ms or more.

With such large variations in temporal extent, a straightforward neural network approach can not be used. Instead, we use a *convolutional* neural network which can model long duration words with a tractable number of parameters. This general architecture efficiently collapses variable-length inputs into a fixed-size representation. As in previous work on convolutional models applied to images, we can introduce arbitrary numbers of convolution and pooling layers to achieve a highly expressive multi-layer architecture (LeCun et al., 1998).

Using convolution and pooling, we obtain a fixed-dimensional vector representation z of an acoustic input span v. The convolutional network thus represents a function $z = h(v)$ which becomes increasingly nonlinear as we increase the number of filters in each layer, the number of layers, and other features of the network architecture. This is detailed in Section 2.1.

Ultimately the model must predict a word vector $\hat{\phi}$ which approximates the true word vector $\phi^{(w)}$ corresponding to the utterance. The model already has substantial nonlinearities in $h(v)$ so we use a simple, affine mapping from z to the predicted vector $\hat{\phi}$. This is discussed in Section 2.2.

By representing words as vectors we have a fixed size output layer to capture an arbitrarily large vocabulary, because each word is simply a point in high-

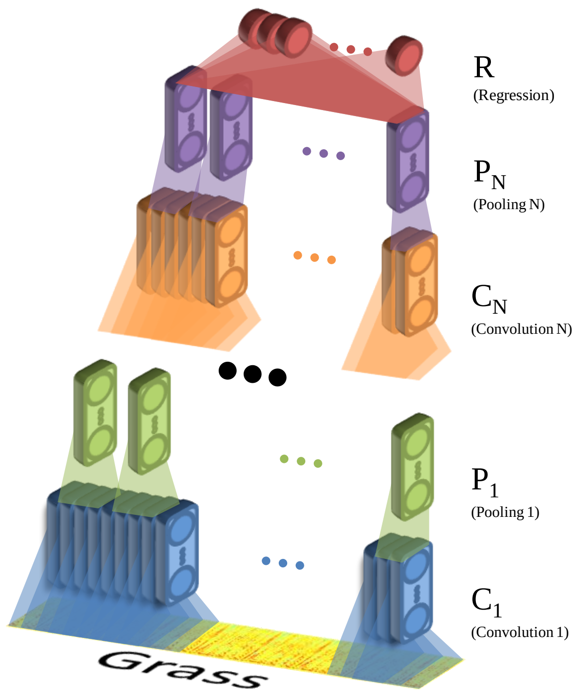

Figure 1. Convolutional network architecture.

dimensional space. This approach results in a network model size which is constant as the vocabulary increases. In contrast, a softmax classifier over all words results in a model which scales linearly with the vocabulary size. This scales poorly when we consider large vocabulary speech recognition tasks, and training classifier parameters for words which occur infrequently is difficult from an optimization perspective.

2.1. Convolution and Pooling

We are given an acoustic input signal $v \in \mathbb{R}^{n^{(0)} \times t^{(0)}}$, where $t^{(0)}$ is the word duration and each column of v is a $n^{(0)}$-dimensional feature computed at an instance in time. By a sequence of convolution and pooling steps, our model must yield a fixed-length output representation $h(v)$.

A filter response, $C^{(1)} \in \mathbb{R}^{n^{(1)} \times t_C^{(1)}}$ is computed by convolving $n^{(1)}$ learned filters with the input signal, and applying a nonlinear activation function to the output.

$$C_i^{(1)} = \sigma(W_i^{(1)} \star v + b_i^{(1)}) \tag{1}$$

where each $W_i^{(1)} \in \mathbb{R}^{n^{(0)} \times \omega_c^{(1)}}$ is a linear convolutional filter, $b_i^{(1)}$ is a scalar bias term, and $\sigma(\cdot)$ a nonlinear activation function. The number of filters $n^{(1)}$ and the width of each filter $\omega_C^{(1)}$ are both free parameters of the architecture. Each column of the resulting activation matrix $C^{(1)}$ thus represents $n^{(1)}$ filter responses at a particular time instant. The number of columns, $t_C^{(1)}$ is determined by a "valid" convolution ($\star$ operator) and the length of the original signal, $t^{(0)}$. Namely:

$$t_C^{(1)} = t^{(0)} - \omega_C^{(1)} + 1 \tag{2}$$

The literature explores several options for the nonlinear function given by $\sigma(\cdot)$, most commonly the logistic function. In our work however we use the *soft sign* function $\sigma(x) = \frac{x}{1+|x|}$ (Glorot & Bengio, 2010). This function has been shown to be more amenable to backpropagation in deep architectures because it reduces gradient saturation problems.

Our architecture next performs a local pooling operation over time to add robustness to minor temporal shifts. This layer transforms the representation $C^{(1)}$ into a pooled representation $P^{(1)}$ by segmenting the response of each filter into non-overlapping temporal regions r_k of width $\omega_P^{(1)}$ and computing a statistic on each. A common choice of statistic is *mean pooling*:

$$P_{i,k}^{(1)} = \frac{1}{|r_k|} \sum_{j \in r_k} C_{i,j}^{(1)} \tag{3}$$

taking a local average over each region. Another common variant is *max pooling*:

$$P_{i,k}^{(1)} = \max_{j \in r_k} |C_{i,j}^{(1)}| \tag{4}$$

which has been shown to work well when applying convolutional networks to images.

If $\omega_P^{(1)}$ is fixed, the length of our new representation is given by:

$$t^{(1)} = \lceil t_C^{(1)} / \omega_P^{(1)} \rceil \tag{5}$$

The resulting representation $P^{(1)}$ is an $n^{(1)}$-dimensional representation of the audio at each instant in time, and there are an unknown number of time instances. Thus we can generalize equations 1...5 to create a second convolution response $C^{(2)}$ and pooling layer $P^{(2)}$, treating $P^{(1)}$ as an input signal. This convolution and pooling process can be extended to an arbitrary number of layers, forming a *deep* convolutional network and adding to its expressive power.

Thus far, the network described transforms a variable-length input into a variable-length feature representation, whose length scales linearly with that of its input. However, the function we wish to ultimately encode, $h(v)$, must yield a fixed-dimensional representation z which does not depend on word duration. To capture this, we treat $P^{(N)}$ as a special case. Unlike the lower layers, where the region sizes $\omega_P^{(1)}, \cdots, \omega_P^{(N-1)}$ were fixed hyperparameters, we let $\omega_P^{(N)}$ scale with the size of its input.

$$\omega_P^{(N)} = \lceil t_C^{(N)} / K_P^{(N)} \rceil \tag{6}$$

where $K_P^{(N)}$, the number of top-level pooling regions, is a free parameter of our network. $P^{(N)}$, as given by equations 3 or 4, is of fixed size $n^{(N)} \times K_P^{(N)}$ regardless of the duration of the input signal. Flattening $P^{(N)}$ yields a vector z of dimensionality $n^{(N)} * K_P^{(N)}$, the number of hidden units in the final convolutional layer multiplied by the number of top-level pooling regions. These layers thus define $h(v)$. A summary of its parameters are given in Table 2.1.

	Scope[ℓ]	Description
$n^{(\ell)}$	1...N	# filters at $C^{(\ell)}$
$\omega_C^{(\ell)}$	1...N	Width of filters at $C^{(\ell)}$
$\omega_P^{(\ell)}$	1...$N-1$	Width of pool regions at $P^{(\ell)}$
$K_P^{(\ell)}$	N	# pool regions at $P^{(N)}$

Table 1. A list of free parameters in the $h(v)$ network

2.2. Vector Regression

The hidden layers of the architecture provide the vector z, fixed-dimensional representation of the acoustic input v obtained by a parametrized nonlinear transformation.

We assume each word, w, in the vocabulary has an associated real-valued *word vector*, $\phi^{(w)}$. These vectors can represent acoustic, syntactic, and semantic information conveyed by words. There are numerous ways to learn such vectors depending on the application in which the vectors are to be used (Turney & Pantel, 2010; Bengio et al., 2003). For our purposes the vector representations can come from any sort of learning procedure. The important property of using word vectors is the ability to handle an arbitrarily large vocabulary with a fixed number of model parameters. The dimensionality d of the word vectors is a free parameter of the architecture. Reasonable choices for d depend on the size of the vocabulary and algorithm used to generate the word vectors. In practice vocabularies on the order of hundreds of thousands of words can be meaningfully modeled with d as small as 50 or 200. Thus regressing to word vectors vastly reduces the model complexity as compared to learning a parametric classifier for each word in the vocabulary.

We use a linear regression layer R to predict a word vector $\hat{\phi}$ from an acoustic span v,

$$\hat{\phi} = W^{(R)} h(v) + b^{(R)}. \tag{7}$$

The affine mapping defined by $W^{(R)}$ and $b^{(R)}$ project the high-dimensional vector $h(v)$ into the word vector space. We use a simple affine mapping here because we want the network to capture rich transformations in the convolution and pooling layers rather than while doing the regression.

When training the model, we are given a dataset $\mathcal{D}$ containing tuples of the form (v, w) corresponding to the utterance acoustics and word identity respectively. The associated word vector $\phi^{(w)}$ is then taken from a lookup table, as the word representations are fixed in advance of network training. The loss function for the model is given by,

$$\sum_{(v,w)\in\mathcal{D}} ||W^{(R)} h(v) + b^{(R)} - \phi^{(w)}||^2 + \lambda ||\theta||^2. \tag{8}$$

This is a quadratic linear regression loss function with a weight norm penalty. The vector θ corresponds to all weight matrices in the convolution and regression layers, and λ is a scalar parameter controlling the strength of the weight penalty term. We optimize this loss function with respect to all network and regression parameters simultaneously.

3. Experiments

Mapping word-level acoustic information to word semantics or identity is one of the fundamental challenges in large vocabulary speech recognition systems. Existing methods apply a standard HMM approach to most speech tasks. Well-tuned HMM systems are the state-of-the-art approach to recognizing speech given raw acoustic information. These systems however rely on a Markov assumption at the frame level, meaning their associated acoustic models consider only very short temporal spans of the signal. In our experiments, we work with a second-pass speech system, SCARF, which naturally handles integrating information from word-level acoustics. Furthermore, SCARF reasons over n-best lattices generate from a high performance HMM system. This results in improvements to SCARF directly improving large vocabulary speech recognition benchmarks. We first analyze our model's capability as a word classification system, and then perform speech recognition experiments on the RT-03 broadcast news task.

3.1. Dataset

We train our model on 300 hours of English broadcast news audio from the TDT4 corpus. There are many speakers with relatively clean speech and little background noise. The dataset includes sentence-level human transcriptions, which does not immediately satisfy our need for acoustic spans labeled at the word level. We obtain word timing information for the transcribed text from the IBM Attila system (Soltau et al., 2010). n-best lattices derived for this system were released by (Zweig et al., 2011) for experiments in second-pass speech recognition. This results in over 2 million acoustic spans labeled by the word being said during each span.

To represent the acoustic signal we use log Mel-scale filterbank responses generated by HTK. Each frame of acoustic data is derived from 25 ms of audio, and the step size between successive frames is 10ms. We first apply PCA whitening to the input window to reduce its dimensionality. This substantially reduces the input dimension because successive frames of the acoustic signal are highly correlated.

The vectors ϕ used to represent each word form an embedding of the vocabulary in high-dimensional space. This space can fit different intuitions about word similarity, where similarity is encoded by distance between words' vector representations. Word vectors for speech are fundamentally different from word vectors for text alone because word similarity should depend on both the words' pronunciations and word meaning or syntax. For example, in a word vector space built purely from acoustics similar sounding words such as 'grass' and 'grease' could be easily confused because their vector distance is small. In contrast, a vector space which encodes word semantics while ignoring acoustics will have 'grass' and 'grease' much further away. Such a vector space is potentially more useful in recognizing word and phrase variations, such as when using a dialog system to search for songs or movies. However, the difficulty of regressing from acoustics alone to word vectors depends on acoustic sensitivity in the word vector space.

Because we are not aware of a method to learn word vectors which are sensitive to both word acoustics and semantics, we randomly generate unit-norm vectors to use as word representations for this work. In preliminary experiments we found randomly generated vectors perform better than those derived from a neural language model (Turian et al., 2010). We hypothesize this is due to the importance of not confusing frequently occurring function words for speech recognition – such words are relatively close together in syntactically-focused vector spaces. We additionally performed initial experiments using word vectors derived from a pronunciation dictionary wherein words are represented by bag of phoneme vectors. This more acoustically-grounded representation performed only slightly better than random vectors and is thus not used in our final set of experiments for simplicity.

3.2. Network Training

There are several free parameters in the network architecture including the number of layers, window size of convolution filters, and pooling region sizes. We explore one setting for these parameters which limits the architecture size and depth for computational reasons. We train a network with 200 first layer filters, each filter looks at 11 consecutive input frames. Filter responses are passed through the soft sign nonlinearity, and then mean-pooled (see equation 3) using the final pooling layer. The final pooling layer computes an average over 4 temporal regions on the filter responses, resulting in a $4 * 200 = 800$ dimensional representation of the input. The linear regression layer projects this 800-dimensional representation to a 50-dimensional word vector.

Training a convolutional network on hundreds of hours of acoustic data poses a difficult optimization problem. We use mini-batch L-BFGS optimization, which as shown recent success as a technique for training deep neural network architectures (Le et al., 2011a). We use the L-BFGS algorithm as implemented by the minFunc package [1].

3.3. Word Classification

To evaluate the trained network in isolation, we analyze its performance at the task of classifying words. For classification evaluation, we consider test examples from only the 10,000 most frequent words from the training set rather than the full vocabulary. Given input acoustics, our model predicts a word vector $\hat{\phi}$. The classifier output is then the nearest word vector ϕ_w from the set of 10,000 possible words. This nearest neighbor approach allows for such a large vocabulary as compared to a parametric classifier attempting to learn separate parameters for each of the 10,000 words.

Classification performance is evaluated using 10 hours of audio withheld from the training set, corresponding to 87,859 word instances. Word occurrence in speech, like text, follows a power law distribution. Because of this, a baseline classifiers that always choose the most frequent word, or sample from the prior over word occurrence can perform surprisingly well in terms of accuracy. We use these approaches as baselines when evaluating the accuracy of our model. Table 2 shows

[1]http://www.di.ens.fr/ mschmidt/Software/minFunc.html

Classifier	100	10,000
Prior	3.8	0.8
Majority Class	13.1	6.2
Our Method	46.0	20.6

Table 2. Word classification accuracy (%). Classifiers are tested using both the 100 and 10,000 most frequent words from the training set as possible outputs. Our model predicts a vector and outputs a class prediction by choosing the nearest word's vector in Euclidean distance. We include baselines of randomly choosing classes from the empirical distribution over words (Prior), and always choosing the most frequently occurring word (Majority Class).

the accuracy result.

As can be seen, the same model is able to achieve both 46.0% accuracy in a 100-way classification task and 20.6% accuracy in a 10,000-way classification task. This suggests that it is indeed possible to learn a transformation from an acoustic signal into a meaningful word-vector representation, where similarity in feature-space correlates with word similarity. As the significantly under-performing baselines show, this is not simply learning from the prior on word distributions: the output vector $\phi^{(w)}$ carries meaningful information about the input signal v. Even when the number of candidate words it must discriminate against is extremely high, this discriminative power is not lost.

4. Large Vocabulary Continuous Speech Recognition

Word classification is one example of potential applications for our word-level acoustic model. Perhaps the best known and most challenging application for such models is large vocabulary continuous speech recognition (LVCSR). We integrate our model with the SCARF second-pass recognition system to evaluate whether our acoustic model can improve upon an already high-quality speech system.

The standard approach to LVCSR uses hidden Markov models (HMMs) with Gaussian mixture model acoustic models. This basic system has been refined over the past 25 years, resulting in extremely complicated training recipes to achieve state-of-the-art recognition performance (Gales & Young, 2007). Improvements to HMM systems include vocal tract length normalization (Wegmann et al., 1996), minimum phone error training (Povey & Woodland, 2002), feature space maximum likelihood linear regression (Gales, 1998), and boosted maximum mutual information training (Povey et al., 2008). Each of these improvements require a substantial amount of innovation and typically results in less than a 1% absolute improvement on word error rate (WER) of the final speech system. Furthermore, such improvements require a high level of domain expertise, resulting in a community of speech researchers somewhat isolated from the larger machine learning community. Our approach uses machine learning methods without specialized domain knowledge. Improving LVCSR systems using this approach to research has perhaps larger potential as we can more easily include new machine learning ideas to our methods than can be done by working directly with already complex HMM systems.

4.1. SCARF Recognizer

The SCARF speech recognition system of (Zweig & Nguyen, 2010) offers the ability to achieve and improve upon state-of-the-art LVCSR results without the massive engineering effort involved in working with HMM systems directly. SCARF uses segmental conditional random fields to reason about recognition at the word level. It reasons over possible segmentations of the speech signal in time, and the word label of each segment. Specifically, SCARF computes the probability $P(\mathbf{w}|v)$ of a word sequence $\mathbf{w}$ given an acoustic signal v as,

$$\frac{\sum_{\mathbf{q}\in Seg(\mathbf{w},s)} \exp(\sum_{q\in\mathbf{q},k} \alpha_k f_k(w(q), v(q)))}{\sum_{\mathbf{w}'} \sum_{\mathbf{q}\in Seg(\mathbf{w}',s)} \exp(\sum_{q\in\mathbf{q},k} \alpha_k f_k(w(q), v(q)))} \quad (9)$$

For a particular utterance v and transcription hypothesis $\mathbf{w}$, SCARF reasons over all possible $|\mathbf{w}|$-segmentations of the acoustic input. Each segment q in a segmentation $\mathbf{q}$ is a portion of the acoustic signal assigned to a particular word $w \in \mathbf{w}$. Features in SCARF $f_k(w(q), v(q)))$ are a function of both the word $w(q)$ and acoustics $v(q)$ of a particular segment q. For example, a unigram language model feature assigns its feature value based solely on the word hypothesis for the segment $w(q)$ and ignores the acoustics.

Reasoning over all possible utterance transcriptions and segmentations is not tractable, so SCARF relies on a *lattice* to restrict the search space. A lattice is defines a directed graph over time where vertices correspond to start/end times of words and edges to a particular word, figure 2 shows an example. The lattice compactly represents possible transcriptions for the entire time span of the utterance. We use lattices for the 300 hour broadcast news corpus described in section 3.1. The lattices were generated by the IBM Attila HMM recognizer (Soltau et al., 2010) and released publicly by Zweig et al. (Zweig et al., 2011). These lattices correspond to an n-best decoding from the system, but in our experiment we do not provide SCARF with information about what the original recognizer chose as the most likely answer.

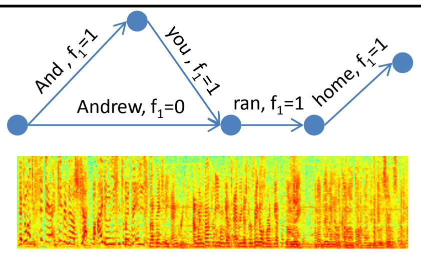

Figure 2. Example word lattice for an utterance. The lattice compactly represents possible segmentations of the utterance along with their transcriptions. We predict a word vector $\hat{\phi}$ for each segment (edge) and compute the distance from to the actual word vector $\phi^{(w)}$ for the segment. A binary feature f_1 is then produced to indicate which edge in an overlapping time span has minimum distance.

SCARF is trained on the lattices using a single feature – a unigram language model. We chose the simple language model and absence of other features to focus solely on the contribution of our novel word-level acoustic model to system performance. Word error rate (WER) is then evaluated on the RT-03 dataset using the Sclite tool. The resulting WER is 20.1% which is, as anticipated, high relative to the 15.6% performance achieved when using the 1-best decoding from Attila as a feature for SCARF.

4.2. Minimum Distance Feature

Our model predicts a word vector $\hat{\phi}^q$ for an acoustic segment q using only the acoustic information $v(q)$. When using SCARF, we also have a word label $w(q)$. We then compute the Euclidean distance $d = ||\hat{\phi}^q - \phi^{w(q)}||$ between the predicted word vector and the word vector corresponding to the word segment label.

After computing distances for all segments in the lattice, we convert them to a binary feature. At a particular point in time there is some *confusion set* C of possible segments that overlap at that time point. In figure 2 the segments ‘And’ and ‘Andrew’ form a confusion set, and the segments ‘you’ and ‘Andrew’ form a separate confusion set. Our final feature function is given by,

$$f_1(w(q), v(q)) = \begin{cases} 1 & \min_{q \in C} ||\hat{\phi}^q - \phi^{w(q)}|| \\ 0 & \text{otherwise} \end{cases}. \quad (10)$$

So any segment that is the minimum distance prediction for a confusion set in the lattice will have feature f_1 active. Figure 2 shows a valid setting of f_1 based on the confusion sets present in the lattice.

4.3. Results

We train SCARF using our minimum distance feature, as well as the same unigram language model feature described in section 4.1. Table 3 shows the resulting WER for our model and the baseline, as well as the performance possible in SCARF when including the 1-best hypothesis from the Attila system as a feature.

Model	Word Error Rate (%)
SCARF baseline	20.1
SCARF w/ Min Dist	19.8
Attila HMM	15.6

Table 3. Speech Recognition Performance. Adding our minimum distance feature improves upon the baseline system, but does not outperform using output of the highly engineered Attila recognizer is as a feature.

Our model achieves a 1.5% relative improvement over the baseline system – a reasonable gain for this challenging LVCSR task. Such a relative improvement is on the order of what is attained by adding yet another training step or feature modification to HMM systems like Attila, but our approach does not rely on domain-specific knowledge. This demonstrates that using convolutional networks to reason over entire words can benefit complete speech recognition systems.

5. Conclusion

We introduced a convolutional network architecture that projects the acoustics of an entire word to a word vector space. This general framework enables the model to handle large vocabularies via regression in place of parametric classification. We demonstrated the architecture’s ability to accurately classify words in a 10,000-way problem. Further, when integrated into a large vocabulary speech recognizer, our model provides improvement over a baseline system. This demonstrates the potential for word-level deep learning approaches in the speech domain. Finally, the convolutional vector regression framework has many architecture parameters, and there are several types of word vector space we can choose. This offers many opportunities for future work in learning deep architectures to project word utterances into word vector spaces. Such models have application not only for speech recognition, but also dialog systems like voice search, and exploring vector spaces to capture both acoustic and semantic similarity.

Acknowledgments

The authors would like to thank the anonymous reviewers for helpful comments. AM was supported as an NSF IGERT Traineeship Recipient under Award 0801700. SM is supported by the Hertz Foundation Google Fellowship and the Stanford Graduate Fellowship.

References

Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. A Neural Probabilistic Language Model. *Journal of Machine Learning Research*, 3(6):1137–1155, 2003. ISSN 1532-4435.

Chopra, S., Haffner, P., and Dimitriadis, D. Combining Frame and Segment Level Processing via Temporal Pooling for Phonetic Classification. In *12th Annual Conference of the International Speech Communication Association*, 2011.

Collobert, R. and Weston, J. A unified architecture for natural language processing. *ICML*, pp. 160–167, 2008.

Dahl, G., Yu, D., Deng, L., and Acero, A. Context-Dependent Pre-trained Deep Neural Networks for Large Vocabulary Speech Recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, pp. 1–13, 2011.

Dietterich, T. and Bakiri, G. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2(19): 263–286, 1995.

Gales, M. and Young, S. The Application of Hidden Markov Models in Speech Recognition. *Foundations and Trends in Signal Processing*, 1(3):195–304, 2007. ISSN 1932-8346.

Gales, M. Maximum likelihood linear transformations for HMM-based speech recognition. *Computer speech and language*, 12:75–98, 1998.

Glorot, X. and Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, volume 9, pp. 249–256, 2010.

Le, Q. V., Coates, A., Prochnow, B., and Ng, A. Y. On Optimization Methods for Deep Learning. In *ICML*, 2011a.

Le, Q. V., Zou, W. Y., Yeung, S. Y., and Ng, A. Y. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *CVPR*, pp. 3361–3368, 2011b.

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

Lee, H., Largman, Y., Pham, P., and Ng, A. Unsupervised feature learning for audio classification using convolutional deep belief networks. In *NIPS 22*, pp. 1096–1104. Citeseer, 2010.

Mnih, A. and Hinton, G. Three new graphical models for statistical language modelling. In *ICML*, pp. 641–648. ACM Press, 2007.

Mohamed, A.-r., Dahl, G., and Hinton, G. Acoustic Modeling using Deep Belief Networks. *IEEE Transactions on Audio, Speech, and Language Processing*, (99):1–1, 2011.

Povey, D. and Woodland, P. C. Minimum Phone Error and I-Smoothing for Improved Discriminative Training. In *ICASSP*, 2002.

Povey, D., Kanevsky, D., Kingsbury, B., Ramabhadran, B., Saon, G., and Visweswariah, K. Boosted MMI for model and feature-space discriminative training. In *ICASSP*, pp. 4057–4060, 2008.

Saxe, A. M., Chen, Z., and Ng, A. Y. On Random Weights and Unsupervised Feature Learning. In *ICML*, number 2009, 2011.

Soltau, H., Saon, G., and Kingsbury, B. The IBM Attila speech recognition toolkit. In *IEEE Spoken Language Technology Workshop (SLT)*, pp. 97–102. IEEE, 2010.

Turian, J., Ratinov, L., and Bengio, Y. Word representations: A simple and general method for semi-supervised learning. In *ACL*, pp. 384–394, 2010.

Turney, P. D. and Pantel, P. From Frequency to Meaning : Vector Space Models of Semantics. *Journal of Artificial Intelligence Research*, 37:141–188, 2010.

Vinyals, O. and Ravuri, S. Comparing multilayer perceptron to Deep Belief Network Tandem features for robust ASR. In *ICASSP*, pp. 2–5, 2011.

Wang, J., Yang, J., Yu, K., Lv, F., Huang, T., and Gong, Y. Locality-constrained linear coding for image classification. In *CVPR*, 2010.

Wegmann, S., McAllaster, D., Orloff, J., and Peskin, B. Speaker normalization on conversational telephone speech. In *ICASSP*, pp. 339–341, 1996.

Zweig, G., Nguyen, P., Van Compernolle, D., Demuynck, K., Atlas, L., Clark, P., Sell, G., Wang, M., Sha, F., Hermansky, H., Karakos, D., Jansen, A., Thomas, S., Sivaram, G., Bowman, S., and Kao, J. Speech Recognition with Segmental Conditional Random Fields: A Summary of the JHU CLSP 2010 Summer Workshop. In *ICASSP*, 2011.

Zweig, G. and Nguyen, P. SCARF: A segmental conditional random field toolkit for speech recognition. In *Interspeech*, 2010.