A new way to learn acoustic events

Navdeep Jaitly
Department of Computer Science, University of Toronto
6 Kings College Rd, Toronto, ON, Canada, M5S 3G4
ndjaitly@cs.toronto.edu

Geoffrey E. Hinton
Department of Computer Science, University of Toronto
6 Kings College Rd, Toronto, ON, Canada, M5S 3G4
hinton@cs.toronto.edu

Abstract

Most speech recognition systems still use Mel Frequency Cepstral Coefficients (MFCC's) or Perceptual Linear Prediction Coefficients because these preserve a lot of the information required for recognition while being much more compact than a high-resolution spectrogram. As computers get faster and methods of modeling high-dimensional data improve, however, high-resolution spectrograms or other very high-dimensional representations of the sound wave become more attractive. They have already surpassed MFCC's for some tasks [1]. Psychologists have argued that high-quality recognition would be facilitated by finding acoustic events or landmarks that have well-defined onset times, amplitudes and rates in addition to being present or absent. We introduce a new way of *learning* such acoustic events by using a new type of autoencoder that is given both a spectrogram and a desired global transformation and learns to output the transformed spectrogram. By specifying the global transformation in the appropriate way, we can force the autoencoder to extract accoustic events that, in addition to a probability of being present, have explicit onset times, amplitudes and rates. This makes it much easier to compute relationships between acoustic events.

1 Introduction

With the recent progress in deep learning, deep belief networks and de-noising autoencoders are now being widely used for automated feature discovery [2,3]. Deep belief nets learn a probabilistic generative model with stochastic latent variables that capture the statistical structure of the data [2]. Denoising autoencoders learn to predict uncorrupted data from corrupted data using a feed-forward neural network that may have a bottleneck layer with only a small number of dimensions [3]. Features discovered by these unsupervised methods have been used for classification tasks, either as fixed preprocessors or as initializations for feed-forward neural networks that are then discriminatively fine-tuned. Although these methods have achieved state-of-the-art results on several problems, we believe that much more useful features can be learned by making better use of prior knowledge about ways in which the data can be transformed without changing the class labels.

Domain-specific prior knowledge, such as geometry, can be explicitly built into the architecture of these models by using a convolutional architecture [4]. It can also be provided implicitly by augmenting the training data with copies of the data that have been transformed in ways that do not change the class information. For example, affine transformations of images do not usually alter the target labels, so many different affine transformations of each training image can be used to vastly expand the size of the training set. In both of these approaches, the assumption is that the use of ge-

ometric prior knowledge will lead to better features and manual explorations of the learned features often provide support for this idea - features typically pick up patterns with specfic characteristics. A feature, for example, may detect edges of a specfic orientation at a specific location.

A major limitation of a typical, learned feature detector is that it only provides information about the presence or absence of a feature. It does not provide explicit information about the precise location, orientation, scale, intensity and contrast of the feature. If the feature detector produces a scalar output, it can provide some information about these "instantiation parameters" but this information is all confounded in a single scalar. With enough feature detectors tuned to overlapping, large regions of the multi-dimensional space of instantiation parameters, it may be possible to implicitly capture the information about the values of the individual instantiation parameters [5]. However, this "coarse-coded" information is not in the right form to make it easy to represent relationships between features.

Consider, for example, the task of checking whether two line segments form a cross. If we have explicit instantiation parameters, we can simply check that the locations are the same and the orientations differ by about 90°. This simple procedure works for all positions and orientations of the line segments. By contrast, if we use detectors for line segments that are specific to a particular position and orientation we need to learn all of the different pairings that form a cross. There is no simple way to check that the parts are in the right spatial relationship independent of the pose of the cross.

A recent paper, [6], introduces a simple trick, called a "transforming autoencoder" that makes it possible to train a group of neurons called a "capsule" to both recognize when a visual entity is present and to provide explicit information about the position, orientation and scale of the current instantiation of the visual entity. Using a transforming autoencoder, a layer of these capsules can be trained without having to specify what visual entity should be detected by each capsule and without having to specify the poses of the visual entities.

Once the lowest layer of capsules has been learned, the explicit representation of the poses of the visual entities makes it easy to use the relationships between the poses, rather than the poses themselves, to activate higher-level capsules. This is a much more efficient way of dealing with changes in viewpoint than replicating matched filters over all possible viewpoints or learning a similar solution by using transformations of the data to massively expand the training set.

The aim of this paper is to show that a similar approach can be used for extracting acoustic events from either the waveform or from a spectrogram. Unlike the features typically produced by neural networks, a detected acoustic event will have an explicit representation of its exact time of occurence, its exact amplitude, and (sometimes) its exact rate in addition to the probability that it is present. These explicit instantiation parameters should make it easy to detect more complex acoustic events as relationships between more primitive events, but in this paper we focus on simply learning the first level of acoustic events.

The idea of acoustic events has a long history in the psychology literature which suggests that the auditory recognition process involves acoustic landmarks which trigger events in the auditory cortex [7-9] and that these events are further grouped together in a hierarchical process [10]. In his position paper Kenneth Stevens draws upon the generative articulatory process to propose a computational model for recognition in which auditory cues in the vicinity of auditory landmarks such as energy bursts are grouped into features. Features are then organized into segmental units which are grouped into words. Recognition then proceeds by mapping observed data into a dictionary of patterns associated with syllables in words in a lexicon. Jansen recently provided a concrete implementation of the event-based approach to word detection [11]. Sparse streams of phone detector events are combined to detect word occurrences. The phone detectors used were discriminatively trained multilayer perceptrons trained on acoustic signals, and only contain information about the presence or absence of events. Our aim in this paper is to use transforming autoencoders to learn acoustic events with explicit instantation parameters in an *unsupervised* manner.

By training a layer of capsules on speech spectrograms we show that we can discover meaningful acoustic events that could serve as landmarks in the recognition process. We postulate that subsequent layers of capsules could combine the primitive acoustic events to form a hierarchy but we have not yet done this. Instead, we demonstrate that the outputs of the first layer of capsules contain a lot of useful information by using them as input to an existing type of speech recognition system which then achieves an accuracy that is close to the state-of-the-art.

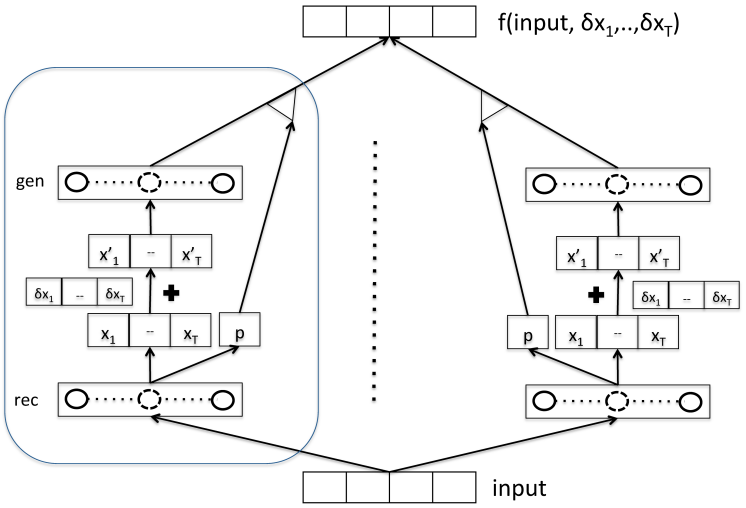

Figure 1: The architecture of a transforming autoencoder. The input is provided to a set of capsules, each of which uses a layer of logistic "recognition" units to compute a probability that its entity is present and to estimate the instantiation parameters that describe exactly how this instance of the entity is instantiated. The values of the instantiation parameters of each capsule are adjusted to take into account the global transformation that relates the input to the target output and the adjusted values are fed into a layer of logistic "generation" units which compute the contribution of the capsule to the output data. This contribution is then gated by the probability of activation of the capsule.

2 Transforming Autoencoders

Here we briefly describe the idea of transforming autoencoders. The reader is referred to [6] for further details.

A standard autoencoder receives a data-vector that is used as both the input to a feed-forward neural network and as the target output. The autoencoder learns to encode the information about the input vector into the activities of a small number of hidden units in its central "bottleneck" layer and to decode the activites of the bottleneck units to produce an accurate reconstruction of its input at the output layer. By using several hidden layers and non-linear transformations at each layer, an autoencoder can do significantly better than principal component analysis at reconstructing its input vector [12]. There is, however, no easy way to control what the units in the bottleneck layer learn to extract from the input. In particular, there is no way to force the particular way in which an entity is instantiated in the current input vector to be represented separately from the information about its existence. The inability to control the form of the representation that is learned leads some researchers to criticize neural networks as uninterpretable "black boxes".

A transforming autoencoder (see figure 1) receives both an input vector and a desired output vector that is related to the input vector by some simple global transformation such as a small translation of a whole image. It also receives an explicit representation of the global transformation. In the case of a pure translation this would be simply the Δx and Δy. The bottleneck layer of the transforming autoencoder consists of the outputs of a number of capsules. In the case of a pure translation, each capsule would output the probability that its implicitly defined visual entity is present and the real-valued x and y coordinates of that entity relative to an origin that is implictly defined by the capsule. The information about the global transformation is then injected by incrementing the x

and y outputs of every capsule by Δx and Δy [1]. The transforming autoencoder must then construct the transformed image from the transformed capsule outputs. All of the weights in a transforming autoencoder are learned by backpropagating the derivatives of the misfit between the actual and desired outputs. During the training phase, the different capsules learn to extract different entities or similar entities in different parts of the space of instantiation parameters in order to minimize the misfit between the final output and the target. It would also be straightforward to share recognition and generation units between capsules

By explicitly injecting Δx and Δy, we encourage the transforming autoencoder to use the x and y outputs of each capsule to represent the real x and y coordinates of some visual entity. If it uses its x and y outputs in this way, it automatically produces the correct output image when the desired output image is held constant but the input image is translated by n pixels in the x direction and the Δx input is reduced by n.

One limitation of using capsules that output a set of explicit instantiation parameters is that each capsule can only deal with zero or one instance of its visual entity at a time. So for entities that are densely distributed the capsules need to operate over small domains in which there is generally not more than one instance at a time. For more complex, rarer entities the domains can be much bigger, so in a system with multiple layers of capsules we would expect the higher-level capsules to have much bigger domains than the lower-level ones. This does not mean that the higher levels have lost the information about the instantiation parameters (as they do in a multilayer convolutional net with subsampling). Capsules with large domains can encode information about the precise position of their visual entity just as accurately as capsules with small domains.

Until now, transforming autoencoders have only been applied to images, but they are applicable to any domain in which we know how to apply global transformations to the data. Speech is an obvious candidate.

2.1 A Way to Improve Transforming Autoencoders

We can further encourage a transforming autoencoder to use its capsule outputs in the desired way by using the transformed data as an alternative *input*. The idea is that the instantiation parameters extracted from the original and the transformed inputs should differ by exactly the amount specified by the global transformation.

We explicitly enforced this property by adding the following regularization term for each type of transformation.

$$\sum_{i=1}^{N_c} p_i\left(\mathbf{v}\right)\left\{C_i\left(\mathbf{v}\right)+\partial c-C_i\left(\mathbf{g}\left(\mathbf{v},\partial c\right)\right)\right\}^2 \tag{1}$$

where N_c is the number of capsules, $\mathbf{g}\left(\mathbf{v},\partial c\right)$ is the result of applying transformation $\mathbf{g}\left(\cdot,\cdot\right)$ to vector $\mathbf{v}$ using parameter ∂c, $p_i\left(\mathbf{v}\right)$ and $C_i\left(\mathbf{v}\right)$ are the probability and the output respectively, of capsule i for data $\mathbf{v}$. This regularization term requires the capsule values to be equivariant to transformations of the data. i.e. if a transformation of ∂c was applied to the input data, the instantiation parameter computed by each capsule must also transform by ∂c if $p_i\left(\mathbf{v}\right)$ is significant. The regularization is used only to adjust the parameters related to the computation of the instantiation parameters, not to the computation of the capsule probabilities. Doing the latter resulted in much poorer models where equivariance of the generative features was strongly limited.

2.2 Sampling of Capsules

Since our aim is to use capsules to capture acoustic events, we wanted to train a stochastic transforming autoencoder in which the capsules were either active or inactive for a particular input. Such binary activities should prevent cooperation between different capsules in the creation of the reconstruction that may be possible with capsule activities that are more real valued. To achieve this, the capsule activation states were sampled from their logistic probabilities. If the capsules were active, their outputs were added to the output of the transforming autoencoder, otherwise, their outputs were ignored. The error function being optimized was the expected error of reconstruction. While it is

[1] In this paper, the global transformations of the instantiation parameters were randomly generated for every data case.

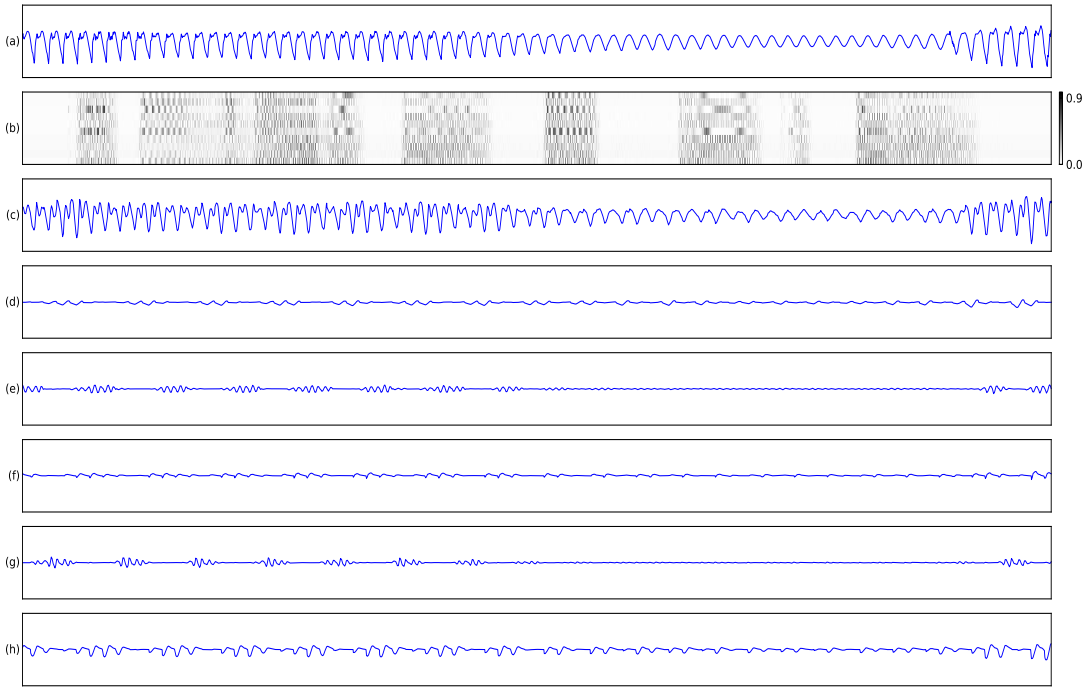

Figure 2: (a) 312.5 ms subsegment of an utterance. (b) Probabilities of activations of capsules. (c) Reconstruction of the utterance from the capsules (logistic outputs). (d,e,f, g, h) Contributions of five different capsules to the reconstruction.

simple to integrate out the states of the capsules in computing this error function and to compute the gradient with respect to the exact value, this involves terms that include all combinations of pairs of capsule states. So, for this paper, we simply estimated this gradient using a single Monte Carlo sample for each data point. Thus, for the capsules whose sampled states were on, the gradients from the output error were backpropagated through the capsules, otherwise they were not. In addition, for the recognition part of the autoencoder, gradients were also backpropagated to account for the change in probabilities of the states. We found that the stochastic transforming autoencoder could only be trained after using a deterministic transforming autoencoder with real-valued activities to initialize the weights. This is probably because the estimate of the gradient is very noisy when a single Monte Carlo sample of the activations is used early in the training when the activation probabilities are mostly near 0.5. Once the reconstruction error was low enough and the capsule activations had less entropy, the estimated gradient became less noisy. Even so, in order to facilitate the learning of the stochastic transforming autoencoder we used a very small learning rate of .001 without any momentum.

3 Experiments

Here we describe how the method outlined above was applied to learn features for both waveforms and spectrograms. For this study we used the Arctic database [2] to train the autoencoder on waveforms and the TIMIT corpus [3] as the source of speech signals for the autoencoder on spectrograms. We used the Arctic database for the former experiment because it contains a large number (=1132) of utterances recorded for an individual speaker and these utterances were recorded under strict recording conditions. This makes the task of learning features from waveforms more feasible. For the second task we used TIMIT database because we wanted to apply the features learnt to the task of phone recognition.

[2] http://www.festvox.org/cmu_arctic/index.html

[3] http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC93S1

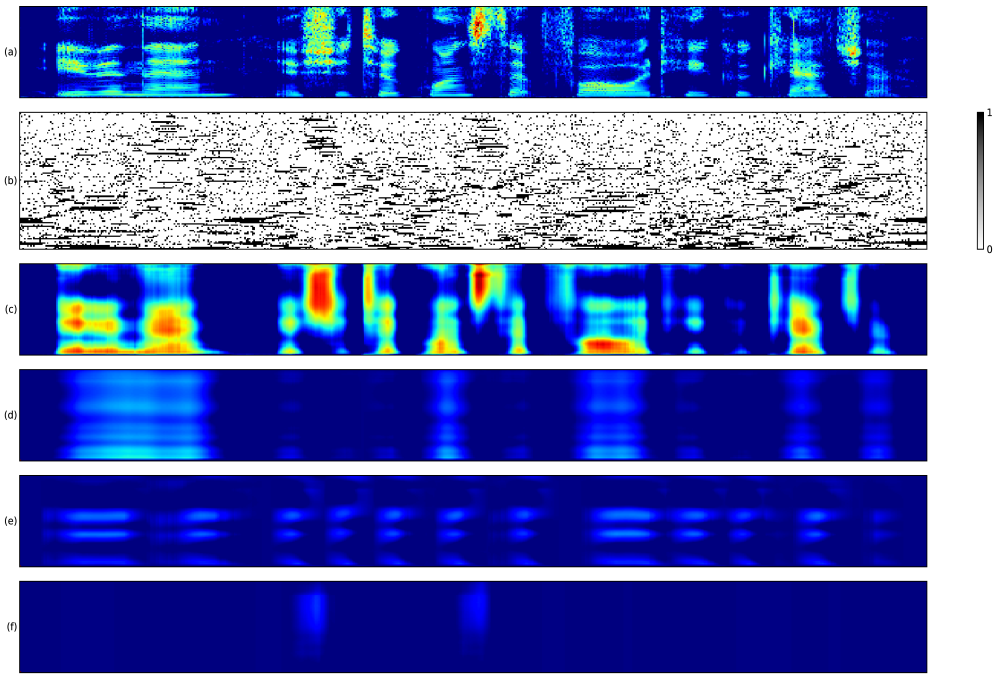

Figure 3: (a) Spectrogram for the utterance: 'Even then, if she took one step forward he could catch her.' (b) States of capsules over the utterance. (c) Reconstruction of the utterance from the binary states, and the values of auxillary variables of capsules. (d,e,f) Contributions of three different capsules to the reconstruction. It can be seen that the first capsule contributes overall energy over duration of several phonemes, probably over an entire syllable. The second capsule contributes principally to three different frequencies, probably three different formant frequencies. The third capsule contributes to fricatives /sh/ and /s/.

3.1 Waveforms

The entire set of waveforms for the Arctic database was first normalized so that the samples had a mean of 0 and a standard deviation of 1. The inputs were random segments of speech of duration 8 ms (=128 samples as the data was recorded at a sampling rate of 16KHz). The transforming encoder learnt to predict the central 4 ms (=64 samples) for each frame. The following transformations were applied to the data:

- **Translations:** The input speech signals were shifted left or right randomly by up to 32 samples. The value of the shift was added to the instantiation parameter corresponding to the temporal location of each capsule.
- **Amplitude Rescaling:** The intensities of the waveform samples were multiplied a random factor between .6 and 1.4. The instantiation parameters corresponding to intensity were multiplied by this scaling factor.

3.2 Spectrograms

Speech signals were converted into spectrograms by the following procedure. Raw signals were transformed into speech spectrograms of duration 10 ms and were advanced by 5 ms per frame. The speech spectrograms were log transformed[4]. 24 such consecutive frames (representing 10 + (24-1)*5 = 125 ms of waveform) were presented as input to the transforming autoencoder. The transforming autoencoder is required to estimate the central 18 frames of the data after transformation (leaving 3 frames out on each side). The following three transformations were applied:

[4]Unless mentioned otherwise, all the spectrograms were log transformed

- **Translations:** The input frames of the spectrogram were shifted left or right randomly by up to three frames.
- **Scaling:** The intensities of the spectrogram (before log transform) were scaled up or down by a random factor between .1 and 10. This was achieved by generating a random number between log(.1) and log(10) and adding it to the log spectrogram.
- **Frequency Resampling:** The log spectrograms were resampled in the frequency domain by a random rate between 0.8 and 1.2. The frequency resampling was achieved by first performing a cubic spline interpolation of each frame of the log spectrogram as a function of frequency, and then interpolating at integer multiples of the resampling rate.

Table 1: Reconstruction error of transforming autoencoder trained on spectrograms over different configurations

# of units	# of capsules	Binary Units	Reconstruction Error
100	50	No	.082
100	100	No	.078
100	150	No	.078
100	200	No	.079
50	50	No	.086
100	50	No	.082
150	50	No	.081
100	50	Yes	.32
100	100	Yes	.21
100	150	Yes	.18
100	200	Yes	.17

4 Results and Discussion

Figure 2 shows a segment of waveform, its reconstruction from the transforming autoencoder (without any transformations) and the contributions of the 5 most active capsules to the reconstructions. These capsule contributions resemble the gamma-tone features that have been discovered by previous methods such as ICA [13, 14] and Stepped Sigmoidal Unit RBMs [15]. [14] also notes that the gamma-tones discovered from their method are similar to the impulse responses of auditory nerves. Unlike-ICA like methods, where a sparse, fat tailed independent prior requires iterative methods for optimization, the optimization of capsules is accomplished by simple back-propagation requiring no optimization over posterior distributions of latent variables. Moreover, once learning is complete, no iteration is required to compute the representation of an input vector.

Figure 3 shows the results for reconstruction of a spectrogram from a transforming autoencoder.

4.1 Peering into the mind of the Capsules

One of the most powerful aspects of capsules is the ability to inspect what they represent. This can be done by inspecting how the contribution of a capsule to the final output changes as we modify the values of its instantiation parameters. These 'fantasy' features represent the manifold over which different capsules contribute to output signals. Figure 4 shows how the output of one capsule trained on spectrograms changes as the frequency resampling and location instantiation parameters are changed over a grid of values. Also shown is how the output changes in response to the changing value of the instantiation parameter corresponding to log(intensity). A similar trend was obtained for fantasies of capsules trained on waveforms.

4.2 Reconstruction Error

In order to explore the effect of different parameters on reconstruction error, we trained a transforming autoencoder with different numbers of capsules and hidden units [5]. In addition we also explore

[5]For the sake of simplicity, the number of recognition units was fixed equal to the number of generation units in this study, although experiments indicate that more generation units result in better reconstructions

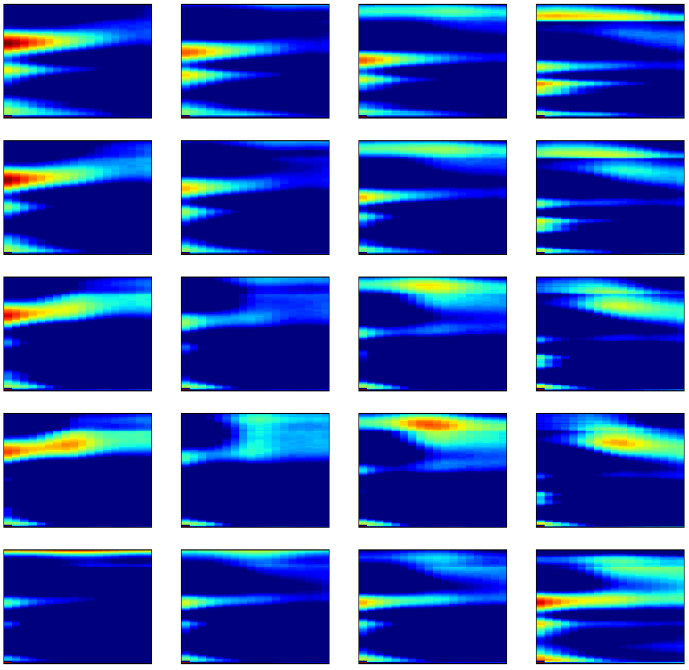

Figure 4: (a) Top 4x4 images: Left to right: Outputs of a capsule as instantiation parameter for frequency is set to 0.8, 0.92, 1.05 and 1.2 respectively. Top to bottom: variation in outputs of capsules as time variable of capsules is set to -4, -1.33, 1.33 and 4 respectively. (b) Bottom row: Outputs of a capsule as the instantation parameter for intensity is set to log(.5), log(0.8), log(1.26) and log(2) respectively.

the effect of using binary features on the reconstruction error. Table 1 shows a summary of these results for reconstruction error over the range of transformations described in section 3.2. As would be expected, it was seen that a larger number of capsules and a larger number of hidden units resulted in the autoencoders producing results with lower reconstruction error. For binary capsules, the expected degradation was observed - since they are able to convey less information to the decoding process, binary transforming autoencoders have higher reconstruction errors.

4.3 Application to phone recognition

Features learnt by a transforming autoencoder with 100 capsules on spectrograms were used to train a two hidden layer neural network, with 2048 nodes in each layer, to predict phoneme labels of corresponding segments of speech on TIMIT. The predictions were decoded using a method similar to that described in [15]. A phoneme error rate (PER) of 22.7 was observed for the development set and 24.8 on the test set. These results are better than the results reported in [15] for a similar set up (2000 hidden nodes and similar encoding rate of speech), but not better than the state of the art reported in [1] or the best system reported in [15]. A more appropriate use of the instantation parameters of the capsules should result in a higher accuracy than our current results.

Conclusions and Future work

We have shown how a transforming autoencoder can be used to discover rich features in the form of capsules in both waveforms and in spectrograms. These capsules can be used to achieve good results on phone recognition tasks with little postprocessing. However, we expect that the use of these capsules in a hierarchical event-based system for speech recognition will result in further improvements because important constraints between instantiation parameters of different capsules can be learnt and used. Future work will aim at developing such a model based on capsules.

References

[1] Dahl, G., Ranzato, M., Mohamed, A. and Hinton, G. E. (2010) Phone Recognition with the Mean-Covariance Restricted Boltzmann Machine. *Advances in Neural Information Processing 23*:469-477

[2] Hinton, G. E., Osindero, S. and Teh, Y. (2006) A fast learning algorithm for deep belief nets. *Neural Computation* **18**:1527-1554.

[3] Vincent, P., Larochelle, H., Bengio, Y. and Manzagol, P. A. (2008) Extracting and Composing Robust Features with Denoising Autoencoders. *Proceedings of the Twenty-fifth International Conference on Machine Learning*:1096 - 1103

[4] LeCun, Y., Bottou, L., Bengio, Y. and Haffner., P. (1998) Gradient-based learning applied to document recognition. *Proceedings of the IEEE* **86**:22782324

[5] Hinton, G. E., McClelland, J. L., and Rumelhart, D. E. (1986) Distributed representations. In *Rumelhart, D. E. and McClelland, J. L., editors, Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 1: Foundations*:77-109. MA: MIT.

[6] Hinton, G. E., Krizhevsky, A. and Wang, S. (2011) Transforming Auto-encoders. *ICANN-11: International Conference on Artificial Neural Networks*; preprint (2011)

[7] Stevens, K.N. (1972) The quantal nature of speech: Evidence from Articulatory-Acoustic data. In *P. B. Denes and E. E. David Jr., editors, Human Communication: A Unified View*: 51-66. NY: McGraw-Hill.

[8] Chistovich, L. A. and Lublinskaya, V. V. (1979) The center of gravity effect in vowel spectra and critical distance between the formants: Psychoacoustical study of the perception of vowel-like stimuli. *Hearing Research* **1**:(3):185-195.

[9] Delgutte B. and Kiang, N. Y. (1984) Speech coding in the auditory nerve: IV. Sounds with consonant-like dynamic characteristics. *Journal of the Acoustical Society of America* **75**(3):897-907

[10] Clements, G.N. (1985) The geometry of phonological features. *Phonology* **2**:225-252.

[11] Jansen, A. (2011) Whole word discriminative point process models. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing 2011*; preprint(2011)

[12] Hinton, G. E. and Salakhutdinov, R. R. (2006) Reducing the dimensionality of data with neural networks. *Science* **313**(5786):504 - 507

[13] Lee, J., Jung, H., Lee, T., and Lee, S. (2000) Speech feature extraction using independent component analysis. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing 2000* **3**(3):1631-1634.

[14] Lewicki, M.S. (2002) Efficient coding of natural sounds. *Nature Neuroscience* **5**(4):356-363.

[15] Jaitly, N. and Hinton, G. E. (2011) Learning a better Representation of Speech Sound Waves using Restricted Boltzmann Machines. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing 2011*; preprint (2011)