Deep Unsupervised Learning from Speech

by

Jennifer Fox Drexler

B.S.E., Princeton University (2009)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2016

Signature redacted

Author ....
Department of Electrical Engineering and Computer Science
May 20, 2016

Signature redacted

Certified by.. ....
James R. Glass
Senior Research Scientist
Thesis Supervisor

Signature redacted

Accepted by... ....
Leslie Kolodziejski
Chair, Department Committee on Graduate Students

Deep Unsupervised Learning from Speech

by

Jennifer Fox Drexler

Submitted to the Department of Electrical Engineering and Computer Science
on May 20, 2016, in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering and Computer Science

Abstract

Automatic speech recognition (ASR) systems have become hugely successful in recent years - we have become accustomed to speech interfaces across all kinds of devices. However, despite the huge impact ASR has had on the way we interact with technology, it is out of reach for a significant portion of the world's population. This is because these systems rely on a variety of manually-generated resources - like transcripts and pronunciation dictionaries - that can be both expensive and difficult to acquire. In this thesis, we explore techniques for learning about speech directly from speech, with no manually generated transcriptions. Such techniques have the potential to revolutionize speech technologies for the vast majority of the world's population.

The cognitive science and computer science communities have both been investing increasing time and resources into exploring this problem. However, a full unsupervised speech recognition system is a hugely complicated undertaking and is still a long ways away. As in previous work, we focus on the lower-level tasks which will underlie an eventual unsupervised speech recognizer. We specifically focus on two tasks: developing linguistically meaningful representations of speech and segmenting speech into phonetic units.

This thesis approaches these tasks from a new direction: deep learning. While modern deep learning methods have their roots in ideas from the 1960s and even earlier, deep learning techniques have recently seen a resurgence, thanks to huge increases in computational power and new efficient learning algorithms. Deep learning algorithms have been instrumental in the recent progress of traditional supervised speech recognition; here, we extend that work to unsupervised learning from speech.

Thesis Supervisor: James R. Glass
Title: Senior Research Scientist

Acknowledgments

First and foremost, I would like to thank everyone involved with the Lincoln Scholars program at MIT Lincoln Laboratory, for giving me the time and space to pursue my own research. I am incredibly grateful to have been given this opportunity. Thanks especially to Wade, Cliff, Joe and Doug for all of their help and guidance.

To my advisor, Jim Glass, thank you for the fruitful discussions, helpful insights, and, above all, your patience.

To my parents, thank you for your love and support, and for just the right amount of prodding. To my friends and family, thank you for being there. To Stephen, you know what you did.

This work was sponsored by the Department of Defense under Air Force Contract FA8721-05-C-0002. Opinions, interpretations, conclusions, and recommendations are those of the authors and are not necessarily endorsed by the United States Government.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Why Unsupervised Speech Recognition

Automatic speech recognition (ASR) systems have become hugely successful in recent years - we have become accustomed to speech interfaces across all kinds of devices. However, despite the huge impact ASR has had on the way we interact with technology, it is out of reach for a significant portion of the world's population. This is because these systems rely on a variety of manually-generated resources - like transcripts and pronunciation dictionaries - that can be both expensive and difficult to acquire.

There are more than 7,000 human languages spoken across the globe, but only a small fraction of these are supported by current speech recognition technologies. Almost 20% of the world's population does not speak one of the top 100 languages[21]. Google voice search supports only 39 languages, the vast majority of which were added to the system in the first four years after its release in 2008. Because of the difficulty of collecting the necessary resources for a new language and the diminishing returns of developing a system for any particular language now that the big languages are covered, it is unlikely that systems will be developed for most languages using the current paradigm.

This type of technology is of significant interest to the US government, which has consistently invested in the development of tools for low-resource languages. The

Babel project[27] is looking into what we can do with very few resources. This has yielded promising results for some low-resource languages. However, this research fundamentally fits into the current paradigm, trying to adapt techniques developed for high-resource languages to low-resource ones. The LORELEI (low-resource languages for emergent incidents) program takes a broader view on the same problem, with one of its goals being to develop technologies that do not rely on manually-created resources[48].

In this work, rather than adapt current speech recognition technologies to the low-resource setting, we approach the problem from another direction. Given the landscape of current technologies, bringing speech technologies to underserved languages requires a rethinking of the traditional ASR architecture. Instead of investing time and energy in manually building resources for one language at a time, we can spend our time devising a system which can be trained with limited-to-no manually-created resources. Our goal is to survey new techniques that operate with no resources - that is, techniques which learn about speech directly from speech, with no manually generated transcriptions. We refer to this learning directly from speech as "unsupervised" learning because we do not have a supervisory signal from a human telling the system what it should learn. Instead, we explore the idea that the properties of speech we wish to learn can emerge from speech itself. We also explore a "weakly supervised" scenario in which we have access only to easily-acquired metadata: the identity of the speaker.

There is a natural analogue to this task: children, after all, learn language directly from speech. At the heart of this project is the question of whether traditional resources are really necessary for effective ASR: can computers learn language in a way that more closely resembles human language acquisition? The answer to this question has the potential to significantly improve ASR capabilities in the vast majority of the world's languages. This research also has the potential to inspire future research in cognitive science, as researchers try to develop plausible models of human language learning.

Current speech recognition technologies owe their impressive performance in part to the recent popularity of deep learning techniques. Deep learning has generated much interest for its impressive performance on a variety of machine learning tasks, most notably in the domain of computer vision, where researchers have explored their use in both supervised [38] and unsupervised [42] learning. In this thesis, we extend that work to the speech domain, where deep learning algorithms have been successfully used in supervised scenarios[28, 23, 26] but have not been fully explored for unsupervised learning.

1.2 Underlying Tasks

The cognitive science and computer science communities have both been investing increasing time and resources into exploring language acquisition from speech. However, a full unsupervised speech recognition system is a hugely complicated undertaking. As a result, much progress has been made towards solving the lower-level tasks necessary for such a system, but a complete system is still a long ways away. As in previous work, we focus on the lower-level tasks which will underlie an eventual unsupervised speech recognizer.

The individual tasks required for a complete system fall at every level in the language hierarchy: acoustics, phonology, morphology, vocabulary, semantics, and syntax. We are still at the lowest levels of this hierarchy, although we evaluate our progress partially by our ability to use what we learn at the lower levels for higher-level tasks.

The lowest level task is representing speech in a way that maintains the linguistic content but suppresses speaker and channel effects. We evaluate these representations by their usefulness in distinguishing same/different pairs of phones and in discovering repeated words or phrases in large corpora of speech.

A slightly-higher level task is to use such representations, in a still unsupervised way, to segment speech into meaningful units of linguistic content. In this thesis, we focus mainly on phoneme-level segmentation, but also look briefly at word segmentation as well.

Unsupervised sub-word modeling, also called unsupervised acoustic modeling, aims to find and model the set of sub-word units that make up a given language. This is related to, and often paired with, the segmentation task. For use in further downstream tasks, discovered segments must be clustered, across speakers and contexts, into their respective phoneme classes.

1.3 Approach

We're coming from a place of pure unsupervised learning - we have the speech and nothing else. While supervised learning is probably better known, unsupervised learning is well-studied in machine learning. In all machine learning, we have inputs, some system that generates outputs, and some way to compare the outputs to the desired outputs and then update the system to produce better output. In supervised learning, the desired outputs are a supervisory signal to the system.

In unsupervised learning, our desired outputs come from the data itself. Specifically, in this work, we look at two types of systems: autoencoders and predictors. In an autoencoder system, our desired output is to recreate the input. Such a system can be useful for generating a representation of the input, e.g. for the representation learning task. In a predictor system, our input is a sequence of items (in this case segments of speech) and our desired output is the next item in the sequence. We will look at how these systems can be useful for segmentation.

This thesis approaches unsupervised learning from speech from a new direction: deep learning. While modern deep learning methods have their roots in ideas from the 1960s and even earlier, deep learning techniques have recently seen a resurgence, thanks to huge increases in computational power and new efficient learning algorithms. Deep learning algorithms have been instrumental in the recent progress of traditional supervised speech recognition; here, we extend that work to unsupervised learning from speech.

Deep learning is a broad field encompassing many possible model structures. We look at two different DNN architectures: feed-forward networks and recurrent networks. The feed-forward networks require fixed-size inputs, while recurrent networks allow us to model variable-length sequences, and so are well suited to modeling larger units of speech. We can use a feed-forward model to represent a fixed-size segment of speech or predict an item given a fixed number of previous items (presented all at once). We can use a recurrent network get representations for variable-length segments of speech or to predict based on some variable-length context.

We explore the use of these deep neural networks for a range of tasks related to speech processing, including representation learning, segmentation, subword modeling, and term detection. All of these tasks require models to learn about the linguistic content of speech while ignoring speaker-related variations. To boost this learning, we deviate from the unsupervised learning paradigm by assuming access to speaker labels. These are relatively easy to acquire in most data collection scenarios and are also available to infants during language acquisition. Given the extreme difficulty of our tasks within the unsupervised learning scenario, it is imperative that we take advantage of whatever constraints we can impose on our learning. We show that making use of these speaker labels greatly improves the performance of these models across all evaluations performed here.

1.4 Thesis Contributions and Outline

This thesis is an exploration of the current research in unsupervised learning from speech. In addition to a broad survey of such work, this thesis is a first step towards understanding how deep learning can be applied to speech, with an eye to using our insights to develop better models.

This thesis is organized as follows. Chapters 2 and 3 present relevant background for this work; Chapter 2 on deep learning and Chapter 3 on speech processing. Chapter 4 gives details on the datasets, tools, and evaluation metrics used for this thesis.

Chapter 5 discusses feed-forward networks for generating feature representations of speech. We present both quantitative and qualitative analysis of learning in deep autoencoders. We also look at how this learning changes in two variants of the standard autoencoder framework, one in which we incorporate speaker information and one in which we take a probabilistic view of deep learning.

Chapter 6 explores recurrent networks for generating feature representations. First, we compare the abilities of feed-forward and recurrent networks in incorporating local context. Next, we investigate the impact of using variable-length context windows in recurrent autoencoders, both on their ability to accurately model speech and to produce useful representations for downstream tasks.

Chapter 7 focuses on predictor models, specifically their use in speech segmentation. We again compare the performance of feed-forward and recurrent models. Finally, Chapter 8 concludes with an overview of the contributions of this thesis and some ideas for future work.

Chapter 2

Background: Deep Learning

Neural networks are a general class of machine learning model composed of units, arranged in layers, and connected to each other by weights. The original inspiration for these networks comes from neuroscience (hence the 'neural' moniker), but they present an extremely oversimplified model of what goes on in the brain. Despite their simplicity, these networks have been hugely successful in recent years in a variety of fields. The resurgence in their popularity is owed both to the introduction of new algorithms that make training much easier and of newly accessible computer hardware that greatly accelerates their performance.

One of the first models to exploit this new hardware was released in 2010, when researchers achieved record performance on the MNIST dataset of handwritten digits using neural network algorithms that were, at that point, several decades old[13]. The key to their success was a neural net implementation that was 50 times faster than previous systems, allowing them to train on a much larger training set than had been previously used. Following that success, neural net models become the top performers on many other benchmark computer vision datasets, including ImageNet in 2012[38]. In speech, [22] set a new record on the small TIMIT dataset with a neural net model in 2013, and [57] achieved state-of-the-art large-vocabulary speech recognition performance in 2014. Also in 2014, deep learning models became the state-of-the-art in machine translation[61]. In all of these cases, neural network models outperformed systems that had been carefully optimized over many decades of research.

In this chapter, we cover the basics of deep learning, and discuss the model architectures that are most relevant to this thesis. For a more thorough overview of deep learning techniques, see [15]. For a complete history of deep learning and its successes, see [59].

2.1 Deep Learning Basics

2.1.1 Defining Neural Networks

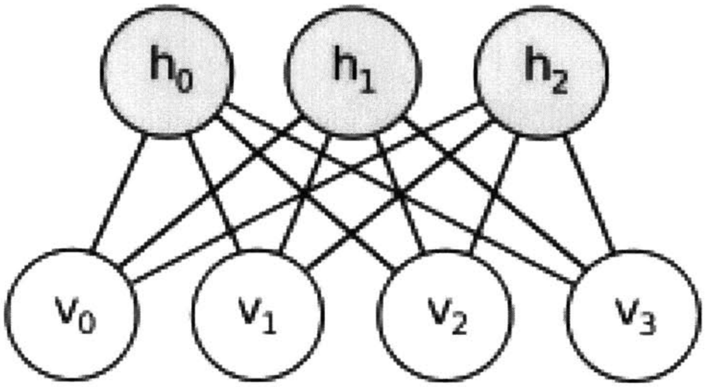


Figure 2-1: Restricted Boltzmann Machine, from [7]. Visible units v_i are the input to the model, while hidden units h_j are latent variables.

The term 'neural network' can be applied to any model comprised of units that communicate with each other through weighted connections. In the modern machine learning context, these weights are learned from data, through a training process that aims to produce a network that can perform a particular task. Arguably the simplest kind of neural network is the Restricted Boltzmann Machine, or RBM - a network with a layer of visible units, v_i in Figure 2-1, and a layer of hidden units, h_j. These networks are designed to model the distribution $P(X)$ of the input data by adding the latent variable H and modeling $P(X|H)$ and $P(H|X)$. For the case of binary units, we define:

$$P(h_j == 1|X) = \sigma(c_j + W_j X)$$

$$P(v_i == 1|H) = \sigma(b_i + W_i' H)$$

where c, b, and W are the learned parameters of the network, and σ is the sigmoid function. These parameters can be learned through gradient descent; see Section 5 of [6] for more details.

The 'restricted' term in RBM refers to the fact that these models are restricted to only connections between input and hidden units - no input-input or hidden-hidden connections. As a result, we have the following independence properties:

$$P(H|X) = \sum_i P(h_i|X)$$

$$P(X|H) = \sum_d P(x_d|H)$$

These properties allow us to, for example, use Gibbs sampling to sample from $P(X)$ or to more easily make statistical inferences about our data.

2.1.2 Supervised Learning with Neural Networks

For many applications, however, we wish to map from inputs to outputs; that is, our goal is not to model the input distribution $P(X)$ but the distribution $P(Y|X)$ for some output Y. Given a training set composed of paired input/output examples, this can be accomplished with a simple feed-forward neural network, illustrated in Figure 2-2.

While the neural network illustrated here has only one hidden layer, networks can be designed with an arbitrary number of hidden layers and, in practice, deeper networks often perform better, depending on the task at hand. These types of feed-forward neural networks, with more than one hidden layer, are often referred to as deep neural networks, or DNNs.

One frequent use case for such neural networks is classification, where our goal is to build a network that outputs the likelihood of an input example belonging to each of a set of categories. This can easily be accomplished with the addition of a softmax layer at the output of the network, which normalizes the activations in the previous layer so that they form a multinomial distribution. Then, the network can

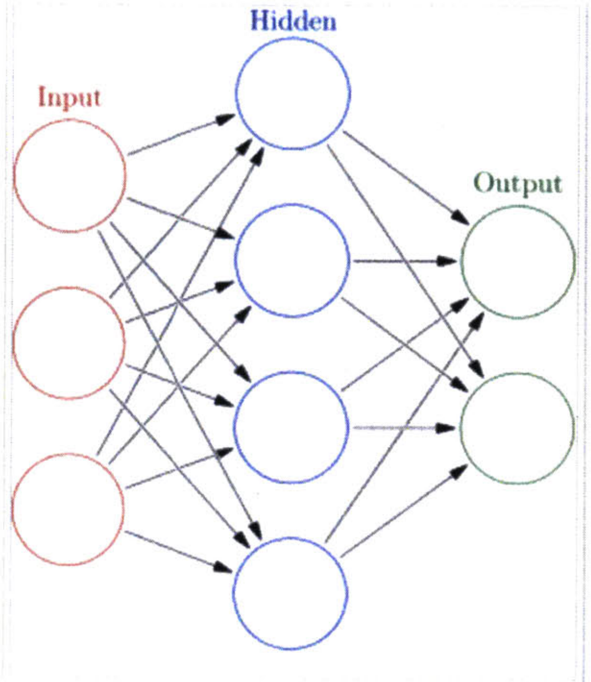


Figure 2-2: Basic feed-forward neural network architecture, from [1]. Units are arranged in layers and connected via weights that are directed from the input to the output of the network.

be optimized to maximize the likelihood of assigning the correct class label to each of the input examples in the training set.

This optimization is most commonly accomplished using the standard backpropagation algorithm [54], which compares the output of the network to the target output, computing an output error. This error is then propagated backwards through the network, updating the weights of the network in proportion to their contribution to the error. Often, a small number of examples are held out of the training set, and training is stopped when the performance of the model on those examples stops improving, even if the performance continues to improve on the training set. This type of early stopping can be very useful in preventing overfitting.

2.1.3 Unsupervised Learning with Neural Networks

In this thesis, we focus on using deep learning in an unsupervised scenario. Unsupervised learning has always been a part of deep learning - RBMs, and their deep counterparts DBNs, are trained in an unsupervised fashion. It is also possible to train DNNs in an unsupervised fashion; the key is to develop and objective function for the network output that is based solely on the input data. In this thesis, we explore two types of neural net models with unsupervised objective functions: autoencoders and predictors.

In autoencoders, the objective function measures how well the network can recreate the input. Instead of using a likelihood metric, as in the case of classification, we use mean squared error (MSE) to directly compare our inputs and outputs. Autoencoders have been especially successful for generating useful representations of input data. For example, Le et al. [39] trained a deep autoencoder on a large set of images, and found that many of the hidden units in their network became scale- and translation-invariant object detectors, for things like human and cat faces, despite never seeing any labels.

Predictor networks are useful for sequential data; the network is trained to take a set of inputs and output the next one in the sequence. For data that is naturally sequential, we don't need any manually generated labels to use this objective function. It is possible to train a feed-forward neural network to perform prediction (and we will investigate this in Chapter 7), but predictors are more commonly implemented as recurrent neural networks, as described in the next section.

2.2 Recurrent Neural Networks

2.2.1 Description

The term recurrent neural network (RNN) can refer to both a broad class of neural networks containing recurrent connections and to the simplest models in that class. The broad class contains all models in which: inputs are fed to the network step by step and the activation of a hidden unit depends on both the model's hidden activations at the previous step and the value of the current input. Recurrent connections are those that connect previous steps with the current one. An example of

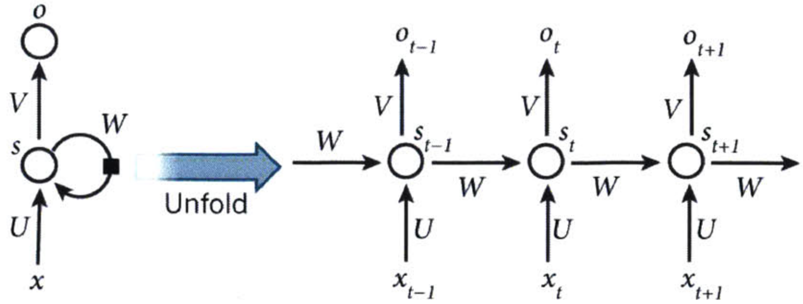


Figure 2-3: A recurrent neural network, unfolded in time, from [9]. x is the input, s is the recurrent hidden layer, and o is the output. Subscripts indicate timestep, so that x_t is the input at time t.

the simplest kind of RNN model is shown in Figure 2-3. These models have recurrent connections between the same layer at different timesteps, and the inputs from the previous timestep and from the previous layer are combined.

As explained in [29], these networks can be difficult to train with backpropagation - as errors are passed backwards in time, they tend to either blow up or vanish. A different type of recurrent neural network, the long-short-term memory (LSTM) network, was designed specifically to combat this issue[30].

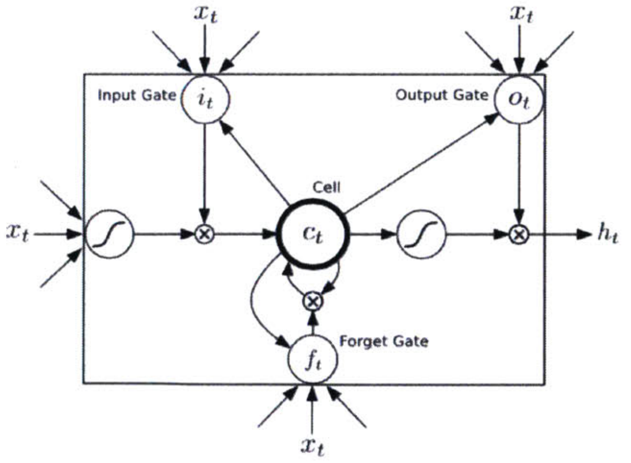


Figure 2-4: The long short-term memory (LSTM) cell, from [22]. At each timestep, the cell accepts input x_t and outputs activation h_t.

LSTM layers are composed of 'cells', shown in Figure 2-4, who output a hidden activation after several internal operations. The LSTM cell has an internal activation and three gates: input, forget, and output. The input gate controls how much influence the input from the previous layer has, the forget gate controls the influence of the previous activation of the cell has, and the output gate controls how much of the cell's information is passed on to the next layer.

In addition to being easier to train than simple RNNs, LSTMs have the property of being able to learn specific kinds of long-range correlations in sequences that simple RNNs cannot. See [30] for more information. In this thesis, we use LSTMs for all recurrent neural network experiments.

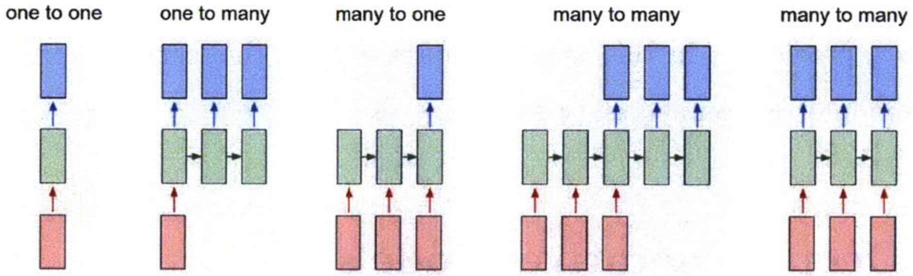


Figure 2-5: A variety of neural network architectures, from [35]. The leftmost is the standard feed-forward network; the other four are recurrent neural networks.

Recurrent neural nets can take many forms, as shown in Figure 2-5. The leftmost model in that figure is not recurrent - it corresponds to the feed-forward neural net discussed in Section 2.1.2. In this thesis, we experiment with the two rightmost recurrent neural network architectures. The second model from the right is an called an encoder-decoder model, which we use for our recurrent autoencoder. We use the rightmost architecture for our prediction network.

2.2.2 Successes

One of the most successful applications of recurrent neural networks is in machine translation. In [61], the authors developed an encoder-decoder model which has quickly become the state-of-the-art in that field. Their model is trained to encode a sequence of text from one language and then output a sequence of text from another language.

In the unsupervised learning space, recurrent predictor models have been very successful for language modeling, a common piece of many natural language processing systems (as well as speech recognizers). The RNNLM model [45] consistently outperforms both standard n-gram models and feed-forward neural networks designed for the same task.

An interesting use of recurrent networks for unsupervised learning comes from [60], where the authors use LSTMs to generate unsupervised representations of small video segments. One compelling contribution of their work is an encoder-decoder framework with two different decoders: one acting as an autoencoder and the other acting as a predictor. In this way, the authors force the learned representation to include the information necessary for both tasks.

2.3 Deep Generative Models

2.3.1 Description

A recent strain of research has sought to marry deep learning and probabilistic modeling. Classical probabilistic models must often make simplifying assumptions about the nature of data distributions, in order to make learning tractable. However, the success of deep learning models, which model incredibly complex functions, in many other fields has ignited a desire for probabilistic models in which data likelihoods can be described by neural networks.

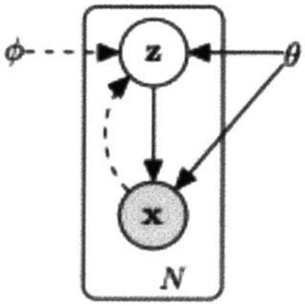


Figure 2-6: The graphical model underlying the variational autoencoder (VAE)[37].

A concrete example is given in Figure 2-6. The solid lines in this model depict a generative process in which an observation x_i is generated by first sampling a latent variable z_i from a prior $p_\theta(z)$ and then sampling x_i from the conditional $p_\theta(x|z)$. In the case where the likelihood $p_\theta(x|z)$ is a neural network with a nonlinear hidden layer, the marginal likelihood $p_\theta(x)$ and posterior probability $p_\theta(z|x)$ are intractable, making standard parameter estimation techniques impossible.

Kingma and Welling [37] devise an algorithm, which they call auto-encoding variational Bayes (AEVB) for exactly this scenario. They introduce a recognition model, $q_\phi(z|x)$, which approximates the true posterior $p_\theta(z|x)$. Using this model, they derive a lower-bound on the marginal likelihood which can be efficiently optimized when $q_\phi(z|x)$ and $p_\theta(x|z)$ are neural networks. The full derivation can be found in [37]. [53] independently develops an almost identical algorithm; their derivation comes at the problem from a different direction and thus provides a useful alternate perspective.

The AEVB algorithm gives rise to a model, which the authors call the variational autoencoder (VAE), in which the parameters ϕ and θ are optimized jointly. The model looks very similar to a standard feed-forward autoencoder, with some minor modifications. One key difference is that the output of the autoencoder, rather than being a direct recreation of the input, is a probability distribution. For the purposes of this thesis, we will explore the case where this probability distribution is a multivariate Gaussian distribution with diagonal covariance. This is accomplished by a final hidden layer that outputs a representation of twice the size of the input: the first half of this representation represents the mean of our Gaussian, while the second half represent the diagonal entries in the covariance matrix. Instead of using MSE as the objective function, as in a standard autoencoder, we use the likelihood of the input data given this output distribution.

2.3.2 Successes

In their original paper, Kingma and Welling [37] demonstrate the ability of their VAE to accurately model images of handwritten digits from the MNIST dataset, such that samples from the model are legible digits themselves. In a follow-up paper[36], the same authors show that by using just a small number of the MNIST labels, they can learn to separate the latent representations of digit class and handwriting style, letting them generate different digits in the same style as an input.

Alternative generative autoencoder models include the generalized denoising autoencoder[8] and the deep autoregressive network[24], both of which demonstrate similar success in sampling novel MNIST digits. Similarly, [11] presents an alternative to the semi-supervised VAE model, suggesting a similarly successful method for disentangling different sources of variation within the latent representation.

2.4 AB Networks

Another neural network architecture which will appear in this thesis is the AB network, sometimes called a Siamese network. These names come from the way such networks are usually drawn, with two identical networks side by side, connected at the top layer. These are not, in fact, two networks: they are two copies of the same network. Each copy is used to transform an input from the original space into an embedding space. The top layer of the network then compares the embeddings of the two inputs. These networks are not fully unsupervised - every pair of inputs requires a label which indicates whether they are of the same type or different types. The network is trained so that examples of the same type are close in the embedding space while examples of different types are far apart in the embedding space.

As we will see in the following chapters, these same/different labels may be much easier to acquire than the category labels used for typical DNNs. We will also discuss the need for similarity metrics for comparing learned representations in neural networks. Representations generated with an ABNet are very attractive in this case, as they are optimized with a specific similarity metric in mind.

These networks have been successful for a variety of tasks, including face verification [64], object recognition [46], and speaker recognition [10].

Chapter 3

Background: Speech Processing

In this chapter, we will give a brief overview of the breadth of prior work related to speech. First, we discuss the state-of-the-art in traditional speech recognition. Next, we look at the field of unsupervised learning from speech. Finally, we will explore work from cognitive science, specifically computational modeling of language acquisition.

3.1 Speech Recognition

Modern speech recognition systems are built on a foundation of manually-generated data. These include lists of phones (the basic sound units of language), lists of words, and pronunciation dictionaries to tie those two together. Additionally, these systems require hundreds or thousands of hours of manual transcriptions, in order to map the speech signal to these sounds and words.

The traditional speech recognition framework is shown in Figure 3-1. The decoder can be formalized as a probabilistic generative model; the goal is to find the sequence of words most likely to have generated an observed segment of speech. Formally:

$$W^* \simeq \arg\max_{W,S} P(O|S)P(S|W)P(W)$$

where O is the sequence of speech observations, S is a sequence of underlying states, and W is a sequence of words. Each probability in the equation above represents

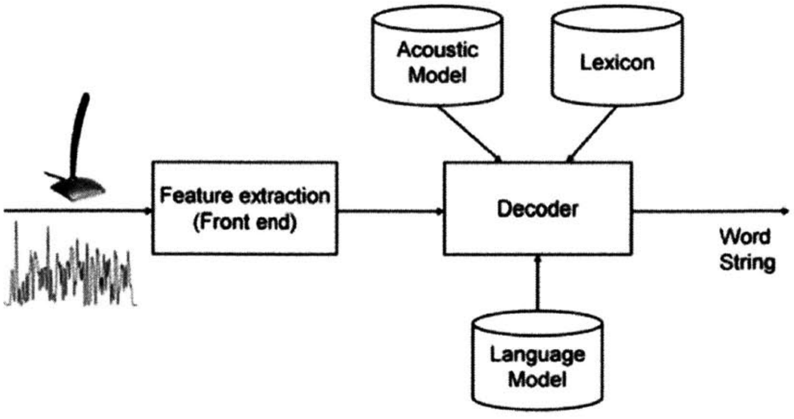


Figure 3-1: Diagram of the traditional ASR pipeline.

a separate piece of the speech recognition pipeline: $P(o|s)$ is the acoustic model, $P(S|W)$ is the lexicon, and $P(W)$ is the language model. Typically, the lexicon for a language is generated by expert linguists, while the acoustic and language models are learned from data. Language modeling is a classic unsupervised learning task - these models are typically built using a large corpus of unannotated text. Acoustic modeling, however, is heavily supervised, relying on hundreds or thousands of hours of manually transcribed speech. The acoustic model is the focus of this thesis.

In the typical acoustic model, each phoneme in a language is modeled with a Hidden Markov Model (HMM) whose states have emission probabilities modeled with a Gaussian Mixture Model (GMM). Training for these models requires either phoneme-level transcriptions of the training speech or word-level transcriptions combined with a lexicon. With this information, it is relatively straightforward to use the expectation maximization (EM) algorithm to learn the best parameters to model each phoneme. Of course, speech recognition is a very rich research area, and there are many techniques that can be used to improve these basic models, including speaker adaptation and state tying. For more details on HMM-GMM models, see [69].

Using the large-scale resources that have been collected for English and a few other widely-spoken languages, speech recognition has become increasingly effective. The improvements made in the past few years are owed largely to deep learning techniques, which have the ability to build powerful models but require more of this manually-generated data than ever before.

Within speech recognition, deep learning is most commonly used to augment or improve acoustic models. One way to do this is to use an HMM-GMM model to generate state-probability labels for each frame. Then a DNN is trained to predict these probabilities, with one output for each HMM state. Such a DNN can then be used as the acoustic model in a speech recognition system. Given large enough networks, so-called hybrid HMM-DNN acoustic models have been shown to outperform HMM-GMM acoustic models on a variety of speech recognition datasets [28]. Alternatively, such a DNN can be used to generate additional features for input to an HMM-GMM model. For this technique, a narrow "bottleneck" layer is placed in the DNN before the softmax layer. The activations of this hidden layer are called bottleneck features and can be used either in place of, or in addition to, the original engineered features. More information on this approach, often called a tandem system, can be found in [25].

Very recently, researchers have begun to investigate the use of deep learning for end-to-end speech recognition, as opposed to just acoustic modeling. These systems take speech signals as input and produce text transcriptions as output. Such training requires large amounts of transcribed data, but only at the sentence level - phonetic transcriptions, pronunciation dictionaries, and even word-level alignments are no longer necessary.

In [26], Hannun et al. develop a recurrent neural network (RNN) capable of end-to-end speech recognition. RNNs are distinguished from DNNs by their recurrent connections, which make the activity in the network dependent on the activity of the network at previous timesteps. The authors are able to train their network from sentence-level transcriptions thanks to prior work from Graves et al. [23] on sequence alignment with RNNs. Using this sequence alignment method, Hannan et al. [26] are able to train their model to output sequences of characters. They then use a language model (trained separately) to convert these character sequences to words.

While Hannun et al. [26] are able to achieve better speech recognition performance than the prior state-of-the-art on several datasets, their model does not represent a complete solution to the problem of difficult-to-obtain training data. While they have done away with low-level transcriptions and dictionaries, their model still requires a very large number of sentence-level transcriptions, which may still be unavailable in many languages. Second, the character output of their model will not transfer well to many languages, including those with irregular spelling and those that are not written phonetically. From a more abstract perspective, it is clear that this method of speech recognition does not match human language learning. Our goal is to build towards a more cognitively plausible speech recognition model, with the hope of eventually developing a system that can be applied to any language with minimal supervision.

3.2 Unsupervised Learning From Speech

Unsupervised speech recognition is an incredibly complicated task and such a system is far from being developed. Within just one component of a traditional speech recognition system, the acoustic model, are many interconnected tasks that must be solved in the zero-resource setting. These include, but are not limited to, discovering the set of low-level, speaker-independent units that comprise language, segmenting speech into such units, and modeling their dynamics. In this section, we review prior work that addresses some or all of these tasks. First, however, we discuss an alternative field of unsupervised learning from speech: spoken term discovery.

3.2.1 Spoken Term Discovery

Rather than perform full speech recognition, spoken term discovery systems have a simpler goal. They are designed to find repeated patterns (generally words or phrases) in a speech corpus. These systems typically operate without attempting to model the underlying structure of a language. A standard algorithm for such a system is segmental dynamic time warping (S-DTW), which can align sequences of unequal length[49]. This basic algorithm has been recently modified for efficiency on large speech corpora, through the use of sparse features[2] and randomized algorithms[32].

As in all unsupervised speech processing, speaker variation can be an obstacle to effective spoken term discovery. As a result, an active area of research related to spoken term discovery addresses the problem of developing features with better speaker-independence properties. One example of such work comes from [70], in which Zhang et al. train an unsupervised GMM and use that GMM to generate features. Specifically, they develop a posteriorgram vector representation in which each element represents the posterior probability that a given frame of speech was generated by a particular component in the mixture. The same authors obtain further improvements in a follow-up work by discriminatively training a DNN using labels generated by the GMM.

3.2.2 Discovered Terms as Supervision for Representation Learning

Rather than focus on the features used as input to spoken term discovery systems, some researchers have recently turned to using the output of those systems to improve feature representations for other downstream tasks.

In [34], the authors introduce a model called the "correspondance autoencoder" (cAE). Rather than reconstruct the input, this feed-forward DNN is trained to reconstruct an unseen example of the same type as the input. These pairings come from alignments of different instances of the same repeated pattern, as discovered by a spoken term discovery system. In [52], the authors compare a cAE model with a denoising autoencoder, which is trained to generate a clean version of speech when presented with a corrputed version. Renshaw et al.[52] argue that the cAE does its own kind of denoising, removing nonlinguistic sources of variation rather than artificial noise.

Thiolliere et al.[66] develop a similar model, but instead choose an ABNet architecture in which two inputs are fed through the same neural network to produce two vector representations. The inputs are either "matched" pairs, as in [34] and [52] or mismatched pairs, drawn from unaligned speech segments. The authors use a cosine distance-based loss function which is designed to push the representations of matched pairs close to each other while keeping the representations of mismatched pairs far apart. Interestingly, the authors see a significant improvement in the speaker-invariance of their representation, despite the fact that 94% of the word pairs extracted by their term detection algorithm were within speaker. [63] use a very similar architecture, finding that they can achieve performance close to that of a supervised acoustic model on this task.

In [62], the authors add an additional form of supervisory information: speaker identity. They use two ABNet models, which share their lowest-level weights - one trained on pairs with same/different linguistic content and one trained on pairs from same/different speakers. The authors find that training these networks jointly improves both the phoneme-discrimination ability of the linguistic representations and the speaker-discrimination of the speaker representations.

3.2.3 Deep Representation Learning With Temporal Coherence

Rather than rely on a spoken term discovery system, another strain of research in this area uses the notion of "temporal coherence" - the idea that frames that are close together in time are much more likely to belong to the same subword unit than frames that are far apart.

In [5], Badino et al. develop a segmental autoencoder in which the model is randomly asked to either re-generate the input or generate the subsequent frame. The authors binarize and cluster the resulting hidden activations to uncover subword units. The segmental autoencoder outperforms both GMM posteriorgrams and features from a standard autoencoder on multiple spoken term discovery and word classification tasks.

Synnaeve and Dupoux[62] use an ABNet architecture with a loss function that does not require a term discovery system. The goal of this loss function is to generate an embedding space in which frames belonging to the same phoneme are close together, while frames belonging to different phones are far apart. Rather than use a spoken term discovery system, Synnaeve and Dupoux simply take consecutive frames as "same" pairs and frames with lags of 15, 20, 25, and 30 frames as "different" pairs. In addition to increasing the similarity of the representations of frames from the same phoneme, the authors find a significant improvement in the speaker-invariance of their representations, despite not explicitly using speaker information.

3.2.4 Subword Discovery and Modeling

The speaker-invariant features discussed in the previous section are especially useful for subword discovery and modeling, the closest analogue to unsupervised acoustic modeling in the field. Approaches to this problem cover a wide range of techniques, including self-organizing units (SOU) [43, 20], a hidden Markov model framework with state-splitting [4], and a Bayesian nonparametric model [41].

[41], the last model mentioned above, is particularly notable for performing joint discovery, segmentation, and modeling of phone-like units. The authors developed a generative model of speech governed by a Dirichlet process which allows the model to determine the number of units that best models the data - unlike many other models in this area, which take the number of units to be learned as given. Lee uses an iterative Gibbs Sampling procedure to infer all of the hidden parameters of the model. For more details, see [41].

An additional line of research in this area uses the output of a spoken term discovery system to provide constraints for unsupervised acoustic modeling. Two instances of the same word or phrase should be composed of the same sequence of subword units; enforcing that fact significantly limits the search space. If these repeated words or phrases are found across speakers, using these constraints can greatly improve the speaker-independence of the discovered units. In [31], Jansen and Church train whole-word HMM-GMM models for each discovered term, then collapse correlated states across terms to form a global set of subword units. In [33], the authors use aligned frame pairs from a spoken term discovery system as constraints for a clustering system, enforcing the idea that both frames in such pairs should fall into the same cluster.

3.3 Human Language Acquisition

In thinking about this problem, we have an obvious model for how a zero-resource speech recognition system might work. Children, after all, learn language predominantly from exposure to speech. Cognitive scientists know a fair bit about the progression of that learning process, and that knowledge could definitely inform a speech recognition system. On the other hand, there is much cognitive scientists still do not know about language acquisition: chief among them, how much of our language knowledge is innate, and how much is learned. Our goal in this thesis is to simultaneously learn from and inform cognitive science research - models built on knowledge gleaned from their results may suggest new research directions for that field.

Human language acquisition has been studied extensively within the fields of Cognitive Science and Psychology. We do not have space here for anything approaching a full review. Instead, we will focus specifically on efforts within the field of computational cognitive science - research which explores the plausibility of different computational models of human language acquisition. Over the course of this work, researchers have learned a lot about which properties of speech can be captured by which types of models. Our connection to this research is twofold: first, we want to make use of the insights that these researchers have found, and, second, we want to suggest potential new models of human language acquisition. Hopefully this can turn into a kind of feedback loop.

One relevant strain of research explores the basic cues that infants might use to segment speech into words. For example Saffran, Aslin, and Newport found that eight-month old infants are capable of statistical learning based on the distributions of sounds within words[56], and that they can compute conditional probabilities[3]. Goldwater et al.[55] later used these insights to develop a Bayesian word segmentation model which suggested other cues that might be equally important for early language learning.

Elsner et al.[16] present a model of how infants might acquire word and phoneme categories simultaneously, showing that joint learning can be advantageous relative to learning these categories sequentially. Fourtassi et al. [19, 18] explore similar themes, finding that top-down cues are especially important for particular cases of phonemic categorization. Similarly, [47] and [44] find that a ‘proto-lexicon’ can provide a significant boost to phoneme learning.

Chapter 4

Data, Evaluation, and Tools

4.1 TIMIT

The Texas Instruments/Massachusetts Institute of Technology (TIMIT) corpus was first released in 1988 by the National Institute of Standards and Technology and has since become a standard test set for speech processing. The dataset is composed of read speech from 630 speakers, each of whom reads 10 sentences. Two sentences were read by every speaker in the corpus; of the remaining sentences, three are unique to one speaker and five are read by multiple speakers. The three unique sentences are general English sentences, while the sentences read by multiple speakers were designed to be phonetically balanced at the bigram level. As a whole, the sentences were designed to cover the entire range of phonetic content of American English. The standard training split for speech recognition on TIMIT first removes the sentences spoken by every speaker, then separates the dataset into 3696 training utterances, 192 development utterances, and 192 test utterances, such that there is no overlap between the sets in either speaker or sentence.

Along with the speech signal, the TIMIT corpus includes time-aligned orthographic, phonetic, and word transcriptions. While we will not use these transcripts to train our models, they are necessary for evaluation. The TIMIT corpus was chosen to evaluate this work in part because of the high quality of these transcriptions. Another reason it was selected for this project is its place within the speech processing community. The performance of other models on this dataset is widely available, making it a useful benchmark. Specifically, Lee and Glass[41] evaluated their model primarily using TIMIT. For this paper, we will directly compare against the results reported there, using the same metrics the authors used. Those metrics are described below.

4.1.1 Phone-Level Evaluation

Lee and Glass[41] make use of the phonetic transcriptions to evaluate their model on two different axes. First, they use precision/recall metrics to compare their segmentation of speech into phone-like units with the actual segmentation. A proposed boundary is considered a true positive if it is within two frames, or 20 ms, of a true boundary. Similarly, a false negative occurs if no boundary is proposed within 20ms of a true boundary. The standard formulas for precision and recall are given below.

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives}$$

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives}$$

As an overall segmentation metric, Lee and Glass[41] use F-score, which is defined as the harmonic mean of the precision and recall.

4.1.2 Spoken Term Discovery

Zhang and Glass[70] developed a metric based on spoken term discovery on the TIMIT dataset, which we adopt here. For the TIMIT corpus, they decided on ten terms that occur in both the train and test splits with varying frequency, as shown in Table 4.1. For a given term, they use all occurences within the training split as examples, and search for matches within the test corpus using a DTW-based system. They then developed two metrics, inspired by information retreival evaluations, to judge the list of matches returned by the system. First is a metric called precision at N, or P@N for short. This metric measures the precision of the top N matches, where N is the

Key Term	Train Instances	Test Instances
age	3	3
surface	3	1
artists	7	1
organizations	7	1
development	9	2
warm	10	1
year	11	3
children	18	2
money	19	4
problem	22	3

Table 4.1: Key terms used for TIMIT spoken term detection evaluation.

number of actual occurrences within the test corpus for the given term. The second metric is equal error rate, or EER. It measures the point at which the false positive and false negative rates are equal. In the ideal system, the top N matches are all correct; P@N is 1 and EER is 0. In the worst case scenario where we never find a match, P@N is 0 and EER is 1.

4.2 Zero Resource Speech Challenge

The study of unsupervised learning from speech has suffered from a lack of standard evaluation resources. In 2015, a group of researchers launched the Zero Resource Speech Challenge[68] to address this issue. The Challenge includes speech data in two languages (English and Xitsonga) and evaluation scripts for two different unsupervised learning tasks.

The Zero Resource Speech Challenge was launched as a workshop at the Interspeech conference in September 2015. The Challenge introduced a new subword modeling task which scores frame-level representations on their phonetic content and speaker independence. This task is described in more detail in the following chapter; here, we give only a high-level overview. This task has two parts: within-speaker and across-speaker. In the within-speaker version, the goal is to find features that can distinguish two instances of the same phone from an instance of a different phone, when all three are spoken by the same speaker. In the across-speaker task, the goal is to pair the matched phones when they are spoken by two different speakers. A variety of papers were presented - here, we will cover those most closely related to this work.

4.2.1 Challenge Design

As described by the developers, the Zero Resource Speech Challenge is driven by two complementary research interests. First, the challenge is designed to explore the scientific question of how "an infant or a system [could] learn language(s) in an unsupervised fashion?"[68]. Second, their goal was to "push the envelope on the notion of adaptability and flexibility in speech recognition systems by setting up the rather extreme situation where a whole language has to be learned from scratch"[68].

More specifically, the organizers sought to develop and provide common evaluation metrics and datasets for the low-level tasks often studied in the field. The released metrics address representation learning for tasks at two levels of linguistic structure: subword modeling and spoken term detection. In both cases, the inputs to the evaluation tools are frame-by-frame transcriptions of the test dataset in terms of the representation to be evaluated. Each frame must be given a timestamp, which allows for regular or irregular frame-spacing.

4.2.2 Data

The data for this challenge comes from two languages: English and Xitsonga. For this paper, we will focus only on the English data, which is casual conversational speech from the Buckeye Corpus[50]. The corpus includes speech from 40 speakers, 30-60 minutes each. The test set is composed of all speech from 12 speakers of these speakers; the remaining data is used for training. All data has been transcribed at the phone- and word-levels; the transcriptions were generated by an ASR system and then manually cleaned up.

Speaker information is available as part of the challenge, as this is readily available in most data collection scenarios, as well as for infants acquiring language. While speaker information is also available for the TIMIT corpus, there is such a small amount of data per speaker that it is difficult to make effective use of it. Thus, we will focus on this data when conducting experiments that include speaker information.

4.2.3 Minimal Pair ABX Evaluation

For the purposes of this thesis, we will focus on the first track of the Zero Resource Challenge, which the organizers refer to interchangeably as a subword modeling task and minimal pair ABX task. The evaluation scripts provided as part of the challenge take as input a low-level representation of speech: fixed-length feature vectors associated with a time in the speech signal. The task is designed to test whether these features allow discrimination between same/different pairs of phonemes both within- and across-speakers.

The minimal pair ABX task is inspired by a standard paradigm used in psychophysics experiments, but has recently been used in this context[58]. A minimal pair is a pair of items that differ along one dimension but are otherwise the same. In the context of subword modeling, the challenge organizers give the example of the minimal pair "beg" and "bag". Specifically, the ABX-discriminability of category x from category y, in this case "beg" and "bag," is the probability that A and X are further apart than B and X according to some distance d over the representations for these sounds when A and X are from category x and B is from category y. Given a set of sounds $S(x)$ from category x and a set of sounds $S(y)$ from category y, we estimate this probability using the following formula:

$$\theta(x,y) = \frac{1}{m(m-1)n} \sum_{a\in S(x)} \sum_{b\in S(y)} \sum_{x\in S(x)\setminus\{a\}} (\mathbb{1}_{d(a,x)<d(b,x)} + \frac{1}{2}\mathbb{1}_{d(a,x)=d(b,x)})$$

where m and n are the number of sounds in $S(x)$ and $S(y)$ and $\mathbb{1}$ is the indicator function. The notion of ABX discriminability defined above is asymmetric in the two categories. We obtain a symmetric measure by taking the average of the ABX discriminability of x from y and of y from x.

For the evaluation, all triphone minimal pairs in the corpus which differ at the central phoneme are enumerated. For the within-talker condition, the scores are combined for all occurances in which A,B, and X come from the same speaker. The scores for a given minimal pair are first averaged across all of the speakers, then over all found contexts for a given pair of central phones. Finally the scores for every pair of central phones are averaged to yield the reported within-talker ABX discriminability. For the across-talker condition, A and B come from the same speaker, and X comes from another speaker. The scores for a given minimal pair are first averaged across all of the pairs of speakers for which this contrast can be made. As with the within-talker measure, the resulting scores are then averaged over all contexts for each possible pair of central phones and finally over all pairs of central phones.

4.3 Data Preprocessing

Both datasets were preprocessed in the same way. First, each waveform was segmented into 25ms frames whose centers were 10ms apart, yielding 15ms frame overlap. For each frame, we use an FFT to get the frequency spectrum, and partition that spectrum into 40 Mel-scaled filterbank spectral features per frame[14]. We then convert these features to dB by taking the log. Throughout this thesis, we refer to this representation simply as filterbank features.

4.4 Tools

For this work, we used a number of tools, both open source and developed within the SLS group. The open-source speech recognition toolkit Kaldi[51] was used for basic data processing. All neural network models were implemented using the deep-learning toolkit Keras[12], running on top of the Theano[65] backend.

For evaluation, we are indebted to Chiaying Lee[41], who made her original model and evaluation tools available to us, Ruslan Salakhutdinov, who provided the code used for [60], and Joost van Amersfoort who created an open-source implementation of the variational autoencoder[67]. Additionally, we used the evaluation tools released as part of the Zero Resource Speech Challenge[68].

Chapter 5

Deep Autoencoders for Representation Learning

Deep autoencoders are the most basic deep learning framework for unsupervised learning. As a result, they have been the subject of much early experimentation in the realm of unsupervised learning for speech. These experiments have repeatedly found that standard autoencoders are not well-suited to learning linguistically meaningful frame-level representations of speech. The likely reasons for this are straightforward: linguistic information is intertwined with nonlingustic information within the speech signal, and a standard autoencoder has no way of learning to separate those streams. Despite the clarity of this issue, possible fixes have not been explored - as seen in the previous chapter, the most effective approaches to the Zero Resource Challenge either resort to some form of supervision or switch to a different model all together.

Nonetheless, we wish to continue to explore autoencoders as they are a natural and useful framework for unsupervised learning. New ideas have been introduced which have been applied to other domains (most notably computer vision) but have never been tested in speech. In this chapter, we will dive into the properties and potential of several types of autoencoders. Our goal is to survey their abilities, their shortcomings, and their potential uses.

In this chapter we will apply a variety of autoencoder architectures to the TIMIT spoken term discovery task, described in Section 4.1.2, and the Zero Resource Chal-

Layers	Layer Size	P@N	EER
1	4	36.6	19.3
	16	44.6	**15.2**
	64	45.1	17.3
3	4	42.9	19.6
	16	**47.6**	**15.4**
	64	42.6	21.7
5	4	32.6	20.2
	16	47.1	18.3
	64	45.1	19.8

Table 5.1: Autoencoder results on TIMIT spoken term discovery task.

lenge minimal pair ABX task, described in Section 4.2.3.

5.1 Autoencoder Baseline

5.1.1 Model

We must first establish the baseline of what standard autoencoders can do on our tasks. We specifically focus on the size of the network, in terms of the number of hidden layers and the number of units in those layers. For simplicity, all hidden layers in each model will be the same size. We look only at networks with odd numbers of hidden layers, with the central hidden layer used as the feature representation for the downstream tasks. We explore networks with one, three, or five hidden layers of four, 16, or 64 hidden units each. Recall that the inputs to these models are 40-dimensional - models with either four or 16 hidden units are performing compression, while models with 64 hidden units have a larger representational capacity. We use a *tanh* nonlinear activation on these hidden layers. After the hidden layers, we add an output layer with a linear activation, to map from the hidden layer size back to the original 40 dimensions.

Layers	Layer Size	Within	Across
1	4	17.6	27.6
	16	18.6	32.9
	64	19.7	33.7
3	4	**16.8**	**27.0**
	16	**16.5**	30.9
	64	19.5	31.1
5	4	17.8	28.7
	16	**16.9**	29.6
	64	18.2	30.1

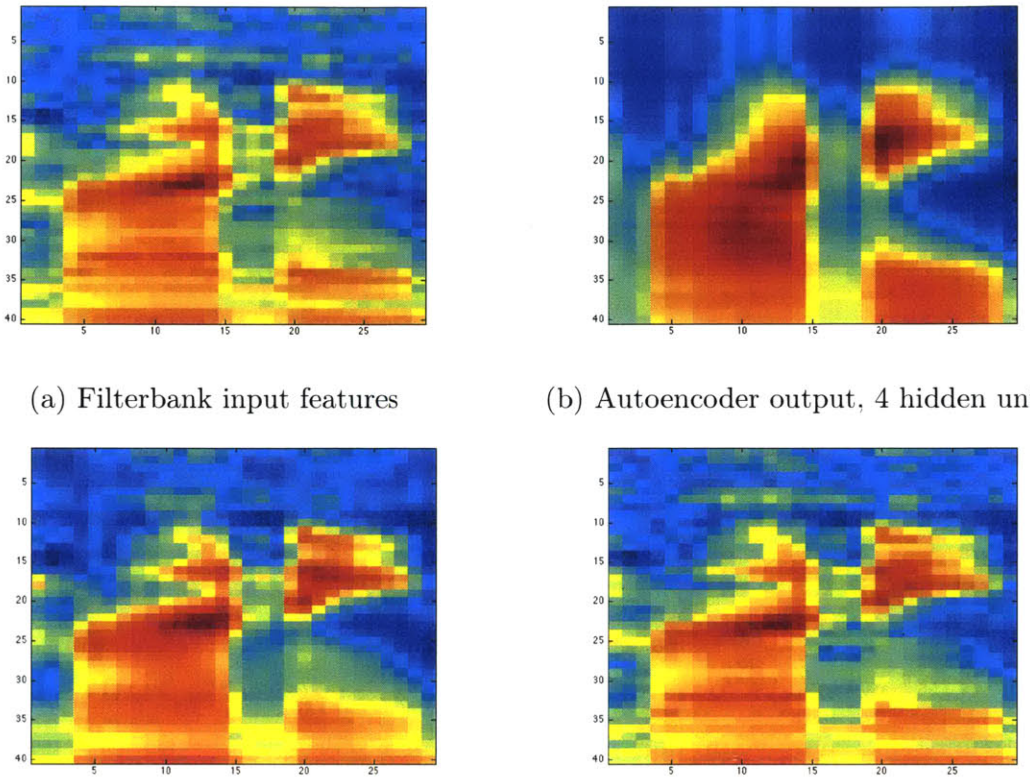
Table 5.2: Autoencoder results on Zero Resource Challenge minimal pair ABX task.

5.1.2 Results

Table 5.1 shows results for a variety of autoencoder sizes on the TIMIT spoken term detection task. As a reminder, recall that higher scores are better for the precision metric, P@N, while lower scores are better for the error metric, EER. The baseline filterbank features produce a P@N of 50.9 and an EER of 13.1 on this task.

The results on the Buckeye Corpus ABX task, shown in Table 5.2, tell a different story. While the much smaller TIMIT task does not show any clear trends, in terms of the best or worst performing network sizes, these results are more easily interpretable. The baseline filterbank performance on this task is 16.4 within-speaker and 29.2 across speaker; lower numbers are better in both cases. On this task, some autoencoders are able to perform only slightly worse than the baseline on the within-speaker task while outperforming the baseline across speaker by almost 8% relative.

Interestingly, the best performance comes from models with very few hidden units, which is a network architecture that has not been explored in prior work. One possible cause of these results is the lossy compression performed by these autoencoders. Based on the ABX results, it appears the network has lost some information about both speaker variations (improving the across-speaker results) and linguistic content (degrading the within-speaker results). Thus, while we have improved upon the original results, in terms of raw numbers, we haven't done it in the most desirable way.



(a) Filterbank input features

(b) Autoencoder output, 4 hidden units

(c) Autoencoder output, 16 hidden units

(d) Autoencoder output, 64 hidden units

Figure 5-1: (a) Filterbank features, male speaker reading word "money". (b) Output of autoencoder with one 4-unit hidden layer, given input from (a). (c) Output of autoencoder with one 16-unit hidden layer. (d) Output of autoencoder with one 64-unit hidden layer.

5.1.3 Qualitative Analysis

For most models tested, we confirmed prior results on this task: hidden representations pulled from a trained autoencoder yield worse performance than the original filterbank features. In one model, with very few hidden units, we find an improvement in across-speaker phoneme discrimination, but at the cost of some within-phoneme performance. In order to understand and improve upon these results, we wish to understand in detail what these autoencoders are learning.

The first step towards understanding these models is to verify that they are effectively recreating their input. If the hidden representations contain all of the information necessary to reproduce their input, then they should also contain all of the information necessary to perform our evaluations at the same level as those inputs. Figure 5-1 shows one example input speech segment from TIMIT, along with the recreated output from three different trained autoencoders. Notice, first, that the models' ability to recreate their input is directly correlated with the number of hidden units in the autoencoder. This trend holds strong regardless of the number of layers in the model. Second, the loss associated with the model does not directly correlate with the performance on the spoken term detection task - the best performance achieved on our task came from a model with 16 hidden units, despite the fact that the 64-unit model is much better able to recreate the input. This fits our understanding of speech, which we know contains many sources of variability in addition to linguistic content - our ultimate goal is to represent some, but not all, of the properties of the input.

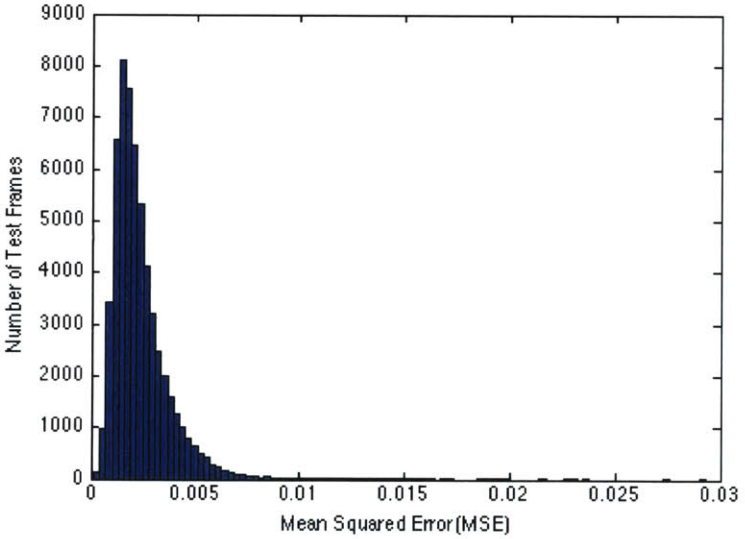
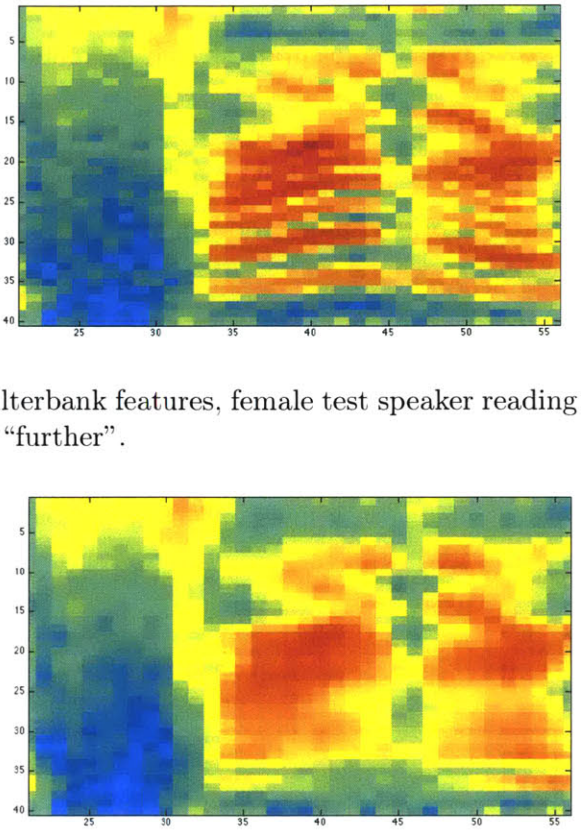


Figure 5-2: Histogram of mean squared error (MSE) of TIMIT test frames.

Another question we could ask is whether the system learns to represent and recreate all test speech equally well, or whether there is a range of reconstruction errors. Figure 5-2 shows a histogram of the MSE of all TIMIT test set frames, for a model with three hidden layers of 16 units each. The distribution is approximately exponential, with the vast majority of the frames falling in a relatively small interval but a long tail of frames with higher MSE. This long tail is comprised almost exclusively



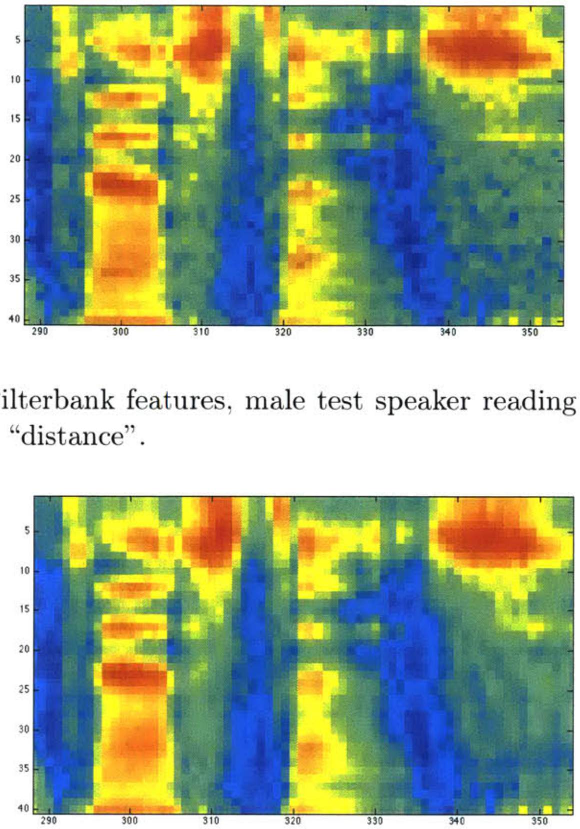
(a) Filterbank features, female test speaker reading the word "further".

(b) Output from autoencoder with three hidden layers of 16 units each, given input from (a).

Figure 5-3: Example autoencoder input and output for the speaker with the highest average MSE in the TIMIT test set.

of frames from female speakers; 26 of the 30 frames with the highest MSE come from one female speaker in particular. The overall average per-frame MSE on the test set is 0.0023. There are two outlier speakers with much higher average per-frame MSE: 0.0031 and 0.0035. These speakers are both female; overall, the female speakers are not as well-modeled as the male speakers. This is unsurprising, as there are twice as many male speakers as female in the TIMIT training set.

Figure 5-3 shows the segment of speech with the highest MSE in the test set - a female speaker reading the word "further". The 'f' is recreated with average fidelity; the two 'er' phonemes are very blurry. Compare this with Figure 5-4, which shows the input and output corresponding to a male speaker with average MSE reading the



(a) Filterbank features, male test speaker reading the word "distance".

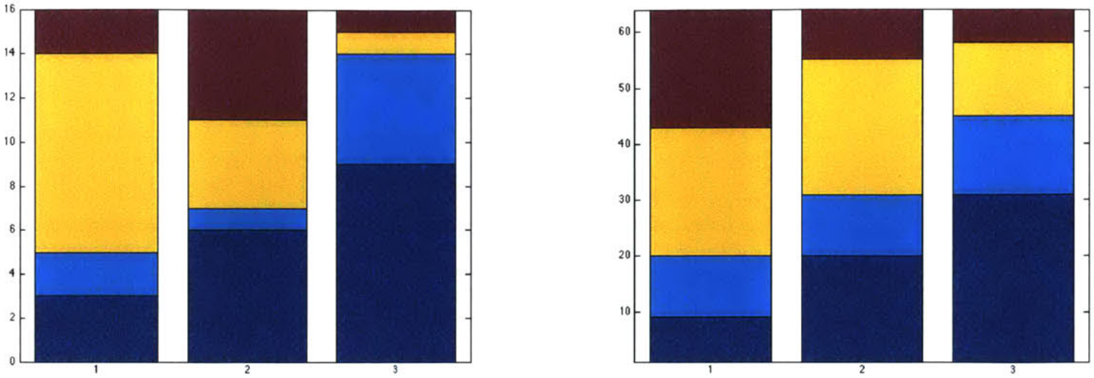
(b) Output from autoencoder with three hidden layers of 16 units each, given input from (a).

Figure 5-4: Example autoencoder input and output from the TIMIT test set. This speech segments comes from a male test speaker whose average MSE is close to the average MSE of the entire test set.

word "distance". The output isn't perfect, but all of the contours are represented.

It's not just that male speakers are well-represented and female ones are not. Within each gender, there is a significant range of per-speaker MSE. There are many possible reasons for this, including proximity to a speaker in the training corpus and recording conditions. We will discuss speaker effects in the next section.

Next, we turn to an analysis of the hidden units. We again use the best TIMIT model: 3 hidden layers, 16 units each. Following [62], we use the ratio of between-class to within-class variance in unit activation to assess the extent to which each unit encodes speaker or phone information. For each unit, we use a two-tailed F-test to compare the variance in the activation of that unit across the entire test set (i.e. be-



(a) 16 hidden units per layer.

(b) 64 hidden units per layer.

Figure 5-5: Phone- and speaker-selectivity of hidden units in autoencoders with three hidden layers. On each chart, the bottom layer is on the left and top layer is on the right. Dark blue units are phone-coding, yellow units are speaker-coding, and cyan units code for both. Dark red units code for neither.

tween classes) to the within-class variance for the gold-standard phoneme categories. A unit is considered to code for phoneme information if its between/within class variance ratio is greater than the median variance ratio of all units in the network. We do the same calculations using the speaker classes to find speaker-coding.

The results of this analysis are shown in Figure 5-5, for two three-layer autoencoders; one with sixteen hidden units per layer and the other with sixty-four. There are two clear trends here: the number of speaker-coding units decreases as we move forward in the network, and the number of phone-coding units increases. This is somewhat surprising, as we have not given the network any explicit phone or speaker information. Nonetheless, this is a very encouraging result - it suggests that these models can learn something about phone and speaker categories from just speech.

This analysis also suggests that we should not be using the middle hidden layer, but should instead be using the last hidden layer. For the model with 16 hidden units per layer, when we use the activations of the last hidden layer as features for the spoken term detection task, we do get a slight improvement in P@N - from 47.6 to 48.0. For the 64 hidden unit model, we see a more significant improvement: 42.6 to 48.0. Note that, in both cases, we could have made this switch using speaker information but not phone information, meaning that this type of analysis could be included as part of the model within our learning paradigm.

From this analysis, we can take away several important points that will inform future experiments. First, we find that a simple autoencoder architecture is able to effectively recreate speech input, with the quality of that recreation directly related to the size of the hidden layers in the model. Second, we find that some test speakers are not well-modeled by our autoencoder. Finally, we are able to see that most hidden units inside the autoencoder are encoding meaningful information - phoneme information, speaker information, or both. Most excitingly, the usefulness of this information for our task reliably increases as we move higher in the network, and we are able to get some gains in performance by choosing the hidden layer with the fewest speaker-selective units.

5.2 Separating Out Speaker Information

The simple autoencoder models tested above do not address a fundamental issue: our ultimate goal is to produce frame representations that are independent of nonlinguistic sources of variability, but we have done nothing to ensure that independence. In this section, we will focus on one source of such variability: speaker identity. Our reasons for focusing on speaker information are twofold: first, speaker identity is the most salient source of nonlinguistic information in our datasets (within each dataset all utterances are recorded on the same channel and are spoken with relatively flat affect), and second, speaker labels are available in most contexts. The rules of the Zero Resource Challenge specifically note that speaker information is likely to be available in most cases, and is certainly available to infants during the language learning process. Therefore, we can use this information without jeopardizing our unsupervised learning framework.

In the previous section, we saw that simple, deep autoencoder models encode more speaker information in the lower layers and less in the higher layers. However, significant amounts of speaker information are still present in the highest layers of the

Training Data	Layers	Layer Size	Within-Speaker
Full Training Set, 7.6M	3	16	16.5
Reduced Training Set, 400K	3	16	16.2
Single Speaker, 240K-390K	3	16	16.2
	5	16	16.2

Table 5.3: Autoencoder results on Zero Resource Challenge minimal pair ABX task. 'Training Data' column indicates data used to train autoencoder, with size given in number of frames. 'Single Speaker' denotes a set of models, each trained on one speaker and used to generate features for that speaker. The 'Reduced Training Set' is a randomly chosen subset of the full training set, with size chosen to match the largest amount of data used to train any one single speaker model.

network, and we would not be able to disentangle that information from the phoneme information in those representations without phone labels.

5.2.1 Single Speaker Training

The simplest way to reduce speaker variability is to remove it from our dataset by focusing on a single speaker. Training on speech from a single speaker fits nicely with the human language learning paradigm; children learn speech from small number of easily distinguishable speakers. It is possible, though very difficult to empirically test, that the lack of speaker variation in children's early language inputs is necessary for effective language learning.

For these experiments, we will focus on the Buckeye data, as the TIMIT corpus does not contain enough speech from any single speaker to train a model. Before, we trained one model on the training data and then used that model to generate representations for all of the test data. Here, we will train a separate model for each of the test speakers, using each model to generate features for only the data on which it was trained. This means that the hidden representations are no longer comparable across speakers. Thus, we focus only on the within-speaker scores.

Table 5.3 shows the best results on the within-speaker discriminability task using single-speaker models, compared to models trained on speakers not included in the test set. Contrary to our hypothesis, the results in all rows of Table 5.3 are comparable. Training on a small, randomly sampled portion of the training set does not diminish the results. Likewise, training on data from a single speaker, matched to the test utterance, is no better than training on a mix of speakers from the training split.

However, the pattern of MSE results are exactly what we would expect. For the same model architecture (in Table 5.3), the MSE of the speaker model is 0.0017, the full training set is 0.0037, and the reduced training set is 0.0039. This means that having more data is better than less (0.0037 v. 0.0039), but having matched data is the best.

The goal of an autoencoder model is to learn an embedding space that covers the range of likely inputs, based on the training data distribution. We hypothesized that, by removing speaker variations from the training data, we would limit the range of inputs that the autoencoder would need to represent. This could allow the embedding to use its space to represent other sources of variation, namely linguistic content. One reason this hypothesis did not work out could be that the model is also representing variation within linguistic units. In other words, the dynamics of phonetic units could be the source of the errors in the within-speaker condition. We'll address this in the next chapter.

5.3 Probabilistic Models

In this section, our first goal is to look first at what probabilistic neural network models can do, in the context of speech, as they have not been previously applied in this way. Second, we will look at how their learning differs from that of the autoencoders seen in the previous section, and whether those differences can be used to improve performance on downstream tasks.

5.3.1 Models

The first model explored in this section is the variational autoencoder described in Section 2.3.1. Recall that this model is composed of two models, one that computes

$q_\phi(z|x)$, a distribution over the latent representation given the input, and one that computes $p_\theta(x|z)$, a distribution over the input given a latent representation sampled from that distribution.

Both $q_\phi(z|x)$ and $p_\theta(x|z)$ are neural networks. In this section, we experiment with several architectures for these networks, matching the overall architecture to the models tested in the previous sections. When $q_\phi(z|x)$ and $p_\theta(x|z)$ do not have any hidden layers, the overall network has one hidden layer: z. When $q_\phi(z|x)$ and $p_\theta(x|z)$ have one hidden layers each, the overall network has three hidden layers. With two hidden layers each, the total is five. Following the previous sections, the hidden layer(s) in $q_\phi(z|x)$ and $p_\theta(x|z)$ have the same number of hidden units as z; we again test networks where that number is four, 16, or 64.

The central hidden layer is always the latent representation z; this representation is what we use for our downstream tasks. Recall that this layer produces a multivariate Gaussian - we use the mean of this Gaussian as the feature representation, paired with a cosine distance metric. Similarly, the output of these networks is also a probability distribution. Instead of MSE, we use the log-likelihood of the input as the objective function for training.

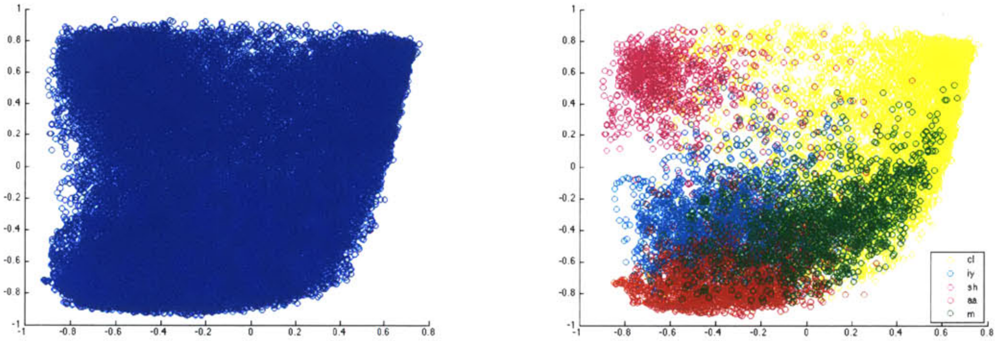
In addition to the variational autoencoder, we also test a middle-ground between it and the standard autoencoder, in which we use the original deterministic hidden layers but have Gaussian output. This gets us around the problem of how to use the latent variables as features, but potentially still gives us the power of probabilistic models. We call this model the Gaussian autoencoder, or GAE.

5.3.2 Results

On, TIMIT, we find again that a network with three hidden layers of 16 units each gives the highest performance on the spoken term discovery task, for both the GAE and VAE. The best performing VAE has P@N 40.6 and EER 25.4; the best performing GAE has P@N 42.6 and EER 20.3. These numbers are significantly worse than both the baseline filterbank features and the deterministic autoencoder results.

5.3.3 Analysis

Despite performing poorly on the tasks used here, these models do present an opportunity for more principled probabilistic analysis. First we'll look at the variational autoencoder. For the initial analysis, we build a model with only two units in the latent hidden representation, for easier visualization. This model does not perform well on our spoken term discovery task, but can nonetheless give us a glimpse into what the model is learning.



(a) 2D latent representations of all test frames

(b) Test frames from selected phone categories

Figure 5-6: 2D-latent VAE representations of TIMIT test frames.

As in [53], we first plot the mean 2D latent representations of all our the TIMIT test data; this is shown in Figure 5-6a. We can immediately see that the VAE makes full use of the available latent space, suggesting that it is effectively learning about what speech looks like. Next, in Figure 5-6b, we plot only the frames from five selected phone categories. While there is some overlap between the representations of these categories, they have been somewhat separated in the latent space. This is an encouraging sign of the model's ability to distinguish between phone categories. Of course, these phones were selected to illustrate this separation - if we had chosen several vowels, for example, we would see a lot more overlap.

Next, we follow [37] in visualizing the learned manifold of our data, shown in Figure 5-7. Our goal is to see how the learned generative model $p_\theta(x|z)$ changes over the latent space. To do this, we start with a linearly-spaced grid inside the unit

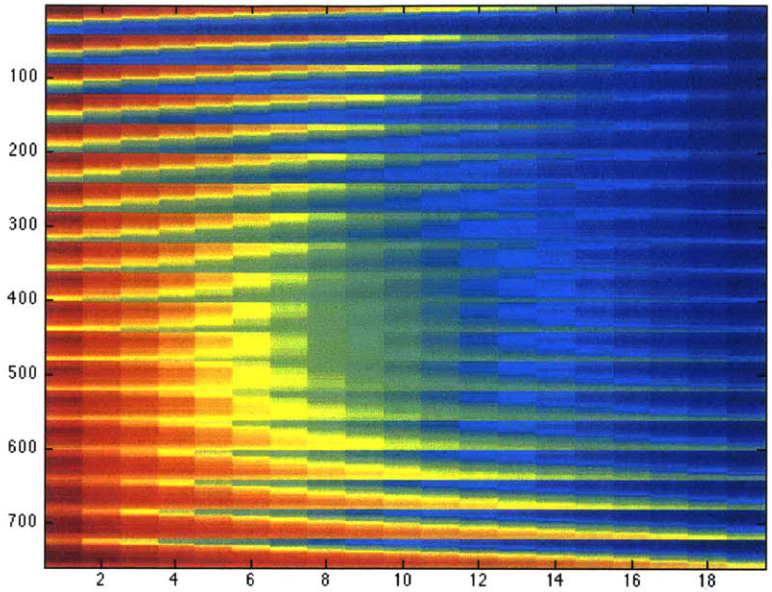


Figure 5-7: Learned data manifold for VAE with 2D latent space. Since the prior of the latent space is Gaussian, linearly spaced coordinates on the unit square were transformed through the inverse CDF of the standard Gaussian to produce values of the latent variables z. For each of these values z, we show the mean of the corresponding generative $p_\theta(x|z)$ learned during training.

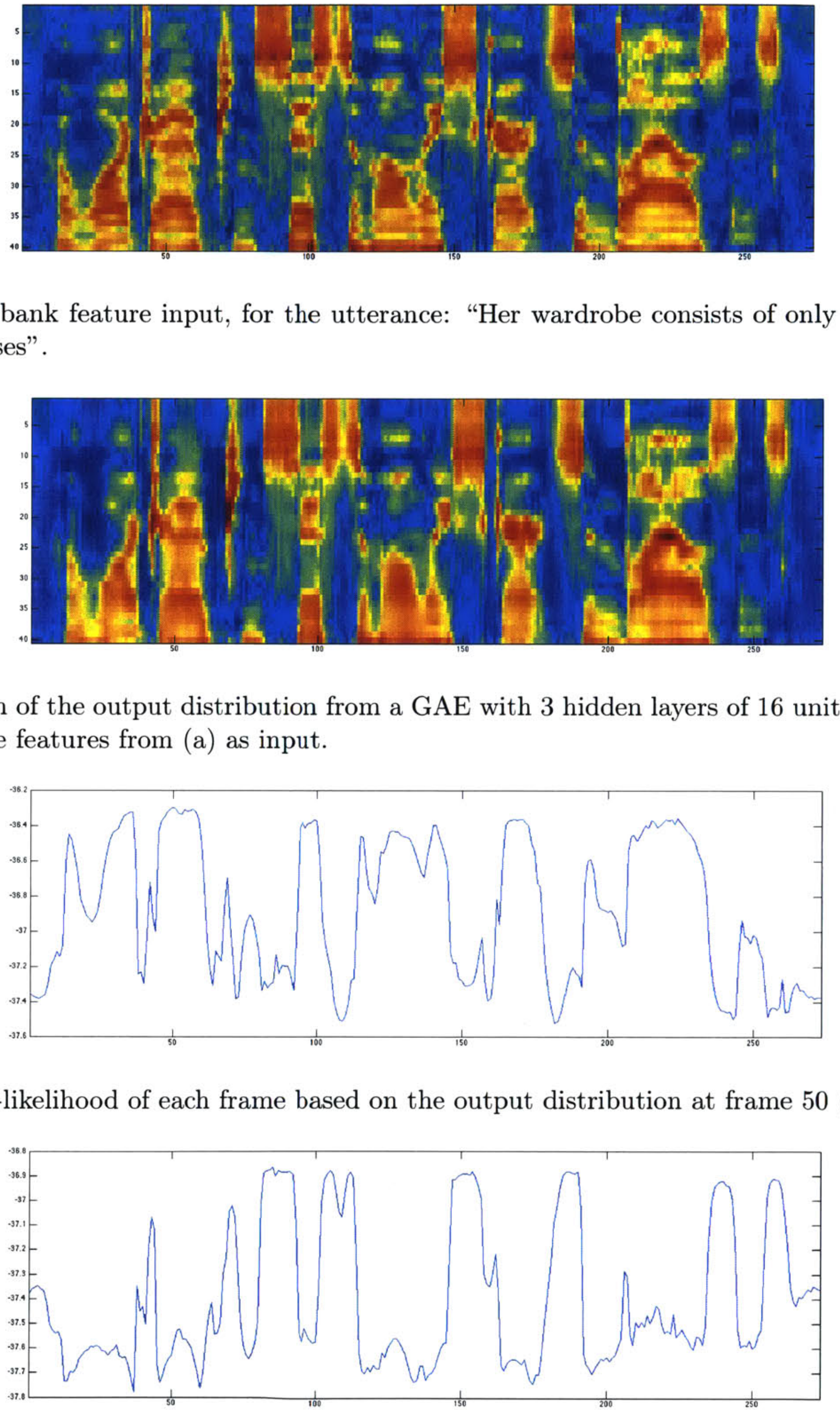
square. We use the inverse CDF of the standard Gaussian to transform each point on the grid into the latent space; the standard Gaussian was used as our prior on the latent representations. For each of these points, z, in the latent space, we use our network to get $p_\theta(x|z)$, and we show the mean of that distribution.

Figure 5-7 shows the range of frames that the VAE model is capable of representing. In the upper right-hand corner, we see frames with little energy - this matches, among others, the set of closure phonemes, shown in yellow in Figure 5-6b. In the upper left corner, we see frames with the energy concentrated in the high-frequency filterbanks. This matches fricatives like 'sh', shown in magenta in Figure 5-6b. Towards the bottom right corner, we see frames with limited energy, concentrated in the lower-frequencies; this looks like nasals, including 'm' which is in green in Figure 5-6b. Finally, in the bottom left, we have high-energy frames, with the energy especially strong in the lowest frequencies. In Figure 5-6b, this is represented by the phonemes 'aa' and 'iy,' shown in red and cyan respectively. It is a bit hard to make out, but further down, where the 'aa' representations are, the energy is highest in the lowest-frequency filterbanks. As we move up, along the left-hand side of the manifold, toward the 'iy' representations, the strongest energy also moves up slightly within the frame.

Despite covering the range of possible speech frames, it is clear that the outputs depicted in Figure 5-7 are blurred. This blurring was expected; we hypothesized that it might bring the representations of similar frames closer together, producing better spoken term discovery results. This clearly was not the case - the GAE is better than the VAE for all architectures, and both are significantly worse than the baseline architecture.

Based on the results, these models are not suited for the task/evaluation we're using them for here. However, because their output is a probablility distribution, both of these models suggest another way to compare frames within the DTW frameworks used for these evaluations. Specifically, we can compute a "similarity" between frames A and B by finding the likelihood of frame A given the output distribution generated by feeding frame B through our model. We can symmetrize this metric by averaging it with the reverse. Such a metric would be very closely related to the KL-divergence between the two output distributions, which could also be easily tested in this context. Unfortunately, implementing these distance metrics in the context of the spoken term detection evaluations is still to be completed. Nonetheless, we can perform some initial analyses.

Figure 5-8 gives a qualitative view of how such an analysis might work. In Figure 5-8a, we see the input features for one sentence in the TIMIT test set. Figure 5-8b shows the mean of the output distribution of the best-performing GAE for each of those input frames. Immediately, we can see that this autoencoder is doing a reasonable job of capturing the input. Figures 5-8c and 5-8d illustrate the potential of our proposed distance metric. For 5-8c, we've selected the 50th frame in the utterance, which falls in the phoneme 'ow' in the word 'wardrobe'. We then computed the likelihood of every frame in the utterance given the output distribution at frame 50. In this



(a) Filterbank feature input, for the utterance: "Her wardrobe consists of only skirts and blouses".

(b) Mean of the output distribution from a GAE with 3 hidden layers of 16 units each, using the features from (a) as input.

(c) Log-likelihood of each frame based on the output distribution at frame 50 ('ow')

(d) Log-likelihood of each frame based on the output distribution at frame 85 ('s')

Figure 5-8: Illustration of Gaussian autoencoder performance and proposed output distribution-based similarity metric.

figure, we can see high likelihoods assigned to the the vowels in the utterance. For Figure 5-8d, we followed the same process for frame 85, which is part of the first ‘s’ in ‘consists’. This time, we can see high likelihoods assigned to all of the ‘s’ instances in the utterance.

Chapter 6

Sequence Representations

In the last chapter, we hypothesized that the relatively poor performance of feature representations pulled from autoencoder models could come from the fact that they are failing to capture the dynamics of speech. In this chapter, we address this issue by expanding our models to represent speech segments rather than individual frames.

We can accomplish this with the same feed-forward architecture explored in the last chapter, by presenting entire speech segments all at once to our model. However, humans process speech sequentially, so it is natural to switch to models which process speech in the same way. Specifically, we will investigate recurrent encoder-decoder models for sequences of speech.

The idea of representing sequences of speech brings up an important question: how do we segment speech into such sequences. One option is to simply use fixed-length sequences that are larger than one frame; this is necessary for feed-forward architectures, which cannot handle variable lengths. For the recurrent models, we can test these same fixed length sequences, but we will also explore a more natural segmentation of speech by using the phoneme-level transcriptions. This last model is a best-case scenario test of what these models could learn if we were able to develop a perfect unsupervised segmentation method. We will discuss segmentation in more detail in the next chapter.

Model	Stack Size	Num Layers	Layer Size	P@N	EER
Filterbank	1	N/A	N/A	**50.9**	**13.1**
	3	N/A	N/A	48.0	16.0
	7	N/A	N/A	48.0	17.7
Feed-Forward	1	3	16	47.6	15.4
	3	3	16	50.5	15.7
	7	3	64	**58.5**	**14.2**

Table 6.1: Baseline and feed-forward autoencoder results for fixed-length sequences of speech, used in TIMIT spoken term discovery task. Stack size is in number of frames.

6.1 Feed-forward Networks

We can get representations for speech segments by simply stacking multiple frames of speech and using those stacked frames as the input to our autoencoder. This is the standard approach in supervised speech recognition[28] as well as unsupervised speech processing[62].

6.1.1 Model

We explore the same autoencoder models as discussed in the last chapter. We stack either three or seven frames of filterbanks, for a total of 120 or 280 input features. We again test models with one, three or five hidden layers, although we increase the tested sizes of these layers to 16, 64, and 128 because of the increase in the input dimensionality. We again use the central hidden layer of our model to extract features for use in the downstream tasks. To get the representation for a particular frame, we use the sequence with that frame at the center as input to the autoencoder.

6.1.2 Results

For this task, we have a simple baseline which uses the stacked filterbank features directly. Next, we have the feed-forward autoencoder model, using these stacked filterbanks as the input. Results from these experiments are in Table 6.1. Results are shown for the best feed-forward architecture tested for each stack size.

Two things are immediately clear from this table. First, stacking frames does not improve the results when these stacked frames are directly used as features for the spoken term discovery task. However, we get a significant improvement from stacking frames when these frames are paired with the feed-forward autoencoder framework. In addition to finally achieving an improvement over the baseline filterbank features, the best autoencoder with seven frames of input performs better at this task than some other models from the literature like [70] and [71]. It also outperforms posteriorgrams from a supervised Thai speech recognition model, although it still significantly underperforms posteriorgrams from a supervised English system[41].

6.1.3 Analysis

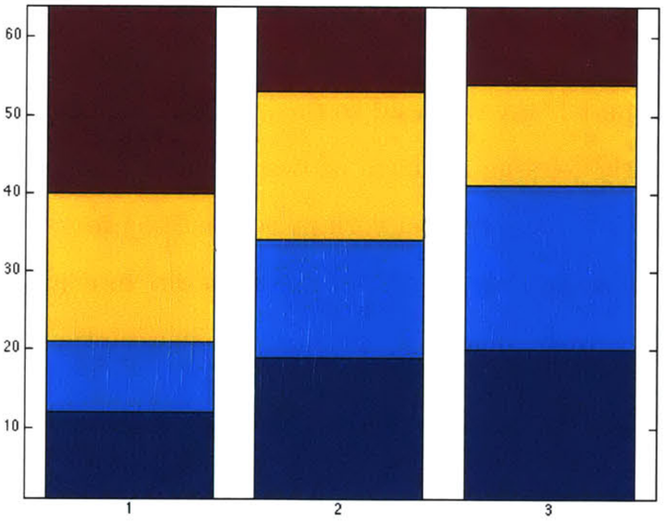


Figure 6-1: Information encoding of hidden units in overall best feed-forward autoencoder model on TIMIT spoken term discovery task. Dark blue units encode phone information, yellow units encode speaker information, and bright blue units encode both. Dark red units encode neither.

As in the previous chapter, we analyze the hidden units of a model by computing the between- and within-category variances for both the speaker and phone categories. The results of this analysis applied to the feed-forward autoencoder with seven-frame input and three hidden layers of 64 hidden units each are displayed in Figure 6-1.

Interestingly, we do not see a decrease in speaker-selective units in the higher layers, as we did before. Nonetheless, we see a consistent, significant increase in the number of phone-selective units. Despite that increase, the features from the last layer of this model result in significantly worse spoken term discovery performance than those from the middle layer - it reduces the performance down to 50.4 P@N. Clearly the number of phone-selective units is not a determiner of performance - the layer with the most phone selective units of any model tested so far in this thesis was the last layer of the model from the previous chapter, and it only achieved a P@N of 48.0. Of course, the number of phone-selective units could not be the whole story - performance must be strongly dependent on exactly how phone information is represented and how the representations of different phones relate to each other.

6.2 Recurrent Autoencoder

The feed-forward models investigated so far in this thesis are not ideal for processing speech. In the stacked-frame scenario of the previous section, which is also used in most prior work in this area, the network must learn to process each position in a segment separately, creating redundancy in the network. In computer vision, this problem has been solved with much success through convolutional neural networks[40], which are partially inspired by human vision.

For speech, we also take a cue from human cognition: speech is processed sequentially, so the natural neural network model to use in this case is a recurrent neural network. Specifically, we explore an LSTM-based autoencoder that generates a representation for a sequence of frames. We compare this directly with a feed-forward autoencoder using stacked frames.

6.2.1 Model

For this model, we use the LSTM encoder-decoder framework, described in Section 2.2 and depicted in the second model from the right of Figure 2-5. The encoder reads the input sequence one frame at a time, feeds it through D dense layers, and

Sequence length	Dense Layers	Layer Size	P@N	EER
3	0	16	41.3	18.4
		64	37.6	18.4
	1	16	29.7	25.4
		64	28.9	22.4
7	0	64	**47.6**	17.6
		128	40.6	20.5
	1	64	36.8	**16.3**

Table 6.2: Results of LSTM encoder-decoder model on fixed-length input sequences.

then feeds it to an LSTM layer. The output of the LSTM is not used, except at the end of the sequence, when it is saved and passed to the decoder LSTM. At each decoding step, the output of the decoder LSTM goes through D more dense layers before generating output. If we feed N frames into the encoder, we pass the encoded representation to the decoder N times, letting the decoder generate N frames. These N frames are then scored against the input frames.

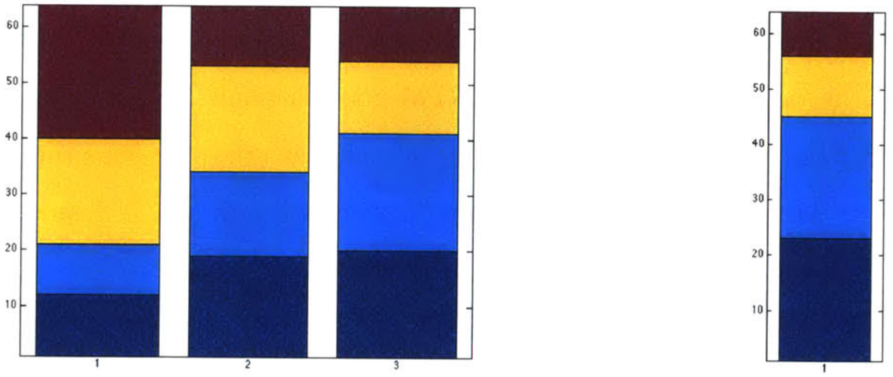
We use the hidden representation that is passed from the encoder to the decoder at the end of the sequence as our feature representation for the downstream tasks. We again follow the standard paradigm for using stacked frames - the representation of a sequence of frames is used as the feature representation for the central frame in the sequence.

6.2.2 Results

As shown in Table 6.2, the LSTM activations from the recurrent model do not perform well as features for the spoken term discovery task. One configuration produces comparable results to the original feedforward autoencoder; the rest give results that are considerably worse. However, the LSTM model is much better able to recreate the input than the feed-forward autoencoder. Using seven stacked frames as input, the average MSE of the test set using the best feed-forward model is 0.0023; the average MSE of the test set using the same size LSTM model is 0.0016.

6.2.3 Analysis

As in the previous chapter, we see that performance on downstream tasks is unrelated to the autoencoder MSE. As expected, the recurrent model is better able to model speech, in the sense that it generates outputs with significantly lower MSE than the feed-forward model on both the test and training sets. However, the feed-forward model produces features which perform significantly above the baseline on the spoken term discovery task, while the recurrent model is unable to match the baseline performance.



(a) Hidden units from feed-forward autoencoder.

(b) Hidden units from LSTM layer of recurrent model.

Figure 6-2: Encoded information from best feed-forward and recurrent autoencoder models trained on fixed-length speech segments. Dark blue units encode phone information, yellow units encode speaker information, and bright blue units encode both. Dark red units encode neither.

In Figure 6-2, we again show the hidden units from the best feedforward network, this time next to the hidden units from the best recurrent netowrk. From the perspective of the number of phone and speaker coding units, the LSTM hidden representations are comparable to the last layer of the feedforward model. Once again, this analysis does not fully account for performance on the spoken term discovery task.

One interesting trend in the recurrent autoencoder results is that smaller networks seem to perform better on this task. In ongoing work, we are repeating these experiments using models with fewer hidden units. We are also exploring ways to analyze the cell-state of the LSTM layers - there may be useful information there that is not included in the hidden representations we're working with.

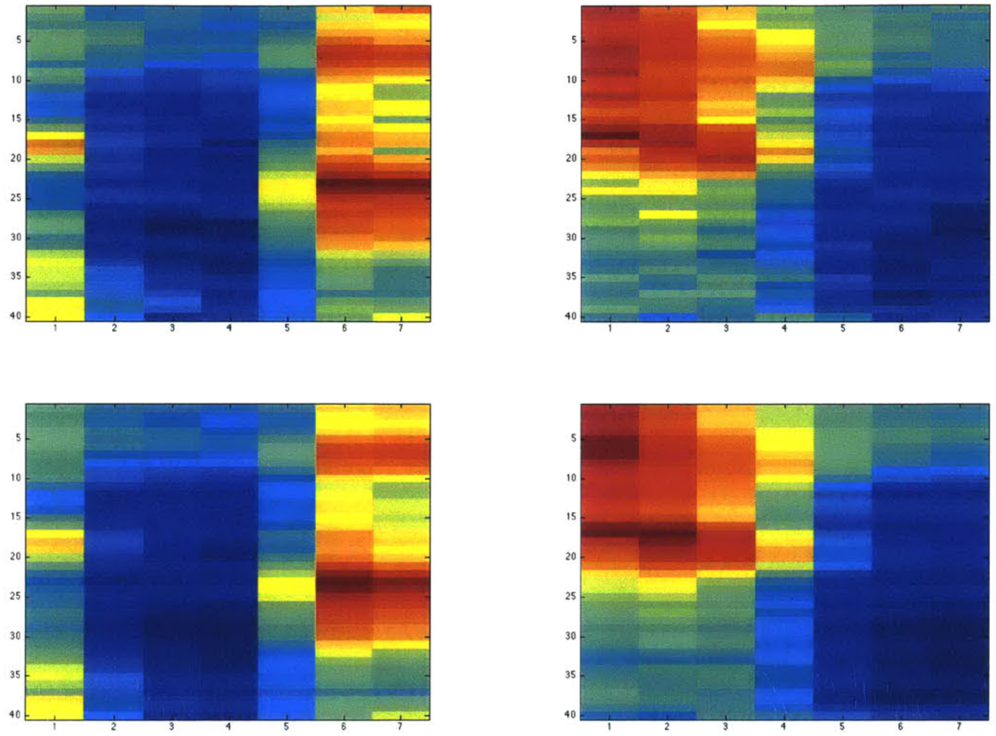


Figure 6-3: Example input speech segments (top row), with corresponding outputs (bottom row) from LSTM autoencoder.

Despite relatively poor performance on the spoken term detection task, the LSTM-autoencoder is an impressive model, in terms of its ability to model speech, and one that warrants further analysis. Figure 6-3 shows some example inputs and outputs of the LSTM autoencoder. Recall that the decoder LSTM is repeatedly given the same input; Figure 6-3 illustrates that it is still able to capture the fine-grained dynamics of the original input. As a result, the space of these learned representations must necessarily be complicated. In ongoing work, we are also exploring alternate distance metrics for comparing these representations, in the hopes of improving the spoken term discovery results.

6.3 Linguistically Meaningful Segments

We have seen that LSTM autoencoders are very powerful models. In the previous section we did not take advantage of their full potential - a key feature of recurrent models is their ability to encode variable-length sequences.

Of course, we cannot segment speech into just any variable-length sequences and expect to be able to compare them in downstream tasks. Instead, we must use a meaningful segmentation - in this case, we use the gold-standard transcription boundaries. These boundaries, which are available for the datasets studied here but typically unavailable, are treated as a best-case scenario for what an unsupervised segmentation system could produce. There are a number of ways to approach the segmentation problem, as will be discussed in the next chapter.

6.3.1 Model

For these experiments, we again use the encoder-decoder model from the previous section. However, the way we use this model to generate features is slightly different. In the previous section, we generated the features for a frame by using a sequence with that frame at the center. Here, many frames belong to the same phoneme, and, as a result, are given the same representation. We experiment with two ways of handling this: first, simply give our task several copies of the same representation in a row and second, only give our task one representation per phone. In the second case, we tag the phone representations with the timestamp at the center of the phone. Unfortunately, the spoken term discovery system we use to evaluate our TIMIT representations cannot currently handle the second option. The Zero Resource Challenge evaulation software, however, can handle both.

6.3.2 Results

For this model we must again modify our baseline to make it more competitive. Since we are giving the model access to phoneme boundaries, it is only fair to give these boundaries to the baseline model as well. To incorporate these boundaries into the baseline, we average the filterbank features across each phone.

On the TIMIT data, this averaging backfires - the phone-averaged filterbank features perform much worse on the spoken term discovery task than the original frame-level features, with a P@N of 40.1 and EER of 20.8. This is likely an effect of the implementation of the spoken term discovery system - having repeated instances of the same frames can complicate the discovery of DTW paths, depending on the exact algorithm used. Nonetheless, the best RNN features produce a P@N of 56.0 and an EER of 18.1. This performance is almost as good as the performance of the best model tested in this thesis so far, despite likely issues with the evaluation.

On the Buckeye data, averaging the filterbank features across each phone does not seem to interfere with the evaluation. Instead, when we represent each frame, we get comparable performance to the original features. When we only represent each phone, we get comparable within-speaker performance but a big improvement in across-speaker performance: from 29.24 to 27.86. Unfortunately, the phone-level models for the Buckeye data are still in progress, and so are left for future work.

6.3.3 Analysis

As these experiments are still ongoing, a complete analysis is left for future work. The fact that we were able to achieve a TIMIT spoken term discovery result far above the phone-averaged baseline is very encouraging for this line of research, and compels further analysis of these models. A preliminary analysis suggests that the feature representations produced by these models encode the length of the encoded sequence along with the phoneme information. In future analysis, we will investigate ways to normalize out this length information from the representation.

Chapter 7

Segmentation

Segmenting the speech signal into linguistically meaningful units is an important task within the field of unsupervised processing of speech. In this chapter, we explore the idea of using neural network models to solve this segmentation task. We draw from research on the problem of word segmentation from both cognitive science[55] and natural language processing[17] to propose a new idea: we train a neural network for prediction and hypothesize that the frames that are hardest to predict are the most likely to be boundaries.

7.1 Neural Nets for Prediction

We've focused on autoencoders for the bulk of this thesis because they provide a natural framework for unsupervised learning, providing a loss function based only on the input data. However, neural networks for prediction are also very well-studied, particularly in the field of language modeling. In [45], Mikolov et al. test two kinds of neural network language models: one in which a fixed context window is presented to a feed-forward prediction model, and one in which full sentences are presented, one word at a time, to a recurrent model. The authors find that both models significantly outperform a standard ngram language model, and that the recurrent models are consistently better than the feed-forward networks. There is no real analogue to ngram language models at the level of speech frames, as they require discrete input, but we can test the other two models in this context.

In language modeling, there are two standard ways of judging the quality of a language model. First, we can look at how well the model describes the test data. For probabilistic models, this is usually accomplished through a metric called perplexity. For our models, which have deterministic output, we use the MSE of the test set as a stand-in for this metric. The other way to evaluate a language model is to use it for a downstream task like speech recognition or machine translation. Here, we'll use phone-level segmentation. All experiments in this chapter were conducted on the TIMIT corpus.

7.2 Feedforward Prediction Networks

7.2.1 Model

In this section, we again experiment with the same feed-forward architectures explored in the previous chapters. This time we limit ourselves to models with one or three hidden layers, again with four, 16, or 64 hidden units each. As in Chapter 6, we now give our model stacked frames as input, but we instead ask it to produce only one frame of output, a prediction of the frame that follows the input sequence. We investigate two different-sized context windows: two frames and four frames.

7.2.2 Results and Analysis

The TIMIT phone segmentation results using feedforward models are shown in Table 7.1. The best model achieves an f-score of 70.05 using a context window of two frames and three hidden layers of 16 units each. All of the scores fall within a relatively narrow window, making all of these models better than a heuristic segmentation approach but worse than a state-of-the-art model that performs joint unit segmentation, clustering, and modeling[41].

Interestingly, the models that use two frames of context are just as good as, if not better than, the models that use four frames of context. This is in sharp contrast to

Context	Num Layers	Layer Size	Train MSE	Test MSE	F-Score
2	1	4	0.0186	0.0179	68.53
		16	0.0119	0.0120	69.16
		64	0.0097	0.0098	68.48
	3	4	0.0184	0.0179	67.63
		16	0.0115	0.0114	**70.05**
		64	0.0091	0.0095	68.34
4	1	4	0.0185	0.0178	68.92
		16	0.0118	0.0118	**69.57**
		64	0.0094	0.0101	67.86
	3	4	0.0184	0.0178	68.10
		16	0.0114	0.0113	**69.61**
		64	0.0090	0.0090	**69.55**

Table 7.1: Results of feed-forward predictor models on TIMIT phone segmentation. Context is in number of frames.

the accepted wisdom in the language modeling field, which suggests that more context is always better. There are two possible explanations for this. One possibility is that the immediate context is adequate for prediction of speech frames, and that additional context only serves to distract. The other possibility is that we do not have enough data in the TIMIT dataset to effectively learn longer-range dependencies, which are necessarily sparse. To answer this question, we have experiments in progress on the larger Buckeye corpus.

7.3 Recurrent Networks for Prediction

7.3.1 Model

For this section, we use the rightmost model from Figure 2-5. Our version of this model has one LSTM layer as well as D dense layers before and after the LSTM layer. We investigate the case where $D = 0$ and the case where $D = 1$. Once again, the number of hidden units in all hidden layers - dense and LSTM - is the same. We test models with four, 16, or 64 hidden units per layer.

During both training and testing, we feed full utterances into this model, one

Num Dense	Layer Size	Train MSE	Test MSE	F-Score
0	4	0.0666	0.1029	62.17
	16	0.0418	0.0622	65.23
	64	0.0325	0.0497	**65.86**
1	4	0.0.673	0.1041	62.34
	16	0.0434	0.0647	64.72
	64	0.0322	0.0509	65.64

Table 7.2: Results of recurrent predictor models on TIMIT phone segmentation task.

frame at a time. At each timestep, we use the LSTM output to generate a prediction for the next frame, which is compared against the actual next frame in the utterance.

7.3.2 Results and Analysis

The segmentation results for the recurrent predictor models are shown in Table 7.2. These models are significantly worse than the feed-forward models, in terms of both prediction MSE and segmentation f-scores.

Given the impressive performance of RNNs for language modeling, the poor performance of these models is unexpected. However, as we have already seen that having access to the previous four frames is no better than having only the previous two, it is unsurprising that access to the entire utterance is not useful in this context. As in the previous section, more experiments are necessary to determine the cause of these results. In addition to training on a larger corpus, we can also try training an LSTM to take two or four frames of input and subsequently output a prediction, as a closer comparison to the feed-forward models.

One other explanation for the relatively poor segmentation f-scores of the recurrent models is that their ability to exploit long-range dependencies makes them better able to predict across phoneme boundaries than feed-forward models. One piece of evidence in favor of this hypothesis are the word-segmentation results we achieve with these same models: the best feed-forward model gets a word segmentation F-score of 34.1, while the best recurrent network gets a word segmentation score of 41.0.

To test this hypothesis, we have another set of experiments in progress. For these experiments, we look at the prediction error at every point in the training of our recurrent network, and we do not back-propagate the errors from the worst predicted frames. The idea is that these frames are likely to be boundaries, so we do not want to train our model to be able to predict across them.

Chapter 8

Conclusion

8.1 Summary of Contributions

This thesis presents a survey of current research in deep learning, unsupervised learning from speech, and the intersection of the two. We narrow our focus to two specific tasks: developing linguistically meaningful representations of speech and segmenting speech into phonetic units. We explore the application of a variety of neural network architectures to these tasks, providing in-depth quantitative and qualitative analysis of their learning. Through this analysis, we make a number of novel observations with important implications for future research in this area.

In Chapter 5, we first explore the use of feed-forward autoencoder networks for generating useful representations of individual speech frames. We find that the best performance comes from models with relatively few hidden units compared to the dimensionality of the original input. Networks with few hidden units have not been explored in previous work but clearly warrant further experimentation.

In an analysis of these models, we find that the hidden units of models with multiple hidden layers encode more speaker information in the earlier layers and more phoneme information in the later layers - such results have been previously seen in models trained with weak supervision from a term discovery system, but have never before been shown in a model trained without any explicit speaker or phoneme information. This result is both surprising and encouraging for the potential of fully unsupervised autoencoder models, and suggests a line of research into even deeper models than those tested here.

We next explore a simple method for removing the challenge of speaker variation from our training data: we train models on data from a single speaker. Contrary to our original hypothesis, we find that removing these speaker variations does not improve the usefulness of the learned representations for within-speaker phoneme comparisons. We hypothesize that the source of within-speaker errors comes instead from the dynamics of speech, which can result in frames from the same phonetic unit by the same speaker having significantly different representations.

We further explore the performance of probabilistic autoencoder models on the representation learning task. We find that this performance is, overall, disappointing - worse than both the baseline filterbank features and the representations from a standard autoencoder models. However, we hypothesize that this poor performance may be due, in part, to our default distance metrics not being well-suited to comparing these representations. We suggest a new distance metric for such representations, and perform an initial qualitative analysis of its usefulness. Despite not being able to produce above-baseline performance on our downstream tasks, we are able to demonstrate significant learning within these models, showing that the representations of different phonetic categories are well-separated in the learned embedding space.

In Chapter 6, we turn our focus from learning representations of individual frames to learning representations of larger speech segments. For fixed-length segments, we test both feed-forward and recurrent models, counterintuitively finding that feed-forward models produce more effective representations. We once again hypothesize that this poor performance may be due to a mismatch between the learned representations and the default distance metrics.

We conduct some preliminary experiments into generating representations for linguistically meaningful units, defined by the gold-standard transcription boundaries. Although many of these experiments are still in progress, we do achieve some encouraging results that give hope for fruitful future research along these lines.

In Chapter 7, we present a novel method of speech segmentation based on neural network prediction models. We hypothesize that difficult-to-predict frames are likely indicative of phonetic boundaries. We find that this segmentation method is effective, producing better phoneme segmentations than a baseline heuristic method. However, we see an unexpected result: access to increased context lengths diminishes segmentation performance. This result suggests that the TIMIT dataset may not be large enough to adequately cover the space of long-range correlations in speech. We propose further experiments with larger datasets to explore this possibility.

8.2 Future Work

The experiments described in this thesis suggest a huge range of possible future work in this area. First and foremost, as suggested in the previous section, a number of our experiments call for a deeper analysis of the connection between learned representations and the distance metrics used during their evaluation. For example, we are in the process of ongoing work in developing a KL-divergence-based distance metric for the multivariate Gaussian distributions learned by probabilistic autoencoder models.

Another area for promising future research is in the incorporation of speaker information into these models. We propose a model in which speaker identity is provided to the network at both the input and at all hidden layers, so that the hidden units are not required to represent any speaker information. For a simple proof-of-concept, this speaker identity could be provided as a one-of-n representation; for a more complex model, we could use a Siamese network to learn speaker embeddings, which would allow us to apply such a network to novel speakers not seen in training.

Our experiments also suggest a variety of possible modifications to the neural network architectures we tested. We could explore deeper models, to determine whether the trend of increasing phonetic representation and decreasing speaker representation holds with more layers. For the probabilistic models, we could learn multivariate Gaussians with full-rank covariances rather than diagonal covariances, which would likely allow these networks to better model the distribution of speech. [60] developed a model that combines these two kinds of objective functions - we could easily apply this model to speech to generate representations that incorporate both past and future context around a given frame.

In the longer term, there is huge space of higher-level learning to be explored. We have only scratched the surface of the learning necessary to achieve full unsupervised speech recognition. A key piece of this learning would be to bring neural network techniques to the problem of joint segmentation and modeling of speech. The joint model described in [41] was successful on a variety of tasks, but has the potential to be further improved through deep learning.

Similarly, hierarchical modeling of speech is an area of great potential. We have seen that unsupervised learning from speech is a complex and difficult problem, and one that can benefit greatly from any additional constraints that are available. One possible such constraint is the notion of a minimal description of a speech that still allows for an accurate recreation. Imagine trying to represent an utterance with the shortest possible sequence of underlying units - these units would likely take the form of words or phrases. If each of these underlying units was also represented as a minimal sequence of lower-level units, these units might take the form of morphemes or phonemes. Such a hierarchical representation could potentially be discovered through judicious training of recurrent autoencoder models. Research from cognitive science has repeatedly suggested that humans benefit from joint learning at many levels of the language hierarchy - if the same holds true for neural networks, such a model could be significantly more effective than current models aimed only at individual, low-level tasks.

Bibliography

[1] Artificial neural network. https://en.wikipedia.org/wiki/Artificial_neural_network/media/File:Colored_neural_network.svg, 2013.

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

[3] Richard N Aslin, Jenny R Saffran, and Elissa L Newport. Computation of conditional probability statistics by 8-month-old infants. *Psychological science*, 9(4):321–324, 1998.

[4] S. Khudanpur B. Varadarajan and E. Dupoux. Unsupervised learning of acoustic sub-word units. In *Proc. ACL*, pages 165–168, 2008.

[5] Leonardo Badino, Claudia Canevari, Luciano Fadiga, and Giorgio Metta. An auto-encoder based approach to unsupervised learning of subword units. In *Proc. ICASSP*, pages 7634–7638. IEEE, 2014.

[6] Yoshua Bengio. Learning deep architectures for AI. *Foundations and trends in Machine Learning*, 2(1):1–127, 2009.

[7] Yoshua Bengio. Restricted boltzmann machines. http://deeplearning.net/tutorial/rbm.html, 2016.

[8] Yoshua Bengio, Li Yao, Guillaume Alain, and Pascal Vincent. Generalized denoising auto-encoders as generative models. In *Advances in Neural Information Processing Systems*, pages 899–907, 2013.

[9] Denny Britz. Introduction to RNNs. http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/, 2015.

[10] Ke Chen and Ahmad Salman. Extracting speaker-specific information with a regularized siamese deep network. In *Advances in Neural Information Processing Systems*, pages 298–306, 2011.

[11] Brian Cheung, Jesse A Livezey, Arjun K Bansal, and Bruno A Olshausen. Discovering hidden factors of variation in deep networks. *arXiv preprint arXiv:1412.6583*, 2014.

[12] Franois Chollet. Keras. `https://github.com/fchollet/keras`, 2015.

[13] Dan Claudiu Ciresan, Ueli Meier, Luca Maria Gambardella, and Jürgen Schmidhuber. Deep, big, simple neural nets for handwritten digit recognition. *Neural computation*, 22(12):3207–3220, 2010.

[14] Steven B Davis and Paul Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 28(4):357–366, 1980.

[15] Li Deng and Dong Yu. Deep learning: Methods and applications. *Foundations and Trends in Signal Processing*, 7(3–4):197–387, 2014.

[16] Micha Elsner, Sharon Goldwater, and Jacob Eisenstein. Bootstrapping a unified model of lexical and phonetic acquisition. In *Proc. ACL*, pages 184–193. Association for Computational Linguistics, 2012.

[17] Margaret M Fleck. Lexicalized phonotactic word segmentation. In *Proc. ACL*, pages 130–138, 2008.

[18] Abdellah Fourtassi, Ewan Dunbar, and Emmanuel Dupoux. Self-consistency as an inductive bias in early language acquisition. In *Proceedings of the 36th annual meeting of the Cognitive Science Society*, 2014.

[19] Abdellah Fourtassi, Thomas Schatz, Balakrishnan Varadarajan, and Emmanuel Dupoux. Exploring the relative role of bottom-up and top-down information in phoneme learning. In *Proc. ACL*, pages 1–6, 2014.

[20] A. Garcia and H. Gish. Keyword spotting of arbitrary words using minimal speech resources. In *Proc. ICASSP*, 2006.

[21] Raymond G Gordon and Barbara F Grimes. *Ethnologue: Languages of the world*, volume 15. SIL international Dallas, TX, 2005.

[22] Alan Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *Proc. ICASSP*, pages 6645–6649. IEEE, 2013.

[23] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proc. ICML*, pages 369–376. ACM, 2006.

[24] Karol Gregor, Ivo Danihelka, Andriy Mnih, Charles Blundell, and Daan Wierstra. Deep autoregressive networks. *arXiv preprint arXiv:1310.8499*, 2013.

[25] Frantisek Grezl and Petr Fousek. Optimizing bottle-neck features for LVCSR. In *Proc. ICASSP*, pages 4729–4732. IEEE, 2008.

[26] Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, et al. Deepspeech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567*, 2014.

[27] Mary Harper. Iarpa babel. `https://www.iarpa.gov/index.php/research-programs/babel`, 2011.

[28] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, and Brian Kingsbury. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *Signal Processing Magazine, IEEE*, 29(6):82–97, 2012.

[29] Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, and Jürgen Schmidhuber. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. *A field guide to dynamical recurrent neural networks*, 2001.

[30] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

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

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

[33] Aren Jansen, Samuel Thomas, and Hynek Hermansky. Weak top-down constraints for unsupervised acoustic model training. In *Proc. ICASSP*, pages 8091–8095, 2013.

[34] Herman Kamper, Micha Elsner, Aren Jansen, and Sharon Goldwater. Unsupervised neural network based feature extraction using weak top-down constraints. In *Proc. ICASSP*, pages 5818–5822. IEEE, 2015.

[35] Andrej Karpathy. The unreasonable effectiveness of recurrent neural networks. `http://karpathy.github.io/2015/05/21/rnn-effectiveness/`, 2015.

[36] Diederik P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, pages 3581–3589, 2014.

[37] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[38] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[39] Quoc V Le, Rajat Monga, Matthieu Devin, Kai Chen, Greg S Corrado, Jeff Dean, and Andrew Y Ng. Building high-level features using large scale unsupervised learning. In *Proc. ACL*. ACM, 2012.

[40] Yann LeCun and Yoshua Bengio. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, ·3361(10):1995, 1995.

[41] Chia-ying Lee and James Glass. A nonparametric bayesian approach to acoustic model discovery. In *Proc. ACL*, pages 40–49. Association for Computational Linguistics, 2012.

[42] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proc. ICML*, pages 609–616. ACM, 2009.

[43] S. Lowe A. Chan M. Siu, H. Gish. Unsupervised audio pattern discovery using hmm-based self-organized units. In *Proc. Interspeech*, 2011.

[44] Andrew Martin, Sharon Peperkamp, and Emmanuel Dupoux. Learning phonemes with a proto-lexicon. *Cognitive Science*, 37(1):103–124, 2013.

[45] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Proc. Interspeech*, volume 2, page 3, 2010.

[46] Hossein Mobahi, Ronan Collobert, and Jason Weston. Deep learning from temporal coherence in video. In *Proc. ICML*, pages 737–744. ACM, 2009.

[47] Céline Ngon, Andrew Martin, Emmanuel Dupoux, Dominique Cabrol, Michel Dutat, and Sharon Peperkamp. (non) words,(non) words,(non) words: evidence for a protolexicon during the first year of life. *Developmental Science*, 16(1):24–34, 2013.

[48] Boyan Onyshkevych. Low resource languages for emergent incidents. `http://www.darpa.mil/program/low-resource-languages-for-emergent-incidents`, 2014.

[49] A. Park and J. Glass. Unsupervised pattern discovery in speech. *IEEE Trans. on Audio, Speech, and Language Processing*, 16(1):186–197, 2008.

[50] Mark A Pitt, Keith Johnson, Elizabeth Hume, Scott Kiesling, and William Raymond. The buckeye corpus of conversational speech: Labeling conventions and a test of transcriber reliability. *Speech Communication*, 45(1):89–95, 2005.

[51] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, Jan Silovsky, Georg Stemmer, and Karel Vesely. The kaldi speech recognition toolkit. In *Proc. ASRU*. IEEE Signal Processing Society, December 2011.

[52] Daniel Renshaw, Herman Kamper, Aren Jansen, and Sharon Goldwater. A comparison of neural network methods for unsupervised representation learning on the zero resource speech challenge. In *Proc. Interspeech*, 2015.

[53] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

[54] DE Rumelhart, GE Hinton, and RJ Williams. Learning internal representations by back-propagating errors. *Parallel distributed processing*, 1, 1986.

[55] T. Griffiths S. Goldwater and M. Johnson. A bayesian framework for word segmentation: exploring the effects of context. *Cognition*, 112:21–54, 2009.

[56] Jenny R Saffran, Richard N Aslin, and Elissa L Newport. Statistical learning by 8-month-old infants. *Science*, 274(5294):1926–1928, 1996.

[57] Haşim Sak, Andrew Senior, and Françoise Beaufays. Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition. *arXiv preprint arXiv:1402.1128*, 2014.

[58] Thomas Schatz, Vijayaditya Peddinti, Francis Bach, Aren Jansen, Hynek Hermansky, and Emmanuel Dupoux. Evaluating speech features with the minimal-pair ABX task: Analysis of the classical MFC/PLP pipeline. In *Proc. Interspeech*, pages 1–5, 2013.

[59] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015.

[60] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhutdinov. Unsupervised learning of video representations using lstms. *arXiv preprint arXiv:1502.04681*, 2015.

[61] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[62] Gabriel Synnaeve and Emmanuel Dupoux. Weakly supervised multi-embeddings learning of acoustic models. *arXiv preprint arXiv:1412.6645*, 2014.

[63] Gabriel Synnaeve, Thomas Schatz, and Emmanuel Dupoux. Phonetics embedding learning with side information. In *Proc. SLT*, pages 106–111. IEEE, 2014.

[64] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proc. CVPR*, pages 1701–1708, 2014.

[65] Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688, May 2016.

[66] Roland Thiolliere, Ewan Dunbar, Gabriel Synnaeve, Maarten Versteegh, and Emmanuel Dupoux. A hybrid dynamic time warping-deep neural network architecture for unsupervised acoustic modeling. In *Proc. Interspeech*, 2015.

[67] Joost van Amersfoort. Variational-autoencoder. `https://github.com/y0ast/Variational-Autoencoder`, 2015.

[68] Maarten Versteegh, Roland Thiolliere, Thomas Schatz, Xuan Nga Cao, Xavier Anguera, Aren Jansen, and Emmanuel Dupoux. The zero resource speech challenge 2015. In *Proc. Interspeech*, 2015.

[69] Steve Young. A review of large-vocabulary continuous-speech recognition. *Signal Processing Magazine*, 13(5):45, 1996.

[70] Yaodong Zhang and James R Glass. Unsupervised spoken keyword spotting via segmental DTW on gaussian posteriorgrams. In *Proc. ASRU*, pages 398–403. IEEE, 2009.

[71] Yaodong Zhang, Ruslan Salakhutdinov, Hung-An Chang, and James Glass. Resource configurable spoken query detection using deep boltzmann machines. In *Proc. ICASSP*, pages 5161–5164. IEEE, 2012.