1 What I'm doing & Why it's Extremely Cool

I'm creating Neural Nets which better classify data from unseen conditions, without any explicit adaptation of model parameters or data transformations.

My approach is extremely cool because it doesn't require tons of data, it is not specific to some dataset, and it can be used to build any Neural Net (not just for speech recognition).

2 Overview of Speech Recognition

This section contains an overview of the training and testing procedures for standard automatic speech recognition (ASR) pipelines. The overview will provide the reader with a technical grounding in ASR, so that the rest of the dissertation will have some point of reference.

- **Gaussians + HMMs**
- **Neural Nets + HMMs**
- **end-to-end Neural Nets**

3 Background Literature

Here I will cover the literature relevant to working with small (or completely new) datasets. There are two main approaches, (1) adapt a model from one training dataset to a new, smaller dataset; (2) create a model that is robust enough to handle data from multiple domains.

- **Model Adaptation: (e.g. Speaker; Language)**
- **Model Robustness: (e.g. Noise; Channel)**

4 Experiments

This section contains the main contributions of the dissertation research.

This dissertation investigates training methods for acoustic modeling in the Neural Net + HMM ASR pipeline.

I aim to produce acoustic models which perform better (i.e. lower Word Error Rates) on datasets which are not similar the original training dataset.

I investiage the effectiveness of different tasks (eg. linguistic tasks vs machine learning tasks) in a Multi-task Learning framework.

4.1 Data

The differences between the training and testing data will be (1) the recording noise conditions, (2) who the speaker is, or (3) what language the speaker is using. The following table shows which data sets are used for each audio condition.

		Corpus	
		Train	**Test**
Audio Condition	**Noise**	TIDIGITS	Aurora 5
	Speaker	LibriSpeech-A	LibriSpeech-B
	Language	LibriSpeech	Kyrgyz Audiobook

Table 1: Speech Corpora

4.2 Model Training Procedure

This dissertation investigates the creation of new tasks for MTL, either using (1) linguist-expert knowledge, (2) ASR Engineer-expert knowledge, or (3) general Machine Learning knowledge.

The latter two knowledge sources are useful for buidling acoustic models, but not much else. On the other hand, the final knowledge source (general machine learning concepts) can be applied to *any* classification problem.

The three knowledge sources will be abbreviated as such:

- (LING) **Linguistic Knowledge**
- (ASR) **Traditional Speech Recognition Pipeline**
- (ML) **General Machine Learning**

Each of these categories contains a wealth of ideas, but I will consolidate each into three experiments. With three experiments for each knowledge source, my dissertation will contain nine (9) experimental conditions (for each audio condition).

Specifically, I will use the following concepts to create new tasks to be used in MTL training:

	Knowledge Source		
	LING	**ASR**	**ML**
Experiments	voicing	monophones	k-means
	place	1/2 triphones	random forests
	manner	3/4 triphones	bootstrapped resamples

Table 2: Experimental Setup

Each of these tasks will be added to a Baseline model. More specifically, the Baseline model will be a Neural Net with a single output layer (Task A), and the tasks above will be added as a second task (Task B). You can think of the tasks as simply a new set of labels for the existing data set. For example, when the Ling task of voicing is used, any audio segment labeled `[b]` will be assigned the new label `voiced`.

When these experiments will be applied to each of the three audio conditions, we get the following 30 experiments:

Data Condition	**Train Data**	**Test Data**	**MTL Training Tasks**	**Num. Exps**
Noise	TIDIGITS	Aurora 5	Basline	1
			Baseline + LING	3
			Baseline + ASR	3
			Baseline + ML	3
Speaker	LibriSpeech-A	LibriSpeech-B	Baseline	1
			Baseline + LING	3
			Baseline + ASR	3
			Baseline + ML	3
Language	LibriSpeech + Kyrgyz-A	Kyrgyz-B	Baseline	1
			Baseline + LING	3
			Baseline + ASR	3
			Baseline + ML	3
				30

Table 3: Experimental Setup

4.3 Task Creation Specifics

1. LING

   All of the linguistic knowledge tasks view the phoneme as a bundle of features.

   Using standard features from articulatory phonetics (voicing, place, and manner), the following tasks generate labels for each data point by collapsing the given phoneme labels along one of these three dimensions.

   All information from one dimension is removed from the labeled data. This forces the classifier to rely on audio signal features which do not relate to that dimension. The DNN must project the input data into a new space for classification, using only information from the other two dimensions.

   (a) VOICING
   Voicing information is removed from the data labels.

   (b) PLACE
   All place information is removed from the data labels.

   (c) MANNER
   All manner information is removed from the data labels.

2. ASR

   All the following tasks relate to the structure of the phonetic decision tree used in the traditional ASR pipeline to cluster context-dependent triphones. In GMM training the leaves of the decision tree are then assigned their own Gaussians, and in DNN training the same leaves are used as labels during training via backprop.

   The main intuition behind these experiments is that in using the decision tree labels as targets for the DNN classifier, we are performing model transfer. The decision tree and it's associated Gaussians perform classification, and we are merely training a DNN to perform that same task. So, the decision tree can be thought of as a single task for the DNN to learn.

   However, the DNN only sees the leaves of the decision tree. It doesn't see any of the branches, or any of its wonderful linguistic structure. So, in order to force the DNN to learn the information hidden in the decision tree, the following tasks are like cross-sections of the tree, slicing it from leaves up. The DNN then has to learn how to read these cross-sections, and how to map data onto each layer.

   If we slice the tree at the roots, we have the MONOPHONES. If we slice down half-way (1/2 TRIPHONES), we have more contextual information than monophones but less than full triphones. If go a little farther down (3/4 TRIPHONES), we get even more context, but less general information about the original phoneme.

   (a) monophones
   When we chop the tree at the roots.

   (b) 1/2 triphones
   Chop the tree half-way down.

   (c) 3/4 triphones
   Chopping a little further.

3. ML

   The following tasks do not make use of any linguistic knowledge or any part of the ASR pipeline. The only things needed to perform these tasks is labeled data.

   The two approaches above, leverage information about linguistics or ASR pipeline in order to force a DNN to learn structure about the data. The labeled data we want to classify

is just a set of audio features and labels. These labels contain no information about how they might relate to each other, even if that information may be exploitable and useful for classification.

In the LING and ASR sections, we are forcing the DNN to learn that phonemes have features or that certain contexts correlate to audio features.

However, a more interesting problem is how to learn this structure in a data set when we don't have that knowledge *a priori*.

The following tasks force the DNN to learn structure in the data without any knowledge about that structure. In order to do so, I make the assumption that the data does in fact have heirarchical relations. That is, I assume the labels were made by something like a phonetic decision tree, and I try to recover its structure.

(a) k-means

Standard k-means on the data, with the caveat that labels cannot be split across clusters. A first round of clustering is performed, and then all data from the same original label are shifted to the cluster with the most data points from that label. Then, centroids are recalculated, and data is re-clustered. This adapated k-means should find related data points in the same clusters. If k-means is working, we would expect to be able to recover phonemes (monophones) from the labeled triphone data.

(b) random forest

In another attempt to cluster triphones along phonetic categories, the random forest procedure works as follows: (1) take a random subset of the labels, (2) train a random forest with all data points associated with those labels, (3) re-classify all the rest of the data with the new random forest. In this way, we will reduce the number of labels (eg. out of 2,000 triphone labels I choose 500), and classify unseen data as its closest seen data point.

(c) bootstrapped resamples

In this approach, new labels are not generated at all. The separate tasks for the DNN are just different samples of the data.

Some sub-samples may exhibit a more useful decision plane than others, and if we randomly subsample for multiple tasks, the different decision planes will all have something in common. The individual peculiarities of one sub-sample will have to be ignored for the DNN to perform well on all tasks.