Incorporating Weak Statistics for Low-Resource Language Modeling

by

Scott Novotney

A dissertation submitted to The Johns Hopkins University in conformity with the requirements for the degree of Doctor of Philosophy.

Baltimore, Maryland

February, 2014

© Scott Novotney 2014

All rights reserved

Abstract

Automatic speech recognition (ASR) requires a strong language model to guide the acoustic model and favor likely utterances. While many tasks enjoy billions of language model training tokens, many domains which require ASR do not have readily available electronic corpora. The only source of useful language modeling data is expensive and time-consuming human transcription of in-domain audio. This dissertation seeks to quickly and inexpensively improve low-resource language modeling for use in automatic speech recognition.

This dissertation first considers efficient use of non-professional human labor to best improve system performance, and demonstrate that it is better to collect more data, despite higher transcription error, than to redundantly transcribe data to improve quality. In the process of developing procedures to collect such data, this work also presents an efficient rating scheme to detect poor transcribers without gold standard data.

As an alternative to this process, automatic transcripts are generated with an ASR system and explore efficiently combining these low-quality transcripts with a small amount of high quality transcripts. Standard n-gram language models are sensitive to the quality of the highest order n-gram and are unable to exploit accurate weaker statistics. Instead, a log-linear language model is introduced, which elegantly incorporates a variety of background models through MAP adaptation. This work introduces marginal class constraints which effectively capture knowledge of transcriber error and improve performance over n-gram features.

Finally, this work constrains the language modeling task to keyword search of words unseen in the training text. While overall system performance is good, these words suffer the most due to a low probability in the language model. Semi-supervised learning effectively extracts likely n-grams containing these new keywords from a large corpus of audio. By using a search metric that favors recall over precision, this method captures over 80% of the potential gain.

Thesis Committee

Prof. Aren Jansen, Prof. David Yarowsky and Prof. Sanjeev Khudanpur (Advisor)

Acknowledgments

Thank you to my advisor Sanjeev Khudanpur. His guidance saw me through and gave me the confidence to take a breath, figure things out for myself, and move forward. Thank you to Rich Schwartz and Owen Kimball, who encouraged me to pursue a Ph.D. in the first place and had the patience as I wandered my way through. I've come to know the voices of Sanjeev, Rich and Owen very well these last five years on our weekly conference call/practice defense. Thank you to all the students at CLSP for pursuing so many different interests and making it a great community. Annie, Carolina, Keith, Michael, Mike, Omar, and Puyang especially kept me sane.

Last but not least, I couldn't have done it without coffee. My deepest thanks to the late Fred Jelinek for insisting on an espresso machine at CLSP, among his many contributions to this field.

Dedication

To my wife, Cristy, thank you for your love and understanding of language models. She met me at the beginning, stuck with me through it all and I'm lucky that she will be with me for the adventure ahead.

To my parents, who have taught me so much more than reading and mathematics, thank you for your inspiration and support.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The inspiration for this dissertation is the operational need to deploy automatic speech recognizers for domains with very limited resources. Many languages and diglossia, such as conversational Arabic or dialectical Hindi, lack the large electronic corpora available to build state of the art recognizers. And new tasks within a language constantly emerge, requiring in-domain transcription of that new resource condition. Current technological solutions require that for the target language and domain, the developer provide with tens of hours of transcribed audio and hundreds of thousands of tokens of text. These resources are used to estimate the so called acoustic and language models. But these resources are expensive and time-intensive to obtain and there is a need to make do with less.

This dissertation assumes that for any task that requires automatic speech recognition, there must be an abundance of audio in need of transcription. This audio has the potential to usefully augment the small amount of in-domain transcripts available. The success of semi-supervised *acoustic modeling* demonstrated that as little as one hour of manual transcripts was sufficient to deploy an effective automatic speech recognition (ASR) system in a new domain [1]. Yet the other half of the speech recognition equation, *language modeling*, has not significantly benefited from semi-supervised methods.

Ideally, the goal is to estimate a language model from a **large amount** of **in-domain** and **accurate** samples. When one of those three conditions are missing, then the task becomes *low-resource* language modeling. Initial language modeling work assumed a small amount of in-domain text was available. For instance, the Brown corpus [2] has only one million tokens (small amount, in-domain, accurate). Later work considered using a large amount of out-of-domain and accurate text under the umbrella of domain adaptation (large amount, out of domain, accurate). This dissertation considers the final combination: a large amount of in-domain, but inaccurate samples.

This new resource condition arises when a low-quality transcriber (either human or automatic) provides poor quality labels. These labels are from the domain, but are untrusted. And with these noisy labels arises a natural set of questions: How should one use this pool of data in conjunction with labels that *are* trusted? Can predictive power be traded off for more reliable statistics that are robust to transcription errors? Does knowledge of a downstream task improve the quality of semi-supervised estimation?

The fundamental task of this dissertation is to estimate a probability distribution from a variety of weak constraints about a domain. The distribution of interest is a language model for use in automatic speech recognition. These statistics may be a small sample of accurate transcripts, a large sample of inaccurate transcripts, accurate, but weakly predictive statistics, or constraints about the nature of the end task. Incorporating these weak signals requires moving beyond conventional back-off language models to a log-linear language model. This model provides a probabilistic framework through which these weak constraints and more can be encoded. The following chapters will detail a series of best practices for building a language model with a limited budget for use in automatic speech recognition.

1.1 Problem Description

Large Vocabulary Continuous Speech Recognition (LVCSR) is one of the most difficult subtasks within automatic speech recognition. Examples include automatic closed captioning of broadcast news, transcription of lectures, or call center phone calls. In addition to unique acoustic difficulties (noisier environments, lower frequency range of telephony audio, etc. . .), LVCSR has a massive search space of possible labels. *Large Vocabulary* speech recognition allows for tens of thousands of words and more to appear in the recognition output. *Continuous* speech recognition requires transcription of word sequences, not just isolated words. Contrast this task with one of the most common interactions with ASR - isolated digit recognition with only ten possible labels.

Language models are critical in tackling this huge search space of possible labels. Beyond distinguishing between homonym pairs (*bow* vs *bough*), language models help prune the search space during recognition and offer discriminative information complementary to the acoustic model. A language model provides an opinion on how well a hypothesis matches some domain. And all state of the art language models offer this opinion by computing the likelihood under a statistical model estimated from training data.

Language models benefit from ever larger amounts of training samples. For simpler speech recognition tasks, not many samples are required because the hypothesis space is relatively small. Isolated word recognition needs only individual word frequencies. Phonetic recognition, while continuous, has a very small vocabulary size. However, LVCSR domains do not saturate since the space of events is infinite. Since humans continually produce novel word sequences, no amount of finite training data will be sufficient to see every possible sentence. To overcome this, language models compare relative frequencies of word subsequences. And the longer the length of these subsequences (better known as n-grams) the better the predictive power of the model. Thus more training data results in more training samples either of rare words or of rare n-grams.

Figure 1.1 shows the reduction in cross-entropy (a measure of language modeling predictive power) as a function of the number of training tokens. Different amounts of training data from 100 to 20 million tokens were sampled from conversational English transcripts. Simple unigram language models (commonly known as a bag of words model) saturate at around the 10,000 token mark. But it is clear that even bigram models are not saturated with 20 million tokens, let alone the more predictive trigram and five-gram models typically used in state of the art language models. These 20 million tokens were manually transcribed at a rate of 20 times real time - requiring over 40,000 person hours.

Language models are not task independent - there is no one *English* model, *French* model or *Arabic* model that performs well across all tasks. Each task (such as voice mail transcription or lecture data) has a unique vocabulary and changes relative word frequencies. This effect can result in dramatic differences in language model power. For instance, the

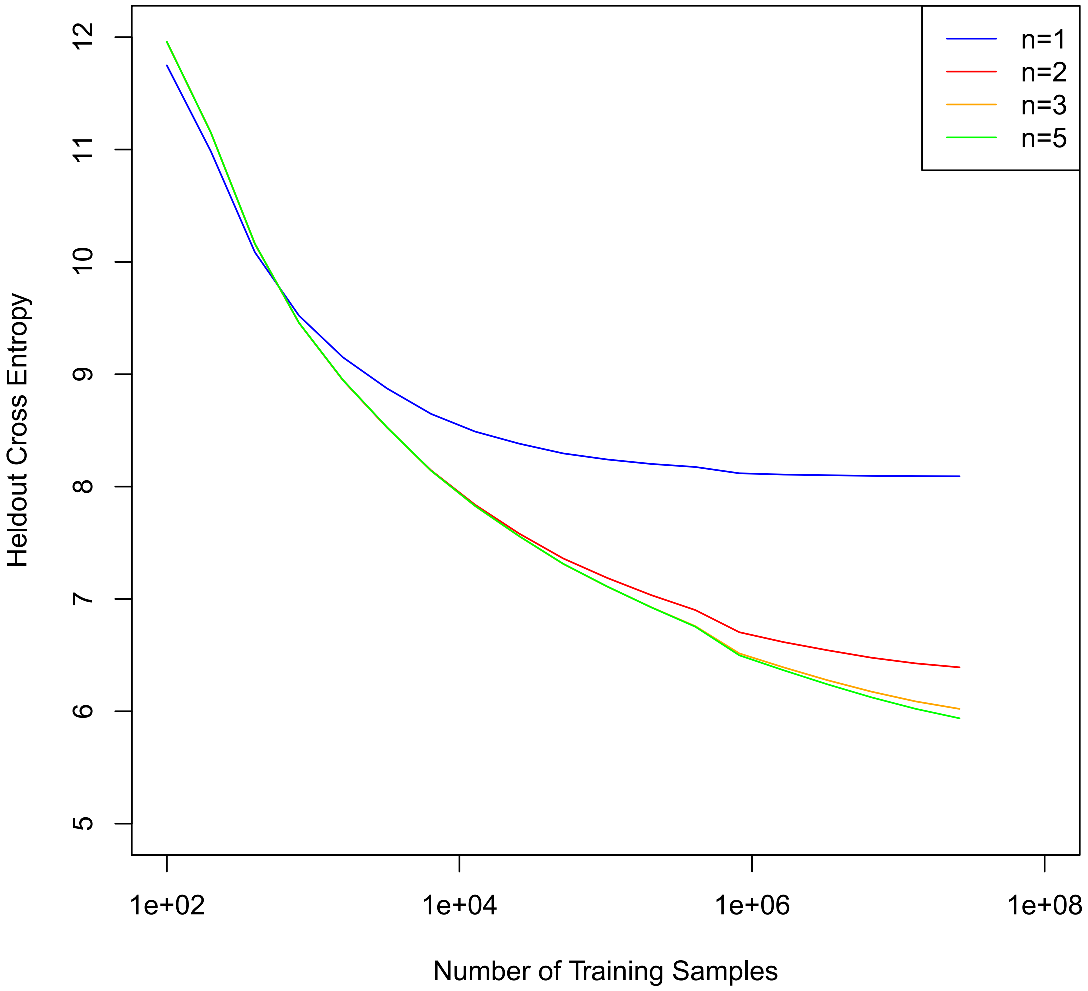

Figure 1.1: *Language Modeling Benefits from More Data* - Held-out cross-entropy (a measure of language modeling performance) decreases as language models are trained on more tokens from conversational English transcripts. Unigram language models saturate at around 10K tokens, but higher order *n*-grams do not.

LDC provides transcripts of conversational Levantine Arabic transcripts (LDC2006S29). However, it actually consists of four distinct *sub*-dialects Jordanian, Levantine, Lebanese and Palestinian which are statistically very different. Building a language model on one sub-dialect and testing it on another increases perplexity (defined in Section 2.5) by 50% on average. Even worse, 14M tokens of Modern Standard Arabic, a formalized Arabic dialect, is significantly worse on a Levantine test set than 7,000 tokens of Levantine Arabic. The best data for language model estimation will always be in-domain transcripts.

	Jordanian	**Lebanese**	**Levant**	**Palestinian**	**All**
Jordanian	**320**	500	340	445	405
Lebanese	545	**280**	610	350	435
Levant	370	520	**340**	465	410
Palestinian	425	330	470	**294**	380
All	347	285	385	284	**330**

Table 1.1: *Perplexity on Levantine Sub-Dialects* - The conversational Levantine corpus actually consists of four different sub-dialects, each drastically different from the rest. Building a language model on one dialect and testing on another reduces performance by 50% on average. Separating the sub-dialects out decreases average perplexity by 10%.

Unfortunately, the only source of training data for many LVCSR domains comes from time consuming and expensive manual transcription. Estimates for high-quality transcript range from 20 times slower than real time up to 100 times slower, depending on the task [3]. Accurate transcription of hard audio requires multiple listening passes, transcriber mediation and additional quality control. These additional steps add up to significant investments of human labor.

In other cases, such as voice search, the text may be known, but the audio needs eliciting. Recent work in deploying Arabic and Cantonese voice search systems details the laborious effort and “hands on” effort [4] [5]. Special effort was made to collect a variety of acoustic environments, speakers and dialect inflections. Rapid and inexpensive deployment for rare languages and domains is impractical with such steep costs. And while other domain data may be available in the language of interest, it may be widely mismatched, as in the case of Modern Standard Arabic to conversational Arabic.

Key to this dissertation is the assumption that any task which requires automated processing will have abundant amounts of untranscribed speech. Unfortunately lacking in labels, the speech nonetheless presents a useful opportunity to improve machine learning performance. By trading off transcription quality for cost (in time and money), it is possible to generate inexpensive labels for this large corpus of data. Humans can be given minimal training and instructed to value time over transcription quality (e.g. inter-labeler disagreement). Even further, inexpensive labels can come from an automatic classifier, but at the cost of even higher error rates.

Prior work (detailed extensively in Section 4.1.1) has used the output of inexpensive transcribers essentially as is. In the case of automatic classifiers or redundant manual transcripts, transcripts were weighted by the intuitive expected counts of alternate word hypotheses. Is this all that one can hope to extract from this noisy signal? If so, then the effectiveness of low-resource language modeling is left to the whim of the transcription quality. Might there instead be signals in the noisy output robust to transcription error? How does one quantify that robustness and incorporate these signals into a language model?

Formally, the task is to estimate a marginal distribution $P(Y)$, where the random variable Y is a sequence over a vocabulary $\mathcal{V}$, which is assumed to be fixed and finite. There exists some joint distribution $P(X, Y)$ from which a large sample of observations $\{x_1, x_2, \ldots, x_N\}$ are drawn. However, instead of also observing a corresponding $\{y_1, y_2, \ldots, y_N\}$ for each x_i,there is a separate *posterior* distribution $P(Y|x_i)$ for each sample. These posteriors come from an imperfect transcriber, either human or automated and capture the uncertainty about the true label associated with each x_i. There may also be available a smaller set of observations $\{y_1, \ldots, y_M\}$ whose labels are trusted with M much smaller than N. The question then arises, how best should the posteriors $P(Y|x_1) \ldots P(Y|x_N)$ be utilized in conjunction with $y_1, \ldots y_M$ to estimate $P(Y)$?

1.2 Proposed Solution and Road Map

This dissertation improves upon the state of the art by trading off predictive power for increased robustness. Instead of placing all hope on improving estimates of the standard n-gram statistics, this work incorporates weaker domain knowledge into language modeling. This knowledge, which would be superfluous given accurate transcripts, is easier to estimate from a small amount of in-domain data or is robust to high transcription error.

Standard language models, which smooth counts and then interpolate relative frequencies, require accurate estimation of the highest order n-gram. Instead, this work uses a log-linear language model for semi-supervised estimation, which is competitive with state of the art smoothing techniques for supervised estimation. The log-linear framework motivates a principled method of MAP adaptation to best use noisy n-gram statistics in conjunction with a small set of n-gram statistics. Most importantly, it allows for a variety of weak domain knowledge to be encoded through marginal class constraints.

- Chapter 3 first considers how best to allocate a small budget for manual transcription with the goal of deploying an LVCSR system. Non-expert transcription provides vast savings despite nearly 25% disagreement with professionals. It is better to collect *more* data, not *better* data. Additionally, non-experts can efficiently be used to rate other non-experts in the absence of gold standard data.

- Chapter 4 then uses an automatic classifier to produce transcripts for *semi-supervised* language model estimation. With error rates double that of Chapter 3, performance is modest at best. This dissertation shows that standard back-off language models require improved statistics of the highest order n-gram and are unable to benefit from accurate lower order statistics. Instead, a Bayesian framework improves semi-supervised estimation with a log-linear language model. Finally, this dissertation introduces *marginal class constraints* as a principled and flexible way of encoding domain knowledge of *transcriber error*.

- Chapter 5 introduces a weak constraint over the *application* of the language model. Instead of transcription accuracy over all words,the focus is on search performance for words which never appear in training. Semi-supervised language modeling dramatically improves search performance thanks to these two constraints. If additional human labor is available, this work proposes a method of *directed transcription* that combines semi-supervised learning with manual labeling.

Chapter 2

Background

2.1 Overview of Automatic Speech Recognition

State of the art automatic speech recognition is dominated by statistical modeling. Applications ranging from digit recognition to voice mail transcription are formulated in a classic framework from information theory - *the noisy channel model.* Under this model, as illustrated in Figure 2.1, a human thinks of some word sequence, $\mathbf{W}$, which is then encoded into acoustic vibrations by traveling first through their vocal chords, mouth and then on through the air and possibly electronic media such as a telephone wire. This acoustic signal, $\mathbf{X}$ is then received by a decoder, which then produces the best guess of the original word sequence $\mathbf{W}$.

The work of ASR research lies in creating and improving the decoder of Figure 2.1, which in all modern system utilizes statistical models estimated from speech data [6]. It is tasked with finding the most probable word sequence $\hat{\mathbf{W}}$ given the acoustic signal (or evidence) by computing the posterior probability $P(\mathbf{W}|\mathbf{X})$, where

$$\hat{\mathbf{W}} = \arg\max_{\mathbf{W}\in\mathcal{V}^\star} P(\mathbf{W}|\mathbf{X}) \propto \underbrace{\arg\max_{\mathbf{W}\in\mathcal{V}^\star}}_{\text{decoding}} \underbrace{P(\mathbf{X}|\mathbf{W})}_{\text{acoustic model}} \times \underbrace{P(\mathbf{W})}_{\text{language model}} . \tag{2.1}$$

This factors the estimation task into the likelihood $P(\mathbf{X}|\mathbf{W})$ and the prior $P(\mathbf{W})$. While there are many components to a full LVCSR system, the three main ones are the acoustic model, $P(\mathbf{X}|\mathbf{W})$, language model, $P(\mathbf{W})$, and the decoder.

Acoustic models compute the *likelihood* of an acoustic observation being generated by a given word sequence. Language models compute the *prior* probability of the word sequence appearing in the target domain. Finally, the decoder uses the acoustic and language model as given inputs to *efficiently search* the space of word hypotheses.

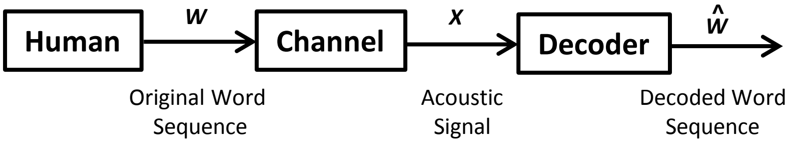

Figure 2.1: Noisy Channel Model of Speech Recognition

The following sub-sections 2.1.1, 2.1.2 and 2.1.3 briefly cover the broad topics of feature extraction, acoustic modeling and decoding. Section 2.2 more thoroughly explores the focus of this dissertation, language modeling. What follows is not a discussion of all possible LVCSR models, but the solid foundation seen in most state of the art systems.

2.1.1 Feature Extraction

The goal of feature extraction is to convert the acoustic waveform into a sequence of real-valued, multi-dimensional vectors that capture variations in the short-term spectral energy distribution over time. A feature vector is typically extracted as follows[1]. Short segments of the audio, usually 25 ms in duration, are windowed and overlapped to yield a sequence of *frames* once every 10 ms. The Discrete Fourier Transform (DFT) is applied to each frame, and a time-varying power spectrum is obtained by computing the squared magnitude of the DFT of each frame. While it is possible to use the power spectra directly as features for acoustic modeling, speaker-invariant information is better captured by a transformation that attempts to separate the source and vocal tract configuration from the waveform. The most common features based on this principle are the *Mel Frequency Cepstral Coefficients* [7]. After computing the power spectrum of a frame, spectral energies are binned according to the Mel-scale (which pitch perception in humans) and the logarithm is applied. Finally, the discrete cosine transform is applied to the vector to yield the Mel cepstrum (a reverse of spec-trum). The final feature vector is 39 dimensions, corresponding to the first 13 coefficients of the Mel cepstrum along with the first and second derivatives.

Many other transformations can be applied such as Linear Discriminant Analysis [8], mean and variance normalization and vocal track length normalization [9], all with the goal of normalizing the original audio so that it is better suited for pattern recognition. Other common techniques for feature extraction rely on data-driven methods which take the labels into account. For instance, a multi-layer perceptron model can itself take as

[1] Thank you to my colleague Michael Carlin for assistance with this explanation.

input MFCCs and output a posterior distribution over phones, which is then used as the input for an acoustic model [10]. Regardless of the method, the goal is still the same: to transform the acoustic waveform into a fixed-rate sequence of continuous feature vectors more amenable to statistical learning.

2.1.2 Acoustic Modeling

Acoustic models compute the likelihood of a sequence $\mathbf{X}$ of acoustic feature vectors given a particular word sequence, $\mathbf{W}$. Since both $\mathbf{X}$ and $\mathbf{W}$ are sequences, the likelihood $P(\mathbf{X}|\mathbf{W})$ is modeled by further hypothesizing a latent a sequence of *states*, with each acoustic vector emitted by one state. The resulting model for $P(\mathbf{X}|\mathbf{W})$ is commonly referred to as a *hidden Markov Model* (HMM). State-of-the-art acoustic modeling uses *quinphones* as states. The acoustic realization of a phoneme differs depending on the phonetic context around it - vowels may become neutral, voiced may become unvoiced, etc... Therefore, each phoneme is sub-divided into as many as $|\mathcal{P}|^4$ different classes based on the two preceding and two succeeding phonemes, where $\mathcal{P}$ is the phone set (typically around 30 to 50 phonemes). A phoneme in a particular ±2 phoneme context is called a quin-phone. This is much too large a space to see enough samples in a speech corpus, so quinphones are *clustered* depending on *phonetic questions* [11]. These questions consider broad phonetic categories and for example, may cluster the phonemes /t/ and /d/ together when they are preceded by a vowel and followed by a fricative. The number of clusters is often a function of the training corpus size, with ranges falling from 500 to 10,000 unique clusters. quinphones are typically further divided into a sequence of five states, each corresponding to one frame of output. This small HMM structure (the same across quinphones) allows skipping between states and self-loops, meaning quinphones last between 10ms-50ms.

Once the HMM state space has been decided, the acoustic likelihood of a particular frame x_i and state s_i is

$$P(x_i|s_j) = \sum_{k=1}^{N} \pi_k^j \cdot \mathcal{N}(x_i; \mu_k^j, \Sigma_k^j) \tag{2.2}$$

where N Gaussians each have mean μ_k^j, variance Σ_k^j over a 39 dimensional density function and mixture weight π_k^j. This *Gaussian mixture model* (GMM) is well suited for acoustic events due to their multi-modal behavior. Gaussian mixture models allow another method of parameter sharing: *tied mixtures*. The set of Gaussians are shared across all states and are state independent. However, the mixture weights are *state*-dependent. This increases the number of samples per Gaussian and reduces the number of parameters to estimate per state (N v. $39 \cdot 2N + N$).

Now that states have an emission probability $P(x_i|s_k)$, the remaining piece of the acoustic model is estimation of the state sequence using a *Hidden Markov Model* (HMM). The acoustic model computes the likelihood of a sequence of feature vectors generated by a *sequence* of acoustic states. With one state required per vector and potentially hundreds of thousands of states, naive search of this state space is intractable. Key to the HMM is that the prior probability of observing one state only depends on the previous state. The most likely state $\hat{s}$ at time-step i given acoustic vector x_i and the previous states $s_1, \ldots s_{i-1}$ is efficiently found by making a Markov assumption that

$$P(s_i|x_i, s_1, \ldots, s_{i-1}) = \underbrace{P(x_i|s)}_{\text{emission}} \cdot \underbrace{P(s|s_{i-1})}_{\text{transition}}, \tag{2.3}$$

so that,

$$\hat{s} = \arg\max_{s} P(s|x_i, s_1, \ldots, s_{i-1}) = \underbrace{P(x_i|s)}_{\text{emission}} \cdot \underbrace{P(s|s_{i-1})}_{\text{transition}} . \tag{2.4}$$

Since state-level alignments are almost never available, *training* of an acoustic model is semi-supervised. Typically, a training corpus of audio and transcripts is aligned only at a show level or conversation side. To compensate for this, acoustic model training uses two iterative techniques. First, successively finer grained models are trained, often starting with speech activity detection (SAD) [12] with only a few dozen states. Later models may move to one phone-per-state, eventually adding tri- and quinphones. The goal is to provide better initialization to the next model. For instance, a SAD system can filter out the starting and ending silence, ensuring the phone model does not train on non-speech acoustics.

The second iterative strategy is classic to speech recognition and is known as the *Baum-Welch algorithm* [13]. This is an application of Expectation-Maximization to Hidden Markov Models and guarantees increased likelihood of the training data despite not having good initial guesses of the state sequence. After maximum likelihood estimation, *discriminative training* treats the model as a discriminative classifier and directly minimizes error rate instead of maximizing likelihood. These techniques have shown sizable gains and are standard in modern recipes [14].

Other models are slowly supplanting the HMM-GMM acoustic model. The Markov assumption need not be made (resulting in a Conditional Random Field [15]) and recent research in deep neural networks has replaced acoustic likelihood computation of a GMM [16]. Nonetheless, this section outlines the computation of the acoustic likelihood $P(\mathbf{X}|\mathbf{W})$

that will give state of the art performance. The final remaining piece combines the prior probability $P(\mathbf{W})$ to efficiently search the huge hypothesis of word sequences.

2.1.3 Decoding

Intelligent decoding is required because the hypothesis space of possible word sequences is massive. Quickly and efficiently finding the most probable word sequence is based on *Viterbi decoding* which is a dynamic programming algorithm to find the most likely state sequence of an HMM. Similar to acoustic model training, decoding uses an iterative strategy with progressively finer-grained models. The motivation is to prune unlikely paths from the search space with a quicker, but coarser, model. The first *forward* pass finds probable word boundaries [17]. The second *backward* pass uses Viterbi decoding to find the most probable word sequence. *Lattices* are used to capture the uncertainty of the backward-pass models. This data structure is an acyclic directed graph which compactly represent alternate utterance hypotheses. Together, the forward and backward passes, generated with weaker models, produce a much smaller search space. A more complex model (with longer context dependent phonemes and larger language model states) then *re-scores* the lattice and extracts the 1-best output. This hypothesis is used for *speaker adaptation* [18] which modifies the original acoustic model to better match the automatic transcript. All three passes - forward, backward and re-scoring - are repeated with the adapted models to result in the final transcript. Decoding has a variety of parameters which trade off between speed and accuracy. It can be as fast as 100 times faster than real time (able to decode 100 hours of speech in one hour), real time, or many times slower than real time with higher resulting accuracy, depending on the amount of computing resources available for

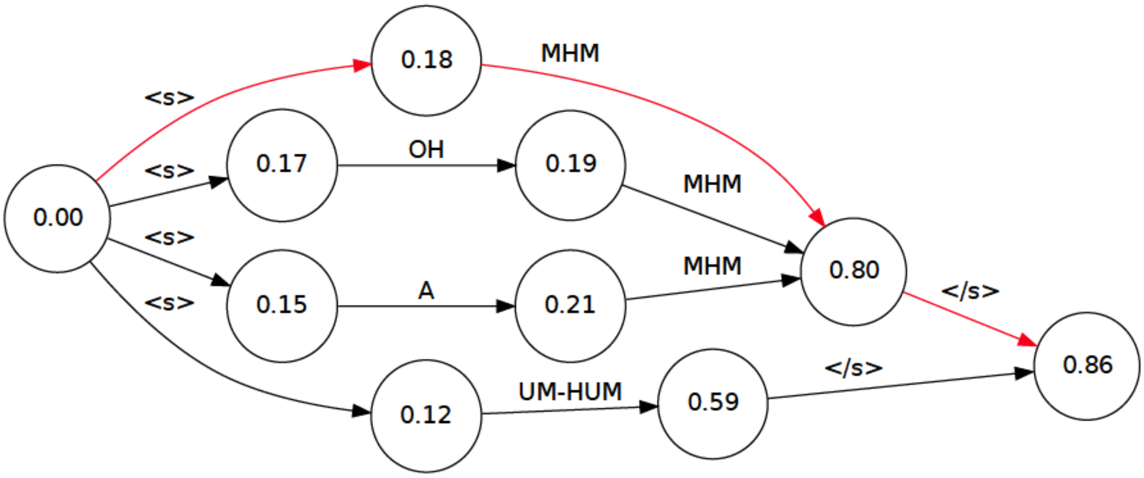

Figure 2.2: *An example lattice for the utterance "UM-HUM"* - Each node is a time marker (starting at 0 seconds and ending at 0.86 seconds) and each arc represents a different output token (utterance markers <s> and </s> included).

indexing [19]. Typical recognition speeds are highly dependent on the task. Real time voice search requires real time decoding and efficient indexing of very large corpora is typically 50 times faster than real time. The LVCSR system used in this dissertation is ten times slower than real time.

Pass	**Unadapted**	**Speaker Adapted**
Forward	52.6	41.0
Backward	46.9	35.4
Re-score	44.1	33.6

Table 2.1: *Example WERs of Multi-pass Decoding* - Starting with the unadapted quick-match forward pass at 52.6% WER, initial WER decreases to 46.9% for full Viterbi decoding. Re-scoring with larger acoustic models decreases WER further to 44.1%. Unsupervised speaker adaptation provides a sizable gain and another round of decoding produces the final WER of 33.6%.

2.2 Language Modeling

Language models compute the likelihood that a word sequence $w_1, \ldots, w_n = w_1^n$ was drawn from some training corpus representative of a domain as

$$P(w_1, w_2, \ldots w_n) = \prod_{i=1}^{N} P(w_i | \underbrace{w_1, \ldots, w_{i-1}}_{\text{history}}) \, . \tag{2.5}$$

Since language is ever evolving, novel *histories* (or words sequences) will emerge which were unseen in the training corpus. To allow for these novel events to occur with non-zero probability, some independence assumption must be made to reduce the space of seen histories. The most popular is the Markov assumption, which essentially ignores the influence of words which occurred too far back. For example, a *tri*-gram model collapses word sequences which have the same two preceding words, so the likelihood of seeing word w_i is truncated to

$$P(w_i|w_1, w_2, \ldots, w_{i-1}) \cong P(w_i|w_{i-2}, w_{i-1}) \, . \tag{2.6}$$

Stronger independence assumptions, such as unigram models of language, require a smaller number of samples to be well estimated, but have less predictive power. Weaker assumptions – such as a whole sentence model in the limit – have strong predictive power, but high bias to the training data. Language modeling, like all statistical tasks, is a trade-off between predictive power and well-estimated distributions.

For many language modeling tasks, data sparsity is not a concern. Copious amounts of training data are sometimes available for the construction of large n-gram models with five or six word histories. Language models in machine translation now use the entirety of the web for training [20]. Tasks for new domains in ASR can often leverage existing corpora for language modeling [4]. For instance, broadcast news recognizers train language models on billions of words of newswire corpora.

However, there do exist situations that lack sufficient amounts of training data for reasonable performance. Automatic speech recognition suffers from this task in two ways. First, diglossia such as Arabic, Hindi and Chinese sometimes do not have standardized orthography or lack available electronic resources on the web [21]. Hundreds of millions of people speak these languages daily through electronic mediums like telephony and broadcast media. Yet the written language is significantly different from their oral communications. For these domains, the only source of training samples is from expensive and time consuming manual transcription of the spontaneous colloquial speech. Second, new domains in need of automatic processing continually emerge. One recent application of speech recognition on a mobile platform cited 131 unique domains [22] – none of which existed before 2008. While these new domains may be from a resource rich language, both the vocabulary and relative word frequencies differ from existing corpora, again requiring in-domain transcription. Much of language modeling research has investigated efficient ways to use alternate data sources to improve language modeling in a new domain.

2.2.1 Smoothing

Since the space of possible word sequences is infinite, but training data finite, a language model must be "smoothed" from the maximum likelihood estimate to allow novel sequences. The simplest method proposed, derived from actuarial research, uniformly adds a small amount of mass to any event unseen in training data [23]. Better methods build on this basic technique by balancing between the predictive power of higher order n-grams and the accurate frequency estimates of lower orders [24–27]. The first class of smoothing heuristics, starting with Turing and Good [25], assumes that events seen in training are most likely over-estimated and thus rare events have more mass removed than more common events. The second heuristic used, called Jelinek-Mercer smoothing [26], differentiates higher order n-gram probability estimates with the same count by interpolating with lower order statistics. Interpolated modified Kneser-Ney is the current state of the art smoothing method [28]. This technique focuses on discounting for rare events (generally counts from zero to three) where interpolation has a larger impact on probability estimation. A Bayesian extension of this technique estimates different discounts for *all* seen n-grams instead of one discount for all n-grams with the same count in the training data [29].

Noted early on [28], the importance of smoothing lessens as the amount of available data increases. State of the art systems in machine translation or speech recognition for English use billions or trillions of words to estimate distributions using five or more word histories. In fact, smoothing has become a bottleneck for some large scale applications that a degenerate smoothing technique was preferred since it allowed for faster computational access to a very large language model [20] .

2.2.2 Efficiently Acquiring Labels

Simply budgeting for elicitation or transcription of in-domain data should be the first choice of any language modeling engineer. After all, there is no data like more data.However, this approach is often unfeasible. First, the cost of transcription and elicitation may greatly exceed available budgets. Domains such as rare languages, medical domains or sensitive topics may require expert transcription by a limited pool of transcribers. Also, data acquisition costs may outweigh the financial profits for deploying automatic systems to less common languages. Second, the significant time investment in transcription limits rapid deployment of systems to new domains. Historical estimates of transcription for speech are twenty hours of effort per hour of transcribed audio. And third, while transcription costs are linear with the amount of data transcribed, system performance is typically logarithmic. Due to the statistical nature of language models, an order of magnitude increase in training samples will only linearly improve system performance, diminishing the value of additional samples. Transcribing the large volumes of audio required for state of the art recognition is impractical for all but the most important domains. Efforts to lessen this burden focus on either reducing the cost of transcription or the need for labels in the first place.

Research into cost-effective data acquisition is called active learning. For the majority of machine learning scenarios, data samples are cheap while data labels are expensive. Active learning selectively annotates samples from a large pool according to criteria expected to increase model performance. The aim is to match or beat the performance that a statistical model would achieve if the entire pool were labeled. Active learning has benefited a wide variety of NLP tasks [30] and is an active area of research in theoretical machine learning [31].

Improving language models through active learning has been investigated in the automatic speech recognition community. In this setting, the word sequences modeled are actually the latent labels of speech audio. Applying active learning to call center data achieved the same performance as random sampling with 27% less labels [32]. An ASR system trained on a small amount of data automatically labeled a large pool of utterances from call center data. Utterances with the least estimated confidence were manually transcribed and added to the language model. Similar work on active learning of acoustic models for automatic speech recognition extended the selection criteria [33]. More efficient gains can be had by not selecting the least confident samples, but also those expected to be the most informative.

Instead of reducing the number of samples in need of labeling, the community has also considered methods to reduce the labor costs of annotation. Due to budget limitations, the 2000 hour English conversational telephone speech corpus Fisher necessitated a much cheaper transcription methodology than previously used. Transcribers were instructed to ignore capitalization, punctuation, or other careful annotation of mispronunciations and background noise. Although the transcription quality was much lower than previous efforts, systems trained on the data suffered little degradation [34, 35].

Further efforts have considered "crowdsourced" annotation of a wide variety of NLP tasks, ranging from parallel data [36] to many linguistic labels [37]. In this scenario, non-experts are given minimal training and instead rely on innate human knowledge to annotate data. Non-expert annotation has also been extended to non-English languages. Colloquial Arabic data was elicited and collected through crowdsourcing by human translation of Modern Standard Arabic newswire [38, 39]. The ultimate extension of this work is research into annotation games, which acquire labels as a side effect of a non-expert's enjoyment. Examples include "mind matching" games where two players attempt to match labels of images [40].

Research across various crowd sourcing platforms consistently demonstrates that reducing annotator skill requirements leads to significant cost savings with minimal impact on statistical systems. However, crowdsourcing is dependent on the data reaching a large audience, which may not be possible for sensitive data like medical, business or government data. Language engineers may not enjoy the benefits of scale for these limited domains as potential transcribers need to be vetted. Additionally, it can be difficult to train non-experts for detailed annotation tasks. This poses a problem for labeling tasks which are not intuitive to an average layperson – for example more complex linguistic annotations like syntax or named entities. These challenges may preclude the use of crowdsourcing platforms for some domains, but the lessons of this thesis still motivate the use of non-expert annotators.

2.2.3 Domain Adaptation

A small amount of in-domain data may benefit from a larger pool of related out of domain data. The corpus statistics are accurate for that domain, but differ from the in-domain corpus in terms of vocabulary and relative word frequency.

The first value of out of domain data is providing a larger vocabulary of likely words in the new domain. While this dissertation assumes the vocabulary is known, research has successfully progressed on learning new vocabulary words for a domain. Broadcast news recordings can benefit from newswire articles written the same day [41]. Audiovisual recordings have meta-data such as keywords, document summaries or archivist notes [42].

If additional data is not available or relevant, sub-word language models can provide likelihood estimates. Depending on the task, these may take the form of character, phoneme, or syllable models, or other sub-word fragments. Partitioning words into finer units reduces the vocabulary size and increases the number of training samples from the training data. These two qualities lead to better estimated densities. However, these models have less predictive power than full word models since less context is taken into account. The best sub-word models are hybrids with full-word estimates for known words [43]. A Bayesian approach to automatically learn both sub-word units and probability estimates from phonetic lattices reports significant progress on phoneme error rate [44]. Finally, simply mapping unknown words to a common token provides some probability estimate, allowing the detection, if not recognition of out of vocabulary terms. This coarse technique is effective as the majority of tasks have expansive vocabularies with very low out of vocabulary rates.

The second task of domain adaptation is to best combine probability estimates from out-of-domain data with small amounts of in-domain data. Typically, the out-of-domain distributions are well estimated, but diverge from in-domain distributions, resulting in poor system performance on the new domain. Simple count pooling, in which the in-domain data and automatic counts are treated as one corpus, is consistently out-performed by language model interpolation. A variety of language models trained from different domains are interpolated with learned weights that maximize likelihood on held out in-domain data [22]. Interpolation and count merging have both been generalized with MAP (maximum a posteriori probability) adaptation. Instead of finding the vector of interpolation weights which maximize the likelihood of the development data, a prior over the weights is also modeled (typically a Dirichlet prior) [45].

Recent extensions to interpolation used a hierarchical Bayesian framework to better model data variability across domains [46]. These Bayesian methods have worked very well, achieving significant performance gains with very small amounts of data. However, these models are sensitive to hyper-parameters, often cumbersome and computationally expensive contrasted with the efficiency of interpolating in- and out of domain language models.

The previous methods directly used probability estimates from models trained on out of domain data. An alternative is to extract higher level statistics which are robust across domains. Factoring these statistics into a language model then eases the estimation burden on the available in-domain data. For instance, class transition probabilities of a class language model may be estimated from out of domain data [47]. This leaves only the word to class memberships to be estimated from the limited amount of in-domain data.

Higher level statistics in the form of topic and speaker role in meeting data have also been used to adapt language models [48]. Adaptation used a hierarchical Dirichlet process (similar to hierarchical Pitman-Yor models) to estimate unsupervised topic models. While it combined the best of Bayesian modeling knowledge and use of higher-level statistics, it did not result in meaningful improvements in speech recognition performance.

One limitation of incorporating higher-level statistics is the loss in predictive power inflicted when moving away from a word based n-gram language model. The independence assumptions necessary to factor the probability prediction weaken the resulting language model. For these strategies to be worthwhile, the gain in estimating higher-order statistics must outweigh the loss in a weaker model. The answer to this question is so far empirical, with previous results reporting varying degrees of success.

Domain adaptation is viable when large amounts of related corpora are available. Empirical results are strongest for closely related out of domain corpora. For instance, voice search applications benefit from large text web searches. When there is text from a closely related domain, adaptation should be the obvious first attempt. Finding these close domains is unfortunately left to the ingenuity of the researcher. Rare languages which differ widely from existing corpora will suffer due to a lack of available domains. However, prior work on domain adaptation from heterogeneous corpora has not demonstrated large success. Different vocabularies and widely different word usage limit the usefulness of out of domain corpora.

2.2.4 Beyond n-gram Features

The frustrating simplicity of n-gram language models has long inspired research into more complex methods that better capture linguistic information. Features such as trigger pairs [49], syntax [50] and topic information [51] have all been proposed but see limited use in current systems. This is due to the complexity of integrating such knowledge into an efficient model, the limited gain in application performance and the limited availability of annotated training data. More fruitful methods have focused on reducing data sparsity by merging context histories. One approach projects word histories to a continuous space using artificial neural networks [52, 53]. These non-linear projections better capture long term histories and semantic concepts. However, these multilayer networks can be difficult to train and deploy in an end system such as automatic speech recognition. It is also unclear if they offer additional robustness to training from erroneous data. Class-based language models [54] reduce data sparseness by estimating broader equivalence classes through a variety of (usually) information-theoretic approaches. Instead of semantic or syntactic inspired classes, recent work merged histories to reduce training bias and demonstrated significant gain over the typical n-gram formulation [55].

Despite extensive research into more complex models, deployed systems prefer to use a smoothed non-parametric n-gram language model. The ease of implementation and strong baseline make the simplest model difficult to beat. The reasons are not well understood by the research community, but so far additional features have failed to capture the broad semantic knowledge engaged by humans.

2.3 Log-Linear Modeling

One can arrive at a log-linear model from two motivations. The first desires a model whose predictions match what is known and assume nothing else. This principle of *maximum entropy* states that a model should predict known quantities at the same rate as empirical observations, but be uninformative about everything else. The empirical data *constrains* the space of all possible distributions. Among the many distributions which satisfy those constraints, the one which is maximally uninformative is a log-linear model.

The second motivation seeks to directly use the log-linear function since it has nice modeling characteristics. (In other fields it is known as the Ising model, the soft-max output function of neural networks, it is used in Markov Random fields and many other models throughout the statistics literature.) This function defines a valid probability since it is always greater than zero and the probabilities sum to one, thanks to the denominator (known as the partition function Z). Smoothing is implicitly built in since unseen events have probability $\frac{1}{Z}$.

2.3.1 Formal Model

Let X be a discrete random variable over a vocabulary of symbols $\mathcal{X}$. Then the probability that X takes on a value x conditioned on the learned parameters Θ is given as

$$P(X = x) = \frac{\exp \sum_{i=1}^{K} \theta_k \cdot f_k(x)}{\sum_{x' \in \mathcal{X}} \sum_{i=1}^{K} \theta_k \cdot f_k(x')} \tag{2.7}$$

where K *feature functions* $f_k : \mathcal{X} \to \mathcal{R}$ map an observation to a real value and $\theta_k \in \mathcal{R}$. It is the task of the domain expert to define these feature functions. Typically, feature functions are derived from seen observations from some set of data. “what is known” is defined by the expected frequency of these feature functions and a model is desired that will predict these features at the same rate. The task is to then estimate the parameters θ_k. First, the space of events $\mathcal{X}$ for language modeling is defined and K feature functions f_k, such that in expectation under the model P,

$$\mathbb{E}_P[f_k(X)] = c_k,\ k = 1, 2, \ldots, K \tag{2.8}$$

where feature f_k is observed a fraction $c_k \in [0, 1]$ of the time. If these constraints are consistent (which is unfortunately untrue for several real world applications), then there exists a unique solution which can be iteratively found [54].

It is at this point that this dissertation switches to the log-linear motivation as it offers a more principled and interpretable framework. By dropping the maximum entropy criterion and simply assuming log-linearity,the goal is no longer to match constraints, but instead have a corpus $D = \{x_1, x_2, \ldots\}$ and a model $P(X)$ parameterized by Θ and set of K feature functions $F(X)$. Under the principle of Bayesian estimation,both D and Θ are treated as random variables. The probability of some future data X is estimated by integrating over the uncertainty of Θ and computing $P(X|D) = \int_\Theta P(X|\Theta, D)P(\Theta|D)$. Unfortunately this integral is not always practically computable and in such cases the maximum likelihood point-estimate of Θ is used, which is given by

$$
\begin{aligned}
\arg\max_{\Theta}\ P(D|\Theta) = \max_{\Theta}\ \log P(D|\Theta) = \max_{\Theta}\prod_{i=1}^{|D|} P(x_i|\Theta) && (2.9)\\
= \max_{\Theta}\sum_{i=1}^{|D|}\log P(x_i|\Theta) && (2.10)\\
= \max_{\Theta}\sum_{i=1}^{|D|}\log\frac{\exp(\Theta\cdot F(x_i))}{\sum_{x'\in\mathcal{X}}\exp(\Theta\cdot F(x'))} && (2.11)\\
= \sum_{i=1}^{|D|}\sum_{k=1}^{K}\theta_k\cdot f_k(x_i) - \sum_{i=1}^{|D|}\log\sum_{x'\in\mathcal{X}}\exp(\sum_{k}\theta_k f_k(x')) && (2.12)
\end{aligned}
$$

resulting in a convex function w.r.t Θ, which can be optimize using iterative first-order Newtonian methods [56]. Each θ_i is updated round-robin and this process repeated until convergence. The parameter updates have an elegant solution given by

$$
\begin{aligned}
\frac{\partial\log P(D|\Theta)}{\partial\theta_j} &= \sum_{i=1}^{|D|}\sum_{k=1}^{K}\theta_k\cdot f_k(x_i) - \sum_{i=1}^{|D|}\log\sum_{x'\in\mathcal{X}}\exp(\sum_{k}\theta_k f_k(x')) && (2.13)\\
&= \sum_{i=1}^{|D|}\left(f_j(x_i) - \frac{\sum_{x'\in\mathcal{X}}\exp\sum_k\theta_k f_k(x')f_j(x')}{\sum_{x'\in\mathcal{X}}\exp\sum_k\theta_k f_k(x')}\right) && (2.14)\\
&= \sum_{i=1}^{|D|}\left(f_j(x_i) - \sum_{x'\in\mathcal{X}}P_\Theta(x')f_j(x')\right) && (2.15)\\
&= \frac{1}{|D|}\sum_{i=1}^{|D|}f_j(x_i) - \sum_{i=1}^{|D|}\sum_{x'\in\mathcal{X}}P_\Theta(x')f_j(x') && (2.16)\\
&= \mathbb{E}_{\tilde{P}}[f_j] - \mathbb{E}_{P_\Theta}[f_j] && (2.17)
\end{aligned}
$$

which is the difference between the empirical count $\mathbb{E}_{\tilde{P}}[f_j]$ and the expected count of f_j under the model $\mathbb{E}_{P_\Theta}[f_j]$ and once it becomes zero for all θ_i, training will converge. Iterative estimation will converge to the maximum likelihood estimates, thus the constraints must either be smoothed or else a stopping criterion must be used to prevent over-fitting to the training data.

2.3.2 Gaussian Prior as Smoothing

Another method of preventing over-fitting is to instead find the *maximum a posteriori probability* or MAP estimate by adding a prior over Θ. Gaussian priors (and any from the exponential family) are amenable to efficient estimation, empirically out-perform other priors and are well-matched to empirical behavior [57]. They have a direct relation with an L_2 prior over the parameter space and penalize parameters for deviating from some nominal value, typically zero. Still solvable with Newton's methods, parameter estimation stops once

$$\mathbb{E}_{P_\Theta}[f_j] = \mathbb{E}_{\tilde{P}}[f_j] - \frac{\theta_j}{\sigma_j^2} \tag{2.18}$$

where σ_j^2 is now a specified *variance* of the prior over θ_j. This is analogous to absolute discounting (Equation (4.6)), where the empirical counts of a feature $\mathbb{E}_{\tilde{P}}[f_j]$ are discounted by a quantity $\frac{\theta_j}{\sigma_j^2}$. Due to the log-linearity of $P_\Theta(X)$, as it grows exponentially, θ_j grows linearly. Thus the Gaussian prior is a logarithmic discount of the empirical frequencies.

MAP estimation requires specification of a per-feature variance σ_j^2. Most models tie all sigmas together due to the difficulty of interpretation and estimation of these variances. The variance encodes the domain expert's confidence in the training data D. If one truly believes that the empirical frequencies estimated from D reflect the underlying distribution, then the variance is set to infinity, negating any discount. However, all real world situations will require a finite variance. A very small variance enforces a high penalty for large parameter values and thus pushes the estimated distribution closer to the uniform distribution (the maximum entropy solution). As the variance increases, parameter estimates are free to wander away from uniform distribution and converge to the maximum likelihood estimate. This interpretation is represented graphically in Figure 2.3. The uniform distribution maximizes the entropy of the estimated distribution. As the variance increases, the MAP estimate moves towards the maximum likelihood estimate.

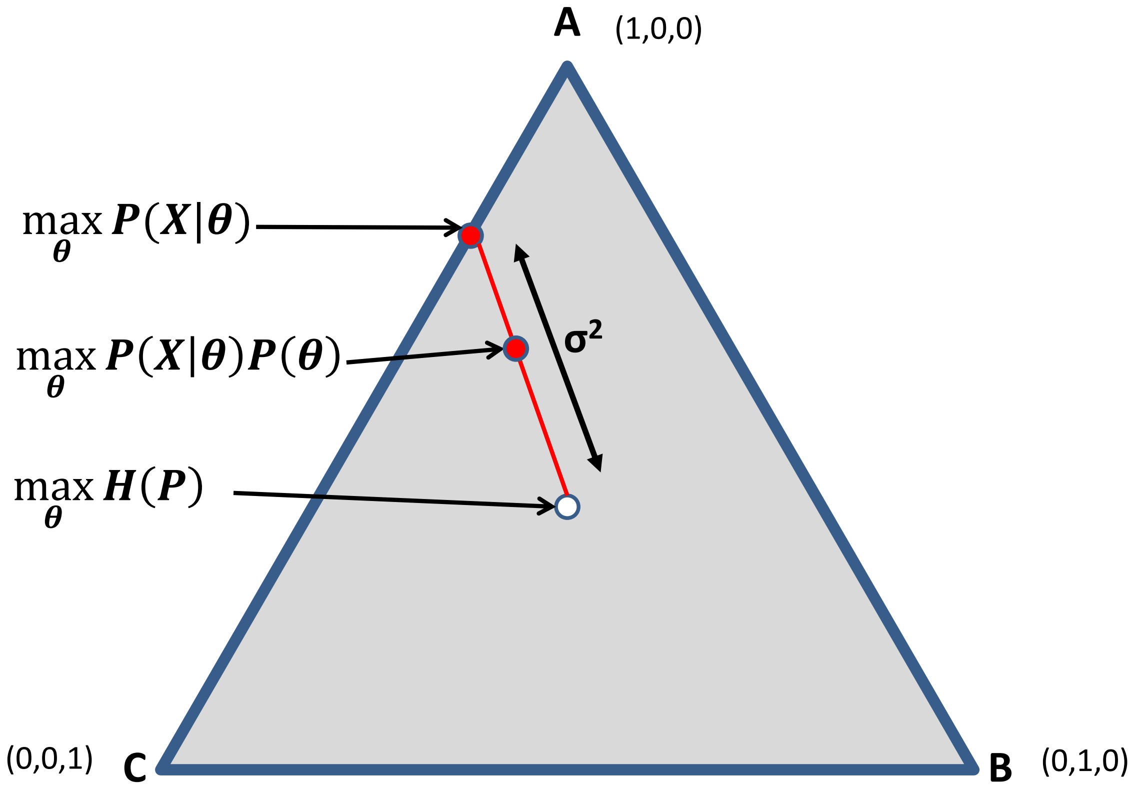
A (1,0,0)
$\max_{\theta} P(X|\theta)$
$\max_{\theta} P(X|\theta)P(\theta)$
$\max_{\theta} H(P)$
σ^2
(0,0,1) C
B (0,1,0)

Figure 2.3: *Probability Simplex* - The triangle represents the space of all probability distributions over the three word vocabulary (A,B,C). At the center lies the white dot representing the uniform distribution which maximizes entropy. The red dot represents the empirical frequencies from training data: (A=7,B=0,C=3) and is the maximum likelihood estimate. The MAP estimate is a compromise between these two extremes, controlled by the variance.

Under this new view, one can consider many other possible priors as starting points for MAP estimation. The prior is no longer a parameter regularizer, but an expression of some prior belief over the model parameters. Therefore, this use of MAP estimation is commonly referred to as MAP *adaptation* since the updated constraint

$$\mathbb{E}_{P_\Theta}[f_j] = \mathbb{E}_{\tilde{P}}[f_j] - \frac{\theta_j - \theta_i^0}{\sigma_j^2} \tag{2.19}$$

introduces θ_i^0, which was present, but equal to zero, in Equation (2.18). This is the prior over θ_i and need not be zero and need not be tied across parameters. For instance, Θ^0 may be the parameters of log-linear model estimated on out of domain data or noisy samples. Section 4.7.1 will explore this in detail.

2.4 Semi-Supervised Learning

Machine learning fall into three regimes. The classical formulation is supervised learning: each training sample has an associated label and the goal is to learn a mapping between observed data and (latent) labels. Supervised tasks can either learn the joint density of the data and labels (*generative* modeling) or else focus on the conditional density of the label given data (*discriminative* modeling). At the other extreme, unsupervised machine learning estimates interesting structure from observed data, typically a density which generated the data. Language modeling fits this description, where the interesting "structure" is a distribution over all possible word sequences estimated from finite data. Other tasks infer latent structure such as unsupervised topic modeling or clustering.

Between these two scenarios lies semi-supervised learning. In this regime, a subset of samples have labels while the (typically larger) remaining set of samples does not. The machine learning community has partitioned semi-supervised techniques into to two subsets [58]. First is transductive learning, where a labeled training set is used to classify an unlabeled test set. For example, k-NN classifiers use a small subset of labeled samples to propagate labels to the unlabeled data. The other task is inductive learning, where the desired outcome is a prediction function over all possible domain samples. Instead of only classifying the observed unlabeled data, the algorithm cares about performance on future unseen data. A classic example is the expectation maximization algorithm [59], which learns latent structure by maximizing the likelihood of the observed data. It is employed most commonly in HMM parameter estimation [13] and Gaussian mixture estimation [60].

Learning Regime	**Labels**	**Example**
Supervised	Yes	Maximum Likelihood, standard classifiers
Semi-Supervised	Partial	Expectation Maximization, k-NN classifier
Unsupervised	No	Clustering, Topic modeling

Table 2.2: *Machine Learning Regimes* - Traditional machine learning considered only **supervised** learning where each sample has a label. **Unsupervised** machine learning attempts to learn structure of the observed data. Lying between the two is **semi-supervised learning** which exploits structure of the data with the goal of improving labeling accuracy.

Semi-supervised learning is particularly appealing to the NLP community because data is practically free while labels are expensive. However, semi-supervised learning has not provided the magic cure. While theoretical studies have confirmed that additional samples should help [61], reality is mixed, with empirical studies reporting gains and losses across a variety of models and domains. One theory is that unlabeled data is very sensitive to modeling assumptions since many possible models could generate the unlabeled data, but fewer could explain unlabeled *and* labeled data. This is an ongoing area of interest for the machine learning community and there are multiple literature reviews available for further reading [58].

This dissertation falls under inductive learning. The task is to estimate parameters of a model for use on some future, unseen task. While there may be a large quantity of audio, the goal is not necessarily to label the corpus, but instead to infer transcripts for use in estimating the statistical parameters of a language model.

2.4.1 Self-Training

One of the earliest methods of semi-supervised learning, self-training (or self-learning) is a general technique where a model provides automatic training data for itself [62–64]. More of a technique than algorithm, its general idea is to first train a model on the available supervised training data. Labels are then inferred over unlabeled samples and used to re-train the model. Interesting algorithmic choices lie in selecting a subset of the data to use in model retraining. Similar to active learning, estimates of model confidence can be used to select training data from the larger pool of samples. However, since the initial supervised model is not accurate, the labels have some probability of error. Theoretical analysis has not managed to explain the empirical successes of self-training. Some analytical work has equated it to a version of the EM algorithm without entropy constraints [65].

Sometimes mistakenly called “unsupervised training” in the automatic speech recognition community, self-training of acoustic models has yielded great success. Initial experiments assumed that high-quality automatic transcripts (with error rates below 20%) were required to improve performance [66]. This high threshold for selecting the unlabeled data, combined with small amounts of unlabeled audio available for selection resulted in only forty-five minutes of audio out of 25 hours to improve a three hour baseline model. Later experiments conducted acoustic model self-training with very small amounts of labeled data and showed that a low error rate was unnecessary and all the audio should be used. Ten minutes of transcribed broadcast audio were sufficient to achieve a 33% relative reduction in WER when using 135 hours of unlabeled audio [67]. More recent work [1] with conversational English used one hour of labeled audio and 2000 hours of unlabeled audio to achieve 80% of the possible reduction compared to manual transcription. Further work explored better confidence estimation methods and data selection [68–71]. Instead of estimating confidence of entire speech utterances or words, confidences at the acoustic frame level were used to weight sample updates [72]. The success of self-training did not depend on these fine-grained confidence estimates as coarser confidences provided strong results as well. One limitation of semi-supervised acoustic modeling is the failure to benefit from discriminative training of acoustic models [73]. The machine learning community in general has not been able to estimate discriminative models using unsupervised data, i.e. via self-training [74].

These previous efforts relied on a strong in-domain language model to improve the acoustic model. While a reasonable assumption for many scenarios, no work has analyzed the importance of this additional side information to guide recognition. The language model remains fixed and no previous work has considered training both at the same time. In more recent work, semi-supervised language modeling used self-training to modestly improve performance [45, 73, 75, 76]. Since a speech recognizer uses both an acoustic and language model, acoustic model self-training is closely related to a similar task, co-training.

2.4.2 Co-Training

Co-training relies on data samples having multiple “views” or independent subsets of the features which are then used to train independent classifiers. For example, a web site has features based on the text on the page as well as the hyper-links referring to and referred from the web page [77]. In this way, two or more classifiers can be trained that have different decision boundaries. They are then used to classify the topic of a large pool of unlabeled data and similar to self-training, are then re-trained.

Instead of simple majority voting or model combination, a more interesting selection strategy is to train classifiers on different subsets of the data. One classifier is trained on samples that it has low confidence but which the other model has high confidence and vice-a-versa.

This technique has successfully been applied to noun phrase identification [78], named entity recognition [79] and part of speech tagging [80]. Another intuitive example is sentence segmentation of audio utterances [81]. Single channel audio recordings of meeting data need to be segmented into separate speaker segments. Prosodic information, such as pitch, and lexical information provide independent views of the data and out-perform a joint classifier. Other examples include co-training statistical machine translation models [82] and multilayer perceptron and HMM-GMM acoustic models [83].

Co-training is sensitive to having orthogonal subsets of the data that on their own are sufficient to train models with reasonable performance. Arbitrary partitions of the feature space are not guaranteed to lead to good models [84] and the suitability of this method is dependent upon natural data partitions. It does, however, suggest a method for bootstrapping an automatic speech recognition system with minimal transcribed data. Both the acoustic and language models provide confidence estimates (in the form of posterior probabilities) at the utterance and word level. It may be possible to co-train both models in a semi-supervised setting [85].

2.5 Metrics

This is an empirical dissertation with emphasis on system performance on real speech corpora. Differences in training data, language modeling features and more result in widely different distributions, but not all meaningfully impact some downstream task. Since formal evaluations using humans require large investments in time, the following metrics, measured on held-out data, are inexpensive to compute and correlate well with real gains.

2.5.1 Language Model Evaluation

Computed on either the training or held-out data, *perplexity* [86] quantifies a language model's ability to reduce uncertainty. Perplexity is derived from a more common measurement of information theory: cross entropy.

A language model provides a conditional probability distribution over words w given histories h: $P(w|h)$. The language model P is then evaluated by how well it matches an empirical distribution $\tilde{P}$ estimated on some held-out data by computing

$$H(\tilde{P}, P) = -\sum_{i}^{N} \tilde{P}(h_i, w_i) \log_2 P(w_i|h_i) \tag{2.20}$$

$$= -\sum_{h,w} \frac{c(h,w)}{N} \log_2 P(w|h) \tag{2.21}$$

where $c(h, w)$ is the count of the history h followed by word w in a corpus of N tokens. This metric corresponds to the number of bits required to encode the word w_i following the history h_i. Smaller number of bits means the language model more closely captures a held-out distribution. Perplexity is then defined as

$$\mathrm{PPL} = 2^{H(\tilde{P},P)}\,. \tag{2.22}$$

Both metrics have been historically reported in the research, but perplexity dominates because the effective branching factor of the model is more interpretable. For work whose end task is language modeling (such as compression), then perplexity or cross entropy is sufficient. However, for other end tasks such as speech recognition, the language model is but one sub-model of the entire recognition pipeline. Improvements in language modeling (measured by perplexity) may not carry over to other tasks.

An illustrative example is better modeling of proper names: they are infrequent token by token, but as a class occur often enough that class-based language models can often see great reductions in perplexity. However, these proper names tend to be *phonetically long* and thus easier for a speech recognizer to correctly output provided they have accurate pronunciation entries in the lexicon. This often means that despite very low language model scores, content-rich words like proper names are often produced correctly. Additionally, language models cannot be compared if their vocabularies differ, which is a frequently changed condition in speech recognition research. Thus a metric more suitable for the target task of this dissertation, speech recognition, is required.

2.5.2 Large Vocabulary Continuous Speech Recognition Evaluation

LVCSR is a sequence prediction task over an unknown vocabulary. Evaluating system performance requires more than calculating accuracy on a word by word basis. The standard metric in the community is *Word Error Rate* (WER). It is a function of the number of insertions, deletions and substitutions of the words in the reference. To count the instances of these three categories, an alignment between the hypothesis and reference text is made by minimizing Levenshtein distance. Then each word in the *hypothesis* is counted as correct ($\#C$), substitution ($\#S$), or insertion ($\#I$) if it doesn't align with any reference word. Additionally, any words in the reference that did not align are counted as deletions ($\#D$) so that

$$\text{WER} = \frac{\#I + \#D + \#S}{\#\text{ ref. word tokens}}\,. \tag{2.23}$$

Lower WER is better, with state of the art systems achieving less than 10% on domains such English voice search [87] or Broadcast News closed captioning [88]. Note that out of vocabulary terms are automatically incorrect, as they cannot appear in the recognition hypothesis.

Debates about the appropriateness of WER continue in the community, but it has become the dominant metric as it is task agnostic, easily computable and well correlated with other task dependent metrics. A frequent criticism is that content-less words such as function words and hesitations are weighted the same as content words. For the purposes of this dissertation, WER will be the figure of merit except in Chapter 5, which will discuss other downstream tasks.

2.5.3 Semi-Supervised Evaluation

For all the experimental corpora in this dissertation, manual transcripts are available. This dissertation can report the effectiveness of semi-supervised learning by computing the upper-bound performance for a given corpus. The semi-supervised methods (which require no additional labeling) are contrasted with that of manually labeling the entire corpus. With this upper bound, **Recovery** can then be computed. A semi-supervised experiment has three performance measures:

- WER_I - The WER of the *initial* models trained before semi-supervised training.
- WER_S - The WER of the *semi-supervised* models after semi-supervised training.
- WER_T - The WER of the *supervised* models trained with full supervision.

$$\text{WER Recovery} = \frac{\text{WER}_I - \text{WER}_S}{\text{WER}_I - \text{WER}_T} \tag{2.24}$$

A WER Recovery of 100% states that semi-supervised training is as effective as supervised training. This dissertation reports Recovery in addition to absolute gains since it is a valuable indicator for the usefulness of the semi-supervised methods on future domains.

If the upper bound for a semi-supervised result is lo If a semi-supervised method barely outperforms the baseline, it may be that a model trained on manually labeled data may be modestly better as well. Thus the lackluster results are not due to the semi-supervised method, but the usefulness of the underlying data. A high recovery indicates that the semi-supervised method matches supervised performance, but there was little gain to be had or that the model is unable to exploit the additional data.

Chapter 3

Utilizing Non-Expert Transcription

Successful speech recognition depends on substantial investments in data collection. Even after training on 2000+ hours of transcribed conversational speech, over a billion words of language modeling text, and 100,000 word hand-crafted pronunciation dictionaries, state of the art systems still have an error rate of around 15% for conversational English [19]. This error rate is high enough that one out of ten words is wrong, noticeably impacting user perception of the system quality. Transcribing the large volumes of data required for Large Vocabulary Continuous Speech Recognition (LVCSR) of new languages to useful levels of accuracy appears prohibitively expensive. Recent work has shown that crowdsourcing platforms such as Amazon's Mechanical Turk[1] can be used to cheaply annotate data for natural language processing applications [37, 89, 90]. This chapter focuses on reducing the cost of transcribing conversational telephone speech (CTS) data. Other approaches may reduce the *amount* of transcription required, but if the transcription cost is lowered with minimal degradation in transcription quality, it would be possible to deploy LVCSR models

[1] http://www.mturk.com

trained on very large corpora. Previous measurements of crowd-sourced annotation stopped at agreement/disagreement with professional annotation. This dissertation takes the next logical step and measure the utility of end-systems trained with non-professional transcription. This comparison focuses on the end task of these transcripts - deploying an LVCSR system.

Mechanical Turk is an online labor market where workers (or Turkers) perform simple human intelligence tasks (HITs) for small amounts of money (or micro-payments) – frequently as little as $0.01 per HIT. Since HITs are well suited for tasks that are difficult for computers, but easy for humans, this is a natural fit for annotation required for language processing tasks [37]. Mechanical Turk has even spawned a business that specializes in manual speech transcription of podcasts, voicemails and dictation.[2]

Automatic speech recognition (ASR), particularly for conversational speech, is a difficult problem. Characteristics like rapid speech, phonetic reductions and speaking style limit the value of non-CTS data, necessitating in-domain transcription. Luckily, strong methods exist to bootstrap the acoustic model from small amounts of transcription. Even a few hours of transcription are sufficient to bootstrap with semi-supervised methods like self-training [67], which will be explored in Chapter 4.

The speech community has built effective downstream solutions for the past twenty years despite imperfect recognition. In topic classification, 90% accuracy is possible on conversational data even with ca. 80% word error rate (WER) [91]. Other successful tasks include keyword search from speech [92] and spoken dialog processing [93]. Inexpensive transcription could quickly open new languages or domains (like meeting or lecture data)

[2] http://castingwords.com/

for automatic speech recognition.

This chapter makes the following points:

- Quality control isn't as crucial for crowd-sourced transcription as previously thought because a system built with non-professional transcription is only 6% worse in WER for $\frac{1}{30}$ the cost of professional transcription.
- Resources are better spent collecting more data than improving data quality.
- Transcriber skill can be accurately estimated without gold standard data.

3.1 Previous Work

Research into Mechanical Turk by the NLP community has largely focused on comparing the quality of annotations produced by non-expert Turkers against annotations created by experts. The first application of Mechanical Turk to NLP conducted a comprehensive study across a variety of NLP tasks [37]. They showed that high agreement could be reached with gold-standard expert annotation for these tasks through a weighted combination of ten redundant annotations produced by Turkers. Similar trends were also made for machine translation evaluation [36] , and furthermore, Turkers could accomplish complex tasks like creating reading comprehension tests. Mechanical Turk could also be used to improve an English isolated word speech recognizer by having Turkers listen to a word and select from a list of probable words at a cost of $20 per hour of transcription [90]. Turkers provided transcripts of verbal instructions to robots with clean speech. By using five redundant transcriptions, the average transcription disagreement with experts was reduced from 4% to 2% [94] .

Few studies have attempted to transcribe speech data in non-English languages. Although studies in other disciplines such as machine translation report great success [38] [95], speech transcription via crowd-sourcing has suffered from a dearth of foreign-language speakers. One study attempted to collect Amharic and Swahili, both under-resourced African languages with limited success [96]. After rejecting 90% of the submitted jobs and waiting three months, the authors were left with 1.5 and 0.75 hours of transcription, respectively. The transcription quality showed little degradation compared to professional transcription, but the lack of fluent speakers led the team to work with a Kenyan university to find sufficient workers.

Previous efforts at reducing the cost of transcription include the EARS Fisher project [34], which collected 2000+ hours of English CTS data – an order of magnitude more than had previously been transcribed. To speed transcription and lower costs, [3] created new transcription guidelines and used automatic segmentation. These improved the speed of transcription from fifty times real time to six times real time, and made it cost effective to transcribe 2000 hours at an average of $150 per hour. Acoustic models trained on the "quick" transcripts exhibited almost no degradation in performance, although discriminative training of the model was sensitive to transcription errors.

Other work has considered how to factor speech transcription into smaller subtasks with smaller cognitive load. Instead of collecting many independent transcripts of the same utterance, recent work used human labor to correct likely errors by humans [97]. Conducted on English meeting data, two independent transcripts of an utterance were merged and disagreements blanked out. A second pass then corrected the disagreement region. This reduced expert disagreement from 23.1% to 17.5%. When widened to a pool of five workers, the labeling error was further reduced to 15.1%. However, collecting five independent transcripts and then combining them also reduced error rate to 15.2% - statistically identical to the two pass approach. Moreover, the authors reported that Turkers were *twice as likely* to report the error correction task as difficult versus straightforward transcription.

One approach used human judgment to rate utterance confidence (as opposed to *estimated* confidence in Section 4.4.2) [98]. An LVCSR system produced automatic transcripts of English spoken dialog queries. These transcripts, along with the matching audio snippet, were presented to a human for classifying as either correct, incorrect, or unintelligible. This removed 17% of the untranscribeable data and verified that 54% of the training data was correctly recognized by the ASR system. The remaining incorrect utterances were then passed on for human transcription.

Although this two-pass strategy reduced average Turker transcription error rate from 13.7% to 8.1% and improved LVCSR performance when trained on these utterances from 64.6% to 62.3%, the two pass approach was less cost-effective than simply transcribing more data. The study confirmed the conclusions of this chapter: intelligent data collection is beaten out by simple transcription. Verification vs. transcription is an appealing use of human labor, but also susceptible to human bias. One study reported that human transcribers - employed directly by the author - incorrectly approved automatic transcripts as correct since the first pass was acceptably close to what the transcriber heard [99].

Other work extended to a data-driven number of passes [100] by requesting redundant transcription for utterances with high disagreement. The minimum number of transcribers is then two, but over half of the utterances of spoken business names were deemed reliable after two passes. The authors also utilized automatic transcription for a modest improvement- but intentionally did not show it to the human transcribers for fear of cheating (another clue to confirmation bias). Similar to other reports, [101] Mechanical Turk workers show a bias towards minimizing the number of edits when shown automatic transcript.

3.2 Experiment Description

3.2.1 Corpora

Most experiments were conducted on a twenty hour subset of the English Switchboard corpus [102] where two strangers converse about an assigned topic. Two sets of transcription were used as a gold standard: high quality transcription from the LDC and those following the Fisher quick transcription guidelines [3] provided by a professional transcription company. All English ASR models were tested on the carefully transcribed three hour Dev04 test set from the NIST HUB5 evaluation.[3] A 75k word lexicon taken from the EARS Fisher training corpus covers the LDC training data and has an out-of-vocabulary (OOV) rate of 0.18% on the Dev04 transcripts.

Experiments were also conducted in Korean and collected Hindi and Tamil data from the Callfriend corpora[4]. Participants were given a free long distance phone call to

[3] http://www.itl.nist.gov/iad/mig/tests/ctr/1998/current-plan.html

[4] http://www.ldc.upenn.edu/CallFriend2

talk with friends or family in their native language, although English frequently appears. Since Callfriend was originally intended for language identification, only the 27 hour Korean portion has been transcribed by the LDC.

3.2.2 Transcription Task

Using language-independent speech activity detection models, the audio was segmented each ten minute conversation into five second utterances, greatly simplifying the transcription task [103]. Utterances were assigned in batches of ten per HIT and played with a simple flash player with a text box for text entry. All non-empty HITs were approved and no bonuses were awarded except as described in Section 3.4.1.

3.2.3 Measuring Annotation Quality

The usefulness of the transcribed data is ultimately measured by how much it benefits a speech recognition system. Factors that cause disagreement (word error rate) to increase between Turkers and the gold standard do not necessarily impact system performance. These include typographical mistakes, transcription inconsistencies (like improperly marking hesitations or the many variations of `um`) and spelling variations (`geez` or `jeez` are both valid spellings). Additionally, the gold standard is itself imperfect, with typical estimates of inter-labeler disagreement around five percent. Therefore, this dissertation judges the quality of Mechanical Turk data by comparing the performance of one LVCSR system trained on Turker annotation and another trained on professional transcriptions of the same dataset.

3.3 Establishing Best Practices

As an initial test to see how cheaply conversational data could be transcribed, one hour of test data from Hub5 Dev04 was uploaded to Mechanical Turk. The cost was first $0.20 per HIT ($0.02 per utterance). This test finished quickly, with an average disagreement with professionals to be 17%. Next, despite a lower payment of $0.10 per HIT, disagreement was again 17%. Finally, the price dropped to $0.05 per HIT or $5 per hour of transcription and again disagreement was nearly identical at 18%, although a few Turkers complained about the low pay.

Using this price, the full twenty hours was redundantly transcribed three times. 1089 Turkers participated in the task, yielding transcription at the rate of 10 hours of transcription per day. On average, each Turker transcribed 30 utterance (earning 15 cents) at an average disagreement of 23%. Transcribing one minute of audio required an average eleven minutes of effort (denoted 11xRT). 63 workers transcribed more than one hundred utterances and one prolific worker transcribed 1223 utterances.

3.3.1 Comparing Non-Professional to Professional Transcription

Table 3.1 details the results of different selection methods for redundant transcription. For each method of selection, an acoustic and language model was built and report WER on the held-out test set (transcribed at very high accuracy).

First, one of the three transcriptions per utterance were selected at random (as if the data were only transcribed once) and repeated this three times with little variance. Selecting utterances randomly by *Turker* performed similarly. Performance of an LVCSR

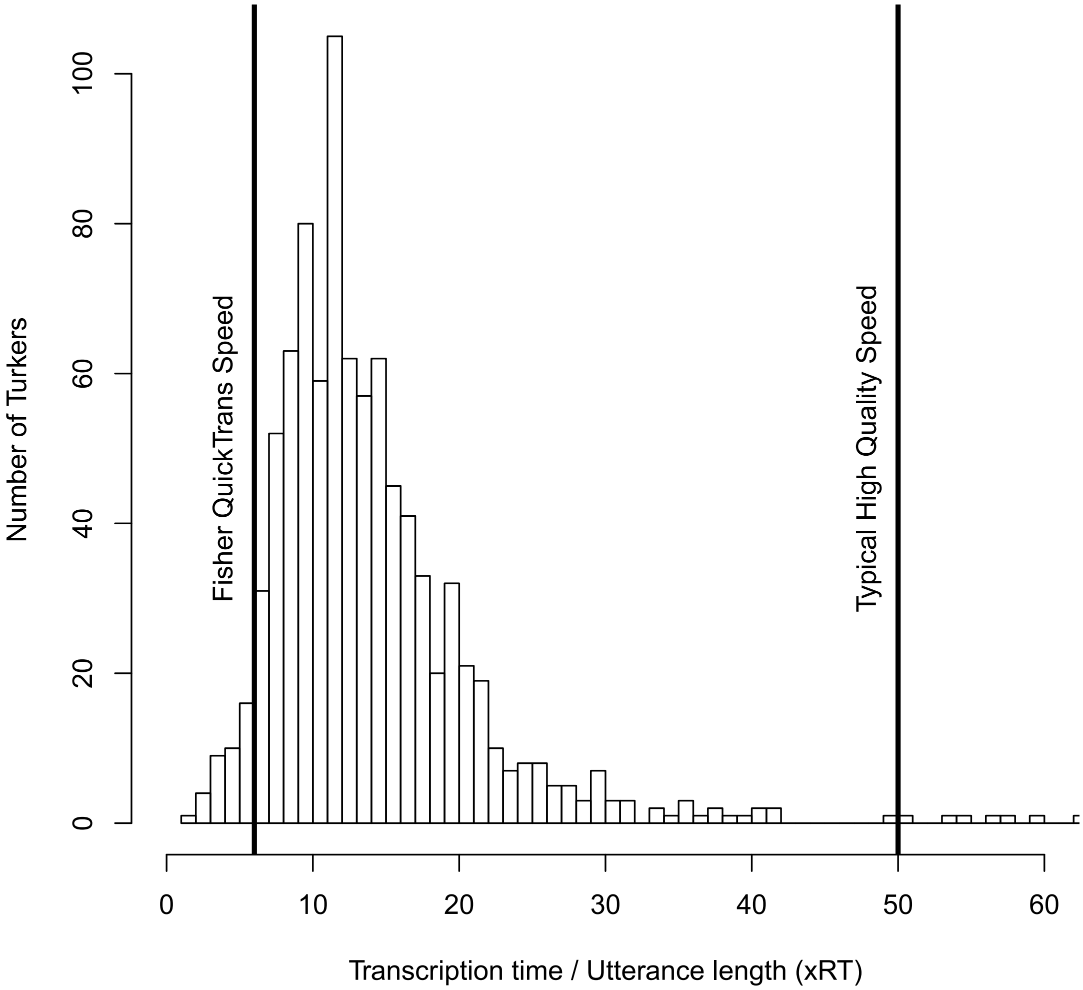

Figure 3.1: *Turker Transcription Rate on the English Switchboard Corpus* - 63 Turkers transcribed 1223 utterances (avg of 19 utterances per Turker). The time to completion per utterance was recorded and when divided by the length of the utterance gives the average transcription speed per Turker on the x axis. A histogram shows that on average, Turkers transcribed almost as fast as the historically fastest 'QuickTrans'. The average is 11xRT compared to 6xRT for the 2004 QuickTrans effort.

system trained on the non-professional transcription degrades by only 2.5% absolute (6% relative) despite a disagreement of 23%. This is without any quality control besides throwing out empty utterances. The degradation held constant as the amount training data was swept from one to twenty hours. Both the acoustic and language models exhibited the logarithmic reduction in WER with the amount of training data. Independent of the amount of training data, the acoustic model degraded by a nearly constant 1.7% and the language model by 0.8%, relative to careful (professional) transcription.

To evaluate the benefit of multiple transcriptions, two oracle systems were built. The *Turker oracle* ranks Turkers by the average error rate of their transcribed utterances against the professionals and selects utterances by Turker until the twenty hours is covered (Section 3.3.3 discusses a fair way to rank Turkers). The *utterance oracle* selects the best of the three different transcriptions per utterance. The best of the three Turkers per utterance wrote the best transcription two thirds of the time.

The utterance oracle only recovered half of the degradation for using non professional transcription. Cutting the disagreement in half (from 23% to 13%) reduced the WER gap by about half (from 2.5% to 1%). Using the standard system combination algorithm ROVER [104] to combine the three transcriptions per utterance only reduced disagreement from 23% to 21%. While previous work benefited from combining multiple annotations, this task shows little benefit.

The LVCSR system *does* show more sensitivity to transcription quality when the acoustic model is discriminatively trained [14]. After maximum likelihood estimation with the Baum-Welch algorithm, the acoustic models are adjusted to directly minimize prediction

Transcription	**Disagreement with LDC**	**ASR WER**
Random Utterance	23%	42.0
Oracle Turker	18%	41.1
Oracle Utterance	13%	40.9
Contractor	$< 5\%$	39.6
LDC	-	39.5

Table 3.1: *Quality of Non-Professional Transcription* - Even though disagreement for random selection without quality control has 23% disagreement with professional transcription, an ASR system trained on the data is only 2.5% worse than using LDC transcriptions. Optimal quality control could reduce disagreement significantly. Either ranking the best Turkers (row 2) or selecting the best transcript on a per-utterance basis (row 3) would reduce disagreement, but cut WER by only 1%. Regardless of the method, the upper bound for quality control recovers only 50% of the total loss. These are still significantly worse than hiring a contractor directly (but at much lower cost).

errors. Table 3.2 details the sensitivity of discriminative training depending on the quality of the transcripts. The first row selected a random utterance with an average disagreement to the LDC transcripts of 23%. The upper bound on quality control has a disagreement of 13% compared against professional transcription.

All three transcripts are used to train maximum likelihood models, which are then discriminatively trained using the minimum phone error criterion. While all three systems benefit from discriminative training, it is more effective using the high quality LDC transcripts. These results do not invalidate non-expert transcription. The cost savings are still dramatic for maximum likelihood models. It does, however, indicate that after a certain point, labor should be spent on improving the transcripts one has versus collecting new data.

		ASR WER		
Transcription	**Disagreement**	**ML**	**MPE**	**Gain**
Random Utterance	23%	42.0	41.4	0.6
Oracle Quality Control	13%	40.9	40.1	0.8
LDC	-	39.5	38.2	1.3

Table 3.2: *Discriminative Training is Sensitive to Transcription Quality* - Discriminatively trained acoustic models are standard in state of the art system. Since the method focuses on errors versus maximum likelihood, it is more sensitive to the quality of transcription. Maximum likelihood models trained using random Turkers are only 6% worse despite 23% disagreement with the LDC. However, discriminative training is only able to improve error rate by half as much (0.6 vs. 1.3) compared to LDC transcripts.

3.3.2 Combining with External Sources

While in-domain speech transcription is typically the only effective way to improve the acoustic model, out-of-domain transcripts tend to be useful for language models of conversational speech. As mentioned in Chapter 1, many low-resource languages do not enjoy well matched out of domain corpora. Typically, the only electronically available corpora will be newspapers or transcripts of television and radio news broadcasts. Broadcast News (BN) transcription is particularly well suited for English Switchboard data as the topics tend to cover news items like terrorism or politics. A small one million word language model was used as a proxy to simulate a resource-poor language and interpolated it with varying amounts of LDC or Mechanical Turk transcriptions. Figure 3.2 details the results. When no outside data is available, the language model immediately benefits from more Mechanical Turk transcription, with a constant degradation of 0.8% WER compared to LDC data. However, four hours were required to improve over the 1M words of BN transcription - compared to an immediate gain for one hour of LDC transcripts - although the gap to LDC data decreased to 0.6% WER.

3.3.3 The Value of Quality Control

With a fixed transcription budget, should one even bother with redundant transcription to improve an ASR system? To find out, an additional 40 hours of Switchboard was transcribed using Mechanical Turk. Disagreement to the LDC transcriptions was 24%, similar to the initial 20 hours. The two percent degradation of test WER when using Mechanical Turk compared to LDC held up with 40 and 60 hours of training.

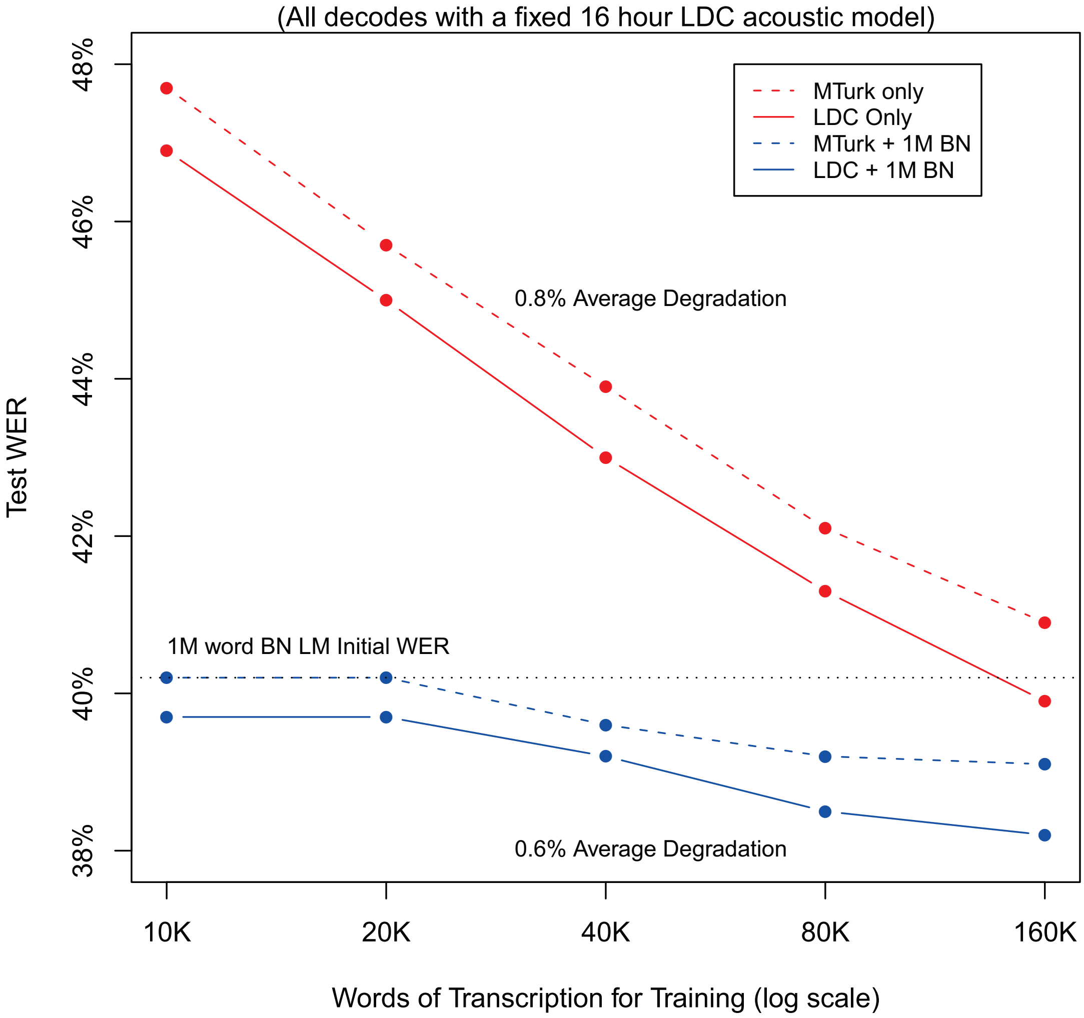

Figure 3.2: *Improving the Language Model* - All decodes used a fix 16 hour LDC acoustic model, with the amount of in-domain transcription for the *language model* training varying along the x axis. One million tokens of broadcast news were used as to build an initial language model. Interpolated with the available in-domain transcripts (either Mechanical Turk or LDC data), this additionally resource significantly improve absolute test WER. However, it does not dramatically change the usefulness of Mechanical Turk, with an average of 0.8% degradation becoming 0.6% with the out of domain data.

Given a fixed budget of 60 hours of transcription, the quality of 20 hours transcribed three times was contrasted to 60 hours transcribed once. The best one could hope to recover from the three redundant transcriptions is the utterance oracle. Oracle and singly transcribed data had 13% and 24% disagreement with LDC respectively. System performance was 40.9% with 20 hours of the former and 37.6% with 60 hours of the latter. Even though perfect selection cuts disagreement in half, three times as much data helps more.

One may be tempted to argue that *more* duplicate transcription is necessary. If there was some method that could recover LDC quality transcripts with N non-professional transcribers,one would have to compare to $20 \cdot N$ hours of transcription, which for $N = 3$ is already better than 20 hours of LDC. But if the two percent degradation holds, there is some operating point where the value of adding three times the amount of training is less than the value of quality control. That point is far away. Moving from 200 to 2000 hours of English Fisher training data only reduced WER by 3%.

The 2004 Fisher effort averaged a price of $150 per hour of English conversational telephone speech transcription. The company CastingWords produces high quality [105] English transcription for $90 an hour using Mechanical Turk by a multi-pass process to collect and clean Turker-provided transcripts, assumed to be of comparable quality to the private contractor used earlier. The price for LDC transcription is not comparable here since it was intended for more precise linguistic tasks. Extrapolating from Figure 3.3, the entire 2000 Fisher corpus could be transcribed using Mechanical Turk at the same cost of collecting 60 hours of professional transcription.

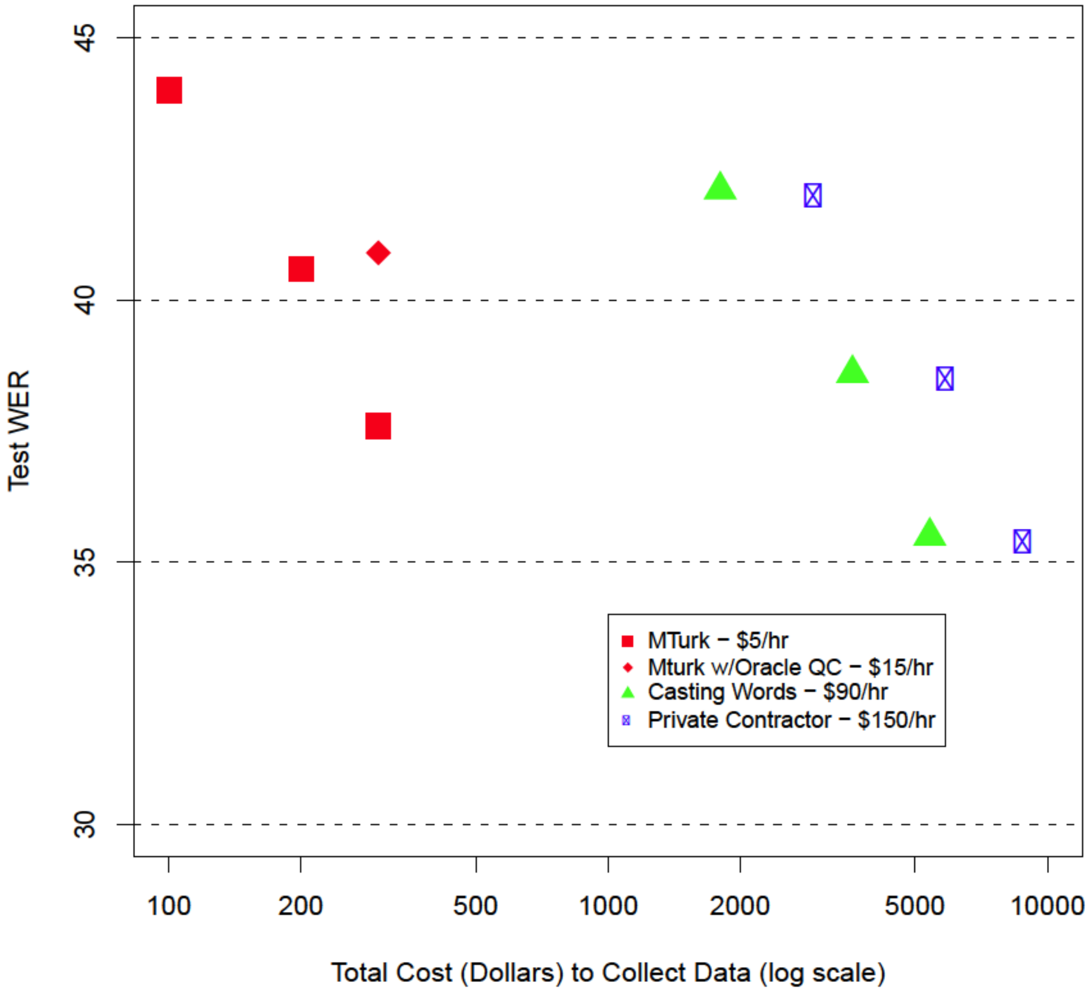

Figure 3.3: *Comparing Cost of Reducing WER* -Historical cost estimates are $150 per hour of transcription (blue circles). The company Casting Words uses Turkers to transcribe English at $90 per hour which was estimated to be high quality (green triangles). Transcription without quality control on Mechanical Turk (red squares) is drastically cheaper at $5 per hour. With a fixed budget, it is better to transcribe more data at lower quality than to improve quality. Contrast the oracle WER for 20 hours transcribed three times (red diamond) with 60 hours transcribed once (bottom red square).

3.4 Collection in Other Languages

To test the feasibility of improving low-resource languages, attempts were made to collect transcriptions for Korean, Hindi and Tamil CTS data. Korean was the only language with reference LDC transcriptions to use as a test set and thus could be used to build an LVCSR system.

3.4.1 Korean

Korean is spoken by roughly 78 million speakers world wide and is written in Hangul, a phonetic orthography, although Chinese characters frequently appear in written text. Since Korean has essentially arbitrary spacing [106], this work reports Phoneme Error Rate (PER) instead of WER, which would be unfairly penalized. Both behave similarly as system performance improves. English system performance with non-professional transcription (Section 3.3) degraded 2.5% for WER (39.5% to 42%) and 1.9% when computing PER (34.8% to 36.7%).

Ten hours of audio were uploaded to be transcribed once, again segmented into short snippets. Transcription was very slow at first and cost $0.20 per HIT to attract workers. Next, a separate HIT was posted to refer Korean transcribers, paying a 25% bonus of the income earned by referrals. This was quite successful as two referred Turkers contributed over 80% of the total transcription (at a cost of $25 per hour instead of $20). Three hours of transcriptions were collected after five weeks, paying eight Turkers $113 at a transcription rate of 10xRT.

Average Turker disagreement to the LDC reference was 17% (computed at the character level). Using these transcripts to train an LVCSR system, instead of those provided by LDC, degraded PER by 0.8% from 51.3% to 52.1%. For comparison, a system trained on the entire 27 hours of LDC data had 41.2% PER.

Although performance seems poor, it is sufficiently good to bootstrap with acoustic model self-training [67]. The language model can be improved by finding 'conversational' web text found with n-gram queries extracted from the three hours of transcripts [107].

3.4.2 Hindi and Tamil

As a feasibility experiment, one hour of transcription was collected in Hindi and Tamil, paying $20 per hour of transcription. Hindi and Tamil transcription finished in eight days, perhaps due to the high prevalence of Turkers in India [108]. While both languages lacked professional transcripts, Hindi speaking colleagues viewed some of the data and pointed out errors in English transliteration, but overall quality appeared fine. The true test will be to build an LVCSR system and report WER.

3.5 Quality Control sans Quality Data

Although this chapter has shown that redundantly transcribing an entire corpus gives little gain, there is value in *some* amount of quality control. System performance could be improved by only rejecting Turkers with high disagreement, similar to confidence selection for active learning or unsupervised training [1]. But if Turkers transcribing a truly new domain, there is no gold-standard data to use as reference, so disagreement must be estimated against erroneous reference. This section provides a practical method for quality control without gold standard reference transcription.

3.5.1 Estimating Turker Skill

Using the twenty hour English transcriptions from Section 3.3, each Turker was compared against the professional transcription for all utterances longer than four words. Note that each utterance was transcribed by an arbitrary subset of three distinct Turkers, so there is not a single set of utterances transcribed by all Turkers. Each Turker transcribed a different subset of the data with only partial overlap with any other Turker.

For a particular Turker, the disagreement with other Turkers was estimated by using the two other transcripts as reference and taking the average. The distribution of Turker disagreement follows a gamma distribution, with a tight cluster of average Turkers and a long-tail of bad Turkers as measured against the gold standard. Figure 3.4 shows the density estimate of Turker disagreement when calculated against professional and non-professional transcription. Estimating with non-professionals (even though the reference is 23% wrong on average) is surprisingly well matched to professional estimate.

Given that non-disagreement is a good estimate of disagreement over all of a Turker's utterances, how few need be redundantly transcribed by other non-professionals? For each Turker, one utterance was randomly selected and computed the "proxy" disagreement. This is contrasted with the estimate to the "true" disagreement against professional transcription over all of the utterances and repeatedly sample 20 times. Then the number of utterances was increased and used to estimate non-disagreement until all utterances by that Turker are selected. As few as fifteen utterances need to be redundantly transcribed to accurately estimate three out of four Turkers within 5% of the "true" disagreement.

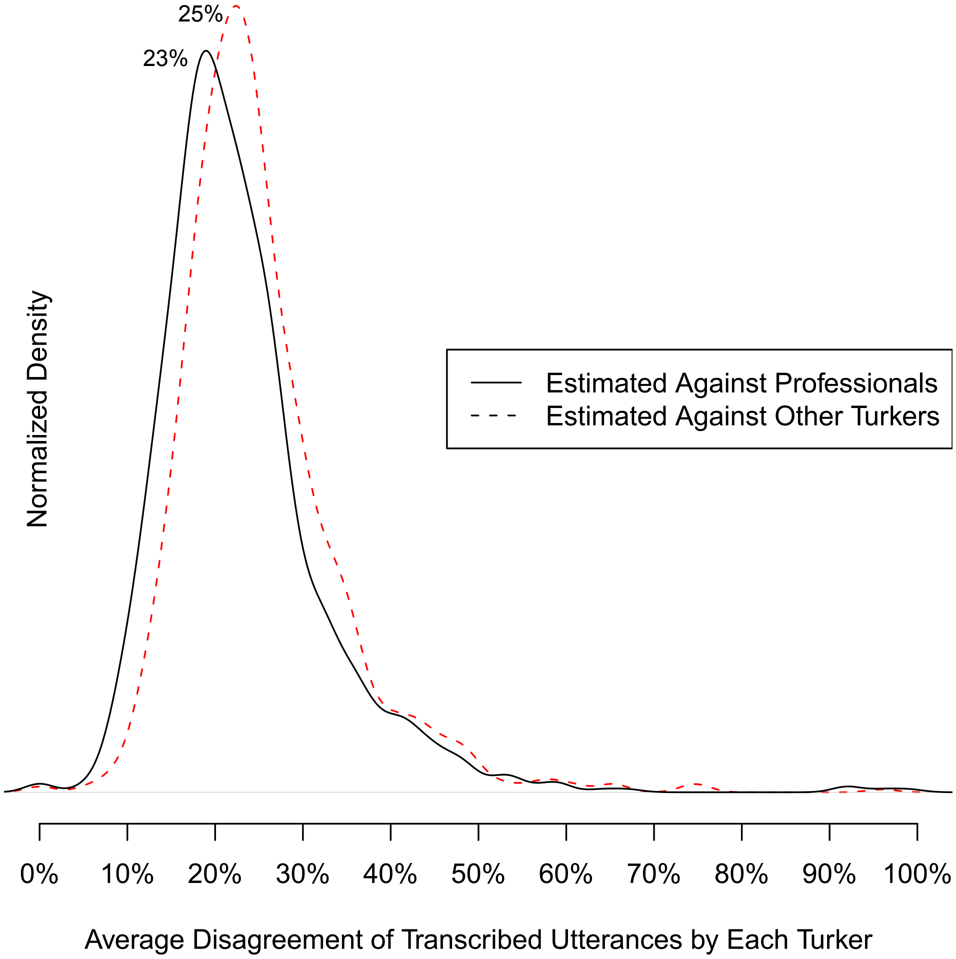

Figure 3.4: *Distribution of Turker Skill* - Each Turker was judged against professional and non-professional reference and assigned an overall disagreement. The distribution of Turker disagreement follows a gamma distribution, with a tight cluster of average Turkers and a long-tail of bad Turkers. Estimating with non-professionals (even though the reference is 23% wrong on average) is surprisingly well matched to professional estimate. Turker estimation over-estimated disagreement by only 2%.

Figure 3.5 shows a box plot of the differences of estimated to true disagreement on *all* utterances. Since using other Turkers, especially bad ones, as reference will unfairly overestimate disagreement, this estimate can be improved by iteratively re-ranking each Turker by their estimated disagreement. Instead of taking the average of each Turkers disagreement against the two other Turkers that transcribed a particular utterance, only the (estimated) better of the two was used to compute disagreement. This cut the average disagreement estimation error in half from 3% to 1.7%.

3.5.2 Finding the Right Turkers

Since a Turker's skill can be accurately predicted with as few as fifteen utterances on average, Turkers can be ranked by their true and estimated disagreements. By thresholding on true disagreement, either good Turkers can be selected or equivalently bad Turkers rejected. The ranking can be viewed as a precision/recall problem to select only the Turkers with true disagreement below the threshold. Figure 3.6 plots each Turker where the X axis is the disagreement and the Y axis is the estimated disagreement. Each Turker is a point with true disagreement (X axis) plotted against estimated (Y axis) disagreement. The estimated disagreement correlates surprisingly well with the true disagreement even though the transcripts used for the proxy reference are 23% wrong on average relative to the true transcripts. By setting a selection threshold, the space is divided into four quadrants. The bottom left are correctly accepted: both non-professional and disagreement are below the threshold. The top left are incorrectly rejected: using their transcripts would have helped, but they don't hurt system performance, just waste money. The top right are correctly rejected for having high disagreement. The bottom right are the troublesome

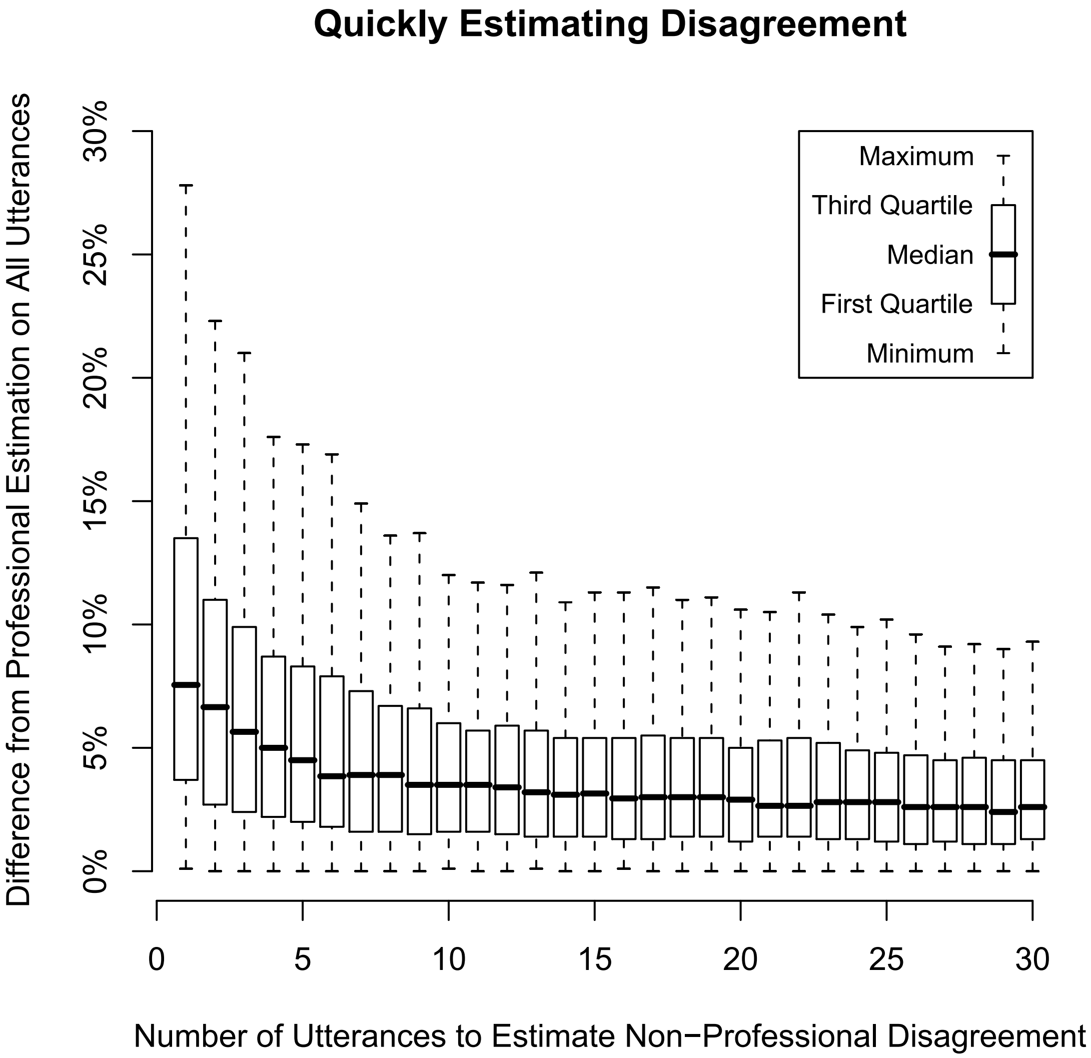

Figure 3.5: *Quickly Estimating Disagreement* - Box plot of the difference of non-disagreement with a fixed number of utterances to disagreement over all utterances. While error is expectedly high with one utterance, 50% of the estimates are within 3% of the truth after ten utterances and 75% of the estimates are within 6% after fifteen utterances.

false positives that are included in training but actually may hurt performance. Luckily, the ratio of false negatives to false positives is usually much larger. Sweeping the disagreement threshold from zero to one generates Figure 3.7, which reports F-score (the harmonic mean of precision and recall). It is difficult to find only good Turkers since the false positives outnumber the few good workers. However, rejecting bad Turkers becomes very easy once past the mean error rate of 23%. It is better to use disagreement estimation to reject poor workers instead of finding good workers.

This section suggests a concrete qualification test by first transcribing 15-30 utterance multiple times to create a gold standard. Using the transcription from the best Turker as reference and approving Turkers with a WER less than the average WER from the initial set.

3.6 Experience with Mechanical Turk

It was expected that managing Turker transcription would require the most effort. But the vast majority of Turkers completed the effort in good faith with few complaints about pay. Many left positive comments[5] despite the very difficult task. Indeed, the author's own disagreement on a few dozen English utterances were 17.7% and 26.8% despite an honest effort.

Instead, normalizing the transcriptions for English acoustic model training required the largest use of time. Every single misspelling or new word had to be mapped to a pronunciation in order to be used in training. Initially, any utterance with an out of vocab-

[5] One Turker left a comment "You don't grow pickles!!" in regards to the misinformed speakers she was transcribing.

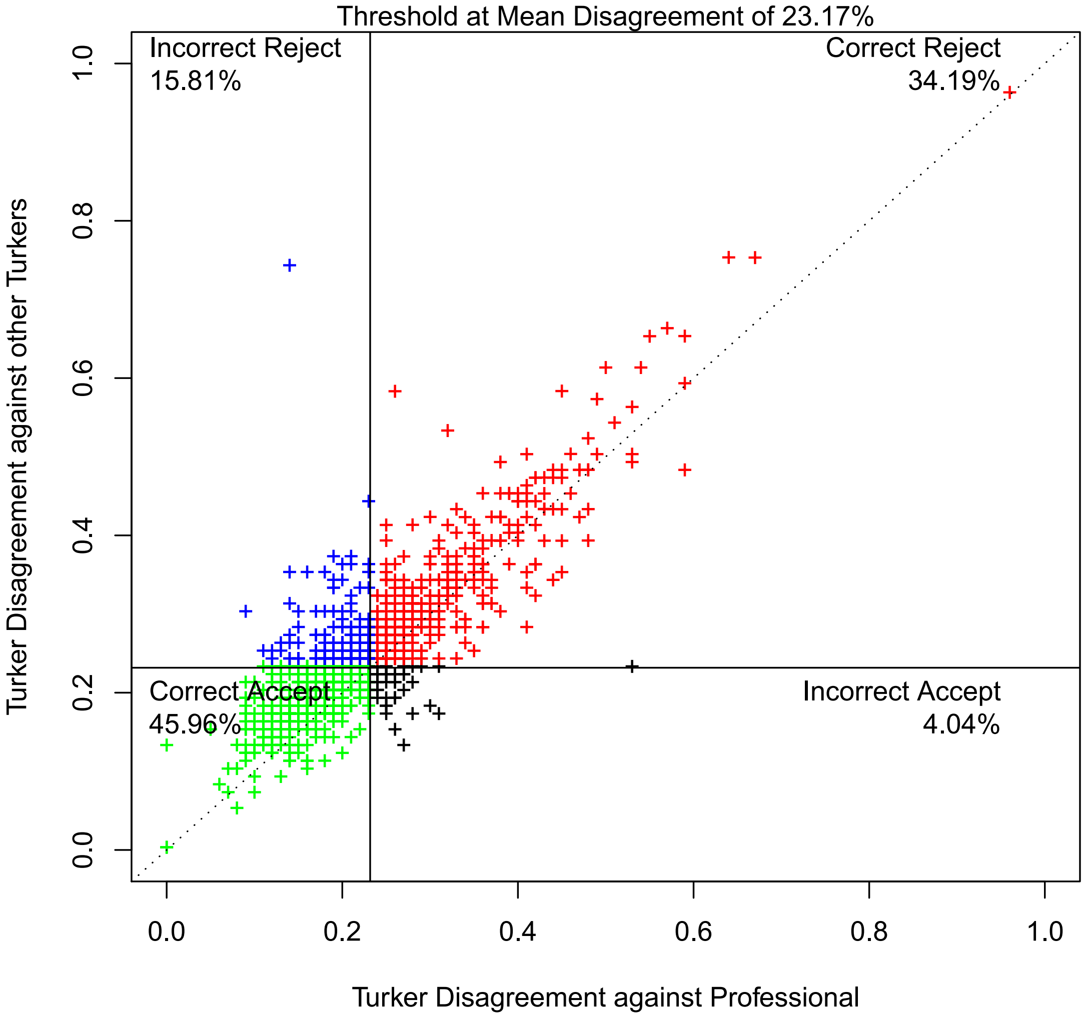

Figure 3.6: *Rating Turkers* - Each Turker is a point with professional (X axis) plotted against non-professional (Y axis) disagreement. The non-disagreement correlates surprisingly well with disagreement even though the transcripts used as reference are 23% wrong on average. By setting a selection threshold, the space is divided into four quadrants. The bottom left are correctly accepted: both non-professional and disagreement are below the threshold. The top left are incorrectly rejected: using their transcripts would have helped, but they don't hurt system performance, just waste money. The top right are correctly rejected for having high disagreement. The bottom right are the troublesome false positives that are included in training but actually may hurt performance. Luckily, the ratio of false negatives to false positives is usually much larger.

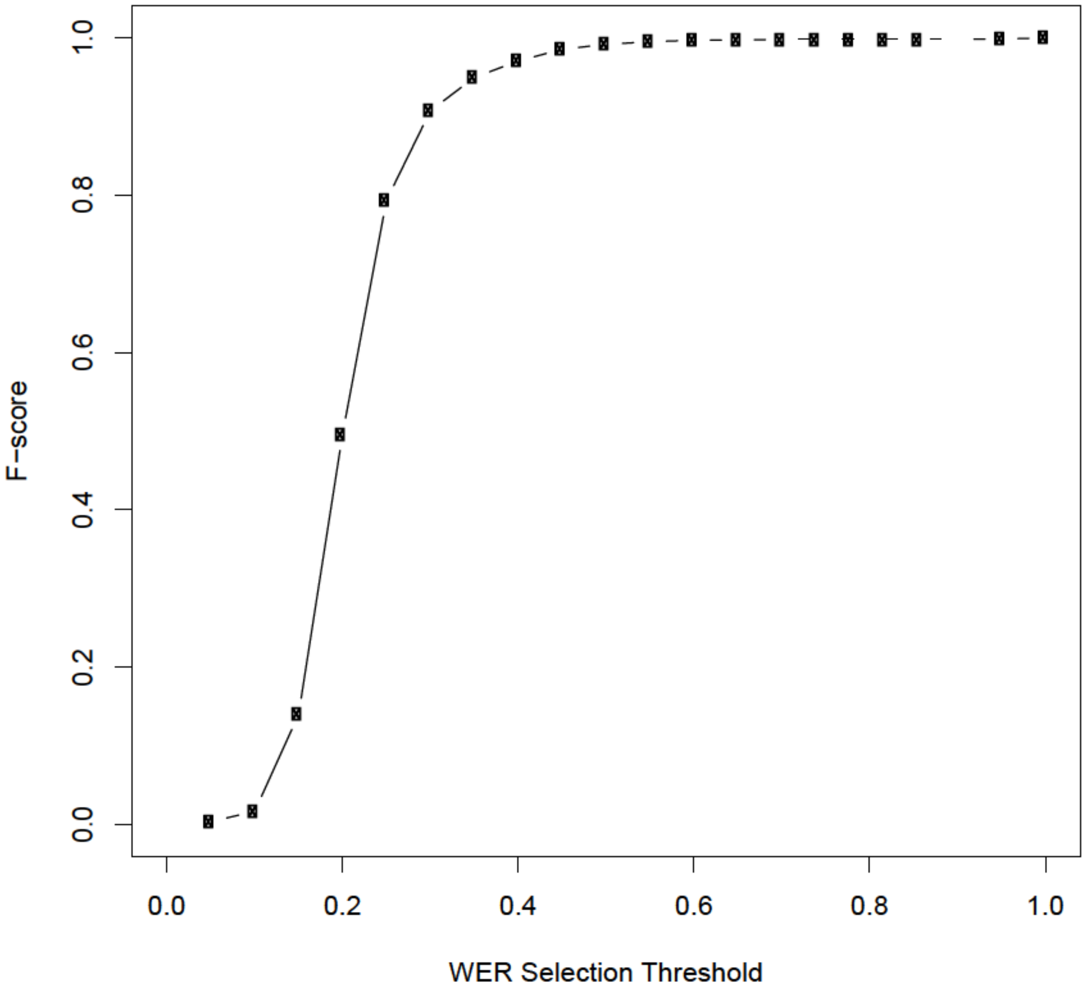

Figure 3.7: *Selecting Turkers* - The desired disagreement threshold was swept and used to select Turkers below that threshold using their estimated skill. Since the set of true Turkers below that threshold was known, the F1 score was computed - the harmonic mean of recall and precision. It is difficult to find only good Turkers since the false positives outnumber the few good workers. However, rejecting bad Turkers becomes very easy once past the mean error rate of 23%. It is better to use disagreement estimation to reject poor workers instead of finding good workers.

ulary word was discarded, but after losing half of the data,a set of simple heuristics was used to produce pronunciations. Even though there were a few thousand of these errors, they were all singletons and had little effect on performance. Turkers sometimes left comments in the transcription box such as "no audio" or "man1: man2:". These errant transcriptions could be detected by force aligning the transcript with the audio and rejecting any with low scores [109]. Extending transcription to thousands of hours will require robust methods to automatically deal with errant transcripts, and may additionally run the risk of exhausting the available pool of workers.

Modeling Korean pronunciations was straightforward since the language is phonetic. Syllables were split into component Jamo characters and treated the graphemes as the pronunciation. Finding Korean transcribers required the most creativity. There was success in interacting with the transcribers, providing feedback, encouragement and paying bonuses for referring other workers. Cultivating workers for a new language is definitely a "hands on" process. For Hindi and Tamil, Turkers sometimes misinterpreted or ignored instructions and translated into English or transliterated into Roman characters. Additionally, *some* linguistic knowledge is required to classify phonemic categories (like fricative or sonorant) required for acoustic model training.

Our goal was not to scientifically examine Mechanical Turk, but to demonstrate that linguistic resources can be collected without the need of high-quality transcription. While it was an accessible platform for research, in-house transcription for real world applications would be preferable. It was difficult to re-engage excellent transcribers, relying on a one-time bonus to motivate them to continue work. There was also considerable difficulty effectively communicating the transcription instructions. A ten minute face to face conversation would have improved the experience for both the researchers and the transcribers.

3.7 Discussion

Unlike previous work which studied the intrinsic quality of Mechanical Turk annotations alone, this chapter judged its value in terms of the real task: improving ASR system performance. Despite relatively high disagreement with professional transcription, data collected with Mechanical Turk was nearly as effective for training speech models. Since this degradation is so small, redundant annotation to improve quality is not worth the cost. Resources are better spent transcribing additional speech. In addition to English, this chapter demonstrated similar trends in Korean and also collected transcripts for Hindi and Tamil. Finally, this chapter proposed an effective procedure to reduce costs by maintaining the quality of the annotator pool without needing high quality annotation.

This chapter used Mechanical Turk and conversational English as a test bed for efficient transcription regimes with non-expert transcribers. Applying these results to real low-resource domains, such as conversational Hindi or medical domains would require additional effort. For other languages, the largest stumbling block is access to a large labor pool. Based from the United States, Mechanical Turk is dominated by English speaking workers. This gave us, the employer, market power to set lower wages, enjoy quick turn around and raise the bar for transcriber quality. As reported on Korean, more rare languages will demand higher wages.

Other issues arise for non-written languages such as diglossia or African American

Vernacular English (AAVE). Agreeing on a transcription vocabulary would be too onerous for non-expert transcribers in a non-controlled marketplace. However, one could utilize non-experts in a two-pass strategy of vocabulary *reduction.* First, non-experts transcribe a corpus of audio from the target language. Since there is no one correct canonical written form, the written vocabulary will be larger than the true set of word types in the audio. The set of word types seen in the collected transcripts would then form the initial vocabulary.

Our task would then be to detect and merge different orthographies of the same acoustic realization. This differs from automatic spelling correction as first, there is no canonical vocabulary and second, there is additional information from the underlying acoustics. One could employ either automated or human-in-the-loop methods to first detect likely word pairs and second to decide if they are of the same acoustic signal. Non-experts could be presented with an audio snippet and two orthographic spellings and tasked with the binary decision. This vocabulary reduction would increase the number of training samples for language model estimation and reduce the artificially high vocabulary size.

Moving within English, but to other *domains* brings a different set of challenges. While conversational telephone speech is relatively difficult acoustically, tougher noise environments exist. This dissertation did not evaluate how well non-expert transcribers can overcome acoustic noise. They may be more inclined to skip acoustically difficult audio, resulting in a recognizer which does not have examples of acoustic noise. Furthermore, meeting or lecture data may have multiple speakers on one microphone, which was not accounted for in this work.

Another difficulty is that some domains may be sensitive and not widely releasable such as financial interactions or medical recordings. These situations prevent the use of Mechanical Turk or other open platforms, which reduces the worker pool.

Finally, novel domains are likely to have novel words, which may be unfamiliar to a non-expert. Biology lectures, doctor notes and stock trading conversations will all have specific, uncommon vocabularies. One could compensate for this by priming a transcriber with words likely to appear in an utterance. Using the methods of Chapter 5, one could first decode an utterance and present a list of keywords (with proper spellings) likely to appear. Or post-transcription efforts could detect for mis-transcriptions. First, a transcript could be expanded into a lattice through a phonetic or word-edit confusion network. This would add domain specific keywords which may have been misspelled into the search lattice. This heavily constrained lattice could then be re-decoded and the correct word (with correct pronunciation) may be preferred.

This chapter is the foundation for the next two. When deploying an LVCSR system to a new domain with limited resources, one should not exhaust all human labor collecting transcription. Instead, deploy some labor to collect a few hours (less than a dozen) for acoustic and language modeling training and for evaluation.

Chapter 4 will assume these dozen hours of manual transcripts are available for use in bootstrapping a semi-supervised language model. The methods of Chapter 4 could be applied to these results in two ways. First, one could weight the non-expert transcripts of this chapter by the estimated transcriber quality (Section 3.5.1). Section 4.7 then details how one could place more emphasis on the high-confidence transcripts by adapting to them through MAP adaptation. Second, if some high quality data is available, then these non-expert transcripts could serve as a useful background model from which to adapt.

Chapter 5 presents an alternative use of human labor than direct transcription in Section 5.6. Non-experts can be effective when the annotation task is constrained to focus on the end problem of keyword search.

Chapter 4

Semi-Supervised Language Model Estimation

After a constrained budget for human transcription has been spent (see Chapter 3), automatic classifiers are an additional source of inexpensive, but noisy labels. Unlike humans, they have no innate transcription ability and must be bootstrapped from either out of domain data or from whatever in-domain transcripts are available. This chapter considers language model estimation from automatic output by an LVCSR system trained on a small amount of in-domain data.

Labels provided by an LVCSR system will significantly differ from human judgments. First, the classifier is able to provide a posterior estimate over the entire space of labels while human transcription requires redundant judgment for alternate hypotheses. Second, the classifier can inexpensively label a large audio corpus, while the human transcription budget might not be large enough. Third, and perhaps most importantly, the low-resource condition of this dissertation means the error rates will be much higher than human transcribers. Because of these differences, the best techniques for building a language model in a semi-supervised manner may differ from the previous chapter.

This chapter will explore the most efficient use of automatically generated transcripts in conjunction with a small amount of high-quality transcripts. Section 4.4 motivates the use of expected counts for semi-supervised learning and details improved n-gram count estimates. Section 4.5 explains the limited success of language model self-training with back-off models. Section 4.7 compares log-linear models to back-off models using standard n-gram features. Finally, Section 4.8 uses the log-linear framework to introduce *marginal class constraints* to encode domain knowledge of transcription errors.

4.1 Previous Work

4.1.1 Semi-Supervised Language Modeling

Prior work has considered the choice of the initial model to generate labels, the choice of data and the choice of methods to filter data output. This line of research has received little attention from the NLP community since for most tasks text is the observed data in need of labeling. Additionally, the machine translation community typically deals with translating from rare languages into more common ones – which benefits from large language modeling corpora in the more common language. The most fertile area of research has been the automatic speech recognition community, where such techniques can most likely have an impact.

Previous work has mostly considered estimating n-gram features for use in a back-off language model. Seventeen hours of call center speech were decoded with an LVCSR system built with voice mail transcripts in [45]. From the seventeen, four hours of automatic transcripts estimated to be most correct were selected. Unweighted n-gram counts from the automatic transcripts were used for MAP adaptation of the initial language model with a Dirichlet prior. This resulted in 50% of the total possible gain if the four hours were manually transcribed – a 4% absolute reduction in WER. Interestingly, when self-adapting on the one hour held-out test data, performance did not increase.

The small gain for self-adaptation is supported in another experiment in creating adaptive spoken dialog systems [75]. Call center data was again used to adapt an existing language model with expected n-gram counts from lattice posteriors. Instead of using all the automatic data, words with posterior scores below a threshold were mapped to a common token. The remaining counts were then pooled with the initial language model. This form of count thresholding did not significantly impact results, with an increase in word accuracy of 0.5% to 0.8% versus the upper bound of 2.2%.

Further work combined semi-supervised learning with active learning [76]. Utterance confidences were estimated and used to sort automatically decoded data from interactive dialog systems. The automatic transcripts with confidence above a threshold were added to an existing language model. The optimal performance achieved 40% of the possible gain. Transcribing the remaining low-confidence utterances further improved performance. This marrying of semi-supervised learning with active learning more efficiently reduced WER than randomized transcription.

Another source of in-domain text is from automatic translations of data from a similar domain in another language [110]. English conversational telephone speech (CTS) was translated into Czech using commercial systems trained on broadcast news data. 780M words of web data and movie transcripts were combined with automatic translations of 11M words of English CTS transcriptions. The addition of the automatic transcripts reduced WER on Czech CTS by 1.5% (no upper bound is known).

In general, these methods produced estimated n-gram counts from automatic output which were then used as training data for a standard generative n-gram language model. The key strength of these methods was access to inexpensive and plentiful sources of audio. Yet the previous experiments have only used dozens of hours of automatic labels. This lowered the upper bound of potential gain and may lower the achieved gain of the semi-supervised methods. Unfortunately, there is no consistent explanation or convincing analysis which explains the modest results of semi-supervised learning.

Previous work has not explored semi-supervised learning of other classes of language models, such as log-linear or continuous space models. There may be classes of models which are more robust to errors in recognition. Additionally, features beyond n-gram counts may be more robustly estimated from automatic output.

Semi-supervised discriminative language models have also received attention from the research community [111]. Unlabeled broadcast news audio was used to create neighborhoods of likely word confusions of an LVCSR system. A log-linear exponential language model was then discriminatively trained by expanding the newswire text with the in-domain confusion neighborhoods. In a similar manner to domain adaptation, higher-level statistics (confusion neighborhoods) were estimated, but this time from noisy in-domain data.

Further semi-supervised discriminative language modeling generated pseudo confusions from a channel noise model of Turkish ASR [74]. A perceptron re-ranker was discriminatively trained using negative examples hallucinated from text reference. The resulting model modestly outperformed the baseline system. In this manner, in-domain speech audio was not required, although a channel noise model must be learned from data.

Semi-supervised discriminative training is a tall challenge since discriminative training relies on negative examples to push parameters in the right direction. Further, discriminative models outperform generative models (like n-gram language models) only with large amounts of training data. Even worse, the gains for discriminative language models are not significantly greater than classic n-gram language models. Although this chapter does consider a log-linear model, it will not use a discriminative model.

4.1.2 Semi-Supervised Log-Linear Modeling

Log-linear language models were first introduced in the NLP community as a form of self-adaptation [112]. Unigram constraints on in-domain data were enforced by minimization of Kullbacker-Leiber distance of the modeled distribution to a target distribution and an arbitrary set of constraints. The resulting minimum description length estimate allowed for on-line adaptation of the language model for a dictation system. This work follows inspiration from the wider statistical community of density estimation by the minimum cross-entropy and minimum discrimination information criteria [113] [114] [115]. When the target distribution is uniform, these models take on the name of maximum entropy models, which is well developed in the statistics literature [116]. The inspiring principle is that a model should satisfy constraints, but be uniformly agnostic to unspecified constraints. Conditional maximum entropy models, which are obtained by applying the maximum entropy principle to conditioned models, are an important tool in the NLP community, providing state of the art results ranging from named entity tagging to machine translation.

Further work [117] introduced maximum entropy language models as a general framework to incorporate a variety of features beyond n-grams. Motivated as an extension of linear interpolation, the first experiments used trigger features to beat performance with n-gram features alone. While they incorporate n-gram features, maximum entropy models extend beyond back-off smoothing of count estimates. Parameters are learned by minimizing a loss function, compared to the heuristic motivations of earlier smoothing work.

Additional features such as topic information [118], cache models [119], and classes [120] were also integrated. State of the art performance for language modeling was recently achieved with a class-based log-linear language model [55]. By approximating features seen in training with class estimates, this method reduced training bias and resulted in significant reductions in error over standard n-gram language models.

The primary limiting factor preventing wider adoption is the large computational cost during parameter estimation. Computation of the loss-function gradient requires computation over the entire space of labels – the thousands of words in a vocabulary. Compared to other NLP tasks like part of speech or named entity tagging, the larger event space prevents log-linear models from scaling to massive amounts of training data. Research into more efficient computation [121] and approximations [122] [123] allowed for easier training and application of maximum entropy models.

Further research continued the original motivation of log-linear models: domain adaptation. Log-linear models are well suited for domain adaptation since they easily incorporate arbitrary classes of constraints. If domain-specific probabilities are known, the MDI criterion provides a rigorous framework to incorporate such knowledge that outperforms count pooling and interpolation [124]. More general constraints, such as unigram marginal probabilities, have also been used to adapt background models to a new domain. This useful constraint requires only small amounts of in-domain data to reliably estimate [125].

Recent work married hierarchical Bayesian modeling with maximum entropy models [126] [127]. Parameters across different domains are jointly learned to maximize performance on all test sets. This kind of loss functions encourage parameters from similar domains to have similar values – which helps overcome data sparsity for domains with small amounts of data. Experiments on Estonian automatic speech recognition demonstrated that this method outperformed count pooling and model interpolation of both n-gram and maximum entropy models. However, the authors cite considerable shortcomings – the increased memory and computation requirements and the sensitivity of the models to specification of hyper-parameters.

A method of unsupervised learning of conditional log-linear models, contrastive estimation, was used to improve part of speech tagging of text [128]. Perturbations of true word sequences were created by word insertions, deletions and substitutions. Model parameters were then estimated by generating latent variables which preferred the true observed data over the degraded negative examples. One difficulty in applying this model to semi-supervised learning of language from speech is the wide gulf between acoustic feature vectors and latent word sequences. Meaningful transformations of acoustic features which result in negative language modeling samples may be difficult to create.

In between fully supervised and unsupervised learning lies work in learning from multiple labels. An observed data sample may have multiple labels associated with it instead of the single label classically assumed in machine learning. There may be valid multiple labels, such as alternative descriptions of an image, or the labels may arise from ambiguity in the data or annotator disagreement. The machine learning community has extended formalisms to this new domain, such as k-NN classification [129]. However, the set of possible labels have been very small (say less than ten) while in this work, this chapter will consider a huge space of possible labels.

Log-linear models are especially relevant to multi-label learning since a natural loss function for this domain is KL divergence instead of likelihood. Since there is no longer one set of joint observation and label pairs, but multiple labels for each observation, there is not one set of data for which to maximize likelihood. Similar to the MDI criterion, the model family that minimizes KL divergence is a log-linear model. Applications of this formalism in the NLP community have visited classification problems such as image labeling [130], face identification [131] or named entity tagging [132].

Previous work either assumed each label was equally likely or had posterior probabilities. Importantly, they assumed these probabilities were correct. Additionally, some work assumed the prior probability of the labels were known, which is the very knowledge this dissertation is attempting to discover. Finally, the set of ambiguous labels considered in prior work was small for each instance. It is not clear that the previous techniques could cope with a label space of a large vocabulary speech recognizer (50,000+ words).

4.2 Experimental Description

The following experiments are aimed at improving the language model alone. Unlike self-training (Section 2.4.1) where a classifier generates labels for itself, this chapter is more appropriately viewed as co-training (Section 2.4.2). Since the language model is a prior only over the transcripts, it has no interaction with the observed audio and is unable to generate training samples. Except for Section 4.7, the acoustic model training data will be fixed throughout this chapter to ten hours of high-accuracy in-domain transcripts. This resource condition represents a middle point between the low-resource settings. Previous work on low-resource acoustic modeling [85] [1] used as little as one hour of transcribed data, but in conjunction with a strong 1B token language model. Section 4.7 will more fully explore a range of in-domain data from as little as 2.5 hours to 40 hours.

For convenience, this work will use the following notation to describe a semi-supervised training experiment. The condition $X + Y$ (such as "10+190") means that X hours of in-domain speech transcripts were used to build the initial acoustic and language model and Y hours of unlabeled speech were used for further semi-supervised training of the language model.

4.2.1 Corpora

The experiments in this chapter are on the Fisher English conversational speech corpus. As described earlier (Section 3.2), the corpus is a collection of phone calls between strangers about assigned topics. The amount of in-domain transcripts will vary in this work from 2.5 to 40 hours of manually transcribed audio. These transcripts were provided by the

Fisher QuickTrans effort [3] with negligible error rate and so this work does not explicitly model the manual transcription errors. The 400 hours of manually transcribed Fisher audio will be treated as the unlabeled corpus. This data will be decoded with the initial LVCSR system and used to generate expected n-gram counts from the domain. Since in reality it *is* manually transcribed, this chapter will also be able to measure the performance of manual transcription on this set, which will serve as an upper bound on the performance of the semi-supervised training methods. The vocabulary is a fixed 75k word phonetic dictionary which happen to cover the unlabeled audio but not the entire held-out test set. WER was calculated on a held-out three hour test set from the HUB-5 2004 evaluation.

4.2.2 LVCSR Pipeline

As in other chapters, this one uses BBN Technologies BYBLOS LVCSR system detailed in Section 2.1.2. The acoustic model is a multi-pass state-of-the-art LVCSR system that uses state-clustered Gaussian tied-mixture models [19]. MFCCs with mean and variance normalization were used as well as Vocal Tract Length normalization. In this chapter, only used maximum likelihood estimation instead of discriminative training or acoustic model self-training, to gain scientific insight. Section 4.6 shows that there is little impact of a stronger acoustic model on semi-supervised language modeling. While a deployed system should use whatever tools are available to achieve the lowest WER, the acoustic model in this work is a reasonably strong system. Decoding requires three passes: a forward and backward pass using triphone models and a trigram LM to generate an n-best list, which is then re-scored using quinphone acoustic models. These three steps are repeated after speaker adaptation of the acoustic model using constrained maximum likelihood regression.

It is possible to build semi-supervised acoustic models [1] (and real world deployments *should*), but to control for conflating variables, this chapter does not.

4.2.3 Determining Significance

The goal of this dissertation is to propose novel methods which significantly impact LVCSR performance. Some of the semi-supervised results reported in this chapter will be ambiguously close to the baseline system in terms of WER. This is partly due to the weakness of the semi-supervised methods, but also the historically low impact of improved language modeling when measured by WER.

Significance is a multi-faceted concept. One could argue for human evaluation studies to measure the impact on a real world task. Recent work has used non-experts to evaluate statistical machine translation systems with good effectiveness [38]. However, this dissertation does not claim the modest results in this chapter will significantly impact human perception of automatic transcription. The total possible gain for semi-supervised language modeling in this chapter is 9% absolute WER. Perfectly capturing this gain would mean an additional one out of ten words will be correct. Of course, many small steps lead to big gains, so this chapter report reductions in WER. For those results that are questionably close, statistical significance is reported. The research community has proposed the Matched Pairs Sentence-Segment Word Error (MAPSSWE) Test as a standard for comparing two systems [133].

This is a t-test for estimating the mean difference of normal distributions with unknown variances. Unlike other statistical tests, The MAPSSWE test varies the sample length to ensure the validity of the independence of assumption. Instead of comparing at the utterance or word level, the test constructs sub-utterance phrases bounded by words correctly recognized by both systems. This increases the number of samples, but ensures that all samples have the same acoustic and linguistic context. The acoustic stream was divided into segments such as the errors in one are statistically independent of errors in any neighboring segment.

Let N_1^i and N_2^i be the WER of the i^{th} segment by systems one and two respectively. Let $Z^i = N_1^i - N_2^i, i = 1, 2, \ldots, n$, where n is the number of segments. Then μ_Z is the unknown mean difference in the WER of the two systems. The null hypothesis is that $\mu_Z = 0:$ the two systems have no difference in WER in the limit.

To test this hypothesis, let the estimated mean

$$\hat{\mu}_Z = \sum_{=1}^{n} \frac{Z_i}{n} \tag{4.1}$$

and the estimated variance

$$\hat{\sigma}_Z^2 = \frac{1}{n-1} \sum_{i=1}^{n} (Z_i - \mu_Z)^2 \tag{4.2}$$

such that the test statistic

$$T = \frac{\hat{\mu}_Z}{(\hat{\sigma}_Z / \sqrt{n})} \ . \tag{4.3}$$

Then for large enough n (> 50 has been proposed), T will be approximately normal with unit variance. Under the null hypothesis H_0, μ_Z has zero mean, T will as well. The statistic $p = 2\Pr(Z \geq |T = t|)$, where $Z \sim \mathcal{N}(0, 1)$ - note this is a two-tailed test since the only desired knowledge is if systems one and two are *different*, not specifically which is *better*. For a specified confidence level α, H_0 can be rejected. This value was calculated on the automatic transcripts of the held-out data using tools provided by NIST [134]. Throughout this chapter, significance is reported when $p < 0.001$ for the key experimental results.

4.3 Language Model Description

As detailed in Section 2.2, a language model uses conditional distributions to estimate the probability of a word sequence $w_1, w_2, \ldots w_n$ occurring from some domain as

$$P(w_1, w_2, \ldots w_n) = \prod_{i=1}^{n} P(w_i | \underbrace{w_1, \ldots, w_{i-1}}_{\text{history}}) \,. \tag{4.4}$$

While there are a variety of models to tackle this problem, this work considers the standard *non-parametric* back-off language models and *log-linear* models. Both models are not whole-sentence models but instead rely on the Markov assumption, where the conditional distributions $P(w_i|w_1, \ldots w_{i-1})$ are collapsed into the same histories depending on the n-gram length (up to trigrams for this work). Back-off language models are essential in speech recognition for their efficient estimation and compact representation. While estimation is more complex, the advantages of log-linear models will become clear in Section 4.8.

4.3.1 Non-Parametric Modeling

The language modeling literature typically conflates s with model *formulation.* One might state they are using a “Modified Kneser Ney smoothed trigram” language model and the computation is implicitly understood. As noted in prior work [24], most smoothing methods for the conditional probability of word w_i following history w_{i-n+1}^{i-1} can be expressed (albeit clumsily) as

$$P(w_i|w_{i-n+1}^{i-1}) = \begin{cases} \alpha(w_i|w_{i-n+1}^{i-1}) & \text{if } c(w_{i-n+1}^{i}) \geq 0 \\ \gamma(w_{i-n+1}^{i-1}) P(w_i|w_{i-n+2}^{i-1}) & \text{if } c(w_{i-n}^{i}) = 0 \end{cases} \tag{4.5}$$

where n is the *order* of the language model ($n = 3$ for trigram LMs), $c(w_i^j)$ is the count of a word sequence w_i^j in the training corpus, $\alpha(w_i|w_{i-n+1}^{i-1})$ is the probability estimate when the word sequence w_{i-n+1}^i was seen in the training data and $\gamma(w_{i-n+1}^{i-1})$ is a normalization factor so that the *back-off* estimate $P(w_i|w_{i-n+2}^{i-1})$ is scaled appropriately to make $\hat{P}(w_i|w_{i-n+1}^{i-1})$ sum to one over all w_i.

An alternate interpretation also shows smoothing as interpolation with lower-order estimates [28] with a one-to-one mapping between back-off and interpolation. These models are classified as *non-parametric* because there are no estimated parameters from data once the choice of smoothing method is made. The task of *smoothing* is to compute the α's and γ's from the observed n-gram counts. Modified Kneser-Ney smoothing [28] is standard for supervised language model estimation in LVCSR and is defined as

$$P(w_i|w_{i-n+1}^{i-1}) = \frac{c(w_{i-n+1}^i) - D(w_{i-n+1}^i)}{\sum_{w_i} c(w_{i-n+1}^i)} + \gamma(w_{i-n+1}^{i-1})P(w_i|w_{i-n+2}^i) \tag{4.6}$$

which is a form of *absolute discounting* since the relative frequency estimate $\frac{c(w_{i-n+1}^i)}{\sum_{w_i} c(w_{i-n+1}^i)}$ is discounted by a constant $D(k)$ that depends on the frequency k of the highest order n-gram. Typically, counts less than three are given their own discounts, given by

$$\begin{aligned} D_1 &= 1 - 2\frac{n_1}{n_1 + 2n_2} \cdot \frac{n_2}{n_1} \\ D_2 &= 2 - 3\frac{n_1}{n_1 + 2n_2} \cdot \frac{n_3}{n_2} \\ D_{3+} &= 3 - 4\frac{n_1}{n_1 + 2n_2} \cdot \frac{n_4}{n_3} \end{aligned}$$

which depends on knowledge of the unique number of words that occurred $n_i = 1, 2, 3$ times, which requires integer counts. This requirement, logical for actual observed data, prevents the use of expected counts. Instead, Witten-Bell smoothing was used, which is competitive with modified Kneser-Ney and is defined as

$$P(w_i|w_{i-n+1}^{i-1}) = \frac{c(w_{i-n+1}^{i}) + N(w_{i-n+1}^{i})P(w_i|w_{i-n+2}^{i})}{\sum_{w_i} c(w_{i-n+1}^{i}) + N(w_{i-n+1}^{i-1})} \tag{4.7}$$

where $N(w_{i-n+1}^{i-1})$ is the number of unique words seen following the history w_{i-n+1}^{i-1}. Instead of the absolute discounting of modified Kneser-Ney, Witten-Bell is a linear interpolation of the highest order estimate and a back-off with the weight depending on $N(\cdot)$ Note that this is a function of n-gram *types* and not observed *token* counts. The use of this statistic $N(\cdot)$ is motivated by the 'fertility' of a history - those that are seen in the training data with many unique word types following should be more likely to see a novel word following it. Witten-Bell smoothing is competitive with modified Kneser-Ney for use in speech recognition. Non-parametric language model estimation is attractive from an engineering perspective. Estimation requires no more than $O(T + B + U)$ time for all unique unigram, bigram and trigram types seen in the data.

4.3.2 Log-Linear Language Modeling

Like non-parametric language models, a log-linear language model is also typically a *conditional* model since it does not model entire sentences. Instead, the outcome space is a fixed vocabulary plus the end of sentence marker. The probability of a history $h \in \mathcal{H}$ followed by $w \in \mathcal{V}$ is

$$P(w|h) = \frac{\exp \sum_{k=1}^{K} f_k(h, w) \cdot \theta_k}{\sum_{w' \in \mathcal{V}} \exp \sum_{k=1}^{K} f_k(h, w') \cdot \theta_k} \tag{4.8}$$

where K feature functions $f_k : \mathcal{H} \times \mathcal{V} \rightarrow \mathcal{R}$ and estimated model parameters $\theta_k \in \mathcal{R}$. Although there is one model and one set of parameters $\{\theta_k\}$, there is a separate distribution for all possible histories h, for instance $\mathcal{V} \times \mathcal{V}$ possible bigram histories for a trigram language model (plus one more for the start of sentence token). Tying these distributions together are the feature functions $\{f_k\}$ for language modeling and their associated feature weights. A feature function takes as input the $|\mathcal{H}| \times |\mathcal{V}|$ space of possible histories and words in the vocabulary and maps them to a real value. Language model feature functions are often binary valued, mapping to one if a feature "fires" on a history and word pair, but zero otherwise. As expected, the most commonly used functions are n-gram features. Equations (4.9) and (4.10) show an example unigram and bigram feature functions. For a history word pair, Equation (4.9) essentially asks "is the word CAT, regardless of the history?" by defining the binary function

$$f_{\text{CAT}}(h, w) = \begin{cases} 1 & w = \text{CAT}, \\ 0 & \text{otherwise} \ ; \end{cases} \tag{4.9}$$

and it can similarly be asked if "is the word CAT *and* the preceding word FAT?" by creating the bigram feature function

$$f_{\text{FAT CAT}}(h, w) = \begin{cases} 1 & h = \text{FAT} \ \& \ w = \text{CAT}, \\ 0 & \text{otherwise} \ ; \ . \end{cases} \tag{4.10}$$

Feature functions are not limited to lexical entries. The original use offor log-linear language models [122] was for incorporating "trigger" features that allow for longer-history contexts. This flexibility allows log-linear models to elegantly incorporate a variety of information as long as it can be expressed through a feature function and associated with an "expected count". Section 4.8.1 will use this fact to introduce class constraints that fire for groups of words. Histories need not be specific word subsequences either, and can be a class of word subsequences from a particular topic, speaker or other common meta features.

While the space of possible feature functions is infinite, only those that are *constrained* matter to the language model. The goal of estimation in log-linear modeling is that the expected frequency under the model match the empirical frequency. Any unconstrained feature will have an unspecified empirical frequency and the optimal feature weight will be zero under the maximum entropy principle. In the previous two examples, the constraints for Equation (4.9) might be the "unigram" frequency y of the word 'CAT' in some training data and for Equation (4.10) would be the "bigram" frequency of FAT CAT. The constraints need not be empirical frequencies, but some reasonable value. The total number of parameters (K in Equation (4.8)) in the model is the number of constrained features. For supervised modeling with n-gram features, this would be on the number of unigrams, bigrams and trigrams seen in training data.

Parameter estimation of a log-linear language model requires computing

$$\mathbb{E}[f_k] = \sum_{w \in \mathcal{V}} \sum_{h \in \mathcal{H}} P(h,w) f_k(h,w) \tag{4.11}$$

which the naive implementation per iteration is $\mathcal{O}(\mathcal{V}^3)$ for a simple trigram model, which is impractically slow for the vocabulary sizes of LVCSR systems. The first key speed up [135] is to approximate the joint probability $P(h,w)$ with the conditional likelihood under the model multiplied by the empirical frequency of the history: $P(w|h)\tilde{P}(h)$. This removes the sum over all histories - $\mathcal{O}(\mathcal{V}^2)$ - and reduces it to the seen histories from training data - $\mathcal{O}(|D|)$. Second, use the hierarchical training technique which reduces the training time to that of a standard back-off language model: $\mathcal{O}(U + B + T)$ [121] where $\mathcal{U}, \mathcal{B}$ and $\mathcal{T}$ are the number of unigram, bigram and trigram types seen in training data, respectively.

Finally, this work also takes advantage of the hierarchical nature of n-gram features to encode the log-linear model into ARPA format [121]. This allows the use of a log-linear model not just in n-best re-scoring, but in the full forward and backward passes of the LVCSR system and benefit from the improved model throughout decoding.

4.4 Estimating Expected Counts

Estimating a log-linear language model requires specifying the empirical counts of each feature function. Under the supervised scenario, these counts are drawn from in-domain text. Now, under the *semi*-supervised scenario, audio can be used to also produce feature counts through some method. To map from audio to n-gram counts, the audio is decoded using the LVCSR system described in Section 4.2.2. The acoustic and language models are trained on the available in-domain data (from 2.5 to 40hrs depending on the condition). The decoder produces lattices (Section 2.1.3) with very unlikely paths pruned. These lattices are then collapsed down to the sufficient statistics for an n-gram language model: expected counts of word sequences up to length n. For a word sequence $w_1, \dots w_n$, the expected occurrence in an audio utterance X under the model P is

$$\mathbb{E}_P[w_1, \dots w_n | X] = \sum_H P(H|X)\ c(w_1, \dots w_n \in H) \tag{4.12}$$

where H is a complete utterance hypothesis and P is the posterior probability of the hypothesis provided by the LVCSR system. Summing over all hypotheses and computing $P(H|X)$ is efficiently done by the forward-backward algorithm over the lattice.

4.4.1 Information Theoretic Motivation for Expected Counts

Semi-supervised language modeling can be formulated as follows. N acoustic observations (utterances) are assumed labeled $x_1, x_2, \ldots, x_N$. The task is to estimate the parameters Θ of the model $P_\Theta(W)$, where $W \in \mathcal{V}^\star$ is a sequence over a vocabulary $\mathcal{V}$. The automatic classifier provides a posterior distribution over word sequences $P(W = w_1^n | x_i)$. For illustration, let $P_\Theta(w_1^n)$ be an un-smoothed bigram language model such that the probability of a word sequence w_1^n is given by

$$P_\Theta(w_1, w_2, \ldots, w_n) = \prod_{i=1}^{N} P_\theta(w_i | w_{i-1}). \tag{4.13}$$

The sufficient statistics for Θ are the conditional frequencies of seeing w_j follow w_i. Since there is no set of samples $w_1, w_2, \ldots, w_N$ for use as training data, likelihood cannot be maximized under the model $P_\Theta(w_1, w_2, \ldots, w_N)$. What is available are a set of posterior probabilities which reflect the uncertainty of the model conditioned on the acoustic data $x_1, \ldots x_N$. are forced to estimate one distribution, P_Θ, which best captures the uncertainty of the N different posterior probabilities $P(W|x_i)$.

One criterion for finding this average over distributions is Kullback Leibler divergence. The acoustic samples are treated as equally likely and minimize the sum KL divergence for all $P(W|x_i)$ with respect to $P_\Theta(w_1^n)$. Since KL divergence is convex with respect to $P_\Theta(w_1^n)$, there is one unique solution that minimizes this loss function. This distribution can be found subject to the constraint that the conditional probabilities sum to one using the Lagrangian method given by

$$L(\Theta) = \sum_{i=1}^{N} KL(P(W|x_i) || P_\Theta(w_1^n)) + \sum_{w_0 \in \mathcal{V}} \lambda(w_0) \sum_{w \in \mathcal{V}} P(w|w_0) \tag{4.14}$$

where λ enforces the sum to one constraint over the vocabulary for each word history. For each parameter θ_{w_0,w_1} in Θ, $L(\Theta)$ is differentiated with respect to θ_{w_0,w_1} as the partial derivative

$$\frac{\partial}{\partial\theta_{w_0,w_1}} L(\Theta) \tag{4.15}$$

$$= \frac{\partial}{\partial\theta_{w_0,w_1}} \left[\sum_{i=1}^{N} \sum_{w_1^n \in \mathcal{V}^\star} P(w_1^n|x_i) \log \frac{P(w_1^n|x_i)}{P_\Theta(w_1^n)} + \lambda \sum_{w\in\mathcal{V}} \theta_{w_0,w} \right] \tag{4.16}$$

$$= \frac{\partial}{\partial\theta_{w_0,w_1}} \left[\sum_{i=1}^{N} \sum_{w_1^n \in \mathcal{V}^\star} P(w_1^n|x_i) \log P(w_1^n|x_i) - P(w_1^n|x_i) \log P_\Theta(w_1^n) + \lambda \sum_{w\in\mathcal{V}} \theta_{w_0,w} \right] \tag{4.17}$$

$$= \frac{\partial}{\partial\theta_{w_0,w_1}} \left[\sum_{i=1}^{N} H(P(w_1^n|x_i)) - \sum_{w_1^n \in \mathcal{V}^\star} P(w_1^n|x_i) \log \prod_{j=1}^{n} \theta_{w_{j-1},w_j} + \lambda \sum_{w\in\mathcal{V}} \theta_{w_0,w} \right] \tag{4.18}$$

and since the entropy of the posterior distribution $H(P(w_1^n|x_i))$ is assumed to be independent of the parameterization of $P_\Theta(w_1^n)$, it can be ignored in Equation (4.18), so that

$$\frac{\partial}{\partial\theta_{w_0,w_1}} L(\Theta) = -\sum_{i=1}^{N} \frac{\partial}{\partial\theta_{w_0,w_1}} \left[\sum_{w_1^n \in \mathcal{V}^\star} P(w_1^n|x_i) \sum_{j=1}^{n} \log \theta_{w_{j-1},w_j} + \lambda \right] \tag{4.19}$$

$$-\sum_{i=1}^{N} \frac{\partial}{\partial\theta_{w_0,w_1}} \left[\sum_{w_1^n \in \mathcal{V}^\star} P(w_1^n|x_i) \sum_{w_k,w_l} \log\, \theta_{w_k,w_l}{}^{c(w_l,w_k)\in w_1^n} \right] + \lambda \tag{4.20}$$

$$-\sum_{i=1}^{N} \sum_{w_1^n \in \mathcal{V}^\star} \frac{P(w_1^n|x_i) \cdot c(w_0,w_1) \in w_1^n}{\theta_{w_0,w_1}} + \lambda = 0\,. \tag{4.21}$$

$$\tag{4.22}$$

This implies that

$$\theta_{w_0,w_1} = P(w_1|w_0) = \frac{\sum_{i=1}^{N} \sum_{w_1^n \in \mathcal{V}^\star} P(w_1^n|x_i) \cdot c(w_0, w_1 \in w_1^n)}{\lambda}\,. \tag{4.23}$$

Solving for λ using the Lagrangian constraint that $\sum_{w \in \mathcal{V}} \theta_{w_0,w} = 1$

$$\sum_{w \in \mathcal{V}} \frac{\sum_{i=1}^{N} \sum_{w_1^n \in \mathcal{V}^\star} P(w_1^n|x_i) \cdot c(w_0, w) \in w_1^n}{\lambda(w_0)} = 1, \tag{4.24}$$

$$\lambda(w_0) \sum_{i=1}^{N} \sum_{w_1^n \in \mathcal{V}^\star} P(w_1^n|x_i) \cdot \sum_{w \in \mathcal{V}} c(w_0, w) \in w_1^n \,. \tag{4.25}$$

which further simplifies by defining the expected count of the sub-sequence $w_0, \ldots w_k$ given $P(w_1^n|x_i)$ as

$$\mathbb{E}_{P(w_1^n|x_i)} c(w_0, \ldots, w_k) = \sum_{w_1^n \in \mathcal{V}^\star} P(w_1^n|x_i) \cdot c(w_0, \ldots, w_k \in w_1^n) \tag{4.26}$$

Our target expectation for the bigram feature is therfore the expected a posteriori count

$$P(w_1|w_0) = \frac{\sum_{i=1}^{N} \sum_{w_1^n \in \mathcal{V}^\star} P(w_1^n|x_i) \cdot c(w_0, w_1 \in w_1^n)}{\sum_{i=1}^{N} \sum_{w_1^n \in \mathcal{V}^\star} P(w_1^n|x_i) \sum_{w \in \mathcal{V}} c(w_0, w \in w_1^n)} \tag{4.27}$$

$$= \frac{\sum_{i=1}^{N} \mathbb{E}_{P(w_1^n|x_i)} c(w_0, w_1)}{\sum_{i=1}^{N} \mathbb{E}_{P(w_1^n|x_i)} c(w_0)} \,. \tag{4.28}$$

which intuitively is the expected count of the bigram divided by the expected count of the unigram. Expected counts provide the best approximation of the entire set of posteriors. The key assumption, however, is that onne wishes to fully capture the uncertainty of the posteriors. Empirical results in the following sections will show that this is not always the best course of action.

4.4.2 Alternate Count Estimates

Besides the entire expected count, one can limit count estimation to the one-best output from the recognizer. After all, if the recognizer was perfect, one would ignore the second-best. Of course, this is not true and so one can weight the one-best output by its posterior probability. Per-token posterior probabilities can be efficiently computed using a lattice or else by summing over a large n-best list and computing the weighted count of hypothesis posteriors $P(w|X) = \sum_H P(H)c(w \in h)$. Additionally, previous work [1] has successfully used a *confidence model* to improve semi-supervised acoustic modeling. A confidence model estimates the probability of a word token in the one-best output being correct, which is different than the *posterior* probability of a word as computed from the lattice itself. It improves over posterior estimation since orthogonal information can be included in the confidence estimation.

Our confidence model [136] is a generalized linear model which takes as input a variety of continuous features. The probability of correctness that a word token $\hat{w}$ equals the reference word w is defined as

$$P(\hat{w} = w) = \frac{\exp(\sum_{i=1}^{N} \lambda_i \cdot x_i)}{1 + \exp(\sum_{i=1}^{n} \lambda_i \cdot x_i)} \tag{4.29}$$

where there are 137 real valued features $f_1, \ldots f_n$ that include measurements of the instance $\hat{w}$ as well as lexical features of $\hat{w}$, each of which has an estimated weight λ_i. These include 37 features such as lattice posterior probabilities, duration, signal to noise ratio, the number of times the word appeared in acoustic training, the number of tri-phones that appeared in training and many others [136]. Additionally, the top 100 words appear as binary indicator features, allowing for a word-specific bias.

The parameters of the model, Λ, are estimated via maximum likelihood training on a held-out set. As expected, the lattice posterior probabilities are the most predictive feature, but other useful features include the frequency of the word in training, average tri-phone coverage and phonetic length. Other models, such as neural nets, and additional features have given modest only gains over this robust recipe and so this confidence model is used here without such further improvements.

Once all tokens w in the one-best hypothesis have estimated confidences $\chi(w)$, the *confidence-weighted* count of an n-gram w_1^n which occurred $\tilde{c}(w_1^n)$ in the corpus is

$$\hat{c}(w_1^n) = \sum_{i=1}^{\tilde{c}(w_1^n)} \prod_{j=1}^{n} \chi_i(w_j) \tag{4.30}$$

so that the confidence of each instance of w_1^n is simply the product of the individual word confidences.

This assumes that the probability of a word being correct is independent of its neighbors. Furthermore, it heavily discounts longer n-grams since the average n-gram token is proportional to $\bar{\chi}^n$. However, experiments with directly modeling n-gram confidences, with a separate GLM for each order, showed very modest improvements [85] in confidence accuracy or improved semi-supervised language modeling. Other combinations, such as geometric or arithmetic mean, max and min were also empirically out-performed by the simple multiplication of Equation (4.30). This method will be used for the simplicity of estimation and its competitive performance.

4.5 Limits of Non-Parametric Models

First, this chapter reaffirms previous work using standard non-parametric n-gram language models on one data condition. 190 hours of Fisher audio was decoded with an acoustic and language model trained on ten hours of high quality manual transcripts at WER of 41.8%. The extracted counts were used to build a separate LM which was then interpolated with the initial LM from the 10 hours (100k tokens) with the optimal interpolation weight determined on held-out data. Then the 3hr Hub5.Dev04 test set was decoded using a fixed 10 hour acoustic model and the new estimated language model. Since the 190 hours does have manual transcripts, the upper bound ws computed and used to judge success of semi-supervised estimation when compared against estimation from manual transcription.

4.5.1 Semi-Supervised Estimation

This section experiments with using the full expected counts, using unweighted one-best output or using confidence weighted one-best output in Table 4.1.

Method	**WER**	**WER Recovery**
10hr Baseline	41.8	-
10 + Expected Counts	42.5	-11%
10 + Unweighted one-best	41.7	1%
10 + Confidence Weighted one-best	41.3	7%
200hr supervised upper bound	35.0	-

Table 4.1: *WER Results for 10+190 Non-Parametric LM* - 190 hours of Fisher English was decoded with a 10hr AM and LM. Then the full expected counts, unweighted one best, posterior-weighted one best or the confidence weighted one best were extracted. The confidence model was trained on a three hour held-out set. The different counts were then used to build a language model which was interpolated with the initial 10 hour LM. Expected counts, while being theoretically optimal, provide no gain as they induce too many hallucinated n-grams. The optimal use is the confidence weighted counts, which demonstrates the orthogonal information capture by the confidence model. Still, these methods recover at best 7% of the possible gain for transcripts. The gain for confidence weighted one-best transcripts are statistically significant ($p < 0.001$), while the unweighted counts are not significant ($p > 0.1$)

The best fair result used confidence-weighted counts from the one-best output and improved performance by 0.5% WER - which is only 7% of the possible gain for manually transcribing the 190 hours. Disappointingly, using the expected counts performs significantly worse than any method based on the one-best. Ideally, one would want to capture the model's full belief in the space of word distributions represented by the complete lattice posteriors. However, the degradation indicates that the low-resource recognizer is a very poor model of English conversational speech.

One clue as to why is revealed by measuring the recall of the n-gram types. This is the percentage of unique n-gram types in the reference also seen in the automatic counts. 56.63% of the trigram types seen in the reference appear in the one-best. Adding millions of additional trigram types seen in the lattice only increases the recall to 57.75%. The true word sequences uttered by the speakers appear so far down the list of alternate hypotheses that they are pruned out. While increasing the lattice size would improve recall, incorrect word sequences would dominate.

The quality of the lattices are so poor for two reasons. First, the vast majority of words in the decoding dictionary are *over*-counted. A 75,000 vocabulary was used despite only having training samples for 5,000 word types. The choice of a large vocabulary in recognition ensures that new content words will be added to the language model. However, the vocabulary was not carefully pruned as it is a stand in for a typical low-resource vocabulary, possibly scraped from the web. This means that the remaining 70,000 word types appear with equal probability in the language model. Many of these words are misspellings, inaccurate transcriber marks and otherwise 'invalid' and should never occur in conversational speech. The huge space of words combine to create a massive space of n-grams, all which crowd out the rare valid n-grams. Because the language model is so sparse, it is unable to prefer "valid" n-grams from hallucinations.

Second, the posteriors produced by the recognizer were not optimized for estimating n-gram counts from a speech corpus. Instead, they were designed to minimize one-best error rate. One could directly attempt to improve the quality of the posteriors instead of using the standard LVCSR recipe.

With the availability of development data, one could optimize the posteriors to match the empirical frequencies. One could consider minimizing KL divergence of the learned and empirical distributions. The goal is no longer interested in transcription accuracy, but in *frequency* accuracy. It is desirable that the recognizer to predict unigram, bigram and higher order statistics at the same rate as some amount of truth. In contrast to previous work with confidence estimation, this is no longer a *token* decision on a per-sample basis, but instead a *type* estimate across an entire corpus. This raises difficulty of estimation as one can no longer make independence assumptions for each sample. The next section will explore the potential gain to be had for improved posterior estimates.

4.5.2 Closing The Gap

Given the lack-luster performance of semi-supervised language modeling, where should one invest effort to improve semi-supervised performance? The upper bound for improved posterior estimation is perfect token confidences. The *token oracle* was computed for n-grams by setting to zero any occurrence which contains an incorrect word in the one-best. Likewise, one can ignore confidence-weighted counts for correct n-gram tokens and set their counts to one. Note that this oracle does not include n-grams unseen in the one best output. If the trigram "A B C" was hypothesized but "A B D" actually occurred, then the count of “A B C” is decremented by one, but “A B D” is *not* incremented. Likewise, if “A B C” occurred in the reference elsewhere, but still missed in this particular instance, it is still not incremented by one. This oracle only gives the upper bound on perfect confidence estimation and the results are detailed in Table 4.2.

Model	**WER**	**WER Recovery**
10 hour baseline	41.8	-
10+190 semi-supervised	41.3	7%
10+190 semi-sup. w/token oracle	39.2	38%
200 supervised	35.0	-

Table 4.2: *Gains for Oracle Token Estimates* - All results decoded with 10hr AM. If perfect confidences were known, then all hallucinated n-grams would be removed. Likewise, correct n-gram tokens would be counted wholly. This gives a sizable gain over the fair result, recovering almost 40% of the potential gain.

The second form of oracle knowledge is of the n-gram *type.* Instead of knowing on a token by token basis, the oracle provides information at a coarser level of the frequency of that type in the corpus. It cannot be said where it occurred, but that it occurred somewhere in the corpus. The set of n-gram types were partitioned into three different bins.

- **Seen** - Those seen in the one best and the reference.
- **Unseen** - Those unseen in the one best, but occur in the reference.
- **Other** - Those that do not occur in the one best or the reference.

The huge swath of *Other* types are ignored since there is no estimate of the true count. For the *Seen* and *Unseen* categories, there are two forms of oracle knowledge:. The *Type* count is a binary decision which is true if the n-gram occurs one or more times in the corpus. For *Seen* n-grams, their counts are unmodified and *Unseen* n-grams have their counts set to zero. The *Token* count gives the correct count of both seen and unseen n-grams. Table 4.3 gives a few examples of these count oracles for different n-grams.

	Count		**Oracle Estimate**	
Category	**One Best**	**Reference**	**Type**	**Token**
Seen	2	4	2	4
Seen	2	0	0	0
Unseen	0	3	1	3
Other	0	0	0	0

Table 4.3: *Examples of Oracle Categories* - The space of n-grams are divided into three categories, Seen, Unseen and Other. *Seen* n-grams are seen in the hypothesis output one or more times. The *Type* oracle removes hallucinated n-grams but does not correct the counts of hits. The *Token* oracle corrects their counts. For *Unseen* n-grams, which occurred in the reference but were unseen in the one best, the *Type* oracle sets their counts to one and as expected, the *token* oracle sets their counts to the true amount. All other n-grams unseen in the one best or reference are left untouched.

Table 4.4 shows the improvements for the increasingly stronger oracle knowledge. First off, removing all hallucinated n-grams would give the largest improvement in performance. Then fixing the remaining counts of the seen n-grams would give over half of the potential gain. For the unseen n-gram types, inferring first that their count should be more than zero is as important as their true count. This is because the majority of unseen n-gram types in the one best are rare and have a reference count of one.

The oracle results from Tables 4.2 and 4.4 improved all n-gram types, unigrams, bigrams and trigrams alike. Acting on these oracle results would require improving all three

Language Model	**WER**	**Recovery**	**Proportion of Gain**
10 hour baseline	41.8		
10+200hr fair	41.3	7%	7%
+ Seen Type	39.3	37%	31%
+ Seen Token	38.2	53%	16%
+ Unseen Type	36.7	75%	22%
+ Unseen Token	35.0	100%	25%
200hr Supervised	35.0		

Table 4.4: *Gains for Oracle Type Estimates* - All results decoded with 10hr AM. Sorted by increasing strength of domain knowledge, the n-gram type oracles demonstrate the knowledge necessary to cover the gap from semi-supervised to fully supervised performance. Removing hallucinations (row 3) provides the largest relative gain and fixing the remaining observed n-gram counts would give half of the total possible gain. The remaining half lies in fixing the missed n-gram counts which occurred in the reference but not in the one best. These n-grams must be teased apart from the huge space of all possible n-grams unseen in the one best. *Note that col 3. does add up to 100% despite rounding indicating 101%.*

orders of n-grams. However, it may be much easier to improve the count estimate of a single word versus a trigram. Table 4.5 breaks down the gains by fixing either just the unigrams, unigrams and bigrams and finally fixing all three orders. Completely fixing unigram counts provides no gain and fixing bigram counts provides a very modest 13% recovery. This is because a back-off language model is not able to capitalize on these weaker statistics. Instead, the action is all in the highest order n-grams.

Since smoothing methods were designed under the assumption that observed n-grams were correct, the interpolation weight with lower order n-grams do not take the trustworthiness into account. Attempts at training a bigram language model resulted in an LM that was significantly worse than the semi-supervised trigram language model. If one is restricted to a back-off model, then further gains must come from improving trigram count estimates, which the next section attempts to do. Failing that, one must move to a different model which can take advantage of improved estimates of broader categories of events.

4.5.3 Attempts at Improving Count Estimates

The previous section demonstrated that improved confidence estimates could improve semi-supervised performance. This section explored one step towards this work by calibrating word confidences. While the parameters were estimated to maximize training likelihood, the confidence model (Section 4.4.2) still has a systematic bias. If the confidence model correctly predicted the probability of a word token being correct, then half of the word tokens with a confidence of 0.5 should be correct. However, the model over-predicts the confidence of words within that range. At low (0 to 0.2) and high (0.9 to 1) confidences, the model is fairly well calibrated. But within the most frequent confidence ranges of 0.2

	WER (WER Recovery)		
Language Model	**1-gram**	**2-gram**	**3-gram**
10 hour baseline	-	-	41.8
10+200hr fair	-	-	41.3
+ Seen Type	41.3	41.0	39.3
+ Seen Token	41.3	40.8	38.2
+ Unseen Type	41.3	40.6	36.7
+ Unseen Token	41.3 (0%)	40.5 (13%)	35.0
200hr Supervised	-	-	35.0

Table 4.5: *Oracle Type Estimates by Order* - All results decoded with 10hr AM. The same oracle experiments from Table 4.4 were repeated but separated by n-gram order. The oracle improvements increase in domain knowledge from the top left to the bottom right. Real improvements for a non-parametric LM require correct trigram counts.

to 0.9, the model is falsely confident.

To correct for this, the confidence model was calibrated using 100 hours of heldout data. All confidences were binned within 0.01 and then the final mapping was a linear interpolation between these points. The community evaluates improvements to confidence estimation using normalize cross entropy (NCE) defined as

$$\text{NCE} = \frac{H_{\text{base}} - H_{\text{cond}}}{H_{\text{base}}} \tag{4.31}$$

where

$$H_{\text{cond}} = \sum_{i=1}^{N} \log\left(c_i \cdot \delta(y_i = 1) + (1 - c_i) \cdot \delta(y_i = 0)\right), \tag{4.32}$$

and

$$H_{\text{base}} = -n \cdot \log(\frac{n}{N}) - (N - n)\log(1 - \frac{n}{N}) \tag{4.33}$$

where there are N word tokens along with their confidence $c_i \in [0,1]$ and their true word score $y_i \in 0, 1$ and n of the N tokens are correct. A higher NCE indicates better confidence quality. Calibrating the confidence scores improved NCE from 0.068 to 0.144 (state of the art confidences across a variety of speech tasks typically fall within 0.1 to 0.2). However, this doubling of NCE failed to meaningfully impact semi-supervised performance. Word error rate improved by 0.1% absolute with these improved confidences. This section extended this further to directly calibrating higher order n-gram counts, with a separate mapping for each order. This led to no noticeable gain over the calibrated word confidences. Calibration has little value for semi-supervised learning because it maps all tokens within a range to a similar value. What is really required is a better confidence model, which can learn to deweight hallucinations and improve the scores of hits.

The previous section demonstrated that over half of the potential gain could be recovered by correcting the counts of trigrams seen in the 1-best output. In particular, setting the counts of incorrect trigrams to zero is a big part of the solution. This task of count regression is to predict the number of occurrences in the reference transcript of an n-gram *type* given features observable from the ASR output. In addition to the observed count, a variety of features were used observable from either the ASR output or the initial training data. These features included averages of the confidence features from Section 4.4.2 as well as the average confidence, 37 in total.

As an initial test,the 400 hours of English Fisher audio was divided into two 200 hour train and test splits. After decoding both sets with a 10 hour LVCSR system, all n-gram types seen in the 200 hours of training were used to train an artificial neural network. Each type was a training instance where the target output was the normalized frequency of the n-gram in the 200 hours of reference and the inputs were 37 features described above. The ANN had one hidden layer, with 100 hidden units found to be optimal and was trained with back-propagation to minimize cross entropy.

Performance was evaluated by computing the mean squared error between the predicted frequency and true frequency on the held-out 200 hour set. As described in Table 4.5.3, the ANN effectively reduces root mean squared error from 9.46 to 3.08. However, these reductions in RMSE do not carry over to perplexity or WER. Using the modified counts increased perplexity from 237 to 241 and failed to improve WER. This mismatch between training criterion and end performance is due to the large imbalance of hallucinated n-grams, whose “target” count is zero.

Method	**Average**	**Hits**	**Hallucinations**
1-best	9.46	2.23	10.00
ANN Estimate	3.08	8.03	1.62
Set All To Zero	4.00	12.01	0.00

Table 4.6: *Breakdown of RMSE by True Count* - The ANN significantly reduces root mean squared error (RMSE) on average over the 1-best counts (col 2). However, the average is heavily biased towards n-grams whose true count is zero – hallucinations. This results in a model which tends towards zero, giving good performance on reducing hallucinations (col 3), but poor performance on the true hits (col 2). The ANN can at least outperform simply setting all n-gram counts to zero - giving perfect performance on the hallucinations (row 3).

Table 4.5.3 shows that nearly 75% of the trigrams seen in the 1-best output should have a true count of zero. Unigram hallucination rates are much lower and an ANN trained on just unigrams had much smaller variance between hits and hallucinations. In contrast, the trigram ANN had much higher variance on RMSE: 8.03 for hits versus 1.62 for hallucinations. This indicates that the trigram net learned to set the hallucinations near zero at a cost of accuracy for actual trigrams. Unfortunately, the improved unigram neural net did not benefit task performance. As described in Section 4.5.2, a back-off LM is dependent upon accurate trigram counts, so the better performing unigram neural net did not impact language model performance.

Nonetheless, the balanced model led to attempts at different methods of *unigram* regression. Knowing the count of word types would undoubtedly improve performance of higher order counts. Attempts at using various models (linear regression, negative binomial and Gaussian process models) and many variants of the target function (log, binned, frequency, raw counts) saw no real success for word regression. Table 4.5.3 shows the most

Order	Number Types	Percent Hallucinated
Unigram	34K	23%
Bigram	1M	62%
Trigram	3.5M	74%

Table 4.7: *Hallucination Rates of 1-best N-grams* - If an n-gram is seen in the 1-best, but not in the reference, then it is considered a hallucination. The large number of hallucinations for bigrams and trigrams means the majority of target values for n-grams is zero.

predictive features of the 37 used for a linear model on the predicted frequency. As expected, the one-best count is the most predictive. Notably, the estimated confidence is *not* predictive. Other features, such as the estimated probability of false alarm, whether a word was seen in the original ten hours, and others only reduced MSE from 0.4158 to 0.3876 - only a 6% reduction. This leads to the conclusion that current methods lack observable features necessary to estimate whether a particular word is over or under generated in a

Feature	**Weight**	**MSE**
1-best Count	-0.057	.4158
+ pFA	-0.132	.4062
+ In-Training	0.288	.3941
+ Pronunciation Length	-0.024	.3905
+ Num. Unique Speakers	0.004	.3876

Table 4.8: *Predictive Features for Word Regression* - Starting with the 1-best count from the automatic transcripts, successively useful features are listed (col 1) along with their weight (col 2) and cumulative reduction in mean squared error (col. 3) after use in logistic regression. Of the 37 features used, only these five were useful and in total, reduced MSE by only 6% relative from 0.4158 to 0.3876. This led to negligible accuracy in predicting word frequency.

large corpus of unlabeled speech. While singletons as a *class* are over generated, or long words tend to be right, the within class variance is so great that the model cannot predict for a specific word what its true count should be. And unfortunately, such knowledge of groups of words cannot be encoded into a back-off language model. The rest of this chapter will therefore focus on a log linear model which *can* capture such knowledge.

4.6 Impact of Improved Acoustic Modeling

Given the modest results of the previous section (and most language modeling research), it is natural to worry that improvements in other aspects of the recognizer might wipe out any of these gains. In particular, improvements in the acoustic model historically dominate those of the language model. As described in Chapter 2, the model uses Gaussian mixtures estimated with maximum likelihood for the acoustic models. While this is a reasonably competitive system, two improvements to the acoustic model could potentially deflate any gains in this chapter. This section explores the impact of improved acoustic modeling on the initial semi-supervised language modeling results.

If one suffers from limited resources for the language model, then one undoubtedly also lacks acoustic training samples. For if one had a strong in-domain acoustic model, then the training text would be well matched for use in language modeling. Thus in this limited resource domain, it is natural to consider acoustic model self-training. As more fully described in Section 2.4.1, semi-supervised estimating for automatic speech recognition originally began with the acoustic model. Unlike the results of this chapter, as well as prior language modeling work, semi-supervised acoustic modeling is quite successful. Across a variety of domains and resource conditions, training an acoustic model from automatic transcripts provides a consistent gain.

In contrast to the language model, the acoustic model may be better suited for semi-supervised learning for three reasons. First, acoustic models classify over a continuous space: real-valued feature vectors of speech cepstra. There is a notion of distance in this space and so any mislabeling is not a 0/1 loss, but along a spectrum Second, there is an extensive amount of domain knowledge encoded in state of the art acoustic models. Knowledge of allophones (through state clustering), phonetic smoothing (tri-phones) and more all constrain the space of possible models. A back-off language model has little such domain knowledge. At its most basic, it simply memorizes relative frequencies of words conditioned on the previous history. Finally, parameter estimation is more robust for acoustic modeling. From the very foundation of the Baum-Welch algorithm, acoustic models have relied on semi-supervised methods since state-level alignments are almost never available. Acoustic modeling research has taken label noise into account, resulting in multi-pass alignments during training and data-driven state clusters. While the underlying transcripts may be inaccurate, the acoustic observations used during training actually occurred.

The impact of semi-supervised acoustic modeling was evaluated on the 10 to 200 hour condition reported earlier. As in the LM work, first a ten hour system was built and used to decode 190 hours of audio at an average WER of 41.8%. Instead of weighting transcripts by word confidences, the utterances were ranked by their estimated hypothesis confidence using a similar model to the word confidence system [1]. The top half was selected (which prior research suggested as a successful strategy) and re-trained the acoustic model by pooling the ten and one hundred hours. This resulted in a 1.6% absolute reduction in WER for improving the acoustic model alone. Estimating a semi-supervised back-off language model (described in Section 4.5), achieves a further 0.4% reduction in WER. Table 4.9 fully details the results.

Next, the impact of semi-supervised acoustic modeling on language modeling can be contrasted. If the acoustic model is trained on only ten hours of in-domain data and kept fixed through the semi-supervised language modeling, the system achieves a 7.4% WER Recovery. Then, if the acoustic model is first improved through semi-supervised estimation, the results hold with a 7% WER Recovery. While the absolute performance reduces slightly, the two models are learning mostly complementary information.

To capture the effect of a much improved acoustic model, an AM was built on a separate 200 hours of manually transcribed audio. This represents a dramatic improvement in acoustic modeling with WER decreasing from 41.8% to 33.0% for improving the AM alone. The dramatic *improvement* in semi-supervised language modeling is clear. The 9% reduction in WER leads to higher quality transcripts and results in a stronger language model. The semi-supervised LM reduces WER by 1.5% (compared to the 0.5% for the 10hr condition) and achieves 25% WER Recovery.

Acoustic models may also be improved not through better parameter estimation, but better model design. One recent improvement in the last decade is discriminative acoustic modeling [14]. Instead of maximizing the likelihood of acoustic observations, the models were estimated with the *Minimum Phone Error* criterion [137] F_{MPE}, which given

AM / LM	**10hr**	**10+190**	**200hr**	**LM Recovery**
10hr	41.8	41.3	35.0	7.4%
10+190hr	39.2	38.8	33.5	7.0%
200hr (separate)	33.0	31.5	27.0	25%

Table 4.9: *Impact of Acoustic Model Self-Training* - Semi-supervised language modeling is complementary to semi-supervised acoustic modeling. Starting with ten hours of in-domain transcripts, 190 hours of audio was decoded, producing automatic transcripts at 42% WER. A semi-supervised AM and LM was estimated, resulting in an improvement of 3% WER. Holding the acoustic model fixed to only the ten hours (row 1) reduces WER by 0.5% and achieves 7.4% WER Recovery. If instead a semi-supervised AM is estimated (row 2), the semi-supervised LM improves by 0.4% and 7% Recovery. To evaluate some much improved acoustic model, a separate AM was trained on 200 separate hours of Fisher transcripts (row 3). Because the transcripts are much improved (42% WER vs. 33%), semi-supervised language model estimation improves as well, achieving 25% Recovery. All three semi-supervised language models (col. 3) generate statistically significant ($p < 0.001$) transcripts than the baseline ten hour LM.

N acoustic samples $X_1, \ldots X_N$ and the matching reference transcripts $W_1, \ldots W_N$,

$$F_{\text{MPE}} = \sum_{i=1}^{N} \sum_{\hat{W}} P(\hat{W}|X_i) A(\hat{W}, W_i) \tag{4.34}$$

where the probability of sentence W being correct is

$$P(W|X_i) = \frac{P(X_i|W)^\gamma P(W)^\gamma}{\sum_{W'} P(X_i|W')P(W')} \tag{4.35}$$

and $A(\hat{W}, W_i)$ is the *phone* accuracy of the proposed word sequence $\hat{W}$ and reference word sequence W. This criterion maximizes the expected phone accuracy and consistently leads to stronger acoustic modeling. The reason for this gain is that discriminative training takes into account the language model score (through the posterior) and not just the acoustic likelihood.

To measure the impact of discriminative AM training on semi-supervised language modeling, the amount of initial labeled transcripts ranged from 2.5 to 40 hours. These transcripts were used to build a baseline acoustic model (with maximum likelihood parameter estimates) and Kneser-Ney language model. Four hundred hours of Fisher audio was decoded and extracted confidence-weighted one-best counts. These counts were then used to build a language model in conjunction with the original transcripts.

The baseline language model and semi-supervised language model were paired with either the ML or MPE acoustic model. Table 4.10 details the impact of discriminative acoustic modeling on semi-supervised language modeling. Using discriminative models on such small amounts of data does not result in dramatic reductions in WER. However, it does remove some of the absolute gain of semi-supervised acoustic modeling.

4.7 Using Expected Counts as Priors

In addition to incorporating weaker constraints (see Section 4.8.1), log linear language models provide a Bayesian framework for using counts from semi-supervised training data. As discussed in Section 2.3.2, log linear models typically use a Gaussian prior over the parameters centered at zero. This prior penalizes parameter weights which become too large - or equivalently move the model too far away from the uniform distribution. This section will explore using expected counts as a prior for a small amount of in-domain transcripts.

Under the semi-supervised learning regime, there may be a small amount of in-domain transcripts that is trusted to use in conjunction with a large amount of noisy transcripts. Both sets of data are untrustworthy for different reasons: the first is accurate, but under-sampled while the second is large, but inaccurate. However, one trusts that n-

AM	**LM**	**2.5hr**	**5hr**	**10hr**	**20hr**	**40hr**
ML	Init	55.9	46.2	40.3	38.0	33.6
ML	Semi-sup	53.7	44.4	39.0	36.9	32.5
WER Improvement		**2.2**	**1.8**	**1.3**	**1.1**	**1.1**
MPE	Init	54.0	44.3	38.9	36.9	32.0
MPE	Semi-sup	52.7	42.8	37.4	36.2	31.7
WER Improvement		**1.3**	**1.5**	**1.5**	**0.7**$^!$	**0.3**$^!$

Table 4.10: *Impact of Discriminative Acoustic Modeling* - WER on heldout test set. Two different acoustic models (ML v. MPE) were compared on varying amounts of in-domain data (2.5 to 40 hours). Discriminative training reduces WER by 1%-2% absolute (compare rows 1 and 3). Then a semi-supervised language model was estimated on 400 hours of automatic transcripts decoded with an ML AM and LM built on in-domain transcripts (separate for each column). Finally, the table reports the difference in WER between the initial LM and the semi-supervised LM when decoded with a MLE AM (row 5) vs. a MPE AM (row 6). Discriminative acoustic modeling does reduce the impact of semi-supervised language modeling, removing 0.5% absolute. However, especially at lower resources, semi-supervised language modeling provides complementary information to the discriminative language model. Note that all differences in rows 5 and 6 are statistically significant ($p < 0.001$) except for the gains with the discriminative model for 20 and 40 hours, denoted with !, which are not statistically significant ($p > 0.1$).

grams in the small amount of in-domain transcripts *actually occurred* - regardless of anything else, the model should match the constraints from that data. The goal is not to match them exactly and in the absence of other domain knowledge, the principle of maximum entropy argues that one should fall back on a uniform distribution - a Gaussian centered at zero. However, assuming one don't have a malevolent LVCSR system, the expected counts provide a better estimate of the parameters than the uniform. Whether one should adapt *from* the expected counts or *to* them is an empirical question that depends on their quality compared to the available in-domain data.

First, this section confirms that log linear models provide competitive performance with state of the art smoothing methods. Table 4.11 demonstrates that under the *supervised* training condition on 10 and 200 hours of manually transcribed data, a log-linear model with Gaussian smoothing performs almost identically as modified Kneser-Ney smoothing. Semi-supervised performance is improved, increasing WER Recovery from 7% to 16% using MAP adaptation. This improvement is statistically significant, indicating that a log-linear language model better exploits automatic n-gram counts.

Model / Training Data	**10hr**	**10+190hr**	**200hr**
Kneser-Ney Smoothing	41.8	41.3	35.0
Log-Linear w/Gaussian Prior	41.7	40.6	35.1**

Table 4.11: *Comparison of Log-Linear to Non-Parametric LM* - WER computed on held-out test set. A log-linear model with tuned Gaussian prior offers identical performance with state of the art smoothing for the fully supervised scenario (10hr and 200hr of manual transcripts). For semi-supervised estimation, MAP adaptation from expected counts to truth offers slightly better performance than standard smoothing. The two models generate statistically different ($p < 0.001$) for both the ten hour and semi-supervised condition and ($p < 0.01$) for the 200hr supervised condition.

The following two sections evaluate the effectiveness of using priors along two dimensions. First, the amount of in-domain English Fisher transcripts ranged from 2.5 to 40 hours (34K to 515K tokens). Second, the semi-supervised expected counts are placed between two extremes of background models. Four million words of Broadcast News transcripts was used to represent a lower bound for many conversational corpora. Tokens were evenly sampled from the HUB-4 corpus (LDC97T22) which consists of a range of news programs from ABC, CNN and NPR collected in 1996. No special effort was made to ensure a low OOV rate on English Fisher or to condition data collection based on the available transcripts. The text was normalized by removing all punctuation, standardizing abbreviations and converting all words to upper case.

The second reflects an upper bound on out of domain resource - targeted web transcripts [107]. Four million tokens of web data were selected to be conversational like using the entirety of the manual transcripts of Fisher. The resulting corpus contains web chats, television show transcripts and more. Section 4.7.1 will compare the value of these three background corpora in isolation while Section 4.7.2 will extend MAP adaptation to jointly using multiple background priors.

4.7.1 MAP Adaptation to the Fisher Corpus

The three background corpora are treated as static priors and make a piece-wise comparison between them. To map from data to parameter priors on Θ, first train a log-linear model on the background corpus. This log-linear model contains all unigrams, bigrams and trigrams seen in the 4M tokens of training data (350K unique features) be it broadcast news (BN), web text (Web) or automatic transcripts from the unlabeled audio.

A Gaussian prior is placed over these learned parameters centered at zero with tied variance. The variance is tuned on a held-out set of target *Fisher* data to obtain the models $\Theta_{\text{BN}}, \Theta_{\text{Web}}, \Theta_{\text{unsup}}$.

In the second step, the parameter vectors of this learned model now serves as the mean of a Gaussian prior over the target Fisher model Θ_{Fisher} . The background data is thrown away as it is incorporated through the prior weights $\Theta_{\text{BN}}, \Theta_{\text{Web}}, \Theta_{\text{unsup}}$, respectively. Instead of adapting from the uniform prior ($\Theta = 0$), now train a log-linear model on the target Fisher data adapting from these weights. Both models, the prior mean and the Fisher language model, constrain features that are seen in the in-domain data (125K features for 40hrs). However, the unconstrained features (n-grams) which appear in the background data but do *not* appear in the target data will still fire in the adapted model. In the case where a feature appears in both corpora, MAP estimation will ensure that it's feature weight is closer to the target in-domain data. Controlling this is again the variance, which is estimated on held-out data. A very small variance will ensure that the model essentially stays at the prior and as it increases to infinity, the parameters will converge at the maximum likelihood estimates on the Fisher data. The updated constraints with MAP adaptation are

$$\mathbb{E}_{P_\Theta}[f_j] = \mathbb{E}_{\tilde{P}}[f_j] - \frac{(\theta_j - \theta_i^0)^2}{\sigma_j^2} \tag{4.36}$$

where instead of a penalty equal to the parameter size, it is now the penalty of the difference from the prior Θ^0. This formulation easily incorporates the maximum entropy prior with $\Theta^0 = 0$. Figure 4.1 visually represents the empirical results below. The amounts of in-domain data were varied ranging from 2.5 to 40 hours of transcripts. Three different log linear models were estimated for each background corpus: Broadcast News, Web, and the

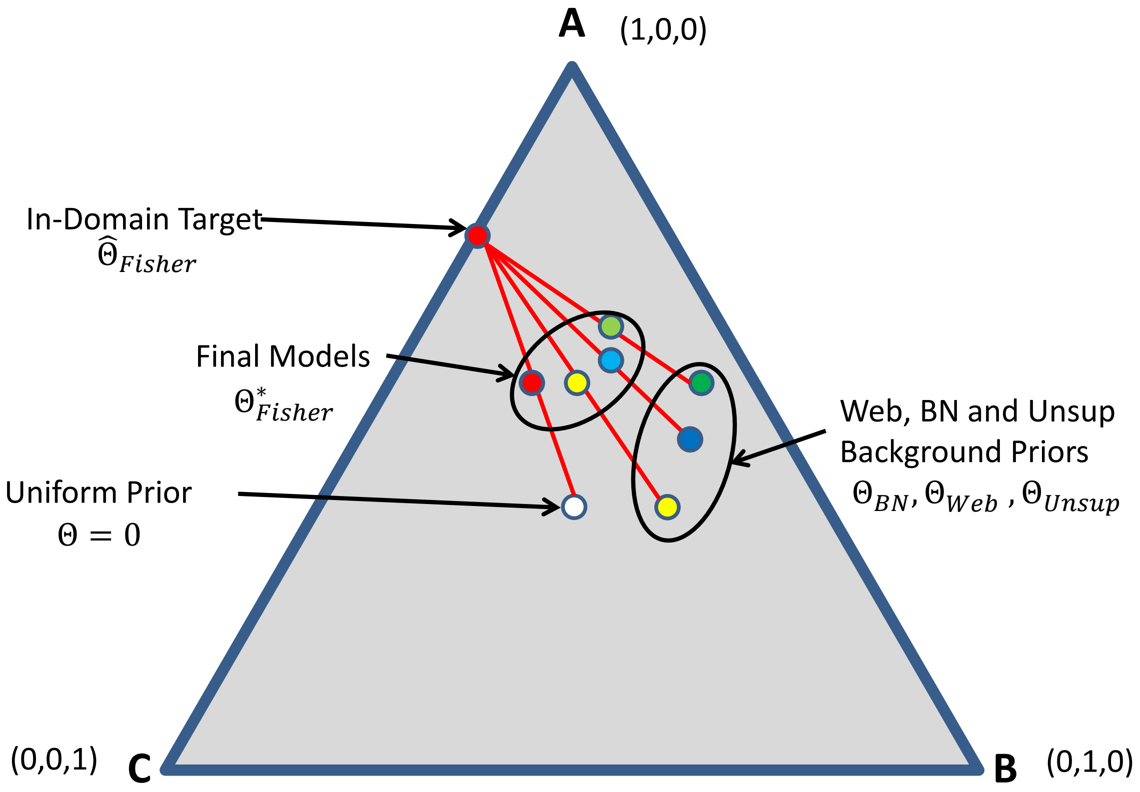
A (1,0,0)
In-Domain Target $\widehat{\Theta}_{Fisher}$
Final Models Θ^*_{Fisher}
Web, BN and Unsup Background Priors $\Theta_{BN}, \Theta_{Web}, \Theta_{Unsup}$
Uniform Prior $\Theta = 0$
(0,0,1) C
B (0,1,0)

Figure 4.1: *Illustration of MAP Adaptation* - The *probability simplex* in three dimensions is a useful tool to illustrate MAP adaptation of a log linear model. The background language models of Figure 4.2 are represented by three fixed background corpora (yellow, green, blue dots). Three models are first separately estimated on these corpora (not shown) using the uniform prior (white dot). Then each background model is adapted to the same in-domain corpus (red dot). The adapted final models were mapped to an ARPA format for use in decoding of a held-out test set.

expected counts from audio. This gives four different starting points for adaptation: the uniform prior and the three background models.

Note that the quality of the expected counts improves with the size of in-domain transcription. To generate them, a language and acoustic model was first built on the in-domain data. The WER of these models improved from 55% to 35% as they go from 2.5 to 40 hours. Then, the confidence weighted expected counts were extracted (Section 4.4.2). So although the audio corpus is fixed across the experimental condition, the extracted expected counts are different. For each different amount of in-domain data (2.5 to 40 hours), the optimal variance was estimated on held-out data and map the resulting optimal log-linear model to ARPA format. Instead of just reporting perplexity,these language models were then used to decode the evaluation corpus. Each LM was paired with an acoustic model trained on the in-domain data alone. Semi-supervised acoustic modeling was not run and the vocabulary was fixed through all experiments. Section 4.6 shows the minimal impact of acoustic model self-training on semi-supervised language modeling. Table 4.12 and Figure 4.2 details the results.

There are a number of conclusions to draw. First, the broadcast news language model Θ_{BN} is a terrible model of conversational speech. A language model built on 4M tokens has 8% higher WER (absolute) than just 35k tokens of in-domain transcripts. Nonetheless, it provides orthogonal information to the in-domain transcripts. MAP adaptation from Θ_{BN} to the in-domain data Θ provides a 1% to 2% absolute gain over using just the in-domain data alone.

At the other extreme, 4M tokens of conversational web text is an excellent model

LM / AM	**2.5hr**	**5hr**	**10hr**	**20hr**	**40hr**
Broadcast News	64.3	56.5	51.9	50.9	47.0
Automatic Trans.	55.9	46.4	40.9	38.9	34.5
Web Text	48.6	40.9	36.7	36.1	32.4
Target	55.9	46.2	40.3	38.0	33.6
BN→Target	53.8	44.6	39.7^{*}	$37.7^{!}$	33.2
Auto→Target	53.7	44.4	39.0	36.9	32.5
Web→Target	$48.4^{!}$	40.2	$36.5^{!}$	35.7^{*}	31.5

Table 4.12: *Held-out WER with MAP Adaptation* - Raw numbers from Figure 4.2. All adapted results (rows 5-7) are significantly different ($p < 0.001$) from both the background (rows 1-3) and target (row 4) models. When denoted with $*$ ($p < 0.01$) or ! ($p > 0.1$), results are not significantly different.

of Fisher. It is over 3% better (absolute) than 40 hours (500k tokens) of in-domain transcripts. But adapting *from* this better model *to* the worse in-domain models still provides a consistent benefit. The tuned variance allows the models not to wander too far from an accurate prior. Under this standard domain adaptation setting, MAP estimation provides a robust procedure which gives the best of both worlds.

Lying in between these two corpora are the semi-supervised results. Note again that the quality of the expected transcripts improves with more in-domain transcripts. The higher quality results in a better estimated semi-supervised model. Unlike the broadcast news corpus, which tapers off in value, the expected counts provide a consistent 2% reduction in WER, even as the amount of labeled data grows from 2.5 up to 40 hours.

MAP adaptation is fundamentally weighting the two corpora. Pooling simply

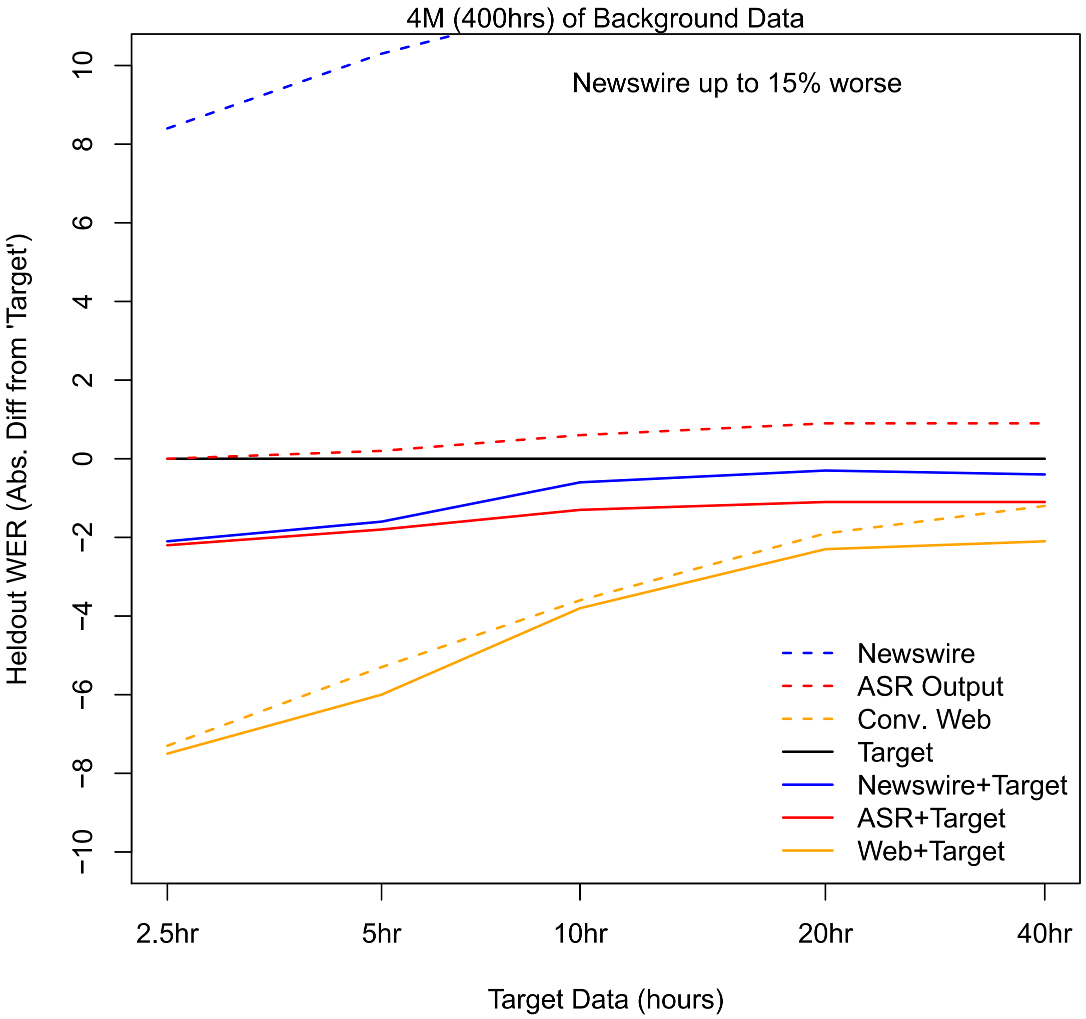

Figure 4.2: *WER Improvements with MAP Adaptation* - This section reports the **absolute difference** in WER from building a language model using just the in-domain target data. Each decode by column used an acoustic model trained on just the target data, so the WER improves along the x-axis. Decoding with just the background models (dashed lines) shows a range of performance from terrible (broadcast news) to very good (conversational web data). MAP adaptation from the background to the target data (straight line) always improves performance over either model (solid lines). Adapting from expected counts gives a modest 2% WER reduction on average over just the target data. And unlike the newswire corpus, the gains do not taper off as the quality of the expected counts improve with more target data.

merges the counts together, so that an n-gram seen once in both corpora is then seen twice in total. Instead, the two corpora can be *weighted* and then merged. The counts in the weighted corpus are scaled by a constant factor. For instance, say the bigram "A B" is seen in the first corpus once and "A C" is seen three times in the second. Then the first corpus count (of one) is multiplied by a factor, say 9, so that "A B" is now 3 times as likely as "A C" versus the other way around. Then the counts are merged together and normalize, so that the marginal count of A is $9 + 3$ instead of originally $1 + 3$.

Similar to the variance parameter of the Gaussian prior in MAP adaptation, the optimal corpus weight can be swept. Starting with a high weight on the background corpus (and performance identical to training on the expected counts alone) the target weight increases until the resulting model is trained on the limited amount of target data alone. After weighting, a log linear model is estimated on the new set of merged counts. Performance was almost identical to MAP estimation. Furthermore, Figure 4.3 highlights that across the varying qualities of expected counts and in-domain transcripts, the optimal weighting of the two corpora was almost one to one.

One caveat of these log-linear methods is that the optimal model requires tuning a variance parameter. Luckily the parameter requires a small amount of held-out data to tune and is not overly sensitive to the training data. The variance is tied across all parameters, so there is only one free hyper-parameter to estimate. To do so,variances were sampled uniformly across a range which was set empirically after some experimentation. The optimal variance typically fell within 100^2 to 1000^2.

To test the sensitivity of these estimates, a small test on bigram language models

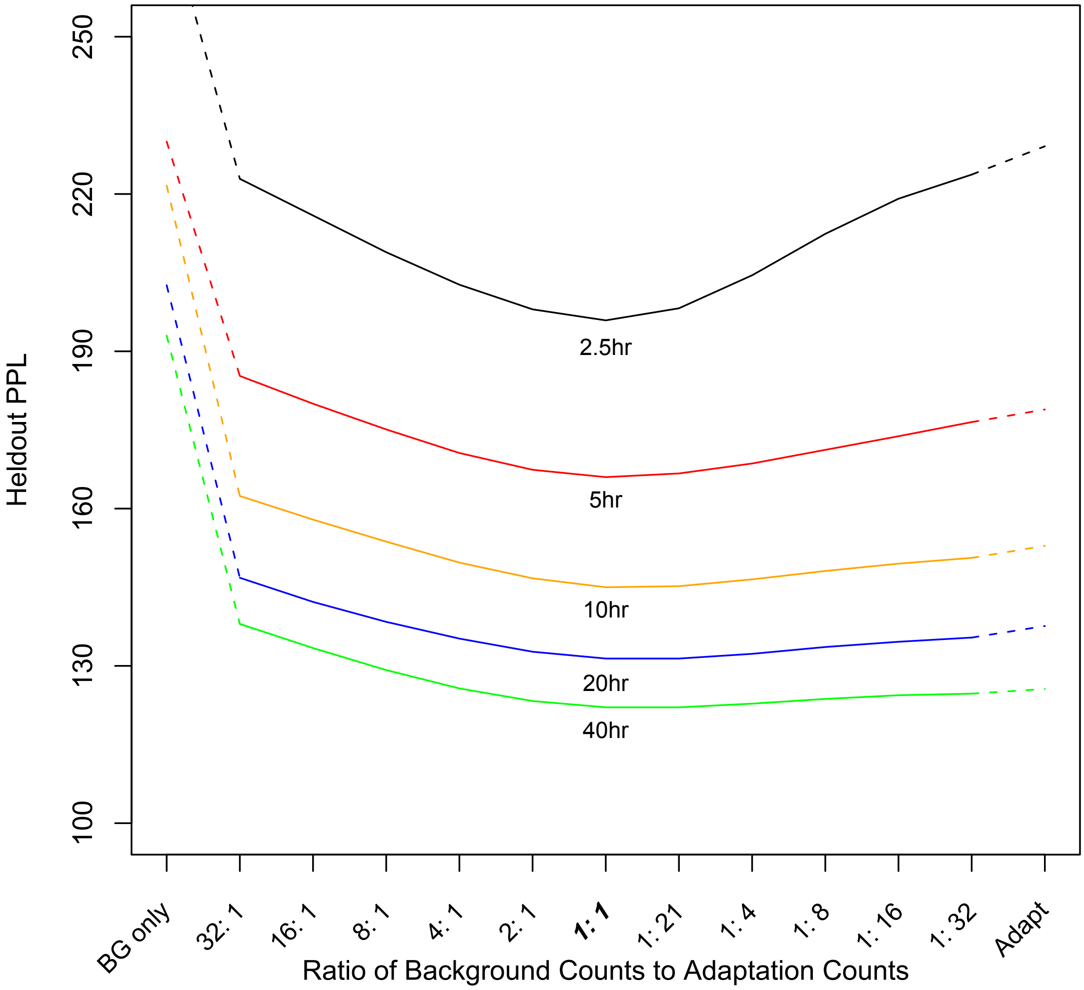

Figure 4.3: *Connection Between Count Weighting and MAP Adaptation* - 400 hours of expected counts (decoded with the respective target AM/LMs) were weighted with the target transcripts. These pooled counts were then used to build a language model and calculate held-out perplexity. As the ratio of background to target weights increases (x axis), the foreground corpus increases in importance. Surprisingly, the optimal weight for each corpus (despite vastly different expected counts quality and target corpus size) was roughly one-to-one.

was conducted. Three different sets of in-domain transcripts were used: 2.5, 10 and 40 hours of manually transcribed Fisher. Three different background priors were selected: the uniform, conversational web data and automatic transcripts. For each of the 9 combinations in the cross product of prior and target distribution, a range of variances was swept from 1 to $10,000^2$. Figure 4.4 details the results. When operating under the typical regime of a uniform prior, one finds that a tight variance enforces the model to randomly guess, resulting in a perplexity equal to the vocabulary size. As the variance increases, the optimal estimate increases along with the amount of training data. Finally, as it continues on towards infinity, the three models converge at the maximum likelihood estimates.

When the three models start from the same, but now non-uniform prior, a tight variance ensures they have the same perplexity as the background model. Again, more training data results in higher optimal variances. However, as the variance increases, the perplexity does not converge to that of using just the target data alone. Instead, it reaches the perplexity estimate for *pooling* the data together. A large variance no longer prefers the target data, but instead equally weights the two sets. Regardless of the method, perplexity is convex w.r.t. variance, fairly smooth and can be found through a simple grid search. This section demonstrated that expected counts can provide a substantial gain in performance and compares competitively with other sources of background data. The next section will explore the optimal use of multiple background corpora at once.

4.7.2 Hierarchical Bayesian Adaptation

The model need not be limited to the choice of one prior. While the previous section used broadcast news and web corpora to place semi-supervised estimation in context,

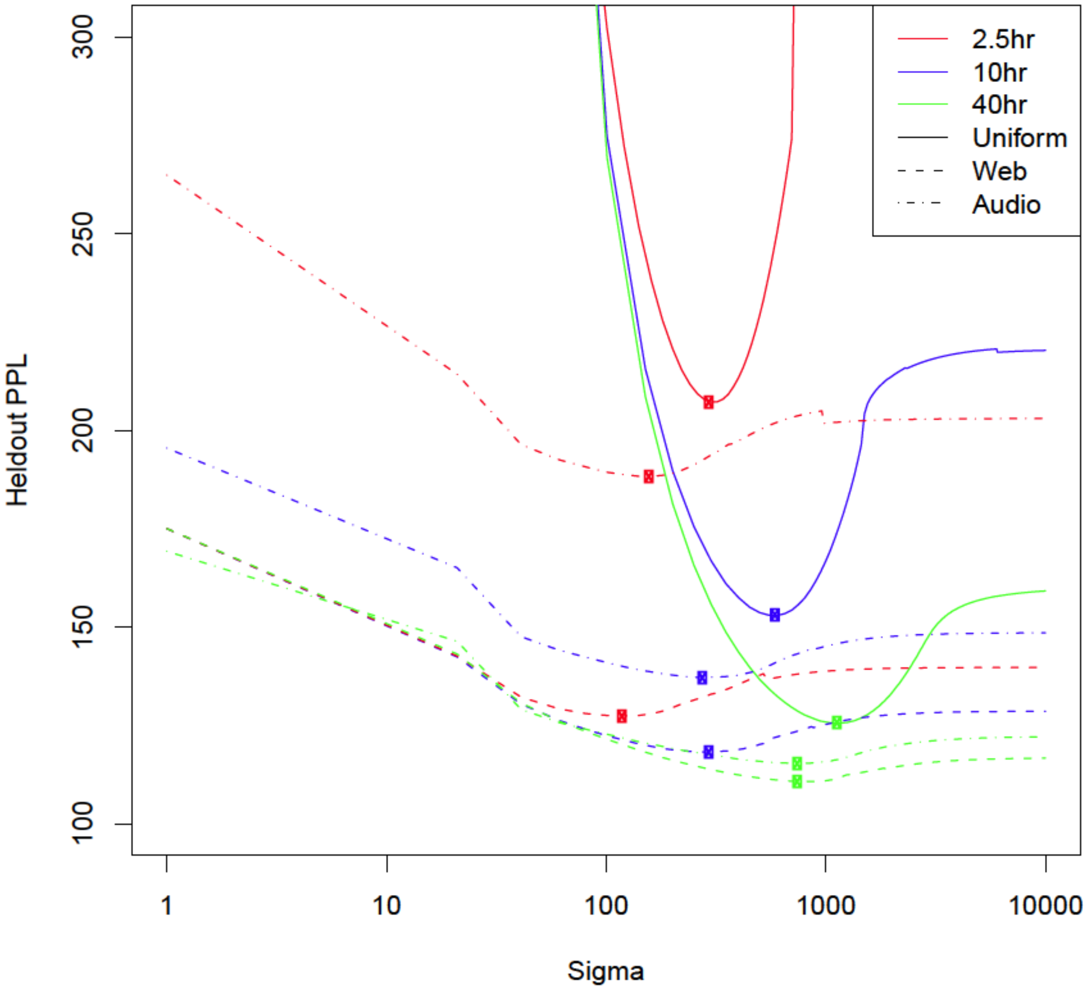

Figure 4.4: *Impact of variance on MAP adaptation* - Three different target corpora (2.5, 10 and 40 hours) were combined with three different background models (uniform, conversational web and automatic counts from audio). For the nine different models in the cross-product, the variance was swept from 1 to 10000^2 an then heldout perplexity was computed. The optimal variance for each model is circled. When adapting from the uniform distribution (solid lines), increasing amounts of training data result in larger variances. Adapting from a better prior, such as the web data (dashed and dotted lines), results in the same starting point, but ends at a different optimal condition. Finally, the automatic counts from audio are three different background priors, but the conclusion still holds.

one would want to use all three in an operational setting to build the best language model possible. When multiple diverse corpora arise, the standard recipe in language modeling is to linearly interpolate them instead of pooling them into one big corpus. With this scenario, there are now multiple *background* corpora with differing benefits for the target domain.

Under the log-linear adaptation framework, there is now a *multi-modal* prior over the parameters. Instead of one Gaussian distribution centered over the estimated parameters of a background corpus, there is a different mean for each background corpus. How then should one best utilize these different corpora to improve the target distribution (English Fisher in this chapter)? These disparate corpora could be pooled, used to train a separate language models and interpolate, or as explored in this section, extended through MAP adaptation to *Hierarchical Bayesian Adaptation* [46].

A statistical model is deemed "hierarchical" when the priors placed over the model parameters are themselves random variables and allowed to change as a function of the data. In the previous sections, the mean of the prior over the log linear parameters was fixed - either at zero or some other mean. Whether training the initial background model (with a zero prior) or the target model (with a background prior), estimation fixed the prior and only allowed the target parameters to update. Incorporating background data was thus a two step process - first estimate the background model and then estimate the target model. Hierarchical Bayesian adaptation instead does this estimation in one step by training on both corpora (background and target) at once with a *shared* prior.

When only the learned parameters Θ are free parameters, the objective function becomes

$$\arg\max_{\Theta} \underbrace{P(X|\Theta)}_{\text{lklhd}} \underbrace{P(\Theta|\mu = \theta_0, \sigma^2 = s)}_{\text{fixed prior}} \tag{4.37}$$

where each θ_i has an associated Gaussian prior with a mean θ_0 (little caps to denote a point estimate) and variance s that is fixed during estimation. Under the hierarchical model, the prior Θ_0 is now a free parameter along with Θ and the objective becomes It has its own Gaussian prior (labeled a meta-prior) which is fixed at zero, where

$$\arg\max_{\Theta,\Theta^0} \underbrace{P(X|\Theta)}_{\text{lklhd}} \underbrace{P(\Theta|\mu = \Theta_0, \sigma^2 = s)}_{\text{learned prior}} \underbrace{P(\Theta_0|\mu_0 = \vec{0}, \sigma_0^2 = s_0)}_{\text{fixed meta-prior}} \tag{4.38}$$

The likelihood of the data X still only depends on Θ, as it is conditionally independent of Θ_0. The domain expert does not specify a mean and a variance for the data, but instead two variances: σ_d^2 for the domain and σ_0^2 for the higher level prior. While in principle these parameters could be free parameters, estimation is no longer tractable requiring the use of approximate estimation likes Gibb's sampling. Instead these variance are tied across parameters and are optimized using a grid search method for held-out perplexity, as in previous sections. Introducing the global prior is unnecessary for the case of one Θ and one corpus X. However, when a second set of observations and a separate set of learned parameters is introduced, the power of this model emerges as the objective now becomes

$$\arg\max_{\Theta_1,\Theta_2,\Theta^0} \underbrace{P(X_1|\Theta_1)}_{\text{target lklhd}} \underbrace{P(\Theta_1|\mu = \Theta_0, \sigma^2 = s_1)}_{\text{shared prior}} \underbrace{P(X_2|\Theta_2)}_{\text{bkgd lklhd}} \underbrace{P(\Theta_2|\mu = \Theta_0, \sigma^2 = s_2)}_{\text{shared prior}} \underbrace{P(\Theta_0|\mu_0, \sigma_0^2)}_{\text{fixed meta-prior}} \tag{4.39}$$

when one estimates two joint models on both sets of target and background corpora. The empirical Bayes estimate was used to find the MAP estimate of each in a round-robin fashion. The updates for each parameter of the global prior $\theta_{0,i}$ are

$$\sum_d \frac{\theta_{0,i} - \theta_{d,i}}{\sigma_d^2} = \frac{\theta_{0,i}}{\sigma_0^2} \tag{4.40}$$

which is a penalized weighted average of the two corpus-dependent parameter weights. Once this parameter has been updated, the corpus specific parameters were iteratively updated until convergence. The global priors were then updated these steps again were iterated again until convergence. The set of features for all models (the corpus specific and global prior) is the union of all constrained features in the corpora. Although the target domain may have an order of magnitude less constrained features, it will have all the background features firing in all the corpora. The corpus specific parameters share global prior mean θ_i^* but have separate fixed variances. These variances allow for interesting interaction between the corpora. If both are set arbitrarily close to zero, then both corpora are forced to stay near the shared prior - in essence count pooling. If both range to infinity, then both models ignore the shared prior and converge on the maximum likelihood estimate of the individual corpora. The art is in balancing these two variances to improve end task performance.

A large variance has two effects. First, in Equation (4.39), it decreases the penalty for moving away from the global prior. Second, in Equation (4.40), a large variance decreases the impact of the corpus specific parameter on the global weight. As the corpus-specific variance increases, its model becomes more ‘independent’ of the other corpora. This method was compared with the two-step MAP adaptation using the web, semi-supervised expected counts and target corpora of the previous section. Using multiple corpora resulted in improvements in perplexity. However, the reduction in WER from the resulting model is not statistically significant. Additionally, since the broadcast news corpus was so poor,it was not included it in the remaining experiments.

Figure 4.5 gives a visual overview of these experiments using the probability simplex. Instead of training one background model on the out of domain corpus, fixing it and adapting, one joint model will be estimated using all three corpora (red, blue and yellow dots). The three models will all have the same set of features (n-grams) and a shared global prior (green dot). Since this work ultimately cares about performance on English Fisher, the corpus-specific model (red dot) will be used to compute performance.

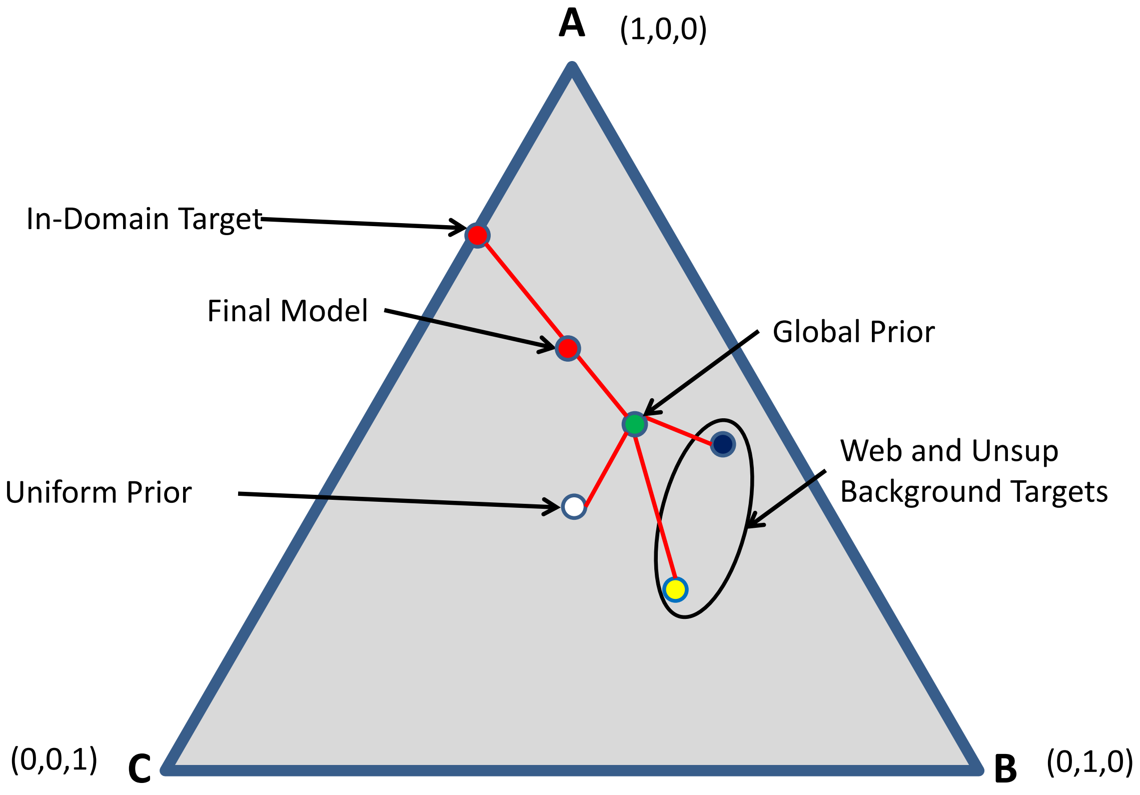

Figure 4.5: *Visualization of Hierarchical Bayesian Adaptation* - Instead of comparing separate models as in Figure 4.1, one model is jointly learned on all three corpora. The two background corpora (blue and yellow dots) influence the global prior (green dot) which in turn influences the in-domain corpus (red dot). This influence is moderated by a separate variance for all three corpora and the global prior.

Before the web data is combined with expected counts, there are two different algorithmic choices. Using the web corpus as a background model, the MAP adaptation experiments (denoted $A \rightarrow B$) of the previous section were repeated using joint estimation (denoted $A + B$). As seen in Figure 4.6, MAP adaptation out-performs joint estimation. However, the gap between the two schemes decreases as the size of the target corpus increases. The reason for this is the connection between corpus size and variance. Small amounts of target data have higher sample variance, requiring a smaller prior variance. This in turn gives them a larger weight in the global prior update, reducing the benefit of the web corpus. MAP adaptation factors these two requirements out, resulting in a better model. Under this scheme, where one only cares about performance on one corpus, there is no interest in improving the model of web text and thus do not benefit from joint estimation.

Although expected counts reduced WER by around 2% absolute, they were completely dominated by the conversational web corpus. Figure 4.7 shows that one can extract additional gain over the web corpus through joint estimation. Despite the semi-supervised counts being significantly worse than either the target or web corpora, they provide additional value. However, the optimal method of combination was *not* to train one joint model on all three corpora. Given the results in 4.6, the following strategies were compared with results shown in Figure 4.7:

- **Web** → **Target** - Ignore the expected counts from the unlabeled audio.
- (**Web** + **Unsup**) → **Target** - Jointly learn web and unsup and adapt to target.
- **Web** + **Unsup** + **Target** - Jointly learn all three.

The choice of method does not significantly differ (less than five points of perplexity). Joint

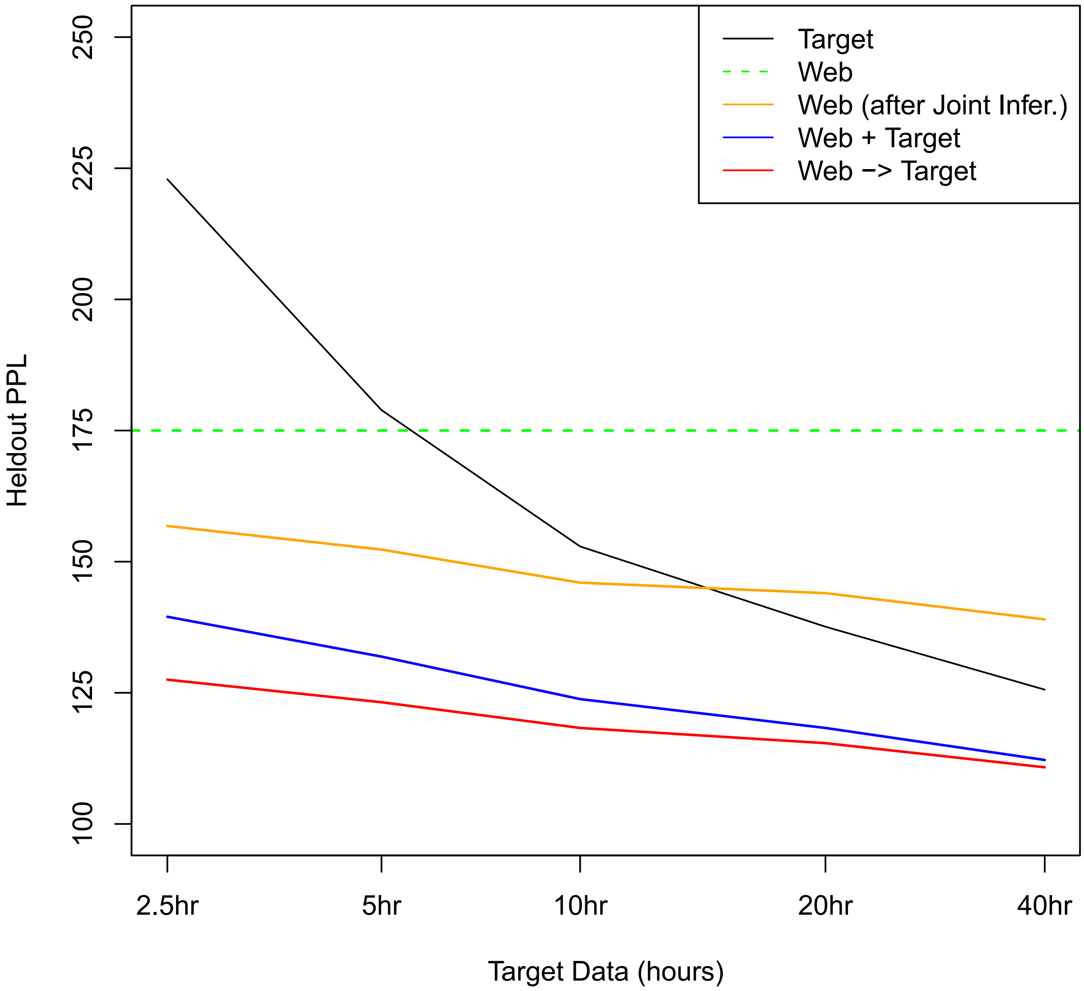

Figure 4.6: *Comparing MAP vs. Joint Inference* - Held-out perplexity was statistically insignificant between the two models. A fixed 4M token corpus of web data was combined with varying amounts of in-domain transcripts (2.5 to 40hrs). The perplexity of the web corpus (blue line) is constant across the x-axis as it does not change. Performance increases as more in-domain data is added (black line). When adapting from the web to the in-domain corpus, either MAP adaptation (yellow line) or joint estimation (green line) can be used. However, MAP outperforms joint adaptation consistently since the target corpus is the only one of interest. Although the web corpus improves on held-out performance when jointly adapted (orange line), the target model is best.

estimation is sensitive to the size of the target corpus due to the variance sensitivity discussed earlier. Because of this, the two step process of first training a joint model on all background corpora and then adapting to the target domain provides the best performance. Despite the algorithmic differences, the semi-supervised counts do provide a modest additional gain. Even in this best case scenario, where a strong out of domain corpus is significantly better than in-domain transcripts, semi-supervised language modeling provides a small, but consistent gain.

This section explored the best method for combining a variety of unreliable n-gram statistics. MAP adaptation provides a robust framework for using expected counts in conjunction with a small amount of in-domain transcripts. Expected counts provide a 2% absolute reduction in WER on average for conversational English. Hierarchical domain adaptation provides an extension to combine multiple background corpora.

4.8 Incorporating Weak Constraints

The log-linear formulation allows for more robust use of expected counts (Section 4.7.1) and incorporates multiple background corpora as multi-modal priors (Section 4.7.2). The greater motivation for this model is the ability to incorporate a variety of features. Recall that the key to a log-linear model is the *feature function* which maps from a history, word pair to a real valued vector. The dot product of this vector with the model parameters gives a score, which is then normalized with the partition function, defined as

$$P(w|h) = \frac{\exp \sum_{k=1}^{N} f_k(h,w) \cdot \theta_k}{\sum_{w' \in \mathcal{V}} \exp \sum_{k=1}^{N} f_k(h,w') \cdot \theta_k}. \tag{4.41}$$

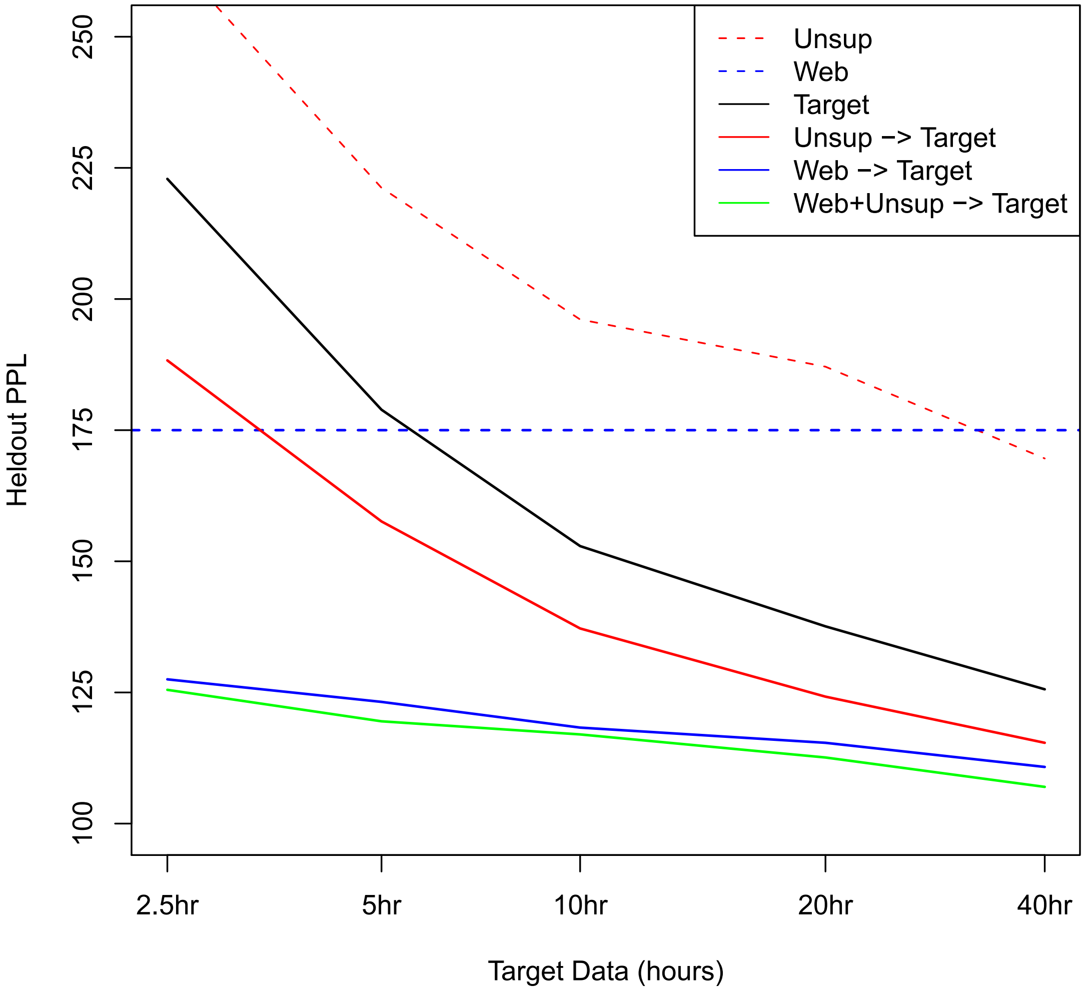

Figure 4.7: *Combining Multiple Priors* - 4M tokens of web text and 400 hours of automatically generated expected counts were used as background corpora for in-domain transcripts. Held-out perplexity is reported due to the small changes in WER across models. The expected counts (red dashed line) are worse than the web counts (blue dashed line) even after adaptation (red line and blue line respectively). Still, they manage to provide additional information, improving performance (green line). Joint estimation allows for both models to be used instead of having to select only one.

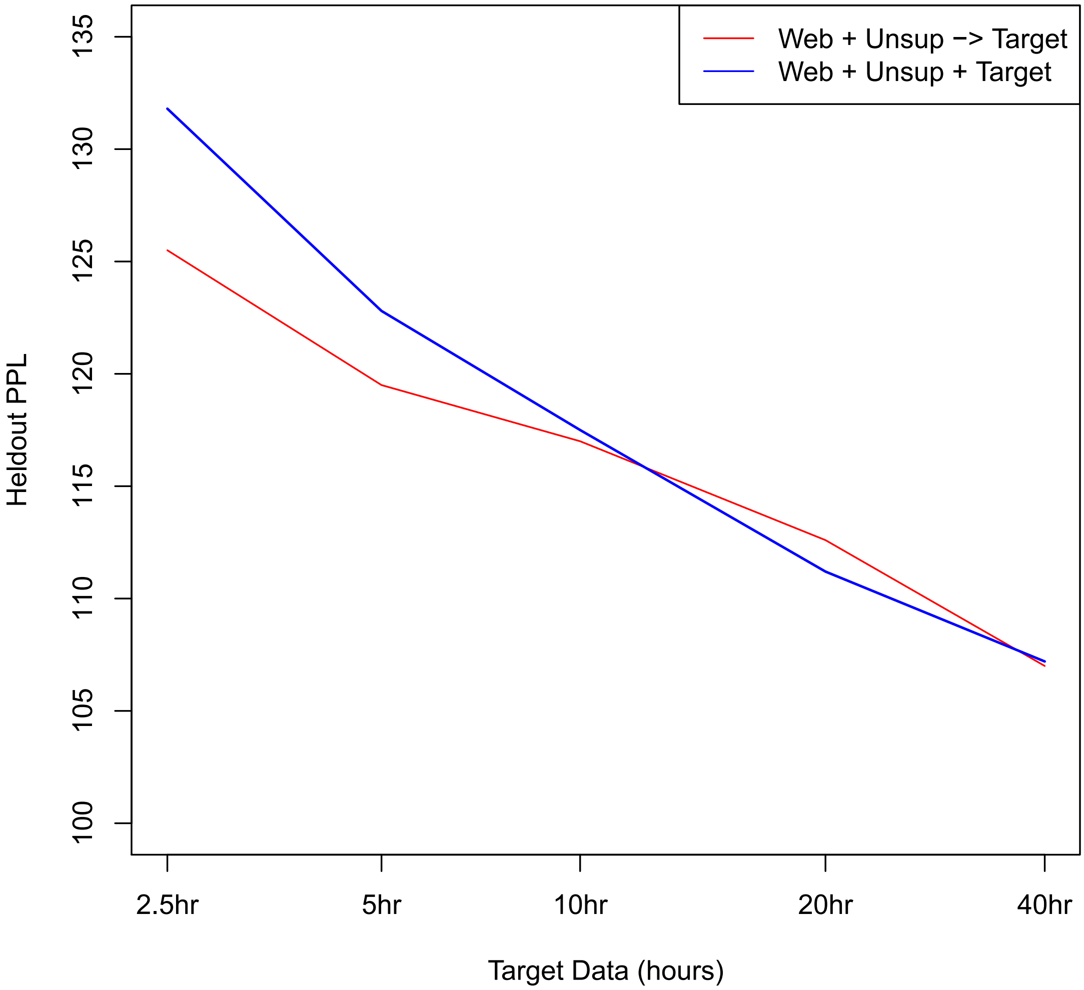

Figure 4.8: *Joint Inference is Sensitive to Data Size* - The optimal method of combining 4M tokens of web text with 4M tokens of expected counts was contrasted. At small amounts of data, joint estimation is sensitive to the size of the target variance, resulting in worse performance (blue line). However, jointly estimating one background model on the expected counts and web corpora and then adapting leads to best performance under all conditions (red line).

The original motivation for log-linear models was to incorporate additional features beyond n-grams. Unlike a non-parametric model, the features are all on equal footing - a unigram feature fires with some weight along with a trigram feature. There is no notion of "backing off" to a lower order n-gram. Additionally, the model easily incorporates features such as part of speech tags, class n-grams, skip n-grams, trigger words, etc... This flexibility will be useful in two ways. First, one may trust coarser (lower order) statistics over detailed one (higher order) and second, this chapter will introduce a broad category of features called *marginal class constraints.*

As an example of the power of a log-linear model, assume there exists a method of estimating the true unigram frequencies of the Fisher English corpus. This is a weaker form of knowledge than the full set of bigrams or trigrams seen in 200 hours of manual transcripts. As detailed in Table 4.4, a standard n-gram language model is not able to use these statistics. There is no reduction in WER for having the correct unigram counts and barely any for having correct bigram counts.

However, a log-linear language model can exploit these statistics through MAP adaptation. First a background model was trained using 190 hours of expected counts decoded with a ten hour acoustic and language model. These are the same expected counts from the earlier non-parametric work. Next, the expected counts from the 190 hours were adapted to the 10hr manually transcribed n-gram constraints, resulting in a WER reduction from 41.7% to 40.6%. Then unigram counts were estimated for all words seen in the 190 hours of manual transcripts - not the held-out test set - and adapted the semi-supervised model to these unigram counts. Since each word appears in the model as a unigram feature (even if unseen during training) adaptation is quick and efficient. This resulted in a sizable 1.6% WER reduction and an increase in WER Recovery from 16.7% to 40.9%. MAP adaptation essentially groups the unigram features into a separate category with tied variance. This step could in principle be done during model estimation by implementing per-feature variance. While this is a powerful advantage of log-linear models, it is difficult to estimate per-feature variances given small amounts of data. Furthermore, it is unlikely that one would know word counts without knowing any higher order n-gram statistics. The next section will consider a more likely scenario - there is a notion of how broad classes of words behave.

Model / Training Data	**10hr**	**10+190hr**	**Oracle Unigram**	**200hr**
Non-Parametric	41.8	41.3	41.3	35.0
Log-Linear	41.7	40.6	39.0	35.1

Table 4.13: *Gains for Incorporating Unigram Oracle* - All decodes with 10hr AM. Four sets of statistics for estimating a non-parametric model with Witten-Bell smoothing or a log-linear LM are used. In the fully supervised cases (10hr and 200hr) of manual transcripts, both models perform the same. The log-linear model makes better use of semi-supervised expected counts (10 + 190hr) and is amenable to adaptation to lower order constraints (oracle unigram). The non-parametric LM is unable to due to the back-off nature of its likelihood computation.

4.8.1 Marginal Class Constraints

While one may not have direct domain knowledge in the form of n-gram counts, one likely have knowledge of the types of errors made by the transcribers. Ideally, this would be a word by word confusion matrix resulting in the true unigram frequencies for the domain. However, learning such a matrix requires a large amount of parallel data of low and high quality transcripts to estimate such a model. Instead, one may have a small amount of data from which to draw broad conclusions, not at the word level, but over *groups* of words. And since the goal is to modeling language, the kinds of errors that matter are those that affect the frequency of the group.

For example, an LVCSR system tends to under-generate hesitations in the one-best output. Precisely, the frequency of the group of hesitations *as a whole* is under counted in the one-best compared to the manual transcripts. Of course, one can make such statements about individual words, but then one is limited to those word types seen in any held-out data. If instead words are clustered together, one can make broader conclusions about words which do not appear in the held-out data, but belong to a class which does occur in held-out data.

This form of broad categorical knowledge requires two pieces of information: a defined subset of the vocabulary and a frequency estimate for that subset as a whole. At one extreme is the entire vocabulary which occurs with a frequency of one - a superfluous constraint. And at the other is a single word type with its frequency - useful, but high variance. In between are lies the opportunity to inject knowledge not of the domain, but of transcription error.

A log-linear model easily incorporates this knowledge through MAP by the addition of *marginal class features.* After a group of words have been selected to belong to class $\mathcal{C}$, the feature $f_{\mathcal{C}}$ is added to the model defined by

$$f_{\mathcal{C}}(h,w) = \begin{cases} 1 & \text{if } w \in \mathcal{C} \\ \\ 0 & \text{otherwise}\,. \end{cases} \tag{4.42}$$

Similar to unigram features, the feature is a *marginal* feature since it fires independent of the history h. There may be multiple, overlapping features that fire simultaneously. To take advantage of these constraints, MAP adaptation is used after semi-supervised adaptation.

To verify the usefulness of these new class constraints, this section reports results using a small corpus of phonetic transcripts. The phonetic vocabulary of the English Fisher word dictionary was used for a total vocabulary size of 49 phones. Then 30,000 tokens were generated for training by replacing a word transcript with the phonetic pronunciation. Next, this background corpus was randomly corrupted at three levels of error - 25%, 50% and 75% - by randomly swapping a true phone for another random phone.

To generate meaningful phone classes, an agglomerative tree was built over the phone data estimated on truth by maximizing mutual information. This resulted in phone classes such as the vowels, fricatives and other meaningful phonemes belonging in subclasses. The root of the three had all phones in one class and the leaves were all 49 phones in their own class. Cutting the tree at a given node depth results in a partition of the vocabulary and defines the set of phone classes. The number of classes increased from two to four, eight, etc... all the way to the 49 unigram classes. For a given partition, the *true* frequency of each of the class in that partition was used as the form of domain knowledge.

Performance was compared across two dimensions - the quality of the background model (25% to 75% error) and the quality of the domain knowledge (2 classes to 49). Figure 4.9 confirms the intuition that stronger domain knowledge is more important when faced with higher error. The heavily corrupted background language model (75% corruption) greatly benefits from any form of domain knowledge while the low corruption only shows a meaningful gain when given the true unigram counts.

After confirming the potential of marginal class frequencies on phone data, the features were applied to the English Fisher corpus. Errors were extracted from the one-best output from 190 hours of English Fisher decoded with a ten hour LVCSR system. Several types of errors that the recognizer tended to make were systematic. The following categories are defined:

1. **Short Words** - Words less than three phonemes. The recognizer tends to underproduce these words.

2. **Long Words** - Words with more than six phonemes.

3. **Special Characters** - Words that included on alphanumeric characters such as '[' or '-'. These entered the vocabulary due to transcriber notations in a larger set and should not actually appear.

4. **In-Training** - Words that appeared in the original ten hours of transcription.

5. **Hesitations** - Manually defined set of words that denote hesitations - UH-HUH, UH, UM etc. . .

6. **Singletons** - Words that only appear once in the one-best output of the 190 hours.

Notice that these sets vary in size and greatly overlap each other. The word 'UH-' is an in-training, short hesitation with a special character. After defining these classes, their frequency was estimated from a three hour manually transcribed held-out set. Their frequencies were estimated from the 190 hours of manual reference as well as from the test set,

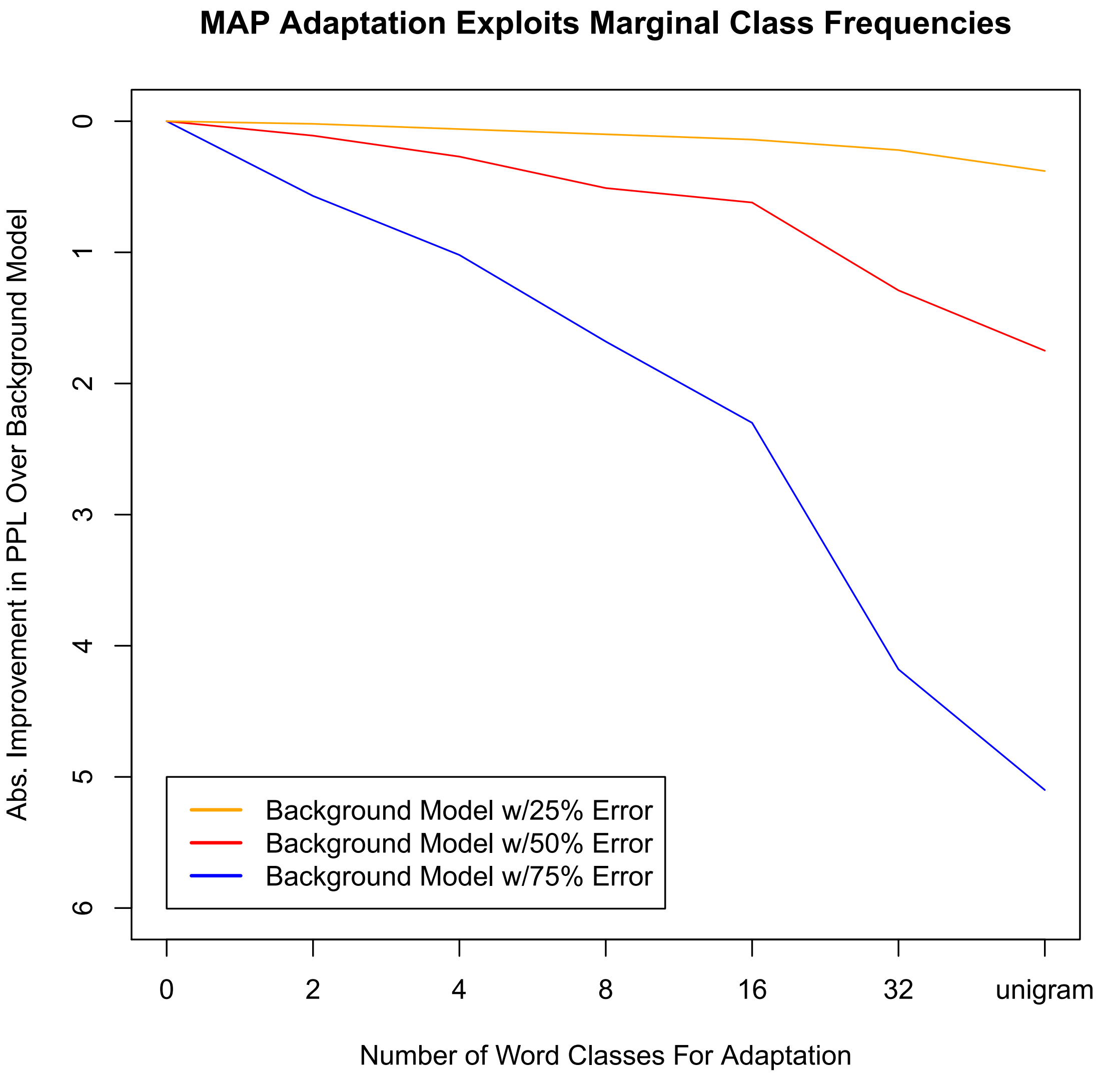

Figure 4.9: *Value of Class Constraints Increases with Noise* - 30K tokens of phonetic transcripts were artificially corrupted at 25%, 50% and 75% random error by swapping a training token with one of the other 49 phonemes. These background models were then adapted with *true* class constraints derived from the actual transcripts. The number of classes ranged from two to 49 (each phone in its own class) and were built by agglomerative clustering. Knowledge of marginal class constraints are of greater value at high error rates. The high-error background model (blue line) benefits the most while the lower error background model (orange line) barely shows a gain.

but found all three estimates to be nearly identical. Note that estimating these frequencies from the one-best would result in an unimproved model. Drawing the class frequency estimates from the one-best would not violate any of the constraints.

Table 4.14 details the results. Class constraints were able to capture 80% of the possible gain for knowing unigram counts of the 190 hours of manual transcripts. This indicates that the domain knowledge (the set of classes) represents the transcription errors well. However, after adaptation to the ten hour model, the gain reduces to only 30%. The classes do not necessarily provide additional information that is present in manual transcription. All results are significant when adapting from 190 hours ($p < 0.001$), except for the hesitations, ($p < 0.05$). While there is a substantial reductions in perplexity, these do not carry over to reductions in WER due to scoring. Hesitations are optionally deletable during scoring - the recognizer does not get penalized for deleting them. When starting with the $10 + 190$ condition, only the special character class and the final set of all, except phones were statistically significant ($p < 0.001$).

Given a small amount of manual data, this section demonstrated that marginal class constraints could improve performance *and* improve over simply extracting unigram counts from the manual data. But of course, one could extract higher order n-grams from the data used to estimate the class frequencies. In Figure 4.10, the amount of manually transcribed held-out data was swept and used to compute three sets of statistics. One - using the classes described in Table 4.14; two - the unigram counts seen in the held-out data; and three - the unigrams and bigrams. The 190 hour background model was adapted to each of these statistics and report the gains in perplexity. The domain knowledge captured by the

	190hr unsup.		10+190hr	
Feature	**PPL**	**PPL Rec.**	**PPL**	**PPL Rec.**
Baseline	208.2	-	146.6	-
Singletons	208.1	0%	146.6	0%
Short Words	205.0	6%	146.3	2%
Special Chars	204.1	8%	**142.6**	**27%**
In-Training	202.6	11%	146.4	1%
Hesitations	**179.4**	**56%**	146.6	0%
All	181.6	51%	144.2	16%
All - phones	166.6	80%	142.1	30%
held-out Unigram Counts	174.7	65%	129.5	
190hr Manual Unigram Counts	156.6	-	131.9	-

Table 4.14: *Gains for Incorporating Class Constraints* - Log-linear language models built with n-gram features were adapted with different marginal class constraints. Perplexity (PPL) is reported on a held-out test set as well as PPL Recovery for each class. The upper bound for this method is the unigram constraints - each word is in its own class with the correct class frequency. The most useful class differs depending on the availability of the 10 hour constraints. Knowledge of hesitation frequencies (row 6) is most useful, but redundant once the ten hours is available. Words which contain non alphanumeric characters such as restarts and pauses are over-estimated in the ten hours (row 4). Combining all classes gives 80% of the possible gain - indicating that the manually designed classes capture much of the transcription error. Furthermore, the class structure (row 8) beats out using the unigram counts directly using the held-out data (row 9) indicting the value of additional domain knowledge.

class constraints improves performance when only small amounts of data are present. Their frequency estimates quickly saturate with a very small number of manually transcribed tokens. However, there reaches a point where it is better to use the n-gram statistics directly.

4.8.2 Evaluating Constraint Robustness

While one may notice that non-professional or automatic transcribers tend to make certain kinds of errors, these errors impact the work only as much as they impact downstream performance. This chapter proposes the following metric to evaluate the usefulness of a proposed class or constraint. It may be of use on a future condition with limited manual transcripts or for selective constraint adaptation.

For any proposed constraint, there are three estimates of its frequency: the *new estimate* (from some other set or from expected counts), the *truth* and the *prior*. Given these three counts, one desires a constraint estimate to be *close* to the truth and *more predictive* than the prior. Since constraints are a function of history - word pairs, one can map a corpus to a set of binary events, one for each history and word token. Then it is considered for that pair if the feature fires in the *reference*. This gives a sequence of N trials, where N is the size of the corpus and the goal is to model the rate at which the constraint in question fires. The natural model is then a simple binomial model defined as

$$P(X = r) = \binom{N}{r} p^r (1-p)^{N-r} \tag{4.43}$$

where p is the estimated frequency and one seeks to estimate the likelihood of seeing r firings in N trials. There are two proposed estimates of p - the new estimate $\hat{p}$ and the

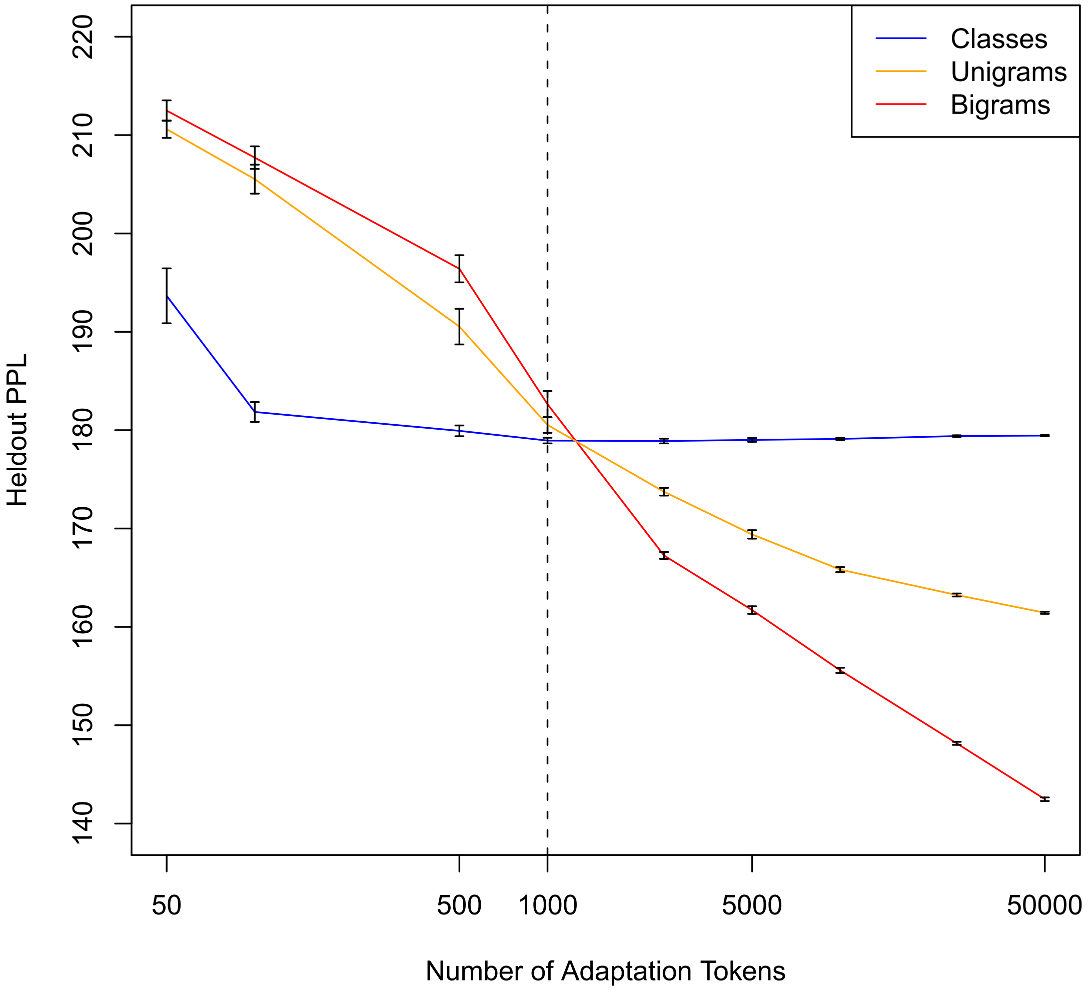

Figure 4.10: *Marginal Class Constraints Help Low-Resource Modeling* - Using the set of manually designed classes from Table 4.14, their frequencies were estimated on increasing amount of manual transcripts. Either unigram or bigram constraints were also estimated. Thus for each amount of data, three separate sets of constraints were extracted: class, unigram or unigram and bigram counts. The class estimates converge quickly, showing the value of domain knowledge. However, with enough true training tokens, it is better to use the data directly instead of through the class constraints.

prior p_0. The new estimate should be used if it is 'closer' to the truth but otherwise stick with the prior. So a log likelihood ratio test is formed and the statistic

$$\begin{aligned}
\Lambda &= \log \frac{P(X = r|p = \hat{p})}{P(X = r|p = p_0)} && (4.44)\\
&= \log \frac{\binom{N}{r}\hat{p}^r(1-\hat{p})^{N-r}}{\binom{N}{r}p_0^r(1-p_0)^{N-r}} && (4.45)\\
&= \log \left(\frac{\hat{p}}{p_0}\right)^r \left(\frac{1-\hat{p}}{p_0}\right)^{N-r} && (4.46)\\
&= r\log\frac{\hat{p}}{p_0} + (N-r)\log\frac{1-\hat{p}}{1-p_0} && (4.47)\\
& && (4.48)
\end{aligned}$$

which can be quickly computed and has an interpretable value. **Positive** - The reference is farther from the prior than the estimate, so the *estimate is better.* **Close to Zero** - The prior and estimate are about the same - estimate is not predictive. **Negative** - The reference is closer to the prior, so the *estimate is worse.* This statistic has another motivation from information theory. If $\tilde{P}$ is the empirical frequency of the constraint in the held-out data, then one can express Equation (4.47) in terms of KL divergence as

$$\frac{\Lambda}{N} = \frac{r}{N}\log\frac{\hat{p}}{p_0} + \frac{(N-r)}{N}\log\frac{1-\hat{p}}{1-p_0} \tag{4.49}$$

$$= \tilde{P}(X)\log\frac{\hat{P}(X)}{P_0(X)} + \tilde{P}(\neg X)\log\frac{\hat{P}(\neg X)}{P_0(\neg X)} \tag{4.50}$$

$$= \tilde{P}(X)\log\frac{\hat{P}(X)}{P_0(X)} - \tilde{P}(X)\log\frac{\hat{P}(X)}{\hat{P}(X)} + \tilde{P}(\neg X)\log\frac{\hat{P}(\neg X)}{P_0(\neg X)} - \tilde{P}(\neg X)\log\frac{\hat{P}(\neg X)}{P_0(\neg X)} \tag{4.51}$$

$$= \left[\tilde{P}(X)\log\frac{\hat{P}(X)}{P_0(X)} - \tilde{P}(\neg X)\log\frac{\hat{P}(\neg X)}{P_0(\neg X)}\right] + \left[\tilde{P}(X)\log\frac{\hat{P}(X)}{\hat{P}(X)} - \tilde{P}(\neg X)\log\frac{\hat{P}(\neg X)}{P_0(\neg X)}\right] \tag{4.52}$$

$$= D(\tilde{P}||P_0) - D(\tilde{P}||\hat{P}). \tag{4.53}$$

The result is the difference of the KL divergences of the two estimates. This has a nice graphical interpretation as shown in Figure 4.11. It is not enough for the new estimate to be close to the truth, it must also be significantly closer to it than the prior to be truly useful. This metric was evaluated on all the word types that appear in the one-best output of the 190 hour decode. Table 4.15 details the top improved. These are the set of words which the estimate from the 190 hours expected counts are improved over the original ten hours. What immediately stands out are the topical words unseen in the ten hours. Topics such as *minimum wage*, *sports* and *entertainment* were randomly assigned to the conversant. And it is these topic-dependent words that are most improved by semi-supervised counts - the very category of words one would hope to recognize. Conversely, words which are *worst* estimated are hesitations and short words. These are acoustically varied words and are typically one or two phonemes long. Of course, computing this metric requires knowledge of the true estimate. So it doesn't matter if the new estimate is an improvement over the

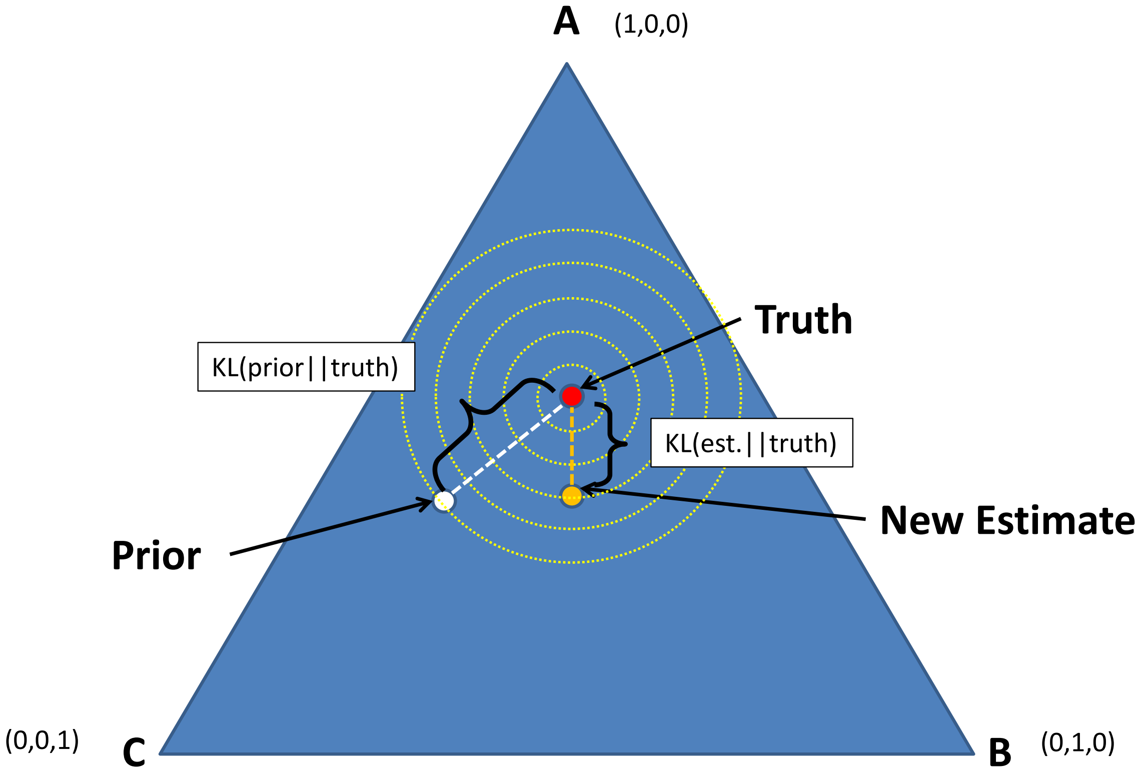

Figure 4.11: *Visualization of Semi-Supervised Metric* - The proposed metric compares the difference of the prior estimate to the true estimate and the new estimate to the truth. A value greater than zero means that the new estimate is closer to the truth than the ten hours and is an improvement. Conversely, a negative value means the prior is closer and the new estimate should be ignored.

		Log Probability		
Word	**Score**	**10hr**	**Unsup**	**Truth**
WATCH	0.003384	-2.164	-2.666	-2.938
MINIMUM	0.002756	-5.912	-3.976	-3.193
WAGE	0.001147	-5.912	-5.154	-3.180
SPORTS	0.000939	-2.622	-3.011	-3.277
BASEBALL	0.000810	-2.821	-3.494	-3.612
FOOTBALL	0.000778	-2.767	-3.187	-3.533
COMEDY	0.000672	-5.912	-4.155	-3.737
BASKETBALL	0.000664	-2.896	-3.559	-3.664
HOBBY	0.000584	-5.329	-4.114	-3.630
WATCHING	0.000503	-2.855	-3.242	-3.444
SMOKE	0.000492	-4.956	-4.104	-3.545
READ	0.000478	-4.567	-3.846	-3.447
HOBBIES	0.000471	-5.328	-3.923	-3.742
MOVIES	0.000391	-4.723	-3.750	-3.610
SPORT	0.000381	-2.969	-3.230	-3.759

Table 4.15: *Most Improved Words From ASR Output* - 190 hours of Fisher English was decoded with a 10hr AM/LM LVCSR system. All words seen in the one-best output were ranked by the semi-supervised metric. The most improved words (with highest score) are the topical words unseen in the original ten hours but frequently appear in the 190 hours.

prior as one would simply use the truth. The value of this proposed metric is on future held-out tasks where one believes the broad class categories to be robust across domains. Table 4.16 ranks the manually defined classes from Section 4.8.1 by this metric.

Hesitations are better estimated by a small amount of manually transcribed data. These conclusions can be used on future conversational tasks where held-out data is lacking.

	Log Probability			
Class	**held-out**	**Unsup.**	**Truth**	**Metric**
Special Chars	.0042	.0720	.0668	.1235
Hesitation	.0067	.0449	.0293	.0175
In-Training	.9032	.9285	.9556	.0129
Singletons	.0075	.0147	.0074	-0.002
Phone Length < 6	.8705	.8451	.9123	-0.011
Phone Length < 4	.7212	.6027	.7243	-0.032
Phone Length < 2	.0817	.0309	.0842	-0.044

Table 4.16: *Marginal Classes Ranked by Metric* - The manual classes used in Section 4.16 are ranked according to their usefulness. Two estimates are compared - using 3hrs (30k tokens) of held-out data and the expected counts from 190 hours decoded with a 10hr AM and LM. When the true frequency of these groups are known, the semi-supervised metric was computed and used to rank the classes by their usefulness. Positive scores should use the unsup. frequency while negative scores should use the held-out frequency.

4.9 Discussion

This chapter explored using semi-supervised language modeling to improve performance when there is no budget for human labor. This chapter used an LVCSR system trained on small amounts of data to generate a large amount of automatic transcripts. While theoretically well motivated, expected n-gram counts generated by a lattice do not perform well. Instead, a word-level confidence model of the one-best provides a better estimate of n-gram counts seen in transcripts. A non-parametric language model has a modest, but significant gain at high error rate. However, it is unable to exploit weaker constraints such as correct unigram counts.

To incorporate these statistics, a log-linear language model was used, which is competitive with state of the art smoothing when estimated on manual transcription. It offers slightly stronger semi-supervised performance by using the expected counts as a prior for MAP adaptation. An extension to Hierarchical Bayesian adaptation allows for multiple background corpora to be expressed as a multi-modal prior. The true power of the log-linear model is the incorporation of marginal class constraints.

For many situations,there may be knowledge of the kinds of transcription errors made by the transcriber (automatic or human). This chapter proposed expressing this domain knowledge by adding class constraints to a log-linear language model. By restricting constraints to marginal classes (which depend only on the lexical identity), the log-linear model can still be expressed in a back-off format for use in decoding. An improved estimate of these class constraints can easily be incorporated through MAP adaptation and gives sizable gains over standard semi-supervised language modeling.

There are many components to the semi-supervised pipeline for future work to consider. Improved posterior estimation (Section 4.5.1) would result in more accurate counts. Vocabulary induction through semi-supervised means would remove much of the unlikely chaff that appears in the recognition output. Reducing the vocabulary size without a loss in word type recall would improve the quality of the recognition lattices. Sub-word recognition, with a vocabulary size of a few thousand types, might be particularly well suited for this chapter. While this work was unsuccessful, future work in count regression could incorporate additional features estimated from out of domain data. And answering the question as to *why* an n-gram over or under generates would be a worthwhile contribution. With this information, it may be possible to introduce features observable in the recognition output.

Further down the pipeline, one could consider alternate model formulations besides log-linear models. While this chapter emphasized their use for incorporating domain knowledge, an alternate approach motivation would be to increase parameter sharing. The success of semi-supervised acoustic modeling is likely due to the continuous space of Gaussian mixture models. This reduces the impact of labeling errors, which are now no longer binary losses.

One could consider a continuous space language model [138] to learn a projection of word types to a low dimension space. Previous work has learned both the projection and output layer weights using one corpus. Instead, the projection could be estimated on a large amount of semi-supervised training data. The final discriminative output layer could then be adapted with a small amount of in-domain data. A continuous space model could also be used to learn a recognition robust parameter space. Instead of estimating a space which places semantically related words near one another, one could design a space such that acoustically confusable words are connected. Thus when one word is seen in the recognition output, its acoustic neighbors also receive some training probability mass.

The results of this chapter feed back to Chapter 3 in two ways. First, if human transcription quality is similar to that of the recognizer, the recognition output may provide complementary information. Recognizers make different systematic errors than humans. For instance, they do not misspell words, but freely create grammatically unlikely sentences. Second, non-expert transcribers could be tasked with acquiring weaker sources of domain knowledge. While Chapter 3 elicited complete utterance transcripts, one could ask for a wide range of alternate statistics. Relative word rankings could be estimated by tasking workers with sorting words from the vocabulary. Part of Speech frequencies could also be estimated. Novel n-grams could be elicited by presenting n-grams with the final word removed. But these intriguing ideas must be contrasted with direct transcription. Chapter 5 will consider semi-supervised language modeling for a more constrained task. Instead of the transcription quality of all words, the chapter will focus on the search quality of a subset of words.

Chapter 5

Keyword Search for Unseen Terms

A language model is important for many automatic speech applications beyond word transcription. A large vocabulary continuous speech recognition (LVCSR) system is a crucial component of audio keyword search. In this task, a user searches a corpus of audio recordings (such as telephone conversation in a call center or lectures) for instances of a query (either single or multiple words). The retrieval task is then to return a set of individual recordings in the corpus and time offsets into those recorings which contain potential instances of the keyword. As an example, Google provided an index of many of the speeches by the 2008 U.S. presidential candidates, Barack Obama and John McCain. Users were able to search for such phrases as "economy" or "immigration reform" and listen to how the two candidates spoke on those matters.

This is not a trivial search since automatic speech transcription is imperfect. Unlike text retrieval, where much research is in (i) determining the semantic intent behind the query word, and (ii) evaluating the relevance of each document in the collection to the users' intent, audio retrieval is attempting to perfect the first step: creating an index of spoken words.

Good keyword search performance on speech corpora requires well trained acoustic and language models and a large vocabulary to cover potential search terms. And such state of the art systems typically require hundreds of hours of expensive and time-consuming in-domain transcripts. Yet users are constantly searching for new terms not present in the LVCSR system's training data. The typical formulation for this scenario is detecting and recovering out of vocabulary (OOV) terms: a word present in the search audio that was not in the decoder and thus unseen in the recognition output.

However, once orthographically searched by a user, the term becomes known to the system for re-decoding. In such settings, if the system has the option to re-process the search audio, then the search term is no longer OOV, but simply **O**ut **O**f **T**raining (OOT). OOT terms, by definition, are present in the decoding dictionary, but lack training data in the acoustic model (AM) and language model (LM). While an LVCSR system can in principle recognize these terms if they are present in the recognition vocabulary, the system's accuracy in doing so is significantly degraded.

The appropriate method to improve search performance on OOT terms depends upon the available resources. Budgets may be limited or out of domain corpora may not be available in sufficient quantities (e.g. colloquial Arabic or Hindi sub-dialects). This work explores two contrasting options: find instances in some manually transcribed corpus (audio or text) or make do without human transcription (with or without audio). In this work, pronunciations are not a great concern as they can either be elicited or reasonably generated from an orthography [139]. Instead, the focus is on efficiently generating training samples of the keyword of interest in the new domain.

5.1 Previous Work

A closely related task is improving search performance on OOV terms after recognition has been performed using a fixed system. The goal is to still successfully return audio recordings despite the search term being unseen in the recognition vocabulary. The two subtasks are to first automatically estimate a pronunciation from the written query and second, search through an audio corpus after recognition. Typical approaches use a subword system and approximate matching with reasonable success [140] [43]. Yet prior work has been reluctant to re-engineer models after querying due to decoding speed constraints or use of a large corpus of audio. This work differs from OOV search in two ways: the pronunciation is assumed known and re-recognition of the audio data is allowed.

Though called a different name, previous work on improving performance on OOT terms exploited parallel data to extract new vocabulary terms and LM training data. Broadcast news recordings can benefit from newswire articles written the same day [41]. Audio-visual recordings have associated meta-data such as keywords, document summaries or archivist notes [42].

As described earlier in Chapter 4, semi-supervised approaches for language modeling considered the low-resource scenario. However, the work focused on word error rate (WER) averaged over *all* words and report modest gains [45] [85]. This work adapts these techniques to focus on only a subset of words and optimize a different metric.

5.2 Experimental Description

As in earlier chapters, the primary domain investigated here is English conversational telephone speech (CTS). State-of-the-art WER for such domains is around 20% using 2000 hours of speech, billions of words for LM training and very large decoding dictionaries. In contrast, LVCSR systems can achieve less than 10% WER on an easier domain like broadcast news. The relatively high WER leaves room for semi-supervised methods to improve performance.

The following experiments were conducted on English CTS since it is the largest transcribed corpus in the CTS domain. This allows comparisons of the semi-supervised methods to fully supervised transcription. The insights developed here should hold across other domains and languages, as there are no language-specific assumptions made here.

5.2.1 Corpora

For initial training data, the 370 hour Switchboard corpus [102] was used from which acoustic and language models were trained and the decoding dictionary was derived. The search corpus was the 1850 hour Fisher collection, again English CTS. Fisher was partitioned into a 150 hour evaluation corpus and a 1700 hours 'unlabeled' corpus which was treated as untranscribed for use in semi-supervised training. In order to ensure a representative sample, the test corpus and query terms were chosen such that the frequency of the queries in the 150 hours matched their relative frequency in the 1800 hour corpus.

5.2.2 Selecting Queries

To evaluate this task, this work required queries that didn't appear in Switchboard so that they would be OOT to the system and appeared in Fisher a fair number of times so that semi-supervised methods could meaningfully improve performance. Of the 25,000 word types appearing only in the 1850 hours of Fisher data, those that occurred at least 30 times or more in Fisher were selected. This list was manually pruned to remove alternate spellings, plural forms and other unsuitable queries, leaving 126 test terms. One third of the samples for each word were held out for the 150 hour evaluation set and the rest left in the 1700 hour development corpus. These 126 terms followed Zipf's rule writ small, with a few occurring hundreds of times and a long tail occurring a minimum of 20 times in the 1700 hour corpus. The resulting set contained examples like ENRON, OSAMA, GOOGLE, KOBE, and OVEREATING. These topical terms reflect that Fisher was collected in the early 2000's while Switchboard was early 1990's when these examples were not yet topics of conversation.

5.2.3 LVCSR and Indexing System

A state-of-the-art same multi-pass LVCSR system [19] was used with state clustered Gaussian tied-mixture quinphone acoustic models. See Section 2.1 for a more complete description. The language models used in this chapter are based on the standard trigram language models with Witten-Bell smoothing. Since a large amount of transcribed data is available, the standard trigram back-off LM offers competitive performance without needing the large memory and training times of a log-linear model. Decoding speed was around ten times faster than real time.

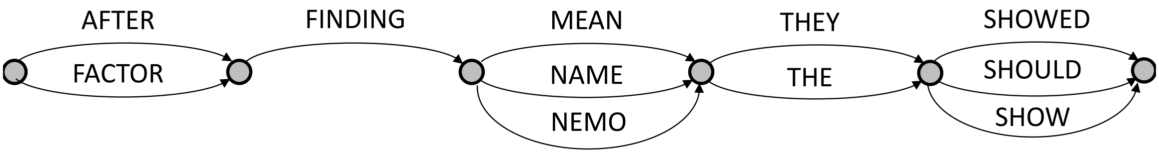

Figure 5.1: An example confusion network

First, the recognition lattices produced by the LVCSR component were converted into a confusion network of competing words. A confusion network is an acyclic directed graph with a sequences of 'bins' compactly representing the set of hypothesized word sequences for an utterance ̇The nodes of the network fall between words while each arc represents a hypothesized word. Each bin corresponds to a word 'slot' where all words within that slot are overlapping in time.

Confusion networks offer a very compact and efficient way to store the occurrence of word tokens (along with their posterior probability) and are easily converted into a reverse index. However, they have two distinct limitations when compared to lattices. First, they are inexact with respect to time. Two words of unequal length that overlap the same time window may be forced into the same bin, making it difficult to recover exact word timings.

Second, confusion networks force multi-word tokens into one bin. For instance, if the words WHATEVER, WHAT and EVER are in the decoding dictionary, two acceptable hypotheses would be WHATEVER and the two word sequence WHAT EVER. But since a confusion network does not allow for word arcs to skip time nodes, the word WHATEVER would be forced to be in the same bin either as WHAT or EVER as seen in Figure 5.2.

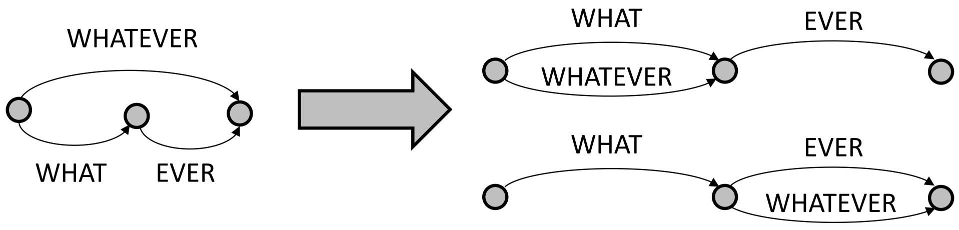

Figure 5.2: Limitations of confusion network representation

Despite these two limitations, confusion networks are an effective method for search of single keywords, which is the focus of this chapter. Extensions to multi-word queries would require use of a lattice or alternate methods. For each instance of the keyword found in the confusion nets across a corpus, the identity of the utterance which contains it as well as a posterior score within the bin is placed in a list. It is this list which is then evaluated for search effectiveness.

5.2.4 Metrics

Keyword search is a ranked retrieval task common throughout text processing. The audio recordings of this chapter are analogous to text documents. The indexer includes a score (such as a word posterior score) and thus imbues a ranking over the set of returned results (in contrast to *un*ranked retrieval). There is no one relevant document that a user is searching for, but instead all documents that contain the keyword. For this task, relevance was judged at the utterance level, where utterances are created with automatic segmentation and typically one to ten seconds long.

The appropriate metric for this task is mean average precision (MAP) defined as

$$MAP(D) = \frac{1}{|D|} \sum_{j=1}^{|D|} \frac{1}{m_j} \sum_{k=1}^{m_j} \text{Precision}(R_{jk}) \tag{5.1}$$

where D is the set of documents which contain a keyword and R_{jk} is the set of ranked document results from the top until document d_k is reached - m_j such documents are returned by the system. Precision(R_{jk}) is computed as the percent of relevant documents in R_{jk} divided by the size of R_{jk}. The expression computes the precision at each recall point of the documents in D, approximating the area under the curve of a recall/precision plot, with unreturned word tokens given a precision of 0. Multiple queries are equally weighted, thus the total MAP score for a set of queries is simply the average over all queries. Higher MAP scores are better, with a minimum of zero and maximum of one.

The typical speech metrics such as WER corresponds to only one operation point on the recall/precision curve since it does not consider results deeper in the lattice. Thus it is not a well suited metric for search performance. Nonetheless,this chapter reports transcription performance by measuring WER on just the test queries, which is labeled Keyword Word Error Rate (KWER). Other search metrics such as Actual Term Weighted Value (ATWV) [141] require a hard threshold corresponding to some operational tolerance of false alarms and misses [141]. This tolerance is difficult to estimate without an operational task in hand and thus MAP is the preferred metric. Finally, due to the rarity of the 126 OOT terms, overall search performance on in-training terms was barely effected by the following methods. Thus, this chapter only reports performance on these new OOT terms.

5.3 Establishing Bounds

There are two separate bounds for the proposed methods. The first is an operational bound using whatever text resources are available for language modeling. This captures the upper bound available to an engineer and places the semi-supervised methods in the wider perspective of other possible methods for improving the language model. The second is the performance bound on the semi-supervised method when compared against manual transcriptions of the same audio. This perspective on semi-supervised performance is more of a diagnostic as to how well the algorithms exploit the available information.

Knowledge of the two bounds help answer where future work should focus when the semi-supervised gain is modest. If semi-supervised estimation attains the supervised (manually transcribed) upper-bound, then it is effective ; else there is room for additional algorithmic improvement. If the gain for both methods (semi-supervised and fully supervised) are small, then the limited success of semi-supervised estimation does not reflect on the quality of the estimation method. If the use of other text resources is more effective than supervised training, then the whole idea of semi-supervised training is moot and one may consider other means of obtaining language modeling text.

5.3.1 Resource Bounds

The lower bound on resources is the 370 hours of manually transcribed Switchboard. The standard technique for OOT words is to add them to the decoder with shared acoustic parameters and unseen probability in the LM. Because the pronunciation is known, the set of acoustic states (quinphones) is extracted and mapped to clustered quinphone states built from Switchboard. The acoustic model need only build word models for these new terms in order to recognize them. Without any evidence of how these new terms occur in language, the language model can only accommodate these new terms through the unseen unigram probability shared among all other unseen words.

Performance on these terms could be improved further by reducing data sparsity with sub-word recognition [43]. The same 370hr corpus is used to derive a sub-word vocabulary of commonly occurring phone sequences. These sequences, typically three or more phones, are designed to represent the phono-tactics of the language and allow for recognition of unseen keywords. During recognition, the sub-word vocabulary plus the original vocabulary from Switchboard was used to index the audio. At search time, a new keyword will not be present in the recognition output. However, since its pronunciation is known, a dynamic programming algorithm aligns the sub-word representation of the keyword with the closest sub-word unit found in the recognition output. Thus these "hybrid" systems can recognize new queries without requiring their recognition as entire words.

However, sub-word search degrades for queries which *do* appear in the initial training. This is due to two factors. First, the language model loses long-span context because it no longer considers the two preceding words (which may span ten or more phones) but instead the two preceding sub-words. Second, precision is lowered because the word length is decreased on average. When a long word is present in the audio, a full word recognizer tends to prefer the long word, since its phonetic sequence is not often confusable with a combination of short words. Additionally, other factors like word insertion penalties and reduced language model likelihood tend to dis-prefer multiple short words when one long word will do. These factors lead to a sub-word system to have higher recall, but lower precision when compared to full word recognition systems.

If available, out of domain data may prove useful, but is constrained by the mismatch to the in-domain corpus and availability in the language of interest. English web data was used as an upper-bound on the potential of out-of-domain text. n-grams were extracted that contain at least one of the keywords from the Google Web N-gram corpus (LDC2006T13) and added them to the Switchboard LM. Thirty billion tokens of indexable text were required to find enough n-gram samples of the terms to raise MAP to 0.60. While this is a valid strategy for English Fisher, only a few dozen of the world's spoken languages have sufficient tokens available on the web.

Finally, if human labor is available, one could directly transcribe in-domain data, though at significant investment. The 1700 hours of Fisher development data were added to the acoustic model. A higher MAP was achieved by only adding the n-grams that contain one of the target keywords to the LM, adding an implicit 'boost' to the LM probabilities of the keywords over those estimated from the entire corpus. Table 5.1 details the gains for additional resources.

Sub-word phonetic recognition (row 2) outperforms full word recognition (row 1) on these OOT terms with the same amount of supervised training data. Since the OOT terms are unseen in language model training, the sub-word recognition system is better able to benefit from parameter sharing. Extracting n-grams from the web helps (row 3), but requires 30B words of search-able text. This amount of indexable data is not available for many domains of interest. Improving both the AM and LM with manually

System	**MAP**	**KWER**
Switchboard LVCSR	.45	77.4
Switchboard Phonetic	.50	-
Google N-grams	.60	59.3
Targeted Trans.	.70	29.3

Table 5.1: *Improving MAP with Additional Resources* - MAP and keyword WER measured on 126 OOT terms. Sub-word phonetic recognition (row 2) outperforms full word recognition (row 1) on these OOT terms with the same amount of supervised training data. Extracting n-grams from the web helps (row 3), but requires 30B words of search-able text. Improving both the AM and LM with manually transcribed samples from the Fisher development corpus (row 4) makes these words now fully in-training.

transcribed samples from the Fisher development corpus (row 4) makes these words now fully in-training. This is the upper bound in terms of resource: a copious amount of in-domain transcription.

5.3.2 Semi-Supervised Bounds

Semi-supervised methods should be measured against the gain for supervised labels of the same data. Knowledge of this upper bound indicates the head room available for the semi-supervised techniques. Table 5.2 breaks down the gain for manual transcription of Fisher when improving either the AM or LM.

The acoustic model is able to cluster novel quinphone states for these OOT terms thanks to knowledge of phonetic questions. This parameter sharing means that the baseline Switchboard acoustic model has a reasonable estimate for the acoustic states in the novel terms. The language model benefits from manual samples more than the acoustic model due to a lack of parameter sharing. The n-gram language model does not benefit from this

	Switchboard LM	**Improved LM**
Switchboard AM	.45	.65
Improved AM	.59	.70

Table 5.2: *Value of In-Domain Transcripts* - MAP measured on test queries. Manually transcribed utterances from Fisher that contained the 126 terms were added to either the AM or the LM trained on Switchboard. The LM benefits from manual samples (top right) much more than the AM (bottom left).

parameter sharing. Since these keywords are completely unseen in training contexts, any likelihood calculation will back-off to a unigram probability. Furthermore, their unigram frequency estimate is identically computed as the unseen probability. The acoustic model degrades gracefully thanks to knowledge beyond lexical identity: phonetic similarity. The language model does not have such information available about semantic identity.

Since established semi-supervised methods exist for the acoustic model [85] and there is a larger possible improvement for the language model, this chapter now changes focus to semi-supervised methods of language modeling. Therefore, the upper bound is not 0.7 MAP, but 0.65 MAP and the acoustic model will be fixed to the 370 hours of Switchboard for all further experiments.

5.4 Unigram Probability Clamping

Once a user searches for a query, it becomes more important than the other unseen words with uniform back-off probability in the language model. But lacking any evidence, to what value should the probability be raised?

One straightforward approach is to arbitrarily clamp or 'boost' the unigram probabilities of the terms. This requires the user to specify a frequency of the class of words as a whole. In a back-off language model, this is carried out by increasing the unigram probability for the class and re-normalizing the other unigrams. A log-linear language model can also incorporate this constraint by adding a class feature and manual setting of the class frequency as in Section 4.8.

Tuned on the eval set, the optimal clamp raised the MAP score for the 126 keywords from the Switchboard baseline of 0.45 to 0.56. This method is fairly robust to specification of the unigram clamp. Figure 5.3 details the gain in MAP as a function of the total unigram frequency of three different classes in the test data. While fairly stable, the optimal boost is different for each class and correlated with true frequency of the set of keywords in the test set.

As the class frequency rises, recall increases at the cost of lower precision. See Figures 5.4 and 5.5. These opposite trends arise from the implicit 'boost' to unigram probability, resulting in a larger set of returned documents. Recall can only increase with a higher clamp (and does so), but precision degrades. Thus the optimal unigram clamp is a trade-off of these two rising and falling metrics. MAP captures the trade-off between the two, where at higher clamps the further gains in recall are outweighed by the loss in precision.

In addition to requiring development data, this solution does not allow for different unigram frequencies per keyword. It assigns the same unigram boost for all terms, despite drastically different empirical unigram frequencies as in Figure 5.6. The 126 keywords are plotted according to their true frequency in their held-out data. Unigram boosting (red

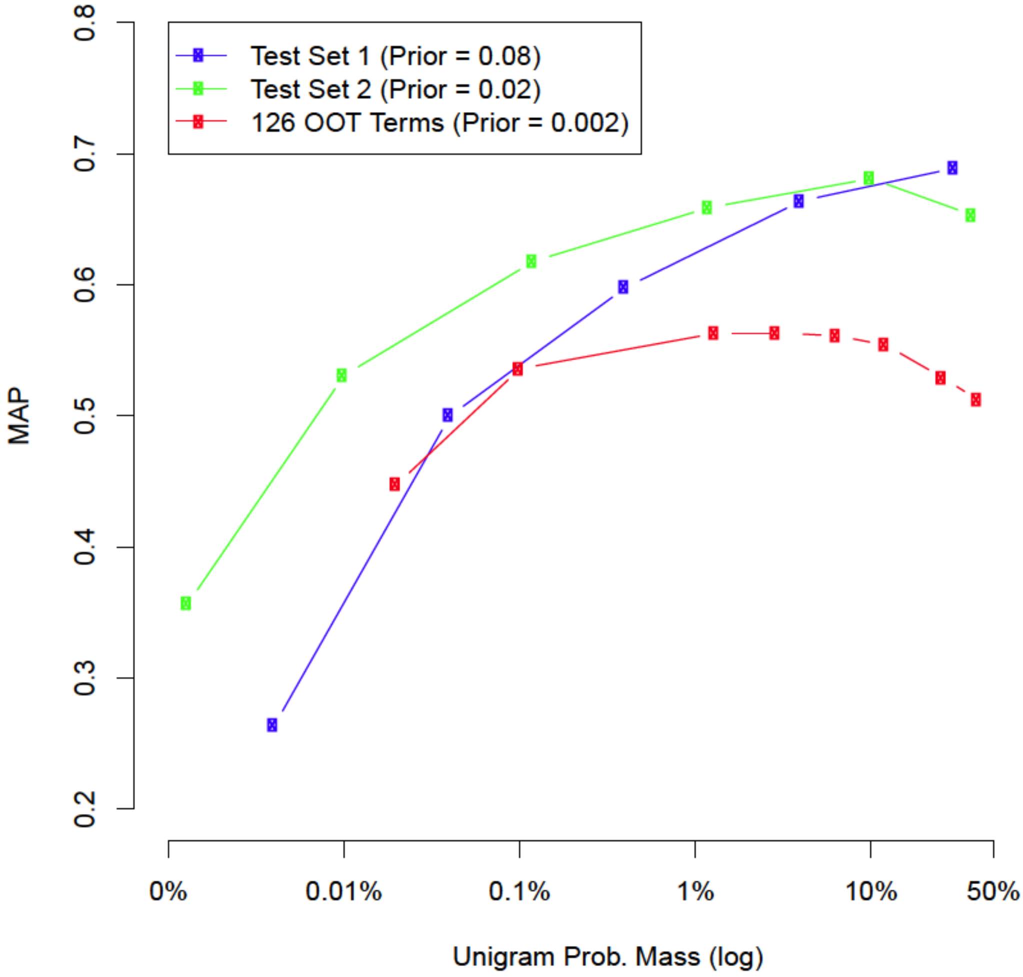

Figure 5.3: Impact of Unigram Clamping on MAP

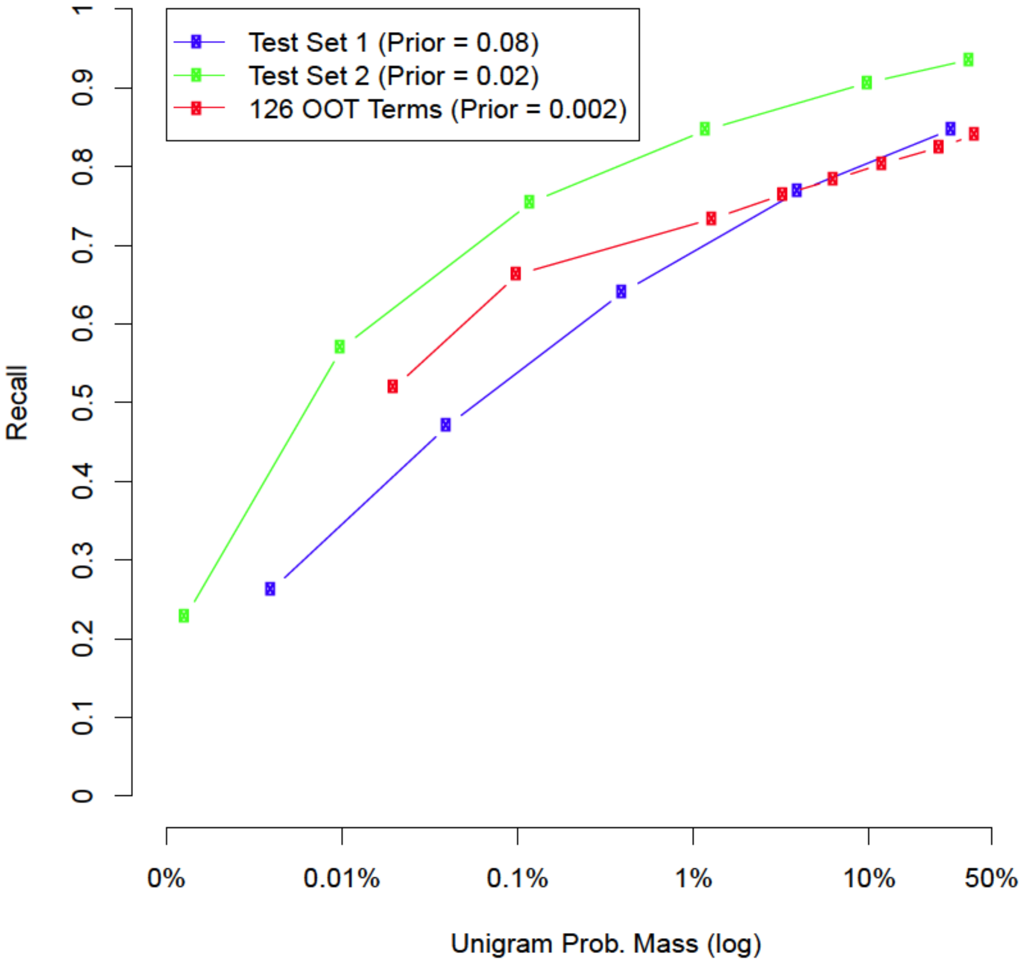

Figure 5.4: Impact of Unigram Clamping on Recall

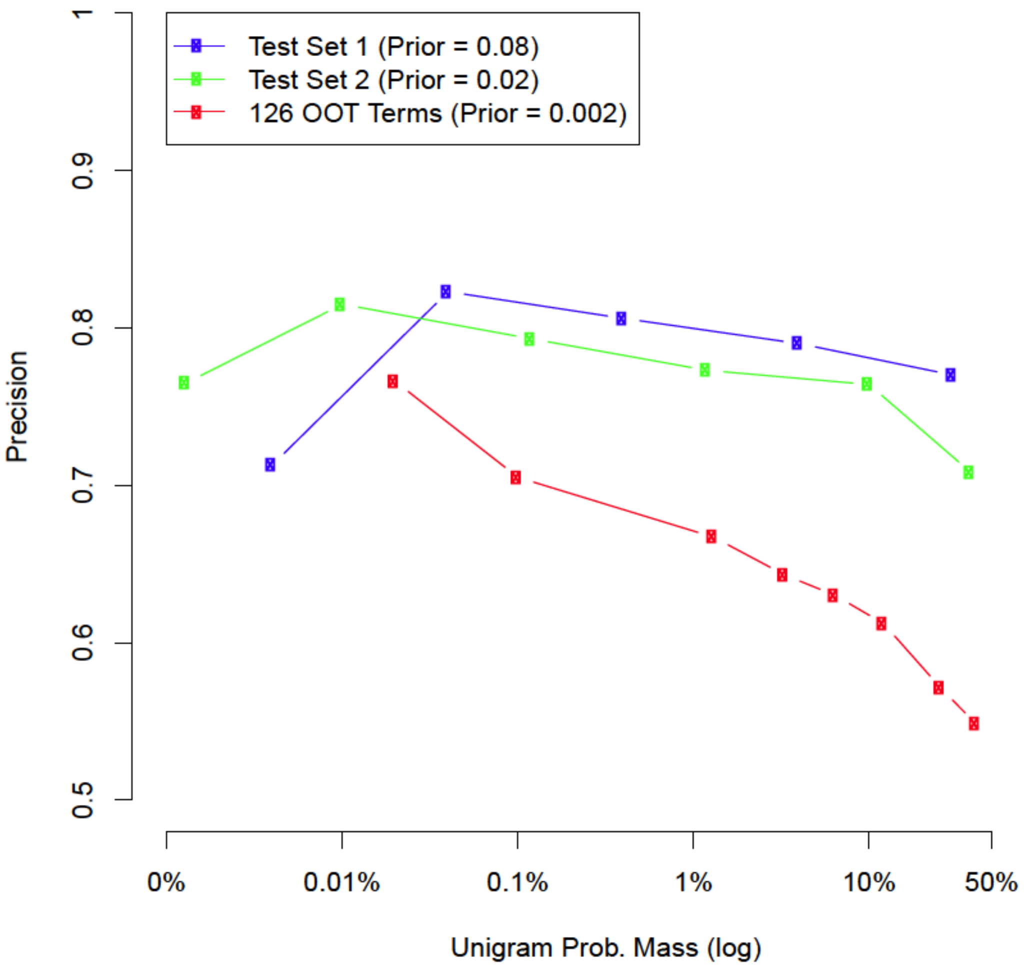

Figure 5.5: Impact of Unigram Clamping on Precision

line) can only approximate this distribution with a horizontal line.

Per-term unigram clamping could do significantly better, raising MAP from 0.56 to 0.61. But now, instead of one parameter, there are 126 to estimate without any data to extract statistics. Even worse, the optimal per-term clamp had no relation to the frequency of the terms in either the reference or the ASR output.

Finally, as the number of target keywords increases (126 in this work) the impact of language model clamping will decrease. In the limit, clamping would provide no benefit if all words in the vocabulary were targeted.

5.5 Semi-Supervised Language Modeling

The task of this chapter differs from previous ones in two important ways. This chapter aims to increase *search* performance (MAP) over a *subset* of the vocabulary (OOT keywords). With this goal in mind, this section considered two extensions of the standard recipe from Chapter 4. First, the 1700 hours of Fisher was decoded with the 370 hour Switchboard system which included the terms in the decoding vocabulary. When measured on the 126 keywords alone, the system had a baseline WER of 77.4% and a MAP of 0.45. The overall WER across all words (in-training or out) was 35.1%.

n-gram extraction methods were contrasted along two dimensions. First, extracting all n-grams in the 1700 hours of audio or only those that contained one of the 126 keywords. Second, extracting from just the one-best or alternate hypotheses.

Extracting all n-grams (regardless of whether they contained a keyword) proceeded as detailed in Chapter 4. Each instance of an n-gram in the output was weighted by the

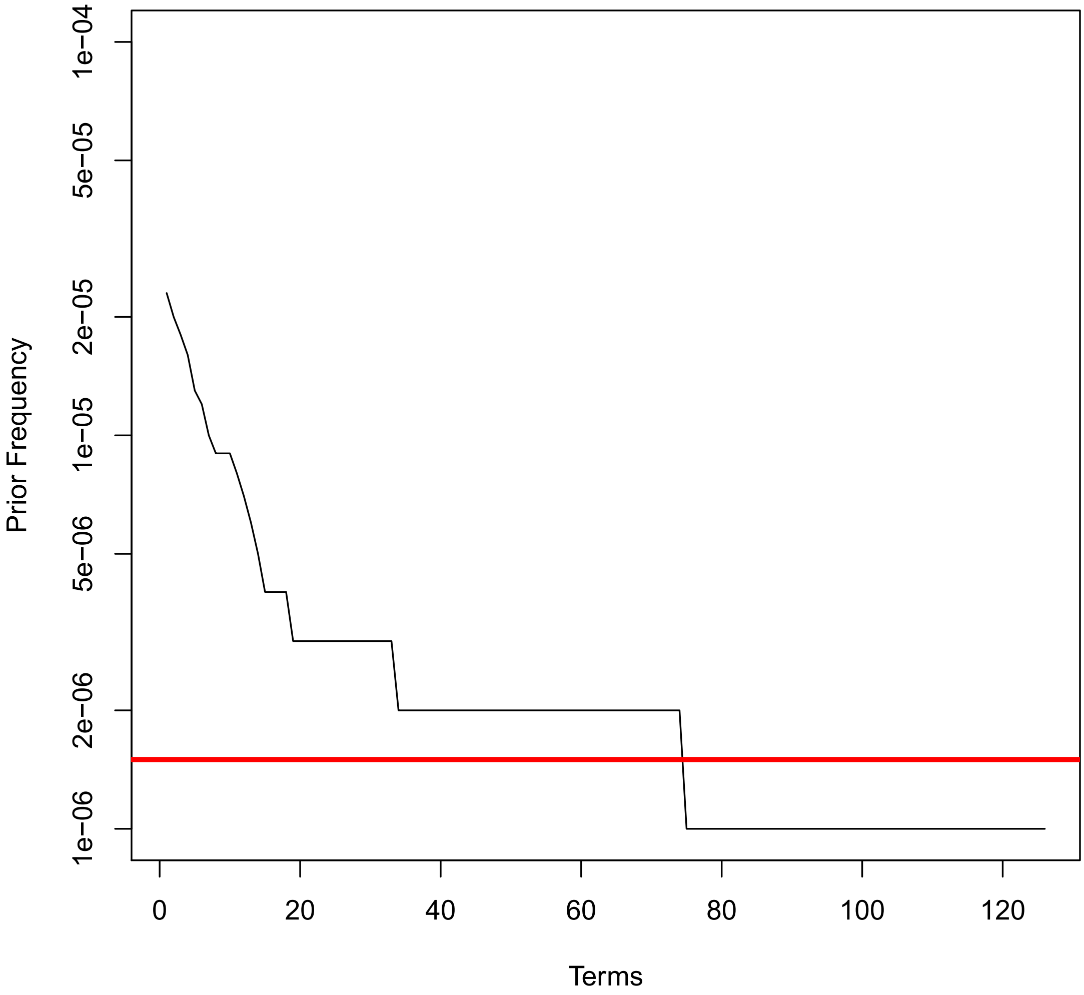

Figure 5.6: *Visual Representation of Unigram Clamping* - The 126 keywords are ranked by their true frequency in the test data (black line). Unigram clamping can at best approximate this Zipfian curve with a straight line (red). Modifying the unigram clamp can only move the red line up or down. A better per-term method would allow for more subtle effects.

product of the word posteriors whether from the one best or deeper in the lattice. Due to the relatively low WER of 35% (compared to 50%+ from most earlier work) the lattices were pruned fairly tightly. Experiments with multiple pruning depths did not see a significant change in overall performance.

Extraction of keyword specific n-grams differed from all n-grams: only the five word neighborhoods around a keyword were considered. Figure 5.7 shows the neighborhood for the keyword NEMO. The left and right contexts were always fixed to the one-best (at a 35% WER). In one method, n-grams were extracted only if the keyword was in the one-best as in Figure 5.7. The second method considered neighborhoods where the keyword was deeper than the one-best as in Figure 5.8 but still present in the confusion network. This method "bubbled up" the keyword to replace the one-best, but kept its posterior score fixed at the original value.

After each of the two methods of selection, all n-gram tokens that contained the keyword were extracted: the one unigram count, the two bigrams and three trigrams. These posterior-weighted tokens were then summed over the 1700 hours to give a posterior-weighted count of an n-gram. Note that n-grams were added where a new keyword occurs in the context, not just as the dependent word. This is because the language model is used for whole utterance decoding, not jus scoring of keywords.

Table 5.3 details the results measuring MAP on the keywords, Keyword WER and average WER. Estimating LM probabilities from automatic transcripts of the 1700 hours improves over both the LVCSR baseline (row 1) and optimal clamping tuned with development data (row 2). Extracting all n-grams from the 1-best (row 3) is strictly better

	Language Model Method	**MAP**	**KWER**	**WER**
1.	Switchboard Baseline	.449	77.4	35.1
2.	Optimal Unigram Clamping	.561	72.1	36.6
3.	All n-grams from 1-best	.542	63.6	35.6
4.	All n-grams from lattice	.540	65.6	37.4
5.	Term n-grams from 1-best	.565	**63.2**	36.6
6.	Term n-grams from c. net	.581	85.9	36.6
7.	2-pass (row 5 then 6)	**.589**	93.0	36.9
8.	Swbd+Fisher Reference	.648	53.5	34.9

Table 5.3: *Semi-Supervised Language Modeling* - Estimating LM probabilities from automatic transcripts of the 1700 hours improves over both the LVCSR baseline (row 1) and optimal clamping tuned with development data (row 2). Extracting all n-grams from the 1-best (row 3) is strictly better than using lattices (row 4). Adding only n-grams that contain a term from the 1-best (row 5) improves MAP and KWER. Going deeper than the 1-best for terms (row 6) improves MAP at significant cost to KWER. Iterating the term 1-best method, re-decoding, and then using the term c.net method (row 7) outperforms both, achieving 70% of the possible supervised gain in MAP (row 8).

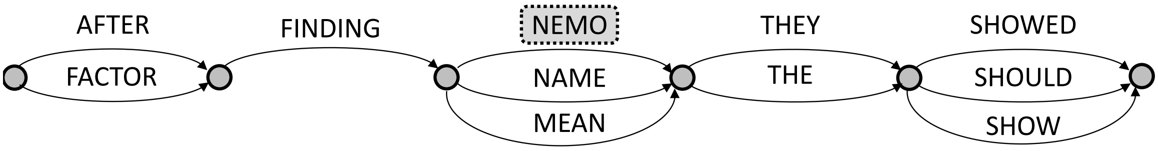

Figure 5.7: Five word neighborhood around keyword Nemo in the 1-best

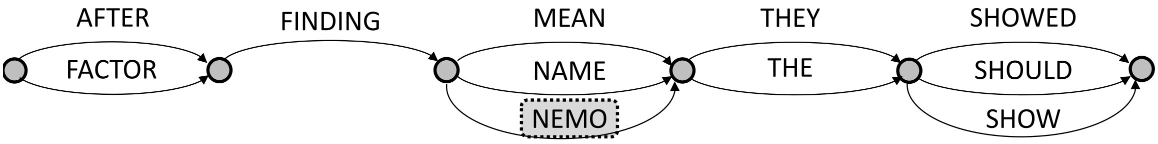

Figure 5.8: Example of Nemo not in 1-best

than using lattices (row 4) across all metrics. This is line with previous results for semi-supervised language modeling: the lattice does not offer any additional value. Adding only n-grams that contain a term from the 1-best (row 5) improves MAP and KWER. Going deeper than the 1-best for terms (row 6) improves MAP at significant cost to KWER. Iterating the term 1-best method, re-decoding, and then using the term c.net method (row 7) outperforms both, achieving 70% of the possible supervised gain in MAP (row 8).

Contrast the performance difference when using lattice counts for *all* words (rows 3 and 4) versus just new keywords (rows 5 and 6). When extracting all possible n-grams from a lattice, the 1-best is overwhelmed by the high generation of novel n-grams deeper down. However, the methods in this chapter gingerly “dip their toe” into the lattice by allowing instances of only the keywords.

Adding all occurrences of a term (method 6 in Table 5.3) seen in the automatic output raises the probability in more contexts. This leads to higher recall, but at some cost of precision: a net gain for MAP. The 'tail' of the recall/precision curve gains more than the 'head' loses. WER is one operating point near the head and thus is hurt, but benefits from the more conservative strategy of extracting n-grams from the 1-best (row 5).

Combinations of unigram clamping, count thresholding and semi-supervised learning were attempted. The optimal method was two passes of semi-sup training (method 7 in Table 5.3). This combination gives 70% of the total possible supervised gain without requiring parameters to tune, external resources or human transcription.

5.6 Directed Transcription

If manual labor is available for improving performance, how should one most efficiently use that effort? If all words mattered equally, the standard approach would be to transcribe representative speech from the domain. But in many applications, only the test queries matter and they are rare: with a corpus frequency of 0.03% in this chapter. Only one out of 3333 words transcribed would be useful for improving language model performance on the test queries. A method to 'direct' the transcriber where to look for speech containing such words could therefore greatly improve cost effectiveness.

To increase the precision of transcription, the semi-supervised techniques from the previous section were applied first. This results in substantial improvements in the quality of search results. After automatic indexing, a set of search results with posterior probabilities were generated. Given a list of putative results from a KWS system, this work considered three choices a transcriber could make:

1. Examine most likely vs. least likely results first, the former yields true examples while the latter helps to reduce false alarms.

2. Only verify whether a putative result is correct (removing incorrect results from the LM) vs. additionally transcribe the five word window containing the correct results.

3. Once the budget is used up, train the LM on just the verified/transcribed n-grams vs. also add the remainder of the automatically generated n-grams.

Of the eight possible combinations,this work simulated the five plausible choices. First, the 1700 hours of Fisher was decoded with the Switchboard system. Then, added to the Switchboard LM were n-grams from the 1-best transcript that included one of the OOT terms (method 5 in Table 5.3). Then the audio was re-decoded and extracted all instances of the terms in the resulting confusion networks. The recovered speech segments had a 5% precision and 75% recall – compared to the 0.03% precision and 100% recall of undirected transcription. Since the reference transcripts were available, it was straightforward to simulate human labor. A wide range of available hum effort was considered, ranging from 10 hours of work to 270, the maximum needed to process all found results.

Five different methods of transcription were simulated, depending on three choices: 1) Verifying vs. transcribing 2) examining most likely vs. least likely results first and 3) including unsupervised n-grams or not. For each transcription method,a new language model was trained and used to re-decode the heldout test set and finally report gain in MAP in Figure 5.9.

The best use of human effort depends on the amount of total effort available. Another round of semi-supervised training (requiring no effort) is worth 30 hours of directed transcription. (Second Pass ST-LM). Manually removing incorrect automatic n-grams is most effective up to 40 hours (solid line w/filled squares). Past that point, it is best to manually transcribe the most likely n-grams, ignoring the remaining automatic n-grams (solid line w/no squares). Note that directed transcription cannot achieve the gain from transcribing all 1700 hours since recall is not 100% for the semi-supervised methods. Verifying the most likely n-grams is most effective given one work week of effort. Past that, it is best to ignore the automatic n-grams and directly transcribe the most likely results first. Transcribing all of the 1700 hours would require 200 months of effort and improve MAP to 0.65. Semi-supervised learning combined with directed transcription reaches 0.64 MAP at a cost of 1.5 months - nearly identical search performance at 1/125th the cost.

5.7 Discussion

Semi-supervised language modeling improves keyword search performance significantly on words not seen in the training data. The technique requires a large amount of un-transcribed speech that contains a handful of instances of the set of words along with their pronunciations. However, no human transcription or additional resources are required to obtain substantial improvement. A two-pass strategy of first improving WER then MAP gives 70% of the possible gain for manual transcription. Directed learning can then efficiently close the remaining gap at a small fraction of human effort. The success of this work contrasts with that of previous semi-supervised language modeling results due to two factors: the focus on a small subset of terms and a different metric which favors recall.

A number of issues arise when applying this work to other tasks. First,only those

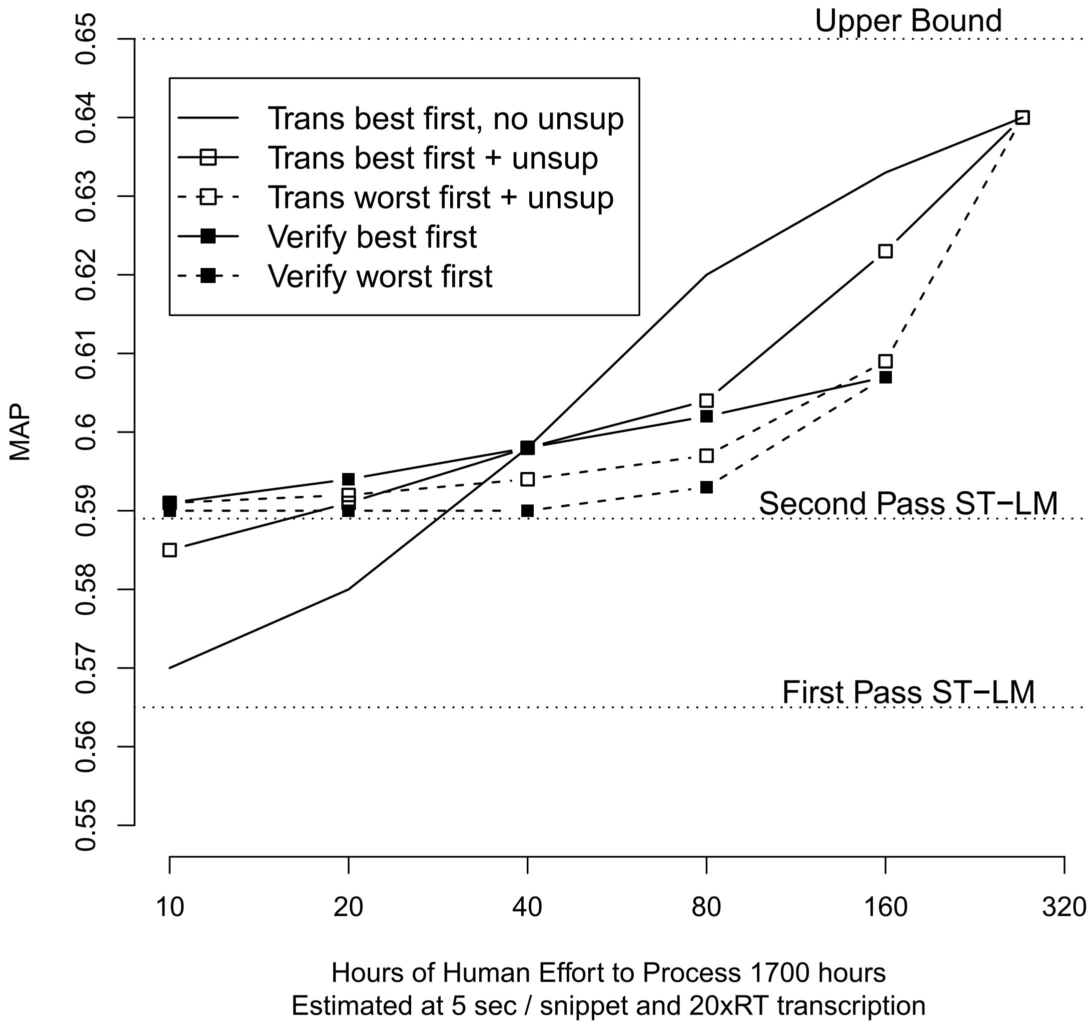

Figure 5.9: *Comparisons of Directed Transcription* - Covering the remaining gap from semi-supervised to fully supervised requires 320 hours of manual effort. With a more constrained budget, it is best to have a human only verify results. If more effort is available, then it helps to also correct the neighboring words around a result. At all times it is best to consider the most likely results first.

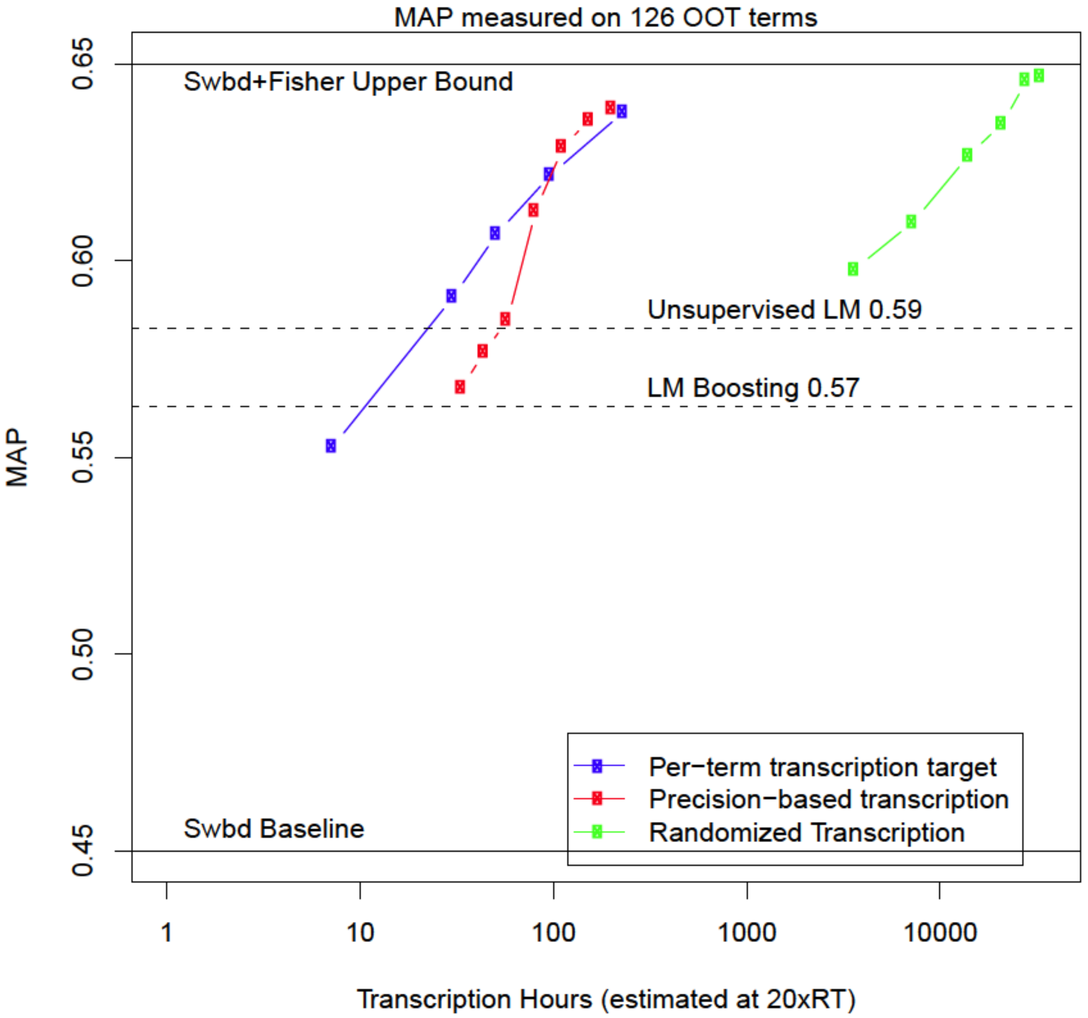

Figure 5.10: *Cost Effectiveness of Directed Transcription* - Randomly transcribing the 1700 hours of unlabeled audio (green line) is orders of magnitudes more expensive when computing MAP performance *on the 126 OOT terms*. Since they are relatively rare, it is much more efficient to first perform semi-supervised language modeling and then generate a list of results for manual transcription. It is best to transcribe the most likely results first and so one might wonder if only the easier to score keywords of the 126 terms might be transcribed. However, whether each keyword is allocated the same amount of transcription effort (blue line) or the entire group is treated the same (red line), there is not a significant difference in overall keyword performance.

keywords with zero training examples were considered. While this target task is relevant for many applications, it is the lowest possible bound for semi-supervised performance. Future work should consider the benefit of semi-supervised performance with increasing amounts of training data. The Bayesian methods of Chapter 4 would be well suited to capitalize on a small amount of examples in conjunction with a large number of automatic samples.

Second, this chapter did not utilize *negative* training samples provided by directed transcription. When the simulated user marked a putative n-gram sample as incorrect, it was simply discarded. But the act of discarding provides actionable intelligence. If a putative sample is discarded, then the keyword was false alarmed, indicating that the likelihood was too high for that word. Therefore, one could decrease the likelihood of that word in the context, or, failing that, by reducing the unigram probability.

Third, extensions to multi-token queries will require additional effort. Shorter words will have lower initial precision. The length of the keywords in this chapter made them acoustically distant to other terms. This led the acoustic model to prefer the keywords, despite a low language model score, in clear acoustic instances. While the query may never have been seen, the component terms may have training samples, which should be leveraged.

In this chapter, the key lessons of Chapters 3 and 4 are applied to a relevant end task. The directed learning results of this chapter were geared for non-expert transcribers, who could easily verify words. While the previous chapter reported modest WER Recoveries of 7%, this chapter achieved an 80% Recovery by constraining the estimation task.

Chapter 6

Conclusions

Estimating strong statistical language models for automatic speech recognition requires large quantities of training text matched to the target domain. For many domains, such as conversational telephone speech, there is little matched text available from which to estimate a strong language model. Generating in-domain text requires careful transcription of in-domain audio which is typically impractical. One source of matched text is from an inaccurate transcriber – either human or automated – who can inexpensively, but inaccurately, transcribe in-domain audio. Prior work has left the performance of language models to the whim of transcriber accuracy. This dissertation is an investigation of methods to overcome data scarcity for statistical language model estimation. While the domain in this work was English conversational telephone speech, the contributions and established "best practices" will apply to other speech domains and tasks such as machine translation or optical character recognition.

This dissertation makes the following contributions:

- It is the first to demonstrate that quality control of non-expert annotators is unnecessary when the annotations are used for estimating a statistical model. Automatic speech recognizers show little degradation when estimated on non-expert transcripts, despite high disagreement with experts. It also is the first to contrast spending a fixed transcription budget on redundant non-expert transcription (with the goal of improving transcript quality) versus transcribing as much audio as possible without quality control. This dissertation demonstrates that manual labor is best spent collecting more transcripts, not creating better quality ones. Finally, this work proposes a new method of estimating non-expert transcription skill without requiring expert transcriptions.

- It is the first to analyze why the gains of semi-supervised language models are modest, especially in light of successful semi-supervised acoustic modeling. Standard backoff language models depend on accurate highest-order n-gram counts, which an ASR system cannot accurately produce. The dissertation then categorizes systematic errors made by the recognizer and is the first to propose a new class of language modeling features to compensate for these errors. This work then explores semi-supervised estimation of a log-linear model and is the first to demonstrate substantial gains for incorporating the newly proposed features.

- It is the first apply semi-supervised language modeling to the task of spoken term detection with substantial improvements in search performance. This is in marked contrast to the modest impact of semi-supervised language modeling in the literature.

Furthermore, it is the first to propose a method of directed transcription which uses an ASR system to suggest audio segments in a corpus likely to contain a keyword to a transcriber. This new method achieves the same improvement as transcribing the entire audio corpus at several orders of magnitude savings in human labor.

Promising directions for further study should focus on extracting *reliable* and *informative* statistics from noisy data. While language models benefit the most from n-grams, the log-linear framework can flexibly incorporate a variety of domain knowledge. If knowledge can be expressed as a mapping from observation to real value and constrained as an expected count, it can fit in a log-linear language model. Coarser features such as part of speech frequencies, n-gram classes and more may be more reliably estimated from inaccurate transcripts. Additional corrective features to compensate for transcriber inaccuracies would also be helpful. The features used in Chapter 4 were discovered through experience. Future work could apply the relevance metric suggested in Section 4.8.2 to discover groups of words which are systematically over or under-produced in an agglomerative hierarchy.

Extending to low-resource languages will pose further issues. The starting point is not 2,000 hours of transcribed speech, but true low-resource domains. Highly inflected languages will suffer much higher out of vocabulary rates and require a much larger vocabulary and lead to very sparse language model training data. This dissertation also did not consider pronunciation, assuming for this dissertation that accurate pronunciations for every word is available. The reality for many vocabularies is that many rare words - the very topical words of interest in Chapter 5 - pronunciations are automatically derived from orthography. Many low-resource languages use graphemic pronunciations which are grossly mismatched. Linguists will be crucial to enumerate phonetic inventories, provide pronunciations and crucially, interpret transcription errors for insight into systematic problems.

Bootstrapping to new domains within a language poses vocabulary problems as well. Acoustically distinct domains will suffer worse automatic transcription accuracy due to poorly matched acoustic models. Semantically distinct domains will differ in terms of vocabulary and relative word use. Semi-supervised language model estimation holds some promise here as it can estimate the differing word statistics if transcription accuracy is high enough. Domains within a language may also be isolated from any high-resource domains. As noted in the introduction, conversational Arabic is distinct enough from broadcast news that language model adaptation fails. However, by backing off to coarse statistics such as parts of speech, it may be possible to exploit multi-lingual information across languages.

Instead of focusing on semi-supervised methods, future work could instead consider more efficient uses of human labor. A transcription budget of ten hours of transcripts may instead be spent on five hours of transcription and five hours of user corrections, part of speech tagging, or some other more efficient use. As mentioned earlier, low-resource language modeling really means many weak signals. The type of signal will depend heavily on the domain and it is up to the domain expert and language model scientist to experiment with features which are robust to noise as well as informative.

Bibliography

[1] J. Ma and R. Schwartz, "Unsupervised versus Supervised Training of Acoustic Models," in *Proceedings of Annual Conference of the International Speech Communication Association*, 2008, pp. 2374–2377.

[2] H. Kucera and W. N. Francis, *Computational analysis of present-day American English.* Providence, RI: Brown University Press, 1967.

[3] O. Kimball, C.-L. Kao, T. Arvizo, J. Makhoul, and R. Iyer, "Quick Transcription and Automatic Segmentation of the Fisher Conversational Telephone Speech Corpus," in *Proceedings of 2004 Rich Transcriptions Workshop*, 2004.

[4] F. Biadsy, P. Moreno, and M. Jansche, "Google's Cross-Dialect Arabic Voice Search," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2012, pp. 4441–4444.

[5] Y.-H. Sung, M. Jansche, and P. Moreno, "Deploying Google Search by Voice in Cantonese," in *Proceedings of Annual Conference of the International Speech Communication Association*, 2011, pp. 2865–2868.

[6] F. Jelinek, L. Bahl, and R. Mercer, "Design of a linguistic statistical decoder for

the recognition of continuous speech," in *Information Theory, IEEE Transactions on*, vol. 21, no. 3. Piscataway, NJ, USA: IEEE Press, Sep. 2006, pp. 250–256.

[7] S. B. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," in *Readings in speech recognition.* Morgan Kaufmann Publishers Inc., 1990, pp. 65–74.

[8] N. Kumar, "Investigation of silicon auditory models and generalization of linear discriminant analysis for improved speech recognition," Ph.D. dissertation, 1997.

[9] E. Eide and H. Gish, "A parametric approach to vocal tract length normalization," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, 1996, pp. 346–348.

[10] S. Thomas, P. Nguyen, G. Zweig, and H. Hermansky, "MLP based phoneme detectors for Automatic Speech Recognition," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2011, pp. 5024–5027.

[11] W. Reichl and W. Chou, "Decision tree state tying based on segmental clustering for acoustic modeling," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, 1998, pp. 801–804.

[12] J. Sohn, N. S. Kim, and W. Sung, "A statistical model-based voice activity detection," in *Signal Processing Letters, IEEE*, vol. 6, no. 1, 1999, pp. 1–3.

[13] L. Baum and T. Petrie, "Statistical inference for probabilistic functions of finite state Markov chains," in *Annals of Mathematics and Statistics*, vol. 33, 1966, pp. 1554–1563.

[14] D. Povey, D. Kanevsky, B. Kingsbury, B. Ramabhadran, G. Saon, and K. Visweswariah, "Boosted MMI for model and feature-space discriminative training," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2008, pp. 4057–4060.

[15] G. Zweig, P. Nguyen, D. V. Compernolle, K. Demuynck, L. E. Atlas, P. Clark, G. Sell, M. Wang, F. Sha, H. Hermansky, D. Karakos, A. Jansen, S. Thomas, S. G. S. V. S., S. Bowman, and J. T. Kao, "Speech recognition with segmental conditional random fields: A summary of the JHU CLSP 2010 Summer Workshop," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2011, pp. 5044–5047.

[16] G. Hinton, L. Deng, D. Yu, G. Dahl, A.-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep Neural Networks for Acoustic Modeling in Speech Recognition," in *Signal Processing Magazine*, 2012.

[17] S. Austin, R. Schwartz, and P. Placeway, "The forward-backward search algorithm," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, 1991, pp. 697–700.

[18] M. J. Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," in *Computer speech & language*, vol. 12, no. 2. Elsevier, 1998, pp. 75–98.

[19] R. Prasad, S. Matsoukas, C. Kao, J. Ma, D. Xu, T. Colthurst, O. Kimball, R. Schwartz, J. Gauvain, L. Lamel *et al.*, "The 2004 BBN/LIMSI 20xRT English

conversational telephone speech recognition system," in *Proceedings of Annual Conference of the International Speech Communication Association*, 2005, pp. 1645–1648.

[20] T. Brants, A. C. Popat, and F. J. Och, "Large Language Models in Machine Translation," in *Computational Linguistics*, vol. 1, no. June. Google Patents, 2007, pp. 858–867.

[21] N. Hibash, "On Arabic and its Dialects," in *Multilingual Magazine #81*, vol. 17, no. 5, 2006.

[22] C. Allauzen and M. Riley, "Bayesian Language Model Interpolation for Mobile Speech Input," in *Proceedings of Annual Conference of the International Speech Communication Association*, 2012, pp. 1429–1432.

[23] G. Lidstone, "Note on the general case of the Bayes-Laplace formula for inductive or a posteriori probabilities," in *Transactions of the Faculty of Actuaries*, no. 8, 1920, pp. 182–192.

[24] R. Kneser and H. Ney, "Improving backing-off for m-gram language modeling," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, 1995, pp. 181–184.

[25] I. Good, "The population frequencies of species and the estimation of population parameters," in *Biometrika*, vol. 40, no. 3 and 4, 1953, pp. 237–264.

[26] F. Jelinek and R. L. Mercer, "Interpolated estimation of Markov source parameters from sparse data," in *Proceedings of the Workshop on Pattern Recognition in Practice*, 1980.

[27] I. H. Witten and T. C. Bell, "The zero-frequency problem: estimating the probabilities of novel events in adaptive text compression," in *IEEE Transactions on Information Theory*, vol. 37, no. 4. IEEE, 1991, pp. 1085–1094.

[28] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," in *Proceedings of the annual meeting on Association for Computational Linguistics*, 1996.

[29] Y. Teh, "A Hierarchical Bayesian Language Model based on Pitman-Yor Processes," in *Proceedings of the annual meeting on Association for Computational Linguistics*, vol. 44, no. July. ASSOC COMPUTATIONAL LINGUISTICS-ACL, 2006, pp. 985–992.

[30] S. Arora and S. Agarwal, "Active Learning for Natural Language Processing," Carnegie Mellon LTI, Tech. Rep., 2012.

[31] S. Dasgupta and J. Langford, "A tutorial on active learning," in *Proceedings of The International Conference on Machine Learning*, 2009.

[32] D. Hakkani-Tur, G. Riccardi, and A. Gorin, "Active Learning For Automatic Speech Recognition," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2002, pp. 3904–3907.

[33] T. M. Kamm and G. G. L. Meyer, "Automatic Selection of Transcribed Training Material," in *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding*, 2001, pp. 417–20.

[34] C. Cieri, D. Miller, and K. Walker, "The Fisher Corpus : a Resource for the Next Generations of Speech-to-Text," in *Proceedings of the 4th International Conference on Language Resources and Evaluation*, 2004, pp. 69–71.

[35] P. Woodland and H. Chan, "Some Results on CTS Quick Transcription and Fisher Data," Cambridge University, Tech. Rep., 2003.

[36] C. Callison-Burch, "Fast, Cheap, and Creative : Evaluating Translation Quality Using Amazons Mechanical Turk," in *Language and Speech*, vol. 1, no. August. Association for Computational Linguistics, 2009, pp. 286–295.

[37] R. Snow, B. O'Connor, D. Jurafsky, and A. Y. Ng, "Cheap and fast–but is it good?: evaluating non-expert annotations for natural language tasks," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2008, pp. 254–263.

[38] O. F. Zaidan and C. Callison-Burch, "Crowdsourcing translation: professional quality from non-professionals," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, ser. HLT '11. Stroudsburg, PA, USA: Association for Computational Linguistics, 2011, pp. 1220–1229.

[39] R. Zbib, E. Malchiodi, J. Devlin, D. Stallard, S. Matsoukas, R. Schwartz, J. Makhoul, O. F. Zaidan, and C. Callison-Burch, "Machine translation of arabic dialects," in *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, ser. NAACL HLT '12. Stroudsburg, PA, USA: Association for Computational Linguistics, 2012, pp. 49–59.

[40] L. Von Ahn and L. Dabbish, "Labeling images with a computer game," in *Proceedings of the 2004 conference on Human factors in computing systems CHI 04*, ser. CHI '04, E. Dykstra-Erickson and M. Tscheligi, Eds., vol. 6, no. 1, School of Computer Scienc Carnegie Mellon Uni. ACM Press, 2004, pp. 319–326.

[41] C. Auzanne, J. S. Garofolo, J. G. Fiscus, and W. Fisher, "Automatic language model adaptation for spoken document retrieval," in *Proceedings of RIAO 2000 Conference on Content-Based Multimedia Information Access*, 2000, pp. 132–141.

[42] A. Allauzen and J.-L. Gauvain, "Open Vocabulary ASR for Audiovisual Document Indexation," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2005, pp. 1013–1016.

[43] I. Bulyiko, O. Kimball, M.-H. Siu, J. Herrero, and D. Blum, "Detection of unseen words in conversational Mandarin," in *Proceedings of Annual Conference of the International Speech Communication Association*, 2012, pp. 5181–5184.

[44] G. Neubig, M. Mimura, S. Mori, and T. Kawahara, "Bayesian Learning of a Language Model from Continuous Speech," in *IEICE Transactions on Information and Systems*, vol. E95.D, no. 2, 2012, pp. 614–625.

[45] M. Bacchiani and B. Roark, "Unsupervised Language Model Adaptation," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2003, pp. 220–224.

[46] J. R. Finkel and C. D. Manning, "Hierarchical Bayesian domain adaptation," in *Proceedings of the Annual Conference of the North American Chapter of the Association*

for Computational Linguistics, no. June. Association for Computational Linguistics, 2009, pp. 602–610.

[47] K. Ries, “A Class Based Approach to Domain Adaptation and Constraint Integration for Empirical M-Gram Models,” in *Proceedings of Annual Conference of the International Speech Communication Association*, 1997, pp. 1983–1986.

[48] S. Huang and S. Renals, “Unsupervised Language Model Adaptation Based on Topic and Role Information in Multiparty Meetings,” in *Proceedings of Annual Conference of the International Speech Communication Association*, vol. 1, 2008, pp. 833–836.

[49] R. Lau, R. Rosenfeld, and S. Roukos, “Trigger-based language models: a maximum entropy approach,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2. Ieee, 1993, pp. 45–48.

[50] A. Emami and F. Jelinek, “A Neural Syntactic Language Model,” in *Machine Learning*, vol. 60, no. 1-3. Springer, 2005, pp. 195–227.

[51] H. Wallach, “Structured Topic Models for Language,” Ph.D. dissertation, University of Cambridge, 2008.

[52] H. Schwenk, “Continuous space language models,” in *Computer Speech and Language*, vol. 21, no. 3. Elsevier, 2007, pp. 492–518.

[53] T. Mikolov, S. Kombrink, L. Burget, J. Cernocky, and S. Khudanpur, “Extensions of recurrent neural network language model,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2011, pp. 5528–5531.

[54] P. Brown, P. DeSouza, R. L. Mercer, V. Della Pietra, and J. Lai, “Class-based n-gram models of natural language,” in *Computational Linguistics*, vol. 18, no. 1950. MIT Press, 1992, pp. 467–479.

[55] S. F. Chen, “Shrinking exponential language models,” in *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Morristown, NJ, USA: Association for Computational Linguistics, 2009, pp. 468–476.

[56] R. Malouf, “A comparison of algorithms for maximum entropy parameter estimation,” in *Proceedings of the 6th conference on Natural Language Learning*, ser. COLING-02, vol. 20, 2002, pp. 1–7.

[57] S. F. Chen and R. Rosenfeld, “A gaussian prior for smoothing maximum entropy models,” DTIC Document, Tech. Rep., 1999.

[58] O. Chapelle, B. Scholkopf, and A. Zien, *Semi-Supervised Learning*, 1st ed. MIT Press, 2006.

[59] A. Dempster, N. Laird, and D. B. Rubin, “Maximum Likelihood from Incomplete Data via the EM Algorithm,” in *Journal of the Royal Statistical Society*, vol. 39, no. 1, 1977, pp. 1–38.

[60] D. Cooper and J. Freeman, “On the asymptotic improvement in the outcome of supervised learning provided by additional nonsupervised learning,” in *IEEE Transactions on Computers*, vol. C-199, no. 11, 1970, pp. 1055–1063.

[61] V. Castelli, "The Relative Value of Labeled and Unlabeled Sample," Ph.D. dissertation, Stanford University, 1994.

[62] J. Scudder, "Probability of error of some adaptive pattern-recognition machines," in *Transactions on Information Theory*, vol. 11, 1965, pp. 363–371.

[63] S. Fralick, "Learning to recognize patterns without a teacher," in *IEEE Transactions on Information Theory*, vol. 22, 1967, pp. 1947–1975.

[64] A. Agrawala, "Learning with a probabilistic teacher," in *IEEE Transactions on Information Theory*, vol. 16, 1970, pp. 373–379.

[65] M. R. Amini and P. Gallinari, "Semi-Supervised logistic regression," in *Proceedings of the European Conference on Artificial Intelligence*, 2002, pp. 390–394.

[66] G. Zavaliagkos and T. Colthurst, "Utilizing untranscribed training data to improve performance," in *DARPA Broadcast News Transcription and Understanding Workshop*, 1998, pp. 301–305.

[67] L. Lamel, J.-L. Gauvain, and G. Adda, "Lightly supervised and unsupervised acoustic model training," in *Computer Speech and Language*, vol. 16, no. 1, 2002, pp. 115–129.

[68] C. Gollan and H. Ney, "Towards Automatic Learning in LVCSR: Rapid Development of a Persian Broadcast Transcription System," in *Proceedings of Annual Conference of the International Speech Communication Association*, 2008, pp. 1441–1444.

[69] F. Wessel and H. Ney, "Unsupervised Training Of Acoustic Models For Large Vo-

cabulary Continuous Speech Recognition," in *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding*, 2001, pp. 23–31.

[70] L. Wang, M. Gales, and P. C. Woodland, "Unsupervised training for Mandarin broadcast news and conversation transcription," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2007, pp. 2–4.

[71] J. Ma and S. Matsoukas, "Unsupervised Training on a Large Amount of Arabic Broadcast News Data," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2007, pp. 349–352.

[72] C. Gollan, S. Hahn, R. Schluter, and H. Ney, "An Improved Method for Unsupervised Training of LVCSR Systems," in *Proceedings of Annual Conference of the International Speech Communication Association*, 2007, pp. 2101–2104.

[73] K. Yu, M. J. F. Gales, and P. C. Woodland, "Unsupervised Training with Directed Manual Transcription for Recognising Mandarin Broadcast Audio," in *Proceedings of Annual Conference of the International Speech Communication Association*, 2007, pp. 1709–1712.

[74] A. Celebi, H. Sak, E. Dikici, M. Sarac, M. Lehr, E. Prud, P. Xu, N. Glenn, D. Karakos, S. Khudanpur, B. Roark, K. Sagae, I. Shafran, D. Bikel, Y. Cao, K. Hall, E. Hasler, P. Koehn, A. Lopez, M. Post, and D. Riley, "Semi-Supervised Discriminative Language Modeling For Turkish ASR," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2012, pp. 5025–5028.

[75] R. Gretter and G. Riccardi, "On-Line Learning Of Language Models With Word Error

Probability Distributions," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2001, pp. 557–560.

[76] M. Nakano and T. J. Hazen, "Using Untranscribed User Utterances for Improving Language Models based on Confidence Scoring," in *Proceedings of Annual Conference of the International Speech Communication Association*, 2003, pp. 417–420.

[77] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Proceedings of the Workshop on Computational Learning Theory*, 1998, pp. 92–100.

[78] D. Pierce and C. Cardie, "Limitations of co-training for natural language learning from large datasets," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2001, pp. 1–9.

[79] M. Collins and Y. Singer, "Unsupervised models for named entity classification," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 1999, pp. 100–110.

[80] W. Wang, Z. Huang, M. Harper, S. R. I. International, M. Park, and W. Lafayette, "Semi-supervised learning for part-of-speech tagging of mandarin transcribed speech," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. IV, 2007, pp. 137–140.

[81] G. Tur, "Co-adaptation: adaptive co-training for semi-supervised learning," in *Proceedings of Annual Conference of the International Speech Communication Association*, 2009, pp. 3721–3724.

[82] C. Callison-Burch and M. Osborne, "Co-training for statistical machine translation," Ph.D. dissertation, Masters thesis, School of Informatics, University of Edinburgh, 2002.

[83] T. Fraga-Silva, V.-B. Le, L. Lamel, and J.-L. Gauvain, "Incorporating MLP features in the unsupervised training process," in *Proceedings of the Workshop on Spoken Languages Technologies for Under-resourced Languages*, 2012.

[84] K. Nigam and R. Ghani, "Understanding the behavior of Co-training," in *Proceedings of the Ninth International Conference on Information and Knowledge Management*, 2000, pp. 15–17.

[85] S. Novotney, R. Schwartz, and J. Ma, "Unsupervised acoustic and language model training with small amounts of labelled data," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2009, pp. 4297–4300.

[86] F. Jelinek, R. L. Mercer, L. R. Bahl, and J. K. Baker, "Perplexity a measure of the difficulty of speech recognition tasks," in *The Journal of the Acoustical Society of America*, vol. 62, 1977, p. S63.

[87] J. Schalkwyk, D. Beeferman, F. Beaufays, B. Byrne, C. Chelba, M. Cohen, M. Kamvar, and B. Strope, "Your Word is my Command: Google Search by Voice: A Case Study," in *Advances in Speech Recognition.* Springer, 2010, pp. 61–90.

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

[89] O. F. Zaidan and C. Callison-Burch, “Feasibility of human-in-the-loop minimum error rate training,” in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2009, pp. 52–61.

[90] I. McGraw, A. Gruenstein, and A. Sutherland, “A self-labeling speech corpus: Collecting spoken words with an online educational game,” in *Proceedings of Annual Conference of the International Speech Communication Association*, 2009, pp. 3031–3034.

[91] L. Gillick, J. Baker, J. Bridle, M. Hunt, Y. Ito, S. Lowe, J. Orloff, B. Peskin, R. Roth, and F. Scattone, “Application of large vocabulary continuous speech recognition to topic and speaker identification using telephone speech,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1993, pp. 471–474.

[92] D. Miller, M. Kleber, C. Kao, O. Kimball, T. Colthurst, S. Lowe, R. Schwartz, and H. Gish, “Rapid and Accurate Spoken Term Detection,” in *Proceedings of Annual Conference of the International Speech Communication Association*, 2007, pp. 314–317.

[93] S. Young, J. Schatzmann, K. Weilhammer, and H. Ye, “The hidden information state approach to dialog management,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2007, pp. IV–149.

[94] M. Marge, S. Banerjee, and A. Rudnicky, “Using the Amazon Mechanical Turk for

Transcription of Spoken Language," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, March 2010, pp. 5270–5273.

[95] M. Post, C. Callison-Burch, and M. Osborne, "Constructing parallel corpora for six Indian languages via crowdsourcing," in *Proceedings of the Seventh Workshop on Statistical Machine Translation*, ser. WMT '12. Stroudsburg, PA, USA: Association for Computational Linguistics, 2012, pp. 401–409.

[96] H. Gelas, A. S. Teferra, L. Besacier, , and F. Pellegrino, "Evaluation of crowdsourcing transcriptions for African languages," in *Proceedings of Conference on Human Language Technology for Development*, 2011, pp. 128–133.

[97] M. Marge, S. Banerjee, and A. I. Rudnicky, "Using the Amazon Mechanical Turk to transcribe and annotate meeting speech for extractive summarization," in *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, ser. CSLDAMT '10. Stroudsburg, PA, USA: Association for Computational Linguistics, 2010, pp. 99–107.

[98] G. Parent and M. Eskenazi, "Toward better crowdsourced transcription: Transcription of a year of the let's go bus information system data," in *Proceedings of the IEEE/ACL Spoken Language Technology*, 2010, pp. 312–317.

[99] B. Strope, D. Beeferman, A. Gruenstein, and X. Lei, "Unsupervised Testing Strategies for ASR," in *Proceedings of Annual Conference of the International Speech Communication Association*, 2011, pp. 1685–1688.

[100] J. D. Williams, I. D. Melamed, T. Alonso, B. Hollister, and J. Wilpon, "Crowd-

sourcing for difficult transcription of speech," in *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding*, 2011, pp. 535–540.

[101] I. McGraw, C. Ying-Lee, I. L. Hetherington, S. Seneff, and J. Glass, "Collecting Voices from the Cloud," in *Proceedings of the Conference on Language Resources and Evaluation*, 2010, pp. 1301–1318.

[102] J. Godfrey, E. Holliman, and J. McDaniel, "SWITCHBOARD: Telephone speech corpus for research and development," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1992, pp. 517–520.

[103] B. Roy and D. Roy, "Fast transcription of unstructured audio recordings," in *Proceedings of Annual Conference of the International Speech Communication Association*, 2009.

[104] J. G. Fiscus, "A Post-Processing System To Yield Reduced Word Error Rates: Recognizer Output Voting Error Reduction (ROVER)," in *IEEE Workshop on Automatic Speech Recognition and Understanding*, 1997, pp. 347–354.

[105] C. Passy, "Turning audio into words on the screen," http://online.wsj.com/article/SB122351860225518093.html, 2008.

[106] S.-S. Kang and C.-W. Woo, "Automatic segmentation of words using syllable bigram statistics," in *6th Natural Language Processing Pacific Rim Symposium*, 2001, pp. 729–732.

[107] I. Bulyiko, M. Ostendorf, and A. Stolcke, "Getting more mileage from web text sources for conversational speech language modeling using class-dependent mixtures," in *Pro-*

ceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics, 2003, pp. 7–9.

[108] P. Ipeirotis, "Mechanical turk: The demographics," http://behind-the-enemy-lines.blogspot.com/2010/03/new-demographics-of-mechanical-turk.html, 2008.

[109] L. Lamel, J.-L. Gauvain, and G. Adda, "Lightly supervised acoustic model training," in *ASR2000-Automatic Speech Recognition: Challenges for the new Millenium ISCA Tutorial and Research Workshop*, 2000, pp. 115–129.

[110] S. Kombrink, T. Mikolov, M. Karafiat, and L. Burget, "Improving Language Models For ASR Using Translated In-Domain Data," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2012, pp. 4405–4408.

[111] P. Xu, D. Karakos, and S. Khudanpur, "Self-Supervised Discriminative Training of Statistical Language Models," in *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding*, 2009, pp. 317–322.

[112] S. Della Pietra, V. Della Pietra, R. L. Mercer, and S. Roukos, "Adaptive Language Modeling Using Minimum Discriminant Estimation," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1992, pp. 633–636.

[113] R. W. Johnson, "Determining probability distributions by maximum entropy and minimum cross-entropy," in *Proceedings of the international conference on APL*, vol. 1. New York, New York, USA: ACM Press, 1979, pp. 24–29.

[114] A. Erkan, "Semi-supervised learning via generalized maximum entropy," in *Proceedings of Journal of Machine Learning Research Workshop*, vol. 9, 2010, pp. 209–216.

[115] L. L. Campbell, "Minimum Cross-Entropy Estimation with Inaccurate Side Information," in *IEEE Transactions on Information Theory*, vol. 45, no. 7, 1999, pp. 2650–2652.

[116] E. T. Jaynes, "Information Theory and Statistical Mechanics," in *Physical Review*, vol. 106, no. 4. American Physical Society, 1957, p. 620.

[117] R. Rosenfeld, "A Maximum Entropy Approach to Adaptive Statistical Language Modeling," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1996, pp. 187–228.

[118] S. Khudanpur and J. Wu, "A Maximum Entropy Language Model Integrating N-grams And Topic Dependencies For Conversational Speech Recognition," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1999, pp. 553–556.

[119] R. Lau, R. Rosenfeld, and S. Roukos, "Adaptive language modeling using the maximum entropy principle," in *Proceedings of the workshop on Human Language Technology HLT 93.* Association for Computational Linguistics, 1993, pp. 108–113.

[120] J. Goodman, "Classes for Fast Maximum Entropy Training," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, Aug. 2001, pp. 561–564.

[121] J. Wu and S. Khudanpur, "Efficient training methods for maximum entropy language modelling," in *Proceedings of International Conference on Spoken Language Processing*, vol. 3, 2000, pp. 114–117.

[122] R. Rosenfeld, "A Whole Sentence Maximum Entropy Model," in *Proceedings of the IEEE Workshop on Speech Recognition and Understanding*, 1997, pp. 230–237.

[123] J. Wu and S. Khudanpur, "Building a topic-dependent maximum entropy model for very large corpora," in *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on*, vol. 1, 2002, pp. I–777.

[124] P. S. Rao, S. Dharanipragada, and S. Roukos, "MDI Adaptation of Language Models Across Corpora," in *Proceedings of Annual Conference of the International Speech Communication Association*, 1997, pp. 1979–1982.

[125] R. Kneser, J. Peters, and D. Klakow, "Language model adaptation using dynamic marginals," in *Proceedings of Annual Conference of the International Speech Communication Association*, vol. 4, 1997, pp. 1971–1974.

[126] T. Alumae and M. Kurimo, "Domain Adaptation of Maximum Entropy Language Models," in *Computational Linguistics*, no. July, 2010, pp. 301–306.

[127] Y.-C. Tam and P. Vozila, "A Hierarchical Bayesian Approach for Semi-supervised Discriminative Language Modeling," in *Proceedings of Annual Conference of the International Speech Communication Association*, 2011.

[128] N. A. Smith and J. Eisner, "Contrastive Estimation : Training Log-Linear Models on

Unlabeled Data," in *Proceedings of the annual meeting on Association for Computational Linguistics*, 2005, pp. 354–362.

[129] Z. Younes, F. Abdallah, and T. Denceux, "Evidential Multi-Label Classification Approach to Learning from Data with Imprecise Labels," in *Proceedings of Information Processing and Management of Uncertainty.* Springer, 2010, pp. 119–28.

[130] R. Jin and Z. Ghahramani, "Learning with Multiple Labels," in *Advances in Neural Information Processing Systems*, S. T. S Becker and K. Obermayer, Eds., vol. 15. MIT Press, 2003, pp. 921–928.

[131] T. Cour and B. Sapp, "Learning from Partial Labels," in *Journal of Machine Learning Research*, vol. 12, no. 2, 2011, pp. 1501–1536.

[132] M. Dredze and P. Talukdar, "Sequence learning from data with multiple labels," in *Proceedings of European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2009, p. 39.

[133] L. Gillick and S. J. Cox, "Some statistical issues in the comparison of speech recognition algorithms," in *Acoustics, Speech, and Signal Processing, 1989. ICASSP-89., 1989 International Conference on*, 1989, pp. 532–535.

[134] D. S. Pallet, W. M. Fisher, and J. G. Fiscus, "Tools for the analysis of benchmark speech recognition tests," in *Acoustics, Speech, and Signal Processing, 1990. ICASSP-90., 1990 International Conference on*, 1990, pp. 97–100.

[135] V. J. D. Pietra, A. Berger, S. Della Pietra, and V. Della Pietra, "A Maximum Entropy

Approach to Natural Language Processing," in *Computational Linguistics*, no. 22-1, 1996, pp. 39–71.

[136] M.-H. Siu, H. Gish, and F. Richardson, "Improved estimation, evaluation and applications of confidence measures for speech recognition," in *Proceedings of Annual Conference of the International Speech Communication Association*, 1997.

[137] D. Povey, "Discriminative training for large vocabulary speech recognition," Ph.D. dissertation.

[138] H. Schwenk, "Continuous space language models," in *Computer Speech & Language*, vol. 21, no. 3. Elsevier, 2007, pp. 492–518.

[139] M. Sarclar, "Pronunciation modeling for conversational speech recognition," Ph.D. dissertation, Johns Hopkins University, 2004.

[140] T. Hori, I. L. Hetherington, T. J. Hazen, and J. R. Glass, "Open-vocabulary spoken utterance retrieval using confusion networks," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2007, pp. 73–76.

[141] J. G. Fiscus, J. Ajot, J. S. Garofolo, and G. Doddingtion, "Results of the 2006 spoken term detection evaluation," in *Proceedings of Annual Conference of the International Speech Communication Association*, vol. 7, 2007, pp. 51–57.

Vita

Scott Novotney was raised in Tacoma, Washington by two teachers: one taught reading and the other mathematics. He received his B.A. in Mathematics and M.Sc. in Computer Science from Johns Hopkins University. He is an Eagle Scout, an AFOL (Adult Fan of Legos), and has a one year old black lab named Lego.