# Réduction des coûts de développement de systèmes de reconnaissance de la parole à grand vocabulaire

Thiago Fraga da Silva

UNIVERSITÉ PARIS SUD

Comprendre le monde, construire l'avenir®

# Université Paris-Sud

## Ecole Doctorale 427 : Informatique Paris Sud

### Laboratoire d'Informatique pour la Mécanique et les Sciences de l'Ingénieur

Thèse de doctorat

Informatique

par

**Thiago FRAGA DA SILVA**

# Reducing development costs of large vocabulary speech recognition systems

**Date de soutenance:** 29/09/2014

**Composition du jury :**

| | | |
|---|---|---|
| Directeur de thèse : | M. Jean-Luc Gauvain | Directeur de Recherche (CNRS-LIMSI) |
| Rapporteurs : | M. Denis Jouvet | Directeur de Recherche (INRIA-LORIA) |
| | Mme. Tanja Schultz | Professeur (Karlsruhe Institute of Technology) |
| Examinateurs : | Mme. Anne Vilnat | Professeur (Université Paris-Sud) |
| | M. Ralf Schlüter | Chercheur Associé (RWTH Aachen University) |
| | M. Driss Matrouf | Maître de Conférences - HDR (LIA-CERI) |

# Acknowledgments

After years of work, I am glad to come trough this section, what happens to contain the last words written in this manuscript. During this period, I was lucky to have the support in many different ways from many people.

First, I would like to thank my adviser Jean-Luc Gauvain, who gave me the opportunity to continue my research work at the Spoken Language Group at CNRS-LIMSI. His guidance and perspicacity were fundamental in the development of my research skills. I would like to thank Lori Lamel, from whom I learned a lot, for her precious advice and for reviewing my writings.

I would like to express my gratitude to Tanja Schultz and Denis Jouvet who reviewed my thesis and provided detailed comments, corrections and suggestions to improve the quality of this manuscript. I am also grateful to the other members of my defense committee, Ralf Schlüter, Driss Matrouf and Anne Vilnat. Thank you all for your remarks, questions, suggestions and the report. Special thanks to Anne who intervened on my behalf to make the completion of my thesis feasible (despite some administration issues).

I would like to remember Thaïs Cristófaro and Léo Almeida, the first who showed me the path of the scientific research, there back in 2004. More than mentors, they have become true friends. My sincere acknowledgments also to Hermann Ney, who made possible my stay at the RWTH-Aachen University for a couple of months to develop part of my research work.

During my thesis, I had the opportunity to live with very nice people at LIMSI. In particular I shared the office with Penny, who has rapidly become my preferred *cobure*. Everyone else envied the beautiful postcard wall we used to share, only because they could see it at a first glance. Nadi at his turn was my preferred *coloc*. We shared many good moments at the lab, at home and elsewhere, là où on ne s'inquiètait pas trop. Thanks to Nadège for picking me up in my first day and for her undeniable gift of making our lives easier, to Thomas for always helping me keeping my glass half full, to Ilya for teaching me we can play soccer seriously and work with some fun, to Son for showing me that soccer works better when structured, to Billy for showing me that a meter might be a very long distance (by the way, this thesis has about a half kilometer of letters) and all the fourish breaks we made together, to Nico, Nico, Clément, Guillaume, Marianna, Phillippe, Hervé, and also to my soccer mates, Sylvain, Mathieu, Tom, Téo.

Beyond the borders of the lab, I owe my thanks to the people that also helped me to make the path through the thesis smoother, friends from the past and friends made along the way. Penso notamente aos meus velhos amigos Gabriela, Guzella, Chasin, Benoît, Leandro e Lelinho que, pessoalmente ou por telefone, sempre estiveram por perto no decorrer dos últimos anos. Não posso deixar de agradecer aos companheiros que a Cité U me apresentou e que marcaram grande parte dessa trajetória, Gyselle, Denise, Carol,

Pedro, Jacques, Maurício, André, Rosangela, Jamacy, Priscilla e Letícia.

Gostaria de agradecer igualmente à minha família, meus pais Geraldo e Vera, meus irmãos, Bruno e Rodrigo, e Fabiana e Bella. Agradeço a Deus por vocês. Jamais teria as palavras que bastariam para expressar todo meu agradecimento pela companhia e suporte que vocês me ofereceram, apesar da distância.

Finalement, j'aimerais remercier Christine, qui a été à mon côté, m'a soutenu et a été plus qu'un pilier pour moi au cours des dernières années. Merci, chérie !

# Abstract

One of the outstanding challenges in large vocabulary automatic speech recognition (ASR) is the reduction of development costs required to build a new recognition system or adapt an existing one to a new task, language or dialect. The state-of-the-art ASR systems are based on the principles of the statistical learning paradigm, using information provided by two stochastic models, an acoustic (AM) and a language (LM) model. The standard methods used to estimate the parameters of such models are founded on two main assumptions: the training data sets are large enough, and the training data match well the target task. It is well-known that a great part of system development costs is due to the construction of corpora that fulfill these requirements. In particular, manually transcribing the audio data is the most expensive and time-consuming endeavor. For some applications, such as the recognition of low resourced languages or dialects, finding and collecting data is also a hard (and expensive) task. As a means to lower the cost required for ASR system development, this thesis proposes and studies methods that aim to alleviate the need for manually transcribing audio data for a given target task. Two axes of research are explored.

First, *unsupervised* training methods are explored in order to build three of the main components of ASR systems: the acoustic model, the multi-layer perceptron (MLP) used to extract acoustic features and the language model. The unsupervised training methods aim to estimate the model parameters using a large amount of automatically (and inaccurately) transcribed audio data, obtained thanks to an existing recognition system. A novel method for unsupervised AM training that copes well with the automatic audio transcripts is proposed: the use of multiple recognition hypotheses (rather than the best one) leads to consistent gains in performance over the standard approach. Unsupervised MLP training is proposed as an alternative to build efficient acoustic models in a fully unsupervised way. Compared to cross-lingual MLPs trained in a supervised manner, the unsupervised MLP leads to competitive performance levels even if trained on only about half of the data amount. Unsupervised LM training approaches are proposed to estimate standard backoff $n$-gram and neural network language models. It is shown that the performance improvements obtained with unsupervised LM training are complementary to those obtained with unsupervised AM training.

Second, this thesis proposes the use of model interpolation as a rapid and flexible way to build task specific acoustic models. In reported experiments, models obtained via interpolation outperform the baseline pooled models and equivalent maximum *a posteriori* (MAP) adapted models. Interpolation proves to be especially useful for low resourced dialect ASR. When only a few (2 to 3 hours) or no acoustic data truly matching the target dialect are available for AM training, model interpolation leads to substantial performance gains compared to the standard training methods.

# Résumé

Au long des dernières décennies, des importants avancements ont été réalisés dans le domaine de la reconnaissance de la parole à grand vocabulaire. Un des défis à relever dans ce domaine concerne la réduction des coûts de développement nécessaires pour construire un nouveau système ou adapter un système existant à une nouvelle tâche, langue ou dialecte. Les systèmes de reconnaissance de la parole à l'état de l'art sont basés sur les principes de l'apprentissage statistique, utilisant l'information fournie par deux modèles stochastiques, un modèle acoustique (MA) et un modèle de langue (ML). Les méthodes standards utilisées pour construire ces modèles s'appuient sur deux hypothèses de base: les jeux de données d'apprentissage sont suffisamment grands, et les données d'apprentissage correspondent bien à la tâche cible. Il est bien connu qu'une partie importante des coûts de développement est dû à la préparation des corpora qui remplissent ces deux conditions, l'origine principale des coûts étant la transcription manuelle des données audio. De plus, pour certaines applications, notamment la reconnaissance des langues et dialectes dits "peu dotés", la collecte des données est en soi une mission difficile. Cette thèse a pour but d'examiner et de proposer des méthodes qui visent à réduire le besoin de transcrire manuellement des données audio pour une tâche cible donnée. Deux axes de recherche ont été suivis.

Dans un premier temps, des méthodes d'apprentissage dits "non-supervisées" sont explorées. Leur point commun est l'utilisation des transcriptions audio obtenues automatiquement à l'aide d'un système de reconnaissance existant. Des méthodes non-supervisées sont explorées pour la construction de trois des principales composantes des systèmes de reconnaissance. D'abord, une nouvelle méthode d'apprentissage non-supervisée des MAs est proposée: l'utilisation de plusieurs hypothèses de décodage (au lieu de la meilleure uniquement) conduit à des gains de performance substantiels par rapport à l'approche standard. L'approche non-supervisée est également étendue à l'estimation des paramètres du réseau de neurones (RN) utilisé pour l'extraction d'attributs acoustiques. Cette approche permet la construction des modèles acoustiques d'une façon totalement non-supervisée et conduit à des résultats compétitifs en comparaison avec des RNs estimés de façon supervisée. Finalement, des méthodes non-supervisées sont explorées pour l'estimation des MLs à repli (*backoff*) standards et MLs neuronaux. Il est montré que l'apprentissage non-supervisé des MLs conduit à des gains de performance additifs (bien que petits) à ceux obtenus par l'apprentissage non-supervisé des MAs.

Dans un deuxième temps, cette thèse propose l'utilisation de l'interpolation de modèles comme une alternative rapide et flexible pour la construction des MAs pour une tâche cible. Les modèles obtenus à partir d'interpolation se montrent plus performants que les modèles de base, notamment ceux estimés à échantillons regroupés ou ceux adaptés à la tâche cible. On montre que l'interpolation de modèles est particulièrement utile pour la

reconnaissance des dialectes peu dotés. Quand la quantité des données d'apprentissage acoustiques du dialecte ciblé est petite (2 à 3 heures) ou même nulle, l'interpolation des modèles conduit à des gains de performances considérables par rapport aux méthodes standards.

# Contents

# List of Figures

# List of Tables

# List of Abbreviations

| | |
|---|---|
| **AM** | Acoustic model |
| **ASR** | Automatic speech recognition |
| **BC** | Broadcast conversations |
| **BN** | Broadcast news |
| **BP** | Brazilian Portuguese |
| **CART** | Classification and regression tree |
| **CAT** | Cluster adaptive training |
| **CMLE** | Constrained maximum likelihood estimation |
| **CMLLR** | Constrained maximum likelihood linear regression |
| **DCT** | Discrete cosine transform |
| **DNN** | Deep neural network |
| **EM** | Expectation-maximization |
| **EP** | European Portuguese |
| **GMM** | Gaussian mixture model |
| **G2P** | Grapheme-to-phoneme |
| **HMM** | Hidden Markov model |
| **ISE** | Integral squared error |
| **KL** | Kullback-Leibler |
| **KN** | Kneser-Ney |
| **LVCSR** | Large vocabulary continuous speech recognition |
| **LM** | Language model |

| | |
|---|---|
| **LSA** | Latent semantic analysis |
| **MAP** | Maximum *a posteriori* |
| **MAPSSWE** | Matched pairs sentence-segment word error |
| **MFCC** | Mel-frequency cepstral coefficients |
| **ML** | Maximum likelihood |
| **MLE** | Maximum likelihood estimation |
| **MLLR** | Maximum likelihood linear regression |
| **MLP** | Multi-layer perceptron |
| **MMI** | Maximum mutual information |
| **MMIE** | Maximum mutual information estimation |
| **MPE** | Minimum phone error |
| **NNLM** | Neural network language model |
| **OOV** | Out of vocabulary |
| **PCA** | Principal component analysis |
| **PER** | Phone error rate |
| **PLP** | Perceptual linear predictive |
| **SAT** | Speaker adaptive training |
| **SOUL** | Structured output layer |
| **TRAP** | Temporal pattern |
| **UBM** | Universal background model |
| **VTLN** | Vocal tract length normalization |
| **WER** | Word error rate |

# Part I.

# Context of this work

# Chapter 1

# Introduction

For many decades scientists have attempted to design machines that are capable of reproducing or imitating human behavior. As speech is the most common form of communication between humans, trying to make machines understand (and reproduce) natural speech has been a widely investigated research topic. The main goal of this research area is to allow verbal communication between humans and machines to be more natural, and, in some cases, to facilitate also the communication between humans.

Automatic speech understanding aims to obtain the underlying message of a spoken utterance. Extracting such information directly from the audio stream (the digital representation of a recorded utterance) is a challenging and complex task. The audio signal that carries the message also contains non-linguistic information that is not always useful for understanding the message. For this reason, the audio stream is usually converted into a sequence of written words before being processed by the final application. Performing this conversion from audio to text is the main goal of automatic speech recognition (ASR).

Interpreting the meaning or the semantics behind words is itself a difficult endeavor, hence representing a separate research area. For instance, it might be particularly hard to understand nuances of irony or figures of speech, such as metaphor, hyperbole or simile. Despite this fact, ASR can be very useful for different applications, such as voice command, audio media indexing, information retrieval, machine translation, dictation or computer-aided communication with people suffering from hearing disabilities.

ASR research has been investigated for at least six decades. In the beginning, ASR was limited to speaker dependent recognition of isolated words pertaining to short list vocabularies. The technology has gradually evolved towards the recognition of more complex tasks. Nowadays, ASR systems are able to cope with large vocabulary continuous speech from multiple speakers with reasonable accuracy. Owing however to the fact that ASR research has focused on the most spoken languages of the world, the highest levels of recognition performance are limited to these few languages.

In the past years, there has been a growing interest on building large vocabulary ASR systems for different domains and languages. An example close to me is the Quaero Programme[1], a European Consortium formed by academic and industrial partners. During a period of six years, speech recognition systems for 23 different languages were developed. Another more recent case is the (ongoing) IARPA Babel Program[2], which focuses on the

[1] http://www.quaero.org

[2] http://www.iarpa.gov/index.php/research-programs/babel

recognition of languages that have a limited amount of resources. This initiative is not restricted only to academia: private companies like Apple, IBM, Google, Microsoft, as many others, have already demonstrated some interest in expanding the use of speech technologies in their products designed for the public.

## 1.1. The statistical framework

Current state-of-the-art ASR systems stand on the principles of statistical pattern recognition. In this approach, by means of the Bayesian rule decision, the recognition task resumes on searching for the most likely word sequence given the speech data and two probabilistic models: the acoustic and the language models. Prior to searching, the audio stream is converted into a suitable sequence of acoustic feature vectors.

The parameters of such probabilistic models are obtained via a training algorithm that attempts to fit a set of available training samples to a suitable objective function. As speech recognition is a classification problem, the correct classification labels of the training data are usually required. Training itself is an automatic process, although choices for algorithms and optimization criteria still rely on expert's skills. In fact, it must be considered that the training data constitute the ground truth of the process being modeled. Concerning these data, two fundamental assumptions need to be considered.

1. The corpus used for training has to be **large** enough to generate reliable estimates and to lead to good generalization on new data.

2. The data has to be **representative** of the target task. In other words, it is assumed that training and test data are independent identically distributed samples drawn from the same probability density function.

Despite the huge progress that has been made on speech recognition over the last decades, developing or adapting a recognition system to a new language or domain is still an expensive and time-consuming task. Evidently, reducing system development costs is a necessary step in order to ensure the rapid expansion of speech recognition technologies.

Factors influencing development costs and time are varied, including, of course, the availability of computational resources. Another important cost factor is the human effort needed. Although a great part of the development process can be performed automatically, human supervision is required, for instance, on architectural decisions (acoustic feature vectors, type of models, estimation criteria, smoothing algorithms, decoding strategy, etc.), data selection or data annotation.

It is well known that one of the most important cost factors involved in system development is the production of manually annotated audio data. Reducing the need for this certainly leads to a reduction of development costs and, consequently, favors the rapid expansion of speech recognition technologies. Hence, the first research direction investigated in this thesis deals with the use of automatically annotated audio data.

Because the similarity between the training data and the target data is also important, gathering a large amount of data is not always sufficient to guarantee suitable performance levels. Furthermore, large quantities of data are not always easily available for certain (low resourced) languages, dialects or domains. In these cases, additional effort is required to

construct representative training corpora and to refine the system parameters. The second axis of research covered in this thesis' work deals with the use of techniques to reduce the need for training data in the target domain, hence reducing the human effort necessary to develop systems for low resourced tasks.

## 1.2. Cost efficient training data production

As mentioned before, the first assumption in the statistical framework is the availability of large amounts of training data. In the case of automatic speech recognition, the training corpora are composed of acoustic and textual data.

In particular, acoustic model (AM) estimation requires both, the audio data and their associated text transcriptions. Usually, dozens to hundreds of hours of data are required for acoustic model estimation in order to achieve suitable performance levels on large vocabulary recognition tasks. While collecting acoustic data is relatively easy (although not always), manually transcribing them is an expensive and time-consuming task. Roughly, 10 to 50 hours of work are required to produce detailed transcriptions for one hour of audio data. Of course, this number may vary depending on the difficulty of the task, the experience of the transcriber and the desired level of transcription details.

A well-known technique that has been growing in popularity as a means to reduce the need for manual annotation for acoustic learning is the method known as *unsupervised training* (Zavaliagkos and Colthurst, 1998; Kemp and Waibel, 1999). In this method, an initial model trained for another task or with a small amount of data is used to process several hours of untranscribed acoustic data and generate approximate transcriptions. These transcriptions are then used as ground truth for AM training. The main drawback of such an approach is that the automatic transcriptions are known to contain many errors that can mislead parameter estimation. Dealing with them is an important issue in order to improve efficiency of unsupervised acoustic model training (Kemp and Waibel, 1999; Lamel et al., 2000; Gollan et al., 2007).

Language model (LM) estimation, on the other hand, requires only textual data. Usually, dozens of millions to billions of words are required to build language models for large vocabulary recognition tasks. Huge amounts of data can be readily gathered from different types of digital media, such as the Web, newspapers, books, etc. Nevertheless, most of these available sources represent a more formal (written) style rather than the spoken style intrinsic to natural speech. Information retrieval techniques can be used to reduce the need for audio transcriptions by searching on the Web for data that represents well the spoken form (Bulyko et al., 2007).

Despite the fact that manual audio transcriptions are expensive to produce, it is not difficult to prove that they play a major role in language modeling, since they truly represent speech. An alternative that naturally arises is the use of automatic transcriptions as ground truth for language model parameter estimation (Bacchiani and Roark, 2003). Unsupervised language modeling is a much more challenging problem compared to unsupervised acoustic modeling (Novotney et al., 2009). This is probably due to the fact that language modeling is a sparser optimization problem and, thus, more sensitive to wrongly assigned classification labels.

In brief, audio transcriptions are a valuable resource for ASR system development, being essential for the estimation of acoustic models and likewise important for the esti-

mation of language models.

## 1.3. Matching the target data

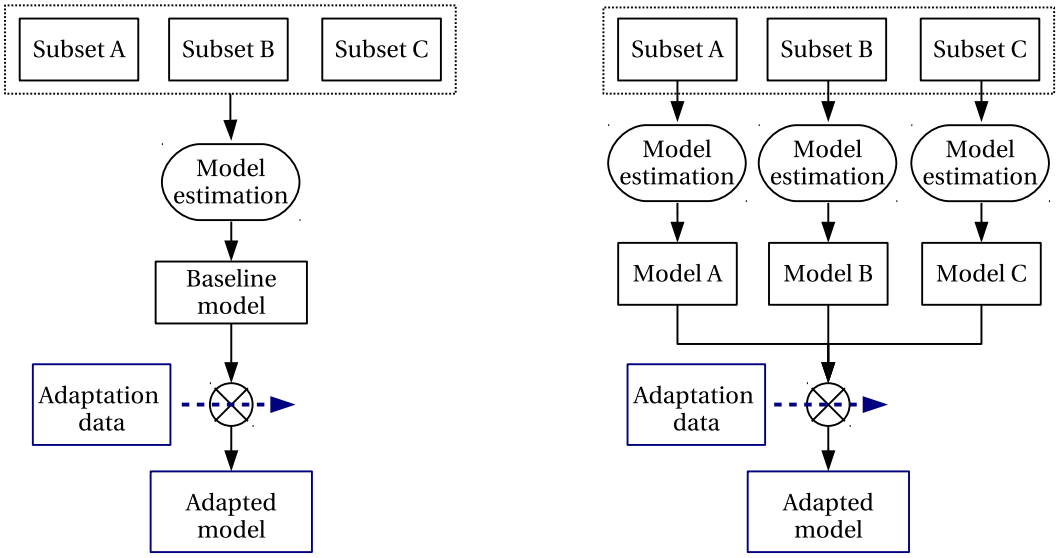
As previously stated, using huge amounts of data is not a sufficient condition to get reliable models. The training data has to match the target task too. Thus, a problem arises in cases where target specific data is scarce. In particular, it is difficult to retrieve (acoustic and textual) data for conversational speech recognition tasks and for low e-resourced languages or dialects.

Data mismatch issues are usually addressed using a suitable model adaptation technique. The process uses *two sources of knowledge*: a well-trained baseline model estimated on a large amount of (possibly) out-of-domain data; and a small amount of adaptation data, from which relevant information related to the target task can be extracted. The main principle, which is illustrated in Figure 1.1a, is to adjust the parameters of the baseline model to approximate the adaptation data. Adaptation also incurs costs related to human effort. First, audio transcriptions (though in a smaller quantity) are required. Second, human supervision might be required to refine the adaptation parameters. These two cost factors can be reduced using either unsupervised AM training or automatic optimization of the adaptation parameters.

Various acoustic and language model adaptation techniques exist. In some of them, the concept of degree of adaptation is intrinsic. An example is the maximum *a posteriori* (MAP) adaptation of acoustic (Gauvain and Lee, 1994) and language (Bacchiani and Roark, 2003) models, where a parameter, which can be set manually or obtained empirically, allows to control the relevance of the two sources of knowledge. In other cases, the adapted model is obtained by forcing the baseline model to satisfy constraints extracted from the adaptation data. This is the case, for instance, of maximum likelihood linear regression acoustic model adaptation approaches (Leggetter and Woodland, 1995) or minimum discrimination information based language model adaptation (Kneser et al., 1997; Federico, 1999).

In the aforementioned methods, it is assumed that data used to estimate the baseline model are homogeneous, drawn from the same distribution function. However, this assumption is not strictly true, since these data actually come from a wide variety of sources. Considering these dissimilarities is a potential way to improve model adaptation. A proposed approach, widely used for language modeling, is to consider the background model as a set of component models, each estimated on an independent subset of the training data. The task specific model is obtained by means of interpolation of the component models, allowing, then, to control the degree of adaptation of each subset with respect to the target specific data. This approach is highly flexible, since no changes on the component models are required while adapting the model to a new task. In fact, adaptation can be performed by simply estimating a few interpolation coefficients. A main drawback of this model combination technique is the data fragmentation introduced by dividing the training data into subsets. This approach is illustrated in Figure 1.1b.

Usually, such a combination approach is not applied for AM training. Acoustic training data dissimilarities are typically addressed via adaptive training approaches (Anastasakos et al., 1996; Kuhn et al., 1998; Gales, 1998a), a family of methods that attempt to remove from the acoustic model the speech variability among the speakers represented




(a) Model adaptation.

(b) Model combination.

Figure 1.1.: Representation schemes of model adaptation and model combination methods.

in the training data. Different from the research direction taken in this thesis, adaptive training methods do not cope with low resourced data tasks.

## 1.4. Scientific goals

The main objective of this thesis is to study and propose new techniques that aim to reduce the human effort required for system development. Two different research axes are considered. First, the use of unsupervised techniques to alleviate the need for manually transcribed audio data for acoustic and language model training. Second, the use of model adaptation and model combination techniques to alleviate the need for task specific data. Of particular interest for this thesis is the use of adaptation and combination techniques in which the adaptation parameters are automatically estimated on the target data, requiring low human supervision. The main scientific goals pursued in this thesis are:

**Investigate unsupervised HMM/GMM based acoustic model training techniques.** It will be shown that unsupervised training is a particular case of the expectation-maximization (EM) algorithm (Dempster et al., 1977), in which the true word sequence labels are not observed. A survey of different approaches to deal with recognition errors will be performed, showing that they are all specific cases of a generalized EM algorithm. Furthermore, this thesis proposes to base acoustic model parameter estimation on multiple weighted decoding hypotheses instead of the 1-best hypothesis. The multiple hypotheses are extracted from the recognition lattices as an attempt to better represent the true transcription labels and lead to more reliable models.

**Investigate unsupervised MLP training for acoustic feature extraction.** Acoustic features extracted from discriminatively trained multi-layer perceptrons (MLP) have been successfully used in large vocabulary ASR tasks (Fousek et al., 2008b). Training such models relies on the use of transcribed audio data. This thesis proposes to extend the use of unsupervised training approaches to estimate the parameters of bottleneck MLP models (Grézl et al., 2007). The proposed method is empirically compared to cross-lingual MLPs (Stolcke et al., 2006).

**Investigate unsupervised language model training.** Unsupervised training of language models has been reported to improve recognition accuracy when applied in conversational speech recognition tasks (Bacchiani and Roark, 2003; Novotney et al., 2009). Here, we propose to use this technique to estimate language models for large vocabulary broadcast speech recognition tasks. Unsupervised training is assessed on the state-of-the-art $n$-gram based models and on neural network based language models. In order to take into accounts reliable information provided by the decoder, namely the confidence measures, a variant of the so-called Kneser-Ney (KN) $n$-gram smoothing which allows the use of fractional counts is proposed.

**Investigate acoustic model interpolation.** Inspired by common language modeling practices, it is proposed to generate task specific acoustic models as a combination of component models, each trained on an independent subset of the training data. The models are combined *a posteriori* via linear interpolation using coefficients automatically estimated on a small amount of task specific data. The method is theoretically compared to adaptive training approaches and empirically compared to pooled and MAP adapted models. The proposed acoustic model interpolation technique induces an increase on the number of model parameters. In order to reduce them and recover suitable computational complexity levels, a theoretically motivated Gaussian mixture model (GMM) reduction algorithm is also proposed.

## 1.5. Outline

This thesis is organized in four parts. **Part I** presents the general motivation and context of this work, including this introduction. **Chapter 2** presents a background of automatic speech recognition, while **Chapter 3** describes the baseline systems used in this work.

**Part II** presents the work addressing unsupervised training. After a brief introduction of this subject, **Chapter 4** describes the work on unsupervised acoustic model training. **Chapter 5** extends the approach to the estimation of bottleneck multi-layer perceptron parameters. This part finishes with an extension to unsupervised language model training in **Chapter 6**.

**Part III** concerns the model combination studies. After a brief overview of this subject, **Chapter 7** describes the theoretical acoustic model interpolation framework. Experimental work in this topic are presented in **Chapter 8**.

**Part IV** gives an overall summary on this thesis work highlighting the main results obtained and suggesting some future research directions.

# Chapter 2

# Automatic speech recognition

The aim of the speech recognition task is to convert a continuous speech audio flow into a sequence of written words that represents the spoken utterance. Current state-of-the-art large vocabulary continuous speech recognition (LVCSR) systems stand on the principles of statistical pattern recognition. In this approach, by means of the Bayesian decision rule, the recognition task can be stated as the problem of searching for the most likely word sequence given two probabilistic models: the acoustic and the language models.

In this chapter, an overview of the statistical methods for LVCSR is presented with special attention to acoustic and language modeling. Due to the importance in the general context of this thesis, the searching algorithm is briefly described as well.

## 2.1. Statistical speech recognition

The current state-of-the-art LVCSR approaches are based on the principles of statistical learning methods. Given an input audio stream, represented by a sequence of observation vectors $\mathcal{X}$, this approach seeks to find the optimal word sequence $\mathcal{W}$ that maximizes the *posterior* probability $P(\mathcal{W}|\mathcal{X})$, that is:

$$\mathcal{W}^* = \arg\max_{\mathcal{W}} \; P(\mathcal{W}|\mathcal{X}) = \arg\max_{\mathcal{W}} \frac{P(\mathcal{W}) \cdot f(\mathcal{X}|\mathcal{W})}{P(\mathcal{X})} \tag{2.1}$$

$$\mathcal{W}^* = \arg\max_{\mathcal{W}} P(\mathcal{W}) \cdot f(\mathcal{X}|\mathcal{W}) \tag{2.2}$$

The passage in (2.1) is the straightforward application of the Bayes' decision rule. In (2.2), the prior probability over the observation vectors, $P(\mathcal{X})$, is ignored, since it does not affect the maximization over $\mathcal{W}$. The key point here is the Bayes' decision rule, which allows the recognition problem to be decomposed into two terms, one representing the acoustic *likelihood* function, $f(\mathcal{X}|\mathcal{W})$, provided by an AM, and another representing the *prior* knowledge over the word sequences, $P(\mathcal{W})$, provided by a language model (LM). These models are usually estimated independently. The maximization procedure over the word sequences is commonly referred to as *decoding* or *search* algorithm.

For LVCSR tasks, modeling the likelihood $f(\mathcal{X}|\mathcal{W})$ directly is impractical. To make the problem more tractable, sub-word units, such as phones, are used (Cohen and Mercer, 1975). Of course, this requires the assumption that any word sequence can be mapped into

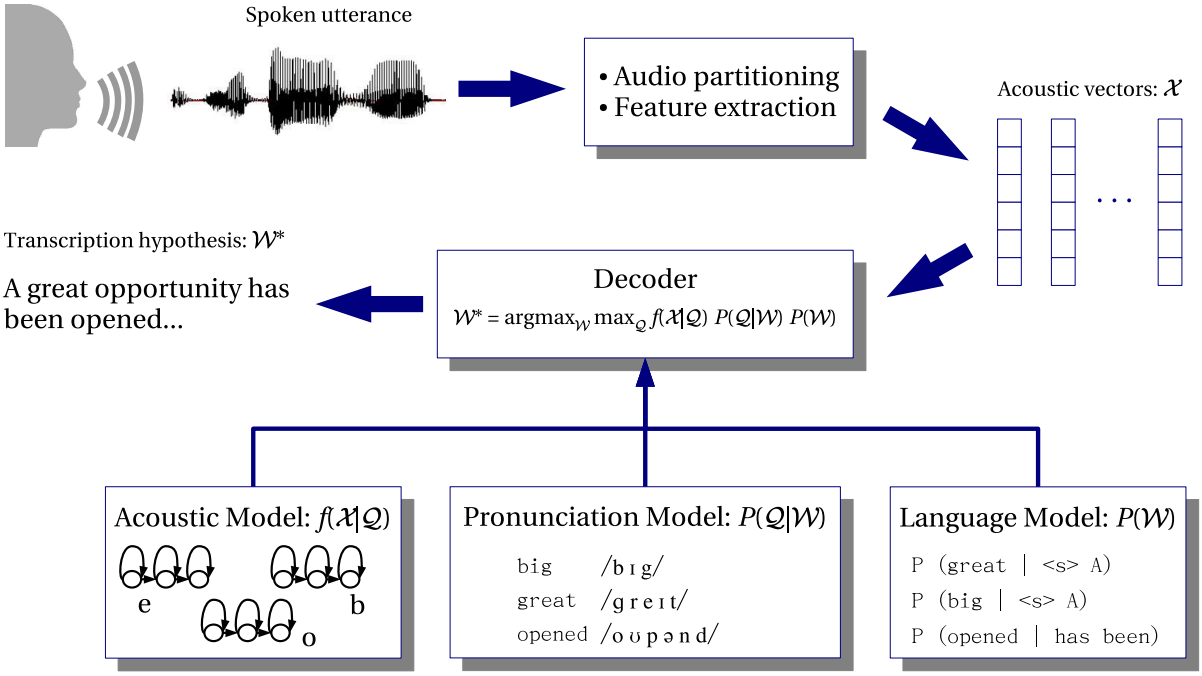



Figure 2.1.: Automatic speech recognition system.

at least one sub-word unit sequence. This mapping function is provided by a pronunciation model.

Considering $\mathcal{Q}$ a phone sequence, and assuming that $f(\mathcal{X}|\mathcal{Q},\mathcal{W}) = f(\mathcal{X}|\mathcal{Q})$, (2.2) can be rewritten as follows:

$$\mathcal{W}^* = \arg\max_{\mathcal{W}} \sum_{\mathcal{Q}} P(\mathcal{W}) \cdot P(\mathcal{Q}|\mathcal{W}) \cdot f(\mathcal{X}|\mathcal{Q}) \tag{2.3}$$

$$\mathcal{W}^* \approx \arg\max_{\mathcal{W}} \max_{\mathcal{Q}} P(\mathcal{W}) \cdot P(\mathcal{Q}|\mathcal{W}) \cdot f(\mathcal{X}|\mathcal{Q}) \tag{2.4}$$

where the acoustic model is denoted by $f(\mathcal{X}|\mathcal{Q})$, the pronunciation model by $P(\mathcal{Q}|\mathcal{W})$ and the language model by $P(\mathcal{W})$. In (2.4), a Viterbi approximation is performed in order to restrict the search space during decoding.

These three models (acoustic, pronunciation and language), together with the searching algorithm constitute the core of typical ASR systems, which is depicted in Figure 2.1. Prior to the global search process, the continuous audio stream has to be converted into a sequence of discrete vectors, what is performed by the acoustic *feature extraction* module.

In the next section, further details of acoustic modeling and acoustic feature extraction will be presented. The following sections will describe language modeling and the searching algorithm. Since pronunciation modeling is not in the scope of this thesis, no further details will be given here.

## 2.2. Acoustic modeling

The acoustic model (AM) is a crucial component in any typical ASR system. It is in charge of providing the likelihood of a sequence of observed acoustic feature vectors for a

given sequence of labels, which can be words or sub-word units. Each acoustic model is commonly represented by a hidden Markov model (HMM), with the output state densities modeled by multivariate GMMs.

### 2.2.1. Hidden Markov models

By means of hidden Markov models, the continuous speech can be modeled as a first order Markov process with unobserved or *hidden* states. HMMs have been used in speech recognition since the 1970's (Baker, 1975; Jelinek, 1976).

Approaches using HMMs are founded on two assumptions. First, it is assumed that the audio stream can be split into segments in which the waveform can be considered to be stationary. Second, it is assumed that the probability of a sample being generated depends only on the current state, or, more specifically, it does not depend on previous states nor previously observed samples. Even if both assumptions are not strictly true for the continuous speech, the HMM-based approach has been proving its effectiveness for speech recognition over the past decades, constituting the state-of-the-art in matters of acoustic modeling. To derive the mathematical framework, let us adopt the following notations:

| | |
|---|---|
| $\boldsymbol{\lambda} = (\boldsymbol{\pi}, \boldsymbol{A}, \boldsymbol{\theta})$ | the HMM parameter vector |
| $i = \{0 \dots N\}$ | the HMM state indexes |
| $\boldsymbol{\pi} = \{\pi_i\}_{i=0\dots N}$ | the vector of state initial probabilities |
| $\boldsymbol{A} = \{a_{i,j}\}_{i,j=0\dots N}$ | the matrix of transition probabilities between states |
| where $a_{i,j}$ | is the transition probability from state $i$ to state $j$ |
| $\boldsymbol{\theta} = \{\theta_i\}_{i=1\dots N}$ | the state emission probability parameter vectors |
| $\mathcal{X} = (x_1, \dots, x_T)$ | a sequence of T observation acoustic vectors |
| $\mathcal{S} = (s_1, \dots, s_T)$ | a sequence of states having generated $\mathcal{X}$ |
| $\mathcal{W} = (w_1, \dots, w_M)$ | a sequence of M words generated by $\mathcal{X}$ |

Following Baum et al. (1970), we will consider the first state of the HMM, $i = 0$, as a non-emitting state. Given these definitions, the likelihood of the acoustic observation vectors $\mathcal{X}$ given the model $\boldsymbol{\lambda}$ can be represented by:

$$f\left(\mathcal{X}|\boldsymbol{\lambda}\right) = \sum_{\mathcal{S}} \pi_{s_0} \prod_{t=1}^{T} a_{s_{t-1},s_t} f\left(x_t|\theta_{s_t}\right) \tag{2.5}$$

where $f\left(x_t|\theta_i\right)$ is the emission probability density function (p.d.f.) at state $i$. In (2.5), the summation is performed over all possible state sequences.

Two distinct and independent processes can be identified in (2.5). The first one, referred to as the Markov process, is parameterized by the state initial probabilities, $\boldsymbol{\pi}$, and the state transition probabilities, $\boldsymbol{A}$. The second one, referred to as the output process, is

parameterized by $\boldsymbol{\theta}$. This later is commonly represented by multivariate Gaussian mixture distributions, which can be expressed by:

$$f\left(x_t|\theta_i\right) = \sum_{k=1}^{K} \omega_{ik} \cdot \mathcal{N}\left(x_t|\mu_{ik}, \Sigma_{ik}\right) \tag{2.6}$$

where $K$ denotes the number of Gaussian components in the mixture $\theta_i$, defined by $\theta_i = \{\omega_{ik}, \mu_{ik}, \Sigma_{ik}\}_{k=1\ldots K}$. For notation clearness, it will be assumed that all the mixtures have the same number of components. Here, $\omega_{ik}$ denotes the mixture weight, $\mu_{ik}$ the mean vector and $\Sigma_{ik}$ the covariance matrix of the $k$-th component. The mixture weights satisfy $\sum_k \omega_{ik} = 1$, and $\mathcal{N}\left(\cdot|\mu, \Sigma\right)$ is a normal p.d.f., defined as:

$$\mathcal{N}\left(x|\mu, \Sigma\right) := \frac{1}{(2\pi)^{\frac{d}{2}} \left|\Sigma\right|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}\left(x-\mu\right)^{\mathrm{T}} \Sigma^{-1} \left(x-\mu\right)\right\} \tag{2.7}$$

where $d$ is the dimensionality of the feature space, $|M|$ and $M^{-1}$ denote, respectively, the determinant and the inverse of a matrix $M$, and $v^{\mathrm{T}}$ denotes the transpose of a vector $v$. In the following section, standard methods used to estimate the HMM and GMM parameters are presented.

## 2.2.2. Parameter estimation

Obtaining the parameters of the HMM-based acoustic models with GMM state observations is a large multidimensional optimization problem. These parameters are estimated in order to fit some appropriate training criterion. Depending on their objective function, the training methods can be divided into two main categories, *generative* or *discriminative*. The first family of methods attempts to maximize the joint probability function, $P\left(\mathcal{X}, \mathcal{C}\right)$, between the observation vectors, $\mathcal{X}$, and their associated classification labels, $\mathcal{C}$. Generative methods attempt to model the manner how the data are generated. On the other hand, the discriminative methods model the conditional probability $P\left(\mathcal{C}|\mathcal{X}\right)$, seeking to maximize the separation distance among the existing classes (Bishop and Lasserre, 2007).

Although discriminative methods usually lead to better generalization performances, we focus here on generative training methods. This choice comes from the fact that generative acoustic models are less sensitive to the errors present in the automatic transcriptions (Wang et al., 2007) which are used to guide the parameter estimation throughout this work. Furthermore, rather than training from scratch, discriminative methods are often used to refine generative model estimates. Thus, it can be expected that improvements on the generative models would reflect on improvements on the discriminative ones.

### Maximum-likelihood estimation

The maximum likelihood estimation (MLE) is probably the most popular method used for HMM training due to the existence of very efficient algorithms, such as the Viterbi (Viterbi, 1967) and Baum-Welch (Baum et al., 1970) algorithms. Both fall into the family

of the EM algorithms (Dempster et al., 1977), which constitute a powerful framework for solving *incomplete* data problems iteratively.

In the following, the mathematical formulation is derived using the same notation as defined in Section 2.2.1. The goal here is to find a model that maximizes the likelihood (or, alternatively, the log-likelihood) of an observation sample $\mathcal{X}$ given the model parameters $\boldsymbol{\lambda}$, that is:

$$\boldsymbol{\lambda}_{ML} = \arg\max_{\boldsymbol{\lambda}} f\left(\mathcal{X}|\boldsymbol{\lambda}\right) = \arg\max_{\boldsymbol{\lambda}} \log f\left(\mathcal{X}|\boldsymbol{\lambda}\right) \tag{2.8}$$

where the likelihood $f\left(\mathcal{X}|\boldsymbol{\lambda}\right)$ is defined in (2.5).

Let us define the *complete* data as $\mathcal{Y} = (\mathcal{X}, \mathcal{S}, \mathcal{L})$, where $\mathcal{X} = (x_1, \ldots, x_T)$ is the observed incomplete data, $\mathcal{S} = (s_1, \ldots, s_T)$ is the sequence of unobserved HMM states and $\mathcal{L} = (l_1, \ldots, l_T)$ is the sequence of unobserved output Gaussian component labels, with $s_t \in [1, N]$ and $l_t \in [1, K]$. The complete data log-likelihood can be expressed by:

$$\log f\left(\mathcal{Y}|\boldsymbol{\lambda}\right) = \log \pi_{s_0} + \sum_{t=1}^{T} \log a_{s_{t-1}, s_t} + \sum_{t=1}^{T} \log \omega_{s_t l_t} \mathcal{N}\left(x_t | \mu_{s_t l_t}, \Sigma_{s_t l_t}\right) \tag{2.9}$$

If the true sequences of HMM states and Gaussian component labels were known, the optimal model estimates could be calculated directly from (2.9). Of course, this is not the case. However, it can be shown that the incomplete data log-likelihood can be expressed in terms of two conditional expectation values calculated over the complete data log-likelihood, as follows:

$$\log f\left(\mathcal{X}|\boldsymbol{\lambda}\right) = Q(\boldsymbol{\lambda}, \hat{\boldsymbol{\lambda}}) - H(\boldsymbol{\lambda}, \hat{\boldsymbol{\lambda}}) \tag{2.10}$$

$$Q(\boldsymbol{\lambda}, \hat{\boldsymbol{\lambda}}) := E\left[\log f\left(\mathcal{Y}|\boldsymbol{\lambda}\right) \middle| \mathcal{X}, \hat{\boldsymbol{\lambda}}\right] \tag{2.11}$$

$$H(\boldsymbol{\lambda}, \hat{\boldsymbol{\lambda}}) := E\left[\log f\left(\mathcal{Y}|\mathcal{X}, \boldsymbol{\lambda}\right) \middle| \mathcal{X}, \hat{\boldsymbol{\lambda}}\right] \tag{2.12}$$

with

$$E\left[\log f(A|B)|C\right] = \sum_{a} \log f(A = a|B = b) \cdot P(A = a|C = c) \tag{2.13}$$

where $\hat{\boldsymbol{\lambda}}$ and $\boldsymbol{\lambda}$ denote two different sets of model parameters. $H(\cdot)$ represents the cross entropy function between two parameterized distributions and satisfies the divergence inequality:

$$H(\hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\lambda}}) \geq H(\boldsymbol{\lambda}, \hat{\boldsymbol{\lambda}}) \; \forall \; \boldsymbol{\lambda}, \hat{\boldsymbol{\lambda}}$$

From this latter and from the definition given in (2.10), it is straightforward to show that, whenever a new model fit $\boldsymbol{\lambda}$ satisfies $Q(\boldsymbol{\lambda}, \hat{\boldsymbol{\lambda}}) > Q(\hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\lambda}})$, it also satisfies $\log f\left(\mathcal{X}|\boldsymbol{\lambda}\right) > \log f(\mathcal{X}|\hat{\boldsymbol{\lambda}})$. This forms the basis of the EM algorithm: the incomplete data likelihood can be maximized indirectly, via maximization of a bound function, $Q(\cdot)$, which is commonly referred to as *auxiliary function*.

By definition the auxiliary function can be decomposed into two terms, which can be maximized independently (Juang, 1985):

$$Q(\boldsymbol{\lambda}, \hat{\boldsymbol{\lambda}}) = E\left[\log f(\mathcal{S}|\boldsymbol{\lambda})\middle|\mathcal{X}, \hat{\boldsymbol{\lambda}}\right] + E\left[\log f\left(\mathcal{X}, \mathcal{L}|\mathcal{S}, \boldsymbol{\lambda}\right)\middle|\mathcal{X}, \hat{\boldsymbol{\lambda}}\right] \tag{2.14}$$

The first term on the right hand side of (2.14) represents the Markov process, while the second one represents the output process.

Usually, acoustic model training is performed in two steps. At the first one, given the current model fit, an *expectation* over the state/frame alignment is performed. With this, the model parameters are updated by *maximization* of the auxiliary function.

**State/frame alignment** The state/frame alignment step, also called segmentation step, consists of associating regions of the audio stream to HMM states, given the current model fit. When manual transcriptions are available, a forced alignment is performed between the observation vectors and the models that represent the word sequences. Henceforth, we will refer to this case as *supervised* training. Alignment can be performed using the Baum-Welch algorithm (Baum et al., 1970), which allows the computation of the following probabilities:

$$\begin{aligned}
\psi_{it} &= P\left(s_t = i\middle|\mathcal{X}, \hat{\boldsymbol{\lambda}}\right) \text{ the probability of being at state } i \text{ at time } t, \text{ given} \\
&\quad \text{that model } \hat{\boldsymbol{\lambda}} \text{ generates } \mathcal{X}. \\
\xi_{ijt} &= P\left(s_{t-1} = i, s_t = j\middle|\mathcal{X}, \hat{\boldsymbol{\lambda}}\right) \text{ the probability of making a transition from} \\
&\quad \text{state } i \text{ to state } j \text{ at time } t, \text{ given that } \hat{\boldsymbol{\lambda}} \text{ generates } \mathcal{X}.
\end{aligned} \tag{2.15}$$

As an alternative, the Viterbi algorithm (Viterbi, 1967) can be used to perform a hard alignment decision, assigning only *zero* or *one* probabilities to $\psi_{it}$ and $\xi_{ijt}$.

If manual transcriptions are not available, an *unsupervised* training method can be applied (see Chapter 4). In this case, the state/frame alignment is performed between the audio stream and their hypothesized transcriptions, which are provided by an existing recognition system.

In both cases, supervised and unsupervised, *seed* or *bootstrap* models are required to provide a first segmentation guess. A straightforward initialization method is known as *flat start*. In this case, all the phoneme models are set equally, with parameters usually obtained via a global estimation over the entire training set (or a uniform sample of it). Another possible solution is to use phoneme models trained to another task, domain, accent or language as seed models. In such a case, it might be required to map the phone sets from the original task to the new one. Flat start usually requires less designing efforts, but may also lead to worse performance levels.

**Parameters update** Given the current state/frame alignment, the model parameters are updated via maximization of the auxiliary function. From (2.9) and (2.11), it can be shown that the auxiliary function can be decomposed as:

$$Q(\boldsymbol{\lambda}, \hat{\boldsymbol{\lambda}}) = Q_{\boldsymbol{\pi}}(\boldsymbol{\pi}, \hat{\boldsymbol{\lambda}}) + Q_{\boldsymbol{A}}(\boldsymbol{A}, \hat{\boldsymbol{\lambda}}) + Q_{\theta}(\theta, \hat{\boldsymbol{\lambda}}) \tag{2.16}$$

with

$$Q_{\boldsymbol{\pi}}(\boldsymbol{\pi}, \hat{\boldsymbol{\lambda}}) = \sum_{i=1}^{N} \psi_{i0} \log \pi_i \tag{2.17}$$

$$Q_{\boldsymbol{A}}(\boldsymbol{A}, \hat{\boldsymbol{\lambda}}) = \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{t=1}^{T} \xi_{ijt} \log a_{ij} \tag{2.18}$$

$$Q_{\theta}(\theta, \hat{\boldsymbol{\lambda}}) = \sum_{i=1}^{N} \sum_{k=1}^{K} \sum_{t=1}^{T} \gamma_{ikt} \log \omega_{ik} \mathcal{N}\left(x_t | \mu_{ik}, \Sigma_{ik}\right) \tag{2.19}$$

where $\psi_{it}$ and $\xi_{ijt}$ are defined in (2.15) and $\gamma_{ikt} = P(s_t = i, l_t = k | \mathcal{X}, \hat{\boldsymbol{\lambda}})$ is the probability of being at the $k$-th Gaussian component of mixture of state $i$ at time $t$, given that $\hat{\boldsymbol{\lambda}}$ generates $\mathcal{X}$. It can be computed as:

$$\gamma_{ikt} = \psi_{it} \frac{\hat{\omega}_{ik} \mathcal{N}\left(x_t \middle| \hat{\mu}_{ik}, \hat{\Sigma}_{ik}\right)}{\sum_{k'} \hat{\omega}_{ik'} \mathcal{N}\left(x_t \middle| \hat{\mu}_{ik'}, \hat{\Sigma}_{ik'}\right)} \tag{2.20}$$

The update equations for the model $\boldsymbol{\lambda}$ can be obtained taking the partial derivatives of the auxiliary function with respect to each of the parameters and setting them to zero. Observing the constraints $\sum_i \pi_i = 1$, $\sum_j a_{ij} = 1$ and $\sum_k \omega_{ik} = 1$, the following equations are obtained:

$$\pi_i = \psi_{i0} \tag{2.21}$$

$$a_{ij} = \frac{\sum_{t=1}^{T} \xi_{ijt}}{\sum_{j=1}^{N} \sum_{t=1}^{T} \xi_{ijt}} = \frac{\sum_{t=1}^{T} \xi_{ijt}}{\sum_{t=1}^{T} \psi_{i(t-1)}} \tag{2.22}$$

$$\omega_{ik} = \frac{\sum_{t=1}^{T} \gamma_{ikt}}{\sum_{k=1}^{K} \sum_{t=1}^{T} \gamma_{ikt}} = \frac{\sum_{t=1}^{T} \gamma_{ikt}}{\sum_{t=1}^{T} \psi_{it}} \tag{2.23}$$

$$\mu_{ik} = \frac{\sum_{t=1}^{T} \gamma_{ikt} x_t}{\sum_{t=1}^{T} \gamma_{ikt}} \tag{2.24}$$

$$\Sigma_{ik} = \frac{\sum_{t=1}^{T} \gamma_{ikt} (x_t - \mu_{ik})(x_t - \mu_{ik})^{\mathrm{T}}}{\sum_{t=1}^{T} \gamma_{ikt}} \tag{2.25}$$

### Maximum *a posteriori* estimation

Another generative training method used for acoustic modeling is the well-known MAP estimation (Gauvain and Lee, 1994). It assumes the existence of an appropriate *prior* distribution, denoted by $g(\boldsymbol{\lambda})$, over the model parameters $\boldsymbol{\lambda}$. Thus, given a sample $\mathcal{X}$

with p.d.f. $f(\mathcal{X}|\boldsymbol{\lambda})$, the MAP estimates of $\boldsymbol{\lambda}$ can be defined as the mode of the posterior probability, $g(\boldsymbol{\lambda}|\mathcal{X})$, or:

$$\boldsymbol{\lambda}_{MAP} = \arg\max_{\boldsymbol{\lambda}} g\left(\boldsymbol{\lambda}|\mathcal{X}\right) = \arg\max_{\boldsymbol{\lambda}} f\left(\mathcal{X}|\boldsymbol{\lambda}\right) g(\boldsymbol{\lambda}) \tag{2.26}$$

The maximization in (2.26) can also be performed using an EM iterative procedure. In such a case, the auxiliary function assumes the form $R(\boldsymbol{\lambda}, \hat{\boldsymbol{\lambda}}) = Q(\boldsymbol{\lambda}, \hat{\boldsymbol{\lambda}}) + \log g(\boldsymbol{\lambda})$. The MAP parameter estimates are defined as the mode of the function $\Psi(\boldsymbol{\lambda}, \hat{\boldsymbol{\lambda}}) = \exp R(\boldsymbol{\lambda}, \hat{\boldsymbol{\lambda}})$.

A careful choice for the prior distributions can substantially simplify the derivation of the parameter update equations. In Gauvain and Lee (1994), the prior knowledge was modeled by a product of Dirichlet and normal-Wishart densities. For the full mathematical formulation, refer to the original paper. As an example, the Gaussian component mean vectors can be re-estimated by:

$$\mu_{ik} = \frac{\tau_{ik} m_{ik} + \sum_{t=1}^{T} \gamma_{ikt} x_t}{\tau_{ik} + \sum_{t=1}^{T} \gamma_{ikt}} \tag{2.27}$$

where $m_{ik}$ and $\tau_k$ are two parameters of the normal-Wishart prior distribution. In particular, $m_{ik}$ can be seen as a prior $d$-dimensional mean vector, while $\tau_{ik} > 0$ is a meta-parameter, which is usually set manually or obtained via some optimization algorithm. By inspection of (2.27), it can be seen that the mean vectors are updated by a weighted sum of the prior parameters and the observation vectors.

The inherent use of MAP estimation is for adaptation. It can be used to adapt a baseline acoustic model (which provides the prior distribution of the parameters) to a specific task, domain, genre or speaker. Another common application is on Gaussian mixture parameter smoothing, a technique employed to improve the model generalization capability. In such a case, the prior distribution over the model parameters can be provided by a global Gaussian density component.

### Discriminative training methods

Different from generative training, the aim of discriminative training methods is the direct or indirect maximization of the recognition accuracy. These methods rely on the optimization of some suitable classification measure to increase the separability between the *correct* and *competing* sequences of class labels.

Some examples of discriminative acoustic modeling methods used for speech recognition tasks are the maximum mutual information estimation (MMIE) (Bahl et al., 1986; Woodland and Povey, 2002), the minimum classification error estimation (Juang and Katagiri, 1992; Juang et al., 1997) and the minimum phone error estimation (Povey and Woodland, 2002). In particular, the MMIE method has been used in some of the systems developed throughout this work. In this case, the model estimates are obtained by:

$$\boldsymbol{\lambda}_{MMI} = \arg\max_{\boldsymbol{\lambda}} \frac{f(\mathcal{X}|\boldsymbol{\lambda}_{\mathcal{W}}) \cdot P(\mathcal{W})}{\sum_{\widetilde{\mathcal{W}}} f(\mathcal{X}|\boldsymbol{\lambda}_{\widetilde{\mathcal{W}}}) \cdot P(\widetilde{\mathcal{W}})} \tag{2.28}$$

where $\boldsymbol{\lambda}_{\mathcal{W}}$ denotes the model that represents the word sequence $\mathcal{W}$ and $P(\mathcal{W})$ denotes a prior distribution over the word sequences, provided by a language model. In the numerator, the correct word sequence is considered, while, in the denominator, the summation is performed over all possible sequences. In practice, for large vocabulary tasks, the denominator is approximated by the summation over the most likely competing sequences, which can be obtained from multiple decoding hypotheses provided by the speech recognizer (Valtchev et al., 1996; Woodland and Povey, 2002).

The MMI estimates can be obtained using an extended version of the Baum-Welch algorithm (Gopalakrishnan et al., 1991). During the estimation process, insofar as possible, the likelihood in the numerator is maximized at the same time as the likelihood of the competing hypotheses is minimized. Furthermore, in order to favor acoustic discrimination, the prior probability $P(\mathcal{W})$ is usually provided by a weakened language model (Schlüter et al., 1999).

## 2.2.3. Model structure

So far, the mathematical framework has been described without taking into consideration any specific acoustic model structure. In the case of large vocabulary recognition, the inclusion of certain constraints is essential to avoid sparsity problems and make the estimation and decoding processes tractable. In this section, some well-known acoustic modeling practices are described.

### The modeling unit

In principle, a single HMM could be used to model a whole word or even an entire utterance. However, for LVCSR tasks, it would be impractical. Usually, the acoustic phenomena are modeled at the phoneme level (or any other suitable sub-word unit). Models for longer contexts (words, utterances) are obtained by composition of the phoneme models, sticking them together as a string.

Consequently, the number of possible state sequences would increase exponentially with the number of phonemes in an utterance, heavily increasing the complexity to compute the data likelihood, given by (2.5). Fortunately, different elements exist to restrain the possible sequences and make this problem tractable. For instance, the pronunciation dictionary is responsible for limiting the number of states considered between two distinct phoneme models. Inside a single phoneme unit, the model topology defines the number of possible paths that can be followed. In Figure 2.2, an example of topology is shown, a left-to-right model. It contains three emitting states and only allows two types of transitions: to the current state itself, or to the state at its "right" position.

Another common approximation performed to reduce computation time is to consider only the most likely state sequence, instead of summing over all possible sequences, what leads to:

$$f\left(\mathcal{X}|\boldsymbol{\lambda}\right) \approx \max_{\mathcal{S}} \left\{ \pi_{s_0} \prod_{t=1}^{T} a_{s_{t-1},s_t} f\left(x_t|\theta_{s_t}\right) \right\} \tag{2.29}$$

In addition, different heuristics exist to discard low probability paths in order to

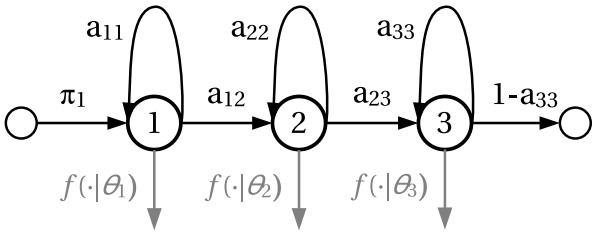



Figure 2.2.: Left-to-right 3-state hidden Markov model.

further reduce computational complexity required for decoding purposes. For this topic, the reader may refer to Knill and Young (1997).

### Context-dependent models

From the fundamental assumptions of the HMM-based framework, no dependency exists between two observation samples. However, it is well-known that the realization of a certain phoneme is highly dependent on the realization of the preceding and the following phonemes due to physical articulatory constraints. In order to capture this *co-articulation* effect, phones are commonly modeled depending on the context. In other words, a certain phone, /a/ for instance, could be represented by different context-dependent triphone models, such as /b,a,s/, /m,a,r/, /t,a,g/, etc., in the acoustic models inventory.

Unfortunately, many of the phone contexts might be underrepresented or not represented at all in the training data, making them impossible to be reliably modeled. A possible solution to overcome this problem is to *backoff* from underrepresented contexts to less-specific contexts.

Another possible solution to this sparsity problem is to *tie* some of the model parameters, forcing them to share a number of observation vectors during the estimation process. Tying can be performed at different levels, such as phones (Lee and Hon, 1988), states (Young and Woodland, 1993), mixtures (Huang and Jack, 1989; Bellegarda and Nahamoo, 1990) or features (Takahashi and Sagayama, 1995). The acoustic models used in this thesis have been trained with a combination of model backoff and state tying techniques.

### Tied-state models

States tying is the procedure of constraining acoustically similar states to share the same output observation distribution density, reducing the effect of sparsity at the same time as preserving some context dependency. Data-driven or phonetic approaches can be used to select which states are more likely to be pooled together.

In the data-driven case, the choice for merging relies on acoustic related measures, based on some distance between the Gaussian parameters that represent the state distributions. Although this approach is straightforward and does not require any external source of knowledge, it is unable to make clustering decisions for unseen contexts.

In the other case, the merging decision relies on the construction of a phonetic decision tree, containing binary *yes/no* questions to decide the branch to be followed. States falling on the same leaf are pooled together. The questions are usually defined based on the phonetic characteristics of the left and right contexts, which clearly requires the use of

an external source of knowledge. More detailed information about the construction and usage of decision tree methods can be found in Bahl et al. (1991) and Young et al. (1994).

## 2.2.4. Feature extraction

The main assumption of the speech recognition task is that the audio signal encodes the acoustic representation of an underlying sequence of words. However, together with the relevant linguistic information, many noisy components exist in the speech signal. Here, the term *noisy* is employed in a large sense to qualify any non-linguistic event, such as speaker identity, speaker emotional state, environment condition, background noise and so on. In this context, the aim of signal analysis is to extract linguistically related acoustic features, being, as best as possible, invariant to non-linguistic phenomena.

### Short-term features

Feature extraction in current state-of-the-art speech recognition systems is based on power spectrum analysis. Typically, the feature vectors are extracted after a processing chain relying uniquely on pre-defined mathematical operations, including windowing, Fourier analysis, warping, filtering and other transforms. Usually, a *short-term* window ranging on about 10 to 40 milliseconds is employed.

The two most widely used processing chains are based on principles of psycho-acoustics, yielding to the well-known Mel-frequency cepstral coefficients (Davis and Mermelstein, 1980) and perceptual linear predictive (Hermansky, 1990) cepstral coefficients. Besides their correlation with human hearing perception and good discrimination, these features present an additional advantage: the channel variation effect can be removed by simple subtraction of the global cepstral mean and variance from the input vectors. This probably constitutes the most straightforward manner to avoid undesirable (non-linguistic) events.

The acoustic feature vector is usually formed by the concatenation of the aforementioned cepstral-like features, the log-energy of the signal and, optionally, pitch features. Furthermore, in order to emulate acoustic dynamics, first and second derivatives of all the coefficients are included in the feature vectors.

### Discriminative features

In the recent years, there has been a growing interest to increase the discriminative power of acoustic features. This task can be done, for instance, by applying a discriminatively trained linear transform to the feature vector (Povey et al., 2005; Zhang et al., 2006). Another approach consists of extracting feature vectors from a discriminative classifier, such as a MLP. This latter approach is investigated in this work.

MLP features have been successfully used in many LVCSR tasks (Ellis et al., 2001; Zhu et al., 2005; Fousek et al., 2008a). When used in combination with short-term features, they lead to substantial gains in recognition accuracy (Fousek et al., 2008b). The MLP feature extraction chain can be summarized as follows:

1. A feature vector, henceforth *raw* feature vector, is extracted from the audio signal via a power spectrum analysis. This vector usually covers a wide temporal context

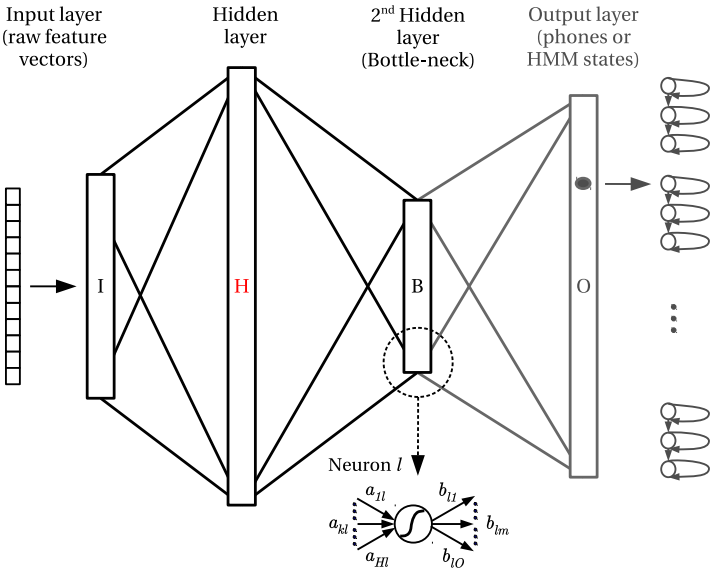



Figure 2.3.: A 4-layer bottleneck multi-layer perceptron (MLP) architecture. Features are extracted from the third-layer (the bottleneck).

in order to increase complementarity between MLP and short-term features and, thus, favor feature combination.

2. The raw vector is processed by the MLP. A feature vector is extracted from either, the output (Hermansky et al., 2000) or a hidden (the *bottleneck*) (Grézl et al., 2007) layer.

3. A decorrelation transform is further applied to the extracted vector to result in the final MLP feature vector.

Different from the processing chain described in the last section, MLP feature extraction stands on a formerly trained neural network model. As an example, a 4-layer bottleneck MLP is shown in Figure 2.3. Typically, MLPs are trained to estimate the posterior probabilities of HMM phone models or HMM states, what evidently requires a model/frame (or state/frame) alignment, such as described in Section 2.2.2. Similarly to HMM training, a forced-alignment can be performed if manual transcriptions are available. If it is not the case, a proposed solution is to extract features from MLPs trained for another language (Stolcke et al., 2006; Tóth et al., 2008) or trained for multiple languages (Grezl et al., 2011; Vu et al., 2012). In this thesis, we have also explored training MLPs with automatic transcriptions provided by the speech recognizer (see Chapter 5).

### 2.2.5. Hybrid MLP/HMM acoustic models

In the previous sections, we focused on the description of the HMM-based acoustic models with GMM output probability densities. An alternative architecture that has been gaining a lot of popularity in the last few years is based on hybrid MLP/HMM architecture. In such a model, the HMM state probabilities are given by the output layer of a MLP neural network.

The HMM/MLP architecture was introduced for automatic speech recognition in Morgan and Bourlard (1995). Due to the increase in power of easily available hardware and development of new training methods, the use of MLPs with several layers, the so-called deep neural networks (DNNs), was shown to substantially outperform HMM/GMM based acoustic models for a variety of tasks (Mohamed et al., 2011, 2012; Hinton et al., 2012). Because of its proved strength, the (deep) MLP/HMM architecture has risen to the new state-of-the-art in terms of acoustic modeling. As a drawback, DNNs are slower to train in comparison to HMM/GMM models.

Despite the recent advances reported on the use of neural networks, the methods proposed in this thesis are assessed in a HMM/GMM based system.

## 2.3. Language modeling

In the previous section, state-of-the-art acoustic modeling techniques have been introduced. This section focus on general aspects of another main component of ASR systems, the language model. For a review of this topic, the reader may refer, for example, to Rosenfeld (2000) or Goodman (2001).

This section begins with a definition of language modeling and motivation purposes. Next, the state-of-the-art $n$-gram based approach is presented. The last subsection presents other language modeling techniques, especially those based on continuous space vector representations.

### 2.3.1. Definition

The role of the language model is to capture regularities in the underlying process of natural language generation, providing a prior knowledge of the language. Approaches for language modeling can be divided into two main groups: deterministic or stochastic. The deterministic approaches rely on the construction of a formal grammar, with rules designed based on the expert knowledge about the language. Conversely, stochastic (or statistical) language model parameters are learned via statistical methods from data observed in a training corpus, although the structure of the model and parameters to be learned are still a matter of expert's choice. The scope of this thesis is limited to statistical language modeling.

Statistical language models have been first used for speech recognition in the 1980's (Bahl et al., 1983; Jelinek, 1985) and have been proven useful in a large variety of natural language processing applications, like machine translation (Brown et al., 1990), optical character recognition (Hull, 1992) or handwriting recognition (Srihari and Baltus, 1992). The language model should be able to provide a *prior* probability to any possible sequence of words that can be represented in the *word vocabulary* space, denoted by $\mathcal{V}$. The prior probability of a particular sequence of $M$ words denoted by $\mathcal{W} = (w_1, \ldots, w_M)$, with $w_m \in \mathcal{V}$, can be written as:

$$P(\mathcal{W}) = P(w_1, \ldots, w_M) = \prod_{m=1}^{M} P(w_m | w_1, \ldots, w_{m-1}) \tag{2.30}$$

From (2.30), it follows that the probability of a word string can be calculated as the product of the probabilities of observing a particular word at the $m$-th position given its *context*, that is, the preceding $m-1$ words.

For LVCSR tasks, the model as presented in (2.30) can neither reliably nor efficiently be estimated from any existing training corpus due to the high dimensionality of the problem. For instance, it would be necessary to estimate about $10^{80}$ parameters to build a language model covering contexts up to 15 words and with a vocabulary containing 100 thousands of words. These problems were the main motivation for the formulation of the so-called $n$-gram language modeling approach, which is described in Section 2.3.2. Before, we describe the most common measures used to evaluate language model performances.

#### Evaluation measures

The quality of a language model can be measured in terms of the (log-) likelihood of a development data set. Given a model $p_{LM}$ and a test data with $M$ samples, $\mathcal{W} = (w_1, \dots w_M, )$, the log-likelihood can be calculated as:

$$\mathcal{L}(p_{LM}, \mathcal{W}) = \sum_{m=1}^{M} \log p_{LM}(w_m) \tag{2.31}$$

As an alternative, results of language modeling techniques are usually reported in terms of perplexity, which is calculated by:

$$PP(p_{LM}, \mathcal{W}) = 2^{H(p_{LM}, \mathcal{W})} \tag{2.32}$$

where $H(p_{LM}, \mathcal{W})$ is the empirical estimate of the cross-entropy:

$$H(p_{LM}, \mathcal{W}) = -\frac{1}{M} \sum_{m=1}^{M} \log_2 p_{LM}(w_m) \tag{2.33}$$

Perplexity can be interpreted as a measure of a per word average branching factor of the language regarding the model. Roughly speaking, the lower the perplexity on test data, the less confusion appears, and, therefore, the better is the model for that data.

Although some correlation exists between perplexity and speech recognition performance, a reduction in perplexity does not always lead to a reduction in the recognition error rate. In general, only relative gains in perplexity of more than 10% can be considered significant (Rosenfeld, 2000). Furthermore, perplexity is only meaningful for comparing models that have the same word vocabulary list (this is not the case of likelihood). Despite that, perplexity is still the principal metric used to assess the quality of language models.

### 2.3.2. $N$-gram based language models

The $n$-gram based framework has been the most widely used for language modeling for the past 30 years, being omnipresent in the state-of-the-art speech recognition systems either

as unique solution, or in combination with other methods. The $n$-gram LM is derived from an approximation to the joint probability of the word sequences, assuming that the most relevant context information is encoded by a short span *history*, that is, the $(n-1)$ preceding words. In other words, it models the language as a $(n-1)$-th order Markov process. The word-based $n$-gram language model can be represented as follows:

$$P(\mathcal{W}) \approx \prod_{m=1}^{M} P(w_m | w_{m-n+1}, \ldots, w_{m-1}) \tag{2.34}$$

In practice, a history of at most 3 words (that is $n = 4$, or 4-gram language model) is used in most of the speech recognition tasks. In an empirical study, Goodman (2001) claimed that no significant improvements could be observed using contexts longer than 4 words (5-grams). Despite the (heavy) approximation performed, the $n$-gram parameters estimation is still a sparse data problem, which is commonly treated using a *smoothing* algorithm.

### The smoothing issue

As a desirable characteristic, an $n$-gram based language model should be able to assign a non-zero probability to any possible word sequence with length $n$ that can be generated from the vocabulary. A natural choice to estimate the $n$-gram conditional probabilities is to maximize the likelihood of the training data. Given a word $w$ and its history $h$ with length $(n-1)$, and denoting $(h, w)$ a word string formed by $h$ followed by $w$, the maximum likelihood (ML) estimates can be obtained by:

$$p_{ML}(w|h) = \frac{C(h,w)}{\sum_{w_i} C(h,w_i)} \tag{2.35}$$

where $C(\cdot)$ denotes the string *counts*, that is, the number of times a sequence is observed in the training corpus. Unfortunately, the ML criterion leads to poor estimates, assigning zero probabilities to any *unseen* sequence, although they may occur in the test data. Increasing the training corpus is not a sufficient solution, since for common LVCSR applications the limits of tractability would be reached before a full $n$-gram coverage.

One of the methods to deal with this problem is to smooth the model probabilities. The smoothing techniques not only address the zero probability issue, but also help to improve the performance of the model in general (Chen and Goodman, 1999). In this later work, Chen and Goodman perform an empirical comparison of some popular smoothing approaches. Among them, the KN (Kneser and Ney, 1995) smoothing algorithm seems to be the most successfully used over the past years for many different speech recognition tasks. As other *discounting* methods (Katz, 1987; Ney and Essen, 1991; Ney et al., 1994), this algorithm deducts certain probability masses from the ML estimates of the more frequent events and redistributes them to the less frequent and unseen events. The *interpolated* KN model (Ney et al., 1997) can be represented as follows:

$$p_{KN}(w|h) = \frac{\max\{C(h,w) - D, 0\}}{\sum_{w_i} C(h,w_i)} + \frac{D \cdot N_{1+}(h,\bullet)}{\sum_{w_i} C(h,w_i)} p_{KN}(w|g) \tag{2.36}$$

with

$$p_{KN}(w|g) = \frac{N_{1+}(\bullet, g, w)}{N_{1+}(\bullet, g, \bullet)} \tag{2.37}$$

where $D$ is a fixed discount parameter and $g$ is a less-specific (shorter) history obtained by removing the farthest word from the history. The functions $N_{1+}(\cdot)$ give the number of unique words that appear in context with a certain string. More specifically, denoting $a$ as an arbitrary word prefix, we can write these functions as follows:

$$N_{1+}(h, \bullet) = |w_i : C(h, w_i) > 0| \tag{2.38}$$

$$N_{1+}(\bullet, g, w) = |a : C(a, g, w) > 0| \tag{2.39}$$

$$N_{1+}(\bullet, g, \bullet) = |(a, w_i) : C(a, g, w_i) > 0| = \sum_{w_i} N_{1+}(\bullet, g, w_i) \tag{2.40}$$

In practice, $p_{KN}(w|g)$ (2.37) also need to be smoothed, leading to:

$$p_{KN}(w|g) = \frac{\max\{N_{1+}(\bullet, g, w) - D_n, 0\}}{N_{1+}(\bullet, g, \bullet)} + \frac{D_n \cdot N_{1+}(g, \bullet)}{N_{1+}(\bullet, g, \bullet)} p_{KN}(w|\hat{g}) \tag{2.41}$$

where the index $n$ in $D_n$ denotes the use of a specific discount parameter for each $n$-gram level and $\hat{g}$ denotes a history shorter than $g$. The discount parameters can be defined arbitrarily or obtained via optimization. Ney et al. (1994) proposed the use of the following formula:

$$D_n = \frac{n_1}{n_1 + 2 \cdot n_2} \tag{2.42}$$

where $n_1$ and $n_2$ denotes, respectively, the number of $n$-grams with counts exactly equal to one (commonly referred to as *singletons*) or two.

Only one discount parameter is considered in the original KN smoothing. Chen and Goodman (1999) proposed the use of three different discounts (applied to events occurring one, two and three or more times), reporting gains in perplexity.

The interpolated model represented in (2.36) is slightly different from the *backoff* model originally proposed by Kneser and Ney (1995). In the interpolated model, a portion of the less-specific distribution, $p_{KN}(w|g)$, is always used. In the backoff model, this distribution is only considered if the sequence $hw$ has not been observed in the training data.

Smoothing is an essential step for $n$-gram language modeling. Nevertheless, it does not exempt the model from other weaknesses, which are discussed hereafter together with refinements that have been proposed to deal with them.

### Shortcomings

The strength of word-based $n$-gram LM may be related to its good coverage and performance achieved, associated to its simplicity. Compared to other models, it is faster to train and more convenient for usage during decoding.

Despite that, it is not difficult to identify some shortcomings of this approach. First, it assumes only a local dependency, failing to generalize well for more structured sentences or reordered sentences. For instance, let us consider the prediction of the word *reduced* in the two phrases below:

1. incoming taxes were reduced as a result of the new financial plan

2. incoming taxes, as a result of the new financial plan, were reduced

Assuming a two words history, the probability of *reduced* occurring would be quite different in these examples, although they are semantically equivalent. It is natural to assume the string *taxes were reduced* more likely to occur than *plan were reduced*.

Not only semantic, but also syntactic relations are neglected in the word-based $n$-gram approach, which assumes a simple word level dependency. Including such higher level information in the model is one of the axes of research in the language modeling field (Rosenfeld, 2000).

Some attempts have been made to deal with the aforementioned shortcomings. For instance, Huang et al. (1993) used skipping $n$-grams to capture long-span dependencies while keeping a tractable history size. In another direction, Brown et al. (1992) proposed to organize words in clusters to alleviate the data sparsity problem and establish some relation between them. This technique can also be used to include linguistic knowledge in the model by, for instance, defining the classes based on part-of-speech tags. Such an information can also be explicitly incorporated into the factored LM (Bilmes and Kirchhoff, 2003), since it allows the inclusion of arbitrary features. Depending on the features used, this later model can generalize both, word and class-based $n$-gram models.

The aforementioned methods are somewhat a derivation of the word-based $n$-gram LM, inheriting its strengths but also some of its weaknesses. In particular, $n$-gram based models suffer from the lack of structure, which is a suitable condition to improve generalization power. Furthermore, the discrete representation of the model space (the vocabulary) associated with its large dimensionality may prevent the estimation of reliable parameters. In the following section, an overview of some language modeling approaches dealing with these problems is presented.

### 2.3.3. Other language modeling approaches

Like factored LMs, the possibility to incorporate arbitrary features is also an intrinsic characteristic of exponential language models (Della Pietra et al., 1992; Rosenfeld, 1996; Berger et al., 1996; Chen, 2009). This flexibility allows to tackle some of the $n$-gram based models weaknesses by including, for example, longer context dependencies and morphological or syntactic information. In addition, this approach can benefit from methods to select the most useful features to be included into the model. As a main drawback, the use and the estimation of exponential models are computationally challenging tasks.

Differently from the approaches presented so far, an underlying structure is present in decision tree based language models (Bahl et al., 1989). This approach relies on partitioning the history space at each node of a tree by asking arbitrary questions about the history. During the growing procedure, the question maximizing the likelihood of the training data is selected at each node. Data falling onto the leaves are used to calculate

the model probabilities. Due to the greedy characteristic of the growing algorithm, finding the optimal tree can be a struggle, limiting the success of the approach. In Xu and Jelinek (2004), the authors have proposed to combine randomly grown decision trees as an alternative solution to avoid local optima, reporting gains in terms of perplexity and speech recognition performance over a baseline $n$-gram LM.

Another method favoring the use of large-span context dependencies is based on the latent semantic analysis (LSA) paradigm and has been proposed by Bellegarda (1997). It allows the representation of words and sequence of words into a continuous space, which is assumed to encode latent semantic information. The probability of a word given a context (an entire document for instance) can be inferred based on the distance between their vectors projected into this space. The LSA language modeling approach relies on the concept of "bag-of-words", ignoring the order in which words appear in the document. Despite the relative simplicity of this assumption, the use of the LSA method can nonetheless help to improve the system recognition performance when combined with the $n$-gram approach due to their complementary characteristic (Bellegarda, 2000).

In fact, relying on a continuous vector representation and providing a structure to the language model seems to be the new trend in the language modeling research field. This combination of factors helps to alleviate from sparse data problems and improve the generalization power of the model. In Sarikaya et al. (2009), for example, the authors have proposed a HMM/GMM-like LM trained with feature vectors extracted from a LSA space. However, the approaches that have been most widely investigated in the recent years are based on neural network models.

### Neural network language models

Neural networks have been used for statistical language modeling since the beginning of the 2000's (Bengio et al., 2000, 2003), and have been successfully applied to speech recognition (Schwenk and Gauvain, 2002; Schwenk, 2007). These reported works used a similar model topology based on *feed-forward* neural networks, such as exemplified in Figure 2.4. This model is structured in multiple layers, with each internal node (neuron) having a sigmoidal activation function and being fully connected to the neighbor layers by means of weighted links.

The model in Figure 2.4 contains four layers. The *input* layer is used to inform the word history, which is represented by $(n-1)$ word vectors, each having the dimension of the vocabulary size $|\mathcal{V}|$ and being encoded as a one-to-n vector, that is, having just one element of the vector set to 1 and the remaining to 0. In the second (the *projection*) layer, each of the word vectors is projected into a $P$-dimensional continuous space. Given the input vector coding, projection can be performed as a simple look-up operation over the $|\mathcal{V}| \times P$ projection matrix. The third (the *hidden*) layer, used for feature space transform, has a dimension $H$. Finally, the *output* layer, with dimension $N <= |\mathcal{V}|$, implements a *softmax* normalization function to compute the word posterior probabilities.

Feed-forward neural network language model (NNLM) training can be performed using a standard *back-propagation* algorithm. This is a computationally expensive task, mainly due to the softmax normalization performed over the large dimensional output vector. A possible way to mitigate this problem is to limit the output vocabulary to a shortlist of most frequent words (Schwenk and Gauvain, 2002). In such a case, the probabilities of the remaining words are obtained from a backoff $n$-gram LM. More recently,

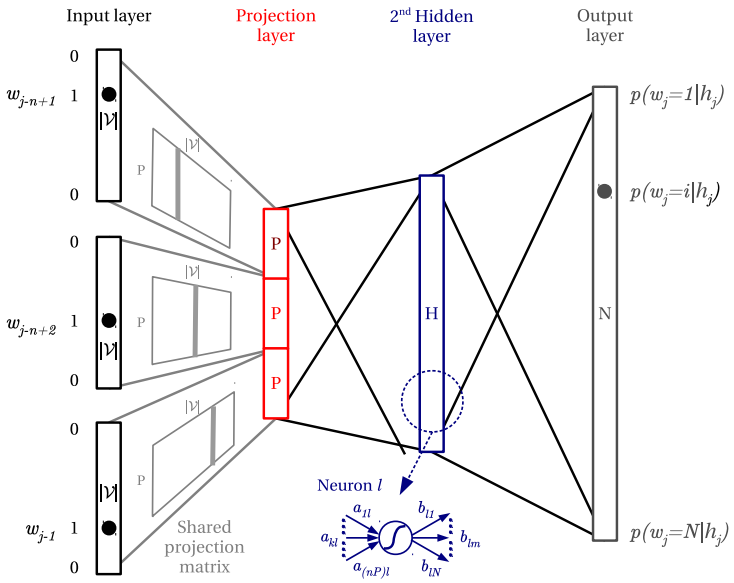



Figure 2.4.: Feed-forward neural network language model architecture.

Le et al. (2011) have proposed the use of a tree-like structured output layer, allowing posterior probabilities to be calculated for the entire vocabulary with a small increase of computational effort. For the same reasons, the use of NNLMs for full decoding is also prohibitive, limiting their usage on rescoring the decoding hypotheses (see Section 2.4).

In common with $n$-gram based models, feed-forward NNLMs limit the history to a few $(n-1)$ preceding words. However, since NNLMs suffer much less with data sparsity problems, longer contexts ($n = 6$, for instance) can be applied with a minor increase of computational complexity (Le et al., 2013). An alternative model, able to encode contexts with arbitrary (and unknown) lengths, is the *recurrent* NNLM (Mikolov et al., 2010), which uses a recursive connection between the hidden and the input layers.

An empirical comparison of different language modeling techniques, including the aforementioned NNLMs, can be found in Mikolov et al. (2011).

## 2.4. Decoding

The goal of the decoding (or search) algorithm is to find the optimal word sequence that represents an observed audio signal, considering the information provided by acoustic, language and pronunciation models. In today's state-of-the-art ASR systems, this large dimensional optimization problem, which is defined in (2.2), is solved using a multi-pass decoding strategy.

Prior to word recognition, an audio partitioning procedure is usually applied. This step is responsible for purging non-speech data from the audio file, dividing it into shorter segments, and clustering segments into homogeneous regions, each assumed to refer to a speaker in a specific acoustic condition (background and channel). Different unsupervised approaches exist to perform this task (like Gauvain et al. (1998); Hain et al. (1998)).

As a major advantage, audio partitioning favors the use of cluster-based acoustic model adaptation techniques, which can substantially improve the recognition perfor-

mance. Other benefits include the reduction of computation time required for decoding and the possibility to extract additional non-linguistic information from the audio stream, such as speech turns or speakers identities.

Once the data is segmented, feature vectors are extracted from the continuous stream (see Section 2.2.4). Optionally, unsupervised methods can be further applied to adapt the speaker independent acoustic model to the specific speaker-based cluster or to select a model that better represents the data. Each segment is, then, decoded independently.

### 2.4.1. Word recognizer

The search algorithm used to solve (2.2) is commonly based on dynamic programming approaches, such as the Viterbi (Viterbi, 1967) or the forward-backward (Baum et al., 1970) algorithms. Although, due to the huge search space involved in LVCSR tasks, pruning methods need to be applied to render the decoding problem manageable. A comprehensive description of techniques employed for decoding can be found in Knill and Young (1997).

Furthermore, decoding is usually performed in various passes as a compromise between computational cost and recognition accuracy. A general solution is to use less accurate (and smaller) models in a first step to generate multiple decoding hypotheses and successively re-evaluate them with more accurate (and bigger) models. The multiple hypotheses can be represented in different forms, like $N$-best lists, lattices or confusion networks. These structures, which are shown in Figure 2.5, can be used to represent the output of the ASR system as well.

An $N$-best list, as the name suggests, is a list of the most likely sentence hypotheses ordered by their likelihood scores. Since each hypothesis is fully represented, $N$-best lists may contain a lot of redundancy, what usually limits the number of competing hypotheses that can be considered. Conversely, a lattice is a compact representation of the searching space and can encode a larger number of hypotheses. It consists of an acyclic graph where nodes represent time stamps and edges represent hypothesized words with their associated likelihood scores. Finally, a confusion network is also an acyclic graph with the particularity that edges representing competing words always have the same source node and the same destination node. Different from lattices, confusion networks contain edges with posterior probability scores and nodes with approximate timing information. Confusion networks can be constructed from lattices (Mangu et al., 1999) or from $N$-best lists.

### 2.4.2. Evaluation measures

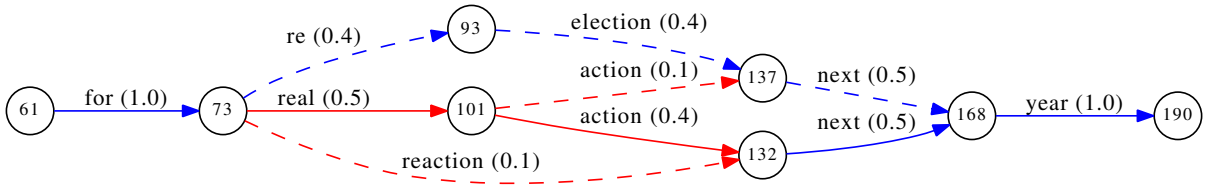
ASR systems performance is commonly reported in terms of word error rate (WER). After aligning the hypothesis and the reference transcriptions using a dynamic programming approach, three types of recognition errors can be identified: substitutions (S), insertions (I) and deletions (D). These errors are used to compute the WER, as follows:

$$WER = \frac{S + D + I}{N} \tag{2.43}$$

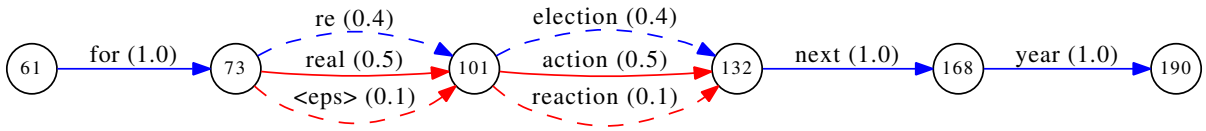
where $N$ denotes the number of words in the reference.

```
score=0.100:  for[61,73,1] real[73,101,0.5] action[101,132,0.4] next[132,168,0.5] year[168,190,1]
score=0.080:  for[61,73,1] re[73,93,0.4] election[93,137,0.4] next[137,168,0.5] year[168,190,1]
score=0.050:  for[61,73,1] reaction[73,132,0.1] next[132,168,0.5] year[168,190,1]
score=0.025:  for[61,73,1] real[73,101,0.5] action[101,137,0.1] next[137,168,0.5] year[168,190,1]
```

(a)




(b)




(c)

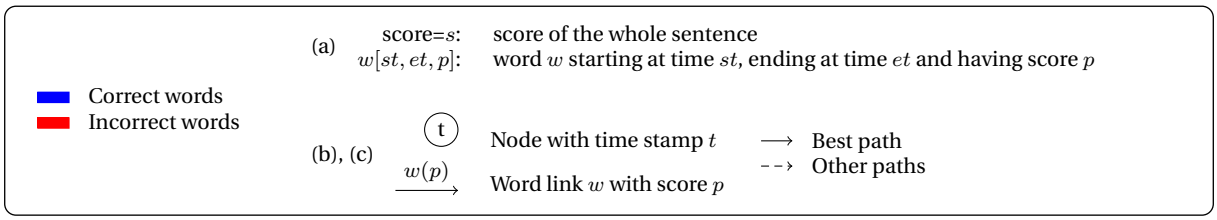



Figure 2.5.: Example of (a) an N-best list, (b) a decoding lattice and (c) a confusion network for the same speech segment.

Depending on the application and language being evaluated, other related measures, like character error rate or morpheme error rate, may be useful as well. In other cases, it might be interesting to report a lower bound error measure to identify strengths and weaknesses of a certain system. An example of such a measure can be obtained by aligning the reference transcriptions with multiple recognition hypotheses ($N$-best lists, lattices or confusion networks). In this case, errors are computed only when none of the hypotheses (partially) matches the reference.

Besides text transcriptions, time stamps and speaker identities, recognition systems are often capable to associate a confidence score to each hypothesized word. Among other applications, such a measure can be used to detect possible misrecognized or out-of-vocabulary words (Young, 1994; Allauzen, 2007) and improve the performance of unsupervised acoustic model training (Zavaliagkos and Colthurst, 1998; Gollan et al., 2007) or adaptation algorithms (Pitz et al., 2000; Padmanabhan et al., 2000; Gollan and Bacchiani, 2008).

A review of various approaches used to estimate confidence measures can be found in Jiang (2005). In Wessel et al. (2001), different measures for LVCSR are presented and empirically compared. In this latter reported work, the best results have been obtained with word posterior probabilities, especially when they are computed from word lattices.

# Chapter 3

# Baseline Automatic Speech Recognition System

All experiments of this work have been performed using the LIMSI toolkit (Gauvain et al., 2002) briefly described in the next section. The following two sections describe the corpora used in this work as well as the baseline systems for Portuguese and English. The baseline Portuguese system was developed as a part of this thesis work, while the baseline English system is the one developed and described in Vergyri et al. (2010).

## 3.1. The LIMSI broadcast recognition system

This section provides a brief description of the baseline LIMSI ASR systems targeting broadcast data, which was first detailed in Gauvain et al. (2002). The systems use HMMs with Gaussian mixture model state output densities and backoff $n$-gram language models as described below.

### 3.1.1. Acoustic modeling

The LIMSI ASR system makes use of HMMs with GMM state output observations for acoustic modeling. Each context dependent phone is modeled by a left-to-right triphone HMM, with state GMMs containing 32 components.

Special symbols are included in the phone inventory to explicitly represent breath noise, hesitation and silence. Breath and hesitation units are modeled in the same way as regular phones (context dependent triphone HMMs). Conversely, silence is modeled by a single GMM, generally containing 1024 components.

To avoid sparseness, model backoff and state-tying are used. Phone contexts insufficiently represented in the training data are backed-off to less-specific contexts. State-tying is done across the same state position of a phone and relies on a classification and regression tree (CART). For each model, the tree is constructed based on questions related to the characteristics of the left and right context phones: for instance, whether it is a consonant or a vowel, if it is voiced or voiceless, word position (word beginning, internal, final) and so on. States falling on the same tree leaf are tied together.

Acoustic feature extraction is based on perceptual linear predictive (PLP) analysis (Hermansky, 1990). Feature vectors are extracted every 10ms using a power spectrum

analysis over a 30ms sliding window. For each window, 12 Mel-frequency cepstral coefficients (MFCC) are computed. These coefficients are normalized on a speaker segment cluster basis using mean and variance removal. A 39-dimensional feature vector is obtained by concatenating the 12 cepstrum coefficients with the log-energy, along with their first and second derivatives. Generally, a 3-dimensional pitch feature vector (pitch with its first and second derivatives) is concatenated to the previous vector, resulting in a 42-dimensional feature vector.

### 3.1.2. Language modeling

Although the most important source of texts for language modeling are the manual transcriptions, these are difficult to obtain. So language modeling makes use of other written sources that are abundant, cheap and can be easily gathered from the Web, such as news, blogs or closed captions. However, texts coming from such sources have not been produced for LM training, but for the consumption by human readers. So, several pre-processing and normalization steps are required (Adda et al., 1997; Adda and Adda-Decker, 1997).

A variety of normalization steps are typically carried out, such as converting text encoding, filtering out HTML tags and titles or separating the texts into one sentence per line. These steps are not strictly dependent on the language, but depend on the source from which texts have been collected. Another essential normalization procedure consists of converting numerical patterns (cardinals, ordinals, dates, currency, etc.) and abbreviations to be close to their spoken forms (Adda et al., 1997).

The ASR systems rely on $n$-gram based language models. Prior to LM estimation a word list vocabulary is selected via the interpolation of unigram component models. First, a component unigram LM is estimated on each source. These models are linearly interpolated with coefficients automatically chosen in order to maximize the likelihood of a development set. The vocabulary is formed by the words with the highest unigram probabilities. The vocabulary is determined as a trade-off between its size and the out of vocabulary (OOV) rate measured on a development set. Words not appearing in the vocabulary are mapped to a common token, the *unknown* word (<UNK>).

Once the vocabulary is selected, component 2-, 3- and 4-gram LMs are trained separately on each of the training subsets. These models are estimated using a modified version of the Kneser-Ney smoothing with three discounting parameters per $n$-gram level (Chen and Goodman, 1999). The EM algorithm is used to estimate the interpolation coefficients in order to maximized the likelihood of a development set. The component models are linearly interpolated using the estimated coefficients to produce the final model.

### 3.1.3. Decoding

Prior to word recognition, an unsupervised audio partitioning procedure is performed to divide the audio stream into homogeneous clusters regrouping segments from the same speaker generally in similar acoustic conditions. After removing non-speech data from the audio file, segment boundaries and speaker labels are jointly estimated using a maximum likelihood iterative procedure (Gauvain et al., 1998). The algorithm uses a GMM to represent each speech segment and agglomerative clustering (see the original paper for further details).

Audio partitioning enables the use of cluster-based AM adaptation techniques. In the LIMSI system, unsupervised constrained maximum likelihood linear regression (CMLLR) and MLLR adaptation (Leggetter and Woodland, 1995) is performed on speaker cluster basis. The automatically extracted speaker labels can also be used to select a gender dependent model (if applicable).

Word recognition is performed in various steps. Using the CMLLR/MLLR adapted acoustic model and a 2-gram LM, a word lattice hypothesis is generated. This lattice is re-scored by a 3-gram and, then, by a 4-gram LM. Re-scoring uses both, lattice expansion and beam pruning. The final recognition hypothesis is obtained after a consensus decoding (Mangu et al., 1999).

## 3.2. The European Portuguese system

Portuguese is one of the most spoken languages in the world, having more than 210 million native speakers. Being spread around different geographical locations (Brazil, Portugal, Africa, East Timor and Macao), it has different dialect and accent varieties. Since the discussion of merits to distinguish these terms is not in the scope of this work, they might be used interchangeably here.

The European Portuguese (EP) and the Brazilian Portuguese (BP) can be considered the most important varieties of the language (Mateus and d'Andrade, 2000). The main difference between them is at the phonological level, namely the pronunciation of unstressed vowels (Mateus and d'Andrade, 2000). However, these two varieties also present differences in vocabulary, word spelling or written style, leading researchers to usually develop dialect-specific ASR systems (Silva et al., 2005; Abad et al., 2009). In fact, there is an international movement to uniform the orthographic system with the adoption of the International Orthographic Agreement by the Community of Portuguese Language Countries (CPLP, 1990) – *Comunidade dos Países de Língua Portuguesa*. Although the agreement was first signed in October 1990, it was ratified only in 2009 and the transition is not yet complete. In Brazil, for instance, the transition is planned to end in 2016 (Brasil, 2012).

Portuguese ASR was first investigated at LIMSI since early 2000's in the context of the European founded Alert Project[1] and has gained more attention in the recent years as one of the target languages of the Quaero Programme[2]. Given the context, both efforts focused on the recognition of the European Portuguese.

The following subsections provide more information about the EP ASR systems developed as part of this thesis work. After describing the development and evaluation corpora, the pronunciation modeling strategy is presented. Given that corpus subsets became available at different stages of this work, two different setups, the *early* and *improved* system setups, are presented.

[1] http://www.alert-project.eu
[2] http://www.quaero.org

| Set name | | Source | Epoch | # of Shows | Duration in hours |
|---|---|---|---|---|---|
| trainR00 | | RTP | 2000 | 9 | 3.1 (3.1)‡ |
| trainRVE | trainR01 | RTP | 2001 | 60 | 55.6 |
| | trainR10 | RTP | 2010 | 173 | 78.3 |
| | trainV09 | Voice of America | 2009 | 350 | 21.2 |
| | trainE10 | Euronews | 2009-2010 | 564 | 18.2 |
| trainQ10 | | Quaero * | 2010 | 184 | 71.7 (55.8)‡ |
| trainQ11 | | Quaero * | 2011 | 304 | 70.8 |
| **Total** | | | | **1644** | **318.9 (58.9)‡** |

‡ The duration in parenthesis refers to the available amount of manually transcribed data. Manual transcriptions for trainQ11 are available, but they were not used in this work.

* The source 'Quaero' refers to the corpora organized under the Quaero Programme. It comprises data from different sources (RTP, TSF, RDP, Antena1 and Antena2).

Table 3.1.: Description of the Portuguese acoustic training data. Duration is measured after music and noise removal.

### 3.2.1. Corpora

#### Acoustic training data

The acoustic data were collected from European Portuguese sources, although speakers of other dialects might be present. The data were divided into four sub-corpora according to their availability date as shown in Table 3.1. Training sets trainR00 and trainRVE were available from the beginning of this work. trainR00, containing about 3 hours of speech with manual transcriptions, was used only for bootstrapping. trainRVE contains about 173 hours of untranscribed speech data. It was subdivided into four subsets according to the epoch the shows were broadcast and the sources they were collected from. The two remaining sets, trainQ10 and trainQ11, were collected and manually transcribed as part of the Quaero Programme. They are comprised of, respectively, about 72 and 71 hours of audio, of which about 56 and 59 have been transcribed. These two data sets were available, respectively, in mid-2011 and mid-2012. In total, about 319 hours of audio data (118 hours with manual transcriptions) were available.

This acoustic training corpus contains both broadcast news (BN) and broadcast conversations (BC) data. The sets trainV09 and trainE10 are mostly BN. They were collected from the archives of the Portuguese branches of 'Voice of America' and 'Euronews' websites. Their data consist of short documents (a few minutes) and have few speakers per file. The remaining sets were collected via satellite or podcasts and were selected to contain more interactive data. They are somewhat longer (a few dozens of minutes) and have a wider variety of speakers.

The sets with R, trainR00, trainR01 and trainR10, were collected from the RTP channel. The 'Quaero' data (Q), trainQ10 and trainQ11, come from five sources (RTP, RDP, Antena1, Antena2 and TSF channels).

| Set name | Source | Epoch | # of Shows | Duration in minutes |
|---|---|---|---|---|
| testR00 | RTP | 2000 | 2 | 76 |
| testR09 | RTP | 2010 | 2 | 71 |
| testV09 | Voice of America | 2009 | 30 | 125 |
| testE10 | Euronews | 2010 | 100 | 113 |
| devQ10 | Quaero * | 2011 | 16 | 199 |
| testQ10 | Quaero * | 2011 | 17 | 207 |
| testQ11 | Quaero * | 2011 | 21 | 214 |
| **Total** | | | **188** | **1005** |

* The source 'Quaero' refers to the corpora organized under the Quaero Programme. It comprises data from different sources (RTP, TSF, RDP, Antena1 and Antena2).

Table 3.2.: Description of the Portuguese acoustic development and evaluation data. Duration is measured after music and noise removal.

### Development and evaluation data

The European Portuguese test data are shown in Table 3.2. They were collected from the same sources as used for acoustic modeling and were divided and named in order to simplify their relation with the training sets (Table 3.1). Thus, `testR00` is associated to `trainR00`, `testR09` to `trainR10` and so on. Both `devQ10` and `testQ10` relate to `trainQ10`. The data used for system development will be specified for each experiment. No overlap exists between any of the training, development or evaluation sets.

### Text corpus

The majority (about 89%) of the written corpus used in this work corresponds to the EP dialect. The remaining data (about 11%) come from Brazilian sources and have been used only on the early setup.

The text corpus is composed of data collected from different sources, such as newspapers, blogs and closed captions, as well as the available audio transcriptions. Information about this corpus is given in Table 3.3. The epoch refers to the date the texts were published according to the meta-data available from the source. The number of words refers to the number of space delimited character strings and was obtained after text normalization.

The vast majority (more than 99%) of the text data come from written sources (predominantly news). The 'Voice of America' and 'Euronews' texts are a mix of news and closed captions. The text data from these two sources were only used for language modeling in the early system and represent less than 0.8% of the total texts. As can be seen in Table 3.3, less than 0.1% correspond to manual transcriptions of audio data. 'RTP' corresponds to the transcriptions of `trainR00`, while 'Quaero' corresponds to `trainQ10`. The two sets comprise about 578 thousand words.

| Dialect | Type | Source | Epoch | # Words |
|---|---|---|---|---|
| European | News | Google News | 2007-2010 | 256.2M |
| | | Cetem Público | 1991-1998 | 172.8M |
| | | Diário Digital | 2001-2009 | 95.1M |
| | | Jornal de Notícias | 1996-2001 | 47.2M |
| | | Diário de Notícias | 2005-2007 | 23.2M |
| | | Portugal Diário | 2007 | 2.2M |
| | | Expresso | 2000-2001 | 2.0M |
| | | Correio | 2001 | 1.1M |
| | | *Total* | | *599.8M* |
| | Blog | Blog Sapo | 2006-2009 | *38.7M* |
| | News / Closed Captions | Euronews | 2004-2010 | 5.2M |
| | | Voice of America | 2009 | 350k |
| | | *Total* | | *5.6M* |
| | Transcriptions | RTP (`trainR00`) | 2000 | 29k |
| | | Quaero (`trainQ10`) | 2010 | 539k |
| | | *Total* | | *568k* |
| | **Total** | | | **644.7M** |
| Brazilian | News | O Povo | 2008-2009 | 33.9M |
| | | AFP | 1994-1998 | 22.3M |
| | | AEN | 2003-2007 | 11.5M |
| | | Folha de São Paulo | 2007 | 1.9M |
| | | NHK | 2004-2007 | 1.4M |
| | | *Total* | | *71.0M* |
| | Blog | Estadão | 2006-2009 | *26.5M* |
| | **Total** | | | **97.5M** |
| **Total** | | | | **742.0M** |

Table 3.3.: Description of the Portuguese textual training data. Last column gives the number of running words, obtained after text normalization. 'M' = $\times 10^6$; 'k' = $\times 10^3$. The Brazilian corpus was used only in early stages of development.

### 3.2.2. Pronunciation modeling

The pronunciation dictionary was built using an rule-based grapheme-to-phoneme (G2P) solution developed by me. For other published work on G2P conversion for European Portuguese, the reader may refer, for instance, to Braga et al. (2006) for a rule-based approach or to Caseiro et al. (2002) for a hybrid rule-based/data-driven approach. In both cases, the authors have used G2P converter for text-to-speech. While text-to-speech usually relies on the use of a single (canonical) pronunciation, ASR frequently needs to cover various pronunciation forms. Owning to the fact that pronunciation variability is not a main topic of this thesis, the G2P converter described here is able to generate a unique pronunciation form. The generation of pronunciation variants is a topic that was well covered in another thesis work developed in our group (Karanasou, 2013).

| | |
|---|---|
| consonants | b, p, t, d, g, k, f, v, s, z, ʃ, ʒ, m, n, ɲ, l, ʎ, ɾ, ʁ |
| oral vowels | i, e, ɛ, a, ɐ, ɨ, ɔ, o, u |
| nasal vowels | ĩ, ẽ, ɐ̃, õ, ũ |
| oral semi-vowels | j, w |
| special units | [breath], [hesitation], [silence] |

Table 3.4.: Phone list used for the European Portuguese ASR system development.

### Grapheme-to-phoneme conversion

The standard Lisbon phonological system was modeled using 35 phones, being 19 consonants, 9 oral vowels, 5 nasal vowels and 2 oral semi-vowels (Cruz-Ferreira, 1995). Two phones for two nasal semi-vowels were assessed in initial stages of development, however they were not kept since no improvement was observed.

In addition to the 35 phones, three special symbols were used to model breath noise, hesitation and silence. Table 3.4 shows the list of acoustic units used for the development of the Portuguese ASR system.

The Portuguese spelling system is somewhat close to the phonological representation, although many exceptions and cases of ambiguity exist. Some examples for which a G2P rule can be straightforwardly inferred are presented in Table 3.5. In the listed examples, the phonological forms are shown within slashes (//). An 'apostrophe' (ˈ) is placed before the stressed syllable, and a 'hyphen' (-) is used to indicate syllable boundaries whenever necessary. The standard Lisbon pronunciation variety is used in these examples (see the online dictionary of the *Instituto de Linguística Teórica e Computacional* (2007)[3]).

The graphemes (b, ç, d, f, j, k, l, p, t, v) have a unique phonological representation[4]. Diacritics are very useful to determine the pronunciation of vowels, although not always unambiguously. In some cases, it is a sequence of two letters that determines the phonemic conversion. For instance, 'ch', 'lh' and 'nh' always generate the phonemes /ʃ/, /ʎ/ and /ɲ/ respectively.

In the cases presented Table 3.6, a deeper analysis of context need to be performed for G2P conversion. This is the case for 's', for which the pronunciation depends on its position in the word (e.g. ***s**aber* /s/ vs. *rua**s*** /ʃ/) or on the left and right contexts (e.g. *ca**s**a* /z/ vs. *fe**s**ta* /ʃ/ vs. *sen**s**o* /s/). However, there are some exceptions to these rules (e.g. *trân**s**ito* /z/ vs. *ân**s**ia* /s/).

The pronunciation of vowels also depends on if they occur in the tonic position (e.g. *c**a**s**a*** /ˈk**a**z**ɐ**/, ***o**v**o*** /ˈ**o**v**u**/, *z**e**l**e*** /ˈz**ɛ**l**ɨ**/) or if they are realized as a glide (e.g. *pa**i*** /ˈpa**j**/ vs. *d**i**a* /dˈ**i**a/). Nevertheless, some cases remain ambiguous (e.g. ***o**vo* /o/ vs. ***o**vos* /ɔ/), while others require additional (non-graphemic) information to determine the pronunciation of a vowel (e.g. *s**e**lo* as a noun /e/ vs. *s**e**lo* as a verb /ɛ/).

Another example that cannot be fully disambiguated via a rule-based G2P system is the case of 'x'. Due to an inherent variation that depends on the origins of the words, 'x' can be realized as /ʃ/, /s/, /z/ or /ks/. For instance, the pronunciation of 'x' in words *li**x**o*, *proli**x**o* and *fi**x**o* (/ʃ/, /ks/, /ks/ respectively) can only be defined as exception rules

[3]Available at http://www.portaldalinguaportuguesa.org

[4]Of course this is limited to the scope of this work. The graphemes (t, d) when followed by (i, e) might be represented by [tʃ, dʒ] in some variants of Brazilian Portuguese, while (l) might be represented by /l/ or /ɬ/ in the standard Lisbon variant (see note 2 in Table 3.5).

| Grapheme | Phoneme | Graphemic form | Phonemic form | Comments |
|---|---|---|---|---|
| b | /b/ | **b**ela | /ˈ**b**ɛlɐ/ | Unique mapping |
| ç | /s/ | cria**ç**ão | /kriɐˈ**s**ɐ̃w/ | |
| d | /d/ | **d**ia | /ˈ**d**ia/ | |
| f | /f/ | **f**avor | /**f**ɐˈvor/ | |
| j | /ʒ/ | **j**ato | /ˈ**ʒ**atu/ | |
| á | /a/ | **á**pice | /ˈ**a**pisɨ/ | Vowels with diacritics having often a unique mapping |
| â | /ɐ/ | **â**nimo | /ˈ**ɐ**nimu/ | |
| é | /ɛ/ | **é** | /ˈ**ɛ**/ | |
| í | /i/ | v**í**cio | /ˈv**i**sju/ | |
| õ | /õ/ | miss**õ**es | /miˈs**õ**jʃ/ | |
| ú | /u/ | ba**ú** | /baˈ**u**/ | |
| ch | /ʃ/ | **ch**á | /ˈ**ʃ**a/ | Two letters with unique mapping |
| lh | /ʎ/ | vel**h**o | /ˈvɛ**ʎ**u/ | |
| nh | /ɲ/ | so**nh**o | /ˈso**ɲ**u/ | |

Table 3.5.: Examples of G2P conversion cases that can be performed in a straightforward manner using a rule-based G2P system. The sign (ˈ) is placed before the stressed syllable.

(if the origin of the word is unknown). Some rules can be defined for specific contexts. For instance, in words beginning with 'ex' followed by a vowel, 'x' becomes a /z/ (e.g. *e**x**ato*).

### Pronunciation lexicon

The pronunciation lexicon was created to represent the Portuguese variant spoken in Lisbon. The rule-based G2P converter developed in this work has four steps:

1. **Pre-syllabification.** A hundred rules that cover almost all syllabification cases. Exceptions are cases that require stress syllable marking to resolve diphthongs (e.g. *di-a* vs. *ín-dia*).

2. **Stress syllable marking.** 18 rules that are used to simplify some of the conversion rules (see above).

3. **Post-syllabification.** Eight rules used to finish word syllabification.

4. **Phonetization.** 264 rules for vowels and 137 for consonants used to convert grapheme into phonemes.

Except for acronyms, for which the uttered and spelled pronunciation forms have been used, a unique pronunciation is generated for each word. This limits the strength of the dictionary since only one form of homographs can be represented. In our case, the form minimizing the phone error rate with respect to a reference dictionary has been used. For instance, *selo* has been transcribed /ˈselu/ rather than /ˈsɛlu/ (Table 3.6).

The pronunciation dictionary could be enhanced by allowing multiple pronunciations for ambiguous cases. To reduce the confusion incurred by the additional pronunciations,

| Grapheme | Phonemes | Graphemic form | Phonemic form | Comments |
|---|---|---|---|---|
| a | /a/ or /ɐ/ | c**a**s**a** | /ˈk**a**z**ɐ**/ | Depends on the syllable stress and the context |
| | | c**a**fé | /k**ɐ**ˈfɛ/ | |
| | | c**a**so | /ˈk**a**su/ | |
| | | sop**a** | /ˈsop**ɐ**/ | |
| | | l**a**ma | /ˈl**ɐ**mɐ/ | |
| e | /e/, /ɛ/, /ɨ/ or /j/ | z**e**l**e** | /ˈz**ɛ**l**ɨ**/ | Depends on the syllable stress, the context and if it is realized as a diphthong or not. Might be ambiguous |
| | | **e**xato | /**i**ˈzatu/ | |
| | | s**e**lo (*noun*) | /ˈs**e**lo/ | |
| | | s**e**lo (*verb*) | /ˈs**ɛ**lo/ | |
| | | c**e**ar | /ˈs**j**ar/ | |
| s | /s/, /ʃ/ or /z/ | **s**aber | /**s**aˈber/ | Depends on word position, syllabification and the context |
| | | rua**s** | /ˈrua**ʃ**/ | |
| | | ca-**s**a | /ˈkaz**ɐ**/ | |
| | | fe**s**-ta | /ˈfɛ**ʃ**tɐ/ | |
| | | **s**en-**s**o | /ˈ**s**ẽ**s**u/ | |
| | | trân-**s**i-to | /ˈtrẽ**z**itu/ | |
| | | ân-**s**ia | /ˈẽ**s**ja/ | |
| x | /ʃ/, /ks/, /z/ or /s/ | li**x**o | /ˈli**ʃ**u/ | Often ambiguous |
| | | proli**x**o | /pruˈli**ks**u/ | |
| | | fi**x**o | /ˈfi**ks**u/ | |
| | | e**x**ato | /iˈ**z**atu/ | |
| | | sinta**x**e | /sĩˈta**s**ɨ/ | |

Table 3.6.: Examples of G2P conversion cases that require syllabification (or context analysis), stress syllable marking or are ambiguous. The sign (ˈ) is placed before the stressed syllable. The 'hyphen' (-) is used to separate syllables.

the acoustic training data (by means of forced alignment or decoding) could be used to discover the correct pronunciation form(s) and to associate their pronunciation probabilities.

### Evaluation of the pronunciation dictionary

The G2P converter described above was assessed as follows. A pronunciation lexicon containing about 9.5k words was downloaded from the online dictionary Infopedia (Infopedia, 2009) on January 2010[5]. Another lexicon was generated for the same list using the G2P converter.

Compared to the reference, the generated pronunciations have a WER of 20.3% and a phone error rate (PER) of 3.5%. About 76% of the wrongly generated pronunciations had a unique phone error, and about 19% had two errors. Many errors are due to ambiguous cases (e.g. *ad**e**ga* /ɛ/ → /e/, *r**o**ça* /ɔ/ → /o/, *au**x**iliar* /s/ → /ks/). Other notable errors come from design choices made as the case of the diphthongs /wẽ/ and /we/ which were respectively represented as /uẽ/ and /ue/ (e.g. *fl**uê**ncia* /uẽ/, *c**oe**so* /ue/). Furthermore, about a third of words having three or more phone errors are words borrowed from foreign

[5]The pronunciations seem to be no longer available in this dictionary.

languages (e.g. *ballet*, *cachet*, *designer*, *jeans*, *online*, *marketing*), for which the G2P rules are not well suited. For many systems, foreign words are typically treated by exception rules.

The G2P converter was also assessed on a decoding task and compared to a rule-based converter previously available at LIMSI. Both systems were used to generate a pronunciation dictionary for a 145k word vocabulary list. Acoustic models were constructed for each generated dictionary using a subset of 11 hours of acoustic data from `trainRVE` (Table 3.1). The models were evaluated on a 4.5 hour development set comprised of `testR00`, `testR09` and `testV09` (Table 3.2). The new G2P converter obtained a 6% WER relative improvement (30.4% vs. 32.5%) over the former converter.

### 3.2.3. Early system

As mentioned before, the training corpora have became available at different stages of this work. For the early system setup, about 176 hours of acoustic data were used. It consists of the 3h of manually transcribed data from `trainR00` and the 173h of unstranscribed data from `trainRVE`.

A 39-dimensional vector containing PLP-like features was used for acoustic modeling. The acoustic models are gender independent and were estimated via MLE. Two sets of acoustic models were trained. The first bootstrap models were estimated on `trainR00` and cover about 3.6k phone contexts with about 1.5k tied states. These bootstrap models were used to initialize unsupervised AM training (Zavaliagkos and Colthurst, 1998) using the 173 hours of untranscribed data in `trainRVE`. The resulting models cover about 15.7k phone contexts and have about 11.5k tied states.

The text data used for language modeling contains about 485 million words. Sources of both European and Brazilian Portuguese data were used with the exception of the 'Quaero' transcriptions and 'Google News'. The texts were normalized by converting abbreviations and numerical patterns to their approximate spoken forms. Acronyms were not explicitly treated, since the pronunciation dictionary already contains uttered and spelled pronunciations. Furthermore, punctuation marks were removed, although words containing a hyphen (e.g. *sexta-feira*) were not split.

A 158k word vocabulary was obtained by merging four different word lists, each selected in order to keep the OOV rate below 1% on their corresponding development sets, `testR00`, `testR09`, `testV09` and `testE10`. This word list was used to estimate 2-, 3- and 4-gram language models on the training texts. The perplexity of the combined development set with the 4-gram LM is 149.

The pronunciation lexicon was generated by an early version of the G2P converter. This early G2P had PER of 4.0%, compared to the 3.5% with the revised G2P converter with respect to the reference dictionary.

This early system was used only for the development of the lattice-based unsupervised acoustic model training method (Chapter 4) and evaluated on `testR00`, `testR09`, `testV09` and `testE10`. Two systems were built for reference: 'System A', which uses the bootstrap AM trained on `trainR00`; 'System B', which uses the acoustic model estimated on `trainRVE` via unsupervised training. Both make use of the same 158k word language model.

The baseline results obtained with the early setup are summarized in Table 3.7. The

| System name | LM | AM | WER(%) | | | | |
|---|---|---|---|---|---|---|---|
| | | | testR00 | testR09 | testV09 | testE10 | Total |
| System A | 485M | trainR00 | 36.1 | 36.2 | 21.4 | 22.5 | 27.0 |
| System B | 485M | trainRVE | 26.4 | 27.3 | 12.9 | 13.7 | 18.1 |
| Relative improvement (%) | | | 26.8 | 24.6 | 39.7 | 39.1 | 33.0 |

Table 3.7.: Baseline results with the early Portuguese ASR system. The language model (LM) was trained on the subset of the corpus available by mid-2011. 'System A' uses the bootstrap acoustic model (AM), trained on 3h of manually transcribed data. 'System B' uses an unsupervised AM trained on 173h of data.

bootstrap system obtained an overall WER of 27.0% and 'System B' an overall WER of 18.1%. The use of 173h of untranscribed data for acoustic modeling reduced the relative WER by one-third.

### 3.2.4. Improved system

Prior to the 2011 Quaero evaluation campaign, new acoustic and textual data became available and were incorporated into the *improved* system.

About 72 hours of data, of which 56 hours have transcriptions associated (`trainQ10`), were added to the acoustic training corpus. A development set (`devQ10`) and an evaluation set (`testQ10`) containing respectively 3.3 and 3.5 hours of data also became available.

A 39-dimensional vector containing PLP-like features was used for acoustic modeling. Acoustic models are gender independent and were estimated via MLE. The use of gender dependent models were also assessed, but did not improve the system performance.

For this system, two additional text normalization steps were applied. First, a 3-gram LM was used to correct the word case of all data, except the manual transcriptions, which are assumed to have the correct case assigned. The word casing LM was trained on pre-selected sources. Second, the new orthographic agreement (CPLP, 1990) was applied. About 800 words were converted into the new orthographic forms using rules automatically extracted from the Infopedia online dictionary (Infopedia, 2009).

With the arrival of the 'Google News' data, the Brazilian text sources were removed after assessing that their usage did not reduce perplexity of the development data. (This agrees with the results reported by Abad et al. (2009)). A decision was taken to consider a development scenario where only news and blog sources were available. Thus, texts from 'Euronews' and 'Voice of America' were also not used even though these were closed captions which are certainly a closer match to spoken language than the news texts.

Two language models were generated. The `LM_10src` was built using the eight EP news text sources, the 'Blogsapo' data and the 'RTP' transcriptions, totaling about 639M words. The 'Quaero' transcriptions were included when training the second model, `LM_11src`. The perplexities of the development set `devQ10` are 138 and 127 respectively with `LM_10src` and `LM_11src`. A relative improvement of 8% is seen by including a relatively small transcription set (539k of 639M words) in the LM training.

The LMs have a 65k word vocabulary [6], which was obtained based on the interpolation of unigram component models trained on the same data as were used to build the `LM_10src`. An OOV rate of 1.1% was measured on `devQ10`. The G2P converter was used to generate the pronunciations for this word list.

Five reference systems were setup to represent different acoustic modeling scenarios. WERs with these five systems are shown in Table 3.8 on `devQ10`, `testQ10` and `testQ11`. 'System C' is the bootstrap system that uses the same AM as 'System A' (Table 3.7), trained only on `trainR00`. The pronunciation lexicon and LMs differ between 'System A' and 'System C'.

The second part of the table shows the performance of systems that use only `trainQ10` for acoustic modeling. 'System D' represents the condition for which the transcriptions of `trainQ10` are not considered available: it uses an AM trained in an unsupervised manner together with `LM_10src`. 'System E' is the complementary supervised system: the manual transcriptions of `trainQ10` were included for acoustic and language modeling (`LM_11src`). The acoustic models of 'System D' and 'System E' cover, respectively, 12.4k and 11.5k phone contexts, and have about 10.1k and 11.5k tied states. The supervised ('System E') outperforms the unsupervised system ('System D') when models are trained on the same data by about 12% relative. The use of the `trainQ10` data brings a 40% relative WER improvement over the bootstrap system for the unsupervised case and a 47% reduction for the supervised training condition.

The third part of Table 3.8 shows the performance of systems that also use the `trainRVE` data for acoustic modeling. The acoustic models of such systems were generated as follows. First, a model was trained on the `trainRVE` using an unsupervised approach. This model was then adapted unsupervised ('System F') or supervised ('System G') using the `trainQ10` data with MAP estimation. Both models cover about 15.7k phone contexts and have about 11.5k tied states. In the supervised case, `trainQ10` transcripts are also used for language modeling, that is, 'System F' uses `LM_10src` and 'System G' uses `LM_11src`. The supervised ('System F') outperforms the unsupervised system ('System G') by about 10% relative. 'System G' and 'System F' outperform the bootstrap system by 43% and 48% relative respectively.

The training data `trainQ11`, which was available only in mid-2012, was not used for the baseline system development. This set was only used to assess unsupervised MLP training approaches (Chapter 5).

## 3.3. The English system

An English ASR system was also used in this work as a basis for the experiments on accent adaptation via model interpolation that are presented in Chapter 8. The data and baseline system are the same as presented in Vergyri et al. (2010) and are briefly described in this section. The system was designed to recognize multi-accented broadcast data coming from six different geographical regions where English is spoken as an official language. For further details, the reader is referred to the original paper.

[6] The vocabulary of the *improved* system (65k) is smaller than the vocabulary of the *early* system (158k) due to the additional text normalization steps applied and the removal of the Brazilian Portuguese text data from the language modeling corpus.

| System name | LM | AM | WER(%) | | | |
|---|---|---|---|---|---|---|
| | | | devQ10 | testQ10 | testQ11 | Total |
| System C | LM_10src | trainR00 | 53.7 | 45.9 | 54.8 | 51.5 |
| System D | LM_10src | trainQ10 | 33.0 | 26.8 | 33.0 | 30.9 |
| System E | LM_11src | trainQ10 | 28.7 | 23.4 | 29.1 | 27.1 |
| System F | LM_10src | trainRVE → trainQ10 | 31.2 | 25.6 | 31.6 | 29.5 |
| System G | LM_11src | trainRVE → trainQ10 | 28.0 | 23.2 | 28.6 | 26.6 |

Table 3.8.: Baseline results with the improved Portuguese ASR system. Transcriptions of trainQ10 are present in LM_11src, but not in LM_10src. Systems D and F use unsupervised acoustic model (AM) training, while systems E and G use supervised AM training. In the two last rows, an unsupervised model trained on trainRVE has been MAP adapted to trainQ10.

| Set (Info) | US | AU | GB | ME | NA | IN | Non-US | Total |
|---|---|---|---|---|---|---|---|---|
| Training (hours) | 316.6 | 33.0 | 55.4 | 27.7 | 8.2 | 9.4 | 133.7 | 450.3 |
| Training (# of shows) | 667 | 461 | 225 | 72 | 34 | 26 | 818 | 1485 |
| Test (min) | 172 | 12 | 48 | 15 | 13 | 15 | 103 | 275 |
| Test (# of shows) | 10 | 4 | 3 | 1 | 1 | 1 | 10 | 20 |
| Test (# of speakers) | 202 | 19 | 45 | 15 | 15 | 15 | 109 | 311 |

Table 3.9.: Duration of the multi-accented English data sets. Accents: United States (US), Australia (AU), Great Britain (GB), Middle East (ME), North Africa (NA), India (IN). Last two columns give the total without and with 'US' data.

### 3.3.1. Corpora

Information about the training and test corpora used to build and evaluate the multi-accented English ASR system are presented in Table 3.9. It consists of a multi-accented corpus collected from six different geographical regions where English is spoken as official language: United States (US), Australia (AU), Great Britain (GB), Middle East (ME), North Africa (NA) and India (IN). The audio data comes from a variety of news sources, mostly collected via satellite with some downloaded from the Web. The geographical region from where the show was broadcast is considered as the accent label for all the speakers in the audio file. The true accent label for each speaker is unknown. In Vergyri et al. (2010), the labels for the 'ME' and 'NA' accents were mistakenly swapped.

Roughly, two thirds of the data come from the 'US' dialect, with the other one third unevenly distributed across the remaining five dialects. The 'NA' and 'IN' dialects are the less represented in the corpus each having less than 2% of the total amount of data. The test subset sizes have been defined based on the distribution of the training data.

### 3.3.2. System overview

The broadcast ASR system used in this work is quite similar to other systems developed at LIMSI (Gauvain et al., 2002), as introduced in Section 3.1. Acoustic modeling uses a 42-dimensional feature vector, including cepstrum, log energy and pitch features. The phone set contains 35 phones besides the special units for silence, breath and hesitation.

| System | US | AU | GB | ME | NA | IN | All | Ave |
|---|---|---|---|---|---|---|---|---|
| Accent independent | 14.34 | 11.92 | 12.84 | 15.90 | 26.47 | 39.28 | 16.07 | 20.12 |
| Speaker-accent-ID | 14.71 | 11.23 | 13.10 | 16.46 | 25.19 | 34.28 | 16.01 | 19.18 |
| Show-accent-ID | 13.95 | 11.91 | 11.98 | 16.46 | 25.19 | 34.28 | 15.39 | 18.96 |

Table 3.10.: Baseline results with the multi-accented English automatic speech recognition system after Vergyri et al. (2010). 'All' corresponds to the WER on the whole test set, while 'Ave' to the average WER when the test subsets are weighted equally. In the original paper, the 'ME' and 'NA' accent labels have been mistakenly swapped.

The baseline acoustic models of Vergyri et al. (2010) were generated as follows. First, an accent- and gender-independent model was estimated on the entire training data set. Gender-specific and accent-specific models were obtained using a joint MAP adaptation (Gauvain and Lee, 1994), followed by one iteration of MMIE (Bahl et al., 1986). Speaker adaptive training (Anastasakos et al., 1996) was also used. The models cover about 18k phone contexts with about 11.5k tied states.

The language models were trained on about 1.2 billion words of texts and were built based on the interpolation of LM components estimated on different subsets of the training data. The system uses a 65k word vocabulary.

The decoder was slightly modified to recognize multi-accented data. Similar to Chen et al. (2001), a GMM-based classifier first determines the most likely accent for each test segment. Based on this decision and the speaker labels automatically extracted during audio partitioning, an accent- and gender-specific model is selected for decoding. Standard decoding is performed, including CMLLR/MLLR adaptation, lattice generation, lattice re-scoring and consensus decoding. Accent classification was compared at the show and speaker level. A 100% accuracy is reported by Vergyri et al. (2010) for a show based accent classification.

The results obtained by Vergyri et al. (2010) are summarized in Table 3.10. The first row shows the accent-independent system. The second and third rows show the multi-accented systems with, respectively, speaker- and show-based accent classification. With show-based classification, a 4% relative WER improvement in average is obtained compared to the accent-independent system (from 16.07% to 15.39%). However, the performance for the 'ME' accent deteriorates.

## 3.4. Summary

This chapter has presented an overview of the two ASR systems used throughout this work in order to provide relevant background for the remaining chapters. First, the LIMSI BN recognition system was generally presented, followed by details of the European Portuguese (EP) and the multi-accented English systems.

The EP recognition system (or, more precisely, the *improved* system setup) was first developed for the 2011 Quaero Evaluation Campaign, in which I have actively participated. I developed and evaluated the G2P converter, updated and refined the text normalization procedure (with the application of the new orthographic agreement), and performed part of the unsupervised acoustic model training.

The majority of the work carried out in this thesis was tested with the improved EP system, although not always using the full acoustic model training corpus in order to simulate some training situations. For example, only the 'Quaero 2010' acoustic data is used in some cases to compare the supervised and unsupervised training approaches. The recognition results with the reference systems have been reported in this chapter.

The multi-accented English data and the baseline system presented in Vergyri et al. (2010) were briefly described. This system is used as the baseline for the experiments on multi-accented data recognition presented in Chapter 8.

# Part II.

# Unsupervised model training

# Unsupervised model training

A great portion of time and cost consumed during automatic speech recognition system development is due to the production of manual audio transcripts. While, for some tasks, large amounts of data can be easily gathered from different sources (Web, radio, TV, etc.), manually transcribing them is an arduous task. Depending on the type of data, a human might spend up to 50 times real time to produce detailed audio transcriptions, and even then, they will not be exempt of errors (Barras et al., 2001).

Large amounts of audio transcriptions are required for acoustic model training. In practice, dozens to hundreds (or even thousands) of hours of speech are used to build AMs for large vocabulary speech recognition. A technique that has been gaining popularity in the speech community as a means to reduce the human effort spent on data annotation is the *unsupervised* AM training (Zavaliagkos and Colthurst, 1998; Kemp and Waibel, 1999). The term *unsupervised* training is used here to define the method in which the model parameters are estimated from automatic (rather than manual) transcriptions. Some authors prefer to refer to the method as *semi-supervised* or *self-supervised* training, since the language model provides some level of supervision. Beyond the speech recognition community, the term unsupervised is used with a different meaning.

Unsupervised acoustic model training consists of an iterative procedure in which the parameter estimation relies on untranscribed audio data. In short, it makes use of an existing ASR system to decode a training set comprised of a large amount of untranscribed audio data. The generated inaccurate transcriptions are used as ground truth to estimate the acoustic model parameters.

Automatic transcriptions are known to contain errors that can mislead the AM parameter estimation. This issue is usually dealt with the use of confidence measures provided by the ASR system to filter or weigh the best decoding hypothesis (Wessel and Ney, 2001; Gollan et al., 2007). A novel approach is proposed in **Chapter 4**. It consists of using multiple decoding hypotheses (rather than the best one) to guide the acoustic parameter estimation. The multiple hypotheses are extracted from the decoding lattices to better represent the true transcription labels and lead to more accurate models. The use of different training strategies is also investigated as an attempt to avoid the propagation of errors from one iteration to another.

Most of the published experiments in unsupervised acoustic model training make use of features extracted directly from the audio stream, such as the PLP features (Hermansky, 1990). In recent years, different research groups have adopted the use of acoustic features extracted from MLP classifiers trained in a supervised manner (Ellis et al., 2001;

Zhu et al., 2005; Stolcke et al., 2006; Tóth et al., 2008; Fousek et al., 2008a; Lamel et al., 2011; Veselý et al., 2012). MLP based features can lead to substantial gains in performance when combined with PLP features.

**Chapter 5** proposes to train the MLP classifiers on untranscribed audio data rather than manually transcribed data. This approach allows the construction of accurate AMs making use of MLP features in a fully unsupervised manner. Both MLPs and acoustic model training rely on the use of automatic transcriptions. This approach is empirically compared to another solution that relies on the use of MLPs trained on data from other languages (Stolcke et al., 2006; Tóth et al., 2008).

In contrast to AM training, audio transcriptions are not mandatory for LM training. Language models can be estimated on huge amounts of text data that can be easily collected from the Web. However, the inclusion of audio transcripts in the LM training corpus is expected to improve the model accuracy, since transcriptions truly represent the spoken style of the target data. An alternative to improve LM performances without relying on transcriptions is to collect conversational-like data from the Web by means of information retrieval techniques (Bulyko et al., 2007). Another alternative is to use audio transcriptions automatically produced by an existing ASR system (Bacchiani and Roark, 2003), or in other words, perform unsupervised language model training. Compared to unsupervised acoustic modeling, unsupervised language modeling is a more challenging task (Novotney et al., 2009). Although little explored, unsupervised LM training has been reported to improve the performance for conversational data recognition tasks (Bacchiani and Roark, 2003; Novotney et al., 2009).

Although not mandatory, it is not difficult to prove that audio transcriptions also play a major role in language modeling since they truly represent the spoken form. An alternative that naturally arises is the use of automatic transcriptions as ground truth for language model parameter estimation (Bacchiani and Roark, 2003). **Chapter 6** investigates the use of unsupervised LM training approaches for a broadcast data recognition task. Like done for unsupervised AM training, confidence based filtering and weighting approaches are proposed to deal with the recognition errors. The use of multiple decoding hypotheses (rather than the best one) is also proposed and assessed empirically. The unsupervised training framework is extended to neural network language models. It is proposed to retrain a baseline NNLM on automatically generated transcription text data.

Some of the work described in this part of this dissertation was originally published in ICASSP 2011 (Fraga-Silva et al., 2011), SLTU 2012 (Fraga-Silva et al., 2012) and Interspeech 2013 (Roy et al., 2013).

# Chapter 4

# Unsupervised HMM/GMM-based acoustic model training

## 4.1. Introduction

Acoustic model training for large vocabulary continuous speech recognition relies on large amounts of transcribed audio data to achieve suitable performance levels. However, obtaining manually annotated audio data is an expensive and time-consuming task. A well-known technique that has been gaining popularity over the last years and can be used as a means to reduce the human effort required to prepare data for acoustic learning is commonly referred to as *unsupervised* acoustic model training (Zavaliagkos and Colthurst, 1998; Kemp and Waibel, 1999; Wessel and Ney, 2001; Lamel et al., 2002b; Wang et al., 2007; Lamel and Vieru, 2010). In such an approach, acoustic model training is based upon inaccurate transcriptions obtained via automatic speech recognition.

Unsupervised training is investigated in this chapter. In the following subsections, we propose to fit all the unsupervised training methods studied here into the iterative expectation-maximization framework (Dempster et al., 1977). In Section 4.2, we investigate confidence-based filtering and weighting approaches used to reduce the impact of the recognition errors present in the automatic transcriptions.

A novel unsupervised AM training approach is proposed in Section 4.3. Multiple decoding hypotheses (rather than the best one) are extracted from the decoding lattices to better approximate the true audio transcriptions. The use of lattices as a material for improving existing algorithms is recurrent in the speech processing community. For instance, they have been used for language recognition (Gauvain et al., 2004), speaker adaptation (Padmanabhan et al., 2000) and system combination (Li et al., 2002).

Some experiments are described along this chapter to validate the theoretical analyses. A further experimental work is conducted in Section 4.4 to compare the aforementioned approaches. In parallel, we investigate the use of different training strategies to assess their impact on the propagation of recognition errors across the training iterations. A summary is presented in the Section 4.5.

The mathematical framework presented takes into account only one possible pronunciation per word. The extension to the multiple pronunciations case is straightforward.

### 4.1.1. Maximum likelihood estimation

Acoustic model parameters can be estimated via maximum likelihood estimation, which can be represented by the following optimization problem:

$$\boldsymbol{\lambda}_{ML} = \arg\max_{\boldsymbol{\lambda}} \log f\left(\mathcal{X}|\boldsymbol{\lambda}\right) \tag{4.1}$$

This is an incomplete data problem that can be efficiently solved using the EM algorithm (Dempster et al., 1977) via the maximization of the following auxiliary function (see Section 2.2.2):

$$\begin{aligned} Q(\boldsymbol{\lambda}, \hat{\boldsymbol{\lambda}}) &= E\left[\log f(\mathcal{Y}|\boldsymbol{\lambda})\middle|\mathcal{X}, \mathcal{W}, \hat{\boldsymbol{\lambda}}\right] \\ &= E\left[\log f(\mathcal{S}|\mathcal{W}, \boldsymbol{\lambda})\middle|\mathcal{X}, \mathcal{W}, \hat{\boldsymbol{\lambda}}\right] + E\left[\log f(\mathcal{X}, \mathcal{L}|\mathcal{S}, \boldsymbol{\lambda})\middle|\mathcal{X}, \mathcal{W}, \hat{\boldsymbol{\lambda}}\right] \end{aligned} \tag{4.2}$$

where $\mathcal{Y} = (\mathcal{X}, \mathcal{W}, \mathcal{S}, \mathcal{L})$ represents the complete data, $\mathcal{X}$ the observed vectors, $\mathcal{W}$ the known word sequence, $\mathcal{S}$ the unknown HMM state sequence and $\mathcal{L}$ the unknown sequence of Gaussian labels. In (4.2), it is assumed the word sequence probability does not depend on the acoustic model parameters, i.e. $P(\mathcal{W}|\boldsymbol{\lambda}) = P(\mathcal{W})$.

In contrast to the equations presented in Section 2.2.2 (c.f. (2.14)), the word sequence labels $\mathcal{W}$ are explicitly shown in (4.2). The iterative EM algorithm used to solve the MLE problem (4.1) can be summarized as follows:

**Forced alignment** Given the current model $\hat{\boldsymbol{\lambda}}$, $\mathcal{W}$ and $\mathcal{X}$, calculate state/frame alignment probabilities $P(\mathcal{S}|\mathcal{X}, \mathcal{W}, \hat{\boldsymbol{\lambda}})$ using the forward-backward algorithm (Bahl et al., 1983) or, alternatively, estimate the best state sequence $\mathcal{S}^*$ using the Viterbi algorithm (Viterbi, 1967).

**Model update** Given $P(\mathcal{S}|\mathcal{X}, \mathcal{W}, \hat{\boldsymbol{\lambda}})$ estimate the new model parameters $\boldsymbol{\lambda}$.

$$\boldsymbol{\lambda}^* = \arg\max_{\boldsymbol{\lambda}} \sum_{\mathcal{S}} \sum_{\mathcal{L}} P(\mathcal{L}, \mathcal{S}|\mathcal{X}, \mathcal{W}, \hat{\boldsymbol{\lambda}}) \cdot \log f\left(\mathcal{X}, \mathcal{L}, \mathcal{S}|\mathcal{W}, \boldsymbol{\lambda}\right) \tag{4.3}$$

where the summation is performed over all possible state and Gaussian label sequences. Alternatively, if the Viterbi alignment is used:

$$\boldsymbol{\lambda}^* = \arg\max_{\boldsymbol{\lambda}} \sum_{\mathcal{L}} P(\mathcal{L}|\mathcal{X}, \mathcal{S}^*, \hat{\boldsymbol{\lambda}}) \cdot \log f\left(\mathcal{X}, \mathcal{L}, \mathcal{S}^*|\boldsymbol{\lambda}\right) \tag{4.4}$$

In both cases, since the true word sequence is known, we have used $P(\mathcal{W}|\mathcal{X}) = 1$.

### 4.1.2. Maximum likelihood estimation without audio transcriptions

Unsupervised AM training is evidently an incomplete data problem (Dempster et al., 1977). In addition to the HMM state and Gaussian level alignments, the true transcription

labels are also unknown. In this case, the auxiliary function to be maximized can be represented as:

$$\begin{aligned} Q(\boldsymbol{\lambda}, \hat{\boldsymbol{\lambda}}) &= E\left[\log f(\mathcal{Y}|\boldsymbol{\lambda}) \middle| \mathcal{X}, \hat{\boldsymbol{\lambda}}\right] \\ &= E\left[\log f(\mathcal{S}|\mathcal{W}, \boldsymbol{\lambda}) \middle| \mathcal{X}, \hat{\boldsymbol{\lambda}}\right] + E\left[\log f(\mathcal{X}, \mathcal{L}|\mathcal{S}, \boldsymbol{\lambda}) \middle| \mathcal{X}, \hat{\boldsymbol{\lambda}}\right] \end{aligned} \tag{4.5}$$

This function can be decomposed as:

$$Q(\boldsymbol{\lambda}, \hat{\boldsymbol{\lambda}}) = Q_{\boldsymbol{\pi}}(\boldsymbol{\pi}, \hat{\boldsymbol{\lambda}}) + Q_{\boldsymbol{A}}(\boldsymbol{A}, \hat{\boldsymbol{\lambda}}) + Q_{\theta}(\theta, \hat{\boldsymbol{\lambda}}) \tag{4.6}$$

where $\boldsymbol{\pi}$ denotes the state initial probabilities, $\boldsymbol{A}$ the state transition probabilities and $\boldsymbol{\theta}$ the state emission probabilities. These terms can be independently maximized and can be written as:

$$Q_{\boldsymbol{\pi}}(\boldsymbol{\pi}, \hat{\boldsymbol{\lambda}}) = \sum_{u \in \mathcal{V}} \sum_{i=1}^{N} \bar{\psi}_{ui0} \log \pi_i \tag{4.7}$$

$$Q_{\boldsymbol{A}}(\boldsymbol{A}, \hat{\boldsymbol{\lambda}}) = \sum_{u \in \mathcal{V}} \sum_{v \in \mathcal{V}} \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{t=1}^{T} \bar{\xi}_{uvijt} \log a_{ij} \tag{4.8}$$

$$Q_{\theta}(\theta, \hat{\boldsymbol{\lambda}}) = \sum_{u \in \mathcal{V}} \sum_{i=1}^{N} \sum_{k=1}^{K} \sum_{t=1}^{T} \bar{\gamma}_{uikt} \log \omega_{ik} \mathcal{N}\left(x_t | \mu_{ik}, \Sigma_{ik}\right) \tag{4.9}$$

where $u$ and $v$ are words in the vocabulary $\mathcal{V}$ and the probabilities $\bar{\psi}$, $\bar{\xi}$ and $\bar{\gamma}$ can be defined as:

$$\begin{aligned} &\bar{\psi}_{uit} = P\left(w_t = u, s_t = i \middle| \mathcal{X}, \hat{\boldsymbol{\lambda}}\right) \text{ the probability of being at word } u \text{ and state } i \text{ at time } t, \text{ given that model } \hat{\boldsymbol{\lambda}} \text{ generates } \mathcal{X}. \\ &\bar{\xi}_{uvijt} = P\left(w_{t-1} = u, w_t = v, s_{t-1} = i, s_t = j \middle| \mathcal{X}, \hat{\boldsymbol{\lambda}}\right) \text{ the probability of making a transition from word } u \text{ and state } i \text{ to word } v \text{ and state } j \text{ at time } t, \text{ given that } \hat{\boldsymbol{\lambda}} \text{ generates } \mathcal{X}. \\ &\bar{\gamma}_{uikt} = P\left(w_t = u, s_t = i, l_t = k \middle| \mathcal{X}, \hat{\boldsymbol{\lambda}}\right) \text{ the probability of being at word } u \text{ and the } k\text{-th Gaussian component of mixture of state } i \text{ at time } t, \text{ given that } \hat{\boldsymbol{\lambda}} \text{ generates } \mathcal{X}. \text{ It can be calculated as:} \\ &\qquad \bar{\gamma}_{uikt} = \bar{\psi}_{uit} \frac{\hat{\omega}_{ik} \mathcal{N}\left(x_t \middle| \hat{\mu}_{ik}, \hat{\Sigma}_{ik}\right)}{\sum_{k'} \hat{\omega}_{ik'} \mathcal{N}\left(x_t \middle| \hat{\mu}_{ik'}, \hat{\Sigma}_{ik'}\right)} \end{aligned} \tag{4.10}$$

The auxiliary functions presented in (4.7), (4.8) and (4.9) are quite similar to those obtained for the standard (supervised) MLE (c.f. (2.17), (2.18) and (2.19)). In the

unsupervised case shown above, summing over all the words with indexes $u$ and $v$ is necessary. Besides that, the definitions for the joint probabilities $\bar{\psi}$, $\bar{\xi}$ and $\bar{\gamma}$ in the unsupervised case also include word indexes observed at time frames $(t-1)$ and $t$ (c.f. (2.15) and (4.10)). Unsupervised AM training can also be solved in an iterative way using the EM algorithm. Besides the state/frame level alignments, word/state alignments are also required for unsupervised AM parameter estimation. They can be jointly estimated via decoding. The unsupervised EM algorithm can be represented as follows:

**Decoding** Given the current model $\hat{\boldsymbol{\lambda}}$ (as well as the language and pronunciation models), decode $\mathcal{X}$, estimating the state/frame alignment probabilities $P(\mathcal{S}|\mathcal{X}, \hat{\boldsymbol{\lambda}})$ and the word sequence posterior probabilities $P(\mathcal{W}|\mathcal{S}, \mathcal{X}, \hat{\boldsymbol{\lambda}})$. Decoding is guided by the following maximization problem:

$$(\mathcal{W}^*, \mathcal{S}^*) = \arg \max_{\mathcal{W}} \max_{S} P(\mathcal{W}) \cdot P(\mathcal{S}|\mathcal{W}, \hat{\boldsymbol{\lambda}}) \cdot f(\mathcal{X}|\mathcal{S}, \hat{\boldsymbol{\lambda}}) \tag{4.11}$$

**Model update** Given $P(\mathcal{W}, \mathcal{S}|\mathcal{X}, \hat{\boldsymbol{\lambda}}) = P(\mathcal{S}|\mathcal{X}, \hat{\boldsymbol{\lambda}}) \cdot P(\mathcal{W}|\mathcal{S}, \mathcal{X}, \hat{\boldsymbol{\lambda}})$, estimate the new model parameters $\boldsymbol{\lambda}$.

$$\boldsymbol{\lambda}^* = \arg \max_{\boldsymbol{\lambda}} \sum_{\mathcal{W}} \sum_{\mathcal{S}} \sum_{\mathcal{L}} P(\mathcal{L}, \mathcal{S}, \mathcal{W}|\mathcal{X}, \hat{\boldsymbol{\lambda}}) \cdot \log f\left(\mathcal{X}, \mathcal{L}, \mathcal{S}|\mathcal{W}, \boldsymbol{\lambda}\right) \tag{4.12}$$

where the summation is performed over all possible word, state and Gaussian label sequences.

For initialization, an acoustic model trained on a small amount of manually transcribed data can be used. Lamel et al. (2002a) showed that bootstrapping can be done using as little as 10 minutes of acoustic data if the available untranscribed data set is large enough. Alternatively, the initial model can be built by combining seed models trained on other languages (Wheatley et al., 1994; Schultz and Waibel, 2001).

The language and pronunciation models are usually kept fixed over the entire acoustic parameter estimation process. Their quality influence the acoustic estimates obtained, given that they provide some level of supervision for unsupervised AM training.

## 4.2. Approximating with the 1-best hypothesis

Summing over all word and state sequences as in (4.12) is a computationally challenging task. A common way to reduce the computational effort required for unsupervised acoustic model estimation is to consider only the best hypothesis provided by the recognizer.

The 1-best unsupervised AM training approach is illustrated in Figure 4.1. An existing recognition system is used to decode a subset of the untranscribed acoustic data. The automatic transcriptions are then used as ground truth for standard acoustic model parameter estimation. The procedure is reiterated until a suitable convergence criterion is attained.

Different strategies can be applied to use the different training subsets at each iteration. For instance, LIMSI usually uses an incremental training approach, doubling the

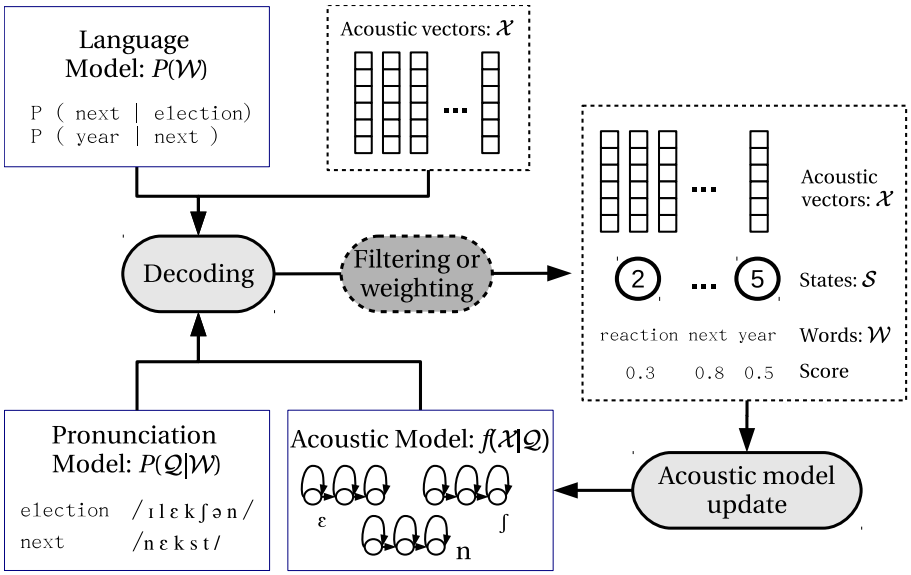



Figure 4.1.: Unsupervised acoustic model training scheme.

amount of data at each iteration (Lamel et al., 2002b). BBN prefers to re-decode the entire training set every iteration (Ma et al., 2006). The impact in acoustic model accuracy of using such strategies is compared in Section 4.4.

### 4.2.1. The influence of recognition errors

The simple substitution of manual transcriptions by automatic ones for acoustic model training is not optimal. Automatic transcriptions contain many errors especially when the initial models are poorly estimated, which is often the case for applications requiring unsupervised training. Dealing with the recognition errors is a major issue in unsupervised AM training, given the fact they can potentially mislead the parameter estimation.

To give an idea of how recognition errors can affect model accuracy, a controlled experiment was conducted. The 72 hour training set `trainQ10` was decoded using an initial AM trained on the 3 hour set `trainR00` and the language model `LM_10src` (see Section 3.2). Since the `trainQ10` manual transcriptions were available, the training set WER could be measured (about 45%). Different training WERs (from 0% to 45%) were simulated by correcting some of the hypothesized words. For each target WER, the hypothesized words were substituted by the reference words if their confidence measures were above a certain threshold. The partially corrected transcriptions were then used as ground truth for standard acoustic model training. These resulting models were then used to decode the 3.5 hour development set `devQ10` in order to assess their quality.

The WER of the `devQ10` set is plotted as a function of the simulated training word (WER) and phone (PER) errors in Figure 4.2. Roughly, an absolute increase of 10% in the training WER (corresponding to about 5% in PER) led to an absolute increase of 1% in the `devQ10` WER. For a single training iteration, the use of automatic transcriptions (45% WER) rather than manual ones (0% WER) results in an absolute increase of 4% in the `devQ10` WER (from 30.7% to 34.8%).

Of course, recognition errors cannot be corrected in such a way in real applications.

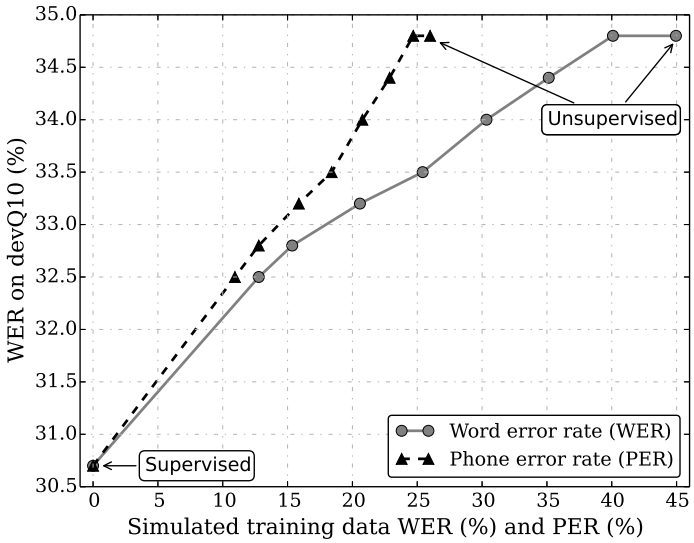



Figure 4.2.: Influence of simulated transcription errors in acoustic modeling. The errors present in the training transcriptions are measured in terms of word error rate (WER) and phone error rate (PER). Points with the same "WER on `devQ10`" correspond to the same acoustic model.

If approximate transcriptions (closed captions) are available though, they can be used to filter errors from the automatic transcriptions (Lamel et al., 2000). In such an approach, closed captions and automatic transcriptions are submitted to forced alignment, and segments failing to align are discarded.

For the more general case where closed captions are not available, a proposed solution is to use the confidence measures provided by the recognizer to filter probable errors out or to weigh the decoding hypotheses. Different confidence-based approaches are detailed in the remainder of this chapter. For a straightforward comparison, they have all been considered as special cases of a generalized unsupervised EM algorithm.

### 4.2.2. Generalizing the unsupervised training algorithm

The generalized EM algorithm described in this section is used to simplify the theoretical comparison among the different confidence-based filtering and weighting approaches described in the following subsections. We will first adopt the following notation. Let

$$C(u, i; t) = P(w_t = u, s_t = i | \mathcal{X}, \boldsymbol{L})$$

be defined as a frame wise confidence score that measures the probability of being at word $u$ and state $i$ at time $t$, given that $\mathcal{X}$ generates the lattice hypothesis $\boldsymbol{L}$. Let us also consider that the probabilities $\bar{\psi}_{uit}$ and $\bar{\xi}_{uvijt}$ presented in (4.10) can be approximated by:

$$\bar{\psi}_{uit} = P(w_t = u, s_t = i | \mathcal{X}, \hat{\boldsymbol{\lambda}}) \approx C(u, i; t) \tag{4.13}$$

$$\bar{\xi}_{uvijt} = P(w_{t-1} = u, w_t = v, s_{t-1} = i, s_t = j | \mathcal{X}, \hat{\boldsymbol{\lambda}}) \approx C(u, i; t-1) \cdot C(v, j; t) \tag{4.14}$$

The latter approximation assumes the independence between the confidence scores calculated for two consecutive frames. Although not strictly true, this approximation greatly simplifies the generalization of the unsupervised training algorithm. It is important to note though that the calculation of the confidence scores usually takes into account the interdependence between consecutive frames as well as between concurrent hypotheses (Wessel et al., 2001). With the approximations from (4.13) and (4.14), the generalized unsupervised acoustic model training algorithm can be summarized as follows:

1. Decode the training data $\mathcal{X}$, generating a lattice $\boldsymbol{L}$ using (4.11).
2. For all occurrences in the lattice $\boldsymbol{L}$, calculate a frame wise confidence score $C(u, i; t)$ for all word $u$ in the vocabulary $\mathcal{V}$ and state $i \in [1, N]$ to be jointly aligned to the frame occurring at time $t$.
3. Modify the confidence score $\bar{C}(u, i; t) = \mathcal{F}\{u, i, t, C(u, i; t)\}$ using a function $\mathcal{F}\{\cdot\}$ specific for each filtering or weighting method.
4. With the modified confidence scores $\bar{C}(u, i; t)$, calculate $\bar{\psi}_{uit}$ and $\bar{\xi}_{uvijt}$ using (4.13) and (4.14) respectively.
5. Update the model parameters using (4.12) and assuming
$$P(\mathcal{W}, \mathcal{S}|\mathcal{X}, \hat{\boldsymbol{\lambda}}) = \prod_t P(w_t = u, s_t = i|\mathcal{X}, \hat{\boldsymbol{\lambda}}) = \prod_t \bar{\psi}_{uit}$$
6. Reiterate until convergence.

Only the function $\mathcal{F}\{\cdot\}$ that modifies the confidence score (Step 3) is redefined to derive each of the unsupervised acoustic model training methods. For instance, if neither filtering nor weighting is applied, this function can be expressed as:

$$\bar{C}(u, i; t) = \begin{cases} 1 & \text{if } w_t^* = u \text{ and } s_t^* = i \\ 0 & \text{otherwise} \end{cases} \tag{4.15}$$

where $w_t^*$ and $s_t^*$ represent, respectively, the word and state aligned to the best decoding hypothesis at time $t$.

### 4.2.3. Confidence measures

Confidence measures provided by the recognition system can be used to filter or weight the decoding hypothesis to reduce the impact of recognition errors.

In the filtering approaches, a segment (sentence, word, phone, frame) is used for training only if its confidence measure is greater than a certain threshold. As confidence measures are not perfect, recognition errors may persist regardless the threshold applied. As a rule of thumb, small thresholds will keep more incorrectly recognized segments, misleading parameter estimation. On the other hand, if a high threshold is chosen, most of the data is excluded and not enough significant information is added to the model, i.e., *the system only learns what it already knows* (Kemp and Waibel, 1999).

As for other system parameters, the threshold can be manually chosen or selected based on an optimization algorithm. A suitable criterion to optimize the confidence threshold is the development set WER obtained by using the acoustic model trained on the filtered audio data. This task can be performed without manual verification, but might be computationally expensive. As many threshold values are evaluated, as many models need to be estimated and used to decode the development set. This procedure can be parallelized to avoid increasing the training time.

An advantage of weighting is that it does not require threshold optimization. The quality of the confidence measures though may affect the performance of the model trained with weighted audio data. This is also the case for the filtering approaches.

### Word-based confidence measures

Word posterior probabilities are probably the most commonly used confidence score for ASR. They can be readily estimated from decoding lattices (Wessel et al., 2001), and to weight or filter the decoding hypothesis used for unsupervised AM training (Kemp and Waibel, 1999; Wessel and Ney, 2001).

A word-based confidence measure can be obtained as follows. Let us consider that a lattice $\boldsymbol{L}$ is generated after decoding a training sample $\mathcal{X} = (x_1, \ldots, x_T)$ of $T$ observation vectors. Let an edge from this lattice be represented by a tuple $e = [\hat{w}, \hat{\mathcal{S}}_w; \hat{t_s}, \hat{t_f}]$, where $\hat{t_s}$ and $\hat{t_f}$ denote starting and finishing times of the edge, $\hat{w}$ is the word associated to the edge and $\hat{\mathcal{S}}_w = (\hat{s}_{t_s}, \ldots, \hat{s}_{t_f})$ is the associated hypothesized state sequence. Posterior probabilities $P(e|\mathcal{X}, \boldsymbol{L})$ can be efficiently calculated for each edge using the forward-backward algorithm.

Let $C_{word}(u, S_u; [t_s, t_f])$ be a confidence measure for a word $u$ aligned to time stamps $[t_s, t_f]$ and having state sequence $\mathcal{S}_u = (s_{t_s}, \ldots, s_{t_f})$. This confidence measure can be defined as the probability of the lattice edge matching these conditions, that is:

$$C_{word}(u, S_u; [t_s, t_f]) = P([\hat{w} = u, \hat{\mathcal{S}}_w = \mathcal{S}_u; \hat{t_s} = t_s, \hat{t_f} = t_f]|\mathcal{X}, \boldsymbol{L}) \tag{4.16}$$

This is not the unique possible definition for a word-based confidence measure. For instance, the sequence of states could be ignored, and the confidence measure would be obtained by summing over all edges matching only words and time stamps. Since the state sequence is fundamental for unsupervised AM training, the definition in (4.16) will be adopted.

For our generalization purposes, let us define the frame wise confidence measure by assigning the word posterior probability to all states aligned with the word:

$$\begin{aligned} C_{word}(u, i; t) =& P(w_t = u, s_t = i|\mathcal{X}, \boldsymbol{L}) \\ =& C_{word}(u, S_u; [t_s, t_f]) \end{aligned} \iff \mathcal{S}_u(t) = i \ \wedge \ t_s \leq t \leq t_f \tag{4.17}$$

where $\mathcal{S}(t)$ denotes the state from sequence $\mathcal{S}$ aligned to time $t$.

### State-based confidence measures

Since acoustic model parameters are estimated on sub-word units, the use of confidence scores estimated at sub-word levels (phones, states) should make more sense for unsupervised acoustic modeling. Gollan et al. (2007) reported improvements by filtering the

training data based on state-based (instead of word) confidence measures. Their proposed measure can be presented as:

$$C_{state}(\bullet, i; t) = \sum_{\substack{e=[\hat{w},\hat{\mathcal{S}}_w;\hat{t_s},\hat{t_f}] \in \boldsymbol{L}: \\ \hat{\mathcal{S}}_w(t)=i \wedge \hat{t_s} \leq t \leq \hat{t_f}}} P(e|\mathcal{X}, \boldsymbol{L}) \tag{4.18}$$

where the summation is performed over all lattice edges having state $i$ aligned to time $t$, with $\hat{t_s} \leq t \leq \hat{t_f}$. Alternatively, $C_{state}(\cdot)$ can be obtained as the marginalization of $C_{word}(\cdot)$ (4.17) for all words in the vocabulary $\mathcal{V}$.

$$C_{state}(\bullet, i; t) = \sum_{\hat{w} \in \mathcal{V}} C_{word}(\hat{w}, i; t) \tag{4.19}$$

### 4.2.4. Weighting by confidence measures

In the 1-best word-weighted unsupervised AM training, the model parameters are estimated taking into account the best decoding hypothesis together with its word-based confidence scores. It can be considered a special case of the algorithm proposed in Section 4.2.2. In this case, the function used to modify the confidence scores (step 3) can be represented as:

$$\bar{C}(u, i; t) = \begin{cases} C_{word}(u, i; t) & \text{if } w_t^* = u \text{ and } s_t^* = i \\ 0 & \text{otherwise} \end{cases} \tag{4.20}$$

where $w_t^*$ and $s_t^*$ represent, respectively, the word and state aligned to the best decoding hypothesis at time $t$ and $C_{word}(\cdot)$ is defined in (4.17).

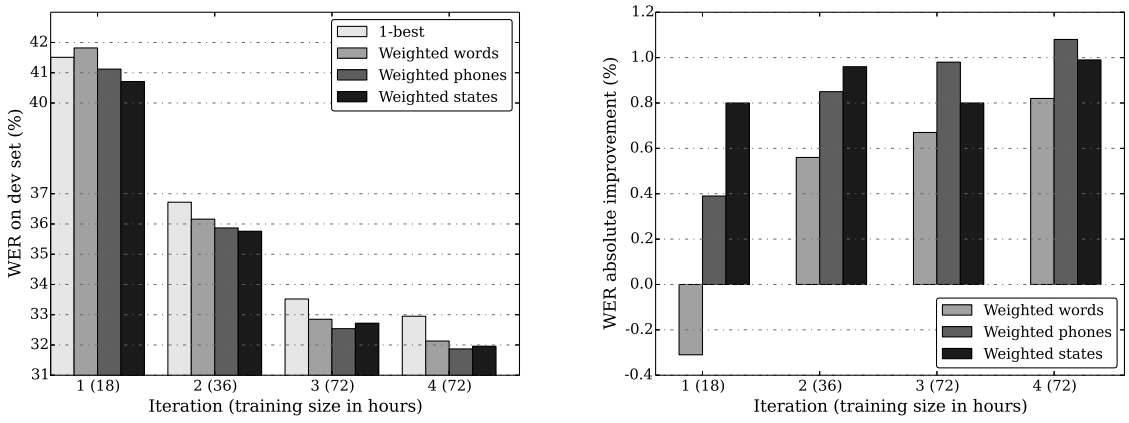
The 1-best state-weighted approach can be defined in two different ways. The first is analogous to (4.20), but using state-based confidence instead:

$$\bar{C}(u, i; t) = \begin{cases} C_{state}(\bullet, i; t) & \text{if } w_t^* = u \text{ and } s_t^* = i \\ 0 & \text{otherwise} \end{cases} \tag{4.21}$$

An alternative definition, that takes into consideration the fact that the state-based confidence measure is a marginalization of the word-based confidence measures, is:

$$\bar{C}(u, i; t) = \begin{cases} C_{word}(u, i; t) & \text{if } s_t^* = i \\ 0 & \text{otherwise} \end{cases} \tag{4.22}$$

This latter definition suggests that weighting at the state level is a smoother approximation to the generalized unsupervised AM training algorithm compared to weighting at the word level. At the state level, a summation is performed over all possible word sequences. At the word level, the summation is approximated by the best word sequence.




(a) (b)

Figure 4.3.: Comparison of 1-best weighting methods for 4 iterations of incremental unsupervised acoustic model training. WER on the development set `devQ10` set (a) and per-iteration absolute WER improvement over the 1-best method (b).

### Experiments

Unsupervised acoustic model training with weighted data was evaluated in a large vocabulary recognition task and compared to the case where neither filtering nor weighting was applied to the automatic transcriptions. Henceforth, this latter will simply be referred to as "1-best" as opposed to the "1-best weighted" variant.

Experiments were carried out using the *improved* EP system (see Section 3.2.4) and using an incremental iterative unsupervised training procedure, as follows.

A bootstrap model was trained on 3 hours of manually transcribed data (`trainR00`). At the first iteration, this model was used to decode a subset of `trainQ10` containing 18 hours of speech data, that is, one fourth of the full training set. The automatic transcriptions were then used for acoustic model training. At the second iteration, the latter models were used to decode half of the `trainQ10` data (the previous 18h + another 18h), which were subsequently used for unsupervised AM training. The same procedure was repeated for the third and fourth iterations using the entire `trainQ10` data set.

At each iteration, four acoustic models were estimated using either the following unsupervised training variants: "1-best" and "1-best weighted" using word, phone[1] or state based confidence measures. The WER on the development set (`devQ10`) was measured at each iteration with each method. The results are summarized in Figure 4.3a. The same language model (`LM_10src`) was used for all tests. Figure 4.3b shows the per-iteration absolute WER improvement of the weighting methods over the "1-best" unsupervised training. For instance, the "Weighted states" model leads to an absolute WER reduction of 1.0% over "1-best" at the fourth iteration (32.0% vs. 33.0%).

Other than the first iteration of the "Weighted words" variant, weighting by confidence

[1]Although they have not been explicitly defined, phone based confidence measures can be obtained in a similar way as the state confidence measures, by changing the matching condition to phone alignments instead of state alignments.

measures always leads to recognition improvements over the "1-best" unsupervised training baseline. Moreover, except for the "Weighted states" method, the absolute improvement increases at each iteration.

At the first iteration, the "Weighted states" method leads to better performance levels with a statistically significant improvement according to the NIST matched pairs sentence-segment word error (MAPSSWE) test (Pallett et al., 1990), with $p < 0.02$. At the second iteration, the state and phone weighting techniques are statistically similar, but significantly better than "1-best" and "Weighted words" ($p < 0.02$). For the last two iterations, the weighting methods perform similarly, but all are better than the "1-best" baseline with an absolute WER improvement of at least 0.8% ($p < 0.001$).

The best result on the `devQ10` set was obtained with the "Weighted phones" method at the fourth iteration (31.9% vs. 33.0% for the 1-best baseline).

### 4.2.5. Filtering by confidence measures

In the 1-best word-filtered unsupervised AM training, the model parameters are estimated taking only the best decoding hypothesis into account, but removing words having confidence scores below a certain threshold. This approach can fit into the algorithm proposed in Section 4.2.2 by using the following function to modify the confidence scores:

$$\bar{C}(u,i;t) = \begin{cases} 1 & \text{if } C_{word}(u,i;t) > C_\tau \text{ and } w_t^* = u \text{ and } s_t^* = i \\ 0 & \text{otherwise} \end{cases} \tag{4.23}$$

where $C_\tau$ is the confidence measure threshold and $C_{word}(u,i;t)$ is defined in (4.17).

The 1-best state-filtered unsupervised AM training can be obtained by replacing $C_{word}(\cdot)$ in the above function by $C_{state}(\cdot)$, defined in (4.19).

#### Experiments

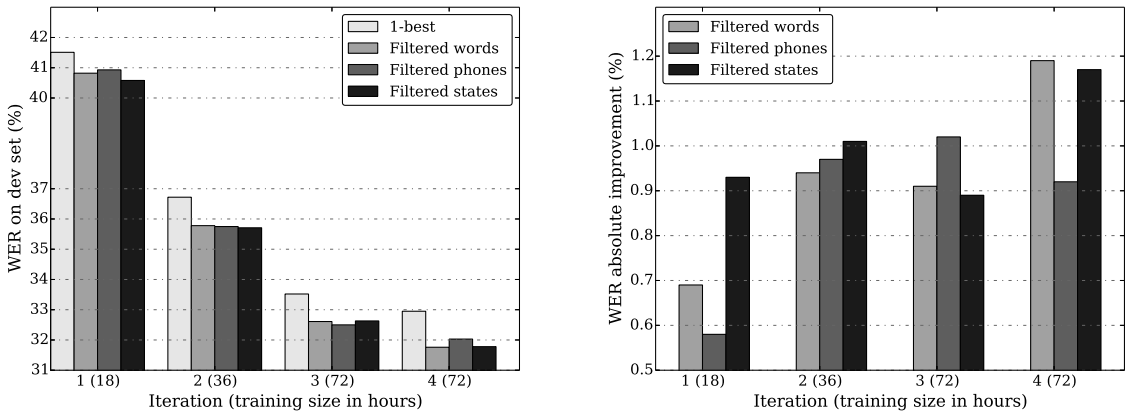
Filtering methods were applied to the same task described in the last section and compared to the "1-best" unsupervised training baseline. The experiments were carried out using a similar setup, but with word, phone or state based confidence measures used to filter low probability training examples.

The WER results on the development set (`devQ10`) are shown in Figure 4.4a. Figure 4.4b shows the per-iteration absolute improvement of the filtering methods over the "1-best" baseline.

The filtering methods always lead to better recognition performance levels than the "1-best" baseline whichever confidence measure is used. Like for weighing, the absolute improvement tends to increase with each iteration.

At the first iteration, the "Filtered states" leads to better performance levels with a statistically significant improvement according to the MAPSSWE test ($p < 0.01$). Although the difference between the filtering methods from the second to fourth iterations is not statistically significant, all the filtering methods outperform the "1-best" baseline ($p < 0.001$).

The best results on `devQ10` were obtained with filtering at the word or state levels. At the fourth iteration, both led to a WER of 31.8%, compared to 33.0% of the baseline, and to 31.9% with the best weighting technique ("Weighted phones").




(a) (b)

Figure 4.4.: Comparison of 1-best filtering methods for 4 iterations of incremental unsupervised acoustic model training. WER on the development set `devQ10` (a) and per-iteration absolute WER improvement over the 1-best method (b).

## 4.3. Approximating with multiple decoding hypotheses

The unsupervised training techniques described in the previous section rely on the use of a unique alignment hypothesis provided by the recognizer to guide the parameter estimation. The standard approach assumes that the best transcription (filtered or not, weighted or not) approximates well the reference.

This work proposes to rely unsupervised acoustic model estimation on multiple decoding hypotheses weighted by their confidence measures. By providing a better approximation to the true audio data transcriptions, the unsupervised AM training algorithm is expected to converge to a better solution, leading to improved model estimates. The multiple alignment hypotheses are obtained from the decoding lattices.

### 4.3.1. The lattice-based training approach

The lattice-based unsupervised acoustic model training can be considered as an approximation to the EM algorithm presented in Section 4.1.2. In this case, the model update equation presented in (4.12) is rewritten as:

$$\boldsymbol{\lambda}^* = \arg\max_{\boldsymbol{\lambda}} \sum_{\mathcal{W},\mathcal{S}\in\boldsymbol{L}} \sum_{\mathcal{L}} P(\mathcal{W},\mathcal{S},\mathcal{L}|\mathcal{X},\hat{\boldsymbol{\lambda}}) \cdot \log f\left(\mathcal{X},\mathcal{L},\mathcal{S}|\boldsymbol{\lambda},\mathcal{W}\right) \tag{4.24}$$

where, in contrast to the 1-best based approach, the first summation is performed over all word and state sequences contained in the decoding lattice $\boldsymbol{L}$.

The number of word and state sequences that are used to re-estimate the acoustic model depends on the size of the decoding lattice, which can be changed during recognition by pruning, expansion and rescoring operations.

Analogously to the 1-best approaches, the lattice-based unsupervised training can be seen as a special case of the generalized algorithm proposed in Section 4.2.2. In this case,

the estimated confidence scores are not modified:

$$\bar{C}(u,i;t) = C_{word}(u,i;t) = P(w_t = u, s_t = i|\mathcal{X},\mathcal{W},\boldsymbol{L}) \tag{4.25}$$

where $C_{word}(\cdot)$ is defined in (4.17),

Optionally, a threshold $C_\tau$ can used to remove low confidence hypotheses, leading to:

$$\bar{C}(u,i;t) = \begin{cases} C_{word}(u,i;t) & \text{if } C_{word}(u,i;t) > C_\tau \\ 0 & \text{otherwise} \end{cases} \tag{4.26}$$

## 4.3.2. Influence of the decoding parameters

In this section, we evaluate how some of the decoding parameters affect the performance of the lattice-based unsupervised training approach.

### Influence of the confidence measures

The performance of the unsupervised training algorithms may vary according to the estimation of the confidence measures, which are all based on the word posterior probabilities. Computing these probabilities depends on the language and acoustic model scores and on the decoding parameters. The formula to calculate the posterior probability for a sequence of words $\mathcal{W}$ can be represented as:

$$P(\mathcal{W}|\mathcal{X}) = \frac{\left(P(\mathcal{W})\cdot f(\mathcal{X}|\mathcal{W})^{1/\alpha}\right)^\beta}{\sum_{\mathcal{W}'\in\boldsymbol{L}}\left(P(\mathcal{W}')\cdot f(\mathcal{X}|\mathcal{W}')^{1/\alpha}\right)^\beta} \tag{4.27}$$

where $P(\mathcal{W})$ and $f(\mathcal{X}|\mathcal{W})$ denote, respectively, the language and acoustic model scores, $\alpha$ the acoustic model scaling factor and $\beta$ the edge exponential scaling factor (Campbell et al., 2007).

Both scaling factors can be used to adjust the word posterior probabilities and thus the confidence scores. More specifically, $\alpha$ controls the likelihood ratio between the acoustic and language models, while $\beta$ controls the likelihood ratio among concurrent hypotheses. As $\beta$ increases, the posterior probability approaches the delta distribution, thereby approaching the 1-best solution. On the other direction, when $\beta = 0$, the confidence scores of all concurrent hypotheses are equal.

We evaluated how the parameter $\beta$ influence the accuracy of models generated using the lattice-based unsupervised training approach. Since the variation of $\alpha$ also changes the 1-best solution (and $\beta$ does not), we have not evaluated its impact.

The experiments were carried out on the *early* European Portuguese system. First, the bootstrap model, trained on 3 hours of manually transcribed audio data (`trainR00`), was used to decode a subset of `trainRVE` containing 11 hours of audio data. Different lattice-based acoustic models were generated on this 11 hour data set for values of $\beta$ varying from 0.1 to 10. Another model was trained on the 1-best audio transcriptions for reference. A language model estimated on about 485M words was used for all tests.

| $\beta$ | 0 (1-best) | 0.1 | 0.5 | 0.8 | 1.0 | **1.2** | **1.5** | 3.0 | 10 |
|---|---|---|---|---|---|---|---|---|---|
| WER (%) | 22.7 | 22.7 | 22.4 | 22.3 | 22.3 | **22.2** | **22.2** | 22.3 | 22.6 |

Table 4.1.: Influence of the edge scale factor ($\beta$) in unsupervised lattice-based acoustic model training using 11 hours of untranscribed training data. The overall WER on the development data (`testR00` + `testR09` + `testV09` + `testE10`) is reported. A threshold $C_\tau = 0.01$ was used. The WER with the bootstrap model is 27.0%.

The systems were evaluated on a development set comprised of 6.4 hours formed by `testR00` (1.3h), `testR09` (1.2h), `testV09` (2.1h) and `testE10` (1.9h). The results are shown in Table 4.1. As a reminder, the overall WER on the four sets with the bootstrap model is 27.0%. The model generated using the 1-best unsupervised training approach obtained a WER of 22.7%, a 16% relative improvement compared to the bootstrap model.

The lattice-based acoustic model obtained an overall WER of 22.2% for the best cases (with $\beta = 1.2$ and $\beta = 1.5$), outperforming the 1-best based model with a 2% relative improvement. The WER varies by only 0.1% absolute for values of $\beta$ ranging between 0.8 and 3.0, which shows a certain robustness of the lattice-based unsupervised training approach with respect to the edge scale factor. When $\beta = 0$, the lattice-based model equals the 1-best based model as can be expected. At the other extreme (with $\beta = 10$) the model accuracy deteriorates due to the fact that all competing hypotheses are weighted equally, increasing the importance of noisy data.

### Influence of the lattice size

The number of hypotheses considered for training is another factor affecting the performance of the lattice-based unsupervised training approach. Intuitively, hypotheses with low confidence scores will not add substantial information to the model. On the contrary, the cumulative effect of many low probability hypotheses can potentially mislead the parameter estimation. In addition, the number of hypotheses also affects the computational complexity required for training.

A straightforward manner to reduce the number of hypotheses is to filter decoding hypothesis segments having confidence scores below a certain threshold. Alternatively, the lattice size can be reduced by decreasing the beam search parameter. In this case, a path is removed from the lattice if its score is below a certain threshold, calculated as a function of the beam search parameter. The variation of the beam parameter modifies the lattice density, which can be measured by the ratio between the number of links (or nodes) over the number of words given by the best hypothesis.

These two methods were assessed using the same experimental setup described in the previous section, that is, performing unsupervised AM training with the 11 hour subset of the `trainRVE` data. First, the beam search parameter was varied from 2 to 10, leading to an average density varying from 4.4 to 347.8 links per word. The experiments showed that varying the lattice density has little impact on the overall WER of the development data set (`testR00` + `testR09` + `testV09` + `testE10`). An absolute WER increase of only 0.2% was observed at the worst case (beam=2, density=4.4) in comparison with the best performance (WER=22.2%), obtained with the default parameter (beam=8, density=123.4).

| Threshold | 0.001 | **0.01** | 0.1 | 0.3 | 0.5 | 0.7 |
|---|---|---|---|---|---|---|
| WER (%) | 22.4 | **22.2** | 22.3 | 22.4 | 22.6 | 22.8 |

Table 4.2.: Influence of the confidence measure threshold in unsupervised lattice-based acoustic model training using 11 hours of untranscribed training data. The overall WER on the development data (`testR00` + `testR09` + `testV09` + `testE10`) is reported. An edge scale factor $\beta = 1.2$ was used. The WER with the bootstrap model is 27.0%.

The impact of confidence measure threshold is slightly more relevant. Table 4.2 shows the results obtained with models trained using different thresholds. The best result was obtained for a threshold of 0.01. The performance is slightly worse when a threshold of 0.001 is applied, possibly due to the presence of noisy data. When the threshold is increased, the system performance is also adversely affected. For threshold values greater than 0.5, only the 1-best hypothesis (furthermore, filtered) is effectively used for training, leading to loss in performance. For a threshold of 0.7 (thus, 1-best) an absolute loss of 0.6% is observed compared to the case with a threshold of 0.01 (WER=22.2%).

### 4.3.3. Comparing lattice-based and 1-best training approaches

An iterative incremental unsupervised training procedure was pursued in order to compare the 1-best and lattice-based unsupervised training approaches. For the lattice-based method, the edge scale factor ($\beta = 1.2$) and pruning threshold ($C_\tau = 0.01$) were optimized for the first iteration and kept fixed during the training process.

In the first iteration, the acoustic models were estimated on a 11 hour subset of the `trainRVE` data. Until the 5th iteration, the amount of training data was doubled at each iteration, each time re-decoding the subset from the previous iteration. In the 5th and 6th iterations, all the 173 hours of the `trainRVE` audio data were used.

The WER performance was evaluated on `testR00`, `testR09`, `testV09` and `testE10`. The results by iteration are reported in Table 4.3. The duration specifies the amount of training data (in hours) used at each iteration. The duration of the lattice-based models takes into account the weight factors applied to the decoding hypotheses, as well as the removal of the low probability segments.

For all the evaluation sets, the lattice-based models outperform the 1-best models. The absolute improvement per iteration varies between 0.3% and 0.6%. According to the MAPSSWE test, these gains are statistically significant ($p < 0.01$). After the last iteration, the absolute overall WER improvement obtained with the lattice-based unsupervised training method is about 0.4% (2% relative) compared to the standard 1-best approach.

### 4.3.4. Comparing lattice-based, 1-best weighting and 1-best filtering approaches

In the last section, the use of multiple decoding hypotheses for unsupervised acoustic modeling was assessed and compared to the standard 1-best unsupervised training. In

| System | Dur | testR00 | testR09 | testV09 | testE10 | Overall |
|---|---|---|---|---|---|---|
| Bootstrap | 3 | 36.1 | 36.2 | 21.4 | 22.5 | 27.0 |
| 1-best (1st iteration) | 11 | 32.7 | 33.1 | 16.0 | 18.1 | 22.7 |
| lattice (1st iteration) | 10 | 32.2 | 32.8 | 15.4 | 17.7 | 22.2 |
| 1-best (2nd iteration) | 22 | 31.1 | 31.3 | 14.6 | 16.6 | 21.2 |
| lattice (2nd iteration) | 20 | 30.5 | 30.9 | 14.2 | 16.2 | 20.7 |
| 1-best (3rd iteration) | 44 | 29.8 | 30.2 | 13.6 | 15.1 | 19.9 |
| lattice (3rd iteration) | 41 | 29.0 | 29.3 | 13.1 | 14.7 | 19.3 |
| 1-best (4th iteration) | 87 | 27.6 | 28.4 | 12.9 | 14.3 | 18.7 |
| lattice (4th iteration) | 80 | 27.5 | 28.0 | 12.6 | 13.9 | 18.4 |
| 1-best (5th iteration) | 173 | 26.6 | 27.3 | 12.9 | 13.5 | 18.1 |
| lattice (5th iteration) | 157 | 26.2 | 26.5 | 12.5 | 13.5 | 17.8 |
| 1-best (6th iteration) | 173 | 26.4 | 27.3 | 12.9 | 13.7 | 18.1 |
| lattice (6th iteration) | 159 | 26.0 | 26.8 | 12.6 | 13.3 | 17.7 |

Table 4.3.: Comparison between the 1-best and the lattice-based acoustic models using an incremental unsupervised training strategy. The WER(%) is reported for each iteration. 'Dur' corresponds to the duration (in hours) of the acoustic data used for training.

particular, the influence of the decoding parameters was evaluated using the *early* European Portuguese ASR system.

This section compares the use of multiple hypotheses with the 1-best weighting and 1-best filtering unsupervised training approaches using the *improved* EP system. More precisely, the 1-best, 1-best weighted at phone level, 1-best filtered at state level and lattice-based unsupervised training methods were assessed using an incremental iterative training procedure.

The experiments were performed as follows. First the bootstrap model, estimated on the 3 hour data set `trainR00`, was used to decode a subset containing 18 hours of the `trainQ10` data set. An acoustic model was estimated on these data using either of the unsupervised training approaches assessed (1-best, 1-best weighted, 1-best filtered and lattice-based). These models were used to decode 36 hours of `trainQ10` (the previous 18h + another 18h). The procedure was repeated for other two iterations using the entire `trainQ10` data set (72 hours).

For the 1-best filtering method, the confidence threshold was optimized at each iteration in order to minimize the WER on the development data set `devQ10`. For the lattice-based method, the threshold was selected at the first iteration and kept fixed. The default edge scale factor ($\beta = 1$) was used.

The models were evaluated on the development set (`devQ10`) at each iteration. The models obtained after the last (4th) iteration, were also assessed on the test sets `testQ10` and `testQ11`. The results are summarized in Table 4.4.

As in the previous experiments, the use of confidence measures for unsupervised acoustic modeling helped to improve the recognition accuracy. The 1-best weighted, 1-best filtered and lattice-based models outperform the baseline 1-best model on about 0.8% absolute on the combined `devQ10`+`testQ10`+`testQ11` set after the last iteration.

At the first iteration, the 1-best filtered and the lattice-based methods outperform

| Method | devQ10 | | | | testQ10 | testQ11 | Overall |
|---|---|---|---|---|---|---|---|
| | 1st (18h) | 2nd (36h) | 3rd (72h) | 4th (72h) | 4th (72h) | 4th (72h) | 4th (72h) |
| Bootstrap | 53.7 | | | | 45.9 | 54.8 | 51.5 |
| 1-best | 41.5 | 36.7 | 33.5 | 33.0 | 26.8 | 33.0 | 30.9 |
| 1-best phone weighted | 41.1 | 35.9 | 32.5 | 31.9 | 26.3 | 32.2 | 30.1 |
| 1-best state filtered | 40.6 | 35.7 | 32.6 | 31.8 | 26.3 | 32.2 | 30.1 |
| Lattice | 40.5 | 35.3 | 32.4 | 31.9 | 26.2 | 32.3 | 30.1 |

Table 4.4.: Comparison of unsupervised acoustic model training approaches using an iterative incremental training strategy. The WER(%) is measured on the development set devQ10 for all iterations, and on the test sets testQ10 and testQ11 for the last iteration. The numbers within parentheses represent the amount of training data (in hours) used at each iteration.

the 1-best and 1-best weighted approaches on the development data set devQ10 (with $p < 0.04$ according to the MAPSSWE test). At the second iteration, the lattice-based model obtains the best performance levels ($p < 0.005$). At the 3rd and 4th iterations, no difference in performance is observed between the 1-weighted, 1-best filtered and lattice-based models.

These results suggest that the use of multiple decoding hypotheses for unsupervised training is well-suited for small training data sets, which usually generate automatic transcriptions with high training WERs. As the errors contained in the automatic transcriptions reduce, the 1-best solution becomes more reliable. This could be taken into account for the lattice-based training approach by, for instance, decreasing the edge scale factor from one iteration to another. However, I suppose that only marginal gains could be obtained by doing so.

## 4.4. Other training strategies

In unsupervised AM training approaches, dealing with recognition errors usually relies on the use of confidence measures. The idea is to remove potential errors from the training data or reduce their impact by weighting the training data, avoiding then the model parameters to overfit on wrongly assigned classification labels.

Overfitting on training data can also be avoided by using a cross-validation (CV) approach. Shinozaki et al. (2009), for instance, applied a CV method to improve the performance of an unsupervised acoustic model adaptation approach. One of the problems with CV techniques relates to the number of choices of independent training subsets (folds). If, on one hand, the use of many folds increase the possibility of getting better generalization, it also leads to data fragmentation. For the same task mentioned above (unsupervised acoustic model adaptation), Kubota et al. (2010) proposed an aggregated technique that reduces data fragmentation. Different from CV, it allows overlap between training subsets to increase the amount of training data at each iteration.

Both methods could be applied to unsupervised AM training. However, besides the fragmentation problem mentioned, another technical issue could arise: the use of CV or aggregated methods would require managing different training subsets. Even if this might

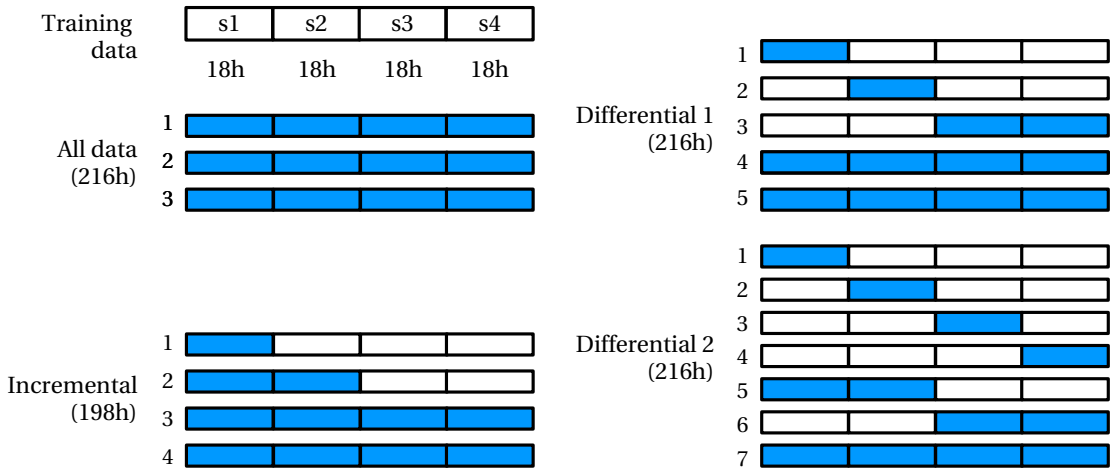



Figure 4.5.: Unsupervised acoustic model training strategies assessed in this work for a training data set divided into four 18 hour subsets. The colored boxes denote the subset used during each iteration. The numbers within parentheses denote the total amount of audio data decoded for all iterations.

be considered a minor issue, dealing with many acoustic models and decoding setups for every iteration increase the complexity of the training procedure.

The main goal of a CV-based unsupervised AM training approach is to avoid the propagation of the recognition errors from one iteration to another. This effect can be approximated by carefully defining a training strategy that uses different portions of the audio data across the training iterations. These strategies are discussed below.

## Training strategies

A straightforward training strategy used for unsupervised acoustic modeling is to used the entire training data set at every iteration (Ma et al., 2006). Alternatively, an incremental procedure can be performed by increasing the amount of data at each time and re-decoding data from previous iterations (Lamel et al., 2002b).

In the *all data* strategy, wrongly assigned labels might be reinforced from one iteration to another. Intuitively, the *incremental* strategy could help to reduce this effect: in every iteration, there is always a portion of the data that was not used to estimate the models from previous iterations.

If this assumption is correct, another proposed training strategy, henceforth *differential*, would work even better. Instead of systematically estimating the models from data obtained at the current and previous iterations, the new acoustic model could be estimated on an entirely new data set. Of course, it would be necessary to increase the amount of data at some point to avoid data fragmentation. Two possibilities were considered in this work: increasing the amount of data after each iteration; or increasing the amount of data after having completed a cycle, that is, after spanning all training subsets.

The aforementioned training strategies are illustrated in Figure 4.5 for a training set equally divided into four subsets. This division was chosen to keep about the same training computational complexity for all methods compared. In particular, about 200 hours of audio data are decoded at each method for the whole iterative procedure.

| Training strategy | # of Iter. | Amount of data decoded (in hours) | WER(%) | | | |
|---|---|---|---|---|---|---|
| | | | 1-best | 1-best weighted | 1-best filtered | Lattice |
| All data | 3 | 216 | 31.5 | 30.4 | 30.7 | 30.3 |
| Incremental | 4 | 198 | 30.9 | 30.1 | 30.1 | 30.1 |
| Differential 1 | 4 | 144 | 30.6 | 30.2 | 30.1 | 30.1 |
| Differential 1 | 5 | 216 | 30.3 | 29.9 | 29.8 | 29.7 |
| Differential 2 | 7 | 216 | **30.2** | - | **29.7** | **29.6** |

Table 4.5.: Comparison of unsupervised acoustic model training strategies. The WER was evaluated on about 10.3h of data comprised of `devQ10`, `testQ10` and `testQ11`. The number of iterations and the total amount of training data decoded are reported. All the acoustic models were trained on the 72 hour data set `trainQ10`.

The number of iterations performed on each method is not the same. For instance, three and seven iterations have to be done for the 'All data' and 'Differential 2' strategies, that is, estimating three and seven acoustic models respectively. However, the number of iterations is not the most important factor determining the complexity of the training strategy: decoding the training data is the most time consuming step of unsupervised AM training approaches. Thus, the experiments were carried out in a manner that all training strategies required the same amount of decoding time.

## Experimental results

In order to compare the four training strategies presented, a set of experiments was carried out using the `trainQ10` acoustic data. This training set was divided into four non-overlapping subsets, each containing about 18 hours of audio data. The number of iterations performed is exactly as presented in Table 4.5, all data (3), incremental (4), differential 1 (5) and differential 2 (7). For each strategy, the 1-best, 1-best weighted, 1-best filtered and lattice-based unsupervised AM training approaches were used.

These strategies were assessed on about 10.3 hours of data, that is, the development set `devQ10` and the test sets `testQ10` and `testQ11`. Table 4.5 summarizes the results obtained. In addition to the WER measured on the combined test sets, the number of iterations and total amount of training data decoded are reported. The results are reported for the last iteration of each method, except for the 'Differential 1' strategy, for which both the 4th and 5th iteration results are presented.

The influence of the training strategy is clearer for the standard 1-best unsupervised training approach. This can be justified by the fact that the other methods (1-best weighted, 1-best filtered and lattice) also used confidence based measures to reduce the impact of the recognition errors. The 'Differential 2' strategy leads to a WER absolute improvement of 1.3% compared to the 'All data' strategy when the 1-best based acoustic models are used to decode the test data.

The use of a *differential* strategy (1 or 2) leads to better performance levels for the 1-best weighted, 1-best filtered and lattice-based models. After the 4th iteration, the 'Differential 1' strategy leads to same performance levels as the 'Incremental' strategy,

even if less time is spent for decoding the training data (216h vs. 144h). At the 5th iteration, 'Differential 1' leads to an absolute WER improvement around 0.3–0.6% compared to 'Incremental' for a similar amount of time spent for decoding. The 'Differential 2' strategy leads to the best overall performances for the cases where it was applied, being marginally better than 'Differential 1' (a 0.1% WER absolute improvement).

It is worth noting that the best performance levels for all strategies were obtained by using the lattice-based unsupervised training approach. The best overall WER was achieved with the 'Differential 2' strategy and the lattice-based models (29.6%). This corresponds to an absolute gain of 1.9% compared to the baseline 1-best 'All data' strategy (31.5%) and 1.3% compared to the 1-best 'Incremental' strategy (30.9%).

## 4.5. Summary

In this chapter, the unsupervised acoustic model training algorithm was investigated. This approach was first presented in the form of an incomplete data problem in which the word sequence is unknown, in addition to the state and Gaussian label sequences.

With a controlled experiment, it was shown that dealing with the recognition errors is a major issue in unsupervised AM training. Different weighting and filtering confidence based approaches used to reduce the impact of errors were described. They were all presented within a proposed theoretical framework as approximations to a generalized EM algorithm.

The weighting and filtering approaches were assessed with confidence measures based on words, phones and states. The experiments suggested that the use of state-based (instead of word-based) confidence measures is beneficial during the first iteration of an incremental unsupervised acoustic model training process. However, no significant improvement was observed after the last iteration.

We proposed the use of multiple decoding hypotheses weighted by their confidence measures to guide the acoustic model parameter estimation. The use of multiple hypotheses extracted from the decoding lattices was justified theoretically as a smooth approximation to the generalized EM algorithm. The influence of different decoding parameters was empirically evaluated. The lattice-based method proved to be robust to the variation of the beam search parameter and behaves quite well for large variations of the edge scaling factor and the confidence measure threshold.

For all the experiments performed, the lattice-based unsupervised training approach led to the best overall performances compared to the standard 1-best, the 1-best weighted and the 1-best filtered approaches. The improvements were significantly better for the first iterations of training, when the acoustic model estimates are quite inaccurate. As long as the model ameliorates, the improvement led by the lattice-based approach reduces compared to the 1-best solution.

The lattice-based and the weighting approaches provide easier training setups compared to filtering, which requires the optimization of the confidence threshold parameter. The main disadvantage of the lattice-based approach is the increase in time required to estimate the acoustic models (about 1.5 times compared to the 1-best approaches). However, this difference should be negligible, since most of the computational complexity of unsupervised acoustic modeling is due to the decoding of the training data.

As an attempt to avoid the propagation of the recognition errors from one iteration to another, two variants of a *differential* training strategy were presented. Both led to better recognition performance levels than two other reported strategies, *all data* (Ma et al., 2006) and *incremental* (Lamel et al., 2002b).

The combination of two techniques proposed in this chapter, the lattice-based unsupervised AM training and the differential strategy, led to the best overall WER performance (29.6%). Absolute improvements of 1.9%, 1.3% and 0.5% were respectively obtained over three reference unsupervised training methods: 1) 1-best with the 'All data' strategy; 2) 1-best with the 'Incremental' strategy; and 3) 1-best filtered with the 'Incremental' strategy.

When trained on the same audio data, the supervised acoustic models outperforms their unsupervised counterpart on about 7.4% relative (27.4% vs. 29.6%). This difference is on the same order as reported in previous work (Lamel et al., 2002b; Ma et al., 2006; Novotney et al., 2009). Furthermore, the relative difference between supervised and unsupervised approaches seems to be invariant to the amount of data used in acoustic modeling. However, it is worth reminding that the performance of unsupervised models can be improved by adding more automatically transcribed data with a small increase in terms of development cost.

# Chapter 5

# Unsupervised multi-layer perceptron training

## 5.1. Introduction

As presented in Chapter 4, unsupervised acoustic model training methods are suitable when sufficient manually transcribed audio data is not available, helping to reduce the human effort required to improve the performance of large vocabulary continuous speech recognition systems. Most of the reported work in unsupervised acoustic modeling make use of *short-term raw* acoustic features like PLP (Hermansky, 1990) cepstral features. Such features are extracted directly form the audio stream via a power spectrum analysis as described in Section 2.2.4.

In recent years, there has been growing interest to increase the discriminative power of the acoustic features. In particular, many recent systems rely on multi-layer perceptron (MLP) classifiers for feature extraction. MLP-based features have been successfully used in a variety of LVCSR tasks (Ellis et al., 2001; Zhu et al., 2005) and, when used in combination with PLP features, can lead to substantial gains in recognition performance (Fousek et al., 2008a,b; Lamel et al., 2011).

In reported work, the MLP classifiers have been trained in a supervised manner, that is, relying on the use of manually transcribed audio data. Even when manually transcribed data for the target language are not available, acoustic modeling can still take advantage of the discriminative features. A proposed solution is to use features extracted from MLPs trained for other languages (Stolcke et al., 2006; Tóth et al., 2008) or for multiple languages (Grezl et al., 2011; Veselý et al., 2012; Vu et al., 2012).

An alternative solution to allow the use of MLP features without relying on manually annotated data is proposed in this chapter. As is done for the HMM parameters, the MLP classifiers are estimated on automatically transcribed data. MLPs trained in an unsupervised manner are empirically compared to cross-lingual MLPs (Stolcke et al., 2006; Tóth et al., 2008). An unsupervised MLP adaptation scheme is also proposed. In this case, an MLP trained for another language is retrained upon the untranscribed data of the target language. Retraining is based on the approaches proposed in Thomas et al. (2012) and Grézl et al. (2014).

The remainder of this chapter is organized as follows. In Section 5.2, a general description of the MLP model is presented, showing current applications of MLPs in acoustic

modeling. Section 5.3 presents the experimental work on the use of bottleneck MLP classifiers for feature extraction. We first evaluate the impact of using manually or automatically transcribed data for the estimation of the HMM and MLP parameters. Then, models estimated with features extracted from unsupervised MLPs or cross-lingual MLPs are empirically compared. A summary is presented in Section 5.4.

## 5.2. Multi-layer perceptron neural networks

As the name suggests, multi-layer perceptron is a type of artificial neural network model formed by multiple layers, in which nodes have a non-linear activation function and are fully-connected to the next layer by means of weighted links (see Figure 5.1).

Let us denote a certain layer containing $L$ neurons by a vector $\boldsymbol{y}$, the previous layer containing $K$ neurons by a vector $\boldsymbol{x}$ and the weighted links by a $K \times L$ dimensional matrix $\boldsymbol{A} = \{a_{kl}\}_{k\in[1,K],l\in[1,L]}$. The value of the $l$-th coefficient of the vector $\boldsymbol{y}$ (neuron $y_l$) can be calculated by:

$$y_l = f\left(b_l + \sum_{k=1}^{K} x_k \cdot a_{kl}\right) \tag{5.1}$$

where $b_l$ represents the $l$-th element of a bias vector $\boldsymbol{b}$ that allows the shift of the activation function with respect to the input vector $\boldsymbol{x}$.

The non-linear activation function $f(\cdot)$ depends on expert choices. Usually, *sigmoidal* activation functions are used for internal layers. In the output layer, a *softmax* function is commonly used to allow the calculation of class posterior probabilities. Thus, the neuron activation functions can be represented as:

$$f(z_l) = \begin{cases} \dfrac{1}{1+e^{-z_l}} & \text{all layers but the output layer} \\ \dfrac{e^{z_l}}{\sum_k e^{z_k}} & \text{for the output layer} \end{cases} \tag{5.2}$$

MLPs have been used for acoustic modeling with two main objectives. First, they can be used to model the output probability density of HMM states, replacing Gaussian mixture models (Section 5.2.2). Alternatively, they can be used to provide acoustic features for a standard HMM/GMM based acoustic model architecture (Section 5.2.3).

### 5.2.1. MLP training

The MLP parameters, namely the weight matrices and bias vectors, are often estimated using the so-called back-propagation algorithm (e.g. (Rojas, 1996, Chapter 7)). This algorithm attempts to minimize the multi-class classification errors, which is performed in two main steps. First, the input vectors are processed by the MLP in order to predict output values. Predicted values and the true reference are used to calculate an error vector. In the second step, the errors are propagated backward and used to update the model parameters.

The back-propagation algorithm requires pairs of input/output vectors. In the case of acoustic modeling, it means having data labeled at the phone or HMM state level. In other words, the frame/phone or frame/state alignment is required. In the same way it is done for HMM training, these associations can be obtained via forced alignment if manual audio data transcriptions are available. In this chapter, it is proposed to use partially accurate alignments provided by an existing speech recognizer.

### 5.2.2. Hybrid HMM/MLP architecture

The hybrid HMM/MLP architecture was introduced for automatic speech recognition in Morgan and Bourlard (1995). In this architecture, an MLP is used to model the HMM state observation probability densities. Thus, instead of a GMM, each state is represented by a node from the output layer of the MLP.

In early years, the hybrid models did not lead to sufficiently high gains of recognition accuracy to justify their usage to the detriment of HMM/GMMs. Their use was limited by their inherent high computational complexity, which restricted the MLPs to have only a few layers and units (especially output units).

The hybrid architecture has gained a lot of popularity in the last few years. Due to the increase in power of easily available hardware and development of new training methods, the use of MLPs with several layers (known as deep neural networks (DNN)) was shown to outperform GMMs for a variety of speech recognition tasks (Mohamed et al., 2011, 2012; Hinton et al., 2012). By increasing the number of layers, more linear space transforms are allowed, increasing the ability of the system to capture underlying model structures. This advent has considerably increased the strength of the HMM/MLP architecture, which has risen to the new state-of-the-art in terms of acoustic modeling. As a drawback, DNNs are slower to train in comparison to HMM/GMM models.

Although the use of DNN based acoustic models is beyond the scope of this thesis, it has been shown that unsupervised training approaches are very likely to be used in such an architecture (Laurent et al., 2014). Future research directions on this topic are discussed in Part IV.

### 5.2.3. MLP for feature extraction

Another common use of neural networks is for acoustic feature extraction (Hermansky et al., 2000; Ellis et al., 2001; Zhu et al., 2005; Grézl et al., 2007; Fousek et al., 2008a). In this solution, the MLP is used to provide feature vectors which are used to estimate standard HMM/GMM based acoustic models. Due to the complementary characteristics of MLP and PLP based features, their combination often leads to substantial gains in recognition performances (Fousek et al., 2008b).

MLP features are derived in three steps. First, a *raw* feature vector, typically covering a wide temporal context (100–500 milliseconds), is extracted from the audio signal via a power spectrum analysis. This input vector is processed by the MLP, from which a feature vector is extracted. A decorrelation transform is applied to this feature vector to produce the final MLP feature vector.

The MLP features can be extracted from the output layer (Hermansky et al., 2000), being, therefore, *probabilistic* features or from a hidden (the *bottleneck*) layer (Grézl et al.,

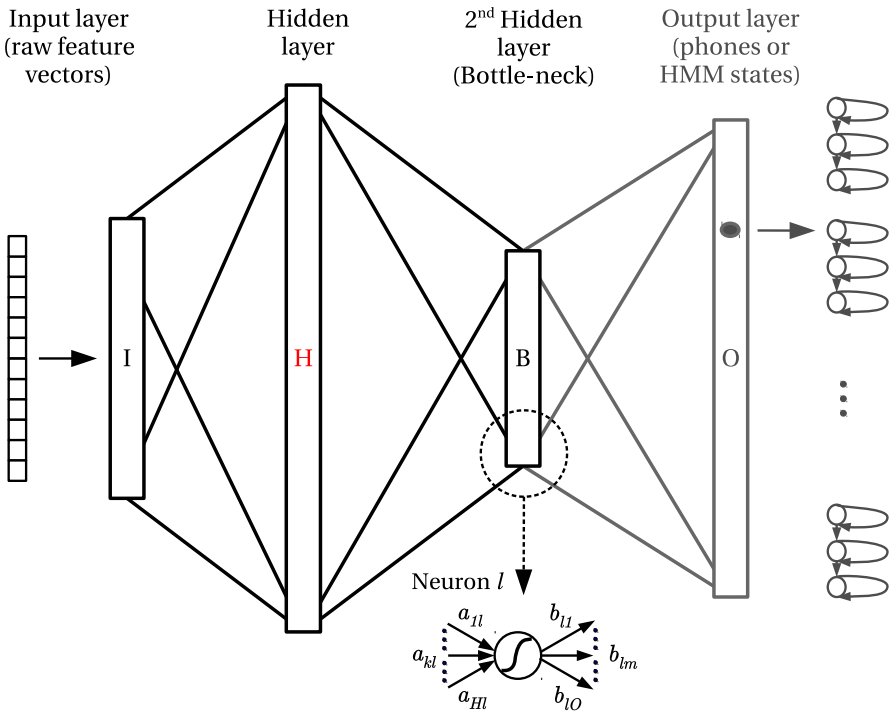



Figure 5.1.: A 4-layer bottleneck multi-layer perceptron (MLP) architecture. Features are extracted from the third-layer (the bottleneck) and processed by PCA transformation. The output layer is used only during training.

2007). In this chapter, bottleneck features were used.

## 5.3. Unsupervised bottleneck MLP training

The bottleneck features are extracted from a hidden layer of an MLP classifier. The main advantage of such a structure is the independence between the size of the feature vector and the number of training targets. This topology can be used to target MLP training to sub-phoneme classes, such as HMM states, without increasing the size of the acoustic feature vector (Grézl et al., 2007).

The 4-layer bottleneck MLP used in this work is represented in Figure 5.1. The raw input vector is based on a temporal pattern (TRAP) analysis (Schwarz et al., 2004): long energy trajectories for a 300 millisecond sliding window in 19 Mel-scale frequency sub-bands of the spectrogram are computed and projected using a discrete cosine transform (DCT). 25 coefficients per band are retained, yielding a 475-dimensional input vector.

The first hidden layer is large (3500 units) enough in order to provide necessary modeling power. The second hidden layer, the bottleneck, contains as many nodes as the desired size of the feature vector. A 39-dimensional vector is generally used in our systems for easy comparison with the PLP-like feature vectors. The output layer computes the target class posterior probabilities. In this work, context-independent HMM phone states were used as targets, since they were shown to outperform phone targets (Grézl et al., 2007). The size of the output layer is proportional to the size of the phone list. The feature vector extracted from the bottleneck layer is decorrelated using a PCA transformation.

As stated in Section 5.2.1, MLP training requires an alignment between the input frame vectors and the state (or phone) output target labels. This task usually relies on manually transcribed data, where the texts are aligned to speech. In order to train an MLP even if manual transcripts are not available, we propose to use the alignment hypothesis provided by a recognizer as output class labels, in a similar way as was performed for the HMM parameters.

In the following subsections, an empirical comparison between supervised and unsupervised MLP training is performed. In addition, MLPs trained in an unsupervised manner are compared to cross-lingual MLPs. The experiments were carried out using a similar setup, which is described in the following section.

### 5.3.1. Experimental setup

The experiments were performed using the LIMSI European Portuguese ASR system (Section 3.2.4). The setup used here is slightly different from the one used for the experiments in Chapter 4. It uses the complete acoustic training data and is executed as follows. First, a bootstrap acoustic model was estimated on the `trainR00` data. This model was used to initiate an incremental iterative procedure to decode and estimate a model on the 173 hour `trainRVE` data set using a lattice-based unsupervised training approach (Chapter 4). This is the model "lattice (6th iteration)" shown in Table 4.3. This latter model was used to decode the 72 hour `trainQ10` and the 71 hour `trainQ11` data sets. In total, 316 hours of audio data (173 + 72 + 71) were used for the experiments reported in this chapter.

The bottleneck MLP architecture shown in Figure 5.1 was used. It uses a 475-dimensional TRAP-DCT acoustic feature vector as input. The first hidden layer has 3500 nodes. The bottleneck layer produces a 39-dimensional feature vector. The output layer contains 109 HMM state targets. There are 35 three-state phones and four units with a single state (silence, breath and two hesitation markers). The MLP feature vector is concatenated to a 39-dimensional PLP-like feature vector (12 cepstral features + log energy along with their first and second derivatives). A 3-dimensional pitch feature vector (pitch with its first and second derivatives) is also appended, resulting in a 81-dimensional feature vector (MLPPLPF0).

MLP training was performed using the QuickNet software (Jonhson, 2004). A similar training procedure was adopted for all cases. First, the data were randomized and split into three non-overlapping subsets containing about 13%, 26% and 52% of data for training, with the remainder reserved for cross-validation. Training is performed in six epochs. The three first epochs use 13% of data, the next two 26% and the last one 52% of data. The learning rates change for each different training subset. In some cases (they will be explicitly mentioned) a fourth epoch was carried out using the entire training data set. The PCA transforms were estimated using a randomly selected subset of the training data.

For all three training subsets (`trainRVE`, `trainQ10` and `trainQ11`), the 1-best hypothesis and the decoding lattices were generated to be used for unsupervised training. More specifically, the MLP parameters were estimated using the 1-best decoding hypothesis, while the lattices were used to estimate the HMM/GMM parameters.

A language model trained on 485M of words (*early* EP system setup Section 3.2.3) was used only to decode the `trainRVE` data. Other than that, all experiments use the language

| Acoustic Model | Amount of data (in hours) | LM | PLP | MLPPLPF0 |
|---|---|---|---|---|
| `trainRVE` / Unsupervised (baseline) | 173 | `LM_10src` | 33.0 | - |
| `trainQ10` / Unsupervised | 72 | `LM_10src` | 31.1 | 29.9 |
| `trainQ10` / Supervised | 72 | `LM_10src` | 29.1 | 27.3 |
| `trainQ10` / Supervised | 72 | `LM_11src` | 28.7 | 27.0 |

Table 5.1.: Comparison of PLP and MLP based acoustic models trained in a supervised or an unsupervised manner. The WER(%) is measured on the development set `devQ10`. `LM_11src` includes the `trainQ10` manual transcriptions in the training corpus, while `LM_10src` does not.

model `LM_10src`, which was trained on about 639M words and does not contain the manual transcriptions of the `trainQ10` data (see Chapter 3). Another model `LM_11src`, which contains the `trainQ10` transcriptions, was used (for reference) together with the MLP and the HMM trained in a supervised manner.

### 5.3.2. Comparing PLP and MLP based acoustic models

The first experiments were carried out to assess the impact of the MLP bottleneck features in acoustic modeling. This comparison made use of `trainQ10` data to train both the MLP and the HMMs in order to be able to compare the supervised and unsupervised training approaches. The systems were evaluated on the development set `devQ10`.

The results are summarized in Table 5.1. The PLP baseline model trained on the `trainRVE` data obtained a WER of 33.0% on `devQ10`. The unsupervised PLP-based acoustic model obtained a WER of 31.1% on `devQ10`, that is, a 5.2% relative improvement over the baseline. This result is different from the one presented in Table 4.4 (31.9%), given the different initial conditions of the two experiments. The use of features extracted from an MLP trained in an unsupervised manner and combined with the PLP+F0 features led to an additional gain in performance, obtaining a WER of 29.9%. This represents a 3.9% relative gain compared to the unsupervised model using only PLP features.

As can be expected, the use of manual (rather than automatic) transcriptions obtains better performance levels for all conditions. When a supervised training approach is used to estimate both the MLP and the HMM parameters, WERs of 29.1% and 27.3% are obtained, respectively, for the PLP and MLPPLPF0 based acoustic models. This represents relative differences of 6.4% and 8.7% with respect to their unsupervised counterparts using the same acoustic training data and the same language model. At a first glance, this difference may appear fairly large, but it is worth recalling that automatic transcriptions are effortless to obtain compared to manual transcriptions. As we will show in Section 5.3.6, the inclusion of more untranscribed data in the HMM and MLP training corpus shorten the difference with respect to the fully supervised approach (leading to a WER of 27.4% on `devQ10`).

When the `trainQ10` manual transcriptions are included in the language model training corpus (`LM_11src`, last row), an additional absolute gain of 0.3–0.4% is obtained.

| | HMM Supervised | HMM unsupervised |
|---|---|---|
| MLP Supervised | **27.3** | 29.4 |
| MLP Unsupervised | 28.3 | **29.9** |

Table 5.2.: Comparison of supervised and unsupervised training approaches used to estimate either MLP and HMM parameters. The WER(%) is measured on the development set `devQ10`. MLP and HMM parameters were estimated on the 72 hour `trainQ10` data set. `LM_10src` was used for decoding.

### 5.3.3. Comparing unsupervised and supervised MLP and HMM training approaches

In the previous section, a comparison between PLP and MLP (precisely MLPPLPF0) based models was performed. In particular, it was shown that the supervised system outperforms the unsupervised one by about 8.7% relative. This difference is higher than for the PLP based models (6.4%). Given this scenario, a logical question is: what is the impact of supervised and unsupervised training on each of the components, MLP and HMM? This section aims to empirically answer this question.

The experimental setup is similar as in the last section. The 72 hour training `trainQ10` data were used to estimate either the MLP and the HMM parameters using either a supervised and an unsupervised approach. Four acoustic models were estimated on the same data using the possible combinations of MLP supervised/unsupervised and HMM supervised/unsupervised. The models were evaluated on the development set `devQ10` using the same language model, `LM_10src`.

Table 5.2 summarizes the results obtained. Systems shown on the main diagonal correspond to those where both components (MLP and HMM) are trained with manual (top left) or automatic (bottom right) transcriptions, leading to WERs of 27.3% and 29.9% respectively.

The system in which only the HMM is trained in an unsupervised manner (top right) has an absolute performance loss of 7.1% compared to the fully supervised system. However, when only the MLP is trained in an unsupervised way (bottom left), the relative difference with respect to the fully supervised system is only 3.5%. Using an unsupervised HMM training approach, the supervised MLP (top right) outperforms the unsupervised MLP (bottom right) on only 1.7% relative (0.5% absolute). These results suggest that the MLP parameter estimation is more robust to the recognition errors present in the training data than the HMM parameter estimation.

### 5.3.4. Filtering by confidence measures

In Chapter 4, it was shown that the use of confidence measures for filtering or weighting are very helpful to improve the unsupervised acoustic model training approach. We therefore decided to evaluate if they could also benefit the unsupervised MLP parameter estimation.

Five different MLPs were estimated using a portion of `trainQ10`. For each case, frames having state based confidence measures (see Section 4.2.3) below a certain threshold were discarded, keeping 100%, 90%, 80%, 60% or 50% of the data. An unsupervised acoustic model was trained with features extracted from each of these MLPs, and combined with the PLP and pitch features. The HMM parameters were estimated on all

the data, using the decoding lattices of `trainQ10`. These five models were evaluated on `devQ10`.

No difference in performance was observed for the models estimated on features generated from the bottleneck MLPs trained on 100%, 90%, 80% and 60% of data. The WER fluctuates less than 0.1%, all leading to a WER of 29.9%. A small degradation in performance (WER = 30.2%) was observed when the MLP is trained on only 50% of data.

Given that filtering the training data did not lead to improvements, and the WER obtained with unsupervised MLP is close to the one obtained with supervised MLP (a 0.5% absolute difference, c.f. Table 5.2), further experiments involving weighting by confidence measures were not performed.

### 5.3.5. Using additional untranscribed acoustic data

In the previous experiments, the MLP and HMM components were trained using only the `trainQ10` data set for comparison purposes between supervised and unsupervised approaches. In this section, the effect of adding other unstranscribed data to train the MLP and HMM components was evaluated.

Three conditions were tested. First, only the 72 hour `trainQ10` set was used, as in the previous experiments. Second, the training subset `trainQ11` was added, totalizing 143 hours. For the third condition, the `trainRVE` data were included, totalizing 316 hours of speech. Data were added respecting this sequence mainly due to the similarity between the `trainQ10` and `trainQ11` sets, which come from the same type of data (the "Quaero" data, as described in Chapter 3).

These conditions served to estimate either the MLP or the HMM parameters via unsupervised training. Several, but not all the possible combinations were assessed. In particular, the HMM parameters were estimated on equal or smaller amount of training data in comparison to the MLP parameters. All the systems were evaluated on `devQ10`.

The recognition performance results are reported in Table 5.3. The first column shows acoustic models trained using only `trainQ10`, but with MLP features (in addition to the PLP and pitch features) extracted from MLPs trained on varied amounts of data. By doubling the amount of data used for MLP estimation (first to second row), the WER decreases from 29.9% to 29.5%. This is almost the same performance obtained with the use of an MLP trained on the manual transcriptions of `trainQ10`. The performance is further improved when more 173 hours of untranscribed data (`trainRVE`) are used for MLP training (third row). A WER of 28.6% is obtained.

On the other hand, the additional data for the estimation of the HMM parameters does not always lead to a gain in recognition performance. When all data are used to estimate the bottleneck MLP parameters (third row), the best performance is achieved with acoustic models trained on `trainQ10` + `trainQ11`, obtaining a WER of 27.8%. A loss in performance (0.1%) is observed when the `trainRVE` data are also included in the HMM training set.

This latter effect is known and can be justified by the fact that part of the `trainRVE` data come from slightly different types of source with respect to the test set. There exist different adaptation techniques that can be used to include mismatching data for acoustic modeling and avoid a loss in performance as was observed here. This topic will be treated

| MLP | Acoustic Models | | |
|---|---|---|---|
| | trainQ10 (72h) | trainQ10 + trainQ11 (143h) | trainRVE + trainQ10 + trainQ11 (316h) |
| trainQ10 (72h) | 29.9 | - | - |
| trainQ10 + trainQ11 (143h) | 29.5 | 28.2 | - |
| trainRVE + trainQ10 + trainQ11 (316h) | 28.6 | **27.8** | 27.9 |

Table 5.3.: Effect of adding unstranscribed data for unsupervised MLP and HMM parameter estimation. The WER(%) was measured on the development set devQ10. The duration of each training set is shown in parenthesis.

in the third part of this thesis.

### 5.3.6. Comparing unsupervised and cross-lingual MLPs

Stolcke et al. (2006) is the first who reported successful use of MLPs to produce features for a different task or language. Another example of cross-lingual portability was reported in Tóth et al. (2008), where an English MLP was used in a Hungarian ASR system. In both cases, probabilistic MLP features were used with supervised training of the acoustic models. We have also reported the successful use of cross-lingual bottleneck MLPs with the unsupervised estimation of the HMM parameters in Roy et al. (2013). Beyond the results published in this latter work, we have observed same performance levels with the use of a supervised English MLP (25.0%) and an unsupervised Hungarian MLP (24.9%). MLP portability is revisited here and compared to the proposed unsupervised MLP training approach.

The experiments were performed using the European Portuguese ASR system. Two bottleneck MLPs, trained for English and French broadcast data, were used for comparison. Other than the output layer, which depends on the phone states in each language, they have the same structure as the Portuguese MLP: 4-layers with dimensions $475 \times 3500 \times 39 \times O$, with $O$ being the number of HMM target states.

The two cross-lingual MLP networks were trained in a supervised manner using roughly the same volume of audio data. The English MLP was trained using 645 hours of speech data, and the French MLP with 600 hours (Le et al., 2010). Both networks were used to produce features for the 72 hour trainQ10 data set, that were used to build acoustic models with MLPPLPF0 features trained on an unsupervised fashion. These two models were used to decode the development data devQ10. The WERs obtained are shown in the second column of Table 5.4 (2nd and 3rd entry). For comparison purposes, the performance of the HMM trained with features extracted from the MLP trained on all of the Portuguese untranscribed data is shown in the first row. The HMM trained with features produced with the English MLP obtains similar WER levels as compared to that trained with using the unsupervised Portuguese MLP (28.7% and 28.6% respectively). A small gain (1.0% relative) is obtained with the features generated with the French MLP (28.3%).

The French MLP was also used to produce features for the 71 hour trainQ11 data set. These features were used to estimate a model on the combined trainQ10 + trainQ11

| MLP | Acoustic Models | |
|---|---|---|
| | `trainQ10` (72h) | `trainQ10` + `trainQ11` (143h) |
| `trainRVE` + `trainQ10` + `trainQ11` (316h) | 28.6 | 27.8 |
| English (645h) | 28.7 | - |
| French (600h) | 28.3 | **27.3** |

Table 5.4.: Comparison of an MLP trained in an unsupervised manner and cross-lingual MLPs. The acoustic models were trained in an unsupervised way using the decoding lattices of `trainQ10` or `trainQ10`+`trainQ11`. The WER(%) is measured on the development set `devQ10`. `LM_10src` was used for decoding.

data set. The recognition results are shown in the third column of Table 5.4. This new model outperforms the one trained with unsupervised Portuguese MLP features (first row) by 1.8% relative (27.3% and 27.8%, respectively).

### Retraining MLPs

The use of cross-lingual MLPs is quite efficient in order to improve the recognition performance compared to PLP models, even when little (or no) manually transcribed data are available. Intuitively, one can expect that the use of an MLP tuned for the target language should outperform the cross-lingual MLPs trained on the same amount of data (Stolcke et al., 2006). Le et al. (2010), for instance, showed that adapting bottleneck MLPs to a specific target condition (channel and dialect) outperforms the use of a generalized (channel and dialect independent) MLP. Adaptation was done by retraining the MLP with a subset of the data representing the target condition, without changing the MLP structure. A similar multi-language adaptation technique was described in Vu et al. (2012). In this latter work, a language independent MLP was trained on a large amount of data with certain phones as output targets. The MLP was then adapted to a new language via retraining. The output layer is replaced in order to represent the new phone targets.

Following these approaches, the French MLP was retrained using all of the Portuguese untranscribed audio data, replacing the output layer by the context independent states of the Portuguese phone inventory. A single training epoch with all the data was performed. For comparison, the unsupervised Portuguese MLP was also retrained on all data. Both models were used to generate features for the `trainQ10`+`trainQ11` data, from which, unsupervised acoustic models were estimated. These models were used to decode the development set `devQ10`, as well as the evaluation sets `testQ10` and `testQ11`. Table 5.5 shows the recognition performances obtained on each of the individual test sets, as well as the global WER.

Retraining both the MLPs leads to improvements in WER. However, the absolute gain is slightly smaller with the French MLP compared to the Portuguese one (0.1% and 0.3% respectively). This is probably due to the change in structure that the French MLP was submitted for retraining.

It is worth noting that the system using a retrained Portuguese MLP with all data (316 hours) and an unsupervised HMM trained on 143 hours (`trainQ10`+`trainQ11`) leads to a WER of 27.4% on `devQ10`. This is similar to the WER obtained with a system having

| MLP | devQ10 | testQ10 | testQ11 | ALL |
|---|---|---|---|---|
| trainRVE + trainQ10 + trainQ11 (316h) | 27.8 | 22.7 | 28.4 | 26.3 |
| French (600h) | 27.3 | 22.6 | 28.1 | 26.0 |
| trainRVE + trainQ10 + trainQ11 (316h) – retrained | 27.4 | 22.6 | 28.0 | 26.0 |
| French (600h) – retrained | 27.2 | 22.6 | 27.8 | 25.9 |

Table 5.5.: Comparison of an MLP trained in an unsupervised manner and a cross-lingual MLP, both retrained on the 316 hours of untranscribed Portuguese data. The acoustic models were trained in an unsupervised way using the decoding lattices of trainQ10+trainQ11. The WER(%) is measured on the development set devQ10 and test sets testQ10 and testQ11. 'ALL' refers to the WER obtained on the combined dev+test sets. LM_10src was used for decoding.

HMM and MLP parameters estimated in a supervised manner using the 72 hour trainQ10 data set (27.3%). This tie has been achieved by adding more data, but with no human effort spent on manual annotation.

## 5.4. Summary

In this chapter, unsupervised MLP training was investigated. After briefly describing MLPs, we focused on the use of bottleneck MLP for extracting features for acoustic modeling. An extensive empirical comparison was performed using such discriminatively trained features. In particular, it was shown that the combined use of MLP features with traditional PLP-like features can substantially reduce the recognition errors of large vocabulary ASR systems even when no manual transcriptions are available.

Supervised and unsupervised training approaches of both, MLP and HMM components, were compared using the same training data set. It was shown that unsupervised MLPs can be used to produce acoustic features that are only slightly worse than the features produced by a supervised MLP. Unlike unsupervised HMM training, no advantage from confidence based data filtering was observed during unsupervised MLP training.

When trained on 72 hours of audio data, the fully supervised HMM with MLP features outperforms its unsupervised counterpart by about 8.7% relative (27.3% vs. 29.9%). The use of 316 hours of data for unsupervised MLP and HMM training led to equivalent performance levels as the fully supervised system trained on 72 hours (WER = 27.4%).

Unsupervised MLP training was compared to cross-lingual MLPs. As reported by previous groups, the use of cross-lingual MLPs is efficient to improve the recognition performance, even when little (or no) manually transcribed data is available for the target language. Unsupervised MLP training, seems to be a good alternative to cross-lingual MLP porting. In our case, an unsupervised Portuguese MLP trained on 316 hours performs as good as a supervised English MLP trained on 645 hours, and slightly worse than a supervised French MLP trained on 600 hours of data.

We have limited the scope of this work to the use of bottleneck MLPs. However, the use of unsupervised training approaches seems to be well suited for DNN based models as well as recently reported in Laurent et al. (2014). A comparison between supervised and unsupervised DNN training approaches, such as performed in this chapter, is considered for future work.

# Chapter 6

# Unsupervised language model training

## 6.1. Introduction

The estimation of language model parameters only requires text data. Usually, dozens of millions to billions of words are used to build $n$-gram based language models for large vocabulary speech recognition. Such amounts of data are readily available on the Web from a variety of sources, such as online newspapers, blogs, forums, tweets and so on. Nevertheless, not all of these sources match the target data in terms of topic or style.

A common approach applied for $n$-gram based language modeling is to build language models for each available source and interpolate these models in a later phase (DeMori and Federico, 1999). Usually, the interpolation coefficients are automatically estimated using an iterative algorithm that aims to maximize the likelihood of a small held-out data set that corresponds to the target task. Thus, it can be expected that higher coefficients are assigned to sources that are similar to the held-out data.

Given that audio transcriptions truly represent the spoken style, they are naturally a desirable source of text to be used in language modeling. Component models trained on manual transcriptions usually have high coefficients assigned during interpolation. However, creating such training resources is costly and time consuming. Like for acoustic modeling, one can envisage to use automatic transcriptions provided by a recognizer to reduce costs required for language modeling. Automatic transcriptions can be used to estimate a component LM which is interpolated *a posteriori* with the baseline model (Bacchiani and Roark, 2003; Novotney et al., 2009). Alternatively, the $n$-gram counts extracted from the automatic transcriptions can be merged to the $n$-gram counts extracted from the other training sources (news, blogs, etc.). Both interpolation and counts merging can be seen as particular cases of MAP adaptation (Bacchiani and Roark, 2003).

This chapter investigates the use of unsupervised training approaches for LM training. First, confidence measures are explored as a means to reduce the impact of the recognition errors. A filtering approach is applied by converting words with low confidence scores to a common "unknown" word. Alternatively, the $n$-gram counts are weighted by the confidence scores. In addition, inspired by the work performed in acoustic model training (Chapter 4), the use of multiple weighted hypotheses is investigated. These hypotheses are extracted from the decoding lattices or from the confusion networks.

Weighting the training data generates $n$-grams with fractional counts. However, the so-called KN discounting algorithm (Kneser and Ney, 1995), which was reported to out-

perform other known smoothing techniques (Chen and Goodman, 1996, 1999), relies only on integral counts. To take advantage of confidence based weighting methods, a variation of the KN smoothing algorithm that handles fractional counts is presented in this chapter.

As an alternative to backoff $n$-gram language models, neural network language models were explored within the unsupervised training framework. Such a model uses a continuous space word representation and has a suitable structure for generalization. These two characteristics, which are not present in standard backoff $n$-gram models, are expected to help on reducing the impact of recognition errors. The experiments were carried out using the structured output layer (SOUL) NNLM (Le et al., 2011).

The remainder of this chapter is organized as follows. In Section 6.2, unsupervised $n$-gram language model training is investigated, focusing on the use of confidence based filtering and weighting techniques. In particular, the use of KN smoothing with fractional counts is described in Section 6.2.4. In Section 6.3, the use of automatic transcriptions to adapt the parameters of an NNLM is investigated. A summary of the chapter is given in Section 6.4.

## 6.2. Unsupervised backoff $n$-gram language model training

A brief overview of $n$-gram language modeling techniques was presented in Chapter 2. In this section, the focus is on the use of automatic transcriptions for LM training. Similar as used for unsupervised acoustic model training, the principle is illustrated in Figure 6.1. In brief, an initial system is used to decode a large amount of untranscribed data. The automatically generated transcriptions are used as ground truth to estimate the language model parameters. If necessary, the process can be re-iterated until some convergence criterion is achieved. In order to reduce the influence of recognition errors, confidence based filtering or weighting techniques can be applied.

### 6.2.1. Filtering by confidence measures

Filtering data for language model estimation is not trivial. The simple removal of a word changes multiple $n$-grams, affecting the model probabilities for different contexts. For the same reason, it is hard to deal with insertion and deletion errors.

Filtering can also be applied at sentence level. Although straightforward, this solution does not cope well with local transcription errors since the decision for filtering is based on the scores of all the words in the sentence. Alternatively, $n$-gram with low probabilities can be removed as performed, for instance, in Novotney et al. (2009).

Another word level filtering approach is proposed here. Instead of being removed, words with low confidence scores are transformed into an "unknown" word (<UNK>). As an example, the transcript:

```
... for[1.0] real[0.5] action[0.4] next[0.6] year[1.0] ...
```

where the number within brackets denotes the word confidence score, leads to the following bi-gram counts if a confidence threshold of 0.55 is applied:

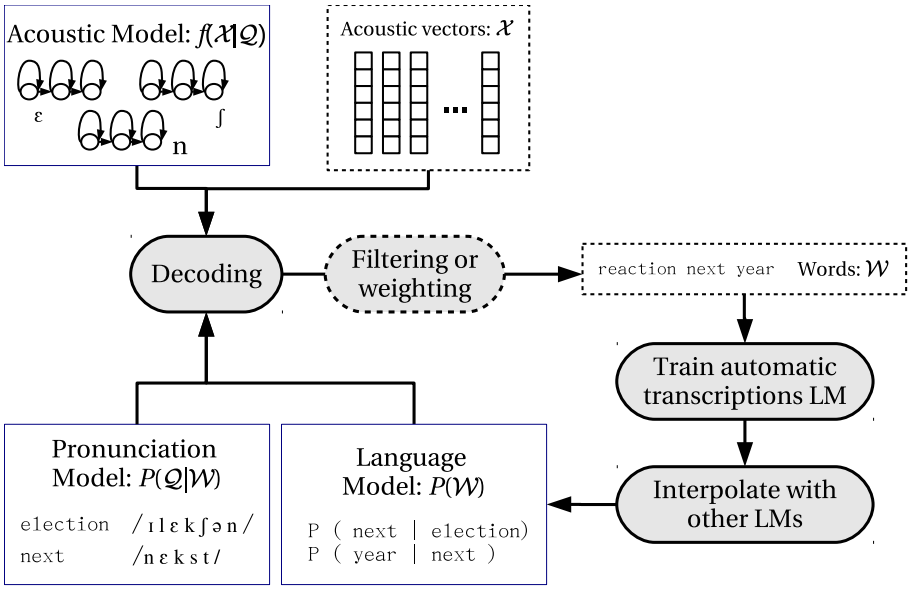



Figure 6.1.: Unsupervised language model training scheme.

```
1 for <UNK>
1 <UNK> <UNK>
1 <UNK> next
1 next year
```

The language models are obtained from these counts using the interpolated version of the KN smoothing (Ney et al., 1997).

### 6.2.2. Weighting by confidence measures

Weighting data for language modeling was performed in a similar way as proposed in Novotney et al. (2009). The (fractional) counts of a particular $n$-gram are obtained as the product of the confidence scores of its compounding words. The use of the average and maximum functions has been assessed as alternatives, but the product led to the best performance levels. If this weighting scheme is applied, the transcript:

```
... for[1.0] real[0.5] action[0.4] next[0.6] year[1.0] ...
```

leads to the following bi-gram counts:

```
0.50 for real
0.20 real action
0.24 action next
0.60 next year
```

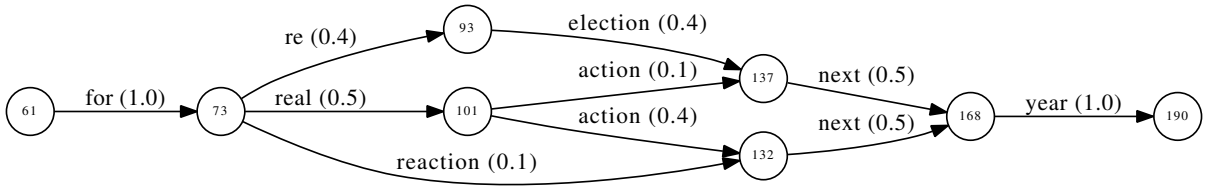
The language models are obtained from these fractional counts using an extended version of the KN smoothing proposed in Section 6.2.4.

### 6.2.3. Using multiple decoding hypotheses

Inspired by the work performed in unsupervised acoustic model training, the use of multiple decoding hypotheses was explored for language modeling. The language models are estimated using the KN smoothing variant proposed in Section 6.2.4, but with $n$-gram count lists obtained from decoding lattices or confusion networks.

#### Getting $n$-gram counts from lattices

The lattice-based $n$-gram counts can be obtained by performing the forward-backward algorithm (Baum et al., 1970), taking into consideration the scores of the past $(n-1)$ nodes in the lattice leading to a particular node. As an example, from the word lattice:

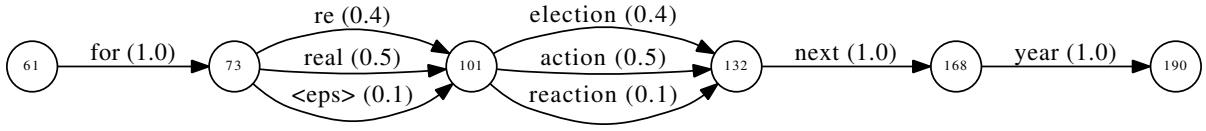



it is possible to extract the following bi-gram counts:

```
0.50 for real
0.40 for re
0.10 for reaction
0.40 re election
0.50 real action
...
```

#### Getting $n$-gram counts from confusion networks

Similar to the 1-best weighting case, confusion network $n$-gram counts are calculated as the product of confidence scores of their compounding words. The difference is that all paths in the confusion network having length $n$ are exploited. Empty transition links (<eps>) are skipped, but their confidence scores are used for the calculation of the $n$-gram counts. A pruning strategy is used to limit the number of $n$-grams extracted. As an example, from the confusion network:




it is possible to extract the following bi-gram counts:

```
0.50 for real
0.40 for re
0.05 for action
0.04 for election
0.01 for reaction
0.20 re action
0.16 re election
...
```

This method was also combined with the filtering technique by transforming low probability words into the unknown word. From the confusion network example shown above and using a confidence threshold of 0.45, the following bi-grams can be extracted:

```
0.50 for real
0.40 for <UNK>
0.05 for action
0.05 for <UNK>
0.20 <UNK> action
0.16 <UNK> <UNK>
...
```

### 6.2.4. Kneser-Ney smoothing with fractional counts

The Kneser-Ney smoothing (Kneser and Ney, 1995) is one of the algorithms the most applied for language modeling. It can be represented as follows. Let $(a, g, w)$ be a sequence of words with length $n$, where $a$ is a prefix word, $g$ is an arbitrary sequence of words with length $(n-2)$ and $w$ is the word to be predicted. The sequence $h = (a, g)$ is the history of $w$ with length $(n-1)$ and $g$ is the less specific history of $w$ with length $(n-2)$. Thus, the interpolated KN smoothed model (Ney et al., 1997) can be represented by:

$$p_{KN}(w|a,g) = \frac{\max\{C(a,g,w) - D, 0\}}{\sum_{w_i} C(a,g,w_i)} + \gamma(a,g) p_{KN}(w|g) \tag{6.1}$$

where $C(s)$ denotes the counts of event $s$, $D$ is the discounting parameter and $\gamma(a,g)$ is the interpolation coefficient with the lower order model $p_{KN}(w|g)$. The coefficients $\gamma(a,g)$ and the lower order probability densities $p_{KN}(w|g)$ are obtained constraining the model to:

$$\begin{aligned} C(g,w) &= \sum_{a_j} C(a_j,g,w) = \sum_{a_j} C(a_j,g) C(w|a_j,g) \\ &= \sum_{a_j} C(a_j,g) p_{KN}(w|a_j,g) = \sum_{a_j} \left[ \sum_{w_i} C(a_j,g,w_i) \right] p_{KN}(w|a_j,g) \end{aligned} \tag{6.2}$$

The standard KN discounting algorithm (see Section 2.3.2) considers that the $n$-gram frequencies are represented as integral counts. An extended version of the KN smoothing that takes into account fractional $n$-gram counts is obtained as follows. The coefficients $\gamma(a,g)$ are obtained by applying the constraint $\sum_{w_i} p_{KN}(w_i|\cdot) = 1$ to (6.1):

$$\begin{aligned} 1 &= \frac{\sum_{w_i} \max\{C(a,g,w_i) - D, 0\}}{\sum_{w_i} C(a,g,w_i)} + \gamma(a,g) \\ \implies \gamma(a,g) &= \frac{\sum_{w_i} \left( C(a,g,w_i) - \max\{C(a,g,w_i) - D, 0\} \right)}{\sum_{w_i} C(a,g,w_i)} \\ \gamma(a,g) &= \frac{\sum_{w_i} \min\{C(a,g,w_i), D\}}{\sum_{w_i} C(a,g,w_i)} \end{aligned} \tag{6.3}$$

The lower order models $p_{KN}(w|g)$ can be obtained by substituting (6.1) and (6.3) in (6.2):

$$C(g,w) = \sum_{a_j} \max\{C(a_j,g,w) - D, 0\} + \sum_{a_j} C(a_j,g)\gamma(a_j,g)p_{KN}(w|g)$$

$$\implies p_{KN}(w|g) = \frac{C(g,w) - \sum_{a_j} \max\{C(a_j,g,w) - D, 0\}}{\sum_{a_j} C(a_j,g)\gamma(a_j,g)}$$

$$p_{KN}(w|g) = \frac{\sum_{a_j} \min\{C(a_j,g,w), D\}}{\sum_{a_j}\sum_{w_i} \min\{C(a_j,g,w_i), D\}} \tag{6.4}$$

Finally, (6.3) and (6.4) can be rewritten as:

$$\gamma(a,g) = \frac{C_<(a,g,\bullet) + DN_>(a,g,\bullet)}{\sum_{w_i} C(a,g,w_i)} \tag{6.5}$$

$$p_{KN}(w|g) = \frac{C_<(\bullet,g,w) + DN_>(\bullet,g,w)}{C_<(\bullet,g,\bullet) + DN_>(\bullet,g,\bullet)} \tag{6.6}$$

Where the functions $C_<(\cdot)$ and $N_>(\cdot)$ are defined as:

$$C_<(a,g,\bullet) = \sum_{w_i : C(a,g,w_i) < D} C(a,g,w_i) \tag{6.7}$$

$$C_<(\bullet,g,w) = \sum_{a_j : C(a_j,g,w) < D} C(a_j,g,w) \tag{6.8}$$

$$C_<(\bullet,g,\bullet) = \sum_{(a_j,w_i) : C(a_j,g,w_i) < D} C(a_j,g,w_i) \tag{6.9}$$

$$N_>(a,g,\bullet) = |w_i : C(a,g,w_i) > D| \tag{6.10}$$

$$N_>(\bullet,g,w) = |a_j : C(a_j,g,w) > D| \tag{6.11}$$

$$N_>(\bullet,g,\bullet) = |(a_j,w_i) : C(a_j,g,w_i) > D| \tag{6.12}$$

Equations 6.3 and 6.4 are similar to those of the interpolated KN model (c.f. (2.36) and (2.37)), except for the presence of the $C_<(\cdot)$ functions and the use of $N_>(\cdot)$ in the place of $N_{1+}(\cdot)$. With integral counts, and assuming $D < 1$, the summations $C_<(\cdot)$ equal zero and $N_>(\cdot)$ equals $N_{1+}(\cdot)$, leading to the original KN equations.

### Calculating the discounting parameter

Another important issue when working with fractional counts is the calculation of the discounting parameter $D$. With integral counts, discounting is obtained by:

$$D = \frac{n_1}{n_1 + 2 \cdot n_2} \tag{6.13}$$

where $n_1$ and $n_2$ denote, respectively, the number of $n$-grams with exactly one or two counts. With fractional counts, though, these definitions are unclear.

Three alternatives are proposed to calculate the discounting parameters. In the first one, (6.13) is used with $n_1$ and $n_2$ values obtained after approximating fractional counts to integer counts. Alternatively, $D$ can be calculated by:

$$D = \frac{\sum_{(a,g,w):C(a,g,w)\leq 1} C(a,g,w)}{\sum_{(a,g,w):C(a,g,w)\leq 2} C(a,g,w)} \tag{6.14}$$

Equation 6.14 is proposed as a simple extension of (6.13), without theoretical justification. In particular, (6.13) is recovered if only integral counts are admitted. As a third option, the discounting parameters can be obtained as a result of an optimization algorithm. In this work, the Powel's method (Press et al., 2007, Section 10.7), such as used by Bisani and Ney (2008), was used in order to minimize the language model perplexity of a development set.

Equivalent levels of perplexity of the development data set were obtained with the three methods. However, the optimization algorithm was used in the remainder of this chapter.

### 6.2.5. Experiments

The experiments were carried out using the *improved* European Portuguese recognition system described in Section 3.2.4. This system was used to decode the training sets `trainRVE`, `trainQ10` and `trainQ11`, containing, respectively, 173, 72 and 71 hours of speech data (see Table 3.1). The 1-best transcriptions that were generated contain, respectively, 1642, 630 and 635 thousand words. From each training set, 2-, 3- and 4-gram component LMs were estimated with $n$-gram counts extracted from, either, 1-best, 1-best weighted or 1-best filtered transcriptions or decoding lattices, confusion networks or filtered confusion networks. The 1-best and 1-best filtered component models were estimated using the standard interpolated KN smoothing with a single discounting parameter per $n$-gram level (Kneser et al., 1997). For the remaining methods, the KN smoothing variant described in Section 6.2.4 was used.

For 1-best filtered and confusion network filtered counts, a confidence threshold of 0.1 was used. Other thresholds around 0.05 and 0.5 were assessed during an optimization procedure, but the results did not change significantly. The scaling edge factor $\beta$ (c.f. (4.27)) was set to 0.8 after tuning.

The component language models estimated on the decoding hypotheses were interpolated with the baseline language model `LM_10src` (see Section 3.2.4), generating *adapted* language models. The interpolation coefficients were optimized to minimize the perplexity of the `devQ10` set.

The perplexities of `devQ10` with the 4-gram component models estimated using the assessed methods are shown in Table 6.1. The perplexity obtained after interpolating the three models (`trainRVE`, `trainQ10` and `trainQ11`) and after interpolating them with the baseline model are also shown. In the best case, with counts extracted from the confusion networks, the perplexity is reduced by 5% relative to the baseline model (from 138 to 131).

The interpolated language models were assessed for a broadcast data recognition task. Each model was used to decode the sets `devQ10`, `testQ10` and `testQ11` using two different acoustic models. The baseline AM was estimated on the 173 hour `trainRVE` data set using an unsupervised acoustic modeling approach. The second acoustic model used was obtained via unsupervised MAP adaptation of the baseline model to the 72 hour `trainQ10` untranscribed data set. The recognition results are summarized in Table 6.2.

| COUNTS | trainRVE (173h) | trainQ10 (72h) | trainQ11 (71h) | 3 MODELS (316h) | + LM_10SRC |
|---|---|---|---|---|---|
| 1-best | 321 | 300 | 300 | 219 | 132 |
| 1-best weighted | 336 | 321 | 325 | 234 | 132 |
| 1-best filtered | 329 | 327 | 333 | 244 | 134 |
| lattice | 321 | 308 | 308 | 236 | 133 |
| confusion network | 308 | 295 | 297 | 220 | 131 |
| confusion network filtered | 375 | 388 | 394 | 284 | 137 |
| manual | - | 270 | - | - | 127 |

Table 6.1.: Perplexity of devQ10 with 4-gram language models estimated on the decoding hypotheses of trainRVE, trainQ10 and trainQ11. The perplexity obtained by interpolating these "3 models" and the perplexity obtained by interpolating them with the baseline model "LM_10src" are given. The performance obtained with the use of a component model trained with manual transcriptions is reported for reference. For reference, the perplexity with the baseline model alone is 138. The numbers within parentheses correspond to the amount of audio data of each training set.

With the baseline AM (Table 6.2a), a small difference in performance is observed with the addition of automatic transcriptions to the LM training corpus, with $p<0.01$ according to the MAPSSWE test (Pallett et al., 1990). No significant difference in performance on testQ10 was observed with any of the methods assessed. Absolute improvements around 0.2% were achieved on devQ10 and testQ11. The best LM (1-best filtered) leads to an overall WER reduction of 0.19% with respect to the baseline model (from 31.16% to 30.97%). For reference, a component LM trained on the trainQ10[1] manual transcriptions was assessed. In this case, an overall WER of 30.62% was achieved, that is, an absolute gain of 0.54% compared to the baseline.

The second set of experiments was performed to assess the case where language and acoustic models are jointly adapted using automatic transcriptions obtained from the same training iteration. With the adapted AM (Table 6.2b) the 1-best weighting and 1-best filtering techniques obtained the best recognition performances for all test sets. According to the MAPSSWE test, these two language models are equivalent and both better than the other models assessed. In particular, the 1-best weighting technique led to an absolute WER reduction of 0.25% compared to the baseline (from 29.47% to 29.22%). For reference, the case that uses a component LM trained with the trainQ10 manual transcriptions has a WER of 29.02%, that is an absolute reduction of 0.45% compared to the baseline.

Language model training with multiple decoding hypotheses does not improve the performance compared to training with 1-best transcriptions. Similar results were reported by Novotney et al. (2009). This might be due to fact that a large number of "incorrect" $n$-grams is extracted from the decoding lattices or the confusion networks, overwhelming the influence of the "correct" $n$-grams.

[1]The only source for which manual transcriptions were available.

| Language model | devQ10 | testQ10 | testQ11 | testQ10 + testQ11 | Overall |
|---|---|---|---|---|---|
| LM_10src (baseline) | 33.58 | 26.43 | 33.50 | 30.02 | 31.16 |
| + 1-best | 33.36 | 26.48 | 33.31 | 29.94* | 31.04* |
| + 1-best weighted | 33.32 | 26.48 | **33.28** | 29.93* | 31.02* |
| + 1-best filtered | **33.24** | **26.39** | **33.29** | **29.89*** | **30.97*** |
| + lattice | 33.37 | 26.45 | 33.45 | 30.00* | 31.08 |
| + confusion network | 33.30 | 26.53 | 33.32 | 29.97* | 31.04* |
| + confusion network filtered | 33.41 | **26.39** | **33.29** | **29.89*** | 31.02* |
| + manual (trainQ10) | 32.94 | 26.00 | 32.95 | 29.52 | 30.62 |

(a) With the baseline acoustic model.

| Language model | devQ10 | testQ10 | testQ11 | testQ10 + testQ11 | Overall |
|---|---|---|---|---|---|
| LM_10src (baseline) | 31.36 | 25.77 | 31.29 | 28.57 | 29.47 |
| + 1-best | 31.06 | 25.85 | 31.25 | 28.59 | 29.38 |
| + 1-best weighted | **30.86** | **25.68** | **31.12** | **28.44*** | **29.22*** |
| + 1-best filtered | **30.87** | **25.69** | 31.26 | 28.51* | 29.27* |
| + lattice | 30.98 | 25.90 | 31.23 | 28.60 | 29.37 |
| + confusion network | 30.97 | 25.89 | 31.22 | 28.59 | 29.36 |
| + confusion network filtered | 31.17 | 25.71 | 31.19 | 28.49* | 29.35 |
| + manual (trainQ10) | 30.68 | 25.48 | 30.92 | 28.24 | 29.02 |

(b) With the adapted acoustic model.

Table 6.2.: Recognition results with component language models trained in an unsupervised manner and interpolated with the baseline language model LM_10src. The performance obtained with the use of a component model trained with manual transcriptions is reported for reference. Stars (*) denote equivalent systems according to the matched pairs sentence-segment word error significance test ($p<0.01$).

## 6.3. Unsupervised neural network language model adaptation

In principle, the lack of a suitable structure for generalization and the use of discrete modeling units make backoff $n$-gram language modeling quite sensitive to wrongly assigned classification labels. Different from backoff models, neural network language models (Bengio et al., 2000; Schwenk and Gauvain, 2002; Schwenk, 2007) use a continuous space word representation and have a suitable structure for generalization. These two characteristics could help to reduce the impact of the recognition errors during unsupervised LM parameter estimation.

An introduction to NNLMs was presented in Section 2.3.3 with focus on *feed-forward* network models (Bengio et al., 2000, 2003). The unsupervised language model training experiments outlined in this section were carried out using a variation of the feed-forward NNLM, the structured output layer (SOUL) NNLM (Le et al., 2011, 2013), which is briefly described below.

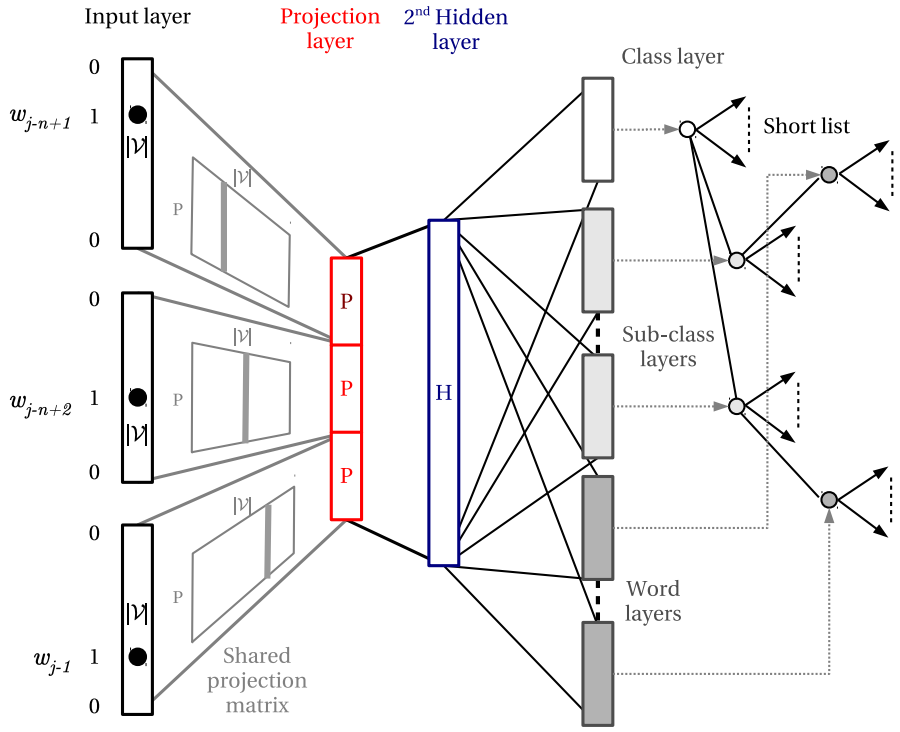



Figure 6.2.: Structured output neural network language model architecture.

### 6.3.1. Structured output layer neural network language model

The main difference of common NNLMs and the structured output layer NNLM (Le et al., 2011, 2013) is (as the name suggests) the output layer. The SOUL model is able to handle output vocabularies with arbitrary sizes (rather than a short list of words) with only a small increase of computational calculation time. This is accomplished by organizing the output layer as a tree-like structure, built based on a word class (and sub-classes) clustering algorithm. The first level of the tree gives softmax posterior class probabilities. From the second to the second to last levels, sub-class probabilities are calculated. The last level of the tree is used to estimate the word posterior probabilities. The probabilities of the most frequent words (the short list) are directly estimated at the first tree level. In other words, words of the short list are considered to represent their own classes (without sub-classes).

Figure 6.2 shows the architecture of the SOUL NNLM used. It consists of a 4-layer feed-forward network model. The input is formed by a vector that represents three history words (that is a 4-gram LM). Each word is projected into a 500-dimensional shared continuous space. The second hidden layer contains 1000 nodes with sigmoidal activation functions. The softmax output layer is structured into a 3-level tree and contains a two thousand word short list.

### 6.3.2. Experiments

The experiments were carried out using the *improved* European Portuguese recognition system, which was used to generate 1-best automatic transcriptions for the training sets `trainRVE`, `trainQ10` and `trainQ11`.

The neural network language models were obtained as follows. First, a baseline NNLM was estimated with the same data used to train the backoff $n$-gram model `LM_10src`: nine written sources and a small amount of transcribed data. The training procedure adopted is similar to the

| NNLM | Baseline | → Unsupervised `trainQ10`, `trainQ11` | → Unsupervised `trainRVE`, `trainQ10`, `trainQ11` | → Supervised `trainQ10` |
|---|---|---|---|---|
| PERPLEXITY | 123 | 114 | 114 | 106 |

Table 6.3.: Perplexity of `devQ10` obtained with neural network language models interpolated with the 4-gram model `LM_10src`. 'Baseline' refers to the NNLM trained only on written text sources. The remaining NNLMs were obtained by adapting the baseline model to the transcriptions of the reported training sets using unsupervised or supervised training. For reference, the perplexity with the baseline backoff $n$-gram model is 138.

one described in Le et al. (2011). A word clustering algorithm is used to build the output layer tree-like structure, which is fixed for the remaining training steps. Ten iterations of stochastic back-propagation training (like in Bengio et al. (2003)) are performed with different learning rates. For each iteration, the training data is randomly sampled to about 25 million words. The development set `devQ10` was used for validation.

Then, the baseline NNLM was adapted to the 1-best transcriptions of `trainQ10`+`trainQ11` and `trainRVE`+`trainQ10`+`trainQ11`. Adaptation was performed by retraining the initial model using the automatic transcriptions as target labels. Five back-propagation iterations were performed. During retraining, only the parameters of the hidden and output layers are updated, but the projection matrix is fixed. Other retraining strategies have been tried (increasing the number of iterations, updating the whole model, filtering the transcriptions by confidence measures), but the described setup led to the best perplexity and recognition performance levels on the development set `devQ10`. For reference, an NNLM was obtained by retraining the baseline model with the manual transcriptions of `trainQ10`.

The adapted NNLM and the baseline NNLM were each interpolated with the backoff 4-gram language model (`LM_10src`) with coefficients estimated to minimize the perplexity of `devQ10`. These perplexities are reported in Table 6.3 for reference. The use of the baseline NNLM aline leads to a 11% relative improvement compared to the `LM_10src` (from 138 to 123). On top of that, adapted NNLMs obtain relative improvements of 7% or 14% for unsupervised (from 123 to 114) and supervised (from 123 to 106) adaptation respectively.

The interpolated models (NNLM+`LM_10src`) were assessed in a recognition task. Decoding lattices for the sets sets `devQ10`, `testQ10` and `testQ11` were generated using the 2- and 3-gram `LM_10src` models. These lattices were rescored in a latter step with each of the interpolated models described above. Two acoustic models were used for decoding, one trained only on `trainRVE` and another obtained by adapting this latter to `trainQ10`. In both cases, the automatic transcriptions of the training data were used to guide AM parameter estimation. The recognition results are summarized in Table 6.4.

Compared to the backoff $n$-gram models (see Table 6.2), the use of a 4-gram SOUL NNLM led to significant increase of recognition accuracy. The baseline SOUL NNLM outperforms the baseline backoff LM on about 1.19% absolute (WER from 31.16% to 29.97%) with the baseline AM, and on about 0.78% with the adapted AM (from 29.47% to 28.69%).

With the baseline AM (Table 6.4a), unsupervised NNLM adaptation did not lead to improvements over the baseline NNLM. When the SOUL NNLM is adapted to the manual transcriptions of `trainQ10`, an absolute gain of 0.32% is obtained (from 29.97% to 29.65%). It is worth noting that this improvement is due only to the neural network model. An additional absolute gain of around 0.4–0.5% can be expected by including a component backoff model trained on the `trainQ10` manual transcriptions.

| Neural network language model | devQ10 | testQ10 | testQ11 | testQ10+ testQ11 | All |
|---|---|---|---|---|---|
| LM_10src (baseline) | 32.41 | **25.38** | **32.15** | **28.81** | 29.97 |
| → *Unsupervised*: | | | | | |
| trainQ10+trainQ11 | **32.24** | 25.45 | 32.15 | 28.85 | **29.94** |
| trainRVE+trainQ10+trainQ11 | 32.26 | 25.56 | 32.24 | 28.95 | 30.01 |
| → *Supervised*: trainQ10 | 31.97 | 25.27 | 31.75 | 28.56 | 29.65 |

(a) With the baseline acoustic model.

| Neural network language model | devQ10 | testQ10 | testQ11 | testQ10+ testQ11 | All |
|---|---|---|---|---|---|
| LM_10src (baseline) | 30.68 | 25.21 | 30.22 | 27.75 | 28.69 |
| → *Unsupervised*: | | | | | |
| trainQ10+trainQ11 | 30.35 | **25.05** | **30.19** | **27.66** | **28.52** |
| trainRVE+trainQ10+trainQ11 | **30.29** | 25.14 | 30.25 | 27.73 | 28.55 |
| → *Supervised*: trainQ10 | 29.89 | 24.82 | 29.88 | 27.39 | 28.19 |

(b) With the adapted acoustic model.

Table 6.4.: Recognition results obtained with neural network language models interpolated to the 4-gram backoff model LM_10src. 'Baseline' refers to the NNLM trained only on written text sources. The remaining NNLMs were obtained by adapting the baseline model to the transcriptions of the reported training sets using unsupervised or supervised training.

With the adapted AM (Table 6.4b), the unsupervised adapted NNLM led to small gains of recognition performance over the baseline NNLM. At the best (when the NNLM is adapted to trainQ10+trainQ11), an absolute overall WER improvement of 0.17% is obtained in comparison with the baseline NNLM (from 28.69% to 28.52%). Although some improvement was achieved on devQ10 (0.33% absolute) and testQ10 (0.16% absolute), this was not the case for testQ11. According to the MAPSSWE test, only the improvements obtained over the joint dev-eval set is significant ($p<0.005$). For reference, when the NNLM is adapted to the manual transcriptions of trainQ10, an absolute improvement of 0.50% is obtained in comparison to the baseline (from 28.69% to 28.19%).

An additional experiment was performed to assess if the use of unsupervised backoff $n$-gram language model training would improve the recognition accuracy on top of the unsupervised adapted SOUL NNLMs. In other words, if the gains obtained with the use of untranscribed data during the estimation of both models were complementary.

No recognition accuracy improvement was observed on devQ10, testQ10 and testQ11. Therefore, the best performance levels obtained with the unsupervised language modeling approaches explored here are due to the NNLM adaptation, which led to an overall WER of 28.52% (with the adapted AM).

## 6.4. Summary

In this chapter, unsupervised language model training was applied to a European Portuguese broadcast recognition task. Two models were explored within the unsupervised training framework, the standard backoff $n$-gram language models and the structured output layer neural

network language models.

First component backoff $n$-gram LMs were estimated with automatic transcriptions generated by an existing ASR system and interpolated to a baseline language model (trained on written text sources). To deal with recognition errors, confidence based filtering and weighting techniques were applied on $n$-gram counts extracted from the 1-best decoding hypothesis and multiple hypotheses (lattices and confusion networks). In contrast with unsupervised acoustic modeling (Chapter 4), the use of multiple decoding hypotheses did not outperform the 1-best hypothesis approach. The best recognition results were obtained when the component models were estimated with 1-best filtered or 1-best weighted $n$-gram counts. At the best, they obtained an absolute improvement of 0.25% compared to the baseline (from 29.47% to 29.22%).

A variant of the Kneser-Ney smoothing algorithm that takes into account fractional counts was presented. The method was used to generate language models from the 1-best weighted hypothesis, lattices and confusion networks counts.

Automatic audio transcriptions were also used to adapt a baseline structured output layer NNLM (Le et al., 2011) trained on text sources. Adaptation was performed by retraining the baseline model with automatic transcriptions as target labels. At the best, unsupervised adaptation obtained an absolute improvement of 0.17% compared to the baseline (from 28.69% to 28.52%). When combined to unsupervised backoff $n$-gram language modeling, no additional improvement was achieved.

In summary, the approaches assessed in this chapter led to absolute WER gains around 0.2–0.3% with respect to the backoff $n$-gram and NNLM baselines. These results corroborates with previously reported work (Bacchiani and Roark, 2003; Novotney et al., 2009): unsupervised language modeling is a much more challenging task than unsupervised acoustic modeling. However, it is worth noting that unsupervised LM leads to gains of recognition performance on top of unsupervised acoustic modeling. That being said, if some data have been automatically transcribed for unsupervised AM training, the additional cost of using the transcripts for LM training is negligible. Thus, using unsupervised LM might be worthwhile.

The approaches presented here were applied for broadcast speech recognition. Higher improvements might be obtained for tasks having larger mismatches between the baseline model and the target data, such as conversational speech recognition (Bacchiani and Roark, 2003; Novotney et al., 2009), dialog or voice search.

# Part III.

# Model combination

# Model combination

The use of large amounts of data to estimate acoustic and language models for ASR is not a sufficient condition to lead to suitable recognition performance levels. The acoustic and text training data need to be similar to the target data. In other words, training and test data have to be drawn from the same probability density function. A problem thus arises in cases where target specific data are scarce as is the case for tasks involving conversational telephone speech recognition or low e-resourced languages and dialects.

The construction of representative training corpora for low e-resourced tasks is especially demanding. Besides the human effort required for manually transcribing the audio data, even collecting the data is costly and time-consuming. In some cases, costs due to fieldwork, rewarding subjects, acquiring legal authorizations, etc., also need to be considered.

Model adaptation techniques can be used to reduce the need for target specific data. Adaptation assumes the existence of two sources of knowledge: a well-trained baseline model that does not represent well the target data, and a small amount of adaptation data from which relevant information related to the target task can be extracted. The main principle is to adjust the parameters of the baseline model to approximate the target specific data, generating an adapted model.

Various acoustic and language model adaptation techniques exist. MAP adaptation, for instance, can be applied to acoustic (Gauvain and Lee, 1994) and language (Bacchiani and Roark, 2003) models. MAP acoustic model adaption uses a hyper-parameter to control the degree of relevance between the baseline model and the target-specific data. In the maximum likelihood linear regression acoustic model adaptation approaches (Leggetter and Woodland, 1995), the adapted model is represented as a linear transformation of the baseline model.

Adaptation methods assume that data used on the estimation of the baseline model are homogeneous, drawn from the same distribution function. However, this assumption is not strictly true, since the training data is usually collected from a wide variety of sources. A proposed approach that can take into account the data heterogeneity and is commonly used for language model training is model interpolation. In this approach, a set of component models are independently estimated and interpolated *a posteriori* using automatically estimated coefficients. Interpolation allows to adjust the relevance of the different training subsets with respect to the target specific data.

This work investigates the use of interpolation to build acoustic models that match the target data well even if only a small (or no) amount of target data is available for training. We argue that interpolation is a faster and more flexible alternative than acoustic model adaptation techniques. This part is organized in a different manner compared to Part II due to the different experimental setups assessed. Thus, **Chapter 7** first describes the theoretical framework for acoustic model interpolation and our proposed methods. The experiments performed and results

obtained are presented in **Chapter 8**.

Some of the work described in this part of this dissertation was originally published in Interspeech 2013 (Fraga-Silva et al., 2013) and Eusipco 2014 (Fraga-Silva et al., 2014).

# Chapter 7

# Acoustic model interpolation

## 7.1. Introduction

In LVCSR tasks, the acoustic realization is usually modeled by hidden Markov models. The parameters of such models are often estimated by maximizing the likelihood of the training data (Juang, 1985). This estimation method assumes two hypotheses. First, the method relies on the use of a large amount of data to generate robust estimates. Second, the ML estimation assumes that the training and the test sets are drawn from the same distribution function. In other words, the acoustic model will be applied to recognize some data that are similar to the training data set.

In the context of low cost ASR system development, the first issue was investigated in Chapter 4. Unsupervised training was used as a means to produce approximate corpus for acoustic modeling. This chapter investigates the second issue.

Acoustic models for speech recognition are often trained on data coming from a variety of sources. The most common approach used for acoustic modeling is to pool together all of the available data, considering them all part of a unique training set. This method leads to good accuracy if the two aforementioned assumptions hold. However, gathering a fair amount of acoustic training data that matches well the target can be difficult for a variety of tasks, like conversational telephone speech recognition (Gauvain et al., 2003) or non-native speech recognition (Fischer et al., 2001; Wang et al., 2003).

This problem is often treated using an adaptation method (Wang et al., 2003; Oh et al., 2007; Vergyri et al., 2010), such as MLLR (Leggetter and Woodland, 1995) or MAP (Gauvain and Lee, 1994; Zavaliagkos et al., 1996) adaptation. In such cases, an existing (and more general) model trained on a large amount data is adapted to a small amount of data that matches better the target task.

Model adaptation is usually performed by adjusting the similarity levels between the baseline model, trained on out-of-domain data, and the in-domain adaptation data. This is clearer with MAP estimation, where a hyper-parameter is used to control which amount of in-domain frames should be used to modify the prior out-of-domain models. In MLLR schemes, adjustment is made by means of matrix transforms used to represent the in-domain data in the same subspace of the out-of-domain data.

These techniques still consider the out-of-domain data as a unique homogeneous training set, assigning the same importance to all samples. Nonetheless, it seems reasonable to consider that different subsets of that data could have different degrees of relevance for the target data. This assumption is considered for language modeling, which is usually performed as follows. First, component language models are trained on separate subsets of the training data and, in a latter

step, interpolated to generate a final model. The degree of relevance of each component model is assigned via interpolation coefficients estimated in order to maximize the likelihood of the target data.

In this work, we propose to apply the same principle for acoustic modeling, assigning mixture coefficients to component acoustic models trained on separate subsets. We show that these coefficients can be estimated using the EM algorithm (Dempster et al., 1977) aiming to maximize the likelihood of a held-out data set. As an alternative, the coefficients can be estimated on-the-fly for each test data set. Two combination methods, both based on model interpolation, are presented. In the first method, the GMMs of the component models are merged. In the second one, the training data are weighted during the AM parameter estimation.

Interpolation of acoustic models has been used in other tasks related to speech processing. For instance, for speech synthesis, it was used to combine models of different speaking styles (Tachibana et al., 2004) or models trained for different speakers (Yoshimura et al., 1997). For speech recognition, model interpolation was used to combine native and non-native acoustic models as an alternative to acoustic model adaptation (Wang et al., 2003; Tan and Besacier, 2007).

GMM interpolation as presented in this chapter is performed by merging the Gaussian mixtures of the component models and properly adjusting the mixing coefficients. The resulting model is a large GMM containing all the parameters of the component models. When several components are merged together, the computational complexity required for decoding may increase considerably. A GMM reduction algorithm is proposed in this chapter as a means to recover suitable computational complexity levels.

The remainder of this chapter is organized as follows. In Section 7.2, some common acoustic model adaptation techniques are briefly described. In Section 7.3, the two interpolation approaches (GMM interpolation and data weighting) are presented. The EM-based method used to estimate the interpolation coefficients is also shown. Section 7.4 described the GMM reduction algorithm proposed in this work. A summary is given in Section 7.5.

## 7.2. Acoustic model adaptation

Speech variability is an important issue in ASR, directly affecting the quality and challenging the robustness of the recognition systems. For the speech recognition task, variability can be seen as a noisy material disturbing the signal carrying the relevant linguistic information. This noise (term applied here in a large sense to qualify non-linguistic events) can be due to variations on gender, accent, age, speaking style, rate of speech, health and so on (Benzeghiba et al., 2007).

Compensating the speech variability is a major issue on ASR that has been well covered in the literature (for a review, refer to Benzeghiba et al. (2007)). Among other techniques, normalizing the cepstral means and variances is probably the most straightforward method to avoid undesirable (non-linguistic) events. Another well known solution is to normalize the vocal tract length among speakers through feature transforms in order to reduce the impact of physiological variations (Lee and Rose, 1996). In this work, we will focus on another type of compensation, the model adaptation.

Acoustic model adaptation is fundamental on state-of-the-art ASR systems. The basic principle is to adjust the parameters of a baseline speaker independent model to the targeted condition (genre, accent, speaker, channel, etc.) represented by the adaptation set. The most popular methods used for this purpose, maximum *a posteriori* (MAP) adaptation, maximum likelihood linear regression (MLLR) and constrained MLLR (CMLLR) are briefly summarized in the following. In this chapter, all the mathematical framework is developed uniquely for the

GMM parameter $\theta$, assuming that an HMM state/frame alignment is used. The extension to the estimation of the HMM parameters should be straightforward.

### 7.2.1. Maximum *a posteriori*

Maximum *a posteriori* estimation (Gauvain and Lee, 1994) was shortly introduced in Section 2.2.2 as a generative method for estimating the acoustic model parameters. As proposed in the original paper, the algorithm is well suited to be used as an adaptation technique. In this section, we are particularly interested on the adaptation of the GMM parameters. The MAP adapted GMM estimates can be obtained as the mode of the posterior probability $g(\theta|\mathcal{X})$,

$$\theta_{MAP} = \arg\max_{\theta} g(\theta|\mathcal{X}) = \arg\max_{\theta} f\left(\mathcal{X}|\theta\right) g(\theta) \tag{7.1}$$

where $g(\theta)$, provided by a baseline acoustic model, is a prior distribution over the GMM parameters $\theta$, $\mathcal{X} = (x_1, \ldots, x_T)$ is a set of adaptation feature vectors with p.d.f. $f(\mathcal{X}|\theta)$.

This problem can be efficiently solved using an iterative EM approach (Dempster et al., 1977). In particular, the Gaussian mean vector $\mu_k$ of the $k$-th component of the adapted model can be obtained as a weighted sum of the prior parameters and the observed adaptation vectors, as follows:

$$\mu_k = \frac{\tau_k m_k + \sum_{t=1}^{T} \gamma_{kt} x_t}{\tau_k + \sum_{t=1}^{T} \gamma_{kt}} \tag{7.2}$$

where $m_k$ is the mean vector of the $k$-th Gaussian component of the baseline model, while $\tau_k$ is a parameter controlling the degree of adaptation: as long as $\tau_k$ increases, more adaptation data is necessary to actually differentiate the adapted parameters from the prior ones. Finally, $\gamma_{kt} = P(k|x_t, \hat{\theta})$ is the probability of being at Gaussian component $k$ at time $t$ given that $\hat{\theta}$ generates $\mathcal{X}$, and can be calculated as:

$$\gamma_{kt} = \frac{\hat{\omega}_k P(x_t|\hat{\mu}_k, \hat{\Sigma}_k)}{\sum_{k'} \hat{\omega}_{k'} P(x_t|\hat{\mu}_{k'}, \hat{\Sigma}_{k'})} \tag{7.3}$$

where $\hat{\omega}_k$, $\hat{\mu}_k$ and $\hat{\Sigma}_k$ are the current estimates of, respectively, the mixture coefficient, the mean vector and the covariance matrix of the $k$-th Gaussian components.

### 7.2.2. Maximum likelihood linear regression

Maximum likelihood linear regression (Leggetter and Woodland, 1995) is an adaptation scheme based on model space linear transformations. In the unconstrained case, the mean and covariance transforms are independent one from another. The general linear transforms of the mean $\mu$ and covariances $\Sigma$ are given by:

$$\hat{\mu} = A\mu + b = W\xi \tag{7.4}$$

$$\hat{\Sigma} = H\Sigma H^{\mathrm{T}} \qquad \text{or} \qquad \hat{\Sigma} = LHL^{\mathrm{T}} \tag{7.5}$$

where $\xi = [\mu^{\mathrm{T}} \quad 1]^{\mathrm{T}}$ is the extended mean vector, $b$ is a bias vector and $W = [A \quad b]$ is the extended linear transform. The covariance matrix can be transformed in either way shown in

(7.5), where $L$ represents Cholesky factor $\Sigma$. $H$, which is the same in both cases, is the transform to be determined.

An EM procedure can be employed to improve the adapted model estimates. First, given the initial model parameters $\theta = (\mu, \Sigma, \omega)$ and the adaptation data $\mathcal{X}$, estimate transforms $W$ and $H$. Given these transforms, use (7.4) and (7.5) to obtain the adapted GMM parameters. Repeat these two steps until convergence. For the formulation used to calculate the model transforms, the reader is referred to the original paper.

### 7.2.3. Constrained maximum likelihood linear regression

Different from standard MLLR, the constrained MLLR (Digalakis et al., 1995; Gales, 1998b) adaptation scheme uses the same transform for the mean and covariance parameters. In this case, the linear transforms can be represented by:

$$\hat{\mu} = \bar{A}\mu - \bar{b} \tag{7.6}$$

$$\hat{\Sigma} = \bar{A}\Sigma\bar{A}^{\mathrm{T}} \tag{7.7}$$

It is straightforward to show that the probability density function of a single Gaussian component of the transformed model can be expressed in terms of the original model as follows:

$$\mathcal{N}(x_t|\hat{\mu}_k, \hat{\Sigma}_k) = |A| \cdot \mathcal{N}\left(Ax_t + b|\mu_k, \Sigma_k\right) \tag{7.8}$$

where $A = \bar{A}^{-1}$, $b = \bar{A}^{-1}\bar{b}$ and $|A|$ denotes the determinant of matrix $A$.

Equation 7.8 express that the CMLLR may be implemented by applying a transformation to the input vector multiplied by the determinant of $A$. In other words, there is no need to actually transform the model parameters, what might be computationally expensive in some cases (Gales, 1998b).

As in the unconstrained case, an EM procedure can be used to improve estimates. In the first step, given the initial model parameters $\theta = (\mu, \Sigma, \omega)$ and the adaptation data $\mathcal{X}$, estimate the transform $W = [A \quad b]$. With $W$, transform the input vectors using $\hat{x}_t = Ax_t + b$. The procedure can be repeated until convergence. During recognition, the relation expressed in (7.8) can be used to compute the likelihood with respect to the adapted model.

### 7.2.4. Adaptive training approaches

The MLLR/CMLLR techniques presented in the previous sections are used to adapt an original model to a new condition that, presumably, was not well represented on the training data. In particular, it can be used to adapt a model to a test segment, a speaker for instance. Such methods can also be applied to the training data in addition to the testing data. Speaker (for instance) dependent transforms are used during the estimation of the mean and covariance parameters. This approach aims to compensate the *inter* speaker variability, letting the model represent only the *intra* speaker variability.

The first proposed method with these characteristics is the speaker adaptive training (SAT) (Anastasakos et al., 1996). It assumes the speaker specific model is a transformation of a unique canonical model. In the standard approach, MLLR was used, while a variation using CMLLR was proposed in Gales (1998b). Alternatively, the speaker specific model parameters can be obtained as a linear interpolation of parameters estimated on subsets of the training data. This method was named cluster adaptive training (CAT) (Gales, 1998a) or eigenvoices adaptation (Kuhn et al.,

1998). In Gales (2001), it was shown that these two latter methods are mathematically equivalent, despite the way the clusters are obtained. In addition, Gales has proposed a generalization of the adaptive training approaches, showing that SAT and CAT are special cases of the same framework.

In the proposed generalized approach, it is assumed that $C$ clusters are available for training and the canonical model consists of $C$ sets of means and a unique set of covariance matrices and mixture coefficients. Thus, only the mean parameters of the canonical model are transformed during training. The general form of the adaptive schemes is:

$$\hat{\mu}_{sk} = W_s \xi_k \tag{7.9}$$

where $\hat{\mu}_{sk}$ is the transformed mean vector of Gaussian component $k$ for speaker $s$, $W_s$ is a speaker dependent transform and $\xi_k$ is the canonical mean vector of component $k$. The size of $\xi_k$ (and consequently $W_s$) depend on the number of clusters used and if a bias component is used or not.

SAT can be seen as a restriction of such a method by setting the number of clusters to 1. On the other hand, CAT uses various speaker clusters plus a bias cluster, with the restriction of having only diagonal transforms. In other words:

$$\textbf{SAT:} \qquad \xi_k = [\mu_k^{\mathrm{T}} \quad 1]^{\mathrm{T}} \qquad W_s = [A_s \quad b_s] \tag{7.10}$$

$$\textbf{CAT:} \qquad \xi_k = [\mu_{1k}^{\mathrm{T}} \ldots \mu_{Ck}^{\mathrm{T}} \quad \mu_b]^{\mathrm{T}} \qquad W_s = [\alpha_{s1} I \ldots \alpha_{sC} I \quad I] \tag{7.11}$$

where $\mu_b$ is a bias cluster, $I$ is the identity matrix and $\alpha$ are scalar numbers.

The adaptive training procedure is a slight variation of the standard EM algorithm used on ML estimation. It can be depicted as follows. Given an initial set of model parameters $\hat{\theta} = \hat{\omega}, \hat{\xi}, \hat{\Sigma}$, estimate speaker specific transforms $W_s$. These transforms are used on the standard ML to estimate a new model $\theta$. For example, for the case where SAT is performed using CMLLR (Gales, 1998b), the update equations of the mean and covariance parameters are very similar to those of standard ML:

$$\mu_k = \frac{\sum_{s=1}^{S} \sum_{t=1}^{T_s} \gamma_{kt} \hat{x}_t}{\sum_{s=1}^{S} \sum_{t=1}^{T_s} \gamma_{kt}} \tag{7.12}$$

$$\Sigma_k = \frac{\sum_{s=1}^{S} \sum_{t=1}^{T_s} \gamma_{kt} (\hat{x}_t - \mu_k)(\hat{x}_t - \mu_k)}{\sum_{s=1}^{S} \sum_{t=1}^{T_s} \gamma_{kt}} \tag{7.13}$$

where $S$ is the total number of speakers, $T_s$ denotes the number of frames for speaker $s$, $\hat{x}_t = A_s x_t + b_s$ is the feature vector adapted with the speaker dependent transform.

## 7.3. Interpolating models

Acoustic model estimation is commonly performed considering the training data as a single homogeneous data set, with the objective function being optimized over all the available data. The ML estimates of the Gaussian parameters are given by:

$$\theta_{ML} = \arg\max_{\theta} f(\mathcal{X}|\theta) \tag{7.14}$$

During decoding, the likelihood function $f(\cdot|\theta)$ is evaluated on the test data, assuming that training and test sets have been generated from the same density function.

Given that the training data are usually heterogeneous, acoustic model interpolation is presented in the following section as a means to weight the degree of relevance of various training subsets with respect to the target specific data.

### 7.3.1. Interpolation of Gaussian mixture models

Linear interpolation is perhaps one of the most straightforward manner to combine models. Here, it was used to combine the GMMs of different component acoustic models in a similar way as done for language modeling. Furthermore, it was assumed that all the acoustic models have the same structure. The mathematical framework will be derived for the GMM of a single state. The extension to the HMM parameters is straightforward.

First, let us assume that the training data can be split into $M$ independent subsets, such as $\mathcal{X} = \{\mathcal{X}_1, \ldots, \mathcal{X}_M\}$, with each subset $\mathcal{X}_m = \{x_{T_{m0}}, \ldots, x_{T_{m0}+T_m-1}\}$ having $T_m$ observation vectors and starting at frame $T_{m0}$. Let $\Theta = (\theta_1, \ldots, \theta_M)$ denote the set of GMMs parameters where each $\theta_m$ has been independently estimated on the associated training subset $\mathcal{X}_m$, each referring to the same HMM state. The likelihood of observing a sample $x_t$ at time $t$, given that the interpolated model has generated $\mathcal{X}$, is:

$$f(x_t|\Theta, \alpha) = \sum_{m=1}^{M} \alpha_m \cdot f(x_t|\theta_m) \tag{7.15}$$

where $\alpha = (\alpha_1, \ldots, \alpha_M)$ represents the set of interpolation coefficients, with the constraint $\sum_m \alpha_m = 1$.

The models $\theta_m$ are GMMs with probability density functions expressed by (for notation clearness, it will be assumed that all Gaussian mixtures have $K$ components):

$$f(x_t|\theta_m) = \sum_{k=1}^{K} \omega_{mk} \cdot \mathcal{N}(x_t|\mu_{mk}, \Sigma_{mk}) \tag{7.16}$$

where $\mathcal{N}(\cdot|\mu, \Sigma)$ is a normal density function with mean vector $\mu$ and covariance matrix $\Sigma$. Substituting (7.16) in (7.15):

$$f(x_t|\Theta, \alpha) = \sum_{m=1}^{M} \sum_{k=1}^{K} \alpha_m \omega_{mk} \cdot \mathcal{N}(x_t|\mu_{mk}, \Sigma_{mk}) = \sum_{l=1}^{L} \beta_l \cdot \mathcal{N}(x_t|\mu_l, \Sigma_l) \tag{7.17}$$

In (7.17), a simple index substitution is performed, with $\beta_l = \alpha_m \omega_{mk}$ and $L = M \times K$. Thus, it follows that the interpolated model is a GMM with $M \times K$ components and mixture weights $\alpha_m \omega_{mk}$. In other words, the interpolated model can be obtained by merging the Gaussian parameters of the component models and adjusting their mixture coefficients. A consequence of this procedure is the increase in size of the interpolated model according to the number of component models used, leading to an increase of computational complexity. However, the same level of complexity can be recovered by applying a GMM reduction algorithm, such as the one proposed in Section 7.4. The GMM interpolation method is illustrated in Figure 7.1.

GMM interpolation has something in common with cluster adaptive training (Gales, 1998a) or eigenvoices (Kuhn et al., 1998) in the sense that these methods use a combination of different

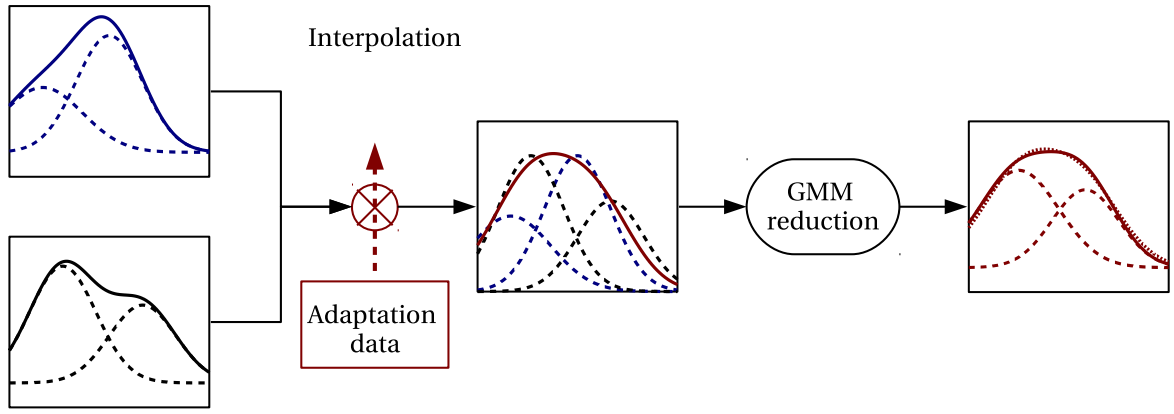



Figure 7.1.: Gaussian mixture model interpolation scheme. Dashed lines correspond to Gaussian components, while solid lines to the global mixture. After GMM reduction, the reduced mixture is plotted by a solid line and the original mixture by a dotted line.

component models based on scalar interpolation coefficients. However, the coefficients are applied at different levels.

More importantly, these methods have quite different purposes. Interpolation aims to generate a task specific model from a set of available component models, which are independently estimated. On the other hand, CAT and eigenvoices (as well as other adaptive training approaches) aim to estimate a canonical model (composed by component models) in which the inter speaker variability is diminished. In the adaptive training approaches, the component model parameters are jointly estimated. This aspect makes interpolation a more flexible approach to be used on adaptation, since any existent model judged to be relevant for a task can rapidly be taken into consideration. Moreover, as shown in Section 8.4, model interpolation can be used on top of speaker adaptive training (Anastasakos et al., 1996).

### 7.3.2. Data weighting

Another approach to take into account the relevance of training subsets for a given task is to re-estimate the model parameters by weighting the observation vectors according to previously selected coefficients. It follows that this approach can also be seen as an interpolation scheme.

Let us consider that the training data can be partitioned into $M$ subsets $\{\mathcal{X}_m\}_{m\in[1,M]}$, each subset having a relevance weight $\alpha_m$, and let $\Theta = \{\theta_m\}_{m\in[1,M]}$ denotes a set of $M$ component mixture models. Let us consider that the model parameters are tied as follows:

$$\begin{aligned} \omega_{mk} &= \omega_k \\ \mu_{mk} &= \mu_k \qquad \forall \quad m \in [1, M] \\ \Sigma_{mk} &= \Sigma_k \end{aligned} \tag{7.18}$$

In this case, the ML estimates of $\Theta$ can be obtained by maximizing the following auxiliary function:

$$Q_\Theta(\Theta; \hat{\Theta}, \hat{\alpha}) = \sum_{k=1}^{K} \sum_{m=1}^{M} \sum_{t=1}^{T_m} P(m, k|x_t, \hat{\Theta}) \cdot \log \omega_k \mathcal{N}\left(x_t|\mu_k, \Sigma_k\right) \tag{7.19}$$

where $\hat{\Theta}$ and $\hat{\alpha}$ denote the current sets of Gaussian mixture parameters and interpolation coefficients; $\omega_k$, $\mu_k$ and $\Sigma_k$ are the new estimates of, respectively, the mixture coefficient, the mean vector and the covariance matrix of the $k$-th Gaussian component. Furthermore,

$$P(m,k|x_t,\hat{\Theta}) = P(m|x_t,\hat{\Theta}) \cdot P(k|x_t,\hat{\theta}_m) \tag{7.20}$$

Given that model parameters are tied (i.e. $\hat{\theta}_i = \hat{\theta}_m \,\forall\, i \in [1,M]$), the second term is actually independent of $m$. For this same reason, the first term can be simplified as:

$$P(m|x_t,\hat{\Theta}) = \frac{P(x_t|\hat{\theta}_m)P(m|\hat{\Theta})}{\sum_{m'} P(x_t|\hat{\theta}_{m'})P(m'|\hat{\Theta})} = \frac{P(x_t|\hat{\theta}_m)\hat{\alpha}_m}{P(x_t|\hat{\theta}_m)\sum_{m'}\hat{\alpha}_{m'}} = \hat{\alpha}_m \tag{7.21}$$

The second term in (7.20) represents the occupancy probability of the $k$-th Gaussian component and is given by (7.3).

The re-estimation equations can be obtained by taking the derivatives of the auxiliary function and equating to zero. They are given by:

$$\omega_k = \frac{\sum_{m=1}^{M}\sum_{t=1}^{T_m}\hat{\alpha}_m\gamma_{kt}}{\sum_{m=1}^{M}\hat{\alpha}_m T_m} \tag{7.22}$$

$$\mu_k = \frac{\sum_{m=1}^{M}\sum_{t=1}^{T_m}\hat{\alpha}_m\gamma_{kt}x_t}{\sum_{m=1}^{M}\sum_{t=1}^{T_m}\hat{\alpha}_m\gamma_{kt}} \tag{7.23}$$

$$\Sigma_k = \frac{\sum_{m=1}^{M}\sum_{t=1}^{T_m}\hat{\alpha}_m\gamma_{kt}(x_t-\mu_k)(x_t-\mu_k)^{\mathrm{T}}}{\sum_{m=1}^{M}\sum_{t=1}^{T_m}\hat{\alpha}_m\gamma_{kt}} \tag{7.24}$$

These equations are similar to the standard ML re-estimation equations (see Section 2.2.2). In particular, they can be obtained by setting $M = 1$. Approximately, $\alpha_m$ can be seen as a factor used to weight the samples of subset $\mathcal{X}_m$.

With respect to the re-estimation equations, data weighting has something in common with speaker adaptive training (Anastasakos et al., 1996), more precisely with the variant using CMLLR (Gales, 1998b). By removing the bias parameter ($b_s = 0$) and restraining the transform to have a diagonal matrix ($A_s = \alpha_s I$), we can rewrite the re-estimation equations of the SAT/CMLLR (7.12) and (7.13) as:

$$\mu_k = \frac{\sum_{s=1}^{S}\sum_{t=1}^{T_s}\alpha_s\gamma_{kt}x_t}{\sum_{s=1}^{S}\sum_{t=1}^{T_s}\gamma_{kt}} \tag{7.25}$$

$$\begin{aligned}\Sigma_k &= \frac{\sum_{s=1}^{S}\sum_{t=1}^{T_s}\gamma_{kt}(\alpha_s x_t-\mu_k)(\alpha_s x_t-\mu_k)^{\mathrm{T}}}{\sum_{s=1}^{S}\sum_{t=1}^{T_s}\gamma_{kt}} \\ &= \frac{\sum_{s=1}^{S}\sum_{t=1}^{T_s}\alpha_s\gamma_{kt}(x_t-\mu_k)(x_t-\mu_k)^{\mathrm{T}}}{\sum_{s=1}^{S}\sum_{t=1}^{T_s}\gamma_{kt}} + \Psi_k\end{aligned} \tag{7.26}$$

with

$$\Psi_k = \frac{\sum_{s=1}^{S}\sum_{t=1}^{T_s}\alpha_s\gamma_{kt}x_t(1-\alpha_s)}{\sum_{s=1}^{S}\sum_{t=1}^{T_s}\gamma_{kt}}$$

Concerning the re-estimation of the mean vectors, both data weighting (7.23) and the restricted SAT/CMLLR (7.25) use a coefficient $\alpha$ to weight the input vectors. Nevertheless, they use slightly different normalization factors. Beyond that, concerning the re-estimation of the covariance matrices, SAT/CMLLR (7.26) contains an additional term $\Psi_k$ compared to data weighting (7.24).

Despite the similarities in the re-estimation formulae, data weighting has different purposes in comparison with adaptive training. While the first attempts to find the best model to represent the adaptation data, the second attempts to find a canonical model that reduces the inter speaker variability. Thus, for data weighting, the set of interpolation coefficients is estimated on the adaptation data, while for SAT/CMLLR, the coefficients (or, more generally, the transforms) are estimated for each training speaker.

### 7.3.3. Estimation of the interpolation coefficients

The interpolation coefficients can be estimated using an EM approach like it is usually done for language modeling (DeMori and Federico, 1999). Given a set of component acoustic models $\hat{\theta}_m$, the interpolation coefficients $\alpha_m$ can be obtained by maximizing the following auxiliary function:

$$Q_\alpha(\alpha; \hat{\Theta}, \hat{\alpha}) = \sum_{m=1}^{M} \sum_{t=1}^{T} \sum_{k=1}^{K} \hat{\alpha}_m \gamma_{kt} \cdot \log \alpha_m \tag{7.27}$$

where $\hat{\alpha}$ represents the current estimates and $\gamma_{kt}$ is given by (7.3). Equation 7.27 represents a partial auxiliary function obtained after a Viterbi approximation for a particular HMM state. Here, the state index $i$ was omitted for simplicity. The re-estimation equation can be obtained by setting to zero the partial derivation of $Q_\alpha$ with respect to $\alpha_m$ constrained by $\sum \alpha_m = 1$, what leads to:

$$\alpha_m = \frac{1}{T} \sum_{t=1}^{T} \frac{\hat{\alpha}_m f(x_t|\hat{\theta}_m)}{\sum_{m'=1}^{M} \hat{\alpha}_{m'} f(x_t|\hat{\theta}_{m'})} \tag{7.28}$$

where $f(\cdot|\hat{\theta}_m)$ is the p.d.f. of a GMM, given by (7.16).

The interpolation coefficients can be estimated on some held-out data or on the test set itself. This task requires an alignment between the data stream and their associated transcriptions in order to calculate the likelihood $f(x_t|\hat{\theta}_m)$ on each of the component models. This alignment can be guided by the manual transcriptions of the held-out data (if available) or can be obtained via decoding. In this latter case, it can be considered a type of unsupervised adaptation scheme.

## 7.4. Gaussian mixture model reduction

Model interpolation as presented in Section 7.3 is performed by merging the GMMs of the component models and properly adjusting the mixture coefficients. The resulting model is a large GMM containing all the Gaussian parameters of the component models. When several components are merged together, the computational complexity required for decoding may increase considerably. A mixture reduction algorithm can be used to recover suitable complexity levels.

Although this is not a recurrent issue in speech recognition, GMM reduction is relevant and well covered in other research areas. For distributed data fusion for instance (Chen et al., 2010), it is usually necessary to combine several GMMs estimated from separate and independent sources.

Given the large number of Gaussian components obtained, a mixture reduction algorithm is necessary to decrease the computational complexity required during processing. For the multiple target tracking task (Ardeshiri et al., 2012; Salmond, 2009), GMM reduction is fundamental given that the number of Gaussian components tends to rapidly increase over time.

Although often necessary to keep the model tractable, parameter reduction entails loss of information. For this reason, most of known approaches define the GMM reduction problem as a minimization of a loss (or divergence) function. A different point of view is given in this section. The reduction problem is defined as a constrained MLE problem, assuming that the training vectors are assembled in clusters.

In the next section, the mixture reduction problem is formally presented, together with a description of some of the most used reduction methods. The proposed algorithm is described in Section 7.4.2.

## 7.4.1. Gaussian mixture model reduction strategies

A straightforward solution to reduce the number of Gaussian components of a given GMM is to prune the components having the lowest mixing coefficients, as applied, for instance, in Blackman (2004). Although, this method can be expected to generate a reduced mixture that may considerably diverge from the original one, especially if the target number of components is small.

Another solution is to estimate the reduced mixture aiming to minimize its divergence with respect to the original mixture. An overview of algorithms based on this concept is given in this section. The reader may refer to Crouse et al. (2011) for another review.

### Problem definition

Let us consider an initial $d$-dimensional multivariate GMM, with $K$ mixture components, represented by $\Phi = (c_1, \cdots, c_K, m_1, \cdots, m_K, S_1, \cdots, S_K)$, where $\omega_k$, $m_k$ and $S_k$ denote, respectively, the mixture coefficient, the $d$-dimensional mean vector and the $d \times d$ covariance matrix for the $k$-th Gaussian component of $\Phi$.

The GMM reduction problem can be defined as the estimation of a model with $L$ components ($L < K$) and parameters $\theta = (\omega_1, \cdots, \omega_L, \mu_1, \cdots, \mu_L, \Sigma_1, \cdots, \Sigma_L)$ in such a way that the reduced model is as close as possible to the original one according to a suitable similarity measure. Alternatively, it can be defined as the minimization of a suitable deviation measure $\mathcal{D}(\cdot)$ between the two mixtures:

$$\theta^* = \arg\min_{\theta} \mathcal{D}(\Phi, \theta) \tag{7.29}$$

The main difference between the algorithms discussed in this section relies on the choice of $\mathcal{D}(\cdot)$ and the manner it is optimized, that is, using global or local decisions.

### Deviation measures

The two most used deviation measures for GMM reduction are the Kullback-Leibler (KL) divergence and the integral squared error (ISE).

The KL divergence (Kullback and Leibler, 1951) constitutes one of the most important concepts in the information theory field. It is a well-known and theoretically motivated asymmetric deviation measure between two probability density functions. The KL divergence of two

multivariate Gaussian distributions $\mathcal{N}_1$ and $\mathcal{N}_2$ is well known and has the form:

$$D_{KL}(\mathcal{N}_1||\mathcal{N}_2) = \frac{1}{2}\left[(\mu_2-\mu_1)^{\mathrm{T}}\Sigma_2^{-1}(\mu_2-\mu_1) + \mathrm{tr}(\Sigma_2^{-1}\Sigma_1) - \log\left(\frac{|\Sigma_1|}{|\Sigma_2|}\right) - d\right] \tag{7.30}$$

where $A^{\mathrm{T}}$, $|A|$ and $\mathrm{tr}(A)$ denote, respectively, the transpose, the determinant and the trace of a matrix $A$.

Unfortunately, the KL divergence between two Gaussian mixtures has no closed form, what makes its use inconvenient for a direct optimization on the GMM reduction problem. Yet, it can be minimized using an iterative clustering algorithm as proposed in Davis and Dhillon (2007).

Scott and Szewczyk (2001) proposed the use of the integral squared error as the optimization criterion for the GMM reduction problem. The ISE between two distributions, $f_1(\cdot)$ and $f_2(\cdot)$ can be represented by:

$$ISE(f_1(x), f_2(x)) = \frac{1}{2}\int_{-\infty}^{\infty}(f_1(x) - f_2(x))^2\,dx \tag{7.31}$$

The main advantage for using such a measure for GMM reduction is that the ISE between two mixtures has a closed form. Therefore, the ISE can be directly minimized using a variety of known optimization algorithms.

### Top-down algorithms

One of the first solutions proposed to the Gaussian mixture reduction problem consists of a greedy *top-down* algorithm (Salmond, 1990). Starting with the original mixture, the algorithm merges the two most similar components at each iteration. In the original work, an *ad-hoc* similarity criterion based on concepts of the statistical analysis of the variance was used. Alternatively, Williams and Maybeck (2003) has proposed to merge components having the lowest ISE, while Runnalls (2007) used an upper bound of the KL divergence.

Though relatively simple, these greedy algorithms have some disadvantages. First, they are based on *local* decisions. Only the similarities between pairs of components are considered at each iteration. The effect on the *global* mixture is not considered. Besides that, these algorithms may be quite costly, especially if the original mixture model contains a large number of components. At each iteration, the similarity between all pairs of Gaussians is calculated.

### Bottom-up algorithms

Rather than starting from the full mixture, Huber and Hanebeck (2008) proposed a *bottom-up* algorithm to generate a reduced approximation of the original mixture. The reduced mixture is initialized with a single Gaussian. Within an iteration, a component is selected to be split into two slightly separate Gaussians. The mixture parameters are adjusted using a progressive Bayes estimation (Hanebeck et al., 2003) in order to minimize the ISE with respect to the original mixture. Hence, different from *top-down* algorithms, this algorithm relies on the optimization of a similarity measure within the overall mixture.

The algorithm proposed in Huber and Hanebeck (2008) was applied to the reduction of a univariate mixture model. The authors have pointed some changes that are needed to extend the algorithm to the multivariate case. Other variations may be applied to this algorithm. For instance, instead of splitting a single Gaussian, all current components might be split at each iteration. In the same way, other global similarity measures could be used to fit the reduced mixture to the original one.

| Algorithm | Type | Deviation measures used |
|---|---|---|
| Blackman (2004) | Prune less relevant components | None |
| Salmond (1990) | Top-down | A measured based on the analysis of the variance |
| Williams and Maybeck (2003) | Top-down | ISE between components |
| Runnalls (2007) | Top-down | Upper-bound of the KL divergence between components |
| Huber and Hanebeck (2008) | Bottom-up with progressive Bayes | ISE between mixtures |
| Valverde et al. (2012) | Progressive Bayes | ISE between mixtures |
| Davis and Dhillon (2007) | Iterative (hard)-clustering | KL divergence between mixtures |
| Schieferdecker and Huber (2009) | Modular (Runnall's + K-means clustering + progressive Bayes) | Upper-bound of the KL divergence between components + ISE between mixtures |

Table 7.1.: Summary of the GMM reduction algorithms described in this section. The progressive Bayes framework is described in Hanebeck et al. (2003).

### Other algorithms

The *top-down* algorithms are often used to generate an initial guess of the reduced mixture model, which is subject to further refinements. Valverde et al. (2012), for instance, used the progressive Bayes framework (Hanebeck et al., 2003) to minimize the ISE, but after initializing the reduced mixture with a greedy algorithm. Schieferdecker and Huber (2009) proposed a modular algorithm that combines the Runnalls' algorithm (Runnalls, 2007), followed by a K-means clustering and an ISE optimization.

Davis and Dhillon (2007) has proposed an iterative solution to refine the reduced mixture based on an agglomerative clustering procedure. This algorithm has been proved to minimize the KL divergence between the original and the reduced mixtures. This solution is similar to the one proposed in this work, but have different theoretical motivation. Furthermore, the proposed solution leads to a *soft* clustering algorithm.

The GMM reduction algorithms described in this section are summarized in Table 7.1.

## 7.4.2. Constrained maximum likelihood estimation

In this section, another interpretation of the mixture reduction problem is given. Instead of minimizing some deviation measure between the original and reduced mixtures, the proposed algorithm is theoretically motivated from the maximum likelihood estimation, with an additional constraint.

### Redefining the GMM reduction problem

Let us consider a $d$-dimensional multivariate GMM with $L$ components and parameters $\theta = (\omega_1, \ldots, \omega_L, \mu_1, \ldots, \mu_L, \Sigma_1, \ldots, \Sigma_L)$. Given a set of $T$ independent identically distributed (i.i.d.)

observation vectors, $\mathcal{X} = (x_1, \cdots, x_T)$, the parameters of the model $\theta$ can be estimated using a ML estimation, as follows:

$$\theta^* = \arg\max_{\theta} \sum_{t=1}^{T} \log f(x_t|\theta) \tag{7.32}$$

where $f(\cdot|\theta)$ represents the p.d.f. of the Gaussian mixture model. By introducing a hidden variable $y$ in (7.32), it can be written as:

$$\theta^* = \arg\max_{\theta} \sum_{t=1}^{T} \log \left( \sum_{y} f(x_t, y|\theta) \right) \tag{7.33}$$

The above equation is true for any hidden variable $y$. In particular, let us consider the case where it refers to a set of $K$ clusters of observation vectors, the clusters being represented by $\{\phi_k\}_{k\in[1,K]}$. Furthermore, let us consider that each vector $x_t$ is constrained to belong to a unique cluster. In other words, each $\phi_k$ can be interpreted as an indivisible packet of training vectors. Mathematically, if a vector $x_t$ belongs to the $j$-th cluster, then:

$$f(x_t|\phi_k) = \begin{cases} 1, & \text{if } k = j \\ 0, & \forall\ k \neq j \end{cases} \tag{7.34}$$

With this constraint, (7.33) can be rewritten as:

$$\theta^* = \arg\max_{\theta} \sum_{t=1}^{T} \log \left( \sum_{k=1}^{K} f(x_t, \phi_k|\theta) \right) = \arg\max_{\theta} \sum_{t=1}^{T} \log \left( \sum_{k=1}^{K} f(x_t|\phi_k, \theta) \cdot f(\phi_k|\theta) \right) \tag{7.35}$$

Then, using (7.34):

$$\theta^* = \arg\max_{\theta} \sum_{t=1}^{T} \log f(y_t = \phi_k|\theta) = \arg\max_{\theta} \sum_{k=1}^{K} n_k \cdot \log f(\phi_k|\theta) \tag{7.36}$$

where $y_t$ refers to the cluster associated to the vector $x_t$ observed at instant $t$, and $n_k$ denotes the number of vectors associated to the cluster $\phi_k$.

The objective function of the proposed GMM reduction algorithm can be obtained by considering that each cluster is, in fact, modeled by a Gaussian component of the original mixture model, that is $\phi_k \sim \mathcal{N}(\cdot|m_k, S_k)$. Thus, representing the original GMM by $\Phi = (c_1, \ldots, c_K, m_1, \ldots, m_K, S_1, \ldots, S_K)$, with $K > L$, and dividing the argument of (7.36) by $T$, we obtain the final form of the constrained maximum likelihood estimation (CMLE) problem:

$$\theta^* = \arg\max_{\theta} \sum_{k=1}^{K} c_k \cdot \log f(\phi_k|\theta) \tag{7.37}$$

where $c_k = \frac{n_k}{T}$ is the mixture coefficients of the $k$-th component of model $\Phi$.

### Expectation-maximization algorithm

The CMLE problem defined in (7.37) is similar to the original MLE formulation. It is also an incomplete data problem which can be solved using an iterative EM approach (Dempster et al., 1977). In this case, the complete data can be defined as $\mathcal{Y} = (\Phi, \mathcal{H})$, where $\Phi = (\{\phi_1, c_1\}, \ldots, \{\phi_K, c_K\})$ is the set of observed Gaussian components and their mixture gains of the original GMM and $\mathcal{H} = (h_1, \ldots, h_K)$ is the sequence of unobserved Gaussian component labels of the reduced model, the one to be estimated.

In this case, the auxiliary function can be written:

$$Q(\hat{\theta}, \theta) = E\left[\log f\left(\Phi, \mathcal{H}|, \theta\right)\middle|\Phi, \hat{\theta}\right] = \sum_{k=1}^{K}\sum_{l=1}^{L} c_k \cdot \gamma_{kl} \cdot \log\left[\omega_l \cdot f(\phi_k|\mu_l, \Sigma_l)\right] \tag{7.38}$$

where

$$\gamma_{kl} = f(l|\phi_k, \hat{\theta}) = \frac{f(l|\phi_k, \hat{\theta})}{\sum_{l'} f(l'|\phi_k, \hat{\theta})} = \frac{\hat{\omega}_l \cdot f(\phi_k|\hat{\mu}_l, \hat{\Sigma}_l)}{\sum_{l'} \hat{\omega}_{l'} \cdot f(\phi_k|\hat{\mu}_{l'}, \hat{\Sigma}_{l'})} \tag{7.39}$$

By definition, the log likelihood of a cluster of vectors $\phi_k$ with respect to a normal distribution parameterized by the mean vector $\mu_l$ and covariance matrix $\Sigma_l$ can be expressed by:

$$\log f(\phi_k|\mu_l, \Sigma_l) = \sum_{x_t \in \phi_k} \log f(x_t|\mu_l, \Sigma_l) = \sum_{t=1}^{T} \log f(x_t|\mu_l, \Sigma_l) \cdot P(x_t|\phi_k) \tag{7.40}$$

In the above equation, $P(x_t|\phi_k)$ represents the probability of a vector $x_t$ to belong to the cluster $\phi_k$, which is 1 only if $x_t \in \phi_k$ and 0 otherwise, according to the constraint added during the definition of the algorithm. Relaxing this constraint, this function can be rewritten as the conditional expectation of the log-likelihood function, as follows:

$$\begin{aligned}\log f(\phi_k|\mu_l, \Sigma_l) =& E\left[\log f(x_t|\mu_l, \Sigma_l)|\phi_k\right] \\ =& \log \mathcal{N}(m_k|\mu_l, \Sigma_l) - \frac{1}{2}\mathrm{tr}\left({S_l}^{-1} S_k\right)\end{aligned} \tag{7.41}$$

where $\mathcal{N}(\cdot|\mu, \Sigma)$ represents a normal distribution with mean vector $\mu$ and covariance matrix $\Sigma$ and $\mathrm{tr}(A)$ the trace of matrix $A$.

The parameter update equations can be obtained by taking the partial derivatives of the auxiliary function given by (7.38) and equating to zero, with the normalization constraint $\sum_l \omega_l = 1$. Using the definition (7.41) in the auxiliary function, we obtain:

$$\omega_l = \sum_{k=1}^{K} c_k \cdot \gamma_{kl} \tag{7.42}$$

$$\mu_l = \frac{1}{\omega_l} \cdot \sum_{k=1}^{K} c_k \cdot \gamma_{kl} \cdot m_k \tag{7.43}$$

$$\Sigma_l = \frac{1}{\omega_l} \sum_{k=1}^{K} c_k \cdot \gamma_{kl} \cdot \left[(m_k - \mu_l)(m_k - \mu_l)^{\mathrm{T}} + S_k\right] \tag{7.44}$$

where $\gamma_{kl}$ is given by (7.39).

### Relation with other clustering methods

The re-estimation equations (7.42) to (7.44) are similar to those used in the cluster algorithm presented by Davis and Dhillon (2007), except for the term $\gamma_{kl}$, which assigns a probability of a cluster $\phi_k$ being at the $l$-th Gaussian component of the model.

The clustering algorithms proposed by the previous authors can be obtained by performing a Viterbi approximation in equations (7.42) to (7.44), i.e.:

$$\gamma_{kl} = \begin{cases} 1 & \text{if } l = \arg\max_{l' \in \{1 \cdots L\}} f(l'|\phi_k, \theta) \\ 0 & \text{otherwise} \end{cases} \tag{7.45}$$

Therefore, the proposed algorithm can be seen as a soft clustering algorithm.

## 7.5. Summary

In this chapter, we proposed to take into account the relevance of different training subsets for acoustic modeling. Inspired by the use of interpolation for language modeling, two combination methods were theoretically presented. In the first one, component models are independently estimated on each training subset and then interpolated. In the second method, the relevance weights are taken into account during the estimation of the acoustic model parameters. The latter is equivalent to train acoustic models with weighted data. In addition, an EM approach was proposed to automatically estimate the interpolation coefficients.

The proposed methods were theoretically compared with adaptive training approaches. It was shown that the re-estimation equations of the data weighting and the speaker adaptive training (Anastasakos et al., 1996) methods are somewhat similar. The GMM interpolation method presents similarities with the cluster adaptive training (Gales, 1998a) and eigenvoices speaker adaptation (Kuhn et al., 1998) approaches in the sense they all make use of scalar coefficients to weight the relevance of multiple component models. However, and more importantly, the interpolation methods have different design goals compared to adaptive training schemes. While data weighting and interpolation are used to build task specific models, adaptive training aims to build canonical models in which the inter speaker variability is reduced.

The main advantage of the GMM interpolation method is the flexibility it provides. Given a set of available component models, task specific models can be rapidly obtained by estimating a few interpolation coefficients and merging the Gaussian mixtures. To some extent, there should be no need to re-train the component models for a new task or when new acoustic data become available. For the implementation suggested in this chapter, this is only true if the structure of the acoustic model (namely the phone set, phone contexts and tied-states table) does not change. Extending the method to allow different phone sets and tied-state tables is a possible direction for improvement. For instance, the model structure could be redefined by specializing the clustering tree to a new phone set as proposed in Schultz and Waibel (2000). Alternatively, the HMM states could be re-clustered based on similarity measures taken over the output Gaussian mixture densities to define a new model structure.

Compared to interpolation, data weighting is less flexible. Whenever additional data are included in the training corpus or a new task is considered, it is necessary to re-estimate all the model parameters.

GMM interpolation leads to an increase of model parameters, which can be compensated using a mixture reduction algorithm. A theoretically correct solution to reduce the number of Gaussian components in an existing mixture was proposed. The solution is based on a constrained version of the standard maximum likelihood estimation and leads to a soft clustering algorithm.

The theoretical framework described in this chapter is empirically assessed in the next chapter.

# Chapter 8

# Experiments with acoustic model interpolation

## 8.1. Introduction

A common approach used for acoustic modeling is to pool all the available training data together as if they were parts of one homogeneous set. However, given that the acoustic data are usually collected from a wide variety of sources, the training data is certainly heterogeneous due to variations across speakers (gender, dialect, accent, background condition, speaking style, etc.) and within a speaker (age, emotional state, etc.) (Benzeghiba et al., 2007). This variability is an important difficulty in ASR, directly affecting the quality and challenging the robustness of recognition systems.

Pooling the training data usually leads to good overall performance levels on test sets also coming from a variety of sources. Generally, it can be expected that some more improvement in performance can be obtained by adapting the pooled model to data coming from the same source as the target set. However, adaptation techniques allow similarity levels to be adjusted only with respect to the complete training data. We hypothesize that more reliable parameter estimates can be obtained by considering different degrees of relevance for different subsets of the training data.

In this chapter, the model interpolation and data weighting schemes proposed in Chapter 7 are empirically assessed. First, in Section 8.2, we empirically motivate the choice for the GMM reduction algorithm proposed in the previous chapter, showing that it outperforms a greedy (Runnalls, 2007) and a hard clustering (Davis and Dhillon, 2007) algorithms. In Section 8.3, the data weighting and GMM interpolation approaches are assessed for the European Portuguese broadcast recognition system. In Section 8.4, the GMM interpolation method is assessed as an alternative to MAP adaptation for generating accent-dependent models for the multiple-accented English data recognition task. We also show that interpolation can be efficiently used to build acoustic models on-the-fly for each test data set and without requiring data from the targeted accent for AM training. A summary is given in Section 8.5.

## 8.2. Gaussian mixture reduction algorithm

The GMM reduction algorithm proposed in Section 7.4 was validated in two different experiments. First, it was applied to reduce the HMM state mixtures of an interpolated model. Second, it was applied to generate a universal background model (UBM) GMM from the acoustic model in order to perform adaptation.

| Initialization | EM clustering | | |
|---|---|---|---|
| | No EM | Hard | Soft |
| Single Gaussian | - | 36.6 | **35.8** |
| Gaussians with highest coefficients | 39.3 | 36.5 | **36.3** |
| Runnalls' algorithm | 36.4 | 36.0 | **35.8** |

Table 8.1.: Recognition results with acoustic models obtained using three different GMM reduction algorithms (Runnalls', hard clustering and soft clustering) for three different types of initialization. For reference, the WER obtained with the model without reduction is 34.5%.

### 8.2.1. Reduction of HMM state mixtures

The GMM reduction algorithm was assessed with the Portuguese broadcast recognition system. An acoustic model containing 160 Gaussians per state was built after interpolating five component models trained on `trainR01`, `trainR10`, `trainV09`, `trainE10` and `trainQ10` (see Section 3.2.1). The same interpolation coefficient (0.2) was set for all models. Each of the state mixtures was reduced to 32 components using the Runnalls' algorithm (Runnalls, 2007), hard clustering (Davis and Dhillon, 2007) and the soft clustering algorithm proposed in Section 7.4.

For the hard and soft clustering algorithms, three types of initialization were assessed. First, the *single Gaussian* that represents the whole mixture is used. Before each re-estimation iteration, all the Gaussians of the reduced models are split until the desired number of components is achieved. Second, the Gaussian components having the highest mixture gains were selected for initialization. Third, the reduced mixture was initialized with the Runnalls' algorithm (Runnalls, 2007) like used in Schieferdecker and Huber (2009).

The models with reduced mixtures were used to decode the development set `devQ10` with the baseline language model `LM_10src`. No speaker adaptation was done during decoding. The results are summarized in Table 8.1. With the same system, the acoustic model without GMM reduction obtains a word error rate of 34.5%. As expected, a loss in performance is observed with the use of GMM reduction. The proposed algorithm outperforms the Runnalls' algorithm (35.8% *vs.* 36.4%). For all three initialization procedures used, the proposed soft clustering algorithm outperformed hard clustering. The best performances obtained with clustering methods are 36.0% (hard) and 35.8% (soft).

### 8.2.2. Estimating universal background models

A universal background model is generally a mixture model that approximates the full HMM/GMM acoustic model. UBMs are useful, for instance, to perform a simplified vocal tract length normalization, speaker recognition or CMLLR adaptation. UBMs are often estimated from scratch using a sampled subset of the training data.

A fast-training alternative is to generate the UBM via GMM reduction. In this case, an initial global mixture is formed by merging all the Gaussian components of the HMM/GMM acoustic model and scaling the mixture gains according to the number of training frames associated to each state. Then, this GMM is reduced to the desired number of components.

UBMs obtained via reduction were compared to equivalent UBMs estimated from scratch for a CMLLR adaptation task. Three systems (with different target languages, acoustic feature vectors and training criteria) were evaluated on their respective development sets. The results are summarized in Table 8.2. Generally, the reduced GMM obtains equivalent (or better) WERs compared to UBMs trained from scratch (absolute reductions from 0.1% to 0.3% were observed).

| Language | Amount of data | Features | Training criterion | UBM estimation | |
|---|---|---|---|---|---|
| | | | | from scratch | GMM reduction |
| Portuguese | 156h | PLP | MLE | 33.4 | **33.1** |
| Portuguese | 378h | MLP+PLP+F0 | MMIE | 23.1 | **23.0** |
| Latvian | 628h | MLP+PLP+F0 | MLE | 21.6 | **21.5** |

Table 8.2.: Recognition results obtained with UBM/GMMs estimated from scratch or via GMM reduction of acoustic models. The results are show in terms of WER(%) on the development sets of the different systems.

| Source | Training | |
|---|---|---|
| | Name | Dur |
| RTP, Voice of America, Euronews | `trainRVE` (`trainR01`+`trainR10`+`trainV09`+`trainE10`) | 173.3 |
| Quaero | `trainQ10` | 71.7 |

Table 8.3.: Information of the European Portuguese corpus used to assess the impact of coefficients for acoustic model interpolation. Duration (Dur) is given in hours.

## 8.3. European Portuguese broadcast data recognition

In this section, acoustic model interpolation is applied to a large vocabulary Portuguese speech recognition task and compared to the pooled and with MAP adapted models. The experiments were carried out using the *improved* LIMSI European Portuguese ASR system described in Section 3.2.4.

### 8.3.1. Impact of interpolation coefficients

A set of experiments was carried out in order to validate the automatic determination of the interpolation coefficients. The training data were separated into two different sets, `trainRVE` and `trainQ10`, comprised respectively of about 173 and 72 hours of data as summarized in Table 8.3.

Two pairs of coefficients were estimated using the EM procedure described in Section 7.3.3, with the estimation guided by either the manual or the automatic transcriptions of `devQ10`. Both obtained similar interpolation coefficients, 0.20/0.80 and 0.25/0.75 for `trainRVE`/`trainQ10`, respectively. Other coefficient pairs (see Figure 8.1) were manually set to assess their influence in the recognition performance. For each pair of coefficients, models were estimated using the GMM interpolation and data weighting methods described in Section 7.3. The models were evaluated on the `devQ10` and `testQ11` sets.

The recognition results obtained are plotted in Figure 8.1. The horizontal axis gives the coefficient values selected for `trainQ10`. The WERs obtained with the two methods, data weighting and GMM interpolation, are generally quite close. The largest difference is 0.3% absolute on `testQ11` for coefficients of `trainQ10` ranging from 0.50 to 0.75. These differences are probably due to the fact that models obtained via data weighting have all the parameters optimized by taking the interpolation coefficients into account. For the models obtained via GMM interpolation, only a few parameters (the coefficients) are adjusted.

The model estimated only on `trainQ10` (coefficient = 1.0) has a significant lower WER than the model estimated only on `trainRVE` (coefficient = 0.0), even if this latter was trained on 2.4

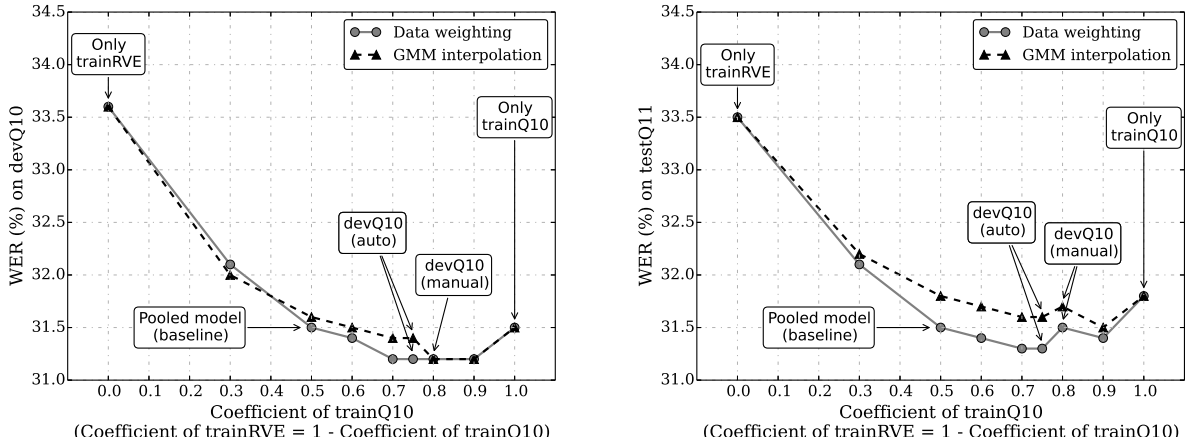



Figure 8.1.: Evaluation of acoustic models obtained using data weighting or GMM interpolation with different pairs of mixture coefficients. WER(%) on the development set `devQ10` set (a) and `testQ11` (b). The sources combined are `trainRVE` and `trainQ10` (see Table 8.3).

times as much data. This is because `trainQ10` comes from the same source and epoch as `devQ10` and `testQ11`, and therefore does a better match to the target data.

The pooled model (baseline) is the one with same coefficients assigned for `trainQ10` and `trainRVE` (coefficient = 0.5). Comparing the WER obtained with the pooled model and the model trained only on `trainQ10` (coefficient = 1.0), it can be seen that adding the 173 hours of `trainRVE` do not lead to improvements on `devQ10` and to an absolute improvement of 0.3% (from 31.8% to 31.5%) on `testQ11`.

Different WERs were obtained by manually varying the interpolation coefficients, reaching a minimum around the coefficients estimated on the automatic and manual transcriptions of `devQ10`, 0.75 and 0.80 respectively. The estimated coefficients obtain absolute improvements of about 0.2–0.3% compared to the pooled models. These results suggest that near optimal interpolation coefficients can be automatically selected without requiring manual transcriptions for the development data.

### 8.3.2. Comparing interpolation, data weighting, pooling and MAP adaptation

A set of experiments was carried in order to assess data weighting and GMM interpolation as a means of generating acoustic models adapted to the target source. The 245 hour untranscribed training data were divided into five subsets, namely `trainR01`, `trainR10`, `trainV09`, `trainE10` and `trainQ10`, according to the show source and broadcast date. The training corpus information is summarized in Table 8.4a.

The systems were evaluated on four test sets coming from the same sources as were used for training, namely the sets `testR00`, `testV09`, `testE10` and `testQ11` as presented in Table 8.4b. For each test set, an associated held-out set was used to estimate the interpolation coefficients. There is no overlap of the held-out data with the training or the evaluation data. The held-out sets `heldE09` and `heldV09` have not been used in other experiments. They correspond to data collected from shows broadcast in 2009 from the respective sources. The system parameters

| Source | Name | Dur |
|---|---|---|
| RTP | trainR01 | 55.6 |
| RTP | trainR10 | 78.3 |
| Voice of America | trainV09 | 21.2 |
| Euronews | trainE10 | 18.2 |
| Quaero | trainQ10 | 71.7 |

(a) Acoustic training corpus.

| Source | Held-out | | Evaluation | |
|---|---|---|---|---|
| | Name | Dur | Name | Dur |
| RTP | trainR00 | 3.1 | testR00 | 1.2 |
| Voice of America | heldV09 | 4.9 | testV09 | 2.1 |
| Euronews | heldE09 | 4.5 | testE10 | 1.9 |
| Quaero | devQ10 | 3.3 | testQ11 | 3.5 |

(b) Held-out and evaluation sets.

Table 8.4.: Information of the European Portuguese corpus used for acoustic model interpolation experiments. The held-out sets were used to estimate the interpolation coefficients. Duration (Dur) is given in hours.

were tuned on devQ10. Automatic transcriptions for the held-out data were generated using the pooled acoustic model estimated on all of the training data.

A baseline acoustic model was created by pooling the 245 hour audio training data. One iteration of unsupervised MAP estimation was used to adapt the pooled model to each source. Adaptation was guided using the automatic transcriptions of either of the trainR01, trainV09, trainE10 and trainQ10 sets.

Source-specific models were also created via data weighting and GMM interpolation. First, five acoustic models were independently estimated on each training subset via maximum likelihood estimation. Interpolation coefficients were estimated on each of the four held-out data sets. These coefficients were then taken into consideration to create four source-specific models via data weighting and other four via GMM interpolation. For reference, another model was generated via GMM interpolation using equally set coefficients.

Acoustic models were estimated using an unsupervised training approach with the 1-best hypothesis considered as ground truth. All the acoustic models have the same structure, covering about 15.7k phone contexts with 11.5k tied-states. The models obtained via pooling, MAP adaptation and data weighting have 32 Gaussians per state, while the models obtained via interpolation have 160 ($5 \times 32$).

The LM_10src language model trained on a corpus comprised of ten different sources and having about 639M words (Section 3.2.4) was used for all experiments.

Table 8.5 summarizes the recognition results obtained on testR00, testV09, testE10 and testQ11. The last row corresponds to the global WER obtained on the entire test set. The second column of the table gives the WERs obtained with the pooled model, which has a WER 22.6% on the combined test set. MAP adapted models obtain a better global WER (22.4%) compared to the baseline, although a loss in performance is observed for the testR00 and testE10 sets. The highest gain obtained via MAP adaptation is observed for the testV09 set (0.5% absolute).

When models are generated using data weighting, the use of estimated coefficients obtains a 0.4% WER absolute gain compared to the use of equally set coefficients (which is equivalent to the baseline pooled model). For models obtained via interpolation, the use of estimated

| Test set | Data weighting | | GMM Interpolation | | MAP |
|---|---|---|---|---|---|
| | equal (pooled) | auto | equal | auto | |
| testR00 | 24.7 | 24.5 | 25.1 | 25.0 | 25.1 |
| testV09 | 14.4 | **13.6** | 14.3 | 13.9 | 13.9 |
| testE10 | 13.2 | 13.1 | 13.7 | 13.6 | 13.3 |
| testQ11 | 31.5 | 31.2 | 31.9 | 31.6 | 31.3 |
| Overall | 22.6 | **22.2** | 22.9 | 22.6 | 22.4 |

Table 8.5.: Recognition results with acoustic models obtained via data weighting, GMM interpolation, pooling or MAP adaptation. The sources combined are those presented in Table 8.4a. WER (%) is measured on the test sets testR00, testV09, testE10 and testQ11. For data weighting and interpolation, the use of equally set coefficients ('equal') is compared to the use of automatically selected ('auto') ones.

| Test set | GMM interpolation of MAP adapted models | |
|---|---|---|
| | equal | auto |
| testR00 | **24.4** | 24.8 |
| testV09 | 13.8 | **13.6** |
| testE10 | 13.2 | **13.0** |
| testQ11 | 31.2 | **31.1** |
| Overall | 22.3 | **22.2** |

Table 8.6.: Recognition results with acoustic models obtained via interpolation of MAP adapted models. The sources combined are those presented in Table 8.4a. WER (%) is measured on the test sets testR00, testV09, testE10 and testQ11. The use of equally set coefficients ('equal') is compared to the use of automatically selected ('auto') ones.

coefficients outperform equally ones on about 0.3% absolute.

The best performance levels are obtained using data weighting with automatically estimated coefficients. Contrary to MAP adaptation, no loss in performance is obtained on the individual test sets. Compared to the pooled and MAP adapted models, absolute WER reductions of 0.4% and 0.2% are respectively obtained.

We observe that models obtained via GMM interpolation perform worse than those obtained via data weighting. This is probably due to the fact that some of the component models were poorly estimated. More precisely, the acoustic models trainV09 and trainE10 were estimated on about 20 hours of audio data. Given that the HMM state-tying table was fixed, a part of the state GMMs was certainly trained with only a few training samples.

To generate acoustic models with "well-trained" mixtures, the pooled model was adapted to each of the five training sources (Table 8.4) via MAP estimation. These five models were interpolated using equally set coefficients and coefficients automatically estimated on each of the four held-out data sets. These models were used to decode testR00, testV09, testE10 and testQ11. Table 8.6 summarizes the results obtained.

The models obtained with the interpolation of MAP adapted models using automatically estimated coefficients outperform on only 0.1% the model obtained with equally set coefficients. Both interpolated models obtain similar WERs (22.3% for 'equal', 22.2% for 'auto') compared

to the models created via data weighting (22.2%, c.f. Table 8.5).

## 8.4. Multiple accented English data recognition

According to Huang et al. (2001), accent is, behind gender, the second principal source of speech variability. Even if substantial progress has been made in the techniques for normalization and speaker adaptation to compensate some of the variability, recognition accuracy has been shown to strongly degrade when the accent of the test speaker is not well represented in the training data (Fischer et al., 2001; Wang et al., 2003; Zheng et al., 2005).

A general approach to deal with accented data is to adapt a prior model to the target accent (Wang et al., 2003). This approach can be extended to the multi-accented data case Huang et al. (2004); Zheng et al. (2005). The main principle is to build different accent-dependent models and use a selector for adaptation. This procedure was adopted by Vergyri et al. (2010) to recognize multiple accented English data. In that work, accent-specific models were created for six different geographical regions where English is spoken as an official language. Similar to Chen et al. (2001), a GMM based classifier was used to select an accent-dependent model for each test segment. On average, a significant improvement was obtained over the accent-independent system, although the accuracy of one of the accents (Middle-East [1]) degraded.

In this section, we revisit the problem addressed in Vergyri et al. (2010). However, instead of using the approach based on MAP adaptation (Gauvain and Lee, 1994), we propose to build the accent-dependent models via interpolation, with coefficients automatically estimated as described in Section 7.3.3. In this case, model interpolation is performed by merging the GMMs of the component models and properly adjusting the mixture coefficients (Section 7.3.1). Accent adaptation is restrained to acoustic modeling in order to validate the interpolation method proposed. We therefore assume that the remaining system components (pronunciation dictionary, language models) provide a good coverage for all the accents represented in the data set.

Interpolation was assessed in two ways. For the first method, accent-dependent models are built via interpolation during the training phase and used during decoding after model selection. For the second method, the component models are interpolated on-the-fly, with coefficients estimated for each test segment. Therefore, instead of making a hard decision for an accent, a smoothed combination is performed.

In the following we described the experiments performed to validate the acoustic model interpolation as an adaptation technique. Before reporting results, the experimental setup is presented. In the second, model interpolation is used as an accent adaptation technique. After, it is used to generate speaker and show specific models in an unsupervised manner on-the-fly. Finally, we show that the technique is also very useful when the target accent is not represented in the training set.

### 8.4.1. Experimental setup

The data and baseline system used in this work are the same as presented in Vergyri et al. (2010) and described in Section 3.3. As a reminder, the most important details are given here.

The duration information of training and test sets of the broadcast news multi-accented English corpus used in this work is reproduced in Table 8.7. For these experiments, two additional subsets were defined, `train6h` and `held-out`. These are non-overlapping randomly selected subsets of the training data and were used to estimate the interpolation coefficients for each accent. In contrast to the experiments carried out on Portuguese, manual transcriptions of

[1] In Vergyri et al. (2010), the labels for the ME and NA accents were mistakenly swapped.

| Set (Duration) | US | AU | GB | ME | NA | IN | Total |
|---|---|---|---|---|---|---|---|
| Training (hours) | 316.6 | 33.0 | 55.4 | 27.7 | 8.2 | 9.4 | 450.3 |
| Train6h (hours) | 6.0 | 6.0 | 6.0 | 6.0 | 6.0 | 6.0 | 36.0 |
| Held-out (hours) | 3.6 | 2.0 | 2.0 | 1.6 | 2.2 | 1.7 | 13.1 |
| Test (min) | 172 | 12 | 48 | 15 | 13 | 15 | 275 |

Table 8.7.: Duration of the multi-accented English data sets. Accents: United States (US), Australia (AU), Great Britain (GB), Middle East (ME), North Africa (NA), India (IN).

the entire training and held-out data sets were available. Thus, the models were trained in a supervised manner.

As a reminder, the multi-accented English system uses a 42-dimensional PLP-like (Hermansky, 1990) acoustic feature vector (12 cepstrum coefficients, log energy and pitch, along with their first and second derivatives).

The phone set contains 35 phones and special units for silence, breath and hesitation markers. The HMM-GMM acoustic models cover about 18k phone contexts and contain 11.5k tied states. They are all gender dependent, speaker adapted (Anastasakos et al., 1996), and were obtained via MMIE (Bahl et al., 1986).

Two sets of models can be considered as baseline for this work, both being described in Vergyri et al. (2010). The first one consists of an accent-independent, gender-dependent model generated by pooling all the training data together. The second are the accent-dependent, gender-dependent models obtained via a joint (accent, gender) MAP adaptation scheme of the accent-independent, gender-independent prior model.

The multi-accent dependent system uses a GMM classifier to select the most likely accent for each test segment. The corresponding accent-dependent model is used for decoding. The decoding in the LIMSI systems (see Section 3.3) includes unsupervised MLLR and constrained CMLLR for speaker adaptation (Leggetter and Woodland, 1995).

The main results obtained by Vergyri et al. (2010) are reproduced in the first two entries of Table 8.8. On average, the multi-accented system (with per show accent identification) performs better than the accent-independent system, with the exception of the ME accent.

## 8.4.2. Accent-adaptation via model interpolation

Model interpolation was assessed via supervised adaptation, but with automatically estimated coefficients. In a first step, the `train6h` subset was used to estimate context-independent models for each accent. Interpolation coefficients were estimated in order to maximize the likelihood of the `held-out` data set. In a second step, the MAP adapted accent-dependent models were interpolated using the estimated coefficients.

After merging the Gaussian mixtures, each accent-specific (interpolated) model contains 192 components per GMM. The mixtures were reduced to 32 components using the algorithm proposed in Section 7.4. During decoding, a GMM classifier was used to select the accent-specific interpolated model for each show, as performed in the baseline multi-accented system. The results obtained with this approach are shown in the second part of Table 8.8.

On average, the interpolated models obtained the best word recognition performances on the `test` data. With respect to the two baseline systems ("Show-accent-ID" and "Accent-independent"), relative improvements of 2.2% and 6.3%, were achieved. The reduced interpolated models also outperform the two baseline systems (1.8% and 6.0% relative), but with a small loss in performance compared to the large interpolated model. Contrary to the "Show-accent-ID"

| System | US | AU | GB | ME | NA | IN | Overall | Ave |
|---|---|---|---|---|---|---|---|---|
| Accent independent | 14.34 | 11.92 | 12.84 | 15.90 | 26.47 | 39.28 | 16.07 | 20.12 |
| Show-accent-ID | 13.95 | 11.91 | 11.98 | 16.46 | 25.19 | 34.28 | 15.39 | 18.96 |
| Interpolated (equal) | 14.45 | 11.64 | 12.30 | 16.34 | 25.49 | 35.81 | 15.84 | 19.34 |
| Interpolated (auto) | 13.83 | 11.55 | 11.08 | 15.79 | 24.29 | 33.52 | **15.05** | **18.34** |
| Interpolated reduced | 13.75 | 11.87 | 11.45 | 15.79 | 24.89 | 33.95 | 15.11 | 18.62 |
| On-the-fly (speaker) | 14.11 | 11.37 | 11.65 | 15.63 | 24.24 | 33.18 | 15.27 | 18.36 |
| On-the-fly (show) | 14.06 | 11.18 | 11.30 | 15.86 | 24.24 | 33.69 | 15.22 | 18.39 |

Table 8.8.: Recognition results using different recognition systems for each of the six regional English accents. 'Overall' corresponds to the WER(%) on the combined `test` set, while 'Ave' to the average WER when each subset is weighted equally (see Table 8.7). The first part shows results for the baseline models; the second and third parts, for models interpolated, either during the training (2nd), or decoding (3rd) phase. Acoustic models were trained on 450 hours of data.

system, the interpolated models obtain similar performance levels as the "Accent-independent" system for the ME data. The improvements obtained with the interpolated models are significant with respect to both baseline systems, according to the MAPSSWE significance test (Pallett et al., 1990).

### 8.4.3. On-the-fly acoustic model interpolation

Acoustic model interpolation was assessed as an unsupervised adaptation technique. The component models are not interpolated during training, but interpolation coefficients are estimated during decoding by maximizing the likelihood of the test data. The estimated coefficients are used to create a segment-specific model on-the-fly by interpolation of the component models. GMM reduction was not applied in this case. The third part of Table 8.8 provides recognition results for speaker- or show-specific interpolated models.

On-the-fly interpolation based on speaker or show test segments outperforms "Show-accent-ID" for all accents but US. On-the-fly interpolation outperforms the accent independent system for all accents, even US, which is well represented on the accent independent model (70% of acoustic training data is US).

### 8.4.4. Leaving target accent out

Another set of experiments was performed to assess the case where the target accent data is not represented in the training corpus. The procedure used to create the component models is similar to the one used in the previous experiments, except that the subset corresponding to the target accent was not used for training. For instance, the acoustic model used to decode the US test data was created using about 134 hours of data rather than 450 hours (450 hours from the entire training data set - 317 from the US training data; c.f. Table 8.7). The amount of audio data used for build acoustic models for each dialect are shown in Table 8.9a.

It was assumed that the `held-out` sets were available. They were used only for MAP adaptation and for estimating the interpolation coefficients. MMIE was not used on top of the MAP adapted models, since it led to loss in recognition performances for the US data. (We did not try for the other accents). The systems were evaluated on the `test` and `train6h` sets, since

| US | AU | GB | ME | NA | IN |
|---|---|---|---|---|---|
| 133.7 | 417.3 | 394.9 | 422.6 | 442.1 | 440.9 |

(a) Amount of data (in hours) used to build acoustic models for each dialect.

| System | US | AU | GB | ME | NA | IN | Overall | Ave |
|---|---|---|---|---|---|---|---|---|
| Accent-independent | 21.31 | 13.99 | 15.29 | 18.77 | 29.33 | 44.69 | 21.69 | 23.90 |
| Adapted to `held-out` | 20.69 | 13.02 | 14.39 | 19.15 | 28.58 | 41.11 | 20.90 | 22.82 |
| Interpolated (auto) | 20.42 | 13.35 | 13.30 | 17.25 | 27.33 | 40.84 | **20.41** | **22.08** |
| Interpolated reduced | 20.66 | 13.48 | 13.79 | 17.45 | 28.08 | 41.00 | 20.70 | 22.41 |
| On-the-fly (speaker) | 20.65 | 12.61 | 13.40 | 17.09 | 28.38 | 40.51 | 20.56 | **22.11** |

(b) WER(%) on `test` sets.

| System | US | AU | GB | ME | NA | IN | Overall | Ave |
|---|---|---|---|---|---|---|---|---|
| Accent-independent | 19.87 | 20.06 | 14.86 | 17.22 | 27.46 | 38.24 | 23.17 | 22.95 |
| Adapted to `held-out` | 19.04 | 17.39 | 14.16 | 17.41 | 25.37 | 34.69 | 21.54 | 21.34 |
| Interpolated (auto) | 19.12 | 18.06 | 12.88 | 15.86 | 24.61 | 35.66 | **21.25** | **21.03** |
| Interpolated reduced | 19.41 | 18.41 | 13.13 | 16.33 | 24.99 | 36.45 | 21.68 | 21.45 |
| On-the-fly (speaker) | 19.40 | 19.10 | 13.20 | 15.71 | 25.36 | 36.06 | 21.69 | 21.47 |

(c) WER(%) on `train6h` (as test) sets.

Table 8.9.: Recognition results for the case where no training data in the dialect was available for acoustic model training. The amount of AM training data used for each dialect is shown in (a). The `held-out` sets were used only for MAP adaptation and to estimate the AM interpolation coefficients. No MMIE was performed. 'Overall' corresponds to the WER on the combined (b) `test` and (c) `train6h` sets, while 'Ave' to the average WER when each subset is weighted equally (see Table 8.7). These results cannot be compared with those in Table 8.8 as the training conditions are different.

these latter were not used for training.

### Results on `test`

The recognition results obtained on the `test` sets are reported in Table 8.9b. First row presents the results with the accent-independent system. The second part of each table shows the multi-accented systems with per show accent-identification, but with accent-dependent models built via MAP adaptation to the `held-out` data, via interpolation or via interpolation followed by a GMM reduction. Last row presents the system with models interpolated on-the-fly for each speaker.

As in previous experiments, all the four multi-accented systems outperform the accent-independent system.

For the `test` sets, the interpolated models outperform the adapted ones with a global relative WER reduction of 2.2% (20.41% vs. 20.90%). Even with reduced mixtures, the interpolated models still outperform the adapted ones for three accents (GB, ME, NA) and obtain similar performance levels for two (US, IN). However, reducing the mixtures of the interpolated model leads to a slight loss in performance. It is important to remind that manual transcriptions of the held-out data were used. As reported in Section 8.3, this has little impact for the estima-

tion of interpolation coefficients. However, it is well-known that supervised MAP estimation is substantially better than unsupervised estimation (c.f. systems F and G in Table 3.8, for instance).

Models interpolated on-the-fly outperform adapted models for five accents, giving similar performance for the US accent. This approach leads to the best recognition performances for three accents (AU, ME, IN) compared to the other four systems. This is an interesting result, since this latter setup does not require any held-out data, as adaptation is performed on speaker cluster basis (on top of CMLLR/MLLR adaptation).

### Results on `train6h`

The models built in the previous experiment were evaluated on the `train6h` data sets to report results on a test set larger than `test`. As a reminder, the `train6h` data were not used for AM training in this condition. The results obtained on the `train6h` sets, shown in Table 8.9c, agree in part with those obtained on the `test` sets. The interpolated models (3rd row) obtain better overall performance levels than the adapted models (2nd row) (21.25% vs. 21.54%). For three accents (GB, ME, NA), the interpolated models outperform the adapted ones. For the US data, these two systems are equivalent. On the other hand, adapted models outperform interpolated models for the AU and IN accents.

On-the-fly interpolation (last row) obtains similar recognition performances than the MAP adapted models (21.69% vs. 21.54%). The former outperforms the latter for two accents (GB, ME) and is similar for the IN data. For the remaining three accents, it is worse. However, on-the-fly interpolation is an unsupervised adaptation technique as it does not use any data from the target dialect, except for the test set itself.

## 8.5. Summary

In this chapter, the GMM reduction algorithm, the acoustic model interpolation and data weighting methods proposed in Chapter 7 were empirically assessed. The proposed GMM reduction algorithm performed better than a greedy (Runnalls, 2007) and a hard-clustering (Davis and Dhillon, 2007) algorithms when used to reduce the mixtures of an acoustic model built via interpolation. The proposed algorithm was also used to generate universal background models from HMM/GMM based acoustic models as a faster alternative than training UBMs from scratch. It was shown that the EM algorithm can be used to estimate near optimal interpolation coefficients even on an untranscribed held-out data set. Estimated coefficients led to better performance levels than manually set coefficients.

The data weighting and GMM interpolation methods were used to build source-specific acoustic models for a Portuguese broadcast recognition task. Both outperformed the baseline pooled model and the MAP adapted models on four evaluation sets. (Although interpolation generates models with a larger number of parameters). It was shown that GMM interpolation requires the use of "well-trained" component models to achieve suitable performance levels.

This work used a single interpolation coefficient assigned to each component model. In non-reported experiments carried out with the Portuguese ASR system, the use of phone specific interpolation coefficients did not bring improvements. Acoustic models for the Portuguese ASR system were estimated using an unsupervised training approach taking the 1-best decoding hypothesis as ground truth. The improvements obtained with model interpolation are expected to be additive to those obtained with confidence-based weighting or filtering approaches (Chapter 4).

The GMM interpolation method was also assessed for the recognition of multiple-accented English data as a means to generate accent-specific acoustic models. Compared to a previously proposed approach (Vergyri et al., 2010) that relies on MAP adaptation, interpolation led to a small but significant gain in performance for all the six accents represented in the corpus. The algorithm proposed in Section 7.4 was used to reduce the state mixtures of the models obtained via interpolation in order to perform a fair comparison with the pooled and the MAP adapted models. The reduced interpolated models consistently outperformed the pooled and the MAP adapted ones, although with some loss in performance compared to the larger interpolated models.

Interpolation was also assessed as an unsupervised adaptation scheme for the multiple-accented English data task. Interpolation coefficients were estimated on-the-fly and used to generate acoustic models for each test speaker or show. This approach has the advantage of not requiring held-out data, but performed slightly worse than accent-specific interpolated models.

Another condition was assessed for the English recognition task: data for the target accent were not used for acoustic model training. Only a small amount of held-out data was used for adaptation and to estimate the interpolation coefficients. This condition required training models on all other data sources for each target accent. GMM interpolation consistently outperformed the accent-independent system on each of the six test accents. For three accents, GMM interpolation led to better performance levels than MAP adapted models.

It was shown that on-the-fly interpolation is especially useful for more extreme conditions in which no data for the target accent are available neither for acoustic model training nor for adaptation. In this condition, on-the-fly interpolation led to substantial gains in performance compared to the baseline accent-independent system.

# Part IV.

# Conclusions

# Conclusions

This last chapter summarizes the contributions of this work and discusses the conclusions that can be drawn from them. Some possible extensions of this work are also suggested.

## Contributions

We investigated different approaches to reduce the costs required for ASR system development. A large part of these costs are due to the human effort required to produce a **large** amount of manual audio transcripts, which are fundamental for acoustic model training and well-suited for language model training. Additional costs are required when the target data is **scarce** in accessible sources (TV, radio, Web, podcasts) due to the extra effort applied for data collection and system refinements. Among other tasks, this is the case for recognizing data from low e-resourced languages and dialects. Two axes of research were explored: 1) the use of unsupervised training methods as a means to reduce the need for manually transcribed audio data; and 2) the use of acoustic model interpolation as a means to reduce the need for target specific acoustic data.

**Unsupervised training approaches** were applied to build three main components of state-of-the-art ASR systems: the acoustic models, the MLPs used to extract acoustic features and the language models. Several conclusions can be drawn from this work.

As has been previously reported, unsupervised AM training approaches can be used to substantially improve the performance of ASR systems without requiring manual transcriptions (Chapter 4). Unsupervised AM training can be further improved by applying confidence based filtering and weighting methods to diminish the impact of recognition errors contained in the automatic transcriptions. The training strategy applied for unsupervised training also matters. We observed that carefully choosing the manner how the training data subsets are used across the training iterations can avoid the propagation of the recognition errors and significantly improve the model accuracy.

A novel approach for unsupervised AM training was proposed, that is, the use of multiple decoding hypotheses (rather than the best one) to guide the acoustic model parameter estimation (Chapter 4). The use of multiple decoding hypotheses consistently outperformed the standard unsupervised AM training approaches, especially when the automatic transcriptions used for training contain a large amount of errors. We proposed and justified theoretically the use of multiple decoding hypotheses as a smother approximation to the true transcription labels than the 1-best case.

In this work, we also proposed to extend the unsupervised training framework to the use of MLP features (Chapter 5). We showed that untranscribed data can be used to efficiently

estimate the parameters of MLPs applied for feature extraction. The proposed method was used to create accurate acoustic models with MLP-based features in a fully unsupervised manner. The results obtained were very competitive to cross-lingual MLPs. An MLP trained on 316 hours of *untranscribed* data from the target language performed similar as a cross-lingual MLP trained on 600 hours of *transcribed* data.

Automatically generated audio transcriptions were also explored as a means to improve the language model estimates (Chapter 6). By adding a relatively small amount of automatic transcription texts into the LM training corpus (only 3M of 639M words), we reported gains in recognition performance for a broadcast news recognition task. As has been previously reported, the experimental work conducted showed that unsupervised LM training is a much more challenging task than unsupervised AM training. Nonetheless, we observed additive gains with the joint use of unsupervised LM and unsupervised AM training. That being said, if some data have been automatically transcribed for unsupervised AM training, the additional cost of using the transcripts for LM training is negligible. Thus, using unsupervised LM might be worthy. The unsupervised LM framework was assessed for standard backoff $n$-gram and extended to neural network language models. Both methods led to similar relative gains in performance with respect to their baseline models. However, NNLMs outperform backoff LMs in absolute values. The gains obtained with both NNLM and backoff LMs were not complementary.

Inspired by common practices applied for language modeling, we proposed to weight the relevance of the different subsets of the training data via data weighting and via GMM interpolation. The theoretical approaches proposed were empirically validate for two tasks: European Portuguese broadcast data recognition and multiple-accented English data recognition.

Data weighting was shown to be a type of interpolation technique, so as GMM interpolation (Chapter 7). The proposed **interpolation approaches** were theoretically compared with adaptive training approaches. It was shown that the re-estimation equations of the data weighting and the speaker adaptive training (Anastasakos et al., 1996) methods have some similarities. In the same way, GMM interpolation and cluster adaptive training (Gales, 1998a) have similarities in the sense that both make use of scalar coefficients to weigh multiple component models. However, while data weighting and interpolation methods aim to build acoustic models for a specific task, adaptive training methods attempt to reduce the speech variability among the training speakers. In addition, we argued that GMM interpolation is a flexible way to generate acoustic models for a target data given that only a few interpolation coefficients need to be re-estimated for each target task or when new data sets are incorporated into the acoustic training corpus.

Data weighting and GMM interpolation were reported to outperform baseline pooled models and MAP adapted models when applied to a European Portuguese broadcast data recognition task (Chapter 8). It was furthermore shown that near optimal interpolation coefficients can be estimated on a small held-out data set using a fully automatic EM-based approach, even if the estimation relies on untranscribed held-out data. These results were confirmed with the multiple-accented English data recognition system by using the GMM interpolation method. With this latter system, low-resourced data conditions were additionally assessed. When only about 2 to 3 hours of target accent data were available for training, interpolation outperformed the baseline pooled models (accent-independent) for the six test accents and the MAP adapted models for three of the accents. For a more extreme condition, when no data were available neither for training nor for adaptation, interpolation has proven to be especially useful. In this case, on-the-fly interpolation led to substantial gains in performance over the pooled (accent-independent) models.

GMM interpolation engendered an increase of model parameters, which was compensated by using a theoretically motivated mixture reduction algorithm. The solution, based on a constrained version of the standard maximum likelihood estimation, outperformed other reduction

algorithms based on greedy merge (Runnalls, 2007) and Gaussian component clustering (Davis and Dhillon, 2007)

## Perspectives

The main goal of this work was to reduce the cost required for ASR development. Given the central role played by acoustic and language models in such systems, the focus of this work was on reducing the human effort required to build these two models. Costs of other development steps might be reduced as well. For instance, instead of relying on expert's knowledge about the target language, unsupervised training methods could be used to automatically discover the acoustic modeling units (Bacchiani and Ostendorf, 1999; Varadarajan et al., 2008; Jansen and Church, 2011). Similarly, the pronunciation lexicon can also be created with low (or no) knowledge about the language (Bacchiani and Ostendorf, 1999; Bisani and Ney, 2008). Investigating the combined use of unsupervised approaches to define acoustic modeling units and generate the pronunciation lexicon (Hartmann et al., 2013) together with the techniques proposed in this thesis is a possible research direction in order to allow the development of accurate ASR systems in a fully unsupervised and low-cost manner.

The unsupervised training methods were assessed for HMM/GMM based acoustic models. Given the recent risen in popularity of hybrid HMM/DNN acoustic models, extending the techniques proposed in this work to such models is another suitable research direction. Unsupervised training of neural networks has already been reported in this dissertation for bottleneck MLPs (Chapter 5) and for (SOUL) NNLMs (Chapter 6). Laurent et al. (2014) reported experiments in unsupervised DNN training as well. Yet, the use of confidence-based filtering and weighting techniques as well as multiple decoding hypotheses (rather than the best one) is to be empirically evaluated. Filtering can be performed straightforwardly by removing the low probability training segments. Similarly, weighting could be performed by adjusting the learning rates according to the confidence scores. Such a weighting technique is also to be explored as an extension to our reported work in unsupervised bottleneck MLP and NNLM training.

A recurrent reported issue with work on DNNs is the difficulty to adapt the neural network to new tasks. Well-known techniques that have been applied for several years for HMM/GMM based acoustic model adaptation do not achieve the same results for DNN based models. Looking for rapid and efficient DNN adaptation techniques is a current relevant research axis given the computational complexity required for training such models. Model combination could be investigated as a possible solution to this issue. Combining neural network layers or the posterior probability scores are straightforward approaches. However, combining several layers in an efficient way to maintain model correctness and model tractability is still a challenge.

On a short term basis, the methods proposed in this work could be enhanced and extended to new applications. In particular, acoustic model interpolation was performed with a fixed model structure and phone set inventory. To extend the method and allow arbitrary models to be merged, it would be necessary to redefine the model structure after interpolation. A possible way to perform this task is to re-cluster the HMM states based on similarity measures taken over the state output mixtures. Clustering tree specialization (Schultz and Waibel, 2000) can also be envisaged to attain this goal.

The proposed acoustic model interpolation methods could be extended to address other tasks thereby validating the conclusions obtained in this work. As already mentioned, when applied to create source-specific and accent-specific acoustic models, interpolation outperforms other methods (data pooling and MAP adaptation). Due to the flexibility it provides, acoustic model interpolation seems to be particularly interesting for adapting ASR systems over time. Rather than retraining the entire model when new training data are delivered, only the new data

could be used to estimate a new component model, which could be combined straightforwardly to the previous model. Another possible application for AM interpolation is to combine manually transcribed and automatically transcribed acoustic data sets. Adaptation methods (like MAP adaptation) are commonly used to do this. Nevertheless, acoustic model interpolation may also outperform MAP adaptation in this case.

# Publications by the author


Fraga-Silva, T., Gauvain, J.-L., and Lamel, L. (2011). Lattice-based unsupervised acoustic model training. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4656–4659.

Fraga-Silva, T., Gauvain, J.-L., and Lamel, L. (2013). Interpolation of acoustic models for speech recognition. In *Proc. Interspeech*, pages 3347–3351.

Fraga-Silva, T., Gauvain, J.-L., and Lamel, L. (2014). Speech recognition of multiple accented English data using acoustic model interpolation. In *Proc. European Signal Processing Conference*.

Fraga-Silva, T., Le, V.-B., Lamel, L., and Gauvain, J.-L. (2012). Incorporating MLP features in the unsupervised training process. In *Proc. International Workshop on Spoken Language Technologies for Under-resourced Languages (SLTU)*.

Lamel, L., Courcinous, S., Despres, J., Fraga-Silva, T., Gauvain, J.-L., Josse, Y., Kilgour, K., Kraft, F., Le, V. B., Ney, H., Nußbaum-Thom, M., Oparin, I., Schlippe, T., Schlüter, R., Schultz, T., Stüker, S., Sundermeyer, M., Vieru, B., and Vu, N. (2011). Speech recognition for machine translation in Quaero. In *IWSLT*, pages 121–128.

Roy, A., Lamel, L., Fraga-Silva, T., Gauvain, J.-L., and Oparin, I. (2013). Some issues affecting the transcription of Hungarian broadcast audio. In *Proc. Interspeech*, pages 3102–3106.

# Bibliography


Abad, A., Trancoso, I., Neto, N., and Viana, M. C. (2009). Porting an European Portuguese broadcast news recognition system to Brazilian Portuguese. In *Proc. Interspeech*, pages 92–95. 33, 41

Adda, G. and Adda-Decker, M. (1997). Normalisation de textes en français : une étude quantitative pour la reconnaissance de la parole. In *Journées Scientifiques et Techniques du Réseau Francophone d'Ingénierie de la Langue de l'AUPELF-UREF*, pages 289–296. 32

Adda, G., Adda-Decker, M., Gauvain, J. L., and Lamel, L. (1997). Text normalization and speech recognition in French. In *Proceedings of the European Conference on Speech Technology*, volume 5, pages 2711–2714. 32

Allauzen, A. (2007). Error detection in confusion network. In *Proc. Interspeech*, pages 1749–1752. 30

Anastasakos, T., McDonough, J., Schwartz, R., and Makhoul, J. (1996). A compact model for speaker-adaptive training. In *Proc. International Conference on Spoken Language (ICSLP)*, volume 2, pages 1137–1140. 6, 44, 106, 109, 110, 117, 126, 134

Ardeshiri, T., Orguner, U., Lundquist, C., and Schon, T. B. (2012). On mixture reduction for multiple target tracking. In *Proc. International Conference on Information Fusion*, pages 692–699. 112

Bacchiani, M. and Ostendorf, M. (1999). Joint lexicon, acoustic unit inventory and model design. *Speech Communication*, 29(2):99–114. 135

Bacchiani, M. and Roark, B. (2003). Unsupervised language model adaptation. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume I, pages 224–227. 5, 6, 8, 50, 85, 97, 101

Bahl, L., Brown, P., De Souza, P., and Mercer, R. (1986). Maximum mutual information estimation of hidden Markov model parameters for speech recognition. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 11, pages 49–52. 16, 44, 126

Bahl, L. R., Brown, P., de Souza, P. V., and Mercer, R. (1989). A tree-based statistical language model for natural language speech recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 37(7):1001–1008. 25

Bahl, L. R., de Souza, P. V., Gopalakrishnan, P., Nahamoo, D., and Picheny, M. (1991). Context dependent modeling of phones in continuous speech using decision trees. In *Proc. DARPA Speech and Natural Language Processing Workshop*, pages 264–270. 19

Bahl, L. R., Jelinek, F., and Mercer, R. (1983). A maximum likelihood approach to continuous speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2:179–190. 21, 52

Baker, J. (1975). The DRAGON system – an overview. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 23(1):24–29. 11

Barras, C., Geoffrois, E., Wu, Z., and Liberman, M. (2001). Transcriber: development and use of a tool for assisting speech corpora production. *Speech Communication*, 33(1):5–22. 49

Baum, L. E., Petrie, T., Soules, G., and Weiss, N. (1970). A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *The Annals of Mathematical Statistics*, 41(1):164–171. 11, 12, 14, 28, 88

Bellegarda, J. R. (1997). A latent semantic analysis framework for large-span language modeling. In *European Conference on Speech Communication Technology*, pages 1451–1454. 26

Bellegarda, J. R. (2000). Exploiting latent semantic information in statistical language modeling. *Proceedings of the IEEE*, 88(8):1279–1296. 26

Bellegarda, J. R. and Nahamoo, D. (1990). Tied mixture continuous parameter modeling for speech recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 38(12):2033–2045. 18

Bengio, Y., Ducharme, R., and Vincent, P. (2000). A neural probabilistic language model. In *Advances in Neural Information Processing Systems (NIPS)*, volume 13, pages 932–938. MIT Press. 26, 93

Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. (2003). A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155. 26, 93, 95

Benzeghiba, M. et al. (2007). Automatic speech recognition and speech variability: a review. *Speech Communication*, 49(10):763–786. 104, 119

Berger, A. L., Pietra, V. J. D., and Pietra, S. A. D. (1996). A maximum entropy approach to natural language processing. *Computational linguistics*, 22(1):39–71. 25

Bilmes, J. A. and Kirchhoff, K. (2003). Factored language models and generalized parallel backoff. In *Proc. Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (HLT-NAACL)*, volume 2, pages 4–6. 25

Bisani, M. and Ney, H. (2008). Joint-sequence models for grapheme-to-phoneme conversion. *Speech Communication*, 50(5):434–451. 91, 135

Bishop, C. M. and Lasserre, J. (2007). Generative or discriminative? Getting the best of both worlds. In *Bayesian Statistics*, volume 8, pages 3–24. Oxford University Press. 12

Blackman, S. S. (2004). Multiple hypothesis tracking for multiple target tracking. *Aerospace and Electronic Systems Magazine, IEEE*, 19(1):5–18. 112, 114

Braga, D., Coelho, L., and Vianna Resende, F. (2006). A rule-based grapheme-to-phone converter for TTS systems in European Portuguese. In *International Telecommunications Symposium*, pages 328–333. IEEE. 36

Brasil (2012). Decreto no. 7.875, de 27 de dezembro de 2012. Brasília, Presidência da República, Casa Civil, Subchefia para Assuntos Jurídicos. Available at http://www.planalto.gov.br/ccivil_03/_Ato2011-2014/2012/Decreto/D7875.htm. Last accessed on Jun 16, 2014. 33

Brown, P. F., Cocke, J., Pietra, S. A. D., Pietra, V. J. D., Jelinek, F., Lafferty, J. D., Mercer, R. L., and Roossin, P. S. (1990). A statistical approach to machine translation. *Computational linguistics*, 16(2):79–85. 21

Brown, P. F., Desouza, P. V., Mercer, R. L., Pietra, V. J. D., and Lai, J. C. (1992). Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479. 25

Bulyko, I., Ostendorf, M., Siu, M., Ng, T., Stolcke, A., and Çetin, Ö. (2007). Web resources for language modeling in conversational speech recognition. *ACM Transactions on Speech and Language Processing*, 5(1):1–25. 5, 50

Campbell, W. M., Richardson, F., and Reynolds, D. A. (2007). Language recognition with word lattices and support vector machines. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume IV, pages 989–992. 63

Caseiro, D., Trancoso, L., Oliveira, L., and Viana, C. (2002). Grapheme-to-phone using finite-state transducers. In *Proc. of the IEEE Workshop on Speech Synthesis*, pages 215–218. IEEE. 36

Chen, H. D., Chang, K. C., and Smith, C. (2010). Constraint optimized weight adaptation for gaussian mixture reduction. In *SPIE 7697, Signal Processing, Sensor Fusion, and Target Recognition XIX*, page 76970N. 111

Chen, S. F. (2009). Shrinking exponential language models. In *Proc. Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (HLT-NAACL)*, pages 468–476. 25

Chen, S. F. and Goodman, J. (1996). An empirical study of smoothing techniques for language modeling. In *Proc. 34th annual meeting on Association for Computational Linguistics*, pages 310–318. 86

Chen, S. F. and Goodman, J. (1999). An empirical study of smoothing techniques for language modeling. *Computer Speech and Language*, 13:359–393. 23, 24, 32, 86

Chen, T., Huang, C., Chang, E., and Wang, J. (2001). Automatic accent identification using Gaussian mixture models. In *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 343–346. 44, 125

Cohen, P. S. and Mercer, R. L. (1975). The phonological component of an automatic speech recognition system. *Speech recognition*, pages 275–320. 9

CPLP (1990). Acordo ortográfico. Available at http://www.cplp.org/id-176.aspx. Last accessed on Jun 16, 2014. 33, 41

Crouse, D. F., Willett, P., Pattipati, K., and Svensson, L. (2011). A look at gaussian mixture reduction algorithms. In *Proc. International Conference on Information Fusion*. 112

Cruz-Ferreira, M. (1995). European Portuguese. *Journal of the International Phonetic Association*, 25(2):90–94. 37

Davis, J. V. and Dhillon, I. (2007). Differential entropic clustering of multivariate Gaussians. In *Advances in Neural Information Processing Systems*, volume 19, pages 337–344. MIT Press. 113, 114, 117, 119, 120, 129, 135

Davis, S. and Mermelstein, P. (1980). Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 28(4):357–366. 19

Della Pietra, S., Della Pietra, V., Mercer, R. L., and Roukos, S. (1992). Adaptive language modeling using minimum discriminant estimation. In *Proc. ACL Workshop on Speech and Natural Language*, pages 103–106. 25

DeMori, R. and Federico, M. (1999). Language model adaptation. *NATO ASI series. Series F: Computer and System Sciences*, pages 280–303. 85, 111

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the al Statistical Society. Series B (Methodological)*, 39:1–38. 7, 13, 51, 52, 104, 105, 116

Digalakis, V. V., Rtischev, D., and Neumeyer, L. G. (1995). Speaker adaptation using constrained estimation of Gaussian mixtures. *IEEE Transactions on Speech and Audio Processing*, 3(5):357–366. 106

Ellis, D. P., Singh, R., and Sivadas, S. (2001). Tandem acoustic modeling in large-vocabulary recognition. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 1, pages 517–520. 19, 49, 73, 75

Federico, M. (1999). Efficient language model adaptation through MDI estimation. In *European Conference on Speech Communication Technology*, pages 1583–1586. 6

Fischer, V., Janke, E., Kunzmann, S., and Ross, T. (2001). Multilingual acoustic models for the recognition of non-native speech. In *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 331–334. 103, 125

Fousek, P., Lamel, L., and Gauvain, J.-L. (2008a). On the use of MLP features for broadcast news transcription. *Text, Speech and Dialogue, Lecture Notes in Computer Science*, 5246:303–310. 19, 50, 73, 75

Fousek, P., Lamel, L., and Gauvain, J.-L. (2008b). Transcribing broadcast data using MLP features. In *Proc. Interspeech*, pages 1433–1436. 8, 19, 73, 75

Fraga-Silva, T., Gauvain, J.-L., and Lamel, L. (2011). Lattice-based unsupervised acoustic model training. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4656–4659. 50

Fraga-Silva, T., Gauvain, J.-L., and Lamel, L. (2013). Interpolation of acoustic models for speech recognition. In *Proc. Interspeech*, pages 3347–3351. 102

Fraga-Silva, T., Gauvain, J.-L., and Lamel, L. (2014). Speech recognition of multiple accented English data using acoustic model interpolation. In *Proc. European Signal Processing Conference*. 102

Fraga-Silva, T., Le, V.-B., Lamel, L., and Gauvain, J.-L. (2012). Incorporating MLP features in the unsupervised training process. In *Proc. International Workshop on Spoken Language Technologies for Under-resourced Languages (SLTU)*. 50

Gales, M. (2001). Multiple-cluster adaptive training schemes. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 1, pages 361–364. 107

Gales, M. J. (1998a). Cluster adaptive training for speech recognition. In *Proc. International Conference on Spoken Language (ICSLP)*, volume 1998, pages 1783–1786. 6, 106, 108, 117, 134

Gales, M. J. (1998b). Maximum likelihood linear transformations for HMM-based speech recognition. *Computer speech and language*, 12(2):75–98. 106, 107, 110

Gauvain, J.-L., Lamel, L., and Adda, G. (1998). Partitioning and transcription of broadcast news data. In *Proc. International Conference on Spoken Language (ICSLP)*, volume 5, pages 1335–1338. 27, 32

Gauvain, J.-L., Lamel, L., and Adda, G. (2002). The LIMSI broadcast news transcription system. *Speech Communication*, 37:89–108. 31, 43

Gauvain, J.-L., Lamel, L., Schwenk, H., Adda, G., Chen, L., and Lefevre, F. (2003). Conversational telephone speech recognition. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages I–212–215. 103

Gauvain, J.-L. and Lee, C. H. (1994). Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains. *IEEE Transactions on Speech and Audio Processing*, 2(2):291–298. 6, 15, 16, 44, 101, 103, 105, 125

Gauvain, J. L., Messaoudi, A., and Schwenk, H. (2004). Language recognition using phone lattices. In *Proc. International Conference on Spoken Language (ICSLP)*, pages 1283–1286. 51

Gollan, C. and Bacchiani, M. (2008). Confidence scores for acoustic model adaptation. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4289–4292. 30

Gollan, C., Hahn, S., Schlueter, R., and Ney, H. (2007). An improved method for unsupervised training of LVCSR systems. In *Proc. Interspeech*, pages 2101–2104. 5, 30, 49, 58

Goodman, J. T. (2001). A bit of progress in language modeling. *Computer Speech and Language*, 15:403–434. 21, 23

Gopalakrishnan, P. S., Kanevsky, D., Nadas, A., and Nahamoo, D. (1991). An inequality for rational functions with applications to some statistical estimation problems. *IEEE transactions on information theory*, 37(1):107–113. 17

Grezl, F., Karafiát, M., and Janda, M. (2011). Study of probabilistic and bottle-neck features in multilingual environment. In *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 359–364. IEEE. 20, 73

Grézl, F., Karafiát, M., Kontár, S., and Cernocky, J. (2007). Probabilistic and bottle-neck features for LVCSR of meetings. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 4, pages 757–761. 8, 20, 75, 76

Grézl, F., Karafiát, M., and Veselỳ, K. (2014). Adaptation of multilingual stacked bottle-neck neural network structure for new language. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7704–7708. 73

Hain, T., Johnson, S., Tuerk, A., Woodland, P., and Young, S. (1998). Segment generation and clustering in the HTK broadcast news transcription system. In *DARPA Broadcast News Transcription and Understanding Workshop*, pages 133–137. 27

Hanebeck, U. D., Briechle, K., and Rauh, A. (2003). Progressive bayes: a new framework for nonlinear state estimation. In *SPIE 5099, Multisensor, Multisource Information Fusion: Architectures, Algorithms, and Applications*, pages 256–267. 113, 114

Hartmann, W., Roy, A., Lamel, L., and Gauvain, J.-L. (2013). Acoustic unit discovery and pronunciation generation from a grapheme-based lexicon. In *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 380–385. IEEE. 135

Hermansky, H. (1990). Perceptual linear prediction (PLP) analysis for speech. *Journal of the Acoustical Society of America*, 87:1738–1752. 19, 31, 49, 73, 126

Hermansky, H., Ellis, D. P., and Sharma, S. (2000). Tandem connectionist feature extraction for conventional HMM systems. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 3, pages 1635–1638. 20, 75

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A.-r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N., et al. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97. 21, 75

Huang, C., Chen, T., and Chang, E. (2004). Accent issues in large vocabulary continuous speech recognition. *International Journal of Speech Technology*, 7(2-3):141–153. 125

Huang, C., Chen, T., Li, S. Z., Chang, E., and Zhou, J.-L. (2001). Analysis of speaker variability. In *European Conference on Speech Communication Technology*, pages 1377–1380. 125

Huang, X., Alleva, F., Hon, H.-W., Hwang, M.-Y., Lee, K.-F., and Rosenfeld, R. (1993). The SPHINX-II speech recognition system: an overview. *Computer Speech and Language*, 7(2):137–148. 25

Huang, X. and Jack, M. (1989). Semi-continuous hidden Markov models for speech signals. *Computer Speech and Language*, 3(3):239–251. 18

Huber, M. F. and Hanebeck, U. D. (2008). Progressive gaussian mixture reduction. In *Proc. International Conference on Information Fusion*, pages 1–8. 113, 114

Hull, J. (1992). Combining syntactic knowledge and visual text recognition: a hidden Markov model for part of speech tagging in a word recognition algorithm. In *AAAI Symposium: Probabilistic Approaches to Natural Language*, pages 77–83. 21

Infopedia (2009). Dicionário Português-Espanhol. Available at http://www.infopedia.pt. Last accessed on Jan 22, 2010. 39, 41

Instituto de Linguística Teórica e Computacional (2007). Portal da língua portuguesa. Available at http://www.portaldalinguaportuguesa.org. Last accessed on Jun 17, 2014. 37

Jansen, A. and Church, K. (2011). Towards unsupervised training of speaker independent acoustic models. In *Proc. Interspeech*, pages 1693–1692. 135

Jelinek, F. (1976). Speech recognition by statistical methods. *Proceedings of the IEEE*, 64:532–556. 11

Jelinek, F. (1985). The development of an experimental discrete dictation recognizer. *Proceedings of the IEEE*, 73(11):1616–1624. 21

Jiang, H. (2005). Confidence measures for speech recognition: A survey. *Speech communication*, 45(4):455–470. 30

Jonhson, D. (2004). Quicknet, speech group at ICSI, berkeley. Available at http://www1.icsi.berkeley.edu/Speech/qn.html. Last accessed on Jun 30, 2014. 77

Juang, B. H. (1985). Maximum-likelihood estimation for mixture multivariate stochastic observations of Markov chains. *AT&T Technical Journal*, 64(6). 14, 103

Juang, B.-H., Hou, W., and Lee, C.-H. (1997). Minimum classification error rate methods for speech recognition. *IEEE Transactions on Speech and Audio Processing*, 5(3):257–265. 16

Juang, B.-H. and Katagiri, S. (1992). Discriminative learning for minimum error classification. *IEEE Transactions on Signal Processing*, 40(12):3043–3054. 16

Karanasou, P. (2013). *Phonemic variability and confusability in pronunciation modeling for automatic speech recognition*. PhD thesis, Université Paris-Sud. 36

Katz, S. (1987). Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 35(3):400–401. 23

Kemp, T. and Waibel, A. (1999). Unsupervised training of a speech recognizer: recent experiments. In *European Conference on Speech Communication Technology*, pages 2725–2728. 5, 49, 51, 57, 58

Kneser, R. and Ney, H. (1995). Improved backing-off for m-gram language modeling. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 1, pages 181–184. 23, 24, 85, 89

Kneser, R., Peters, J., and Klakow, D. (1997). Language model adaptation using dynamic marginals. In *European Conference on Speech Communication Technology*, pages 1971–1974. 6, 91

Knill, K. and Young, S. (1997). Hidden Markov models in speech and language processing. In *Corpus-based methods in Language and Speech processing*, pages 27–68. Springer. 18, 28

Kubota, Y., Shinozaki, T., and Furui, S. (2010). Investigations on ensemble based unsupervised adaptation methods. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4874–4877. 67

Kuhn, R., Nguyen, P., Junqua, J.-C., Goldwasser, L., Niedzielski, N., Fincke, S., Field, K., and Contolini, M. (1998). Eigenvoices for speaker adaptation. In *Proc. International Conference on Spoken Language (ICSLP)*, volume 98, pages 1774–1777. 6, 106, 108, 117

Kullback, S. and Leibler, R. A. (1951). On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86. 112

Lamel, L., Gauvain, J.-L., and Adda, G. (2000). Lightly supervised acoustic model training. In *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 150–154. 5, 56

Lamel, L., Gauvain, J.-L., and Adda, G. (2002a). Lightly supervised and unsupervised acoustic model training. *Computer Speech and Language*, 16:115–129. 54

Lamel, L., Gauvain, J.-L., and Adda, G. (2002b). Unsupervised acoustic model training. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 1, pages I–877. IEEE. 51, 55, 68, 71

Lamel, L., Gauvain, J.-L., Le, V. B., Oparin, I., and Meng, S. (2011). Improved models for Mandarin speech-to-text transcription. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4660–4663. 50, 73

Lamel, L. and Vieru, B. (2010). Development of a speech-to-text transcription system for Finnish. In *Proc. International Workshop on Spoken Language Technologies for Under-resourced Languages (SLTU)*, pages 62–67. 51

Laurent, A., Hartmann, W., and Lamel, L. (2014). Unsupervised acoustic model training for the Korean language. In *Proc. International Conference on Spoken Language (ICSLP)*, Singapoure. 75, 83, 135

Le, H.-S., Oparin, I., Allauzen, A., Gauvain, J., and Yvon, F. (2011). Structured output layer neural network language model. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5524–5527. 27, 86, 93, 94, 95, 97

Le, H.-S., Oparin, I., Allauzen, A., Gauvain, J., and Yvon, F. (2013). Structured output layer neural network language models for speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(1):197–206. 27, 93, 94

Le, V., Lamel, L., and Gauvain, J.-L. (2010). Multi-style MLP features for BN transcription. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4866–4869. 81, 82

Lee, K. F. and Hon, H. W. (1988). Large-vocabulary speaker-independent continuous speech recognition using HMM. In *Proc. IEEE International Conference of Acoustics, Speech, Signal Processing (ICASSP)*, pages 123–126. 18

Lee, L. and Rose, R. C. (1996). Speaker normalization using efficient frequency warping procedures. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 1, pages 353–356. IEEE. 104

Leggetter, C. J. and Woodland, P. C. (1995). Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models. *Computer speech and language*, 9(2):171. 6, 33, 101, 103, 105, 126

Li, X., Singh, R., and Stern, R. M. (2002). Lattice combination for improved speech recognition. In *Proc. International Conference on Spoken Language (ICSLP)*, pages 405–408. 51

Ma, J., Matsoukas, S., Kimball, O., and Schwartz, R. (2006). Unsupervised training on large amounts of broadcast news data. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume III, pages 1056–1059. 55, 68, 71

Mangu, L., Brill, E., and Stolcke, A. (1999). Finding consensus among words: lattice-based word error minimization. In *Proc. European Conference on Speech Communication Technology*, pages 495–498. 28, 33

Mateus, M. H. and d'Andrade, E. (2000). *The phonology of Portuguese*. Oxford University Press. 33

Mikolov, T., Deoras, A., Kombrink, S., and Burget, L. (2011). Empirical evaluation and combination of advanced language modeling techniques. In *Proc. Interspeech*, pages 605–608. 27

Mikolov, T., Karafiát, M., Burget, L., Cernockỳ, J., and Khudanpur, S. (2010). Recurrent neural network based language model. In *Proc. Interspeech*, pages 1045–1048. 27

Mohamed, A.-r., Dahl, G. E., and Hinton, G. (2012). Acoustic modeling using deep belief networks. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 20(1):14–22. 21, 75

Mohamed, A.-r., Sainath, T. N., Dahl, G., Ramabhadran, B., Hinton, G. E., and Picheny, M. A. (2011). Deep belief networks using discriminative features for phone recognition. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5060–5063. IEEE. 21, 75

Morgan, N. and Bourlard, H. (1995). An introduction to hybrid HMM/connectionist continuous speech recognition. In *IEEE Signal Processing Magazine*, page IEEE Signal Processing Magazine. 21, 75

Ney, H. and Essen, U. (1991). On smoothing techniques for bigram-based natural language modelling. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 825–828. 23

Ney, H., Essen, U., and Kneser, R. (1994). On structuring probabilistic dependences in stochastic language modelling. *Computer Speech and Language*, 8(1):1–38. 23, 24

Ney, H., Martin, S., and Wessel, F. (1997). Statistical language modeling using leaving-one-out. In *Corpus-based methods in Language and Speech processing*, pages 174–207. Springer. 23, 87, 89

Novotney, S., Schwartz, R., and Ma, J. (2009). Unsupervised acoustic and language model training with small amounts of labelled data. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4297–4300. 5, 8, 50, 71, 85, 86, 87, 92, 97

Oh, Y., Yoon, J., and Kim, H. (2007). Acoustic model adaptation based on pronunciation variability analysis for non-native speech recognition. *Speech Communication*, 49(1):59–70. 103

Padmanabhan, M., Saon, G., and Zweig, G. (2000). Lattice-based unsupervised MLLR for speaker adaptation. In *ISCA ITRW Automatic Speech Recognition: Challenges for the Millenium*, pages 128–131. 30, 51

Pallett, D. S., Fisher, W. M., and Fiscus, J. G. (1990). Tools for the analysis of benchmark speech recognition tests. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 97–100. 61, 92, 127

Pitz, M., Wessel, F., and Ney, H. (2000). Improved MLLR speaker adaptation using confidence measures for conversational speech recognition. In *Proc. Interspeech*, pages 548–551. 30

Povey, D., Kingsbury, B., Mangu, L., Saon, G., Soltau, H., and Zweig, G. (2005). fMPE: Discriminatively trained features for speech recognition. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 1, pages 961–964. 19

Povey, D. and Woodland, P. C. (2002). Minimum phone error and I-smoothing for improved discriminative training. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume I, pages 105–108. 16

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (2007). *Numerical Recipes: The Art of Scientific Computing*. Cambridge University Press, New York, 3rd edition. 91

Rojas, R. (1996). *Neural networks: a systematic introduction*. Springer. 74

Rosenfeld, R. (1996). A maximum entropy approach to adaptive statistical language modelling. *Computer Speech and Language*, 10(3):187–228. 25

Rosenfeld, R. (2000). Two decades of statistical language modeling: where do we go from here? *Proceedings of the IEEE*, 88(8):1270–1278. 21, 22, 25

Roy, A., Lamel, L., Fraga-Silva, T., Gauvain, J.-L., and Oparin, I. (2013). Some issues affecting the transcription of Hungarian broadcast audio. In *Proc. Interspeech*, pages 3102–3106. 50, 81

Runnalls, A. R. (2007). Kullback-leibler approach to gaussian mixture reduction. *IEEE Transactions on Aerospace and Electronic Systems*, 43(3):989–999. 113, 114, 119, 120, 129, 135

Salmond, D. J. (1990). Mixture reduction algorithms for target tracking in clutter. In *SPIE signal and data processing of small targets*, volume 1305, pages 434–445. 113, 114

Salmond, D. J. (2009). Mixture reduction algorithms for point and extended object tracking in clutter. *IEEE Transactions on Aerospace and Electronic Systems*, 45(2):667–686. 112

Sarikaya, R., Afify, M., and Kingsbury, B. (2009). Tied-mixture language modeling in continuous space. In *Proc. Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (HLT-NAACL)*, pages 459–467. 26

Schieferdecker, D. and Huber, M. F. (2009). Gaussian mixture reduction via clustering. In *Proc. International Conference on Information Fusion*, pages 1536–1543. 114, 120

Schlüter, R., Müller, B., Wessel, F., and Ney, H. (1999). Interdependence of language models and discriminative training. In *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 119–122. 17

Schultz, T. and Waibel, A. (2000). Polyphone decision tree specialization for language adaptation. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 3, pages 1707–1710. IEEE. 117, 135

Schultz, T. and Waibel, A. (2001). Experiments on cross-language acoustic modeling. In *Proc. Interspeech*, pages 2721–2724. 54

Schwarz, P., Matjka, P., and Cernocky, J. (2004). Towards lower error rates in phoneme recognition. In *Proc. International Conference on Text, Speech and Dialogue*, pages 465–472. 76

Schwenk, H. (2007). Continuous space language models. *Computer Speech and Language*, 21(3):492–518. 26, 93

Schwenk, H. and Gauvain, J.-L. (2002). Connectionist language modeling for large vocabulary continuous speech recognition. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 1, pages 765–768. 26, 93

Scott, D. W. and Szewczyk, W. F. (2001). From kernels to mixtures. *Technometrics*, 43(3):323–335. 113

Shinozaki, T., Kubota, Y., and Furui, S. (2009). Unsupervised cross-validation adaptation algorithms for improved adaptation performance. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4377–4380. 67

Silva, E., Baptista, L., Fernandes, H., and Klautau, A. (2005). Desenvolvimento de um sistema de reconhecimento automático de voz contínua com grande vocabulário para o Português Brasileiro. In *XXV Congresso da Sociedade Brasileira de Computação*, pages 2258–2267. 33

Srihari, R. and Baltus, C. (1992). Combining statistical and syntactic methods in recognizing handwritten sentences. In *AAAI Symposium: Probabilistic Approaches to Natural Language*, pages 121–127. 21

Stolcke, A., Grézl, F., Hwang, M.-Y., Lei, X., Morgan, N., and Vergyri, D. (2006). Cross-domain and cross-language portability of acoustic features estimated by multilayer perceptrons. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 1, pages 321–324. 8, 20, 50, 73, 81, 82

Tachibana, M., Yamagishi, J., Onishi, K., Masuko, T., and Kobayashi, T. (2004). HMM-based speech synthesis with various speaking styles using model interpolation. In *Speech Prosody 2004, International Conference*. 104

Takahashi, S. and Sagayama, S. (1995). Four-level tied-structure for efficient representation of acoustic modeling. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 1, pages 520–523. 18

Tan, T. P. and Besacier, L. (2007). Acoustic model interpolation for non-native speech recognition. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 4, pages IV–1009. 104

Thomas, S., Ganapathy, S., and Hermansky, H. (2012). Multilingual MLP features for low-resource LVCSR systems. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4269–4272. IEEE. 73

Tóth, L., Frankel, J., Gosztolya, G., and King, S. (2008). Cross-lingual portability of MLP-based tandem features – A case study for English and Hungarian. In *Proc. Interspeech*, pages 2695–2698. 20, 50, 73, 81

Valtchev, V., Odell, J., Woodland, P., and Young, S. (1996). Lattice-based discriminative training for large vocabulary speech recognition. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 2, pages 605–608. 17

Valverde, G., Tortós, J. Q., and Terzija, V. (2012). Comparison of gaussian mixture reductions for probabilistic studies in power systems. In *Power and Energy Society General Meeting*, pages 1–7. 114

Varadarajan, B., Khudanpur, S., and Dupoux, E. (2008). Unsupervised learning of acoustic sub-word units. In *Proc. Conference of the Association for Computational Linguistics (ACL)*, pages 165–168. Association for Computational Linguistics. 135

Vergyri, D., Lamel, L., and Gauvain, J.-L. (2010). Automatic speech recognition of multiple accented English data. In *Proc. Interspeech*, pages 1652–1655. 31, 42, 43, 44, 45, 103, 125, 126, 130

Veselý, K., Karafiát, M., Grézl, F., Janda, M., and Egorova, E. (2012). The language-independent bottleneck features. In *Proc. IEEE Workshop on Spoken Language Technology (SLT)*, pages 336–341. IEEE. 50, 73

Viterbi, A. (1967). Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269. 12, 14, 28, 52

Vu, N. T., Metze, F., and Schultz, T. (2012). Multilingual bottleneck features and its application for under-resourced languages. In *Proc. International Workshop on Spoken Language Technologies for Under-resourced Languages (SLTU)*, pages 90–93. 20, 73, 82

Wang, L., Gales, M. J. F., and Woodland, P. C. (2007). Unsupervised training for Mandarin broadcast news and conversation transcriptions. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume IV, pages 353–356. 12, 51

Wang, Z., Schultz, T., and Waibel, A. (2003). Comparison of acoustic model adaptation techniques on non-native speech. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 1, pages I–540. 103, 104, 125

Wessel, F. and Ney, H. (2001). Unsupervised training of acoustic models for large vocabulary continuous speech recognition. In *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. 49, 51, 58

Wessel, F., Schluter, R., Macherey, K., and Ney, H. (2001). Confidence measures for large vocabulary continuous speech recognition. *IEEE Transactions on Speech and Audio Processing*, 9(3):288–298. 30, 57, 58

Wheatley, B., Kondo, K., Anderson, W., and Muthusamy, Y. (1994). An evaluation of cross-language adaptation for rapid HMM development in a new language. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume I, pages 237–240. 54

Williams, J. L. and Maybeck, P. S. (2003). Cost-function-based gaussian mixture reduction for target tracking. In *Proc. International Conference on Information Fusion*, volume 2, pages 1047–1054. 113, 114

Woodland, P. C. and Povey, D. (2002). Large scale discriminative training of hidden Markov models for speech recognition. *Computer Speech and Language*, 16:25–47. 16, 17

Xu, P. and Jelinek, F. (2004). Random forests in language modeling. In *ACL Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 325–332. 26

Yoshimura, T., Tokuda, K., Masuko, T., Kobayashi, T., and Kitamura, T. (1997). Speaker interpolation in HMM-based speech synthesis system. In *European Conference on Speech Communication Technology*, volume 97. 104

Young, S. (1994). Detecting misrecognitions and out-of-vocabulary words. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume II, pages 21–24. 30

Young, S. J., Odell, J., and Woodland, P. C. (1994). Tree-based state tying for high accuracy acoustic modelling. In *Proc. ACL Workshop on Human Language Technology*, pages 307–312. 19

Young, S. J. and Woodland, P. C. (1993). The use of state tying in continuous speech recognition. In *European Conference on Speech Communication Technology*, volume 3, pages 220–2206. 18

Zavaliagkos, G. and Colthurst, T. (1998). Utilizing untranscribed training data to improve performance. In *DARPA Broadcast News Transcription and Understanding Workshop*, pages 301–305. 5, 30, 40, 49, 51

Zavaliagkos, G., Schwartz, R., and McDonough, J. (1996). Maximum a posteriori adaptation for large scale HMM recognizers. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 2, pages 725–728. 103

Zhang, B., Matsoukas, S., and Schwartz, R. (2006). Discriminatively trained region dependent feature transforms for speech recognition. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 1, pages 313–316. 19

Zheng, Y., Sproat, R., Gu, L., Shafran, I., Zhou, H., Su, Y., Jurafsky, D., Starr, R., and Yoon, S.-Y. (2005). Accent detection and speech recognition for Shanghai-accented Mandarin. In *Proc. Interspeech*, pages 217–220. 125

Zhu, Q., Stolcke, A., Chen, B. Y., and Morgan, N. (2005). Using MLP features in SRI's conversational speech recognition system. In *Proc. Interspeech*, pages 2141–2144. 19, 50, 73, 75