Université du Québec
Institut national de la recherche scientifique (INRS)
Énergie, Matériaux et Télécommunications (EMT)

Language Modeling for Speech Recognition Incorporating Probabilistic Topic Models

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy

By
Md. Akmal Haidar

Evaluation Jury

External Examiner	John Hershey, Mitsubishi Electric Research Laboratories
External Examiner	Lidia Mangu, IBM Corporation
Internal Examiner	Tiago Falk, INRS-EMT
Research Director	Douglas O'Shaughnessy, INRS-EMT

I would like to dedicate this thesis to my loving parents ...

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other University. This dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration, except where specifically indicated in the text.

Md. Akmal Haidar
2014

Acknowledgements

I would like to express my special appreciation and thanks to my supervisor Professor Dr. Douglas O'Shaughnessy, you have been a tremendous mentor for me. I would like to thank you for encouraging my research and for allowing me to grow as a research scientist. Your wide knowledge and logical way of thinking have been of great value of me. Your understanding, encouraging and personal guidance have provided a good basis for this thesis. Your advice on both research as well as on my career have been priceless.

I would also like to thank my committee members, Dr. Lidia Mangu, Dr. John Hershey, Professor Dr. Tiago Falk for serving as my committee members even at hardship. I also want to thank you for your brilliant comments and suggestions, thanks to you. Moreover, I would like to thank all the staffs especially Madame Hélène Sabourin, Madame Nathalie Aguiar, Mr. Sylvain Fauvel of INRS-EMT, University of Quebec, Place Bonaventure, Montreal, Canada, for their support in the past years. Besides, thanks goes to Dr. Mohammed Senoussaoui for helping me to write the French summary of this thesis.

A special thanks to my parents. Words cannot express how grateful I am to you for all of the sacrifices that you've made on my behalf. Your prayer for me was what sustained me thus far. I would also like to thank my sisters, brothers and all of my friends who supported me to strive towards my goal.

Last but not the least, thanks to almighty, the most merciful and the most passionate, for providing me the opportunity to step in the excellent research world of speech and language technology.

Abstract

The use of language models in automatic speech recognition helps to find the best next word, to increase recognition accuracy. Statistical language models (LMs) are trained on a large collection of text to automatically determine the model's parameters. LMs encode linguistic knowledge in such a way that can be useful to process human language. Generally, a LM exploits the immediate past information only. Such models can capture short-length dependencies between words very well. However, in any language for communication, words have both semantic and syntactic importance. Most speech recognition systems are designed for a specific task and use language models that are trained from a large amount of text that is appropriate for this task. A task-specific language model will not do well for a different domain or topic. A perfect language model for speech recognition on general language is still far away. However, language models that are trained from a diverse style of language can do well, but are not perfectly suited for a certain domain. In this research, we introduce new language modeling approaches for automatic speech recognition (ASR) systems incorporating probabilistic topic models.

In the first part of the thesis, we propose three approaches for LM adaptation by clustering the background training information into different topics incorporating latent Dirichlet allocation (LDA). In the first approach, a hard-clustering method is applied into LDA training to form different topics. We propose an n-gram weighting technique to form an adapted model by combining the topic models. The models are then further modified by using latent semantic marginals (LSM) using a minimum discriminant information (MDI) technique. In the second approach, we introduce a clustering technique where the background n-grams are directed into different topics using a fraction of the global count of the n-grams. Here, the probabilities of the n-grams for different topics are multiplied by the global counts of the n-grams and are used as the counts of the respective topics. We also introduce a weighting technique that outperforms the n-gram weighting technique. In the third approach, we propose another clustering technique where the topic probabilities of the training documents are multiplied by the document-based n-gram counts and the products are summed up for all training documents; thereby the background n-grams are assigned into different topics.

In the second part of the thesis, we propose five topic modeling algorithms that are trained by using the expectation-maximization (EM) algorithm. A context-based probabilistic latent semantic analysis (CPLSA) model is proposed to overcome the limitation of a recently proposed unsmoothed bigram PLSA (UBPLSA) model. The CPLSA model can compute the correct topic probabilities of the unseen test document as it can compute all the bigram probabilities in the training phase, and thereby yields the proper bigram model for the unseen document. The CPLSA model is extended to a document-based CPLSA (DCPLSA) model where the document-based word probabilities for topics are trained. To propose the DCPLSA model, we are motivated by the fact that the words in different documents can be used to describe different topics. An interpolated latent Dirichlet language model (ILDLM) is proposed to incorporate long-range semantic information by interpolating distance-based *n*-grams into a recently proposed LDLM. Similar to the LDLM and ILDLM models, we propose enhanced PLSA (EPLSA) and interpolated EPLSA (IEPLSA) models in the PLSA framework.

In the final part of the thesis, we propose two new Dirichlet class language models that are trained by using the variational Bayesian EM (VB-EM) algorithm to incorporate long-range information into a recently proposed Dirichlet class language model (DCLM). The latent variable of DCLM represents the class information of an *n*-gram event rather than the topic in LDA. We introduce an interpolated DCLM (IDCLM) where the class information is exploited from (*n*-1) previous history words of the *n*-grams through Dirichlet distribution using interpolated distanced *n*-grams. A document-based DCLM (DDCLM) is proposed where the DCLM is trained for each document using document-based *n*-gram events.

In all the above approaches, the adapted models are interpolated with the background model to capture the local lexical regularities. We perform experiments using the '87-89 Wall Street Journal (WSJ) corpus incorporating a multi-pass continuous speech recognition (CSR) system. In the first pass, we use the background *n*-gram language model for lattice generation and then we apply the LM adaptation approaches for lattice rescoring in the second pass.

Supervisor: Douglas O'Shaughnessy, Ph.D.
Title: Professor and Program Director

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The primary means of communication between people is speech. It has been and will be the dominant mode of human social bonding and information exchange from human creation to the new media of future. Human beings have envisioned to communicate with machines via natural language long before the advancement of computers. Over the last few decades, research in automatic speech recognition has attracted a great deal of attention, which constitutes an important part in fulfilling this vision.

In our daily life, we may find many real applications of automatic speech recognition (ASR). For example, in most of the latest cellular phones, especially smartphones, ASR functions are available to do simple tasks such as dialing a phone number, writing a message, or to run an application using voice instead of typing. An automotive navigation system with ASR capability can be found inside cars, which let the driver to focus on driving while controlling the navigation system through voice. Besides, there are many applications of ASR systems that perform advanced tasks such as dictation. Those are just a few examples that describe how the ASRs bring a real value on the daily life [79].

Early stages of ASR systems were based on template-matching techniques. Template-matching refers to the incoming speech signal being compared to a set reference patterns, or templates. The first known template-based ASR system was developed by Davis et al. [25] in 1952. That was a very simple task, which was to recognize a digit (isolated words) from a speaker. Ever since, many small vocabulary (order of 10-100 words) ASR tasks were carried out by several researchers. In the 1970's, significant developments in ASR research began where the size of vocabulary was increasing to a medium size (100-1000 words), with continuous words and these methods were using a simple template-based pattern recognition. In 1980's, the vocabulary size of ASR was further increasing from medium to a large vocabulary size (> 1000 words) and the method was shifted from a template-based approach

to a statistical modeling framework, most notably the hidden Markov model (HMM) framework [31, 79]. In the large vocabulary ASR systems, potential confusion increases between similar sounding words. An ASR system concentrating on the grammar structure (language model) of the language was proposed by Jelinek et al. [62], where the language model was represented by statistical rules that can distinguish similar sounding words and can tell which sequence (phonemes or words) that is likely to appear in the speech.

Language modeling is the challenge to capture, characterize and exploit the regularities of natural language. It encodes the linguistic knowledge that is useful for computer systems when dealing with human language. Language modeling is critical to many applications that process human language with less than complete knowledge [28]. It is widely used in a variety of natural language processing tasks such as speech recognition [6, 57], handwritten recognition [74], machine translation [17], and information retrieval [86]. However, one of the most exciting applications of language models is in automatic speech recognition (ASR), where a computer is used to transcribe the spoken text into written form. An ASR system consists of two components: the acoustic model and the language model (LM). The LM combines with the acoustic model to reduce the acoustic search space and resolve the acoustic ambiguity. It not only helps to disambiguate among acoustically similar phrases such as (for, four, fore) and (recognize speech, wreck a nice beach), but also guides the search for the best acoustically matching word sequence towards ones with the highest language probabilities [55]. ASR systems cannot find the correct word sequence without a LM. They provide a natural way of communication between human and machine as if they were speaking as humans using natural language.

1.1 Background

There is a great deal of variability in natural language. Natural language grammar can be taught by two approaches: rule-based language models and statistical language models. In rule-based language models, grammar is defined as a set of rules that are accurate, but difficult to learn automatically. These rules are manually created by some experts such as linguists. In this approach, a sentence is accepted or rejected based on the set of rules defined in the grammar. This approach may be useful for a small task, where the rules for all possibilities of sentences can be defined. However, in natural language, there are more chances for the sentences to be ungrammatical. Statistical language models are useful to model natural language by creating several hypothesis for a sentence. They can model the language grammar by a set of parameters, which can learn automatically from a reasonable

amount of training data. Therefore, it can save more time as the parameters can be learned automatically.

The most powerful statistical models are the *n*-gram models [61, 63, 75]. These models exploit the short-range dependencies between words in a language very well. To learn its parameters (*n*-gram probabilities) from a given corpus, the *n*-gram models use maximum likelihood (ML) estimation. However, *n*-gram models suffer from the data sparseness problem as many *n*-grams do not appear in a training corpus, even with a large amount of training corpus. For example, if we use a vocabulary V of size 10,000 words in a trigram model, the total number of probabilities to be estimated is $|V|^3 = 10^{12}$. For any training data of manageable size, many of the probabilities will be zero. The data sparseness problem is also caused by the *n*-grams with low frequency of occurrences in a large training corpus that will have an unreliable probability.

To deal with the data sparseness problem, a smoothing technique is considered that ensures some probabilities (>0) to the words that do not appear (or appear with low frequency) in a training corpus. The idea of smoothing is to take out some probability mass from the seen events and distribute it to the unseen events. The method can be categorized depending on how the probability mass is taken out (discounting) and how it is redistributed (back-off). Examples of some smoothing techniques are additive smoothing [70], Good-Touring estimate [36], Jelinek-Mercer smoothing [61], Katz smoothing [64], Witten-Bell smoothing [9], absolute discounting [81], and Kneser-Ney smoothing [66]. The details of them can be found in [19].

Class-based *n*-gram LMs [16] have also been proposed to solve the data sparseness problem. Here, multiple words are grouped into a word class, and the transition probabilities between words are approximated by the probabilities between word classes. However, class-based *n*-grams work better only with a limited amount of training data with fewer parameters than in the word *n*-gram model [16]. The class-based *n*-gram LM is improved by interpolating with a word-based *n*-gram LM [15, 99]. A word-to-class backoff [83] was introduced where a class-based *n*-gram LM is used to predict unseen events, while the word-based *n*-gram LM is used to predict seen events. However, when the class-based and word-based LMs are used together, the parameter size increases more than the independent case, which is not good for low resource applications [80]. In [103], multi-dimensional word classes were introduced to improve the class-based *n*-grams. Here, the classes represent the left and right context Markovian dependencies separately. A back-off hierarchical class-based LM was introduced to model unseen events using the class models in various layers of a clustering tree [106]. In [18], a new class-based LM called *Model M* was introduced

by identifying back-off features that can improve test performance by reducing the size of a model. Here, the n-gram features were shrunken for n-grams that differ only in their histories. Unsupervised class-based language models such as Random Forest LM [102] have been investigated that outperform a word-based LM.

The neural network language model (NNLM) was also investigated to tackle the data sparseness problem by learning distributed representation of words [12, 90]. Here, the $(n-1)$ history words are first mapped into a continuous space and then the n-gram probabilities given the history words are estimated. Later, a recurrent neural network-based LM was investigated that shows better results than NNLM [76, 77].

1.1.1 Incorporating Long-range Dependencies

The improvement of n-gram models can fall into two categories, whether one is introducing a better smoothing method or incorporating long-range dependencies. Statistical n-gram LMs suffer from shortages of long-range information, which limit performance. They use the local context information by modeling text as a Markovian sequence and capture only the local dependencies between words. They cannot capture the long-range information of natural language. Several methods have been investigated to overcome this weakness. A cache-based language model is an earlier approach that is based on the idea that if a word appeared previously in a document it is more likely to occur again. It helps to increase the probability of previously observed words in a document when predicting a future word [69]. This idea is used to increase the probability of unobserved but topically related words, for example, trigger-based LM adaptation using a maximum entropy framework [88]. However, the training time requirements (finding related word pairs) of this approach are computationally expensive.

1.1.2 Incorporating Long-range Topic Dependencies

Language models perform well when the test environment matches nicely with the training environment. Otherwise, adaptation for the test set is essential because the smoothing approaches do not consider the issues such as topic and style mismatch between the training and test data. Actually, it is impossible to collect all forms of topics and styles of a language in the training set. So, in most cases of practical tasks, adaptation of a language model is required. Many approaches have been investigated to capture the topic related long-range dependencies. The first technique was introduced in [67] using a topic mixture model. Here, topic-specific language models are trained using different corpora with different top-

ics and combined in a linear way. Another well-known method is the sentence-level mixture models, which create topic clusters by using a hard-clustering method where a single topic is assigned to each document and used in LM adaptation. Improvements were shown both in perplexity and recognition accuracy over an unadapted trigram model [58].

Recently, various techniques such as Latent Semantic Analysis (LSA) [10, 26], Probabilistic LSA (PLSA) [33, 54], and Latent Dirichlet Allocation (LDA) [13] have been investigated to extract the latent topic information from a training corpus. All of these methods are based on a bag-of-words assumption, i.e., the word-order in a document can be ignored. These methods have been used successfully for speech recognition [10, 33, 39, 72, 73, 78, 80, 95, 96]. In LSA, a word-document matrix is used to extract the semantic information. In PLSA, each document is modeled by its own mixture weights and there is no generative model for these weights. So, the number of parameters grows linearly when increasing the number of documents, which leads to an overfitting problem. Also, there is no method to assign probability for a document outside the training set. On the contrary, the LDA model was introduced where a Dirichlet distribution is applied on the topic mixture weights corresponding to the documents in the corpus. Therefore, the number of model parameters is dependent only on the number of topic mixtures and the vocabulary size. Thus, LDA is less prone to overfitting and can be used to compute the probabilities of unobserved test documents. However, the LDA model can be viewed as a set of unigram latent topic models. The LDA model is one of the most widely used topic-modeling methods used in speech recognition that capture long-distance information through a mixture of unigram topic distributions. In the idea of an unsupervised language model adaptation approach, the unigram models extracted by LDA are adapted with proper mixture weights and interpolated with the n-gram baseline model [95]. To extend the unigram bag-of-words models to n-gram models, a hard-clustering method was employed on LDA analysis to create topic models for mixture model adaptation and showed improvement in perplexity and recognition accuracy [72, 73]. Here, the mixture weights of the topic clusters are created using the latent topic word counts obtained from the LDA analysis. A unigram count weighting approach [39] for the topics generated by hard-clustering has shown better performance over the weighting approach described in [72, 73]. LDA is also used as a clustering algorithm to cluster training data into topics [51, 87]. The LDA model can be merged with n-gram models and achieve perplexity reduction [91]. A non-stationary version of LDA can be developed for LM adaptation in speech recognition [22]. The LDA model is extended by HMM-LDA to separate the syntactic words from the topic-dependent content words in the semantic class, where content words are modeled as a mixture of topic distributions [14]. Style and topic language model

adaptation are investigated by using context-dependent labels of HMM-LDA [56]. A bigram LDA topic model, where the word probabilities are conditioned on their preceding context and the topic probabilities are conditioned on the documents, has been recently investigated [100]. A similar model but in the PLSA framework called a bigram PLSA model was introduced recently [82]. An updated bigram PLSA model (UBPLSA) was proposed in [7] where the topic is further conditioned on the bigram history context. A topic-dependent LM, called topic dependent class (TDC) based n-gram LM, was proposed in [80], where the topic is decided in an unsupervised manner. Here, the LSA method was used to reveal latent topic information from noun-noun relations [80].

Although the LDA model has been used successfully in recent research work for LM adaptation, the extracted topic information is not directly useful for speech recognition, where the latent topic of n-gram events should be of concern. In [20], a latent Dirichlet language model (LDLM) was developed where the latent topic information was exploited from $(n\text{-}1)$ history words through the Dirichlet distribution in calculating the n-gram probabilities. A topic cache language model was proposed where the topic information was obtained from long-distance history through multinomial distributions [21]. A Dirichlet class language model (DCLM) was introduced in [21] where the latent class information was exploited from the $(n\text{-}1)$ history words through the Dirichlet distribution in calculating the n-gram probabilities. The latent topic variable in DCLM reflects the class of an n-gram event rather than the topic in the LDA model, which is extracted from large-span documents [21].

1.2 Overview of this Thesis

We began this thesis by stating the importance of language modeling for automatic speech recognition. The history of language modeling is then stated followed by the improvement of LMs incorporating smoothing algorithms and long-range dependencies. The remainder of this thesis is organized into the following chapters:

Chapter 2: Literature Review

In this chapter, we illustrate the basics of language modeling in speech recognition, language modeling theory, n-gram language models, the language model's quality measurement metrics, the importance of smoothing algorithms, the class-based language model, the language model adaptation techniques, and semantic analysis approaches.

Chapter 3: LDA-based LM Adaptation Using LSM

We form topics by applying a hard-clustering method into the document-topic matrix of the LDA analysis. Topic-specific n-gram LMs are created. We introduce an n-gram weighting approach [40] to adapt the component topic models. The adapted model is then interpolated with a background model to capture local lexical regularities. The above models are further modified [49] by applying an unigram scaling technique [68] using latent semantic marginals (LSM) [96].

Chapter 4: Topic n-gram Count LM

We propose a topic n-gram count LM (TNCLM) [42] using the features of the LDA model. We assign background n-grams into topics with counts such that the total count of an n-gram for all topics is equal to the global count of the n-grams. Here, the topic weights for the n-gram are multiplied by the global count of the n-grams in the training set. We apply hard and soft clustering of the background n-grams using two confidence measures: the probability of word given topic and the probability of topic given word. The topic weights of the n-gram are computed by averaging the confidence measures over the words in the n-grams.

Chapter 5: Novel Topic n-gram Count LM

The TNCLM model does not capture the long-range information outside of the n-gram events. To tackle this problem, we introduce a novel topic n-gram count language model (NTNCLM) using document-based topic distributions and document-based n-gram counts [48].

Chapter 6: Context-based PLSA and Document-based CPLSA

We introduce a context-based PLSA (CPLSA) model [43], which is similar to the PLSA model except the topic is conditioned on the immediate history context and the document. We compare the CPLSA model with a recently proposed unsmoothed bigram PLSA (UBPLSA) model [7], which can calculate only the seen bigram probabilities that give the incorrect topic probability for the present history context of the unseen document. An extension of the CPLSA model defined as the document-based CPLSA (DCPLSA) model is also in-

troduced where the document-based word probabilities for topics are trained in the CPLSA model [50].

Chapter 7: Interpolated LDLM

Since the latent Dirichlet LM (LDLM) model [20] does not capture the long-range information from outside of *n*-gram events, we present an interpolated LDLM (ILDLM) by using different distanced *n*-grams [44]. We also incorporate a cache-based LM into the above models, which model the re-occurring words, through unigram scaling to adapt the LDLM and ILDLM models that model the topical words.

Chapter 8: Enhanced PLSA and Interpolated EPLSA

Similar to the LDLM and ILDLM approaches, we introduce an enhanced PLSA (EPLSA) and an interpolated EPLSA (IEPLSA) model in the PLSA framework. Default background *n*-grams and interpolated distanced *n*-grams are used to derive EPLSA and IEPLSA models [45]. A cache-based LM that models the re-occurring words is also incorporated through unigram scaling to the EPLSA and IEPLSA models, which models the topical words.

Chapter 9: Interpolated DCLM

The Dirichlet class LM (DCLM) model [21] does not capture the long-range information from outside of the *n*-gram window that can improve the language modeling performance. We present an interpolated DCLM (IDCLM) by using different distanced *n*-grams [47].

Chapter 10: Document-based DCLM

We introduce a document-based DCLM (DDCLM) by using document-based *n*-gram events. In this model, the class is conditioned on the immediate history context and the document in the original DCLM model [21] where the class information is obtained from the (*n*-1) history words of background *n*-grams. The counts of the background *n*-grams are the sum of the *n*-grams in different documents where they could appear to describe different topics. We consider this problem in the DCLM model and propose the DDCLM model [46].

Chapter 11: Conclusions and Future Work

We summarize the proposed techniques and the possible future work, where our proposed approach could be considered.

Chapter 2

Literature Review

In this chapter, we describe the foundation of other chapters. We first review the basic uses of language models in speech recognition, the n-gram language models, and the importance of smoothing algorithms in speech recognition. Then, we briefly describe the semantic analysis approaches that extract the semantic information from a corpus. The necessity of language model (LM) adaptation and some adaptation techniques that are relevant to our thesis are illustrated. Finally, the quality measurement parameters of the language models and the experimental setup are explained.

2.1 Language Modeling for Speech Recognition

In reality, human-like performance of speech recognition cannot be achieved through acoustic modeling alone; some form of linguistic knowledge is required. In speech recognition, language modeling tries to capture the properties of a language and predict the next word in a speech sequence. It encodes the linguistic knowledge in a way that helps computer systems that deal with human language. However, a speech recognizer is generated by a combination of acoustic modeling and language modeling. It can be described as in Figure 2.1. The speech input is placed into the recognizer through acoustic data O. The role of the recognizer is to find the most likely word string W' as:

$$W' = \underset{W}{\operatorname{argmax}} P(W|O) \tag{2.1}$$

where $P(W|O)$ represents the probability that the word W was spoken, given that the evidence O was observed. The right-hand side probability of Equation 2.1 can be re-arranged

by using Bayes' law as:

$$P(W|O) = \frac{P(W)P(O|W)}{P(O)}, \tag{2.2}$$

where $P(W)$ is the probability that the word W will be uttered, $P(O|W)$ is the probability that the acoustic evidence O will be observed when the word W is spoken by the speaker, and $P(O)$ is the probability that O will be observed. So, $P(O)$ can be ignored as $P(O)$ is not dependent on the word string that is selected. Therefore, Equation 2.1 can be written as:

$$W' = \underset{W}{\operatorname{argmax}} P(W|O) = \underset{W}{\operatorname{argmax}} P(W)P(O|W), \tag{2.3}$$

where $P(O|W)$ is determined by the acoustic modeling and $P(W)$ is determined by the language modeling part of a speech recognizer.

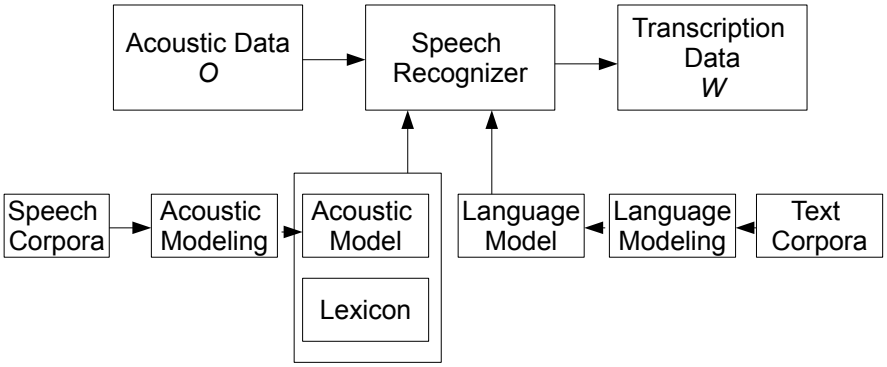

Fig. 2.1 Speech recognition system

2.2 Language Modeling Theory

In recognition and understanding of natural speech, the knowledge of language is also important along with the acoustic pattern matching. It includes the lexical knowledge that is based on vocabulary definition and word pronunciation, syntax and semantics of the language, which are based on the rules that are used to determine what sequences of words are grammatically meaningful and well-formed. Also, the pragmatic knowledge of the language that is based on the structure of extended discourse and what people are likely to say in particular contexts are also important in spoken language understanding (SLU) systems. In speech recognition, it may be impossible to separate the use of these different levels of knowledge, as they are tightly integrated [57].

2.2.1 Formal Language Theory

In formal language theory, two things are important: grammar and the parsing algorithm. The grammar is an acceptable framework of the language and the parsing technnique is the method to see if its structure is matched with the grammar.

There are three requirements of a language model. These are:

- *Generality*, which determines the range of sentences accepted by the grammar.
- *Selectivity*, which determines the range of sentences rejected by the grammar, and
- *Understandability*, which depends upon the users of the system to create and maintain the grammar for a particular domain.

In a SLU system we have to have a grammar that covers and generalizes most of the typical sentences for an application. It should have the capability to distinguish the kinds of sentences for different actions in a given application [57].

2.2.2 Stochastic Language Models

Stochastic language models (SLMs) provide a probabilistic viewpoint of language modeling. By these models we need to measure the probability of a word sequence $W = w_1, \ldots, w_N$ accurately. In formal language theory, $P(W)$ can be computed as 1 or 0 depending on the word sequence being accepted or rejected respectively, by the grammar [57]. However, this may be inappropriate in the case of a spoken language system (SLS), since the grammar itself is unlikely to have complete coverage, not to mention that spoken language is often ungrammatical in real conversational applications. However, the main goal of a SLM is to supply adequate information so that the likely word sequences should have higher probability, which not only makes speech recognition more accurate but also helps to dramatically constrain the search space for speech recognition. The most widely used SLM is the *n*-gram model, which is described in the next section.

2.2.3 N-gram Language Models

Language modeling in speech recognition is important to differentiate the words spoken like *meat* or *meet*, which cannot be recognized by acoustic modeling alone. It also helps in

searching to find the best acoustically matching word sequence with the highest language model probabilities. The probability of a word sequence $W = w_1, \ldots, w_N$ can be defined as:

$$\begin{aligned} P(W) &= P(w_1, \ldots, w_N) \\ &= P(w_1)P(w_2|w_1)P(w_3|w_1w_2)\ldots P(w_N|w_1, w_2, \ldots, w_{N-1}) \\ &= \prod_{i=1}^{N} P(w_i|w_1, w_2, \ldots, w_{i-1}) \end{aligned} \tag{2.4}$$

where $P(w_i|w_1, w_2, \ldots, w_{i-1})$ is the probability that the word w_i will follow, given the word sequence $w_1, w_2, \ldots, w_{i-1}$. In general, $P(w_i)$ is dependent on the entire history. For a vocabulary size of v, v^i values have to be estimated to compute $P(w_i|w_1, w_2, \ldots, w_{i-1})$ completely as there are v^{i-1} different histories. However, in practice, it is impossible to compute the probabilities $P(w_i|w_1, w_2, \ldots, w_{i-1})$ for a moderate size of i, since most histories $w_1, w_2, \ldots, w_{i-1}$ are unique or have occurred only a few times in most available datasets [57]. Also, a language model for a context of arbitrary length would require an infinite amount of memory. The most common solution to this problem is to assume that the probability $P(w_i|w_1, w_2, \ldots, w_{i-1})$ depends on some equivalence classes that are based on some previous words $w_{i-n+1}, w_{i-n+2}, \ldots, w_{i-1}$. Therefore, Equation 2.4 can be written as:

$$\begin{aligned} P(W) &= P(w_1, \ldots, w_N) \\ &= \prod_{i=1}^{N} P(w_i|w_1, w_2, \ldots, w_{i-1}) \\ &\approx \prod_{i=1}^{N} P(w_i|w_{i-n+1}, \ldots, w_{i-1}). \end{aligned} \tag{2.5}$$

This leads to the n-gram language model, which is used to approximate the probability of a word sequence using the conditional probability of the embedded n-grams. Here, "n-gram" refers to the sequence of n words and n=1,2, and 3 represents the unigram, bi-gram and tri-gram respectively. At present, tri-gram models yield the best performance depending on the available training data for language models. However, the interest is also growing in moving to 4-gram models and beyond.

Statistical n-gram language models have been successfully used in speech recognition. The probability of the current word is dependent on the previous $(n-1)$ words. In computing the probability of a sentence using an n-gram model, we pad the beginning of a sentence using a distinguished token $< s >$ such that $w_{-n+2} = \cdots = w_0 = < s >$. Also, to make the sum of the probability of all sentences equal to 1, it is necessary to add a distinguished token $< /s >$ at the end of the sentence and include it in the product of the

conditional probabilities. For example, the bigram and trigram probabilities of the sentence *LIFE IS BEAUTIFUL* can be computed as:

$$P_{bigram}(\ LIFE\ IS\ BEAUTIFUL) = P(LIFE|<s>)P(IS|LIFE)\dots P(</s>|BEAUTIFUL)$$

$$P_{trigram}(\ LIFE\ IS\ BEAUTIFUL) = P(LIFE|<s>,<s>)P(IS|<s>,LIFE)\dots P(</s>|IS,BEAUTIFUL)$$

To compute $P(w_i|w_{i-1})$ in a bi-gram model, i.e., the probability that the word w_i occurs given the preceding word w_{i-1} , we need to count the number of occurrences of (w_{i-1},w_i) in the training corpus and normalize the count by the number of times w_{i-1} occurs [57]. Therefore,

$$P(w_i|w_{i-1}) = \frac{C(w_{i-1},w_i)}{\sum_{w_i} C(w_{i-1},w_i)} \tag{2.6}$$

This is known as the maximum likelihood (ML) estimate of $P(w_i|w_{i-1})$ as it maximizes the bi-gram probability of the training data. For n-gram models, the probability of the n-th word depends on the previous $(n-1)$ words. With Equation 2.6, we can compute the probability of $P(w_i|w_{i-n+1},\dots,w_{i-1})$ as:

$$P(w_i|w_{i-n+1},\dots,w_{i-1}) = \frac{C(w_{i-n+1},\dots,w_i)}{\sum_{w_i} C(w_{i-n+1},\dots,w_i)} \tag{2.7}$$

Let us consider the following 2 sentences in the training data. *LIFE IS BEAUTIFUL* and *LIFE IS GOOD*. The bigram probability of the sentence *LIFE IS BEAUTIFUL* using maximum likelihood estimation can be computed as:

$$\begin{aligned}
P(LIFE|<s>) &= \tfrac{C(<s>,LIFE)}{\sum_w C(<s>,w)} = \tfrac{1}{2} \\
P(IS|LIFE) &= \tfrac{C(LIFE,IS)}{\sum_w C(LIFE,w)} = \tfrac{2}{2} = 1 \\
P(BEAUTIFUL|IS) &= \tfrac{C(IS,BEAUTIFUL)}{\sum_w C(IS,w)} = \tfrac{1}{2} \\
P(</s>|BEAUTIFUL) &= \tfrac{C(BEAUTIFUL,</s>)}{\sum_w C(BEAUTIFUL,w)} = \tfrac{1}{1} = 1
\end{aligned}$$

Therefore,

$$\begin{aligned}
P(\ LIFE\ IS\ BEAUTIFUL) &= P(LIFE|<s>)P(IS|LIFE)P(BEAUTIFUL|IS)P(</s>|BEAUTIFUL) \\
&= \frac{1}{2} \times 1 \times \frac{1}{2} \times 1 \\
&= \frac{1}{4}
\end{aligned} \tag{2.8}$$

However, this technique cannot estimate the probability of unseen data. For example, if we want to find the probability of a sentence *LIFE IS WELL*, the maximum likelihood

estimate assigns zero probability, but the sentence should have a resonable probability. To resolve this problem, various smoothing techniques have been developed, which are discussed in section 2.3.

The n-gram models have some advantages and disadvantages. On the positive side, the models can encode both syntax and semantics simultaneously and they can train easily from a large amount of training data. Also they can easily be included in the decoder of a speech recognizer. On the negative side, the current word depends on much more than the previous one or two words. Therefore, the main problem of n-gram modeling is that it cannot capture the long-range dependencies between words. It only depends on the immediate previous $(n-1)$ words. In reality, the training data is formed by a diverse collection of topics. So, we have to find a model that includes the long-range dependencies too.

2.3 Smoothing

When the training corpus is not large enough, many possible word successions may not be actually observed, which leads to many small probabilities or zero probabilities in the LM. For example, by using the training data in subsection 2.2.3, we see that the ML estimate assigns zero probability to the sentence *LIFE IS WELL* as the count of bigrams *(IS, WELL)* and *(WELL, </s>)* are zero. However, the sentence should have a reasonable probability; otherwise it creates errors in speech recognition. In Equation 2.2, we can see that if $P(W)$ is zero, the string can never be considered as a possible transcription, regardless of how unambiguous the acoustic signal. So, when $P(W) = 0$, it will create errors in speech recognition. This is actually the main motivation for smoothing.

Smoothing describes the techniques to adjust the maximum likelihood estimate to find more accurate probabilities. It helps to find more robust probabilities for unseen data. Smoothing is performed by making the distributions more uniform, by adjusting the low probabilities such as zero probabilities upward, and high probabilities downward. The simple smoothing technique assumes that each n-gram occurs one more time than it actually does, i.e.,

$$\begin{aligned} P(w_i|w_{i-1}) &= \frac{1+C(w_{i-1},w_i)}{\sum_{w_i}[1+C(w_{i-1},w_i)]} \\ &= \frac{1+C(w_{i-1},w_i)}{|V|+\sum_{w_i}C(w_{i-1},w_i)} \end{aligned} \tag{2.9}$$

where V is the vocabulary. Let us consider the training data from section 2.2.3. Here, V = 6 (with both $< s >$ and $< /s >$). The probability of *P(LIFE IS WELL)* can be computed using

Equation 2.9 as:

$$\begin{aligned}
P(LIFE|<s>) &= \tfrac{1+C(<s>,LIFE)}{|V|+\sum_w C(<s>,w)} = \tfrac{3}{8} \\
P(IS|LIFE) &= \tfrac{1+C(LIFE,IS)}{|V|+\sum_w C(LIFE,w)} = \tfrac{3}{8} \\
P(WELL|IS) &= \tfrac{1+C(IS,WELL)}{|V|+\sum_w C(IS,w)} = \tfrac{1}{8} \\
P(</s>|WELL) &= \tfrac{1+C(WELL,</s>)}{|V|+\sum_w C(WELL,w)} = \tfrac{1}{6}.
\end{aligned}$$

Therefore

$$\begin{aligned}
P(\text{ LIFE IS WELL}) &= P(LIFE|<s>)P(IS|LIFE)P(WELL|IS)P(</s>|WELL) \\
&= \frac{3}{8} \times \frac{3}{8} \times \frac{1}{8} \times \frac{1}{6} \\
&\approx 0.0029,
\end{aligned} \tag{2.10}$$

which is more reasonable than the zero probability obtained by the ML estimate.

However, smoothing not only helps for preventing zero probabilities, but also helps to improve the accuracy of the model as it distributes the probability mass from observed to unseen n-grams. There are various smoothing techniques that have been discussed in [19]. Here, we briefly describe only the general form of most smoothing techniques with Witten-Bell smoothing (see section 2.3.2) and modified Kneser-Ney smoothing (see sections 2.3.3, 2.3.4), which are used in all the experiments of this thesis.

2.3.1 General Form of Smoothing Algorithms

In general, most smoothing algorithms take the following form:

$$P_{smooth}(w_i|w_{i-n+1},\ldots,w_{i-1}) =$$

$$\begin{cases}
f(w_i|w_{i-n+1},\ldots,w_{i-1}), & \text{if } C(w_{i-n+1},\ldots,w_i) > 0 \\
bow(w_{i-n+1},\ldots,w_{i-1})P_{smooth}(w_i|w_{i-n+2},\ldots,w_{i-1}), & \text{if } C(w_{i-n+1},\ldots,w_i) = 0
\end{cases}$$

This means that when an n-gram has a non-zero count, we used the distribution $f(w_i|w_{i-n+1},\ldots,w_{i-1})$. Typically, $f(w_i|w_{i-n+1},\ldots,w_{i-1})$ is discounted to be less than the ML estimate. Different algorithms differ on how they discount the ML estimate to get $f(w_i|w_{i-n+1},\ldots,w_{i-1})$.
On the other hand, when an n-gram has not been observed in the training data, we used the lower-order n-gram distribution $P_{smooth}(w_i|w_{i-n+2},\ldots,w_{i-1})$, where the scaling factor $bow(w_{i-n+1},\ldots,w_{i-1})$ is computed to make the conditional distribution sum to one.

Let us consider V to be the set of all words in the vocabulary, V_0 be the set of all w_i words with $C(w_{i-n+1},\ldots,w_i)=0$, and V_1 be the set of all w_i words with $C(w_{i-n+1},\ldots,w_i)>0$. Now, given $f(w_i|w_{i-n+1},\ldots,w_{i-1})$, $bow(w_{i-n+1},\ldots,w_{i-1})$ can be obtained as follows:

$$\sum_V P_{smooth}(w_i|w_{i-n+1},\ldots,w_{i-1})=1$$
$$\sum_{V_1} f(w_i|w_{i-n+1},\ldots,w_{i-1})+\sum_{V_0} bow(w_{i-n+1},\ldots,w_{i-1})P_{smooth}(w_i|w_{i-n+2},\ldots,w_{i-1})=1.$$

Therefore,

$$\begin{aligned} bow(w_{i-n+1},\ldots,w_{i-1}) &= \frac{1-\sum_{V_1} f(w_i|w_{i-n+1},\ldots,w_{i-1})}{\sum_{V_0} P_{smooth}(w_i|w_{i-n+2},\ldots,w_{i-1})} \\ &= \frac{1-\sum_{V_1} f(w_i|w_{i-n+1},\ldots,w_{i-1})}{1-\sum_{V_1} P_{smooth}(w_i|w_{i-n+2},\ldots,w_{i-1})} \\ &= \frac{1-\sum_{V_1} f(w_i|w_{i-n+1},\ldots,w_{i-1})}{1-\sum_{V_1} f(w_i|w_{i-n+2},\ldots,w_{i-1})}. \end{aligned} \tag{2.11}$$

Smoothing is generally performed in two ways. The back-off models compute $P_{smooth}(w_i|w_{i-n+1},\ldots,w_{i-1})$ based on the n-gram counts $C(w_{i-n+1},\ldots,w_i)=0$ and $C(w_{i-n+1},\ldots,w_i)>0$. This first method considers the lower order counts $C(w_{i-n+2},\ldots,w_i)$ only when $C(w_{i-n+1},\ldots,w_i)=0$. The other way is an interpolated model, which can be obtained by the linear interpolation of higher and lower order n-gram models as:

$$\begin{aligned} P_{smooth}(w_i|w_{i-n+1},\ldots,w_{i-1}) &= \lambda_{w_{i-n+1},\ldots,w_{i-1}} P_{ML}(w_i|w_{i-n+1},\ldots,w_{i-1}) \\ &\quad + (1-\lambda_{w_{i-n+1},\ldots,w_{i-1}}) P_{smooth}(w_i|w_{i-n+2},\ldots,w_{i-1}) \end{aligned} \tag{2.12}$$

where $\lambda_{w_{i-n+1},\ldots,w_{i-1}}$ is the interpolation weight, which depends on $w_{i-n+1},\ldots,w_{i-1}$. The key difference between the interpolated models and the back-off models is that for the probability of n-grams with nonzero counts, only interpolated models use additional information from the lower-order distributions. Both the models use the lower order distribution when computing the probability of n-grams with zero counts.

2.3.2 Witten-Bell Smoothing

In Witten-Bell smoothing, the nth-order smoothed model can be defined as a linear interpolation of the nth-order maximum likelihood model and the $(n-1)$th-order smoothed model as:

$$\begin{aligned} P_{WB}(w_i|w_{i-n+1},\ldots,w_{i-1}) &= \lambda_{w_{i-n+1},\ldots,w_{i-1}} P_{ML}(w_i|w_{i-n+1},\ldots,w_{i-1}) \\ &\quad + (1-\lambda_{w_{i-n+1},\ldots,w_{i-1}}) P_{WB}(w_i|w_{i-n+2},\ldots,w_{i-1}). \end{aligned} \tag{2.13}$$

To compute the parameters $\lambda_{w_{i-n+1},\ldots,w_{i-1}}$, we need the number of unique words with one or more counts that follow the history $w_{i-n+1},\ldots,w_{i-1}$, which is denoted by $N_{1+}(w_{i-n+1},\ldots,w_{i-1},*)$ [19] and defined as:

$$N_{1+}(w_{i-n+1},\ldots,w_{i-1},*) = \left\{ w_i : C(w_{i-n+1},\ldots,w_i) > 0 \right. \tag{2.14}$$

The weight given to the lower model should be proportional to the probability of observing an unseen word in the current context $(w_{i-n+1},\ldots,w_{i-1})$. Therefore,

$$1-\lambda_{w_{i-n+1},\ldots,w_{i-1}} = \frac{N_{1+}(w_{i-n+1},\ldots,w_{i-1},*)}{N_{1+}(w_{i-n+1},\ldots,w_{i-1},*)+\sum_{w_i} C(w_{i-n+1},\ldots,w_i)}.$$

Therefore Equation 2.13 can be written as:

$$P_{WB}(w_i|w_{i-n+1},\ldots,w_{i-1}) = \frac{C(w_{i-n+1},\ldots,w_i)+N_{1+}(w_{i-n+1},\ldots,w_{i-1},*)P_{WB}(w_i|w_{i-n+2},\ldots,w_{i-1})}{N_{1+}(w_{i-n+1},\ldots,w_{i-1},*)+\sum_{w_i} C(w_{i-n+1},\ldots,w_i)}. \tag{2.15}$$

Equation 2.15 is an interpolated version of the Witten-Bell smoothing. Therefore, the back-off version of this smoothing can be expressed as:

$$P_{smooth}(w_i|w_{i-n+1},\ldots,w_{i-1}) = \begin{cases} f(w_i|w_{i-n+1},\ldots,w_{i-1}), & \text{if } C(w_{i-n+1},\ldots,w_i) > 0 \\ bow(w_{i-n+1},\ldots,w_{i-1})P_{smooth}(w_i|w_{i-n+2},\ldots,w_{i-1}), & \text{if } C(w_{i-n+1},\ldots,w_i) = 0 \end{cases} \tag{2.16}$$

where

$$f(w_i|w_{i-n+1},\ldots,w_{i-1}) = \frac{C(w_{i-n+1},\ldots,w_i)}{N_{1+}(w_{i-n+1},\ldots,w_{i-1},*)+\sum_{w_i} C(w_{i-n+1},\ldots,w_i)}$$
$$bow(w_{i-n+1},\ldots,w_{i-1}) = \frac{1-\sum_{V1} f(w_i|w_{i-n+1},\ldots,w_{i-1})}{1-\sum_{V1} f(w_i|w_{i-n+2},\ldots,w_{i-1})}.$$

2.3.3 Kneser-Ney Smoothing

Kneser-Ney (KN) smoothing is an extension of absolute discounting [81] where the lower-order distribution combines with a higher-order distribution in a novel manner. The KN smoothing can be described as Equation 2.16 with

$$f(w_i|w_{i-n+1},\ldots,w_{i-1}) = \frac{max(C(w_{i-n+1},\ldots,w_i)-D,0)}{\sum_{w_i} C(w_{i-n+1},\ldots,w_i)}$$
$$bow(w_{i-n+1},\ldots,w_{i-1}) = \frac{D}{\sum_{w_i} C(w_{i-n+1},\ldots,w_i)} N_{1+}(w_{i-n+1},\ldots,w_{i-1},*).$$

where

$$D = \frac{n1}{n1+2n2}$$

where $n1$ and $n2$ are the number of n-grams that appear exactly once and twice in the training corpus.

2.3.4 Modified Kneser-Ney Smoothing

Instead of using a single discount D for all counts, modified KN smoothing [19] has shown better performance, incorporating three different parameters, D_1, D_2, and D_{3+} for the n-grams with one, two, and three or more counts respectively. Therefore, the modified KN smoothing can be described as Equation 2.16 with

$$f(w_i|w_{i-n+1},\ldots,w_{i-1}) = \frac{max(C(w_{i-n+1},\ldots,w_i)-D(C(w_{i-n+1},\ldots,w_i),0)}{\sum_{w_i} C(w_{i-n+1},\ldots,w_i)}$$
$$bow(w_{i-n+1},\ldots,w_{i-1}) = \frac{D_1N_1(w_{i-n+1},\ldots,w_{i-1},*)+D_2N_2(w_{i-n+1},\ldots,w_{i-1},*)+D_{3+}N_{3+}(w_{i-n+1},\ldots,w_{i-1},*)}{\sum_{w_i} C(w_{i-n+1},\ldots,w_i)}.$$

where $N_2(w_{i-n+1},\ldots,w_{i-1},*)$ and $N_{3+}(w_{i-n+1},\ldots,w_{i-1},*)$ are defined analogously as $N_1(w_{i-n+1},\ldots,w_{i-1},*)$ and

$$D(c) = \begin{cases} 0 & \text{if } c = 0 \\ D1 & \text{if } c = 1 \\ D2 & \text{if } c = 2 \\ D3+ & \text{if } c \geq 3 \end{cases}$$

The discounts can be estimated as [19]:

$$Y = \frac{n1}{n1+2n2},\ \ D1 = 1 - 2Y\frac{n2}{n1},\ \ D2 = 2 - 3Y\frac{n3}{n2},\ \ D3+ = 3 - 4Y\frac{n4}{n3}$$

2.4 Class-based LM

To compensate for the data-sparseness problem, class-based LM has been proposed [16]. Here, the transition probabilities between classes are considered rather than words:

$$P_{Class}(w_i|w_{i-n+1},\ldots,w_{i-1}) = P(w_i|c_{i-1},w_{i-1},c_i)P(c_i|c_{i-1},w_{i-1}) \tag{2.17}$$

Assume that $P(w_i|c_{i-1}, w_{i-1}, c_i)$ is independent of c_{i-1}, w_{i-1}, and $P(c_i|c_{i-1}, w_{i-1})$ is independent of w_{i-1}. Therefore, the equation 2.17 becomes:

$$P_{Class}(w_i|w_{i-n+1}, \ldots, w_{i-1}) = P(w_i|c_i)P(c_i|c_{i-n+1}, \ldots, c_{i-1}) \tag{2.18}$$

where c_i is the class assignment of word w_i, $P(w_i|c_i)$ is the probability of word w_i, generated from class c_i, and $P(c_i|c_{i-n+1}, \ldots, c_{i-1})$ is the class n-gram. The class-based LM generates the parameters for word classes instead of words. Therefore, the model significantly reduces the parameter sizes by mapping the words into classes. As a result, the performance of this model is slightly worse compared to a word-based n-gram LM. So, the class-based n-gram model is generally linearly interpolated with the word-based n-gram LM as [80]:

$$P(w_i|w_{i-n+1}, \ldots, w_{i-1}) \approx \lambda P_{Class}(w_i|w_{i-n+1}, \ldots, w_{i-1}) + (1-\lambda)P_{Ngram}(w_i|w_{i-n+1}, \ldots, w_{i-1}). \tag{2.19}$$

2.5 Semantic Analysis

2.5.1 Background

Significant progress has been made for the problem of modeling text corpora by information retrieval (IR) researchers [5]. The basic method proposed by IR researchers is that a document in a text corpus can be reduced to a vector of real numbers, each of which represents a ratio of counts. In the popular *tf-idf* scheme [89], a term-by-document matrix is formed by using *tf-idf* values where the columns contain the *tf-idf* values for each of the documents in the corpus. The *tf-idf* value is calculated by using the number of occurrences of each word or term for each document, which is then compared with its inverse document frequency count, i.e., the number of occurrences of the corresponding word or term in the entire corpus. The scheme reduces the corpus to a fixed-size matrix $V \times M$, where V is the number of words in the vocabulary and M is the number of documents in the corpus. Therefore, the scheme reduces documents of arbitrary length to fixed-length lists of numbers. However, there are two problems that arise in *tf-idf* schemes such as: (i) synonyms, i.e., different words may have a similar meaning, and (ii) polysemes, i.e., a word may have multiple senses and multiple types of usage in different contexts. Therefore, the *tf-idf* scheme provides a small amount of reduction in description length.

Various dimensionality reduction techniques such as LSA, PLSA, and LDA have been proposed to address these issues. They are mainly developed to find out the hidden meaning

behind the text. All of these methods are known as bag-of-words models as they collect words in the documents regardless of their word order.

2.5.2 LSA

Latent semantic analysis (LSA) was first introduced in [26] for information retrieval. In [10], it was brought to the area of LM for ASR. LSA is an approach in natural language processing, which is used to describe the relationships between a set of documents and the words they contain, by producing a set of concepts related to the documents and words. LSA assumes that words that are close in meaning will occur in similar pieces of text. It uses a word-document ($V \times M$) matrix G. The rows V and columns M of the matrix G represent the words and documents respectively. Each word can be described by a row vector of dimension M, and each document can be represented by a column vector of dimension V. Unfortunately, these vector representations are impractical for three reasons [10]. First, the dimensions V and M can be extremely large; second, the word and document vectors are typically very sparse; and third, the two spaces are distinct from one another. To address these issues, a mathematical technique called singular value decomposition (SVD) is used to project the discrete indexed words and documents into a continuous (semantic) vector space, in which familiar clustering techniques can be applied. For example, words or documents are compared by taking the cosine of the angle between the two vectors formed by any two rows or columns. Values close to 1 represent very similar words or documents while values close to 0 represent very dissimilar words or documents [10]. The SVD decomposes the matrix G into three other matrices X, S, and Y as:

$$G = XSY^T \tag{2.20}$$

where X is the $(V \times R)$ left singular matrix with row vectors $x_i (1 \leq i \leq V)$, S is the $R \times R$ diagonal matrix of singular values, Y is the $(M \times R)$ right singular matrix with column vectors $y_j (1 \leq j \leq M)$, S, $R \ll min(V,M)$ is the order of decomposition; and T is transposition.

The LSA model has some limitations. It cannot capture polysemy (i.e., multiple meanings of a word). Each word is represented by a single point in the semantic vector space, and thus each word has a single meaning. The LSA model does not introduce a generative probabilistic model of text corpora. It uses a dimensionality reduction technique to map the term-document matrix into a continuous vector space in which familiar clustering methods are applied to obtain topic clusters.

2.5.3 PLSA

PLSA was introduced to overcome the limitation of LSA. It extracts the semantic information from a corpus in a probabilistic framework. PLSA uses an unobserved topic variable with each observation, i.e., with each occurrence of a word in a document. It is assumed that the document and the word are independently conditioned on the state of the latent topic variable. It models each word in a document as a sample from a mixture model, where the mixture models can be viewed as representations of topic distributions. Therefore, a document is generated as a mixture of topic distributions and reduced to a fixed set of topics. Each topic is a distribution over words. However, the problem with the PLSA model is that it does not provide any probabilistic model at the level of the documents. Each document is modeled by its own mixture weights and there is no generative model for these weights. So, the number of parameters grows linearly when increasing the number of documents, which leads to an overfitting problem. Also, there is no method to assign probability for a document outside the training set [33]. The PLSA model can be described in the following procedure. First a document is selected with probability $P(d_l)$ $(l = 1,\ldots,M)$. A topic t_k $(k = 1,\ldots,K)$ is then chosen with probability $P(t_k|d_l)$ and finally a word w_i $(i = 1,\ldots,N)$ is generated with probability $P(w_i|t_k)$. The graphical representation of the model is described in Figure 2.2. The joint probability of a word w_i and a document d_l can be estimated as:

$$P(d_l, w_i) = P(d_l) \sum_{k=1}^{K} P(w_i|t_k)P(t_k|d_l). \tag{2.21}$$

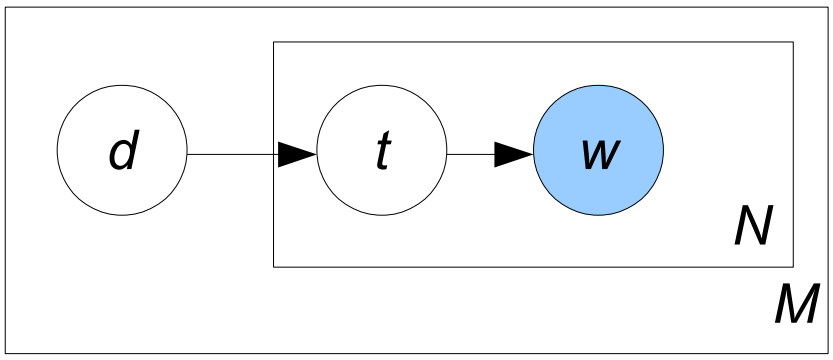

Fig. 2.2 Graphical structure of the PLSA model. The shaded circle represents observed variable.

The topics of the model are generated from the statistics of the corpus of training doc-

uments. The numbers of topics are assumed fixed in this approach. The model parameters $P(w_i|t_k)$ and $P(t_k|d_l)$ are computed by using the expectation maximization (EM) algorithm [33].

2.5.4 LDA

To address the limitations of the PLSA model, the LDA model [13] was introduced where a Dirichlet distribution is applied on the topic mixture weights corresponding to the documents in the corpus. Therefore, the number of model parameters is dependent only on the number of topic mixtures and the vocabulary size. Thus, LDA is less prone to overfitting and can be used to compute the probabilities of unobserved test documents. However, LDA is equivalent to the PLSA model under a uniform Dirichlet prior distribution.

LDA is a three-level hierarchical Bayesian model. It is a generative probabilistic topic model for documents in a corpus. Documents are represented by the random latent topics[1], which are characterized by a distribution over words. The graphical representation of the LDA model is shown in Figure 2.3. Here, we can see the three levels of LDA. The Dirichlet priors α and β are the corpus level parameters that are assumed to be sampled once in generating the corpus. The parameters θ are document-level variables and sampled once per document. The variables t and w are word-level variables and sampled once for each word in each document [13].

The LDA parameters (α,β) are estimated by maximizing the marginal likelihood of training documents. $\alpha = \{\alpha_{t_1},\ldots,\alpha_{t_K}\}$ represents the Dirichlet parameter for K latent topics and β represents the Dirichlet parameter over the words and defined as a matrix with multinomial entry $\beta_{t_k,w_i} = P(w_i|t_k)$. The LDA model can be described in the following way. Each document $d_l = [w_1,\ldots,w_N]$ $(l = 1,\ldots,M)$ is generated as a mixture of unigram models, where the topic mixture vector θ_{d_l} is drawn from the Dirichlet distribution with parameter α. The corresponding topic sequence $t = [t_1,\ldots,t_N]$ is generated using the multinomial distribution θ_{d_l}. Each word w_N is generated using the distribution $P(w_N|t_N,\beta)$. The joint probability of d_l, topic assignment t and topic mixture vector θ_{d_l} is given by:

$$P(d_l,t,\theta_{d_l}|\alpha,\beta) = P(\theta_{d_l}|\alpha)\prod_{i=1}^{N} P(t_i|\theta_{d_l})P(w_i|t_i,\beta). \tag{2.22}$$

The probability of the document d_l can be estimated by marginalizing unobserved vari-

[1]Topics are unobserved in LDA.

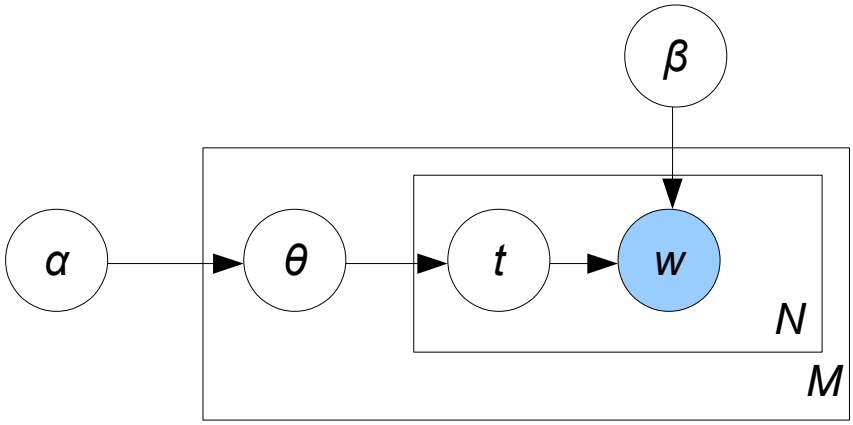

Fig. 2.3 Graphical model representation of LDA. The shaded circle represents observed variable.

ables θ_{d_l} and t as:

$$P(d_l|\alpha,\beta) = \int P(\theta_{d_l}|\alpha)\prod_{i=1}^{N}\sum_{t_i} P(t_i|\theta_{d_l})P(w_i|t_i,\beta)d\theta_{d_l} \tag{2.23}$$

where θ_{d_l} is a K-dimensional random variable that can take values in the $(K-1)$-simplex (a K-vector θ_{d_l} lies in the $(K-1)$-simplex if $\theta_{d_l t_i} > 0$, $\sum_{t_i}\theta_{d_l t_i} = 1$), and has the following probability density on this simplex:

$$P(\theta_{d_l}|\alpha) = \frac{\Gamma(\sum_{t_i}\alpha_{t_i})}{\prod_{t_i}\Gamma(\alpha_{t_i})}\theta_{d_l t_1}^{\alpha_{t_1}-1}\ldots\theta_{d_l t_K}^{\alpha_{t_K}-1} \tag{2.24}$$

where the parameter α is a K-vector with components $\alpha_{t_i} > 0$, and $\Gamma(x)$ is the Gamma function. The Dirichlet distribution is used as the prior for the multinomial distributions. This is because of the conjugate property of the Dirichlet distribution, which results in the posterior integrals in a Dirichlet distribution, which simplifies the model inference. The product of a Dirichlet prior with the multinomial likelihood will yield another Dirichlet distribution of a certain form [13].

The parameters of the LDA model can be estimated using variational inference [13] or Gibbs sampling [35]. The variational inference method uses the variational parameters and Jensen's inequality to obtain an adjustable lower bound on the likelihood. The optimizing values of the variational parameters are obtained by minimizing the KL divergence between the variational distribution and the true posterior $P(\theta_{d_l}, t|d_l,\alpha,\beta)$. For details of calculation, see [13]. In the Gibbs sampling method, an algorithm is described for extracting a set of top-

ics from a large corpus that uses Gibbs Sampling, which is a form of Monte Carlo Markov chain [35]. It simulates a high-dimensional distribution by sampling on lower-dimensional subsets of variables where each subset is conditioned on the values of the others. The sampling is done sequentially and proceeds until the sampled values approximate the target distribution. For details, see [37].

2.6 Language Model Adaptation

Language model (LM) adaptation plays an important role for many research areas like speech recognition, machine translation, and information retrieval. Adaptation is required when the styles, domains or topics of the test data are mismatched with the training data. It is also important as natural language is highly variable since the topic information is highly non-stationary. In general, an adaptive language model seeks to maintain an adequate representation of the domain under changing conditions involving potential variations in vocabulary, content, syntax and style [11].

The training text will be the representative of the style of language that one is attempting to model. If this is not the case, the model is going to be useless. For example, trigram language models are trained by using the training data of a more common corpus like the Wall Street Journal (WSJ). When these models are used in speech recognizers, they yield high accuracy if the speech input is coming from the Wall Street Journal. However, if the models are used for spontaneous conversational speech, the recognizers provide worse results as the Wall Street Journal uses a very different style of language than conversational speech. So, language modelers make a 'general model' by using available training data and consider a little amount of data that is specific to the recognition task and referred to as the adaptation data [23].

2.6.1 Adaptation Structure

The general framework of the adaptation process is shown in Figure 2.4. For a string of N words $W = w_1, \ldots, w_N$, the language model probability can be written as:

$$P(W) = \prod_{i=1}^{N} P(w_i|h)$$

where h represents the history. In the n-gram model, the history is the previous $(n-1)$ words as:

$h = w_{i-n+1}, w_{i-n+2}, \ldots, w_i$

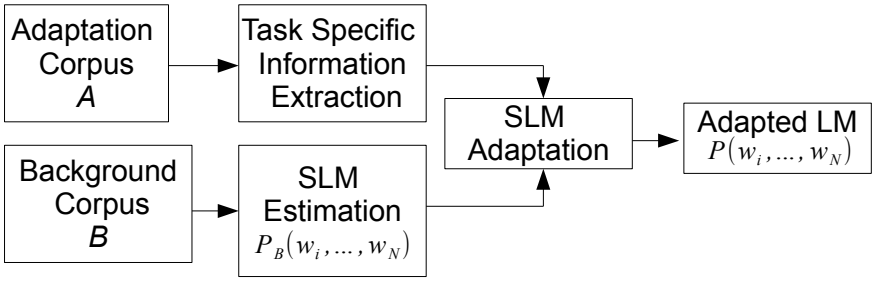

Fig. 2.4 General structure of SLM adaptation

The robust estimate of the language model probability can be found by leveraging two knowledge sources. These are the well-trained background LM with initial estimate $P_B(W)$ but mismatched data with the recognition task and the adaptation LM that is more familiar with the recognition task. For details of various adaptation ideas, see [11]. Here we will briefly describe some adaptation methodologies that are related to our thesis.

2.6.2 Model Interpolation

In this approach, the adaptation corpus A takes the form of the relevant recognition task and then is combined with the background LM to yield better results.

The simplest way to merge two models is via linear interpolation. Given the estimate of word w_i denoted by $P_B(w_i|h)$ and $P_A(w_i|h)$, the linear interpolation can be defined as:

$$P(w_i|h) = \lambda P_B(w_i|h) + (1-\lambda)P_A(w_i|h) \tag{2.25}$$

where λ is the interpolation coefficient whose value is in the range of $0 \leq \lambda \leq 1$. This parameter can be estimated on the adaptation corpus A under the maximum likelihood criterion using the EM algorithm [30].

2.6.3 MDI Adaptation Using Unigram Constraints

In this approach, the adaptation corpus A is used to extract features such that the adapted LM is constrained to satisfy. This kind of adaptation is more powerful than the model interpolation because a different weight is assigned separately for each features [11]. The constrained-based adaptation has been associated with exponential models trained using the maximum entropy (ME) criterion that leads to minimum discriminant information (MDI) estimation [34, 68].

The development of exponential models in an adaptation context is referred to as MDI adaptation. The unigram features can be reliably estimated from the adaptation corpus A. The adaptation approach [34, 68] forms an adapted model by minimizing the KL-divergence between the background model and the adapted model subject to a marginalization constraint for each word w_i in the vocabulary [96] as:

$$\sum_{h} P_A(h).P_A(w_i|h) = P_A(w_i). \tag{2.26}$$

The constraint optimization problem has close connection to the maximum entropy approach [88], which provides that the adapted model is a re-scaled version of the background model:

$$P_A(w_i|h) = \frac{\delta(w_i)}{Z(h)}.P_B(w_i|h) \tag{2.27}$$

with

$$Z(h) = \sum_{w_i} \delta(w_i).P_B(w_i|h) \tag{2.28}$$

where $P_A(w_i|h)$ is defined as the adapted model, $P_B(w_i|h)$ is the background model. $Z(h)$ is a normalization term, which guarantees that the total probability sums to unity, and $\delta(w_i)$ is a scaling factor that is usually approximated as:

$$\delta(w_i) \approx \left(\frac{P_A(w_i)}{P_B(w_i)}\right). \tag{2.29}$$

2.6.4 Mixture Model Adaptation

The mixture model adaptation is based on several component models, each of which is specific to a particular topic or style of language. The probabilities of these component models are then linearly interpolated to obtain the overall language model probability. The idea is that the training corpus is divided into a pre-defined number of topic clusters. Then the n-gram model can be trained for each component. The interpolation weights of the component language models are created in such a way that the interpolated model best matches with the adaptation corpus:

$$P(w_i|h) = \sum_{k=1}^{K} \lambda_{A,t_k} P_{B,t_k}(w_i|h) \tag{2.30}$$

where λ_{A,t_k} is the interpolation weight of the t_k^{th} component model and $P_{B,t_k}(w_i|h)$ is the t_k^{th} component model. There are many methods that have investigated about how the topic

clusters are formed and how the interpolation coefficients are computed [4, 24, 29, 59].

2.6.5 Explicit Topic Models

Topic information is included indirectly in mixture modeling by using topic-specific LMs that constitute the overall background model. Topic information can also be incorporated directly as:

$$P(w_i|d_l) = \sum_{k=1}^{K} P(w_i|t_k)P(t_k|d_l) \tag{2.31}$$

where $P(w_i|t_k)$ are the word probabilities for topics and $P(t_k|d_l)$ are the topic factors for the document d_l. The main difference of this kind of topic contribution with the clustering approaches for example in [59] is that we do not need to assume that a document belongs to exactly one topic. The example of direct contribution of topic information can be seen in the language modeling using LSA [10], PLSA [33], and LDA [13]. However, as these model do not consider the use of syntax or ignore the word order, we need to integrate these models with a background n-gram model to capture the local lexical regularities of the language.

2.7 Performance Measurements

In the literature, there are two metrics that are widely used to measure the performance of a speech recognition system. The most common metric is the perplexity on test data, which is an information theoretic measure of *cross entropy*, and the second one is the word error rate, which is the most popular method for rating a speech recognition system.

2.7.1 Perplexity

Perplexity is the most common metric for evaluating a language model for speech recognition. It can be calculated efficiently and it does not require a speech recognizer. The role of an n-gram language model is to measure how well it predicts the sentences from the test set. For example, let a test set have T sentences as $s^1,\ldots,s^T$, where $s^t = w_1^t \ldots w_{s^t}^t$. The probability of the test set can be computed as the product of the probability of individual sentence probabilities:

$$P(T) = \prod_{t=1}^{T} P(s^t) = \prod_{t=1}^{T} \prod_{i=1}^{s^t} P(w_i^t|h^t) = \prod_{i=1}^{N} P(w_i|h)$$

where we mapped the words w_i^t and histories h^t of all the sentences of the test set into a sequence w_i and h_i. Here, N is the length of the text T measured in words [55].

Now, we can derive a compression algorithm that can encode the text T using $-log_2P(T)$ bits. The *cross entropy* $H(T)$ of the language model $P(w_i|w_{i-n+1},\ldots,w_{i-1})$ on text data T can be defined as:

$$H(T) = -\frac{1}{N}log_2P(T) = -\frac{1}{N}\sum_{i=1}^{N} log_2P(w_i|h).$$

The perplexity $PP(T)$ of a language model $P(T)$ is related to the cross entropy and is the reciprocal of the average probability assigned by the model to each word of the test set T. This is known as test set perplexity.

$$PP(T) = 2^{H(T)} = \frac{1}{\left(\prod_{i=1}^{N} P(w_i|h)\right)^{\frac{1}{N}}}$$

The perplexity can be roughly interpreted as the geometric mean of the branching factor of the test set when applied to the language model. In general, it is true that a lower perplexity model provides better speech recognition performance as the perplexity is a statistically weighted branching measure of the test set [57]. However, in reducing the errors in a speech recognition system, one should have to consider the acoustic similarities between words and a language model that will help to discriminate acoustically similar words. Moreover, the accuracy of the speech recognition system depends not only on the probability of the correct hypotheses but also on the probability of the other candidate hypotheses. By the way, these probabilities are ignored by the perplexity measure [23]. That's why the performance of a speech recognition system should not be measured by perplexity alone.

2.7.2 Word Error Rate

The most popular metric for rating a speech recognition system is the word error rate (WER). It is derived from the *Levenshtein distance*, working at the word level instead of the phoneme level. WER is defined as the total number of errors (word insertions (I), deletions (D) and substitutions (S)) divided by the total number of words (N) actually spoken, i.e.,

$$WER = \frac{(S+D+I)}{N}.$$

Sometimes word recognition rate (WRR) is used when reporting the performance of a speech recognition system. It records the proportion of words that was correctly recognized, and therefore ignores insertion errors.

$$WRR = 1 - WER = \frac{(N-S-D-I)}{N} = \frac{H-I}{N}$$

where H is $N-(S+D)$, the number of correctly recognized words.

2.8 Decoding

In section 2.1, we saw how a language model is combined with the acoustic model in a speech recognition system. In this section, we describe how a language model is used practically to decode a speech signal into one or more hypothesised transcriptions.

The decoding process of a speech recognition system is to find a sequence of words whose corresponding acoustic and language models best match the input signal. The process is described in Figure 2.5.

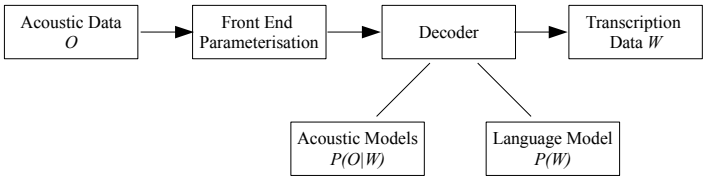

Fig. 2.5 A typical speech recognition system

The input to the system is acoustic data in the form of a waveform. This data is processed by the system's front end to create a set of feature vectors that capture the spectral properties of the speech signal at discrete time intervals. These feature vectors are then passed to the decoder. The role of the decoder is to search for the string of words that best matches the feature vectors, i.e., to find W' such that

$$W' = \underset{W}{\operatorname{argmax}} P(W|O) = \underset{W}{\operatorname{argmax}} P(W)P(O|W). \tag{2.32}$$

It is not necessary that the acoustic model $P(O|W)$ should be a word model. In a large vocabulary speech recognition system, sub-word models such as phoneme models, demi-syllables and syllables are often used. In that case, the word models $P(O|W)$ are obtained by concatenating the sub-word models according to the pronunciation dictionary of the words. When word models are available, the speech recognition becomes a search problem. It searches the sequence of word models that best describes the input waveform against the word models.

As the search space is the set of all possible word strings, it is necessary to find the methods that can reduce the search space to make the search more tractable. Therefore, the decoding process of the system is often considered as a search process. In general, there are two main search approaches: $depth\text{-}first$ and $breadth\text{-}first$. In $depth\text{-}first$ search, the most promising hypothesis is pursued until the end of the speech is reached. The examples

of a *depth-first* decoder are stack-decoders and A^* decoders [60, 65, 84, 85]. In *breadth-first* designs, all hypotheses are pursued in parallel. They exploit the Bellman's optimality principle and are often called Viterbi decoding. In a large vocabulary recognition (LVR) system, the search space is complex and it is necessary to prune the search space. This typically is done by a process called a beam search [38, 92]. The hidden Markov model toolkit (HTK) decoder uses the beam search and Viterbi decoding [104].

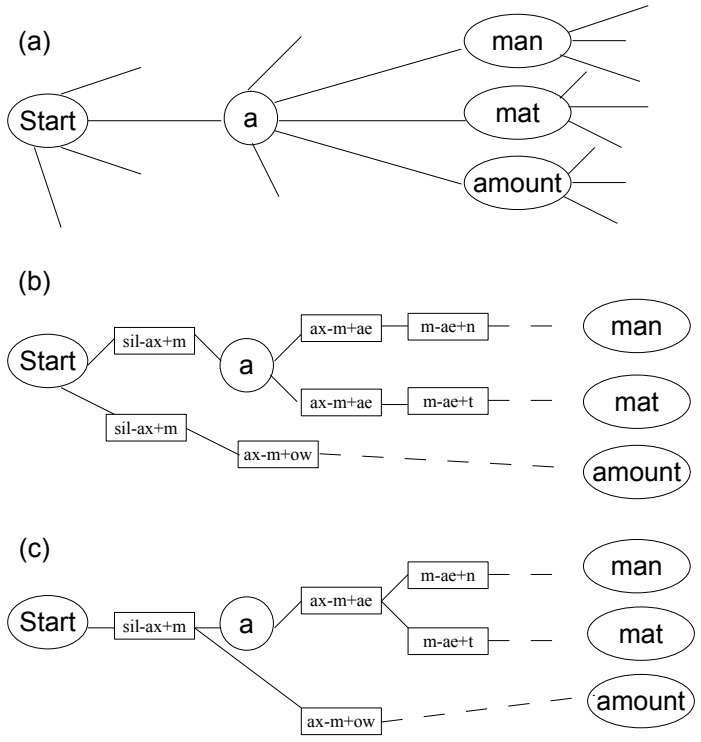

Fig. 2.6 Fragment of decoder network

The decoding problem can be easily described by using a branching tree network. In Figure 2.6(a), at the start node there is a branch to every possible start word. After that, all first words are then connected to all possible following words and so on. It is clear that this tree will become very large for a LVR system. In a small vocabulary system, all the words can be put in parallel and a loop placed around them. However, this arrangement does not allow a trigram model as the available history is limited to one word. Next, consider each word in Figure 2.6(a) is replaced by the sequence of models representing its pronunciation. This is shown in Figure 2.6(b). The models can be joined in parallel within the word if there are multiple pronunciations. All the identical phone models in identical context are then merged. This is shown in Figure 2.6(c). Here, we can notice the cross-word triphones

significantly limit the amount of sharing models possible [104].

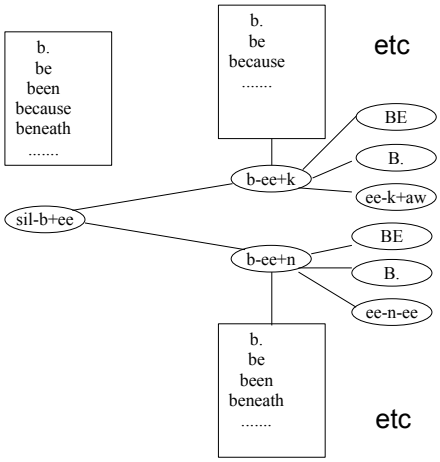

Fig. 2.7 Early application of language models

Therefore, the above network can be viewed as a branching tree of HMM state nodes connected by state transitions and word-end nodes connected by word transitions. Any path from the start node to any point in the tree can be evaluated by adding all the log state transition probabilities, all the log state output probabilities and the log language probabilities. Such a path can be represented by a movable token placed in the node at the end of the path. The token has a score which is the total log probability up to that point and a history that records the sequence of word-end nodes that the token has passed through. Any path can be extended by moving the token from its current node to the adjoining node and updating its score according to its state transition probability, state output probability and the language model probability. So, it can considered as a token passing algorithm. Here, a token is placed at the start node of the tree. Then, for each input acoustic vector, every token is copied into all connecting nodes and the scores are updated. If more than one token lands in a node, only the best scoring node is kept. When all the acoustic vectors have been processed, the word end nodes are scanned and the token with the highest score represents the best path and the most likely word sequence. Although this technique helps to find the best possible path, it takes too much space and time to compute the path. So, pruning is employed to make the algorithm more tractable. In every time frame, the best score in any token is noted and any token whose score lies more than a beam width below this best score is destroyed. Therefore, only a part of the branching tree described above is needed at one time as only the active tokens that are lying within the beam width need to be kept in memory. As tokens move forward, a new tree structure is created in front of

them and the old structure behind is destroyed. So, it is necessary to apply pruning as soon as possible. However, as the identity of each new word is not known until we reach its end node, the merging of phone models in the branching tree causes a problem. The language model provides a powerful constraint that needs to be applied as soon as it is practical in order to keep the number of tokens as small as possible. The HTK decoder allows a list of possible current words in every token (Figure 2.7). When the token is reached at the end of the word, the list is minimized and it contains just a single word. Tokens then receive a language model score, which equals to the most likely word in the current list. As this gets updated on every model transition, the token gets pruned accordingly [104].

2.9 Experimental Tools and Data Sets

We evaluated the LM adaptation approaches using the Wall Street Journal (WSJ) corpus [71]. The SRILM toolkit [94] and the HTK toolkit [105] are used for generating the LMs and computing the WER respectively. The acoustic model from [98] is used in our experiments. The acoustic model is trained by using all WSJ and TIMIT [32] training data, the 40 phones set of the CMU dictionary [2], approximately 10000 tied-states, 32 Gaussians per state and 64 Gaussians per silence state. The acoustic waveforms are parameterized into a 39-dimensional feature vector consisting of 12 cepstral coefficients plus the 0^{th} cepstral, delta and delta delta coefficients, normalized using cepstral mean subtraction $(MFCC_{0-D-A-Z})$. We evaluated the cross-word models. The values of the word insertion penalty, beam width, and the language model scale factor are -4.0, 350.0, and 15.0 respectively [98]. The development test set is the **si_dt_05.odd** (248 sentences, 4074 words) and the evaluation test sets are the Nov' 92 and Nov'93 test data from the November 1992 (330 sentences, 5353 words) and November 1993 (215 sentences, 3849 words) ARPA CSR benchmark test data respectively for 5K vocabularies [71, 101].

2.10 Summary

In this chapter, the use of language models in a speech recognition system is described. The importance of statistical language models over grammar-based models, and smoothing of language models are also discussed. The history of semantic analysis techniques are briefly stated. Then, the importance of adaptive language models and some adaptation techniques that are used in this thesis are explained. A brief overview of the decoding technique in a

speech recognizer in HTK is also described. Finally, the experimental tools, data sets, and measurement metrics that we used in this thesis are described.

Chapter 3

LDA-based LM Adaptation Using LSM

In this chapter, we present unsupervised language model (LM) adaptation approaches using latent Dirichlet allocation (LDA) and latent semantic marginals (LSM). The LSM are the unigram probability distribution over words that are calculated using LDA-adapted unigram models. The LDA model is used to extract topic information from a training corpus in an unsupervised manner. The LDA model yields a document-topic matrix that describes the number of words assigned to topics for the documents. A hard-clustering method is applied on the document-topic matrix of the LDA model to form topics. An adapted model is created by using a weighted combination of the n-gram topic models. The interpolation of the background model and the adapted model gives further improvement. We modify the above models using the LSM. The LSM are used to form a new adapted model by using the minimum discriminant information (MDI) adaptation approach called unigram scaling, which minimizes the distance between the new adapted model and the other model. We perform experiments using the '87-89 Wall Street Journal (WSJ) corpus incorporating a multi-pass continuous speech recognition (CSR) system. In the first pass, we used the background n-gram language model for lattice generation and then we apply the LM adaptation approaches for lattice rescoring in the second pass [49].

3.1 Introduction

The simple technique to form a topic from an unlabeled corpus is to assign one topic label to a document [58]. This hard-clustering strategy is used with leveraging LDA and named entity information to form topics [72, 73]. Here, topic-specific n-gram language models are created and joined with proper mixture weights for adaptation. The adapted model is then interpolated with the background model to capture the local lexical regularities. The

component weights of the n-gram topic models were created by using the word counts of the latent topic of the LDA model. However, these counts are best suited for the LDA unigram topic models. A unigram count weighting approach [39] for the topics generated by hard-clustering has shown better performance over the weighting approach described in [72, 73]. An extension of the unigram weighting approach [39] was proposed in [40] where the weights of the n-gram topic models are computed by using the n-gram count of the topics generated by a hard-clustering method. The adapted n-gram model is scaled by using the LDA-adapted unigram model called latent semantic marginals (LSM) [96] and outperforms a traditional unigram scaling of the background model using the above marginals [41]. Here, the unigram scaling technique [68] is applied where a new adapted model is formed by using a minimum discriminant information (MDI) approach that minimizes the KL divergence between the new adapted model and the adapted n-gram model, subject to a constraint that the marginalized unigram distribution of the new adapted model is equal to the LSM. In this chapter, we present an extension to the previous works [40, 41] where we apply the unigram scaling technique to the interpolation of the background and the adapted n-gram model and note better results over the previous works. In addition, we perform all the experiments using different corpus sizes ('87 WSJ corpus (17 million words) and '87-89 WSJ corpus (37 million words)) instead of using only the 1 million words WSJ training transcription data used in [40, 41] and using different test sets. Also, we use various topic sets instead of using a single topic set.

3.2 Mixture Language Model Using N-gram Weighting

3.2.1 Topic Clustering

We have used the MATLAB topic modeling toolbox [93] for LDA analysis that uses a Gibbs sampler for parameter estimation. We have formed the word-topic matrix, WP, and the document-topic matrix, DP, using LDA analysis [93]. In the WP matrix, an element $WP(w_i,t_k)$ shows the number of occurrences of word w_i in topic t_k over the training set. In the DP matrix, an element $DP(d_j,t_k)$ contains the total number of occurrences of words in document d_j that are from a topic $t_k(k = 1,2,\ldots,K)$.

We have formed the topic set by applying a hard-clustering approach [58, 72, 73] to the DP matrix. Here, we assign a document d_j to a topic t_k as:

$$t_k = \underset{1\leq t_k\leq K}{\arg\max} DP(d_j,t_k) \tag{3.1}$$

i.e., a document is assigned to a topic from which it takes the maximum number of words. Therefore, all the training documents are assigned to K topics. The n-gram topic LM's for K topics are trained. The models are then combined with proper mixture weights to form an adapted model (see section 3.2.2). The idea is portrayed in Figure 3.1.

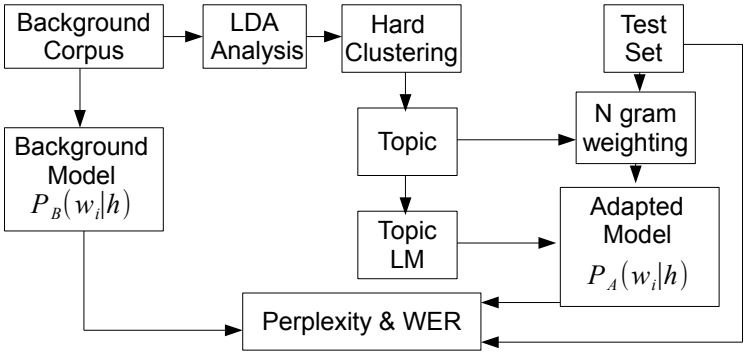

Fig. 3.1 Topic clustering and LM adaptation using n-gram weighting

3.2.2 Adapted Model Generation

A document is generated by a mixture of topics in the LDA model. So, for a test document $d_t = w_1, \ldots, w_N$, a dynamically adapted n-gram model can be created by using a mixture of n-gram topic LMs as:

$$P_A(w_i|h) = \sum_{k=1}^{K} \phi_{t_k} P_{t_k}(w_i|h) \tag{3.2}$$

where $P_{t_k}(w_i|h)$ is the t_k^{th} n-gram topic model, ϕ_{t_k} is the t_k^{th} mixture weight, and h is the preceding n-1 words of the current word w_i. To find topic mixture weight ϕ_{t_k}, the n-gram count of the topics, generated by Equation 3.1, is used [40]. Therefore,

$$\phi_{t_k} = \sum_{j=1}^{N_n} P(t_k|w_{jn}, \ldots, w_{j1}) P(w_{jn}, \ldots, w_{j1}|d_t) \tag{3.3}$$

with

$$P(t_k|w_{jn}, \ldots, w_{j1}) = \frac{C_{t_k}(w_{jn}, \ldots, w_{j1}, t_k)}{\sum_{k=1}^{K} C_{t_k}(w_{jn}, \ldots, w_{j1}, t_k)} \tag{3.4}$$

$$P(w_{jn}, \ldots, w_{j1}|d_t) = \frac{C(w_{jn}, \ldots, w_{j1})}{\sum_{j=1}^{N_n} C(w_{jn}, \ldots, w_{j1})} \tag{3.5}$$

where $C_{t_k}(w_{jn},\ldots,w_{j1},t_k)$ describes the number of times the n-gram $(w_{jn},\ldots,w_{j1})$ is seen in topic t_k, which is created by Equation 3.1. $C(w_{jn},\ldots,w_{j1})$ and N_n are the counts of the n-gram $(w_{jn},\ldots,w_{j1})$ and the number of different n-grams respectively in document d_t.

The adapted (A) n-gram model is then interpolated with the background (B) n-gram model to capture the local constraints using linear interpolation as:

$$P_L(w_i|h) = \lambda P_B(w_i|h) + (1-\lambda)P_A(w_i|h), \tag{3.6}$$

where λ is an interpolation weight.

3.3 LM Adaptation using Latent Semantic Marginals (LSM)

3.3.1 LSM

We computed the LSM by using the technique described in [96]. At first the automatic transcription (recognition results after first pass decoding) is treated as a single document [96]. Then, a Gibbs sampler is applied for the test document to estimate the posterior over the topic mixture weights [53]. The LDA-adapted marginal is then computed as follows [96]:

$$P_{lda}(w_i) = \sum_{k=1}^{K} P(w_i|t_k,\beta).\frac{\gamma_{t_k}}{\sum_{k=1}^{K}\gamma_{t_k}}, \tag{3.7}$$

where γ_{t_k} is the weight of topic t_k for the test document d_t obtained after LDA inference and computed as:

$$\gamma_{t_k} = \frac{DP(d_t,t_k)+\alpha}{DP(d_t,.)+K\alpha}, \tag{3.8}$$

where $DP(d_t,.)$ is the total occurrences of words in document d_t in all topics. K, $DP(d_t,t_k)$ and α are defined as above. $P(w_i|t_k,\beta)$ is the probability of word w_i for topic t_k obtained after applying LDA over the training set and is computed as [37, 53]:

$$P(w_i|t_k,\beta) = \frac{WP(w_i,t_k)+\beta}{WP(.,t_k)+V\beta}, \tag{3.9}$$

where $WP(.,t_k)$ is the total count of words in topic t_k, V is the size of the vocabulary, and β is defined as in section 2.5.4.

3.3.2 New Adapted Model Generation Using LSM

The unigram scaling technique [34, 68] forms an adapted model by minimizing the KL-divergence between the background model and the adapted model subject to the marginalization constraint for each word w_i in the vocabulary [96] as:

$$\sum_h P_{A_1}(h).P_{A_1}(w_i|h) = P_{lda}(w_i). \tag{3.10}$$

The constraint optimization problem has close connection to the maximum entropy approach [88], which provides that the adapted model is a re-scaled version of the background model:

$$P_{A_1}(w_i|h) = \frac{\delta(w_i)}{Z(h)}.P_{B/A/L}(w_i|h) \tag{3.11}$$

with

$$Z(h) = \sum_{w_i} \delta(w_i).P_{B/A/L}(w_i|h) \tag{3.12}$$

where $P_{A_1}(w_i|h)$ is defined as the new adapted model, $P_{B/A/L}(w_i|h)$ is the background, adapted (Equation 3.2) or the interpolated (Equation 3.6) model. $Z(h)$ is a normalization term, which guarantees that the total probability sums to unity, and $\delta(w_i)$ is a scaling factor that is usually approximated as:

$$\delta(w_i) \approx \left(\frac{P_{A_1}(w_i)}{P_{B/A/L}(w_i)} \right)^{\mu}, \tag{3.13}$$

where μ is a tuning factor between 0 and 1. In this paper, we used $\mu = 0.5$ [41, 96]. To compute the normalization term $Z(h)$, we used the same procedure as [41, 68, 96]. To accomplish this, an additional constraint is considered where the total probability of the seen transitions is unchanged:

$$\sum_{w_i:seen(h,w_i)} P_{A_1}(w_i|h) = \sum_{w_i:seen(h,w_i)} P_{B/A/L}(w_i|h). \tag{3.14}$$

The new adapted LM is then computed as:

$$P_{A_1}(w_i|h) = \begin{cases} \frac{\delta(w_i)}{Z_s(h)}.P_{B/A/L}(w_i|h) \text{ if } (h,w_i) \text{ exists} \\ bow(h).P_{A_1}(w_i|\hat{h}) \text{ otherwise} \end{cases}$$

where

$$Z_s(h) = \frac{\sum_{w_i:seen(h,w_i)} \delta(w_i).P_{B/A/L}(w_i|h)}{\sum_{w_i:seen(h,w_i)} P_{B/A/L}(w_i|h)}$$

and

$$bow(h) = \frac{1-\sum_{w_i:seen(h,w_i)} P_{B/A/L}(w_i|h)}{1-\sum_{w_i:seen(h,w_i)} P_{A_1}(w_i|\hat{h})}.$$

Here, $Z_s(h)$ is used to compute the normalization similar to Equation 3.12 except the summation is performed only on the seen alternative words with the same word history h in the LM [96], $bow(h)$ is the back-off weight of the context h to ensure that $P_{A_1}(w_i|h)$ sums to unity and $\hat{h}$ is the reduced word history of h. The idea is described in Figure 3.2.

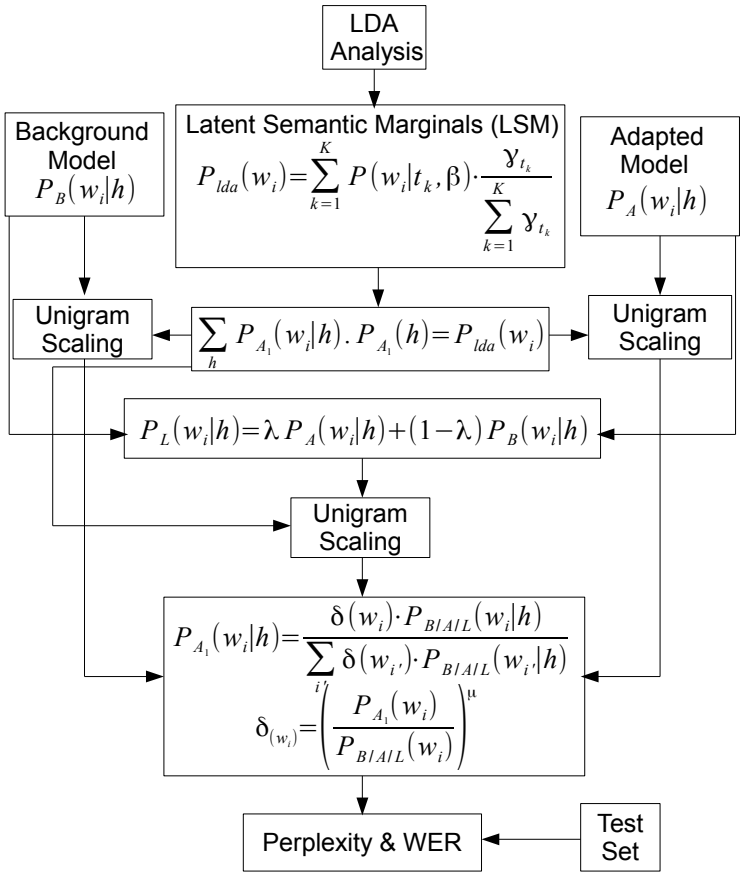

Fig. 3.2 New adapted model generation Using LSM

3.4 Experiments

3.4.1 Data and Parameters

The '87-89 WSJ corpus is used to train the tri-gram background model and the tri-gram topic models using the back-off version of the Witten-Bell smoothing. The language models are closed-vocabulary language models, i.e., the models are generated using the n-gram counts without considering n-grams with unknown words. To reduce the computational cost, we incorporated the cutoffs 1 and 3 on the bi-gram and tri-gram counts respectively. The LDA and the language models are trained using the WSJ 20K non-verbalized punctuation closed vocabulary. We define the α and β for LDA analysis as $50/K$ and 0.01 respectively [37, 53]. The interpolation weights ϕ and λ are computed using the *compute-best-mix* program from the SRILM toolkit. They are tuned on the development test set. The latent semantic marginals (LSM) are created by the automatic transcription. Automatic transcription is the recognition result obtained after first-pass decoding of the evaluation data. The results of the experiments are noted on the evaluation test set. The bold values describe the best results among all topic sizes for the corresponding model.

3.4.2 Unsupervised LM Adaptation Using N-gram Weighting

The perplexities on the November 1993 and November 1992 test sets for different sizes of corpus are described in Tables 3.1 and 3.2 respectively.

Table 3.1 Perplexity results of the tri-gram language models using n-gram weighting on November 1993 test data

Language Model	Topic 25		Topic 50		Topic 75	
	'87	'87-89	'87	'87-89	'87	'87-89
Background (B)	101.7	88.9	101.7	88.9	101.7	88.9
Adapted (A) Model	**98.8**	**87.9**	105.3	91.7	107.5	89.6
(B+A) Model	82.1	73.5	81.5	**73.0**	**81.2**	73.4

From Tables 3.1 and 3.2, we can note that for the stand-alone adapted (A) model, only models with topic size 25 give better results than other topic sizes. This is due to the limitation of the SRILM toolkit that can mix only 10 models at a time and the lack of the local lexical regularities of the background model. However, the interpolation of the

Table 3.2 Perplexity results of the tri-gram language models using *n*-gram weighting on November 1992 test data

Language Model	Topic 25		Topic 50		Topic 75	
	'87	'87-89	'87	'87-89	'87	'87-89
Background (B)	85.4	71.0	85.4	71.0	85.4	71.0
Adapted (A) Model	89.4	78.3	95.6	82.1	100.4	78.8
(B+A) Model	72.2	62.1	71.6	**61.6**	**71.5**	61.9

background and the adapted models (B+A) outperforms all the other above approaches for all topic and corpus sizes.

The WER results of the experiments on different test sets for different corpus sizes are described in Tables 3.3 and 3.4 respectively.

Table 3.3 WER results (%) of the tri-gram language models using *n*-gram weighting on November 1993 test data

Language Model	Topic 25		Topic 50		Topic 75	
	'87	'87-89	'87	'87-89	'87	'87-89
Background (B)	9.2	8.3	9.2	8.3	9.2	8.3
Adapted (A) Model	8.4	**7.4**	8.4	7.7	**8.3**	7.9
(B+A) Model	**7.6**	**7.2**	7.8	7.3	**7.6**	7.5

Table 3.4 WER results (%) of the tri-gram language models using *n*-gram weighting on November 1992 test data

Language Model	Topic 25		Topic 50		Topic 75	
	'87	'87-89	'87	'87-89	'87	'87-89
Background (B)	4.8	4.6	4.8	4.6	4.8	4.6
Adapted (A) Model	5.3	4.5	5.6	4.7	5.5	**4.3**
(B+A) Model	4.2	4.0	4.2	**3.9**	**4.1**	**3.9**

From Tables 3.3 and 3.4, we can note that the stand-alone adapted (A) model for all topic and corpus sizes outperforms the background model for the November 1993 test set, and the best WERs are achieved for topic size 75 using the '87 corpus (9.8% (9.2% to 8.3%)) and topic size 25 using the '87-89 corpus (10.8% (8.3% to 7.4%)). For the November 1992

test set, the best result is obtained only for the '87-89 corpus using topic size 75 (6.5% (4.6% to 4.3%)). Here the WERs are over the background model. The interpolation of the background (B) and the adapted model (A) outperforms all the other above models [40].

3.4.3 New Adapted Model Using LSM

In [41], the unigram scaling of the adapted (A) model through MDI adaptation using LSM was proposed, which outperforms the MDI adaptation of the background (B) model using LSM [96]. Here, we introduce the MDI adaptation to the (B+A) model using the LSM. The idea of MDI adaptation is to minimize the Kullback-Leibler (KL) distance between the adapted model and the other model [34, 68]. The perplexity results of the experiments using the November 1993 and November 1992 test sets for different sizes of corpus are explained in Table 3.5 and 3.6 respectively.

Table 3.5 Perplexity results on the November 1993 test data using tri-gram language models obtained by using LSM

Language Model	Topic 25		Topic 50		Topic 75	
	'87	'87-89	'87	'87-89	'87	'87-89
Background (B)	101.7	88.9	101.7	88.9	101.7	88.9
Adaptation of (B) model	98.8	85.9	97.8	85.7	**97.2**	**84.8**
Adaptation of A model	**97.9**	**86.7**	103.5	89.9	105.1	87.0
Adaptation of (B+A) model	80.7	72.0	79.3	71.0	**78.5**	**70.8**

Table 3.6 Perplexity results on the November 1992 test data using tri-gram language models obtained by using LSM

Language Model	Topic 25		Topic 50		Topic 75	
	'87	'87-89	'87	'87-89	'87	'87-89
Background (B)	85.4	71.0	85.4	71.0	85.4	71.0
Adaptation of (B) model	84.2	70.2	83.9	69.9	**83.6**	**69.7**
Adaptation of A model	88.5	78.2	94.0	81.4	98.3	**77.3**
Adaptation of (B+A) model	71.4	61.6	70.4	60.8	**70.1**	**60.7**

From Tables 3.5 and 3.6, we can note that all the models outperform the background model except for the adapted model using the November 1992 test set. The unigram scaling

of the interpolation of the background (B) and the adapted (A) models outperforms the unigram scaling of the background model [96], the unigram scaling of the adapted model [41], and the (B+A) model [40] for all topic and corpus sizes.

The WER results of the experiments on different test sets for different corpus sizes are described in Tables 3.7 and 3.8.

Table 3.7 WER results (%) on the November 1993 test data using tri-gram language models obtained by using LSM

Language Model	Topic 25		Topic 50		Topic 75	
	'87	'87-89	'87	'87-89	'87	'87-89
Background (B)	9.2	8.3	9.2	8.3	9.2	8.3
Adaptation of (B) model	9.2	**8.0**	**9.0**	**8.0**	9.1	8.1
Adaptation of A model	**8.4**	**7.5**	8.5	7.7	8.6	7.9
Adaptation of (B+A) model	**7.6**	**6.9**	7.7	7.2	**7.6**	7.2

Table 3.8 WER results (%) on the November 1992 test data using tri-gram language models obtained by using LSM

Language Model	Topic 25		Topic 50		Topic 75	
	'87	'87-89	'87	'87-89	'87	'87-89
Background (B)	4.8	4.6	4.8	4.6	4.8	4.6
Adaptation of (B) model	4.8	4.6	4.8	4.7	4.9	4.6
Adaptation of A model	5.3	4.5	5.6	4.6	5.6	**4.4**
Adaptation of (B+A) model	4.2	3.8	4.2	3.8	**4.1**	**3.7**

From Tables 3.3, 3.4, 3.7, and 3.8, we can note that the unigram scaling of the adapted (A) models using LSM outperforms the unigram scaling of the background (B) model using LSM for all topic sizes except for the November 1992 test set using the '87 corpus [41]. For the '87 corpus, the proposed unigram scaling of the B+A model does not give any improvement over the *n*-gram-weighting [40] adaptation approaches except for the November 1993 test set with topic size 50. However, the proposed unigram scaling of the (B+A) models outperforms all the other above approaches [40, 41, 96] and the best results obtained for topic sizes 25 and 75 for the November 1993 and November 1992 test sets respectively using the '87-89 corpus. For the November 1993 test set using topic size 25 and the '87-89 corpus, it gives about 16.9% (8.3% to 6.9%), 13.7% (8.00% to 6.9%), 8.0% (7.5% to 6.9%), and

4.2% (7.2% to 6.9%) over the background model, the unigram scaling of the background model [96], the unigram scaling of the adapted model [41], and the interpolation of the background and the adapted models [40] respectively. For the November 1992 test set using topic size 75 and the '87-89 corpus, it gives about 19.6% (4.6% to 3.7%), 19.6% (4.6% to 3.7%), 15.9% (4.4% to 3.7%), and 5.1% (3.9% to 3.7%) over the background model, the unigram scaling of the background model [96], the unigram scaling of the adapted model [41], and the interpolation of the background and the adapted models [40] respectively. From the above experiments, we can note that adding more data we get better improvement for all topic sizes and test sets.

3.4.4 Statistical Significance and Error Analysis

The significance improvement in WER is done by using a matched-pair-test where the misrecognized words in each test utterance are counted. The p-values of the proposed unigram scaling of the B+A model are measured relative to the background model, the unigram scaling of the background model [96], the unigram scaling of the adapted model [41] and the interpolation of the background model and the adapted models [40], respectively. For the November 1993 test set using topic size 25 and the '87-89 corpus, the p-values are $4.0E$-9, 0.00081, $7.4E$-8, and 0.00175. For the November 1992 test set using topic size 75 and the '87-89 corpus, the p-values are $4.9E$-6, 0.0071, $8.3E$-7, and 0.00989. At a significance level of 0.01, the proposed approach is significantly better than the other models.

Tables 3.9 and 3.10 are used to describe the ASR results for deletion (D), substitution (S), and insertion (I) errors, and also the correctness ($Corr$) and accuracy (Acc) of the trigram language models. From the tables, we can note that the proposed unigram scaling of the B+A model reduces all types of errors, and improves correctness and accuracy relative to the background and other models [40, 41, 96]. Using the proposed approach, the deletion and insertion errors do not change much compared to the background and other models. Therefore, the substitution errors play an important role to improve the performance, i.e., more words can be recognized accurately using the proposed method than the background and other models. We can also note that the improvement of the A model can help to reduce the existing errors in the current approach.

Table 3.9 For the November 1993 test set using topic size 25 and the '87-89 corpus, ASR results for deletion (*D*), substitution (*S*), and insertion (*I*) errors, and also the correctness (*Corr*) and accuracy (*Acc*) of the tri-gram language models

Language Model	*D*	*S*	*I*	*Corr*	*Acc*
Background (B)	0.010	0.064	0.009	0.926	0.917
Adaptation of B model using LSM	0.010	0.061	0.008	0.929	0.920
Adaptation of A model using LSM	0.010	0.058	0.006	0.931	0.925
(B+A) model	0.010	0.055	0.007	0.935	0.928
Adaptation of (B+A) model using LSM	0.009	0.054	0.006	0.937	0.931

Table 3.10 For the November 1992 test set using topic size 75 and the '87-89 corpus, ASR results for deletion (*D*), substitution (*S*), and insertion (*I*) errors, and also the correctness (*Corr*) and accuracy (*Acc*) of the tri-gram language models

Language Model	*D*	*S*	*I*	*Corr*	*Acc*
Background (B)	0.003	0.033	0.010	0.965	0.954
Adaptation of B model using LSM	0.003	0.033	0.010	0.964	0.954
Adaptation of A model using LSM	0.003	0.030	0.011	0.967	0.956
(B+A) model	0.002	0.027	0.009	0.970	0.961
Adaptation of (B+A) model using LSM	0.002	0.026	0.009	0.973	0.963

3.5 Summary

In this chapter, we have proposed unsupervised language model adaptation approaches using LDA, LSM, and unigram scaling. A hard-clustering approach is applied on the document-topic matrix obtained in LDA analysis to form a topic set. Then, an *n*-gram weighting approach is used to compute the mixture weights of the component topic models. An adapted model is computed using the weighted combination of *n*-gram topic models. We have performed the experiments for various topic sizes. We have formed new adapted models by modifying the adapted models using unigram scaling, which minimizes the KL divergence between the new adapted models and the adapted models subject to a constraint that the marginalized unigram probability distributions of the new adapted models are equal to the unigram probability distributions estimated by using the LDA models, called LSM. We created LSM for the automatic (recognition results after first-pass decoding of the evaluation test set) transcriptions. We have interpolated the background model with the adapted model to capture the local lexical regularities and scaled the interpolated model using the above

LSM. We performed the experiments using the WSJ corpus with varying sizes on two different test sets and used the LM adaptation approaches in the second pass of decoding. We compared our approaches with traditional MDI adaptation approaches. We have seen that our proposed approach gives significant reductions in perplexity and WER over the traditional approaches used in the literature. Moreover, we have found that adding more data helps to get better improvement.

Chapter 4

Topic n-gram Count LM

In this chapter, we introduce a novel language model (LM) adaptation approach using the latent Dirichlet allocation (LDA) model. Observed n-grams in the training set are assigned to topics using soft and hard clustering. In soft clustering, each n-gram is assigned to topics such that the total count of that n-gram for all topics is equal to the global count of that n-gram in the training set. Here, the normalized topic weights of the n-gram are multiplied by the global n-gram count to form the topic n-gram count for the respective topics. In hard clustering, each n-gram is assigned to a single topic with the maximum fraction of the global n-gram count for the corresponding topic. Here, the topic is selected using the maximum topic weight for the n-gram. The topic n-gram count LMs are created using the respective topic n-gram counts and adapted by using the topic weights of a development test set. We compute the average of the confidence measures: the probability of word given topic and the probability of topic given word. The average is taken over the words in the n-grams and the development test set to form the topic weights of the n-grams and the development test set respectively [42].

4.1 Introduction

We propose a new LM adaptation approach by considering the features of the LDA model. As a bag-of-words model, each word is independent in LDA. Therefore, each word has equal weight in determining the topic mixtures. Also, latent topics are independent of each other in the LDA topic set. So, we induce a constraint that the total count of an n-gram for all topics is equal to the count of that n-gram in the training set. Here, we compute the topic mixture weights of each n-gram in the training set using the probability of word w_i given topic t_k, $P(w_i|t_k)$ and the probability of topic t_k given word w_i, $P(t_k|w_i)$ as confidence

measures for the words in the *n*-gram under different LDA latent topics. The normalized topic mixture weights are then multiplied by the global count of the *n*-gram to determine the topic *n*-gram count for the respective topics. In soft clustering, each *n*-gram is assigned to all topics with the corresponding topic *n*-gram count. In hard clustering, each *n*-gram with the maximum fraction of the global count is assigned to a single topic, where the topic is selected by the maximum topic weights for the *n*-gram. The topic *n*-gram count LMs are then created using the respective topic *n*-gram counts and adapted by using the topic mixture weights obtained by averaging the confidence measures over the seen words of a development test set. The complete idea is described in Figure 4.1.

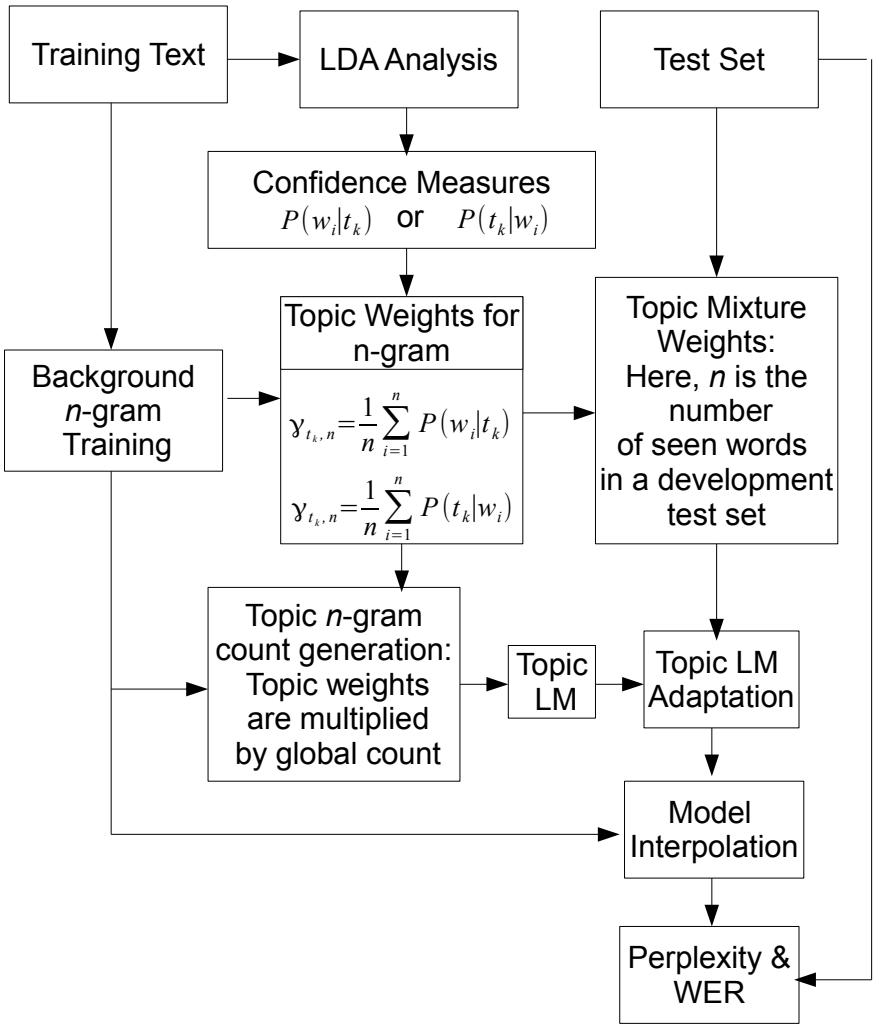

Fig. 4.1 Topic *n*-gram count LM Adaptation

4.2 LDA Training

We have used the MATLAB topic modeling toolbox [93] to get the word-topic matrix WP, using LDA. Here, the words correspond to the words used in LDA analysis. In the WP matrix, an entry $WP(w_i,t_k)$ represents the number of times word w_i has been assigned to topic t_k over the training set.

The probability of word w_i under LDA latent topic t_k is computed as [37, 53]:

$$P(w_i|t_k,\beta) = \frac{WP(w_i,t_k)+\beta}{WP(.,t_k)+V\beta}, \tag{4.1}$$

where $WP(.,t_k)$ is the total count of words in topic t_k, V is the total number of words, and β is defined as in section 2.5.4.

4.3 Topic N-gram Count Language Model

4.3.1 Language Model Generation

We computed the topic mixture weights of the background n-grams in the training set. The topic weights are used to determine the counts of the n-grams for the corresponding topics. Since each word is an independent and equally reliable observation under the LDA model, the probability of each word has equal weight in computing the topic mixtures [52]. Using these features of the LDA model, we proposed two confidence measures (the probability of word w_i given topic t_k ($P(w_i|t_k)$) in equation 4.2 and the probability of topic t_k given word w_i ($P(t_k|w_i)$) in equation 4.3) to compute the topic mixture weights for each n-gram:

$$\gamma_{t_k} = \frac{1}{n}\sum_{i=1}^{n} P(w_i|t_k), \tag{4.2}$$

$$\gamma_{t_k} = \frac{1}{n}\sum_{i=1}^{n} P(t_k|w_i), \tag{4.3}$$

where γ_{t_k} is the weight of the n-gram in topic t_k. $P(t_k|w_i)$ is computed using the Bayes's formula:

$$P(t_k|w_i) = \frac{P(w_i|t_k)P(t_k)}{\sum_{k=1}^{K} P(w_i|t_k)P(t_k)}, \tag{4.4}$$

where $P(t_k)$ is the prior topic probability and $P(w_i|t_k)$ is the word probability in topic t_k. $P(t_k)$ is computed as:

$$P(t_k) = \frac{\sum_{w_i} WP(w_i,t_k) + \beta V}{\sum_{k=1}^{K}(\sum_{w_i} WP(w_i,t_k) + \beta V)}. \tag{4.5}$$

We normalize the topic weights for each n-gram so that the total topic counts for each n-gram is summed to one and then multiply the topic weights with the original count of that n-gram in the training set. The results of the multiplication are the topic n-gram counts for the corresponding topics. This is soft clustering of the background n-grams to all topics with different topic n-gram counts. For example, a tri-gram “a b c" is seen 20 times in the training corpus and for 4 topics, the topic weights of the tri-gram “a b c" are 0.2, 0.3, 0.1 and 0.4, which are computed using equations 4.2 or 4.3. Therefore, the counts for the “a b c" in 4 topics are 4, 6, 2 and 8. We also perform a hard clustering, where each background n-gram is assigned to a single topic with the maximum fraction of the original count of the n-gram in the training set for the corresponding topic. In the above example, the n-gram “a b c" is assigned to topic number 4 with count 8. Here, the topic selection is done as:

$$t_k = \underset{t_k}{\operatorname{argmax}}\, \gamma_{t_k}. \tag{4.6}$$

The topic n-gram language models are then generated using the topic n-gram counts and defined as TNCLM.

4.3.2 Language Model Adaptation

In the LDA model, a document can be generated by a mixture of topics. So, for a test document $d_t = w_1, \ldots, w_N$, the dynamically adapted topic model by using a mixture of LMs from different topics is computed as:

$$P_{ANCLM}(w_i|h) = \sum_{k=1}^{K} \delta_{t_k} P_{t_k}(w_i|h), \tag{4.7}$$

where $P_{t_k}(w_i|h)$ is the t_k^{th} TNCLM, δ_{t_k} is the t_k^{th} topic mixture weight and $P_{ANCLM}(w_i|h)$ is the adapted n-gram count LM. The topic mixture weights are computed as equations 4.2 and 4.3. In those equations, n represents the total number of seen words in the test data.

The ANCLM is then interpolated with the background (B) n-gram model to capture the local constraints using the linear interpolation as:

$$P_L(w_i|h) = \lambda P_B(w_i|h) + (1-\lambda) P_{ANCLM}(w_i|h), \tag{4.8}$$

where λ is an interpolation weight.

4.4 Experiments

4.4.1 Data and Parameters

The '87-89 WSJ corpus is used to train the tri-gram background model and the tri-gram TNCLM using the back-off version of the Witten-Bell smoothing. To reduce the computational cost, we incorporated the cutoffs 1 and 3 on the bi-gram and tri-gram counts respectively. The Witten-Bell smoothing from the SRILM toolkit is used as the TNCLM models are generated using the floating counts. The LDA and the closed vocabulary language models are trained using the 5K non-verbalized punctuation closed vocabulary. We define the α and β for LDA analysis as $50/K$ and 0.01 respectively [37, 53]. The interpolation weight λ is computed using the *compute-best-mix* program from the SRILM toolkit. The topic mixture weights δ and the interpolation weight λ are tuned on the development test set. The results of the experiments are noted on the evaluation test set November 1993 (215 sentences, 3849 words) ARPA CSR benchmark test data for 5K vocabularies [71, 101].

4.4.2 Experimental Results

We tested our proposed approaches for topic sizes 20 and 40. The perplexity results of the ANCLM models are listed in Table 4.1 and Table 4.2, where the topic n-gram counts for the TNCLM models are generated using the confidence measures $P(w_i|t_k)$ and $P(t_k|w_i)$ respectively.

Table 4.1 Perplexity results of the ANCLM model generated using the confidence measure $P(w_i|t_k)$ for the hard and soft clustering of background n-grams

Language Model	20 Topics	40 Topics
Background (B)	83.4	83.4
ANCLM (Hard)	277.3	378.2
ANCLM (Soft)	101.2	109.2
B+ANCLM (Hard)	72.6	72.5
B+ANCLM (Soft)	71.5	70.8

For the stand-alone ANCLM models, the perplexity increases with the number of topics as the models are trained using the bigram cutoff 1 and trigram cutoff 3. However, the

Table 4.2 Perplexity results of the ANCLM model generated using the confidence measure $P(t_k|w_i)$ for the hard and soft clustering of background n-grams

Language Model	20 Topics	40 Topics
Background (B)	83.4	83.4
ANCLM (Hard)	227.9	287.25
ANCLM (Soft)	92.9	101.45
B+ANCLM (Hard)	71.65	71.5
B+ANCLM (Soft)	70.15	69.9

interpolation of ANCLM with the background LM yields improved perplexity results with increasing topics. The perplexity using the background trigram model is 83.4. Therefore the interpolation of background LM and the ANCLM with both the confidence measures outperforms the background LM. From Tables 4.1 and 4.2, we can also note that the ANCLM with the confidence measure $P(t_k|w_i)$ outperforms the ANCLM with the confidence measure $P(w_i|t_k)$.

We evaluated the proposed approaches for speech recognition. We used the unigram scaling approach of the LDA-adapted model (LDA unigram scaling) [96] and the interpolation of the background model with the LDA-adapted n-gram model obtained using the n-gram weighting approach (LDA n-gram weighting) [40] for comparison. We evaluated the WER experiments using lattice rescoring. In the first pass, we used the background n-gram language model for lattice generation. In the second pass, we applied the LM adaptation approaches for lattice rescoring. The experimental results are plotted in Figures 4.2 and 4.3 for the confidence measures $P(w_i|t_k)$ and $P(t_k|w_i)$ respectively. From Figures 4.2 and 4.3, we can note that the LMs generated using the confidence measure $P(t_k|w_i)$ give better results than the LMs generated by the confidence measure $P(w_i|t_k)$. This is obvious as we formed the topic weights of the n-grams in LM generation. Using the $P(t_k|w_i)$ confidence measure, we also found the higher number of bigrams and trigrams over the $P(w_i|t_k)$ confidence measure after considering cutoffs 1 and 3 for the bigrams and trigrams respectively. Using confidence measure $P(w_i|t_k)$ in Figure 4.2, we can see that the hard clustering ANCLM outperforms the LDA unigram scaling, the soft clustering ANCLM outperforms both the LDA unigram scaling and the LDA n-gram weighting for the topic size 20. For topic size 40, only the soft clustering ANCLM outperforms the LDA unigram scaling. In contrast, the hard and soft clustering ANCLM, generated by using the $P(t_k|w_i)$ confidence measure, outperform the LDA unigram scaling and the LDA n-gram weighting approaches respec-

tively for both 20 and 40 topics. We obtained improved WER results for topic size 40. We achieved the WER of 8.1% using the background n-gram LM in the first pass. The LDA unigram scaling, LDA n-gram weighting, ANCLM hard clustering and ANCLM soft clustering approaches with 40 topics reduce WER to 7.9%, 7.5%, 7.6% and 7.3% respectively. These results indicate that the proposed ANCLM with the confidence measure $P(t_k|w_i)$ performs better than the LDA unigram scaling [96] and LDA n-gram weighting [40] approaches. The soft clustering ANCLM generated using $P(t_k|w_i)$ gives significant WER reductions of about 9.9% (8.1% to 7.3%), 7.6% (7.9% to 7.3%), and 2.7% (7.5% to 7.3%) over the background n-gram LM, LDA unigram scaling [96], and LDA n-gram weighting [40] approaches respectively.

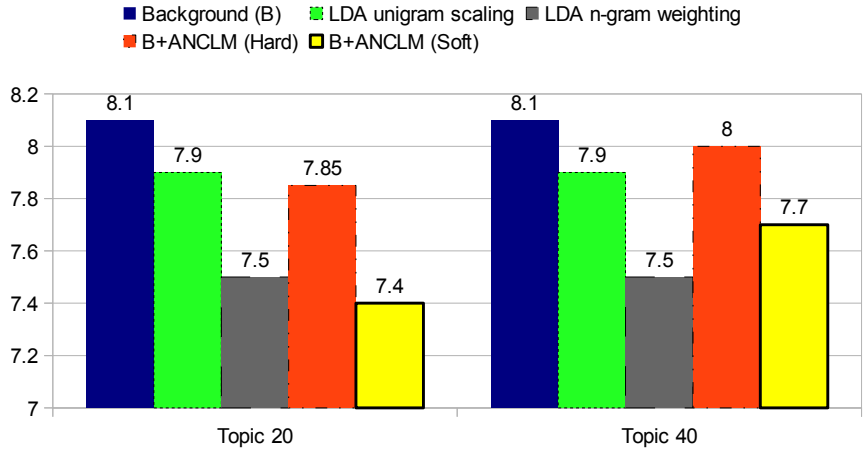

Fig. 4.2 WER results (%) for the ANCLM model developed by using confidence measure $P(w_i|t_k)$

4.5 Summary

In this chapter, we proposed novel LM adaptation approaches where the topic mixture weights of the background n-grams were used to train the topic models. In soft clustering of n-grams, each n-gram of the training set is assigned to all topics using the fraction of the global count of the n-gram for the respective topics. The fraction is determined by the multiplication of the global count with the normalized topic weights for the n-gram such that the total count of that n-gram for all topics is equal to the global count in the training set. In hard-clustering, each n-gram is assigned to a single topic with the maximum fraction of the global n-gram count and the topic is selected with the maximum topic weight.

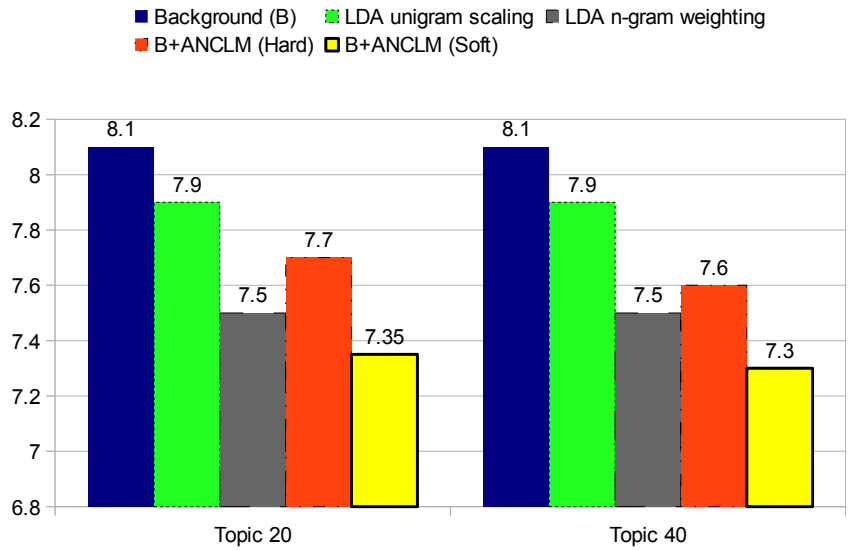

Fig. 4.3 WER results (%) for the ANCLM model developed by using confidence measure $P(t_k|w_i)$

The topic weights of the n-grams are computed by using the average of the probability of word given topic and the probability of topic given word confidence measures for the words in the n-grams under different latent topics of the LDA model. We also introduced a new weighting approach for the LDA topic model adaptation where the topic mixture weights were computed using the average of the confidence measures for the seen words in the development test set. We compared our approaches with recent LDA topic LM adaptation approaches and have seen that our approaches yield better performance.

Chapter 5

Novel Topic *n*-gram Count LM

In this chapter, we introduce a novel topic *n*-gram count language model (NTNCLM) using topic probabilities of training documents and document-based *n*-gram counts. The topic probabilities for the documents are computed by averaging the topic probabilities of words seen in the documents. The topic probabilities of documents are multiplied by the document-based *n*-gram counts. The products are then summed-up for all the training documents. The results are used as the counts of the respective topics to create the NTNCLMs. The NTNCLMs are adapted by using the topic probabilities of a development test set that are computed as above. We compare our approach with a recently proposed TNCLM [42] described in chapter 4, where the long-range information outside of the *n*-gram events is not encountered. Our approach yields significant perplexity and word error rate (WER) reductions over the other approach using the Wall Street Journal (WSJ) corpus [48].

5.1 Introduction

We extend our previous work (Chapter 4) [42] to incorporate the long-range useful information outside of *n*-gram events. In [42], the features of the LDA model were used to create the topic *n*-gram count language model (TNCLM) [42]. Because of bag-of-words characteristics of the LDA model, each word has equal weight in determining the topic mixtures. Also, latent topics are independent of each other in the LDA topic set. A constraint was taken such that the total count of an *n*-gram for all topics is equal to the count of that *n*-gram in the training set. TNCLMs were formed by computing the topic probabilities of background *n*-grams $P(t_k|w_1,\ldots,w_n),(k=1,\ldots,K)$ by averaging the topic probabilities of the words $P(t_k|w_i)$ present in the *n*-grams. $P(t_k|w_1,\ldots,w_n)$ were multiplied with the global count of the *n*-gram $C(h,w_i)$ and then used as the counts of the topics to create the

TNCLMs. The probability of topic t_k given word w_i, $P(t_k|w_i)$ and the probability of word w_i given topic t_k, $P(w_i|t_k)$ were used as confidence measures in determining the topic probability of the n-grams $P(t_k|w_1,\ldots,w_n)$, where $P(t_k|w_i)$ outperforms $P(w_i|t_k)$ [42]. In this chapter, we used only the confidence measure $P(t_k|w_i)$. For details of the TNCLM generation, please see chapter 4. However, the TNCLMs do not capture the long-range important information outside of the n-gram events. Here, we propose a novel TNCLM (NTNCLM) where $P(t_k|w_1,\ldots,w_n)$ are derived by using the topic probabilities of the training documents $P(t_k|d_l),(l=1,\ldots,M)$. $P(t_k|d_l)$ are calculated by averaging the $P(t_k|w_i)$ for words seen in the documents. $P(t_k|d_l)$ are multiplied with the document-based n-gram counts $C(h,w_i,d_l)$ and then summed-up for all training documents. The results are used as the counts of topics to create the NTNCLMs. The TNCLMs and NTNCLMs are both adapted by using the topic mixture weights obtained by averaging the $P(t_k|w_i)$ over the seen words of a development test set d_t. The adapted models are interpolated with a background tri-gram model to capture the local lexical regularities. The complete idea is described in Figure 5.1. In the figure, $\gamma_{t_k,n}$ and γ_{t_k,d_l} represent $P(t_k|w_1,\ldots,w_n)$ and $P(t_k|d_l)$ respectively. N_{d_l} and N_{d_t} describe the the number of words seen in the training document d_l and the development test set d_t. We compare our approach with an adapted n-gram LM obtained by unsupervised language model adaptation using latent semantic marginals [96] and the interpolation of the adapted TNCLM with the background model [42]. We apply the LM adaptation approaches after the first pass decoding and have seen that our approach outperforms the conventional approaches.

5.2 LDA Training

The parameters of the LDA model are computed by using the MATLAB topic modeling toolbox [37, 93]. Here, we obtain a word-topic matrix WP and a document topic matrix DP. An entry $WP(w_i,t_k)$ describes the number of times the word w_i has been assigned to topic t_k over the training set. An entry $DP(d_l,t_k)$ of the DP matrix contains the total occurrences of words in document d_l that are from a topic t_k. We used the above matrices to compute the probability of words given topics and the probability of topics given documents as [37, 53]:

$$P(w_i|t_k,\beta)=\frac{WP(w_i,t_k)+\beta}{WP(.,t_k)+V\beta}, \tag{5.1}$$

$$P(t_k|d_l,\theta_{d_l})=\frac{DP(d_l,t_k)+\alpha}{DP(d_l,.)+K\alpha}, \tag{5.2}$$

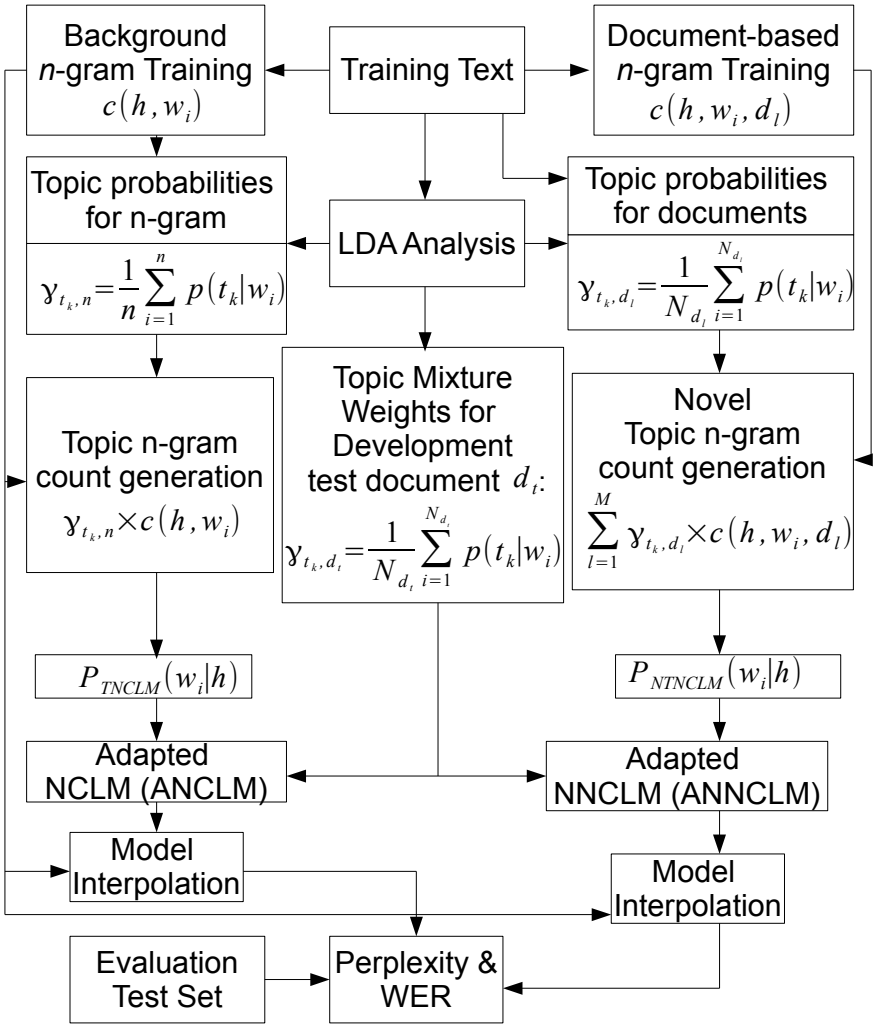

Fig. 5.1 Adaptation of TNCLM and NTNCLM

where $WP(.,t_k)$ is the total count of words in topic t_k, $DP(d_l,.)$ contains the occurrences of words from all topics in document d_l, V is the total number of words, and β is defined as in section 2.5.4.

5.3 Proposed NTNCLM

The TNCLM model does not capture the information outside the n-gram events as it directly uses topic probabilities of words $P(t_k|w_i)$ in generating topic probability of n-grams $P(t_k|w_1,\ldots,w_n)$. To compensate for the weakness of this model, we introduce a novel TNCLM (NTNCLM) that uses topic probabilities of training documents $P(t_k|d_l)$ in computing topic probabilities of n-grams $P(t_k|w_1,\ldots,w_n)$.

The topic probabilities of the training documents d_l ($l = 1,\ldots,M$) are created by averaging the topic probabilities of words present in the respective documents as:

$$P(t_k|d_l) = \frac{1}{N_{d_l}} \sum_{i=1}^{N_{d_l}} P(t_k|w_i), \tag{5.3}$$

where N_{d_l} is the number of words seen in training document d_l. The topic probabilities for each document d_l are then normalized so that the total topic probabilities for each document are summed to one.

The topic probability for an *n*-gram is created as:

$$\begin{aligned} P(t_k|w_1,\ldots,w_n) &= \sum_{l=1}^{M} P(t_k|d_l)P(d_l|w_1,\ldots,w_n) \\ &= \sum_{l=1}^{M} P(t_k|d_l)\frac{C(w_1,\ldots,w_n,d_l)}{C(w_1,\ldots,w_n)}, \end{aligned} \tag{5.4}$$

The topic probability of the *n*-gram is then multiplied with the global *n*-gram count $C(w_1,\ldots,w_n)$ and the product is used as the count of the *n*-gram for the respective topic. The results can be written as:

$$\begin{aligned} C(w_1,\ldots,w_n,t_k) &= P(t_k|w_1,\ldots,w_n) * C(w_1,\ldots,w_n) \\ &= \sum_{l=1}^{M} P(t_k|d_l)C(w_1,\ldots,w_l,d_l), \end{aligned} \tag{5.5}$$

where $P(t_k|d_l)$ are the topic probabilities for training documents created by Equation 5.3. The NTNCLMs are then created by using the respective topic *n*-gram counts. We also introduce other TNCLMs defined as LDA TNCLMs (LTNCLMs) by using Equation 5.5 where the $P(t_k|d_l)$ is computed by using the document-topic matrix *DP* (Equation 5.2).

5.4 LM Adaptation Approach

In the LDA model, a document can be generated by a mixture of topics. So, for a test document $d_t = w_1,\ldots,w_{N_{d_t}}$, the dynamically adapted topic model by using a mixture of LMs from different topics is computed as:

$$P_{ANCLM/ANNCLM/ALNCLM}(w_i|h) = \sum_{k=1}^{K} \delta_{t_k} P_{t_k}(w_i|h), \tag{5.6}$$

where $P_{t_k}(w_i|h)$ is the t_k^{th} TNCLM/NTNCLM/LTNCLM, $P_{ANCLM/ANNCLM/ALNCLM}(w_i|h)$ are the adapted n-gram count LMs and δ_{t_k} is the t_k^{th} topic mixture weight. The mixture weights for the TNCLMs and NTNCLMs are computed as:

$$P(t_k|d_t) = \frac{1}{N_{d_t}} \sum_{i=1}^{N_{d_t}} P(t_k|w_i), \tag{5.7}$$

where N_{d_t} is the number of words seen in the development test document d_t. For the LTNCLMs, the mixture weights are computed using LDA inference [53].

The ANCLM/ANNCLM/ALNCLMs are then interpolated with the background (B) n-gram model to capture the local constraints using linear interpolation as:

$$P_L(w_i|h) = \lambda P_B(w_i|h) + (1-\lambda) P_{ANCLM/ANNCLM/ALNCLM}(w_i|h), \tag{5.8}$$

where λ is an interpolation weight.

5.5 Experiments

5.5.1 Data and Parameters

The '87-89 WSJ corpus is used to train the tri-gram background (B) model and the tri-gram TNCLMs/NTNCLMs/LTNCLMs using the back-off version of the Witten-Bell smoothing. To reduce the computational cost, we incorporated the cutoffs 1 and 3 on the background bi-gram and background tri-gram counts respectively. The Witten-Bell smoothing from the SRILM toolkit is used as the TNCLMs/NTNCLMs/LTNCLMs are generated using the floating counts. The LDA and the closed vocabulary language models are trained using the 5K non-verbalized punctuation closed vocabulary. We define the α and β for LDA analysis as $50/K$ and 0.01 respectively [37, 53]. The development and the evaluation test sets are the **si_dt_05.odd** (248 sentences from 10 speakers) and the Nov'93 Hub 2 5K test data from the ARPA November 1993 WSJ evaluation (215 sentences from 10 speakers) [71, 101]. The topic mixture weights δ and the interpolation weight λ are tuned on the development test set. The results are noted on the evaluation test set.

5.5.2 Experimental Results

We tested our proposed approaches for various topic sizes. The perplexity results of the models are explained in Table 5.1. From Table 5.1, we can note that the proposed approaches

Table 5.1 Perplexity results of the language models

Language Model	25 Topics	50 Topics
Background (B)	83.4	83.4
ANCLM	105.5	134.0
ALNCLM	86.5	111.4
ANNCLM	86.2	110.4
B+ANCLM	75.3	75.6
B+ALNCLM	74.6	74.8
B+ANNCLM	74.7	74.9

outperform the ANCLM [42] in both stand-alone and interpolated form for all topic sizes.

We also evaluated the LM adaptation approaches for speech recognition. We used the unigram scaling approach of the LDA adapted model (LDA unigram scaling) [96] and the interpolation of background models with the ANCLM model [42] for comparison. We evaluated the WER experiments using lattice rescoring. In the first pass, we used the background tri-gram language model for lattice generation. In the second pass, interpolation of the background and the adapted models are applied for lattice rescoring. The experimental results are plotted in Figure 5.2. From Figure 5.2, we can note that the proposed B+ANNCLM gives significant WER reductions of about 9.9% (8.1% to 7.3%), 7.6% (7.9% to 7.3%), 3.9% (7.6% to 7.3%), and 1.4% (7.4% to 7.3%) for 25 topics, and about 7.4% (8.1% to 7.5%), 5.1% (7.9% to 7.5%), 3.8% (7.8% to 7.5%), and 1.3% (7.6% to 7.5%) for 50 topics over the background trigram, LDA unigram scaling [96], B+ANCLM [42] and B+ALNCLM (also proposed by us) approaches respectively. However, the B+ALNCLM model outperforms the background, LDA unigram scaling [96] and B+ANCLM [42] approaches respectively. The significance improvement in WER using the proposed B+ANNCLM is done by using a match-pair-test where the misrecognized words in each test utterance are counted. We obtain the p-values of 0.03 and 0.02 relative to B+ANCLM [42] for the topic sizes 25 and 50 respectively. At a significance level of 0.05, our proposed B+ANNCLM model outperforms the B+ANCLM model [42].

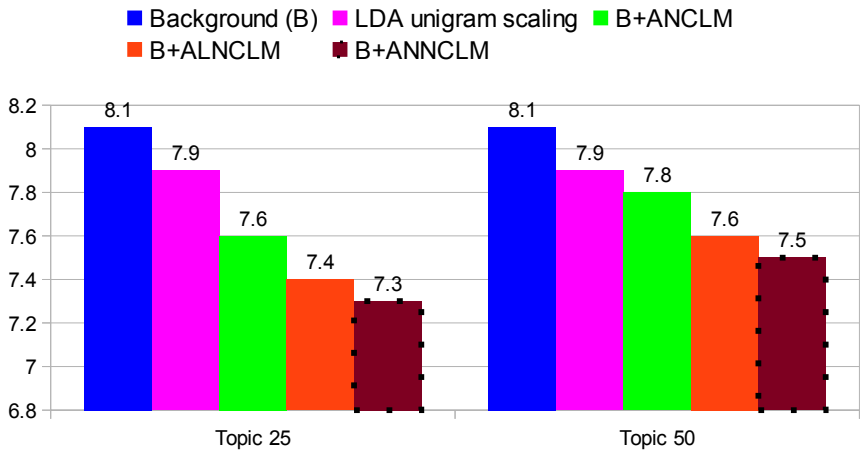

Fig. 5.2 WER results (%) of the language models

5.6 Summary

In this chapter, we proposed a novel TNCLM (NTNCLM) using document-based topic distributions and n-gram counts. The topic probabilities for training documents are created by averaging the confidence measure (topic probability given words) of the words present in the documents. Then, they are multiplied by the document-based n-gram counts and the products are summed up for all the training documents. The results are used as the n-gram counts for the respective topics to create the NTNCLM. We also introduce an LDA TNCLM (LTNCLM) as above where the topic probabilities for documents are created by using the document-topic matrix obtained from the LDA model training. We compare our approaches with a recently proposed TNCLM [42], which uses the above confidence measures to compute the probability of background n-grams and is used as the count of the n-grams for the respective topics. The normalized topic probabilities of the n-gram are multiplied by the global n-gram count to form the topic n-gram count for the respective topics. However, TNCLM does not capture the long-range information outside of the n-gram events. To compensate for the weaknesses of the TNCLMs, the NTNCLMs and LTNCLMs are proposed here. Both TNCLMs, NTNCLMs and LTNCLMs are adapted and then interpolated with a background trigram model to capture the short-range information. The proposed approaches yield better performance over the conventional approaches.

Chapter 6

Context-based PLSA and Document-based CPLSA

In this chapter, we propose a novel context-based probabilistic latent semantic analysis (CPLSA) language model [43] for speech recognition. In this model, the topic is conditioned on the immediate history context and the document in the original PLSA model. This allows computing all the possible bigram probabilities of the seen history context using the model. It properly computes the topic probability of an unseen document for each history context present in the document. We compare our approach with a recently proposed unsmoothed bigram PLSA (UBPLSA) model [7] where only the seen bigram probabilities are calculated, which causes computing the incorrect topic probability for the present history context of the unseen document. The proposed CPLSA model requires a significantly less amount of computation time and memory space requirements than the unsmoothed bigram PLSA model. In the CPLSA model, the word probabilities for topics are computed by the sum of bigram events in all documents. However, in different documents words can appear to describe different topics. To solve this problem, we also introduce a document-based CPLSA model (DCPLSA) [50]. This model is similar to the CPLSA model except that the probability of a word is conditioned on both topic and document. However, it requires larger memory and computation time than the CPLSA model.

6.1 Introduction

In the UBPLSA model [7], the bigram probabilities for each topic are modeled and the topic is conditioned on the bigram history and the document. For each topic, it requires V distributions, where V is the size of vocabulary. So, it needs high computation time and

huge memory space. However, this approach is not practical as it assigns zero probability to the unseen bigrams. Furthermore, in testing, the model computes the topic probabilities for the bigram histories that are present in the test document. However, it cannot compute the topic probabilities for some bigram history contexts that are present in both the training and test set as the bigram probabilities for the corresponding bigram histories are zero because the model assigns zero probability to the unseen bigrams. Therefore, the model cannot compute some bigram probabilities of the test document that should be computed by the training model. However, those bigram probabilities of the test document are computed later by the smoothing process.

In this chapter, we propose a context based PLSA (CPLSA) model where the topic is further conditioned on the history context in the original PLSA model. It allows computing all the possible bigram probabilities for the seen history context in the training set. Therefore, the topic probabilities for the history contexts of the test document can be computed properly. We have seen that the proposed approach gives significantly better results over the UBPLSA model [7]. In addition, it reduces the complexity and memory requirements as it uses unigram probabilities for topics. Moreover, we propose a new document-specific context PLSA (DCPLSA) model. The CPLSA model [43] uses the sum of bigrams in all documents to compute the word probabilities for topics. However, words in the bigrams may describe different topics in different documents. For example, the bigram *White House* can occur in a document where it describes a real estate topic. Also, it can occur in another document that describes a political topic. Therefore, the probability of word given only the topics may not give the appropriate results. This motivates us to introduce a new DCPLSA model where the word probabilities are trained by conditioning on the topics and the documents. However, the DCPLSA model requires more complexity and memory requirement than the CPLSA model.

6.2 Review of PLSA and UBPLSA Models

6.2.1 PLSA Model

The PLSA model [33] extracts semantic information from a corpus in a probabilistic framework. It uses an unobserved topic variable with each observation, i.e., with each occurrence of a word in a document. It is assumed that the document and the word are independent conditioned on the state of the latent topic variable. It models each word in a document as a sample from a mixture model, where the mixture models can be viewed as representations

of topic distributions. Therefore, a document is generated as a mixture of topic distributions and reduced to a fixed set of topics. Each topic is a distribution over words. The model [33] can be described in the following procedure. First a document d_l $(l = 1,2,\ldots,M)$ is selected with probability $P(d_l)$. A topic t_k $(k = 1,2,\ldots,K)$ is then chosen with probability $P(t_k|d_l)$, and finally a word w_i $(i = 1,2,\ldots,N)$ is generated with probability $P(w_i|t_k)$. The probability of word w_i given a document d_l can be estimated as:

$$P(w_i|d_l) = \sum_{k=1}^{K} P(w_i|t_k)P(t_k|d_l). \tag{6.1}$$

The model parameters $P(w_i|t_k)$ and $P(t_k|d_l)$ are computed by using the expectation maximization (EM) algorithm [33].

6.2.2 UBPLSA Model

The PLSA model yields unigram models for topics. To improve the performance, a bigram PLSA model [82] was introduced where the bigram probabilities for topics were trained instead of unigrams in the PLSA model. Before describing the UBPLSA model, the previous bigram PLSA model is briefly explained. Instead of $P(w_i|t_k)$ in Equation 6.1, the bigram PLSA model uses $P(w_i|w_j,t_k)$ in computing the probability of word w_i given the bigram history w_j and the document d_l:

$$P(w_i|w_j,d_l) = \sum_{k=1}^{K} P(w_i|w_j,t_k)P(t_k|d_l). \tag{6.2}$$

The model parameters are computed using the EM procedure [82].

The UBPLSA model was recently proposed in [7], which outperforms the previous bigram PLSA model [82]. Here, the topic probability is further conditioned on the bigram history context. It can model the topic probability for the document given a context, using the word co-occurrences in the document. In this model, the probability of the word w_i given the document d_l and the word history w_j is computed as:

$$P(w_i|w_j,d_l) = \sum_{k=1}^{K} P(w_i|w_j,t_k)P(t_k|w_j,d_l). \tag{6.3}$$

The EM procedure for training the model takes the following two steps: E-step:

$$P(t_k|w_j,w_i,d_l) = \frac{P(w_i|w_j,t_k)P(t_k|w_j,d_l)}{\sum_{k'} P(w_i|w_j,t_{k'})P(t_{k'}|w_j,d_l)}, \tag{6.4}$$

M-step:

$$P(w_i|w_j,t_k) = \frac{\sum_{l'} n(w_j,w_i,d_{l'})P(t_k|w_j,w_i,d_{l'})}{\sum_{i'}\sum_{l'} n(w_j,w_{i'},d_{l'})P(t_k|w_j,w_{i'},d_{l'})}, \tag{6.5}$$

$$P(t_k|w_j,d_l) = \frac{\sum_{i'} n(w_j,w_{i'},d_l)P(t_k|w_j,w_{i'},d_l)}{\sum_{k'}\sum_{i'} n(w_j,w_{i'},d_l)P(t_{k'}|w_j,w_{i'},d_l)}. \tag{6.6}$$

where $n(w_j,w_i,d_l)$ is the number of times the word pair w_jw_i occurs in the training document d_l.

In the UBPLSA model [7], the bigram probabilities for each topic are modeled and the topic is conditioned on the bigram history and the document. For each topic, it requires V distributions, where V is the size of vocabulary. So, it needs high computation time and huge memory space. However, this approach is not practical as it assigns zero probability to the unseen bigrams. Furthermore, in testing, the model computes the topic probabilities for the bigram histories that are present in the test document. However, it cannot compute the topic probabilities for some bigram history contexts that are present in both the training and test sets as the bigram probabilities for the corresponding bigram histories are zero because the model assigns zero probability to the unseen bigrams. Therefore, the model cannot compute some bigram probabilities of the test document that should be computed by the training model. However, those bigram probabilities of the test document are computed later by the smoothing process.

6.3 Proposed CPLSA Model

A problem of the UBPLSA model is that it uses only seen bigrams for training. Therefore, it cannot compute all the possible bigram probabilities in the training phase. It results in incorrect topic probabilities of the test document. This is because the model cannot compute topic probabilities for some history contexts that are present both in the training and test sets. To overcome the limitations of the UBPLSA model, the CPLSA model was introduced [43].

The CPLSA model is similar to the original PLSA model except the topic is further conditioned on the history context as is the UBPLSA model. To better understand the model, the matrix decomposition of the CPLSA model is described in Figure 6.1. Using this model,

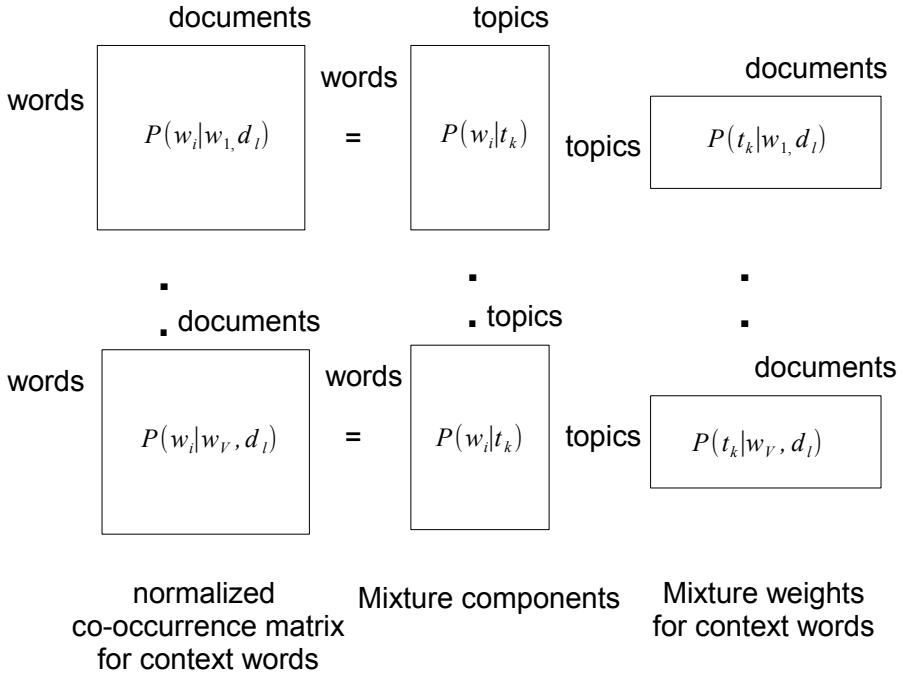

Fig. 6.1 Matrix decomposition of the CPLSA model

we can compute the bigram probability using the unigram probabilities of topics as:

$$P(w_i|w_j,d_l) = \sum_{k=1}^{K} P(w_i|t_k)P(t_k|w_j,d_l). \tag{6.7}$$

The parameters of the model are computed as: E-step:

$$P(t_k|w_j,w_i,d_l) = \frac{P(w_i|t_k)P(t_k|w_j,d_l)}{\sum_{k'} P(w_i|t_{k'})P(t_{k'}|w_j,d_l)}, \tag{6.8}$$

M-step:

$$P(w_i|t_k) = \frac{\sum_{j'}\sum_{l'} n(w_{j'},w_i,d_{l'})P(t_k|w_{j'},w_i,d_{l'})}{\sum_{i'}\sum_{j'}\sum_{l'} n(w_{j'},w_{i'},d_{l'})P(t_k|w_{j'},w_{i'},d_{l'})}, \tag{6.9}$$

$$P(t_k|w_j,d_l) = \frac{\sum_{i'} n(w_j,w_{i'},d_l)P(t_k|w_j,w_{i'},d_l)}{\sum_{k'}\sum_{i'} n(w_j,w_{i'},d_l)P(t_{k'}|w_j,w_{i'},d_l)}. \tag{6.10}$$

From Equations 6.8 and 6.10, we see that the model can compute all the possible bigram probabilities of the seen history context in the training set. Therefore, the model can overcome the problem of computing topic probabilities of the test document using the UBPLSA model, which causes the problem in the computation of the bigram probabilities of the test

document.

6.4 Proposed DCPLSA Model

In the CPLSA model, the word probabilities for topics are computed using the sum of the bigram events in all training documents where the words may appear to describe different topics in different documents. In this section, we describe a new topic model where the document-based word probabilities for topics are trained. The DCPLSA model is similar to the original CPLSA model except that the document-based word probabilities for topics are computed instead of the global word probabilities for topics in the CPLSA model. To better understand the model, the matrix decomposition of the DCPLSA model is described in Figure 6.2. Using this model, we can compute the n-gram probability for a document as:

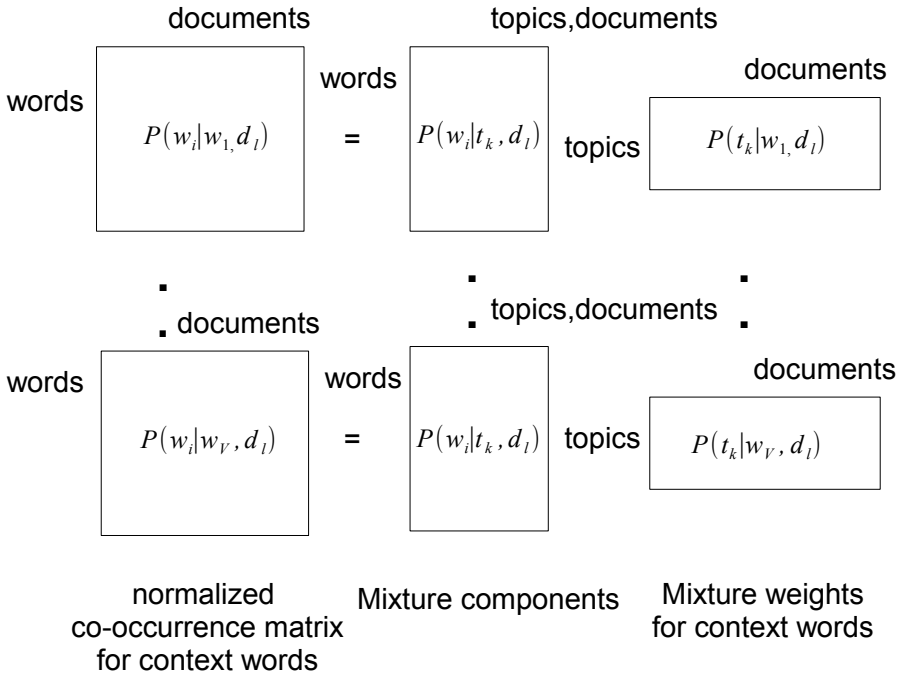

Fig. 6.2 Matrix decomposition of the DCPLSA model

$$P(w_i|w_j,d_l) = \sum_{k=1}^{K} P(w_i|t_k,d_l)p(t_k|w_j,d_l). \tag{6.11}$$

The parameters of the model are computed as: E-step:

$$P(t_k|w_j,w_i,d_l) = \frac{P(w_i|t_k,d_l)P(t_k|w_j,d_l)}{\sum_{k'} P(w_i|t_{k'},d_l)P(t_{k'}|w_j,d_l)}, \tag{6.12}$$

M-step:

$$P(w_i|t_k,d_l) = \frac{\sum_{j'} n(w_{j'},w_i,d_l)P(t_k|w_j,w_i,d_l)}{\sum_{i'}\sum_{j'} n(w_{j'},w_{i'},d_l)P(t_k|w_{j'},w_{i'},d_l)}, \tag{6.13}$$

$$P(t_k|w_j,d_l) = \frac{\sum_{i'} n(w_j,w_{i'},d_l)P(t_k|w_j,w_{i'},d_l)}{\sum_{k'}\sum_{i'} n(w_j,w_{i'},d_l)P(t_{k'}|w_j,w_{i'},d_l)}. \tag{6.14}$$

6.5 Parameter Estimation of the DCPLSA Model Using the EM Algorithm

E-step:

In the E-step, we use Bayes formula to calculate the posterior probability of the latent variable t_k given the observed data d_l, w_j, and w_i as [7]:

$$\begin{aligned} P(t_k|w_j,w_i,d_l) &= \frac{P(t_k,w_j,w_i,d_l)}{\sum_{k=1}^{K} P(t_k,w_j,w_i,d_l)} \\ &= \frac{P(w_j,t_k)P(d_l|t_k,w_j)P(w_i|t_k,d_l)}{\sum_{k=1}^{K} P(w_j,t_k)P(d_l|t_k,w_j)P(w_i|t_k,d_l)} \\ &= \frac{P(t_k|w_j)P(d_l|t_k,w_j)P(w_i|t_k,d_l)}{\sum_{k=1}^{K} P(t_k|w_j)P(d_l|t_k,w_j)P(w_i|t_k,d_l)} \\ &= \frac{P(w_i|t_k,d_l)P(t_k|w_j,d_l)}{\sum_{k=1}^{K} P(w_i|t_k,d_l)P(t_k|w_j,d_l)} \end{aligned} \tag{6.15}$$

In the M-step, the parameters are updated by maximizing the log-likelihood as:

$$\ell = \prod_{j,i,l} P(d_l,w_j,w_i)^{n(d_l,w_j,w_i)}, \tag{6.16}$$

where $n(d_l,w_j,w_i)$ is the frequency of word pair w_jw_i in document d_l and $P(d_l,w_j,w_i)$ is the occurrence probability of the word pair w_jw_i in document d_l.

To estimate the parameters $(\theta = P(w_i|t_k,d_l),P(t_k|w_j,d_l))$, MLE is used to maximize the

log-likelihood as [7]:

$$\begin{aligned}\theta_{ML} &= \underset{\theta}{\operatorname{argmax}} \log(\ell) \\ &= \underset{\theta}{\operatorname{argmax}} \sum_{j,i,l} n(d_l, w_j, w_i) \log(P(d_l, w_j, w_i)) \\ &= \underset{\theta}{\operatorname{argmax}} \sum_{j,i,l} n(d_l, w_j, w_i) \times \big[\log P(d_l, w_j) + \log P(w_i|w_j, d_l)\big].\end{aligned} \tag{6.17}$$

Using Equation 6.11, the above equation is written as:

$$\begin{aligned}\theta_{ML} &= \underset{\theta}{\operatorname{argmax}} \sum_{j,i,l} n(d_l, w_j, w_i) \log P(d_l, w_j) \\ &+ \underset{\theta}{\operatorname{argmax}} \sum_{j,i,l} n(d_l, w_j, w_i) \log \big(\sum_k P(w_i|t_k, d_l) P(t_k|w_j, d_l)\big) \\ &= \underset{\theta}{\operatorname{argmax}} \sum_{j,i,l} n(d_l, w_j, w_i) \log \big(\sum_k P(w_i|t_k, d_l) P(t_k|w_j, d_l)\big)\end{aligned} \tag{6.18}$$

In Equation 6.18, the factor independent of the parameters θ is omitted. Equation 6.18 needs to be differentiated to maximize the log-likelihood. The differentiation of Equation 6.18 with respect to the parameters does not lead to well-formed formulae, so we try to find a lower bound for Equation 6.18 using Jensen's inequality [7]:

$$\begin{aligned}H &= \sum_{j,i,l} n(d_l, w_j, w_i) \log \big(\sum_k P(w_i|t_k, d_l) P(t_k|w_j, d_l)\big) \\ &= \sum_{j,i,l} n(d_l, w_j, w_i) \\ &\times \log \big(\sum_k P(t_k|w_j, w_i, d_l) \frac{P(w_i|t_k, d_l) P(t_k|w_j, d_l)}{P(t_k|w_j, w_i, d_l)}\big) \\ &\geq \sum_{j,i,l} n(d_l, w_j, w_i) \sum_k P(t_k|w_j, w_i, d_l) \\ &\times \log \big(\frac{P(w_i|t_k, d_l) P(t_k|w_j, d_l)}{P(t_k|w_j, w_i, d_l)}\big)\end{aligned} \tag{6.19}$$

The right-hand side of the Equation 6.19 should be maximized. Estimating the parameters by maximizing the lower-bound is a constrained optimization problem as all parameters indicate probability distributions. Therefore, the parameters should satisfy the constraints [7]:

$$\begin{aligned} \sum_i P(w_i|t_k,d_l) &= 1 \;\forall k \\ \sum_k P(t_k|w_j,d_l) &= 1 \;\forall j,l \end{aligned} \tag{6.20}$$

To consider the above constraints, the right-hand side of Equation 6.19 has to be augmented by appropriate Lagrange multipliers as [7]:

$$\begin{aligned} H = & \sum_{j,i,l} n(d_l,w_j,w_i) \sum_k P(t_k|w_j,w_i,d_l) \\ & \times \log\big(\frac{P(w_i|t_k,d_l)P(t_k|w_j,d_l)}{P(t_k|w_j,w_i,d_l)}\big) \\ & + \sum_k \tau_k \big(1 - \sum_i P(w_i|t_k,d_l)\big) \\ & + \sum_{j,l} \rho_{j,l} \big(1 - \sum_k P(t_k|w_j,d_l)\big), \end{aligned} \tag{6.21}$$

where τ_k and $\rho_{j,l}$ are the Lagrange multipliers related to the constraints described in Equation 6.20 [7]. Now, differentiating Equation 6.21 partially with respect to the parameters yields [7]:

$$\begin{aligned} \frac{\delta H}{\delta P(w_i|t_k,d_l)} &= \sum_j n(d_l,w_j,w_i) \frac{\sum_k P(t_k|w_j,w_i,d_l)}{P(w_i|t_k,d_l)} - \tau_k \\ &= 0, \\ \frac{\delta H}{\delta P(t_k|w_j,d_l)} &= \sum_i n(d_l,w_j,w_i) \frac{\sum_k P(t_k|w_j,w_i,d_l)}{P(t_k|w_j,d_l)} - \rho_{j,l} \\ &= 0. \end{aligned} \tag{6.22}$$

By solving Equation 6.22 and applying the constraints in Equation 6.20, the M-step re-estimating formulae, described in Equations 6.13 and 6.14, are obtained as [7]:

$$P(w_i|t_k,d_l) = \frac{\sum_{j'} n(w_{j'},w_i,d_l) P(t_k|w_{j'},w_i,d_l)}{\sum_{i'}\sum_{j'} n(w_{j'},w_{i'},d_l) P(t_k|w_{j'},w_{i'},d_l)}, \tag{6.23}$$

$$P(t_k|w_j,d_l) = \frac{\sum_{i'} n(w_j,w_{i'},d_l) P(t_k|w_j,w_{i'},d_l)}{\sum_{k'}\sum_{i'} n(w_j,w_{i'},d_l) P(t_{k'}|w_j,w_{i'},d_l)}. \tag{6.24}$$

The E-step and M-step are then repeated until convergence is achieved.

6.6 N-gram Probabilities of the Test Document

We used the folding-in procedure [33] to compute the n-gram probabilities of the test document d_t using the above models. For the PLSA model, we keep the unigram probabilities for topics $P(w_i|t_k)$ fixed and used them to compute the topic probabilities $P(t_k|d_t)$ using EM iterations and then compute the unigram probabilities $P(w_i|d_t)$ using Equation 6.1. In the UB-PLSA model, the bigram probabilities $P(w_i|w_j,t_k)$ remain unchanged while computing the topic probabilities $P(t_k|w_j,d_t)$ using EM iterations. The bigram probabilities $P(w_i|w_j,d_t)$ are then computed using Equation 6.3. However, the topic probabilities $P(t_k|w_j,d_t)$ for some histories w_j were assigned zeros (Equations 6.4 and 6.6), as the training model gives zero probabilities to the unseen bigrams in the training model. Therefore, some bigrams of the test document with history context w_j were assigned zero probabilities. The problem is solved by the CPLSA model, which is able to assign probabilities to all the bigrams of the seen history context in the training set. In the CPLSA model, $P(w_i|t_k)$ remains fixed in the EM iterations of the test phase in computing $P(t_k|w_j,d_t)$. Finally, the bigram probabilities $P(w_i|w_j,d_t)$ are computed using Equation 6.7.

For the DCPLSA, we have word probabilities $P(w_i|t_k,d_l)$ for topics of each training document d_l. During testing, we keep $P(w_i|t_k,d_l)$ unchanged and used them to compute $P(t_k|w_j,d_t,d_l)$ for the test document d_t.

The seen bi-gram probabilities of the test document d_t are then computed as:

$$\begin{aligned} P(w_i|w_j,d_t) &= \sum_{l=1}^{M} P(w_i|w_j,d_t,d_l)P(d_l|w_j) \\ &= \sum_{l=1}^{M}(\sum_{k=1}^{K} P(w_i|t_k,d_l)P(t_k|w_j,d_t,d_l))\frac{C(w_j,d_l)}{\sum_{l=1}^{M} C(w_j,d_l)} \end{aligned} \tag{6.25}$$

where $C(w_j,d_l)$ is the count of w_j in the training document d_l. However, for some seen bigrams of the test document, the words of the bi-gram cannot be found together in any of the training documents. Their probabilities are computed as:

$$P(w_j|w_i,d_t) = \sum_{l=1}^{M}(\sum_{k=1}^{K} P(w_j|t_k,d_l)P(t_k|w_i,d_t,d_l))P(d_l) \tag{6.26}$$

where $P(d_l) = 1/M$.

The remaining zero probabilities of the obtained matrix $P(w_i|w_j,d_t)$ are then computed by using back-off smoothing. To capture the local lexical regularities, the model is then interpolated with a back-off trigram background model.

6.7 Comparison of UBPLSA, CPLSA and DCPLSA Models

In all the models, the topic is conditioned to the bigram history context and the document. The UBPLSA, CPLSA and DCPLSA models are differentiated by the word probabilities. In the UBPLSA model [7], bigram probabilities for topics are trained, which are unsmoothed in the training procedure that results an incorrect topic weights of the unseen test document. This is because, for some history contexts, the topic probabilities are assigned zeros as the bigram probabilities in the training model are unsmoothed. So, using the UBPLSA model, some of the bigram probabilities of the test document cannot be computed. However, they are later smoothed in the test phase. That approach is not practical, as for the corresponding history context some other bigrams may be present in the training set. The CPLSA model [43] solves this problem as it uses unigram word probabilities for topics, which helps to assign probabilities to all possible bigrams of the seen history context in the training procedure. Therefore, the model can compute the seen bigram probabilities of the test document. The word probabilities in the CPLSA model are trained by using the sum of the bigram events in all documents. However, the words may appear in different documents to describe different topics. In the DCPLSA model, the unigram word probabilities for topics are further conditioned on the document, which helps to compute the word probabilities for topics in different documents. The CPLSA model requires less memory and complexity than the other models. The memory and complexity requirements for the DCPLSA model [50] are less than the UBPLSA model if the number of seen bigrams is higher than the product of the number of vocabulary words and the documents. As the UBPLSA model, the CPLSA model and the proposed DCPLSA model can also be extended to the n-gram case with increasing complexity and memory space requirements.

6.8 Complexity Analysis of the UBPLSA, CPLSA and DCPLSA Models

The numbers of free parameters for the UBPLSA, CPLSA and DCPLSA models are $V(V-1)K+(K-1)VM$, $(V-1)K+(K-1)VM$, and $(V-1)KM+(K-1)VM$ respectively. Here, V, K, and M represent the number of words, the number of topics and the number of documents, respectively. From the above discussion, we note that the CPLSA model needs fewer parameters, hence requires smaller memory space than the other models. The

DCPLSA model requires fewer parameters than the UBPLSA model as long as the number of documents M is less than the number of vocabulary words V.

In the E-step of the EM algorithm, we have to compute $P(t_k|w_j, w_i, d_l)$ for all i, j, k, l. Therefore, the time complexity of the UBPLSA model [7], the CPLSA model [43] and the DCPLSA model is $O(V^2MK)$. The time complexities for the M-step are $O(KMB)$, $O(VMK)$ and $O(VM^2K)$ for the UBPLSA, the CPLSA and the proposed DCPLSA models respectively. Here, B is the average number of word pairs in the training documents [7]. The size of B is obviously greater than the size of V. Therefore, the CPLSA model also needs less training time than the other models. The DCPLSA model can require less training time as long as $V \times M$ is less than B.

6.9 Experiments

6.9.1 Data and Parameters

We randomly selected 500 documents from the '87-89 WSJ corpus [71] for training the UBPLSA, the CPLSA and the DCPLSA models. The total number of words in the documents is 224,995. We used the 5K non-verbalized punctuation closed vocabulary from which we removed the MIT stop word list [3] and the infrequent words that occur only once in the training documents. After these removals, the total number of vocabulary words is 2628. We could not consider more training documents due to the higher computational cost and huge memory requirements for the UBPLSA model [7] and the DCPLSA models. For the same reason, we train only the bigram UBPLSA, CPLSA and DCPLSA models. Also, we used the same number of documents for the PLSA and CPLSA models for valid comparison. To capture the local lexical regularity, the topic models are interpolated with a back-off trigram background model. The trigram background model is trained on the '87-89 WSJ corpus using the back-off version of the Witten-Bell smoothing; 5K non-verbalized punctuation closed vocabulary and the cutoffs 1 and 3 on the bi-gram and tri-gram counts respectively are incorporated. The interpolation weights are computed by optimizing on the held-out data. The results of the experiments are noted on the evaluation test set November 1992 (330 sentences, 5353 words) ARPA CSR benchmark test data for 5K vocabularies [71, 101].

6.9.2 Experimental Results

We tested the above LM approaches for various sizes of topics. We performed the experiments five times and the results are averaged. The perplexity results are described in

Table 6.1.

Table 6.1 Perplexity results of the topic models

Language Model	20 Topics	40 Topics
Background (B)	69.0	69.0
B+PLSA	62.0	61.9
B+UBPLSA	59.0	58.7
B+CPLSA	57.5	55.8
B+DCPLSA	55.5	53.8

From Table 6.1, we can note that the perplexities are decreased with increasing topic size. The UBPLSA model [7] outperforms the PLSA [33] models and the CPLSA model shows better results than the PLSA [33] and the UBPLSA [7] models respectively. The proposed DCPLSA model outperforms the PLSA [33], the UBPLSA [7] and the CPLSA [43] models respectively. The B+DCPLSA model [50] yields perplexity reduction of about 19.6% (69.0 to 55.5), 10.5% (62.0 to 55.5), 5.9% (59.0 to 55.5) and 3.8% (57.5 to 55.5) for 20 topics and about 22.0% (69.0 to 53.8), 13.1% (61.9 to 53.8), 8.3% (58.7 to 53.8) and 3.6% (55.8 to 53.8) for 40 topics, over the background (B) model, B+PLSA model [33], the B+UBPLSA [7] and the B+CPLSA [43] approaches respectively.

We performed the paired t-test on the perplexity results of the above models with a significance level of 0.01. The p-values for different topic sizes are described in Table 6.2. From Table 6.2, we can note that all p-values are less than the significance level of 0.01.

Table 6.2 p-values obtained from the paired t test on the perplexity results

Language Model	20 Topics	40 Topics
B+UBPLSA and B+CPLSA	$6.0E$-11	$2.8E$-14
B+CPLSA and B+DCPLSA	$6.5E$-12	$3.1E$-13

Therefore, the perplexity improvements of the proposed DCPLSA model [50] over the CPLSA model [43] are statistically significant. Also, the CPLSA model [43] is statistically better than the UBPLSA model [7].

We evaluated the WER experiments using lattice rescoring. In the first pass, we used the back-off trigram background language model for lattice generation. In the second pass, we applied the interpolated form of the PLSA, UBPLSA, CPLSA and DCPLSA models for

lattice rescoring. The experimental results are explained in Figure 6.3. From Figure 6.3, we can note that the proposed DCPLSA model [50] yields significant WER reductions of about 25% (4.0% to 3.0%), 14.3% (3.5% to 3.0%), 9.1% (3.3% to 3%) and 6.25% (3.2% to 3.0%) for 20 topics and about 27.5% (4.0% to 2.9%), 17.1% (3.5% to 2.9%), 14.7% (3.4% to 2.9%) and 9.4% (3.2% to 2.9%) for 40 topics, over the background model, PLSA model [33], the UBPLSA [7] and the CPLSA [43] approaches respectively.

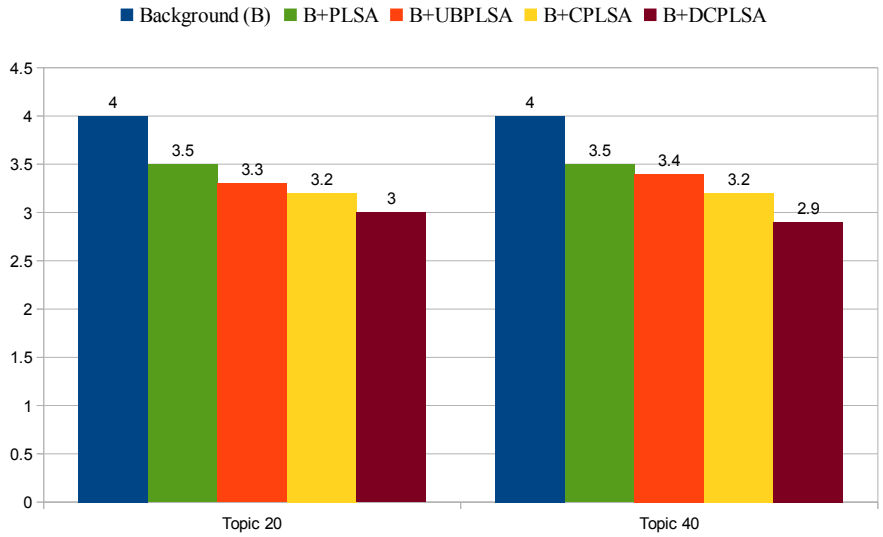

Fig. 6.3 WER results (%) for different topic sizes

We also performed a paired t test on the WER results for the interpolated models with a significance level of 0.01. The p-values of the test are explained in Table 6.3. From Table 6.3, we can see that the p-values are smaller than the significance level of 0.01. Therefore, the WER improvements of the proposed DCPLSA model are statistically significant.

Table 6.3 p-values obtained from the paired t test on the WER results

Language Model	20 Topics	40 Topics
B+UBPLSA and B+CPLSA	4.7E-06	9.3E-06
B+CPLSA and B+DCPLSA	6.9E-06	1.5E-07

6.10 Summary

In this chapter, we introduce a new document-based CPLSA language model for speech recognition. This is an extended work of the CPLSA [43] model, which was investigated

to overcome the limitations of an UBPLSA [7] model. The CPLSA model is similar to the PLSA model except that the topic is further conditioned on the history context. As the UBPLSA model assigns probabilities to the seen bigrams only in the training phase, the model gives zero topic probabilities for some history context of the test document that are seen in the training set. Therefore, some of the bigram probabilities of the test document cannot be computed using the training model, which is not practical. The CPLSA model can compute all the possible bigram probabilities of the seen history context in the training set. It helps to find the topic weights of the unseen test documents correctly and hence gives the correct bigram probabilities to the test document. However, the CPLSA model trains the unigram probabilities for topics by using the sum of bigram events in all documents where the words may appear to describe different topics in different documents. We identify this problem in the CPLSA model and propose the DCPLSA model where document-wise unigram probabilities for topics are trained.

Chapter 7

Interpolated LDLM

In this chapter, we propose a language modeling (LM) approach using interpolated distanced n-grams in a latent Dirichlet language model (LDLM) [20] for speech recognition. The LDLM relaxes the bag-of-words assumption and document topic extraction of latent Dirichlet allocation (LDA). It uses default background n-grams where topic information is extracted from the $(n\text{-}1)$ history words through Dirichlet distribution in calculating n-gram probabilities. The model does not capture the long-range information from outside of the n-gram events that can improve the language modeling performance. We present an interpolated LDLM (ILDLM) by using different distanced n-grams. Here, the topic information is exploited from $(n\text{-}1)$ history words through the Dirichlet distribution using interpolated distanced n-grams. The n-gram probabilities of the model are computed by using the distanced word probabilities for the topics and the interpolated topic information for the histories. In addition, we incorporate a cache-based LM, which models the re-occurring words, through unigram scaling to adapt the LDLM and ILDLM models that model the topical words [44].

7.1 Introduction

In [8], a PLSA technique enhanced with long-distance bigrams was used to incorporate the long-term word dependencies in determining word clusters. This motivates us to use the long-distance n-grams using interpolation to induce the long-term word dependencies into the LDLM model. In this chapter, we capture the long-range information into the LDLM using the interpolated distanced n-grams and cache based models. The n-gram probabilities of the proposed ILDLM model are computed by mixing the component distanced word probabilities for topics and the interpolated topic information for histories. Furthermore, we incorporate a cache-based LM into the LDLM and ILDLM models as the cache-based LM

models different parts of the language than the topic models.

7.2 LDLM

LDA is used to compute the document probability by using the topic structure at the document level, which is inconsistent with the language model for speech recognition, where the n-gram regularities are characterized [20]. The LDLM was developed to model the n-gram events for speech recognition. The graphical model for LDLM is described in Figure 7.1. Here, H and V represent the number of histories and the size of vocabulary, respectively [20]. In this model, the topic mixture vector θ is generated by the history-dependent Dirichlet parameter. A parameter matrix U is merged in the Dirichlet model to consider the word occurrence in each history. The history is represented as a vector h [20]. The model can be described as:

- For each history vector h, the topic mixture vector θ is drawn by a Dirichlet distribution:
$$P(\theta|h,U) \propto \prod_{k=1}^{K} \theta_{t_k}^{u_{t_k}^T h-1}, \tag{7.1}$$
where $u_{t_k}^T$ is the ${t_k}^{th}$ row vector of the matrix U [20].

- For each predicted word w_i of the n-gram events from a multinomial distribution with parameter β, a topic t_k is chosen by using a multinomial distribution with parameter θ. The joint probability of the variable θ, t_k, and w_i conditioned on h can be computed as:
$$P(\theta,t_k,w_i|h,U,\beta) = P(\theta|h,U)P(t_k|\theta)P(w_i|t_k,\beta) \tag{7.2}$$

- The conditional probability in the n-gram language model can thus be obtained as:
$$P(w_i|h,U,\beta) = \int P(\theta|h,U) \sum_{k=1}^{K} P(t_k|\theta)P(w_i|t_k,\beta)d\theta, \tag{7.3}$$
where the integral is computed as:
$$P(t_k|h,U) = \int P(\theta|h,U)P(t_k|\theta)d\theta = \frac{u_{t_k}^T h}{\sum_{j=1}^{K} u_{t_j}^T h}, \tag{7.4}$$
which is an expectation of a Dirichlet distribution of latent topic t_k [20].

Therefore, the probability of an n-gram event using the LDLM (Equation 7.3 and 7.4) can be written as [20]:

$$P_{LDLM}(w_i|h,U,\beta) = \sum_{k=1}^{K} P(w_i|t_k,\beta)\frac{u_{t_k}^T h}{\sum_{j=1}^{K} u_{t_j}^T h}. \tag{7.5}$$

The parameters (U,β) of the model are computed using the EM procedure [20].

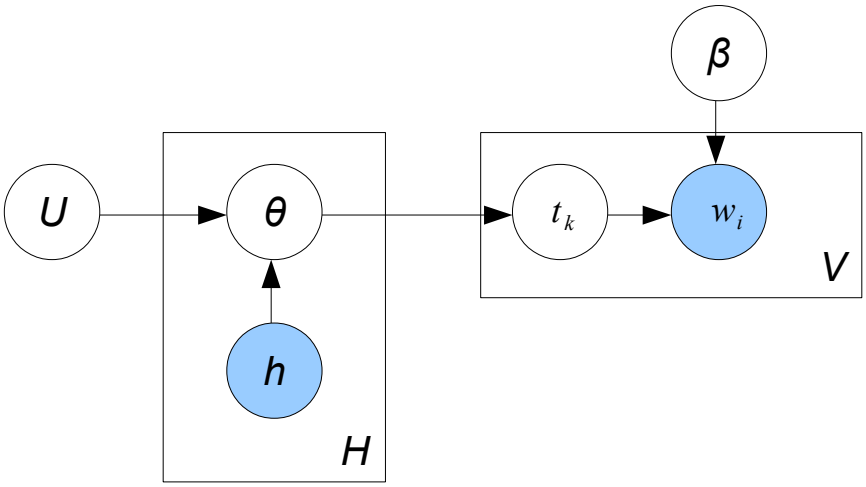

Fig. 7.1 The graphical model of the LDLM. Shaded circles represent observed variables.

7.3 Proposed ILDLM

In [8], a PLSA technique enhanced with long-distance bigrams was used to incorporate the long-term word dependencies in determining word clusters. This motivates us to use the long-distance n-grams using interpolation to induce the long-term word dependencies into the LDLM model.

The LDLM does not capture the long-range information from outside of the n-gram events [20]. To incorporate the long-range information into the LDLM, we propose an ILDLM where the topic information is extracted from interpolated distance n-gram histories through a Dirichlet distribution in calculating the language model probability. In this model, we interpolate the distanced n-gram event into the original n-gram events of the LDLM. The n-gram probability using the ILDLM can be defined as [8]:

$$P_{ILDLM}(w_i|h) = \sum_{k=1}^{K}[\sum_{D}\lambda_D P_D(w_i|t_k,\beta_D)]\frac{u_{t_k}^T h}{\sum_{j=1}^{K} u_{t_j}^T h}, \tag{7.6}$$

where λ_D are the weights for each component probability estimated on the held-out data using the EM algorithm. D represents the distance between words in the n-gram events. $D = 1$ describes the default n-grams. For example, the distanced n-grams of the phrase "Speech in Life Sciences and Human Societies" are described in Table 7.1 for the distance $D = 1, 2$.

Table 7.1 Distanced n-grams for the phrase "Speech in Life Sciences and Human Societies"

D	Bigrams	Trigrams
1	*Speech in, in Life, Life Sciences, Sciences and, and Human, Human Societies*	*Speech in Life, in Life Sciences, Life Sciences and, Sciences and Human, and Human Societies*
2	*Speech Life, in Sciences, Life and, Sciences Human, and Societies*	*Speech Life and, in Sciences Human, Life and Societies*

The parameters of the ILDLM model are computed using the EM procedure by maximizing the marginal distribution of the training data:

$$\sum_{h,w_i,D} n_D(h,w_i) \log P_{ILDLM}(w_i|h), \tag{7.7}$$

where $n_D(h,w_i)$ are the distanced n-grams. In the E-step, the auxiliary function of the new estimates U', β'_D given current estimates U, β_D is calculated by taking the expectation of the marginal likelihood function of Equation 7.7 over the hidden variable t_k [20]:

$$\begin{aligned} Q(U', \beta'_D | U, \beta_D) &= \sum_{h,i',D} n_D(h,w_{i'}) \lambda_D E_{t_k}[\log P_D(w_{i'}, t_k | h, U', \beta'_D) | U, \beta_D] \\ &= \sum_{h,i',D} n_D(h,w_{i'}) \sum_{k=1}^{K} \lambda_D P_D(t_k | h, w_{i'}, U, \beta_D) \log P_D(w_{i'}, t_k | h, U', \beta'_D) \\ &= \sum_{h,i',D} n_D(h,w_{i'}) \sum_{k=1}^{K} \lambda_D P_D(t_k | h, w_{i'}, U, \beta_D) \times \log[P_D(w_{i'} | t_k, \beta'_D) \frac{u'^T_{t_k} h}{\sum_{j=1}^{K} u'^T_{t_j} h}], \end{aligned} \tag{7.8}$$

where $P_D(t_k|h,w_i,U,\beta_D)$ are the posterior probabilities of the latent variables, which are

calculated based on the current estimates as:

$$P_D(t_k|h,w_i,U,\beta_D) = \frac{P_D(w_i|t_k,\beta_D)u_{t_k}^T h}{\sum_{j=1}^K P_D(w_i|t_j,\beta_D)u_{t_j}^T h}. \tag{7.9}$$

In the M-step, the new estimates β'_D and U' are computed. The parameter $\beta'_{t_k,w_i,D}$ is updated as:

$$\beta'_{t_k,w_i,D} = \frac{\sum_h n_D(h,w_i)P_D(t_k|h,w_i,U,\beta_D)}{\sum_{i'}\sum_h n_D(h,w_{i'})P_D(t_k|h,w_{i'},U,\beta_D)}. \tag{7.10}$$

To compute the parameter U', the gradient ascent algorithm is used for maximization [1]. The gradient of the auxiliary function $\bigtriangledown_{u'_{t_k}} Q(U',\beta'_D|U,\beta_D)$ is given by

$$\sum_{h,i',D} n_D(h,w_{i'})\lambda_D P_D(t_k|h,w_{i'},U,\beta_D)[\frac{1}{u_{t_k}^{'T}h} - \frac{1}{\sum_{j=1}^K u_{t_j}^{'T}h}]h$$

Therefore, the new parameter u'_{t_k} at the $(t+1)$ iteration is updated by:

$$u_{t_k}^{'(t+1)} = u_{t_k}^{'(t)} + \eta \bigtriangledown_{u'_{t_k}} Q(U',\beta'_D|U,\beta_D), \tag{7.11}$$

where η is the learning parameter. The model parameters are then estimated with several EM iterations.

7.4 Incorporating Cache Models into LDLM and ILDLM Models

A Cache-based language model was used to increase the probability of words appearing earlier in a document that are likely to occur in the same document. The unigram cache model for a given history $h_c = w_{i-F},\ldots,w_i$, where F is the cache size, is defined as:

$$P_{CACHE}(w_i) = \frac{C(w_i,h_c)}{C(h_c)}, \tag{7.12}$$

where $C(w_i,h_c)$ is the number of occurrences of the word w_i within h_c and $C(h_c)$ is the number of words within h_c that belong to the vocabulary V [69, 97]. The LDLM/ILDLM capture topical words. To capture the local lexical regularities, the models are interpolated with a background (B) n-gram model as:

$$P_L(w_i|h) = \gamma P_B(w_i|h) + (1-\gamma)P_{LDLM/ILDLM}(w_i|h). \tag{7.13}$$

The cache-based LM models re-occurring words that are different from the background model (i.e., models short-range information), LDLM and ILDLM models (i.e., models topical words). Therefore, the cache model can be used to adapt the model $P_L(w|h)$ using unigram scaling as [68, 80]:

$$P_A(w_i|h) = \frac{P_L(w_i|h)\delta(w_i)}{Z(h)}, \tag{7.14}$$

with

$$Z(h) = \sum_{w_i} \delta(w_i).P_L(w_i|h), \tag{7.15}$$

where $Z(h)$ is a normalization term, which guarantees that the total probability sums to unity and $\delta(w_i)$ is a scaling factor, which is usually approximated as:

$$\delta(w_i) \approx (\frac{\rho P_{CACHE}(w_i) + (1-\rho)P_B(w_i)}{P_B(w_i)})^{\mu}, \tag{7.16}$$

where μ is a tuning factor between 0 and 1. In our experiments we used the value of μ as 1. We used the same procedure as [68] to compute the normalization term. To do this, an additional constraint is employed where the total probability of the seen transitions is unchanged:

$$\sum_{w_i:seen(h,w_i)} P_A(w_i|h) = \sum_{w_i:seen(h,w_i)} P_L(w_i|h). \tag{7.17}$$

The model $P_L(w_i|h)$ has standard back-off structure and the above constraint, so the model $P_A(w_i|h)$ has the following recursive formula:

$$P_A(w_i|h) = \begin{cases} \frac{\delta(w_i)}{Z_s(h)}.P_L(w_i|h) \text{ if } (h,w_i) \text{ exists} \\ bow(h).P_A(w_i|\hat{h}) \text{ otherwise} \end{cases}$$

where

$$Z_s(h) = \frac{\sum_{w_i:seen(h,w_i)} \delta(w_i).P_L(w_i|h)}{\sum_{w_i:seen(h,w_i)} P_L(w_i|h)}$$

and

$$bow(h) = \frac{1-\sum_{w_i:seen(h,w_i)} P_L(w_i|h)}{1-\sum_{w_i:seen(h,w_i)} P_A(w_i|\hat{h})},$$

where $bow(h)$ is the back-off weight of the context h to ensure that $P_A(w_i|h)$ sums to unity. $\hat{h}$ is the reduced word history of h. The term $Z_s(h)$ is used to do normalization similar to Equation 7.15 except the summation is considered only on the observed alternative words

with the equal word history h in the LM [96]. We describe the adaptation using the unigram scaling of cache models as (B+LDLM/ILDLM)*CACHE.

7.5 Experiments

7.5.1 Data and Parameters

The '87-89 WSJ corpus is used to train language models. The models are trained using the WSJ 5K non-verbalized punctuation closed vocabulary. A tri-gram background model is trained using the modified Kneser-Ney smoothing incorporating the cutoffs 1 and 3 on the bi-gram and tri-gram counts respectively. To reduce the computational and memory requirements using MATLAB, we trained only the bi-gram LDLM and ILDLM models. For ILDLM models, we considered bigrams for $D = 1, 2$. The learning parameter η is set to 0.01. A fixed cache size of $F = 400$ is used for the cache-based LM. The interpolation weights λ_D, γ and ρ are computed using the *compute-best-mix* program from the SRILM toolkit. They are tuned on the development test set. The results of the experiments are noted on the evaluation test set November 1993 (215 sentences, 3849 words) ARPA CSR benchmark test data for 5K vocabularies [71, 101].

7.5.2 Experimental Results

We keep the unigram (Equations 7.5 and 7.6) probabilities for topics of LDLM and ILDLM, and λ_D of component probabilities for ILDLM unchanged, and used them to compute the matrix U for the test document's histories [33]. The language models for LDLM and ILDLM are then computed using (Equations 7.5 and 7.6). The models are then interpolated with a back-off trigram background model to capture the local lexical regularities. Furthermore, a cache-based LM that models re-occurring words is integrated through unigram scaling with the LDLM/ILDLM that models topical words. We also show the results for PLSA models using unigram scaling where the PLSA unigrams are used in place of cache unigrams in Equation 7.16 and denoted as B*PLSA [33].

We tested the proposed approaches for various sizes of topics. The perplexity results of the experiments are described in Table 7.2. From Table 7.2, we can note that all the models outperform the background model and the performances are better with increasing topics. However, the proposed ILDLM model outperforms the PLSA and LDLM models in all forms (stand-alone, interpolated and unigram scaling) for all topic sizes.

Table 7.2 Perplexity results of the language models

Language Model	40 Topics	80 Topics
Background (B)	70.3	70.3
PLSA	517.8	514.8
LDLM	251.6	153.6
ILDLM	86.9	65.25
B*PLSA	66.6	66.5
B+LDLM	65.1	62.5
B+ILDLM	53.6	52.7
(B+LDLM)*CACHE	59.9	57.5
(B+ILDLM)*CACHE	49.3	48.5

We evaluated the WER experiments using lattice rescoring. In the first pass, we used the back-off trigram background language model for lattice generation. In the second pass, we applied the LM adaptation approaches for lattice rescoring. The experimental results are explained in Figure 7.2. From Figure 7.2, we can note that the proposed ILDLM model yields significant WER reductions of about 20.4% (7.6% to 6.05%), 18.2% (7.4% to 6.05%), and 12.3% (6.9% to 6.05%) for 40 topics and about 22.1% (7.6% to 5.92%), 20.0% (7.4% to 5.92%), and 11.6% (6.7% to 5.92%) for 80 topics, over the background model, PLSA model [33], and the LDLM [20] approaches respectively. The integration of cache-based models improves the performance as they carry different information (capture the dynamics of word occurrences in a cache) than the LDLM and ILDLM approaches. The cache unigram scaling of the ILDLM approach gives 9.4% (6.6% to 5.98%) and 8.3% (6.4% to 5.87%) WER reductions over the cache unigram scaling of the LDLM approach for 40 and 80 topics respectively. We can note that the addition of cache models improves the performance of LDLM (6.9% to 6.6% for 40 topics and 6.7% to 6.4% for 80 topics) more than for ILDLM (6.05% to 5.98% for 40 topics and 5.92% to 5.87% for 80 topics). This might be due to the fact that the ILDLM approach captures long-range information using the interpolated distanced bigrams. Therefore, it is proved that the proposed ILDLM approach includes long-range information into the LDLM model.

7.6 Summary

In this chapter, we proposed an integration of distanced n-grams into the original LDLM model [20]. The LDLM model extracted the topic information from the $(n\text{-}1)$ history words through a Dirichlet distribution in calculating the n-gram probabilities. However, it does

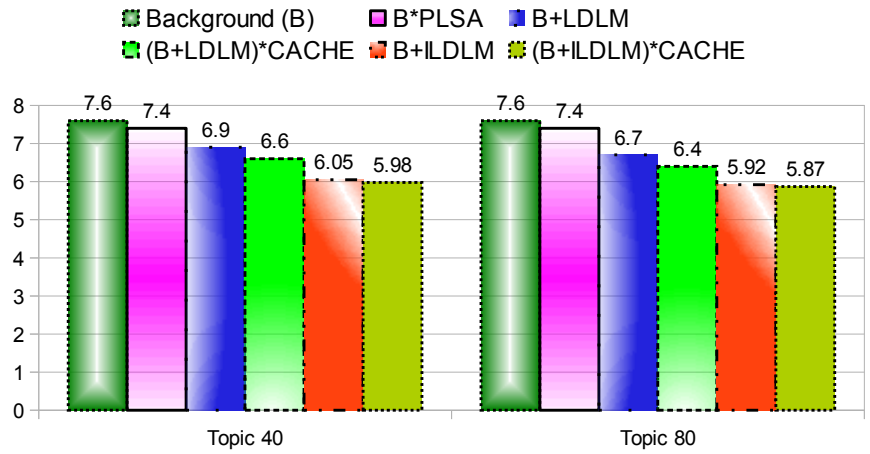

Fig. 7.2 WER Results (%) of the Language Models

not capture the long-range semantic information from outside of the n-gram events. The proposed ILDLM overcomes the shortcomings of LDLM by using the interpolated long-distance n-grams that capture the long-term word dependencies. Using the ILDLM, the topic information for the histories is trained using the interpolated distanced n-grams. The model probabilities are computed by weighting the component word probabilities for topics and the interpolated topic information for histories. We have seen that the proposed ILDLM approach yields significant perplexity and WER reductions over the LDLM approach using the WSJ corpus. Moreover, we incorporate a cache-based model into the topic models using unigram scaling for adaptation and have seen improved performances over the topic models. However, cache unigram scaling of the LDLM gives much better performance than the cache unigram scaling of the ILDLM. This proves that the proposed ILDLM approach captures long-range information of the language.

Chapter 8

Enhanced PLSA and Interpolated EPLSA

In this chapter, we introduce language modeling (LM) approaches using background n-grams and interpolated distanced n-grams for speech recognition using an enhanced probabilistic latent semantic analysis (EPLSA) derivation. PLSA is a bag-of-words model that exploits the topic information at the document level, which is inconsistent for the language modeling in speech recognition. We consider the word sequence in modeling the EPLSA model. Here, the predicted word of an n-gram event is drawn from a topic that is chosen from the topic distribution of the $(n\text{-}1)$ history words. The EPLSA model cannot capture the long-range topic information from outside of the n-gram event. The distanced n-grams are incorporated into interpolated form (IEPLSA) to cover the long-range information. A cache-based LM that models the re-occurring words is also incorporated through unigram scaling to the EPLSA and IEPLSA models, which models the topical words [45].

8.1 Introduction

In [8], a PLSA technique enhanced with long-distance bigrams was used to incorporate the long-term word dependencies in determining word clusters. This motivates us to present LM approaches for speech recognition using distanced n-grams. In this chapter, we use default n-grams using enhanced PLSA derivation to form the EPLSA n-gram model. Here, the observed n-gram events contain the history words and the predicted word. The EPLSA model extracts the topic information from history words and the current word is then predicted based on the topic information of the history words. However, the EPLSA model does not capture the topic information from outside of the n-gram events. We propose in-

terpolated distanced n-grams (IEPLSA) and cache based models to capture the long-term word dependencies into the EPLSA model. The n-gram probabilities of the IEPLSA model are computed by mixing the component distanced word probabilities for topics and the interpolated topic information for histories. Furthermore, a cache-based LM is incorporated into the EPLSA and IEPLSA models as the cache-based LM models a different part of the language than EPLSA/IEPLSA models.

8.2 Proposed EPLSA and IEPLSA Models

8.2.1 EPLSA

Representing a document d_j as a sequence of words, the joint distribution of the document and the previous $(n\text{-}1)$ history words h of the current word w_i can be described as [8]:

$$P(d_j,h) = P(h) \prod_{w_i \in d_j} P_D(w_i|h), \tag{8.1}$$

where $P_D(w_i|h)$ is the distanced n-gram model. Here, D represents the distance between the words in the n-grams. Therefore, the probability $P_D(w_i|h)$ can be computed similar to the PLSA derivation [8, 33]. For $D = 1$, $P_D(w_i|h)$ is the default background n-gram and we define it as the enhanced PLSA (EPLSA) model. The graphical model of the EPLSA model can be described in Figure 8.1. The equations for the EPLSA model are:

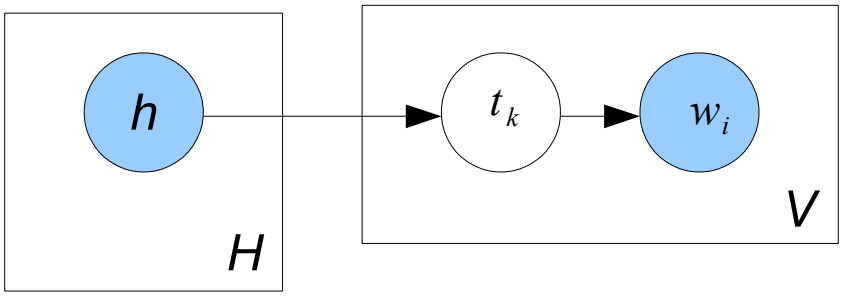

Fig. 8.1 The graphical model of the EPLSA model. The shaded circle represents the observed variables. H and V describe the number of histories and the size of vocabulary.

$$P_{EPLSA}(w_i|h) = \sum_{k=1}^{K} P(w_i|t_k)P(t_k|h). \tag{8.2}$$

The parameters of the model are computed using the EM algorithm as: E-step:

$$P(t_k|h,w_i) = \frac{P(w_i|t_k)P(t_k|h)}{\sum_{k'=1}^{K} P(w_i|t_{k'})P(t_{k'}|h)}, \tag{8.3}$$

M-step:

$$P(w_i|t_k) = \frac{\sum_h n(h,w_i)P(t_k|h,w_i)}{\sum_{i'}\sum_h n(h,w_{i'})P(t_k|h,w_{i'})}, \tag{8.4}$$

$$P(t_k|h) = \frac{\sum_{i'} n(h,w_{i'})P(t_k|h,w_{i'})}{\sum_{k'}\sum_{i'} n(h,w_{i'})P(t_{k'}|h,w_{i'})}. \tag{8.5}$$

8.2.2 IEPLSA

In [8], a PLSA technique enhanced with long-distance bigrams was used to incorporate the long-term word dependencies in determining word clusters. This motivates us to use the long-distance n-grams using interpolation to induce the long-term word dependencies into the EPLSA model.

The EPLSA model does not capture the long-distance information. To incorporate the long-range characteristics, we used the distanced n-grams in the EPLSA model. Incorporating the interpolated distance n-grams in the EPLSA, the model can be written as [8]:

$$P_{IEPLSA}(w_i|h) = \sum_{k=1}^{K}[\sum_{D}\lambda_D P_D(w_i|t_k)]P(t_k|h), \tag{8.6}$$

where λ_D are the weights for each component probability estimated on the held-out data using the EM algorithm and $P_D(w_i|t_k)$ is the word probabilities for topic t_k obtained by using the distanced n-grams in the IEPLSA training. D represents the distance between words in the n-gram events. $D = 1$ describes the default n-grams. For example, the distanced n-grams of the phrase "Speech in Life Sciences and Human Societies" are described in Table 8.1 for the distance $D = 1,2$.

The parameters of the IEPLSA model can be computed as: E-step:

$$P_D(t_k|h,w_i) = \frac{P_D(w_i|t_k)P(t_k|h)}{\sum_{k'=1}^{K} P_D(w_i|t_{k'})P(t_{k'}|h)}, \tag{8.7}$$

M-step:

$$P_D(w_i|t_k) = \frac{\sum_h n_D(h,w_i)P_D(t_k|h,w_i)}{\sum_{i'}\sum_h n_D(h,w_{i'})P_D(t_k|h,w_{i'})}, \tag{8.8}$$

Table 8.1 Distanced n-grams for the phrase "Speech in Life Sciences and Human Societies"

D	Bigrams	Trigrams
1	*Speech in, in Life, Life Sciences, Sciences and, and Human, Human Societies*	*Speech in Life, in Life Sciences, Life Sciences and, Sciences and Human, and Human Societies*
2	*Speech Life, in Sciences, Life and, Sciences Human, and Societies*	*Speech Life and, in Sciences Human, Life and Societies*

$$P(t_k|h) = \frac{\sum_{i'} \sum_D \lambda_D n_D(h, w_{i'}) P_D(t_k|h, w_{i'})}{\sum_{k'} \sum_{i'} \sum_D \lambda_D n_D(h, w_{i'}) P_D(t_k|h, w_{i'})}. \tag{8.9}$$

8.3 Comparison of PLSA, PLSA Bigram and EPLSA/IEPLSA

PLSA [33] is a bag-of-words model where the document probability is computed by using the topic structure at the document level. This is inappropriate for the language model in speech recognition. PLSA bigram models were introduced where the bigram probabilities for each topic are modeled and the topic is conditioned on the document [82] or bigram history and the document [7]. In either approach, the models require V distributions for each topic, where V is the size of the vocabulary. Therefore, the size of the parameters grows exponentially with increasing n-gram order. In contrast, the EPLSA/IEPLSA models developed the word distributions given the history words. The history information is used to form the topic distributions, then the probability of the predicted word is computed given the topic information of the histories. Therefore, the parameter number grows linearly with V [20].

8.4 Incorporating the Cache Model Through Unigram Scaling

A Cache-based language model was used to increase the probability of words appearing in a document that are likely to re-occur in the same document. The unigram cache model for

a given history $h_c = w_{i-F}, \ldots, w_i$, where F is the cache size, is defined as:

$$P_{cache}(w_i) = \frac{n(w_i, h_c)}{n(h_c)} \tag{8.10}$$

where $n(w_i, h_c)$ is the number of occurrences of the word w_i within h_c and $n(h_c) <= F$ is the number of words within h_c that belongs to the vocabulary V [69, 97].

The EPLSA/IEPLSA models capture topical words. The models are then interpolated with a background (B) n-gram model to capture the local lexical regularities as:

$$P_L(w_i|h) = (1-\gamma)P_{EPLSA/IEPLSA}(w_i|h) + \gamma P_B(w_i|h). \tag{8.11}$$

As the cache-based LM (i.e., models re-occurring words) is different from the background model (i.e., models short-range information), EPLSA and IEPLSA models (i.e., model topical words), we can integrate the cache model to adapt the $P_L(w_i|h)$ through unigram scaling as [68, 80]:

$$P_A(w_i|h) = \frac{P_L(w_i|h)\delta(w_i)}{Z(h)}, \tag{8.12}$$

with

$$Z(h) = \sum_{w_i} \delta(w_i).P_L(w_i|h). \tag{8.13}$$

where $Z(h)$ is a normalization term, which guarantees that the total probability sums to unity, $P_L(w_i|h)$ is the interpolated model of the background and the EPLSA/IEPLSA model and $\delta(w_i)$ is a scaling factor that is usually approximated as:

$$\delta(w_i) \approx (\frac{\alpha P_{cache}(w_i) + (1-\alpha)P_B(w_i)}{P_B(w_i)})^{\mu}, \tag{8.14}$$

where μ is a tuning factor between 0 and 1. In our experiments we used the value of μ as 1. We used the same procedure as [68] to compute the normalization term. To do this, an additional constraint is employed where the total probability of the seen transitions is unchanged:

$$\sum_{w_i:seen(h,w_i)} P_A(w_i|h) = \sum_{w_i:seen(h,w_i)} P_L(w_i|h). \tag{8.15}$$

The model $P_L(w_i|h)$ has standard back-off structure and the above constraint, so the model $P_A(w_i|h)$ has the following recursive formula:

$$P_A(w_i|h) = \begin{cases} \frac{\delta(w_i)}{Z_o(h)}.P_L(w_i|h) \text{ if } (h,w_i) \text{ exists} \\ bow(h).P_A(w_i|\hat{h}) \text{ otherwise} \end{cases}$$

where

$$Z_s(h) = \frac{\sum_{w_i:seen(h,w_i)} \delta(w_i).P_L(w_i|h)}{\sum_{w_i:seen(h,w_i)} P_L(w_i|h)} \tag{8.16}$$

and

$$bow(h) = \frac{1-\sum_{w_i:seen(h,w_i)} P_L(w_i|h)}{1-\sum_{w_i:seen(h,w_i)} P_A(w_i|\hat{h})}, \tag{8.17}$$

where $bow(h)$ is the back-off weight of the context h to ensure that $P_A(w_i|h)$ sums to unity. $\hat{h}$ is the reduced word history of h. The term $Z_s(h)$ is used to do normalization similar to Equation 8.13 except the summation is considered only on the observed alternative words with the equal word history h in the LM [96].

8.5 Experiments

8.5.1 Data and Parameters

The '87-89 WSJ corpus is used to train language models. The models are trained using the WSJ 5K non-verbalized punctuation closed vocabulary. A tri-gram background model is trained using the modified Kneser-Ney smoothing incorporating the cutoffs 1 and 3 on the bi-gram and tri-gram counts respectively. To reduce the computational and memory requirements using MATLAB, we trained only the bi-gram EPLSA and IEPLSA models. For IEPLSA models, we considered bigrams for $D = 1,2$. A fixed cache size of $F = 400$ is used for the cache-based LM. The interpolation weights λ_D, γ and α are computed using the *compute-best-mix* program from the SRILM toolkit. They are tuned on the development test set. The results of the experiments are noted on the evaluation test set November 1993 (215 sentences, 3849 words) ARPA CSR benchmark test data for 5K vocabularies [71, 101].

8.5.2 Experimental Results

We used the folding-in procedure [33] to compute the PLSA, EPLSA and IEPLSA model probabilities. We keep the unigram probabilities for topics of PLSA, EPLSA and IEPLSA, and λ_D of component probabilities for IEPLSA unchanged, and used them to compute $P(t_k|d_t)$ for the test document d_t of the PLSA model and $P(t_k|h)$ for the test document

histories of the EPLSA and IEPLSA models. The language models for PLSA, EPLSA and IEPLSA are then computed. The remaining zero probabilities of the obtained matrix $P_{EPLSA/IEPLSA}(w_i|h)$ are computed by using back-off smoothing. The EPLSA and IEPLSA models are interpolated with a back-off trigram background model to capture the local lexical regularities. Furthermore, a cache-based LM that models re-occurring words is integrated through unigram scaling with the EPLSA and IEPLSA models, which describe topical words. We compared our approaches with a PLSA-based LM approach [33] using unigram scaling where the PLSA unigrams are used in place of cache unigrams in Equation 8.14 and denoted as B*PLSA.

We tested the proposed approach for various sizes of topics. The perplexity results are described in Table 8.2. From Table 8.2, we can note that all the models outperform the back-

Table 8.2 Perplexity results of the language models

Language Model	40 Topics	80 Topics
Background (B)	70.3	70.3
PLSA	517.8	514.8
EPLSA	192.9	123.3
IEPLSA	101.2	93.0
B*PLSA	66.6	66.5
B+EPLSA	62.9	59.7
B+IEPLSA	55.1	55.1
(B+EPLSA)*CACHE	58.0	55.1
(B+IEPLSA)*CACHE	50.7	50.7

ground model and the performances are better with increasing topics. The proposed EPLSA and IEPLSA models outperform the PLSA models in every form (stand-alone, interpolated, unigram scaling).

We evaluated the WER experiments using lattice rescoring. In the first pass, we used the back-off trigram background language model for lattice generation. In the second pass, we applied the LM adaptation approaches for lattice rescoring. The experimental results are explained in Figure 8.2. From Figure 8.2, we can note that the proposed EPLSA model yields significant WER reductions of about 10.5% (7.6% to 6.8%) and 8.1% (7.4% to 6.8%) for 40 topics, and about 15.8% (7.6% to 6.4%) and 13.5% (7.4% to 6.4%) for 80 topics, over the background model and the PLSA [33] approaches respectively. For the IEPLSA models, the WER reductions are about 19.6% (7.6% to 6.11%), 17.4% (7.4% to 6.11%), and 10.1% (6.8% to 6.11%) for 40 topics and about 20.4% (7.6% to 6.05%), 18.2% (7.4% to 6.05%),

and 5.5% (6.4% to 6.05%) for 80 topics, over the background model, PLSA [33] and EPLSA approaches respectively. The integration of cache-based models improves the performance as it carries different information (captures the dynamics of word occurrences in a cache) than the EPLSA and IEPLSA approaches. The cache unigram scaling of the IEPLSA approach gives 6.74% (6.52% to 6.08%) and 1.63% (6.13% to 6.03%) WER reductions over the cache unigram scaling of the EPLSA approach for 40 and 80 topics respectively. We can note that the addition of cache models improves the performance of EPLSA (6.8% to 6.52% for 40 topics and 6.4% to 6.13% for 80 topics) more than for IEPLSA (6.11% to 6.08% for 40 topics and 6.05% to 6.03% for 80 topics). This might be due to the fact that the IEPLSA approach captures long-range information using the interpolated distanced bigrams.

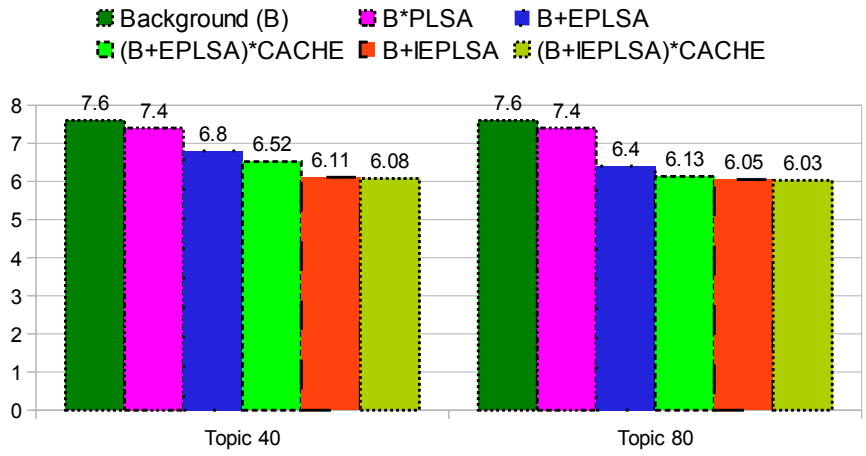

Fig. 8.2 WER Results (%) of the Language Models

8.6 Summary

In this chapter, we have proposed the EPLSA and IEPLSA models for speech recognition by using the background n-grams and the interpolated distanced n-grams respectively. The EPLSA model extracted the topic information from the $(n$-1$)$ history words in calculating the n-gram probabilities. However, it does not capture the long-range semantic information from outside of the n-gram events. The IEPLSA model overcomes the shortcomings of EPLSA by using the interpolated long-distance n-grams that capture the long-term word dependencies. Using the IEPLSA, the topic information for the histories are trained using the interpolated distanced n-grams. The model probabilities are computed by weighting the component word probabilities for topics and the interpolated topic information for the his-

tories. We have seen that the proposed EPLSA and IEPLSA approaches yield significant perplexity and WER reductions over the PLSA-based LM approach using the WSJ corpus. Moreover, we incorporate a cache-based model into the EPLSA and IEPLSA models using unigram scaling for adaptation and have seen improved performances. However, cache unigram scaling of the EPLSA gives much better performance over the EPLSA than the cache unigram scaling of the IEPLSA over the IEPLSA. This proves that the IEPLSA approach captures long-range information of the language.

Chapter 9

Interpolated DCLM

In this chapter, we propose a language modeling (LM) approach using interpolated distanced n-grams in a Dirichlet class language model (DCLM) [21] for speech recognition. The DCLM relaxes the bag-of-words assumption and document topic extraction of latent Dirichlet allocation (LDA). The latent variable of DCLM reflects the class information of an n-gram event rather than the topic in LDA. The DCLM model uses default background n-grams where class information is extracted from the $(n$-$1)$ history words through a Dirichlet distribution in calculating n-gram probabilities. The model does not capture the long-range information from outside of the n-gram window that can improve the language modeling performance. We present an interpolated DCLM (IDCLM) by using different distanced n-grams. Here, the class information is exploited from $(n$-$1)$ history words through the Dirichlet distribution using interpolated distanced n-grams. A variational Bayesian procedure is introduced to estimate the IDCLM parameters [47].

9.1 Introduction

In [21], the DCLM model was proposed to tackle the data sparseness and to extract the large-span information for the n-gram model. In this model, the topic structure in LDA is assumed to derive the hidden classes of histories in calculating the language model. A Bayesian class-based language model was presented where a variational Bayes-EM procedure was used to compute the model parameters. Also, a cache DCLM model was proposed to capture the long-distance information beyond the n-gram window. However, in the DCLM model [21], the class information of the history words was obtained from the n-gram events of the corpus. Here, the long-range information outside the n-gram window is not captured. In this chapter, we present an IDCLM model to capture the long-range

information in the DCLM using the interpolated distanced n-grams. The n-gram probabilities of the proposed IDCLM model are computed by mixing the component distanced word probabilities for classes and the interpolated class information for histories. Similar to the DCLM model, the parameters of the IDCLM model are computed by using the variational Bayesian-EM procedure.

9.2 DCLM

LDA is used to compute the document probability by using the topic structure at the document level, which is inconsistent with the language model for speech recognition where the n-gram regularities are characterized [21]. The DCLM was developed to model the n-gram events of the corpus for speech recognition. In DCLM, the class structure is described by Dirichlet densities and estimated from n-gram events. The graphical model of the DCLM for a text corpus that comprises n-gram events $\{w_{i-n+1}^{i-1}, w_i\}$ is described in Figure 9.1. Here, H and N_h represent the number of history events w_{i-n+1}^{i-1} and the number of collected words that occur following the history w_{i-n+1}^{i-1}, respectively. The (n-1) history words w_{i-n+1}^{i-1} are represented by a (n-1)$V \times 1$ vector h, consisting of n-1 block subvectors, with the entries of the seen words assigned to ones and those of unseen words assigned to zeros [21]. Here, V represents the size of the vocabulary. The vector h is then projected into a C-dimensional continuous class space using a class-dependent linear discriminant function:

$$g_c(h) = u_c^T h, \tag{9.1}$$

where u_c^T is the c^{th} row vector of matrix $U = [u_1, \cdots, u_C]$ [21]. The function $g_c(h)$ describes the class posterior probability $P(c|h)$, which is used in predicting the class information for an unseen history [21]. The model can be described as:

- For each history vector h, the class information c is drawn from a history-dependent Dirichlet prior θ, which is related to a global projection matrix U:

$$P(\theta|h,U) \propto \prod_{c=1}^{C} \theta_c^{g_c(h)-1}, \tag{9.2}$$

- For each predicted word w_i of the n-gram events from a multinomial distribution with parameter β, the associated class c_i is chosen by using a multinomial distribution with parameter θ. The joint probability of the variable θ, c_i, and w_i conditioned on h can

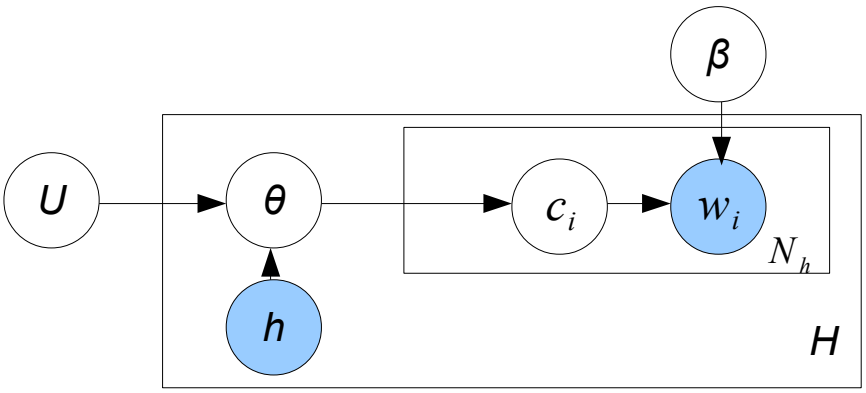

Fig. 9.1 The graphical model of the DCLM. Shaded circles represent observed variables.

be computed as:

$$P(\theta, c_i, w_i|h, U, \beta) = P(\theta|h, U)P(c_i|\theta)P(w_i|c_i, \beta). \tag{9.3}$$

- The conditional probability in the n-gram language model can thus be obtained as:

$$P(w_i|h, U, \beta) = \int P(\theta|h, U) \sum_{c_i=1}^{C} P(c_i|\theta)P(w_i|c_i, \beta)d\theta, \tag{9.4}$$

where the integral is computed as:

$$P(c_i|h, U) = \int P(\theta|h, U)P(c_i|\theta)d\theta = \frac{g_{c_i}(h)}{\sum_{j=1}^{C} g_j(h)}, \tag{9.5}$$

which is an expectation of a Dirichlet distribution of latent class c_i [21].

Therefore, the probability of an n-gram event using the DCLM (Equation 9.4 and 9.5) can be written as [21]:

$$P(w_i|h, U, \beta) = \sum_{c=1}^{C} P(w_i|c, \beta) \frac{g_c(h)}{\sum_{j=1}^{C} g_j(h)}. \tag{9.6}$$

The parameters (U,β) of the model are computed by using the variational bayesian EM (VB-EM) procedure [21].

9.3 Proposed IDCLM

The DCLM does not capture the long-range information from outside of the n-gram window [21]. To incorporate the long-range information into the DCLM, we propose an IDCLM where the class information is extracted from interpolated distance n-gram histories through a Dirichlet distribution in calculating the language model probability. In this model, we interpolate the distanced n-gram events into the original n-gram events of the DCLM. The graphical model of the IDCLM is described in Figure 9.2. In Figure 9.2, H_I contains the

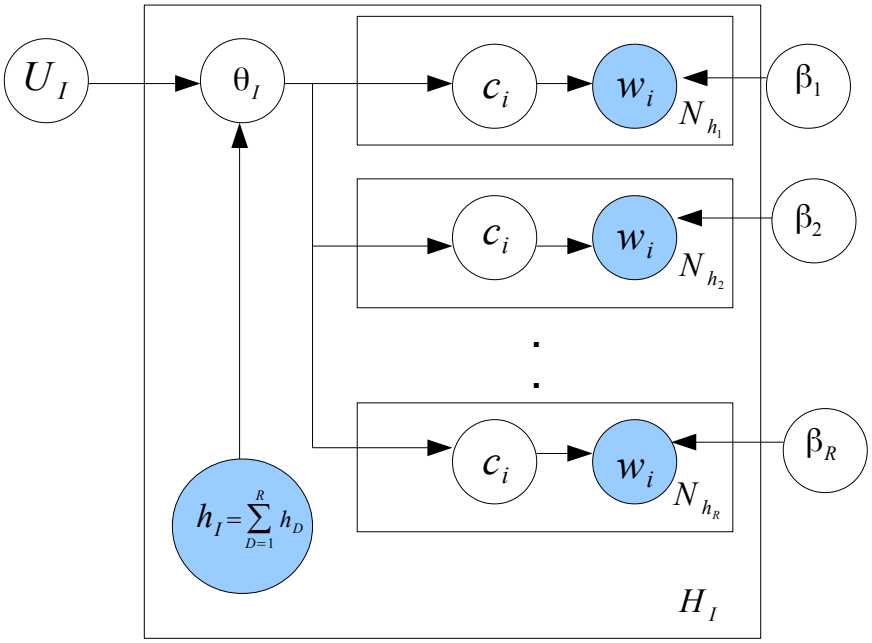

Fig. 9.2 The graphical model of the IDCLM. Shaded circles represent observed variables.

histories of all the distanced D n-grams, D represents the distance between words in the n-gram events, and R describes the maximum length of distance D. When $D = 1$, the n-grams are the default background n-grams. For example, the distanced tri-grams of the phrase "Interpolated Dirichlet Class Language Model for Speech Recognition" are described in Table 9.1 for the distances $D = 1, 2, 3$.

Here, the $(n\text{-}1)V$ dimensional discrete history vector h_I is projected into a C-dimensional continuous class space using a class-dependent linear discriminant function:

$$g_c(h_I) = u_{c,I}^T h_I \tag{9.7}$$

where h_I are the combined histories of all the distanced histories h_D and are defined as

D	Trigrams
1	*Interpolated Dirichlet Class, Dirichlet Class Language, Class Language Model, Language Model for, Model for Speech, for Speech Recognition*
2	*Interpolated Class Model, Dirichlet Language for, Class Model Speech, Language for Recognition*
3	*Interpolated Language Speech, Dirichlet Model Recognition*

Table 9.1 Distanced tri-grams for the phrase "Interpolated Dirichlet Class Language Model for Speech Recognition"

$h_I = \sum_{D=1}^{R} h_D$. Here, $\sum$ represents the logical *OR* operator. $u_{c,I}^{T}$ is the c^{th} row vector of the matrix U_I and $g_c(h_I)$ describes the class posterior probability $P(c|h_I)$.

The n-gram probability of the IDCLM model is computed as:

$$\begin{aligned} P_I(w_i|h_I,U_I,\beta_D) &= \sum_{c_i=1}^{C}\left\{\Big[\sum_D \lambda_D P_D(w_i|c_i,\beta_D)\Big]\times\int P(\theta_I|h_I,U_I)P(c_i|\theta_I)d\theta_I\right\} \\ &= \sum_{c=1}^{C}\Big[\sum_D \lambda_D\beta_{D,ic}\Big]\frac{g_c(h_I)}{\sum_{j=1}^{C} g_j(h_I)} \end{aligned} \tag{9.8}$$

where λ_D are the weights for each component probability estimated on the held-out data using the EM algorithm [8, 27].

The parameters of the IDCLM model are computed using the variational Bayes EM (VB-EM) procedure by maximizing the marginal distribution of the training data that contains a set of n-gram events $S_n = \{w_{i-n+1}^{i-1}, w_i\}$:

$$\begin{aligned} \log P(S_n|U_I,\beta_D) &= \sum_{(w_i,h_I)\in S_n} \log P_I(w_i|h_I,U_I,\beta_D) \\ &= \sum_{h_I}\log\left\{\int P(\theta_I|h_I,U_I)\times\left[\sum_D\prod_{j=1}^{N_{h_D}}\sum_{c_j=1}^{C}\lambda_D P_D(w_j|c_j,\beta_D)P(c_j|\theta_I)\right]d\theta_I\right\} \end{aligned} \tag{9.9}$$

where S_n contains all the distanced n-gram events, N_{h_D} represents the number of collected words that occur following the history h_D in D-distanced n-grams. In Equation 9.9, the summation is over all possible histories in training samples S_n. However, directly optimizing the Equation 9.9 is intractable [21]. A variational IDCLM is introduced where the marginal likelihood is approximated by maximizing the lower bound of Equation 9.9. The VB-EM procedure is required since the parameter estimation involves the latent variables of $\{\theta_I, c_{h_D} = \{c_i\}_{i=1}^{N_{h_D}}\}$.

The lower bound $L(U_I, \beta_D; \hat{\gamma}_I, \hat{\phi}_D)$ is given by:

$$\sum_{h_I}\left\{\log\Gamma\left(\sum_{c=1}^{C} g_c(h_I)\right) - \sum_{c=1}^{C}\log\Gamma(g_c(h_I)) + \sum_{c=1}^{C}(g_c(h_I)-1)\times\left(\Psi(\gamma_{h_I,c}) - \Psi\Big(\sum_{j=1}^{C}\gamma_{h_I,j}\Big)\right)\right\}$$
$$+\sum_{D}\sum_{h_D}\sum_{i=1}^{N_{h_D}}\sum_{c=1}^{C}\lambda_D\phi_{h_D,ic}\left(\Psi(\gamma_{h_I,c}) - \Psi\Big(\sum_{j=1}^{C}\gamma_{h_I,j}\Big)\right)$$
$$+\sum_{D}\sum_{h_D}\sum_{i=1}^{N_{h_D}}\sum_{c=1}^{C}\sum_{v=1}^{V}\lambda_D\phi_{h_D,ic}\delta(w_v,w_i)\log\beta_{D,vc} - \sum_{h_I}\left\{\log\Gamma\Big(\sum_{c=1}^{C}\gamma_{h_I,c}\Big) - \sum_{c=1}^{C}\log\Gamma(\gamma_{h_I,c})\right.$$
$$\left.+\sum_{c=1}^{C}(\gamma_{h_I,c}-1)\left(\Psi(\gamma_{h_I,c}) - \Psi\Big(\sum_{j=1}^{C}\gamma_{h_I,j}\Big)\right)\right\} - \sum_{D}\sum_{h_D}\sum_{i=1}^{N_{h_D}}\sum_{c=1}^{C}\lambda_D\phi_{h_D,ic}\log\phi_{h_D,ic}$$

where $\Psi(.)$ is the derivative of the log gamma function, and is known as a digamma function [21]. The history-dependent variational parameters $\{\hat{\gamma}_{h_I} = \hat{\gamma}_{h_I,c}, \hat{\phi}_{h_D} = \hat{\phi}_{h_D,vc}\}$, corresponding to the latent variables θ_I, c_{h_D}, are then estimated in the VB-E step by setting the differentials $(\partial L(\gamma))/(\partial\gamma_{h_I,c})$ and $(\partial L(\phi))/(\partial\phi_{h_D,ic})$ to zero respectively [21]:

$$\hat{\gamma}_{h_I,c} = g_c(h_I) + \sum_{D}\sum_{i=1}^{N_{h_D}}\lambda_D\phi_{h_D,ic} \tag{9.10}$$

$$\hat{\phi}_{h_D,ic} = \frac{\beta_{D,ic}\exp\left[\Psi(\gamma_{h_I,c}) - \Psi(\sum_{j=1}^{C}\gamma_{h_I,j})\right]}{\sum_{l=1}^{C}\beta_{D,il}\exp\left[\Psi(\gamma_{h_I,l}) - \Psi(\sum_{j=1}^{C}\gamma_{h_I,j})\right]}. \tag{9.11}$$

In computing $\hat{\phi}_{h_D,ic}$ the corresponding $\gamma_{h_D,c}$ is used in Equation 9.11. With the updated $\hat{\gamma}_{h_I}, \hat{\phi}_{h_D}$ in the VB-E step, the IDCLM parameters $\{U_I, \beta_D\}$ are estimated in the VB-M step as [21]:

$$\hat{\beta}_{D,vc} = \frac{\sum_{h_D}\sum_{i=1}^{N_{h_D}}\lambda_D\hat{\phi}_{h_D,ic}\delta(w_v,w_i)}{\sum_{m=1}^{V}\sum_{h_D}\sum_{i=1}^{N_{h_D}}\lambda_D\hat{\phi}_{h_D,ic}\delta(w_m,w_i)}, \tag{9.12}$$

where $\sum_{v=1}^{V}\beta_{D,vc}$=1 and $\delta(w_v,w_i)$ is the Kronecker delta function that equals one when vocabulary word w_v is identical to the predicted word w_i and equals zero otherwise. The gradient ascent algorithm is used to calculate the parameters $\hat{U}_I = [\hat{u}_{1,I}, \cdots, \hat{u}_{C,I}]$ by updating the gradient $\bigtriangledown_{u_{c,I}}$as [21]:

$$\bigtriangledown_{u_{c,I}} \leftarrow \bigtriangledown_{u_{c,I}} + \sum_{h_I}\left[\Psi\Big(\sum_{j=1}^{C}g_j(h_I)\Big) - \Psi(g_c(h_I)) + \Psi(\hat{\gamma}_{h_I,c}) - \Psi\Big(\sum_{j=1}^{C}\hat{\gamma}_{h_I,j}\Big)\right].h_I \tag{9.13}$$

The n-gram probabilities $P_{d_t}(w_i|h_t, U_I, \beta_D)$ of the test document d_t are then computed using Equation 9.8. To capture the local lexical regularities, the model $P_{d_t}(w_i|h_t, U_I, \beta_D)$ is then interpolated with the background (B) trigram model as:

$$P_L(w_i|h) = \mu P_B(w_i|h) + (1-\mu)P_{d_t}(w_i|h_t, U_I, \beta_D). \tag{9.14}$$

9.4 Comparison of DCLM and IDCLM Models

In the DCLM model, the class information for the $(n-1)$ history words is obtained by using the n-gram counts in the corpus. The current word is predicted from the history-dependent Dirichlet parameter, which is controlled by a matrix U and corpus-based histories h [21]. In contrast, the IDCLM model captures long-range information by incorporating distanced n-grams. Here, the class information is exploited for the interpolated $(n-1)$ history words h_I that are obtained from all the distanced n-gram events. Both the DCLM and IDCLM exploit the word distribution given the history words. They perform the history clustering of the corpus. For the DCLM model, the number of parameters $\{U, \beta\}$ increases linearly with the number of history words and is given by $(n-1)CV + CV$. For the IDCLM model, the number of parameters $\{U_I, \beta_D\}$ increases linearly with the number of history words and distance D and is given by $((n-1)CV + CVD)$. The time complexity of DCLM and IDCLM are $O(HVC)$ and $O(H_IVCD)$ with H corpus-based histories, H_I corpus-based interpolated histories, V vocabulary words, D distances and C classes.

9.5 Experiments

9.5.1 Data and Parameters

The LM approaches are evaluated using the Wall Street Journal (WSJ) corpus [71]. The '87-89 WSJ corpus is used to train language models. The background trigrams are trained using the back-off version of the Witten-Bell smoothing and the 5K non-verbalized punctuation closed vocabulary. We train the trigram IDCLM model using $R = 2$ and $R = 3$. Ten EM iterations in the VB-EM procedure were used. The initial values of the entries in the matrix β, β_D were set to be $1/V$ and those in U, U_I were randomly set in the range [0,1]. To update the variational parameters in the VB-E step, one iteration was used. The VB-M step was executed to update the parameters U, U_I by three iterations [21]. To capture the local lexical regularity, trigrams of various methods are interpolated with the background trigrams. The

interpolation weights λ_D and μ are computed by optimizing on the held-out data according to the metric of perplexity. The experiments are evaluated on the evaluation test, which is a total of 330 test utterances from the November 1992 ARPA CSR benchmark test data for vocabularies of 5K words [71, 101].

9.5.2 Experimental Results

Due to the higher memory and training time requirements for the IDCLM model, we trained the DCLM and IDCLM models for class sizes of 10 and 20. The perplexity and WER results are described in Table 9.2 and Figure 9.3 respectively.

Table 9.2 Perplexity results of the models

Language Model	10 Classes	20 Classes
Background (B)	109.4	109.4
B+Class	106.65	107.0
B+DCLM	100.2	100.45
B+IDCLM (R=2)	98.0	97.9
B+IDCLM (R=3)	95.6	95.4

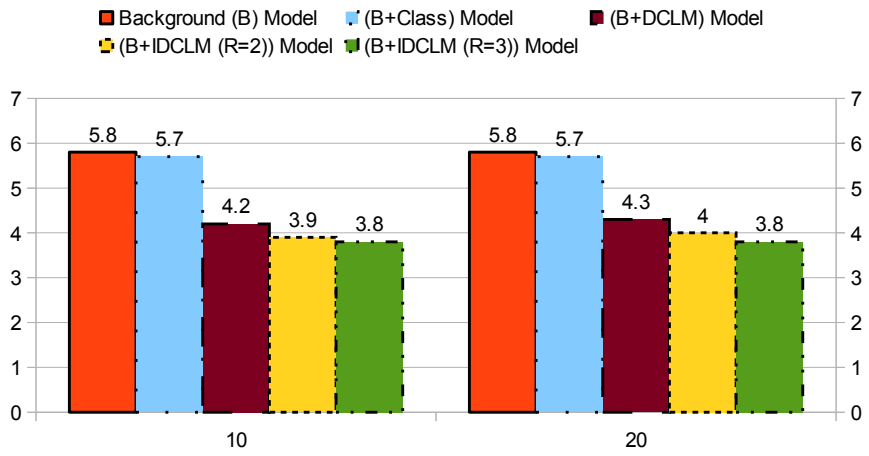

Fig. 9.3 WER results (%) for different class sizes

From Table 9.2, we can note the proposed IDCLM model outperforms the other models for all class sizes. The performance of IDCLM improves with more distances ($R = 3$).

We evaluated the WER experiments using lattice rescoring. In the first pass decoding, we used the background trigram for lattice generation. In the second pass, we applied

the interpolated model for lattice rescoring. The WER results are described in Figure 9.3. From Figure 9.3, we can note that the proposed IDCLM ($R = 3$) model yields a WER reduction of about 34.5% (5.8% to 3.8%), 33.3% (5.7% to 3.8%), and 9.5% (4.2% to 3.8%) for 10 classes and about 34.5% (5.8% to 3.8%), 33.3% (5.7% to 3.8%), and 11.6% (4.3% to 3.8%) for 20 classes over the background trigram, class trigram [16], and the DCLM [21] approaches respectively. The significance improvement in WER is done by using a match-pair-test where the misrecognized words in each test utterance are counted. The p-values are described in Table 9.3. From Table 9.3, we can note that the IDCLM ($R = 2$) is statistically

Table 9.3 p-values obtained from the match-pair test on the WER results

Language Model	10 Classes	20 Classes
B+Class & B+IDCLM (R=2)	$3.8E$-10	$4.3E$-10
B+Class & B+IDCLM (R=3)	$4.7E$-12	$4.7E$-12
B+DCLM & B+IDCLM (R=2)	0.04	0.01
B+DCLM & B+IDCLM (R=3)	0.004	0.006

significant to the class-based LM [16] and DCLM [21] at a significance level of 0.01 and 0.05 respectively. However, the IDCLM ($R = 3$) model is statistically significant to the above models at a significance level of 0.01. We have also seen that the cache DCLM model also gives the same results as DCLM [21] for a smaller number of classes [21].

9.6 Summary

In this chapter, we proposed an integration of distanced n-grams into the original DCLM model [21]. The DCLM model [21] extracted the class information from the $(n\text{-}1)$ history words through a Dirichlet distribution in calculating the n-gram probabilities. However, it does not capture the long-range semantic information from outside of the n-gram events. The proposed IDCLM overcomes the shortcomings of DCLM by incorporating the interpolated long-distance n-grams that capture the long-term word dependencies. Using the IDCLM, the class information for the histories is trained using the interpolated distanced n-grams. The IDCLM yields better results with including more distances ($R = 3$). The model probabilities are computed by weighting the component word probabilities for classes and the interpolated class information for histories. A variational Bayesian EM (VB-EM) procedure is presented to estimate the model parameters.

Chapter 10

Document-based DCLM

In this chapter, we propose a document-based Dirichlet class language model (DDCLM) for speech recognition using document-based n-gram events. In this model, the class is conditioned on the immediate history context and the document in the original DCLM model [21]. In the DCLM model, the class information was obtained from the (n-1) history words of n-gram events of a training corpus. Here, the model uses the counts of the n-grams, which are the number of appearances of the n-grams in the corpus. These counts are the sums of the n-gram counts in different documents where they could appear to describe different topics. Therefore, the n-gram counts of the corpus may not yield the proper class information for the histories. We encounter this problem in the DCLM model and propose a new DDCLM model that overcomes the above problem by finding the class information from the history context of the document-based n-gram events [46].

10.1 Introduction

In the DCLM model [21], the class information of the history words was obtained from the n-gram events of the corpus. Here, the count of an n-gram is the global count of the n-gram in the corpus i.e., the sum of counts in all the documents of the training corpus. However, the n-gram can occur in various documents to represent different topics. For example, the bi-gram *White House* can occur in a document where it describes a real estate topic. Also, it can occur in another document that describes a political topic. Therefore, the class information obtained from the history words of the n-gram events of the corpus may not be appropriate. This motivates us to introduce a document-based DCLM (DDCLM) that uses DCLM for each document. In the DDCLM model, for each document, the class information is calculated from the document-based n-gram events. The predicted word probabilities for

classes are then drawn for the corresponding document. The n-gram probabilities for the test document are computed by averaging the document-based n-gram probabilities.

10.2 Proposed DDCLM

The training of DDCLM is similar to the DCLM model except the document-based n-gram events are used in place of the global n-gram counts in the corpus. The graphical model of the DDCLM model is described in Figure 10.1. For each document d_l, the (n-1)V dimensional discrete history vector h_{d_l} is projected into a C-dimensional continuous class space using a class-dependent linear discriminant function [21]:

$$g_c(h_{d_l}) = u_{c,d_l}^T h_{d_l} \tag{10.1}$$

where u_{c,d_l}^T is the c^{th} row vector of matrix $U_{d_l} = [u_{1,d_l}, \cdots, u_{C,d_l}]$. The function $g_c(h_{d_l})$ describes the class posterior probability $P(c|h_{d_l})$.

The n-gram probability for each document is computed as [21]:

$$\begin{aligned} P_{d_l}(w_i|h_{d_l}, U_{d_l}, \beta_{d_l}) &= \sum_{c_i=1}^{C} P_{d_l}(w_i|c_i, \beta_{d_l}) \int P(\theta_{d_l}|h_{d_l}, U_{d_l}) P(c_i|\theta_{d_l}) d\theta_{d_l} \\ &= \sum_{c_i=1}^{C} P_{d_l}(w_i|c_i, \beta_{d_l}) P(c_i|h_{d_l}, U_{d_l}) \\ &= \sum_{c=1}^{C} \beta_{d_l,ic} \frac{g_c(h_{d_l})}{\sum_{j=1}^{C} g_j(h_{d_l})} \end{aligned} \tag{10.2}$$

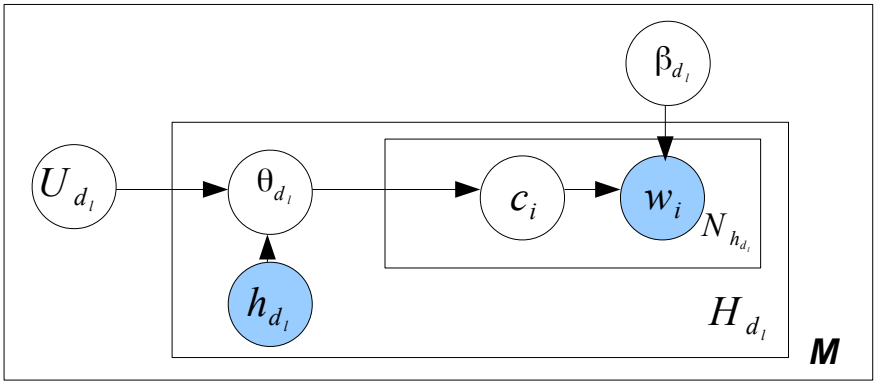

Fig. 10.1 The graphical model of the DDCLM. Shaded circles represent observed variables.

The DDCLM parameters $\{U_{d_l}, \beta_{d_l}\}$ are estimated by maximizing the marginal log likelihood of the training data that contains a set of n-gram events $S_n = \{w_{i-n+1}^{i-1}, w_i\}$:

$$\begin{aligned}\sum_{l=1}^{M} \log P(S_n|U_{d_l}, \beta_{d_l}) &= \sum_{l=1}^{M} \sum_{(w_i,h_{d_l})\in S_n} \log P_{d_l}(w_i|h_{d_l}, U_{d_l}, \beta_{d_l}) \\ &= \sum_{l=1}^{M} \sum_{h_{d_l}} \log \Bigg\{ \int P(\theta_{d_l}|h_{d_l}, U_{d_l}) \Bigg[\prod_{j=1}^{N_{h_{d_l}}} \sum_{c_j=1}^{C} P_{d_l}(w_j|c_j, \beta_{d_l}) \times \\ & P(c_j|\theta_{d_l}) \Bigg] d\theta_{d_l} \Bigg\}\end{aligned} \tag{10.3}$$

where M is the total number of training documents that comprise the corpus, $N_{h_{d_l}}$ represents the number of collected words that occur following the history h_{d_l}. In Equation 10.3, the summation is over all possible histories in training samples S_n. However, directly optimizing Equation 10.3 is intractable [21]. A variational DDCLM is used where the marginal likelihood is approximated by maximizing the lower bound of Equation 10.3. The VB-EM procedure is required since the parameter estimation involves the latent variables of $\{\theta_{d_l}, c_{h_{d_l}} = \{c_i\}_{i=1}^{N_{h_{d_l}}}\}$.

The history-dependent variational parameters $\{\hat{\gamma}_{h_{d_l}} = \{\hat{\gamma}_{h_{d_l},c}\}, \hat{\phi}_{h_{d_l}} = \{\hat{\phi}_{h_{d_l},vc}\}\}$, which correspond to the latent variables $\theta_{d_l}, c_{h_{d_l}}$, are estimated in the VB-E step as [21]:

$$\hat{\gamma}_{h_{d_l},c} = g_c(h_{d_l}) + \sum_{i=1}^{N_{h_{d_l}}} \phi_{h_{d_l},ic} \tag{10.4}$$

$$\hat{\phi}_{h_{d_l},ic} = \frac{\beta_{d_l,ic} \exp\left[\Psi(\gamma_{h_{d_l},c}) - \Psi(\sum_{j=1}^{C} \gamma_{h_{d_l},j})\right]}{\sum_{c=1}^{C} \beta_{d_l,ic} \exp\left[\Psi(\gamma_{h_{d_l},c}) - \Psi(\sum_{j=1}^{C} \gamma_{h_{d_l},j})\right]}, \tag{10.5}$$

where $\beta_{d_l,ic}$ is the probability of the i^{th} word for class c in document d_l, $\Psi(.)$ is the derivative of the log gamma function, and is known as a digamma function [21]. With the updated $\{\hat{\gamma}_{h_{d_l}}, \hat{\phi}_{h_{d_l}}\}$ in the VB-E step, the DDCLM parameters $\{U_{d_l}, \beta_{d_l}\}$ are estimated in the VB-M step as [21]:

$$\hat{\beta}_{d_l,vc} = \frac{\sum_{h_{d_l}} \sum_{i=1}^{N_{h_{d_l}}} \hat{\phi}_{h_{d_l},ic} \delta(w_v, w_i)}{\sum_{m=1}^{V} \sum_{h_{d_l}} \sum_{i=1}^{N_{h_{d_l}}} \hat{\phi}_{h_{d_l},ic} \delta(w_m, w_i)}, \tag{10.6}$$

where $\sum_{v=1}^{V} \beta_{d_l,vc}$=1 and $\delta(w_v, w_i)$ is the Kronecker delta function that equals one when vocabulary word w_v is identical to the predicted word w_i and equals zero otherwise. The

gradient ascent algorithm is used to calculate the parameters $\hat{U}_{d_l} = [\hat{u}_{1,d_l}, \cdots, \hat{u}_{C,d_l}]$ by updating the gradient $\bigtriangledown_{u_{c,d_l}}$ as [21]:

$$\begin{aligned} \bigtriangledown_{u_{c,d_l}} \leftarrow \bigtriangledown_{u_{c,d_l}} + \sum_{h_{d_l}} \Bigg[\Psi\Big(\sum_{j=1}^{C} g_j(h_{d_l}) \Big) - \Psi(g_c(h_{d_l})) \\ + \Psi(\hat{\gamma}_{h_{d_l},c}) - \Psi\Big(\sum_{j=1}^{C} \hat{\gamma}_{h_{d_l},j} \Big) \Bigg] . h_{d_l}. \end{aligned} \tag{10.7}$$

The n-gram probabilities of the test document d_t are computed as:

$$\begin{aligned} P(w_i|h_{d_t}) &= \sum_{l=1}^{M} P_{d_l}(w_i|h_{d_t}, U_{d_l}, \beta_{d_l}) P(d_l|h_{d_t}) \\ &= \sum_{l=1}^{M} P_{d_l}(w_i|h_{d_t}, U_{d_l}, \beta_{d_l}) \frac{C(h_{d_t}, d_l)}{\sum_{l=1}^{M} C(h_{d_t}, d_l)}, \end{aligned} \tag{10.8}$$

where $P_{d_l}(w_i|h_t, U_{d_l}, \beta_{d_l})$ is computed by using Equations 10.2 and $C(h_{d_t}, d_l)$ is the count of h_{d_t} in the training document d_l. However, for some n-grams of the test document, the words of the n-gram cannot be found together in any of the training documents. Their probabilities are computed as:

$$P(w_i|h_{d_t}) = \sum_{l=1}^{M} \Big(\sum_{c=1}^{C} P(w_i|c) P(c|h_{d_t}, U_{d_l}) \Big) \frac{C(h_{d_t}, d_l)}{\sum_{l=1}^{M} C(h_{d_t}, d_l)}, \tag{10.9}$$

where $P(w_i|c)$ is computed as:

$$P(w_i|c) = \sum_{l=1}^{M} P_{d_l}(w_i|c, \beta_{d_l}).P(d_l), \tag{10.10}$$

where $P(d_l) = 1/M$. The remaining zero probabilities of the obtained matrix $P(w_i|h_{d_t})$ are then computed by using back-off smoothing.

To capture the local lexical regularities, the model $P(w_i|h_{d_t})$ is then interpolated with the background (B) trigram model as:

$$P_L(w_i|h) = \lambda P_B(w_i|h) + (1-\lambda) P(w_i|h_{d_t}). \tag{10.11}$$

10.3 Comparison of DCLM and DDCLM Models

In the DCLM model, the class information for the $(n-1)$ history words is obtained by using the n-gram counts in the corpus. DCLM exploits the word distribution given the history words. DCLM performs the history clustering of the corpus. The current word is predicted from the history-dependent Dirichlet parameter, which is controlled by a matrix U and corpus-based histories h [21]. In contrast, document-based class information for the $(n-1)$ history words h_{d_l} is obtained by using the document-based n-gram events in the DDCLM model. The model performs the history clustering of the documents. The DDCLM exploits the document-based word distribution given the history words of the document-based n-grams. Both DCLM and DDCLM models are adopted to characterize the order of sequential words and to estimate the language model for both seen and unseen histories. For the DCLM model, the number of parameters $\{U,\beta\}$ increases linearly with the number of history words and is given by $(n-1)CV+CV$. For the DDCLM model, the number of parameters $\{U_{d_l},\beta_{d_l}\}$ increases linearly with the number of history words and documents and is given by $\sum_{l=1}^{M}((n-1)CV_{d_l}+CV_{d_l})$. Here, V_{d_l} is the number of words present in document d_l from the vocabulary V. The time complexities of DCLM and DDCLM are $O(HVC)$ and $O(\sum_{l=1}^{M}H_{d_l}V_{d_l}C)$ with H corpus-based histories, H_{d_l} document-based histories, V vocabulary words, V_{d_l} document-based observed words from vocabulary V, M documents that comprise the corpus, and C classes.

10.4 Experiments

10.4.1 Data and Parameters

We randomly selected 1000 documents from the '87-89 WSJ corpus [71] for training the DCLM and DDCLM models. The total number of words in the documents is 439,212. We used the 5K non-verbalized punctuation closed vocabulary from which we removed the MIT stop word list [3] and the infrequent words that occur only once in the training documents. After these removals, the total number of words in the vocabulary is 3169. We could not consider more training documents due to higher computational cost and huge memory requirements for the DDCLM model. However, trigram models give better results than the bigram models when more training data are considered. As a small amount of training data can be considered in the DDCLM model, the reliability of trigrams decreases more severely than that of bigrams and the bigrams are more robust than the trigrams [103].

For this reason, we train the DCLM and DDCLM models using the bigrams only. The models are trained by using only those bigrams that contain words from the vocabulary. To capture the local lexical regularity, the models are interpolated with a back-off trigram background model, which is trained on the '87-89 WSJ corpus using the back-off version of the Witten-Bell smoothing; the 5K non-verbalized punctuation closed vocabulary and the cutoffs 1 and 3 on the bi-gram and tri-gram counts respectively are incorporated. However, ten EM iterations in the VB-EM procedure were used. The initial values of the entries in the matrix β,β_{d_l} were set to be $1/V$ and those in U, U_{d_l} were randomly set in the range [0,1]. To update the variational parameters in the VB-E step, one iteration was used. The VB-M step was executed to update the parameters U, U_{d_l} by three iterations [21]. The interpolation weight λ is computed by optimizing on the held-out data. The experiments are evaluated on the evaluation test, which is a total of 330 test utterances from the November 1992 ARPA CSR benchmark test data for vocabularies of 5K words [71, 101].

10.4.2 Experimental Results

The models are trained for various numbers of class sizes. We trained the DCLM and DDCLM models for five times and the results are averaged. The perplexity and WER results are described in Table 10.1 and Figure 10.2. From Table 10.1, we can note the proposed

Table 10.1 Perplexity results of the models

Language Model	20 Classes	40 Classes
Background (B)	69.0	69.0
B+DCLM	61.6	61.4
B+DDCLM	59.8	59.9

DDCLM model outperforms the other models for all class sizes.

We performed the paired t-test on the perplexity results of the DCLM and the DDCLM models with a significance level of 0.01. The p-values for different class sizes are described in Table 10.2. From Table 10.2, we can note that all p-values are less than the significance level of 0.01. Therefore, the perplexity improvements of the proposed DDCLM model over the DCLM model [21] are statistically significant.

We evaluated the WER experiments using lattice rescoring. In the first pass decoding, we used the back-off trigram background language model for lattice generation. In the second pass, we applied the interpolated model for lattice rescoring. We record WERs (%) and error

Table 10.2 p-values obtained from the paired t test on the perplexity results

Language Model	20 Classes	40 Classes
B+DCLM & B+DDCLM	8.58E-07	9.24E-05

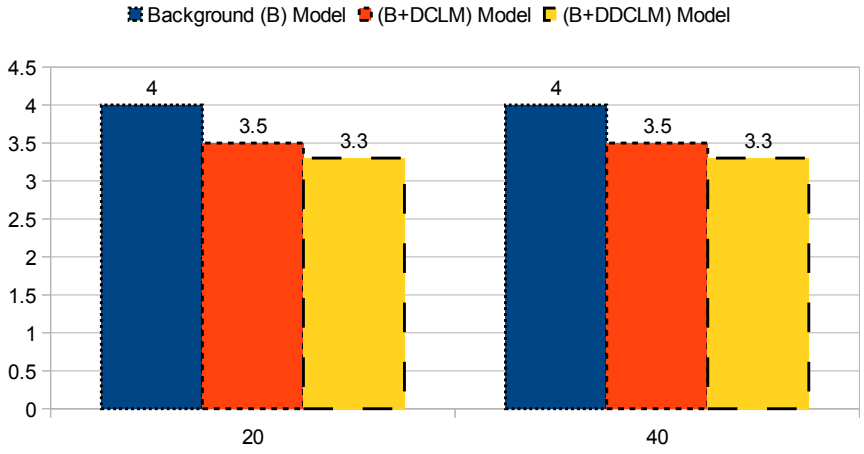

Fig. 10.2 WER results (%) for different class sizes

rate reductions (%) using different LMs for various sizes of classes. The background LM yields a WER of 4.0%. From Figure 10.2, we can note that the proposed DDCLM model yields a WER reduction of about 17.5% (4.0% to 3.3%) and 5.7% (3.5% to 3.3%) for 20 classes and about 17.5% (4.0% to 3.3%) and 5.7% (3.5% to 3.3%) for 40 classes over the background and the DCLM [21] approaches respectively.

We also performed a paired t test on the WER results for the B+DCLM and the B+DDCLM models with a significance level of 0.01. The p-values of the test are explained in Table 10.3. From Table 10.3, we can see that the p-values are smaller than the significance level of 0.01.

Table 10.3 p-values obtained from the paired t test on the WER results

Language Model	20 Classes	40 Classes
B+DCLM & B+DDCLM	0.00024	0.00092

Therefore, the WER improvements of the proposed DDCLM model are statistically significant.

10.5 Summary

In this chapter, a document-based Dirichlet class language model (DDCLM) for speech recognition using document-based n-gram counts is proposed. The class information in the DDCLM model is exploited from the $(n-1)$ history words of the document-based n-grams. This helps to extract the class information correctly for the histories of the n-gram events that appear in different documents to describe different topics whereas in the DCLM model the class information may not be appropriate as it used the global count of the n-grams in the corpus. The n-gram model of the test document is computed by averaging the document-based n-gram models. We compared our approach with the DCLM [21] model where the class information was exploited from the histories of the n-gram counts of the corpus. We have seen better perplexity and WER results using the WSJ corpus over the DCLM model [21].

Chapter 11

Conclusion and Future Work

The two components of the current ASR's system are the acoustic model (the spectral representation of sounds or words) and the language model (the representation of the grammar or syntax). To obtain a robust ASR system, they are very important. Thus, the system's overall performance can be improved by improving the LM. In this research, we have incorporated the probabilistic topic models to improve the LM. In this chapter, we describe our research into three parts in the contributions section, followed by the summary of the experimental results and the future work.

11.1 Contributions

In the first part (chapters 3, 4 and 5) of the thesis, we performed language model adaptation through mixture language models and unigram scaling [68]. Mixture models are the component models created from background corpora. The background corpus is analysed by using the LDA model. The mixture models are adapted in such a way that the adapted model can be best matched with the test environment. The contributions of this part are:

- The mixture component models are created by employing a hard-clustering method into an LDA model. We proposed a weighting approach [40] to adapt the component models. We considered an adaptation technique called unigram scaling, that forms a new adapted model [41, 49] by using a minimum discriminant information (MDI) approach [34, 68], that minimizes the Kullback-Leibler (KL) divergence between the new adapted model and the other model, subject to a constraint that the marginalized unigram distribution of the new adapted model is equal to the LSM. The LSM is the unigram probability distribution over words that are calculated using LDA-adapted

unigram models [96].

- The component models are created by using the features of the LDA model. As LDA is a bag-of-words model, each word has equal importance in determining topic mixtures. We computed the topic probabilities of the n-grams by averaging the topic probabilities for words in the n-grams and then assigned them as the count of the n-grams for different topics. We create the component models using these counts and adapt them by applying a weighting approach, where the mixture weights are created by averaging the topic probabilities for words in the development test set [42].

- We created the component models by using the document-based topic probabilities and document-based n-gram counts. The topic probabilities of the training documents are computed by averaging the topic probabilities of words seen in the documents. The topic probabilities of documents are multiplied by the document-based n-gram counts. The products are then summed-up for all the training documents. The results are used as the counts of the respective topics to create the component models. The component models are then adapted by using the topic probabilities of a development test set that are computed as above [48].

In the second part (chapters 6, 7, and 8) of the thesis, we proposed five new probabilistic topic models that are trained using the expectation-maximization (EM) algorithm. Here, we trained the model parameters by using the observed training data. A folding-in procedure [33] is then applied to compute the topic probabilities of the unseen test data. The n-gram language model of the test set is calculated using the n-gram probabilities for topics and the topic probabilities of the $(n-1)$ history words of the test data. The contributions of this part are:

- We introduced a context-based PLSA (CPLSA) model [43] to overcome the problems of a recently proposed unsmoothed bigram PLSA (UBPLSA) model [7]. Observed n-grams of the training documents are used to train the models. The unigram probabilities for topics are trained using the CPLSA model that helps to compute the correct topic probabilities of the unseen test document as the model allows one to compute all the possible n-gram probabilities of the seen history context.

- We presented a document-based CPLSA (DCPLSA) model [50] that outperforms the CPLSA model. The DCPLSA model can best describe the words that appear in different documents to represent different topics. The model trains the document-based

unigram probabilities for topics instead of corpus-based unigram probabilities for topics in the CPLSA model.

- To improve the LDLM model [20], we proposed an interpolated latent Dirichlet language model (ILDLM) [44] using the distanced n-gram counts, where the topic is drawn from the $(n-1)$ history context using the Dirichlet distribution in computing the n-gram probabilities. We computed the n-gram probabilities of the model by using the distanced word probabilities for the topics and the interpolated topic information for the histories.

- Similar to the LDLM and ILDLM approaches, we introduced an enhanced PLSA (EPLSA) and an interpolated EPLSA (IEPLSA) model in the PLSA framework. In the EPLSA model, the predicted word of observed n-gram events is drawn from a topic that is chosen from the topic distribution of the $(n$-$1)$ history words. The EPLSA model cannot capture the long-range information outside of the n-gram events. To tackle this problem, we presented an IEPLSA model that uses the distanced n-grams. We computed the n-gram probabilities of the model by using the distanced word probabilities for the topics and the interpolated topic information for the histories [45].

In the final part (chapters 9 and 10) of the thesis, we proposed two new Dirichlet class-based language models that are trained using variational Bayesian EM (VB-EM) algorithm. Here, we trained the model parameters using the observed training data. Then, the n-gram probabilities for the unseen test set are computed by using the model parameters. The contributions of this part are:

- We introduced an interpolated DCLM (IDCLM) [47] incorporating interpolated distanced n-grams. Here, the class information is exploited from $(n-1)$ history words through the Dirichlet distribution using interpolated distanced n-grams. We computed the n-gram probabilities of the model by using the distanced word probabilities for the classes and the interpolated class information for the histories.

- We presented a document-based Dirichlet class language model (DDCLM) [46] using document-based n-gram events. Here, the class is conditioned on the immediate $(n-1)$ history words and the document. The model helps to find the proper class information for the n-grams that are used to describe different classes in different documents.

11.2 Summary of the Experimental Results

In chapter 3, we proposed unsupervised LM adaptation approaches [49] using the unigram scaling technique [68] incorporating the LDA model [13] and LSM [96]. A hard-clustering approach [72] was applied to form topic sets, topic-specific LMs were adapted by applying a *n*-gram weighting method [40] to form an adapted model and then interpolated with a background model. All the above models are further modified by using the unigram scaling approach incorporating LSM. We performed experiments on the WSJ corpus using various topic sets for different test sets. The proposed unigram scaling of the interpolation of background and adapted models gives best results for topic sizes 25 and 75 for the November 1993 and November 1992 test sets respectively using the '87-89 corpus. It gives about 16.9% (8.3% to 6.9%), 13.7% (8.00% to 6.9%), 8.0% (7.5% to 6.9%), and 4.2% (7.2% to 6.9%) for the November 1993 test set using topic set 25 and about 19.6% (4.6% to 3.7%), 19.6% (4.6% to 3.7%), 15.9% (4.4% to 3.7%), and 5.1% (3.9% to 3.7%) for the November 1992 test set using topic set 75 over the background model, the unigram scaling of the background model [96], the unigram scaling of the adapted model [41], and the interpolation of the background and the adapted models [40] respectively. The data and parameters are described in section 3.4.1.

In chapter 4, we performed soft-clustering and hard-clustering assignments of background *n*-grams into different topics with counts of the fraction of the global count. The topic weights of the *n*-grams are computed by averaging two confidence measures namely: the probability of topics given words $P(t_k|w_i)$ and the probability of words given topics $P(w_i|t_k)$. The weights are then normalized, multiplied by the global count of the count, and then assigned to different topics as the count of the *n*-grams. The soft-clustering approach gives better results than hard-clustering and the confidence measure $P(t_k|w_i)$ outperforms the confidence measure $P(w_i|t_k)$ as expected. The data and parameters of the experiments are described in section 4.4.1. We performed experiments for various topic sizes (20 and 40) and the best results obtained by topic set 40. The soft clustering approach using $P(t_k|w_i)$ gives significant WER reductions of about 9.9% (8.1% to 7.3%), 7.6% (7.9% to 7.3%), and 2.7% (7.5% to 7.3%) over the background model, unigram scaling of the background model [96], and LDA *n*-gram weighting [40] approaches respectively.

In chapter 5, a novel LM adaptation approach was proposed using the document-based topic distribution and *n*-gram counts. The topic weights of the training documents are created by averaging the topic probabilities given words that are present in the documents. The topic weights of the documents are then multiplied by the document-based *n*-gram counts,

the products are summed up for all training documents, and the results are then used as the *n*-gram counts of the respective topics. We also introduce another approach called ALNCLM where the topic probabilities for documents are created by using the document-topic matrix obtained from the LDA model training. However, we performed experiments for different topic sizes (25 and 50) and the best results obtained by topic set 25. The description of the data and parameters are given in section 5.5.1. For topic set 25, the proposed method gives significant WER reductions of about 9.9% (8.1% to 7.3%), 7.6% (7.9% to 7.3%), 3.9% (7.6% to 7.3%), and 1.4% (7.4% to 7.3%) over the background (B) trigram, LDA unigram scaling [96], B+ANCLM [42] and B+ALNCLM (also proposed by us) approaches respectively.

In chapter 6, we introduced a context-based PLSA (CPLSA) model [43], which can compute the correct topic probabilities of the unseen test document and thus can compute the correct bi-gram probabilities of the test document. Furthermore, we extend the CPLSA model into document-based CPLSA (DCPLSA) model [50], where we used the document-based word probabilities for topics, as the words can appear in different documents to describe different topics. We performed experiments for different sizes of topics (20 and 40), and the best results were achieved by topic set 40. The data and parameters of the experiments are described in section 6.9.1. The proposed DCPLSA model [50] yields about about 22.0% (69.0 to 53.8), 13.1% (61.9 to 53.8), 8.3% (58.7 to 53.8) and 3.6% (55.8 to 53.8) relative improvement for perplexity and about 27.5% (4.0% to 2.9%), 17.1% (3.5% to 2.9%), 14.7% (3.4% to 2.9%) and 9.4% (3.2% to 2.9%) relative improvement for WER using the 40 topic set, over the background model, PLSA model [33], the UBPLSA [7] and the CPLSA [43] approaches respectively.

In chapter 7, to incorporate the long-range information into the LDLM [20], we introduced a interpolated LDLM (ILDLM) [44] using the interpolated long-distanced bi-grams. We performed experiments using various topic sizes (40 and 80) and the best results were achieved by topic set 80. The data and parameters for the experiments are described in section 7.5.1. The proposed ILDLM yields WER reductions of about 22.1% (7.6% to 5.92%), 20.0% (7.4% to 5.92%), and 11.6% (6.7% to 5.92%) for 80 topics, over the background model, PLSA model [33], and the LDLM [20] approaches respectively. Furthermore, we incorporated the cache unigram scaling into the LDLM and ILDLM as the cache-based models capture different information than the LDLM and ILDLM. Here, the experiments over topic set 40 give more WER reductions as there is more room (because of the smaller topic set) to add cache information. Also, the addition of cache models improves the performance of LDLM more than ILDLM as the ILDLM captures the long-range information

by using the interpolated distanced n-grams.

In chapter 8, we introduced enhanced PLSA (EPLSA) and interpolated EPLSA (IEPLSA) [45] in the PLSA framework. The training methods of the models are similar to the LDLM and ILDLM models. We performed experiments using different topic sizes (40 and 80) and achieved the best results by using topic set 80 as expected. The experimental parameters and data are explained in section 8.5.1. The proposed IEPLSA model gives relative WER improvement of about 20.4% (7.6% to 6.05%), 18.2% (7.4% to 6.05%), and 5.5% (6.4% to 6.05%) for 80 topics, over the background model, PLSA [33] and EPLSA approaches respectively. We also incorporated the cache-models into EPLSA and IEPLSA models and have seen the same characteristics as above.

In chapter 9, we proposed an interpolated DCLM (IDCLM) [47] to incorporate the long-range information into the DCLM [21] by using the long-distanced tri-grams. The model is trained by using the variational Bayesian EM (VB-EM) procedure. We conducted experiments for various sizes of classes (10 and 20) and achieved the best results by class size 20. We also performed experiments for different lengths of distance ($R = 2$ and $R = 3$). The data and parameters of the experiments of the experiments are described in section 9.5.1. The proposed IDCLM ($R = 3$) yields significant WER reductions of about 34.5% (5.8% to 3.8%), 33.3% (5.7% to 3.8%), 11.6% (4.3% to 3.8%), and 5.0% (4.0% to 3.8%) for 20 classes, over the background trigram, class trigram [16], the DCLM [21], and the IDCLM ($R = 2$) approaches respectively.

In chapter 10, we introduced a document-based DCLM [46] where the class information for the histories of the document-based bi-gram events is computed correctly. The bi-gram probabilities of the test document are computed by averaging the document-based bi-gram models. We performed experiments for class sizes of 20 and 40. The data and parameters of the experiments are described in section 10.4.1. The proposed DDCLM model yields a significant WER reduction of about 17.5% (4.0% to 3.3%) and 5.7% (3.5% to 3.3%) over the background model and the DCLM [21] approaches respectively.

11.3 Future Work

In chapter 3, the n-gram probabilities of the adapted (A) mixture model can be used as features in a maximum entropy (ME) [88] adaptation framework. In the CPLSA and DCPLSA, the performance can be improved by incorporating more training documents and higher order n-grams. In our future work, we intend to do similar approaches in the LDA framework with more training documents and higher order n-grams. We will incorporate the larger

distance-based n-grams and higher order n-grams in the ILDLM, EPLSA, and IEPLSA models to improve the performance. We will evaluate the proposed IDCLM and DDCLM approaches with neural network-based language models [12, 76, 77, 90] and exponential class-based language models [18]. For the IDCLM, we will find out a way to perform the experiments for higher numbers of classes. For the TNCLM, NTNCLM, ILDLM, EPLSA, and IEPLSA models, we will test our experiments on the evaluation test set November 1992 (330 sentences, 5353 words) ARPA CSR benchmark test data for 5K vocabularies [71, 101].

Chapter 12

Résumé en français

12.1 Introduction

La modélisation de la langue (LM) est le défi de capturer, de caractériser et d'exploiter les régularités de la langue naturelle. Autrement dit, c'est le fait d'encoder les connaissances linguistiques qui sont utiles pour les systèmes informatiques lorsqu'il s'agit de la langue humaine [28]. Elle est largement utilisée dans une variété de tâches de traitement du langage naturel telles que la reconnaissance de la parole [6, 57], la reconnaissance manuscrite [74], la traduction automatique [17], et la récupération de l'information [86]. Cependant, l'une des applications les plus passionnantes de modèles de langage est en reconnaissance automatique de la parole (ASR), où un ordinateur est utilisé pour transcrire le texte parlé en forme écrite.

12.1.1 La modélisation de la langue pour la reconnaissance vocale

Un dispositif de reconnaissance de la parole est constitué d'une combinaison de la modélisation acoustique et de la modélisation du langage. Il peut être décrit comme dans la Figure 12.1. La parole est entrée dans le système de la reconnaissance vocale sous forme des données acoustiques O. Le rôle de la reconnaissance est de trouver le mot le plus probable W' comme suit:

$$W' = \underset{W}{\operatorname{argmax}} P(W|O) \tag{12.1}$$

où $P(W|O)$ représente la probabilité que le mot W a été prononcé, étant donné que l'observation de la séquence acoustique O. La partie droite de l'équation 12.1 peut être réorganisée en se

basant sur la loi de Bayes comme suit:

$$P(W|O) = \frac{P(W)P(O|W)}{P(O)}, \tag{12.2}$$

où $P(W)$ est la probabilité que le mot W sera prononcé, $P(O|W)$ est la probabilité que la séquence acoustique O sera observée lorsque le mot W est prononcé par le locuteur, finalement, $P(O)$ est la probabilité que O sera observée. De ce fait, $P(O)$ peut être ignorée car elle n'est pas dépendante de la séquence de mots sélectionnée. Par conséquent, l'équation 12.1 peut être réécrite comme suit :

$$W' = \underset{W}{\operatorname{argmax}} P(W|O) = \underset{W}{\operatorname{argmax}} P(W)P(O|W), \tag{12.3}$$

où $P(O|W)$ est déterminée par la modélisation acoustique et $P(W)$ est déterminée par la partie de la modélisation du langage du système de la reconnaissance vocale.

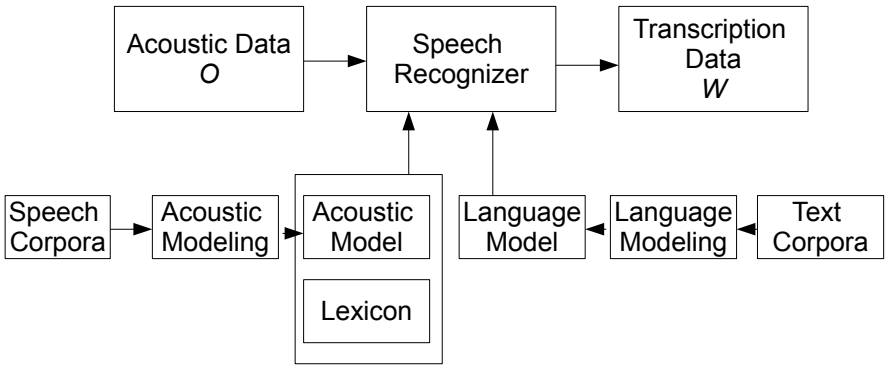

Fig. 12.1 Système de reconnaissance vocale

12.1.2 Outils expérimentaux et les bases de données

Nous avons évalué les approches de l'adaptation LM en utilisant le corpus du Wall Street Journal (WSJ) [71]. La boîte à outils SRILM [94] est utilisée pour générer les Modèles de langage. De plus, la boîte à outils HTK [105] est utilisée pour le calcul du WER. Le modèle acoustique présenté dans [98] est utilisé dans nos expériences. Le modèle acoustique est entrainé en utilisant toute les données d'apprentissage de WSJ et de TIMIT [32], les 40 ensembles téléphoniques de la Dictionnaire CMU [2], environ 10000 états liés, de 32 gaussiennes par état modélisant la parole ainsi que 64 gaussiennes par état modélisant le silence. Les formes d'ondes acoustiques sont paramétrisées dans un vecteur de caractéristiques de

dimension 39 composé de 12 coefficients cepstraux plus le coefficient cepstral d'ordre zéro, coefficients delta et double delta. Le tout est normalisé en utilisant la soustraction cepstrale moyenne $(MFCC_{0-D-A-Z})$. Nous avons évalué les modèles de mots croisés. Les valeurs de pénalité d'insertion du mot, la largeur du faisceau, et le facteur d'échelle du modèle de la langage sont respectivement -4.0, 350.0 et 15.0 [98]. La base de données du développement **si_dt_05**.**odd** (248 phrases, 4074 mots) et la base de données de tests sont respectivement, les données de test Nov' 92 et Nov' 93 de novembre 1992 (330 phrases, 5353 mots) et novembre 1993 (215 phrases, 3849 mots), données de test de référence ARPA CSR pour vocabulaires 5K [71, 101].

12.1.3 Contributions

Les deux composantes du système de l'ASR actuel, à savoir, le modèle acoustique (la représentation spectrale de sons ou de mots) et le modèle de langage (la représentation de la grammaire ou de la syntaxe), sont très importantes pour obtenir un système ASR robuste. Ainsi, les performances globales du système peuvent être améliorées en améliorant le Modèle de langage (LM). Dans cette recherche, nous proposons l'intégration des modèles probabilistes pour améliorer le Modele de langage (LM). Nous décrivons la recherche en trois parties dans les trois sections suivants.

12.2 Adaptation LM en utilisant LDA

Dans la première partie de la thèse, nous avons effectué une adaptation de modèle de langage par le mélange de modèles de langage et l'échelle unigramme [68]. Les modèles de mélange sont les modèles de composantes créés à partir d'un corpus d'apprentissage. Ce corpus est analysé en utilisant le modèle d'allocation latente de Dirichlet (LDA) [13]. Les modèles de mélange sont adaptés de manière à ce que le modèle produit soit adapté à l'environnement de test. Les contributions de cette partie de la thèse sont :

12.2.1 Adaptation LM à base de LDA en utilisant la Sémantique latente marginale (LSM)

Dans cette section, nous présentons les approches non supervisées d'adaptation du modèle de langue (LM) utilisant l'Allocation latente de Dirichlet (LDA) et LSM. La LSM est la distribution de probabilité unigramme sur les mots qui sont calculés en utilisant des mod-

èles unigramme LDA adaptés. Le modèle LDA est utilisé pour extraire des informations dépendantes du sujet à partir d'un corpus d'apprentissage de manière non supervisée. Le modèle LDA fournit une matrice du document-sujet qui décrit le nombre de mots attribués à des sujets pour les documents. Une méthode du regroupement à décision stricte utilise la matrice document-sujet du modèle LDA pour former des sujets. Un modèle adapté est créé en utilisant une combinaison pondérée des modèles *n*-grammes du sujet. L'interpolation du modèle d'arrière-plan et le modèle adapté donne une nouvelle amélioration. Nous modifions les modèles ci-dessus à l'aide de la LSM. La LSM est utilisée pour former un nouveau modèle adapté. Il utilise une approche d'adaptation, à base de l'information minimale discriminante (MDI), appelée l'échelle unigramme, qui minimise la distance entre le nouveau modèle adapté et l'autre modèle [49].

Les données et paramètres

Le corpus '87-89 WSJ est utilisé pour entrainer le modèle de base de tri-gramme et les modèles tri-gramme du sujet. Les modèles sont entrainés en utilisant la version $back\text{-}off$ de lissage de $Witten\text{-}Bell$. Les modèles de langage sont à vocabulaire fermée, c'est à dire, les modèles sont générés en utilisant les comptes de *n*-grammes sans tenir compte des *n*-grammes avec des mots inconnus. Pour réduire le coût du calcul, nous avons intégré les seuils 1 et 3 respectivement sur les comptes de bi-grammes et de tri-grammes. Le LDA et les modèles de langage sont entrainés en utilisant le WSJ 20K de vocabulaire fermée à ponctuation non-verbalisée. Nous définissons respectivement α et β pour l'analyse de LDA $50/K$ et 0,01 [37, 53]. Les poids d'interpolation ϕ et λ sont calculés selon la *compute-best-mix* programme de la boîte à outils de SRILM. Ils sont optimisés sur l'ensemble du développement du test. Les sémantiques latentes marginales (LSM) sont créés par une transcription automatique. La transcription automatique est le résultat de reconnaissance obtenu après une première passe de décodage des données d'évaluation. Les résultats des expériences sont rapportés sur l'ensemble du test.

Adaptation LM non supervisée en utilisant la pondération N-gram

Les perplexités sur les ensembles du test de Novembre 1993 et Novembre 1992 pour les différentes tailles de corpus sont respectivement décrites dans les tableaux 12.1 and 12.2.

Les résultats tels que mesurés par le WER des expériences sur les différents ensembles du test pour les différentes tailles de corpus sont respectivement rapportés dans les tableaux 12.3 et 12.4.

Table 12.1 Résultats tels que mesurés par la perplexité des modèles de langage trigramme utilisant la pondération n-gramme sur les Caractéristiques de test Novembre 1993

Language Model	Topic 25		Topic 50		Topic 75	
	'87	'87-89	'87	'87-89	'87	'87-89
Background (B)	101.7	88.9	101.7	88.9	101.7	88.9
Adapted (A) Model	**98.8**	**87.9**	105.3	91.7	107.5	89.6
B+A Model	82.1	73.5	81.5	**73.0**	**81.2**	73.4

Table 12.2 Résultats tels que mesurés par la perplexité des modèles de langage tri-gramme utilisant la pondération *n*-gramme sur les données de test Novembre 1992

Language Model	Topic 25		Topic 50		Topic 75	
	'87	'87-89	'87	'87-89	'87	'87-89
Background (B)	85.4	71.0	85.4	71.0	85.4	71.0
Adapted (A) Model	89.4	78.3	95.6	82.1	100.4	78.8
B+A Model	72.2	62.1	71.6	**61.6**	**71.5**	61.9

Table 12.3 Résultats WER (%) des modèles de langage à l'aide de pondération tri-gramme sur les données de test Novembre 1993

Language Model	Topic 25		Topic 50		Topic 75	
	'87	'87-89	'87	'87-89	'87	'87-89
Background (B)	9.2	8.3	9.2	8.3	9.2	8.3
Adapted (A) Model	8.4	**7.4**	8.4	7.7	**8.3**	7.9
B+A Model	**7.6**	**7.2**	7.8	7.3	**7.6**	7.5

Table 12.4 Résultats tels que mesurés par le WER (%) des modèles de langage à l'aide de pondération tri-gramme sur les données de test Novembre 1992

Language Model	Topic 25		Topic 50		Topic 75	
	'87	'87-89	'87	'87-89	'87	'87-89
Background (B)	4.8	4.6	4.8	4.6	4.8	4.6
Adapted (A) Model	5.3	4.5	5.6	4.7	5.5	**4.3**
B+A Model	4.2	4.0	4.2	**3.9**	**4.1**	**3.9**

Nouveau modèle adapté à l'aide de la LSM

Les résultats tels que mesurés par la perplexité des expériences utilisant les ensembles du test Novembre 1993 et Novembre 1992 pour les différentes tailles de corpus sont respectivement rapportés dans les tableaux 12.5 and 12.6.

Table 12.5 Résultats tels que mesurés par la perplexité sur les données du test Novembre 1993 à l'aide de modèles de langage tri-gramme obtenus en utilisant la LSM

Language Model	Topic 25		Topic 50		Topic 75	
	'87	'87-89	'87	'87-89	'87	'87-89
Background (B)	101.7	88.9	101.7	88.9	101.7	88.9
Adaptation of (B) model	98.8	85.9	97.8	85.7	**97.2**	**84.8**
Adaptation of A model	**97.9**	**86.7**	103.5	89.9	105.1	87.0
Adaptation of (B+A) model	80.7	72.0	79.3	71.0	**78.5**	**70.8**

Table 12.6 Résultats tels que mesurés par la perplexité sur les données du test Novembre 1992, utilisant des modèles de langage tri-gramme obtenus en utilisant la LSM

Language Model	Topic 25		Topic 50		Topic 75	
	'87	'87-89	'87	'87-89	'87	'87-89
Background (B)	85.4	71.0	85.4	71.0	85.4	71.0
Adaptation of (B) model	84.2	70.2	83.9	69.9	**83.6**	**69.7**
Adaptation of A model	88.5	78.2	94.0	81.4	98.3	**77.3**
Adaptation of (B+A) model	71.4	61.6	70.4	60.8	**70.1**	**60.7**

Les résultats tels que mesurés par le WER des expériences sur les différents ensembles du test pour les différentes tailles de corpus sont rapportés dans les tableaux 12.7 and 12.8.

Signification statistique et analyse des erreurs

L'amélioration significative du WER est réalisée en utilisant un test de paire assortie où les mots mal reconnus dans chaque énoncé du test sont comptés. Les valeurs p de l'unigramme proposé du modèle (B + A) sont respectivement mesurées par rapport au modèle d'arrière-plan, l'unigramme du modèle d'arrière-plan [96], l'unigramme du modèle adapté [42] et l'interpolation du modèle d'arrière-plan et les modèles adaptés [40]. Pour l'ensemble du test Novembre 1993 utilisant la taille du sujet 25 et le corpus '87-89, les valeurs de p sont

Table 12.7 Résultats tels que mesurés par le WER (%) sur les données du test Novembre 1993 à l'aide des modèles de langage tri-gramme obtenus en utilisant LSM.

Language Model	Topic 25		Topic 50		Topic 75	
	'87	'87-89	'87	'87-89	'87	'87-89
Background (B)	9.2	8.3	9.2	8.3	9.2	8.3
Adaptation of (B) model	9.2	**8.0**	**9.0**	**8.0**	9.1	8.1
Adaptation of A model	**8.4**	**7.5**	8.5	7.7	8.6	7.9
Adaptation of (B+A) model	**7.6**	**6.9**	7.7	7.2	**7.6**	7.2

Table 12.8 Résultats tels que mesurés par le WER (%) sur les données du test Novembre 1992, utilisant des modèles de langage tri-gramme obtenus en utilisant LSM.

Language Model	Topic 25		Topic 50		Topic 75	
	'87	'87-89	'87	'87-89	'87	'87-89
Background (B)	4.8	4.6	4.8	4.6	4.8	4.6
Adaptation of (B) model	4.8	4.6	4.8	4.7	4.9	4.6
Adaptation of A model	5.3	4.5	5.6	4.6	5.6	**4.4**
Adaptation of (B+A) model	4.2	3.8	4.2	3.8	**4.1**	**3.7**

4,0E-9, 0,00081, 7,4E-8, et 0,00175. Pour l'ensemble de test Novembre 1992, utilisant des tailles du sujet 75 et le corpus '87-89, les valeurs de p sont 4,9E-6, 0,0071, 8,3E-7, et 0,00989. À un niveau de signification de 0,01, l'approche proposée est nettement meilleure que les autres modèles.

Les tableaux 12.9 et 12.10 sont utilisés pour présenter les résultats de l'ASR pour la suppression (D), la substitution (S), et l'insertion (I) des erreurs, ainsi que l'exactitude (Corr) et la précision (Acc) des modèles trigrammes de langage. En observant ces tableaux, nous pouvons noter que l'unigramme proposé du modèle B + A réduit tous les types d'erreurs et améliore l'exactitude et la précision relative à l'arrière-plan et à d'autres modèles [40, 42, 96]. L'utilisation de l'approche proposée, la suppression et les erreurs d'insertion ne changent pas beaucoup par rapport à l'arrière-plan et à d'autres modèles. Par conséquent, les erreurs de substitution jouent un rôle important pour améliorer la performance, à savoir, plusieurs mots peuvent être reconnus avec précision en utilisant la méthode proposée à l'arrière-plan et à d'autres modèles. Nous pouvons également noter que l'amélioration du modèle A peut aider à réduire les erreurs existantes dans l'approche actuelle.

Table 12.9 Résultats tels que mesurés par l'ASR pour les erreurs de la suppression (D), la substitution (S), et l'insertion (I), et aussi pour l'exactitude (Corr) et la précision (Acc), des modèles de langage tri-gramme obtenus en utilisant l'ensemble du test Novembre 1993 avec la taille du sujet 25 et le corpus 87-89.

Language Model	D	S	I	Corr	Acc
Background (B)	0.010	0.064	0.009	0.926	0.917
Adaptation of B model using LSM	0.010	0.061	0.008	0.929	0.920
Adaptation of A model using LSM	0.010	0.058	0.006	0.931	0.925
B+A model	0.010	0.055	0.007	0.935	0.928
Adaptation of B+A model using LSM	0.009	0.054	0.006	0.937	0.931

Table 12.10 Résultats tels que mesurés par l'ASR pour les erreurs de la suppression (D), la substitution (S), et l'insertion (I), et aussi pour l'exactitude (Corr) et la précision (Acc), des modèles de langage tri-gramme obtenus en utilisant l'ensemble du test Novembre 1992 avec la taille du sujet 75 et le corpus 87-89.

Language Model	D	S	I	Corr	Acc
Background (B)	0.003	0.033	0.010	0.965	0.954
Adaptation of B model using LSM	0.003	0.033	0.010	0.964	0.954
Adaptation of A model using LSM	0.003	0.030	0.011	0.967	0.956
B+A model	0.002	0.027	0.009	0.970	0.961
Adaptation of B+A model using LSM	0.002	0.026	0.009	0.973	0.963

12.2.2 Sujet n-gramme compte LM (TNCLM)

Dans cette section, nous présentons des nouvelles approches d'adaptation du modèle de langue (LM) utilisant le modèle d'allocation latente de Dirichlet (LDA) [13]. Les N-grammes observés dans l'ensemble d'apprentissage sont affectés aux sujets en utilisant des méthodes de regroupement strict et souple. Lors d'un regroupement souple, chaque n-gramme est affecté aux sujets de telle sorte que le nombre total des n-grammes pour tous les sujets est égal au nombre global de n-grammes dans l'ensemble d'apprentissage. Dans ce cas, les poids du sujet normalisés de la n-gramme sont multipliés par le nombre global des n-grammes utilisés pour former le n-gramme du sujet pour les sujets respectifs. Dans le regroupement strict, chaque n-gramme est affecté à un seul sujet avec la fraction maximale du nombre global des n-grammes pour le sujet correspondant. Dans ce cas, le sujet est sélectionné à l'aide du poids maximum pour le sujet du n-gramme. Les comptes de n-grammes des LMs sont créées en utilisant les n-grammes respectifs des sujets et adaptés

en utilisant les poids des sujets d'un ensemble développement. Nous calculons la moyenne des mesures de confiance : la probabilité d'un mot étant donné le sujet $P(w_i|t_k)$ et la probabilité du sujet étant donné le mot $P(t_k|w_i)$. La moyenne est calculée sur les mots dans les n-grammes et dans l'ensemble du développement pour former respectivement les poids du sujet des n-grammes et de l'ensemble de développement [42].

Données et paramètres

Les données et les autres paramètres sont exactement les mêmes que dans la section 12.2.1, sauf le LDA et les modèles de langage sont entrainés en utilisant le vocabulaire fermé non verbalisé de ponctuation WSJ 5K.

Résultats expérimentaux

Nous avons testé nos approches proposées pour les tailles du sujet 20 et 40. Les résultats de perplexité des modèles de ANCLM sont présentés dans le tableau 12.11 et le tableau 12.12, où les comptes des n-grammes du sujet pour les modèles de TNCLM sont respectivement générés en utilisant les mesures de confiance $P(w_i|t_k)$ et $P(t_k|w_i)$.

Table 12.11 Résultats de la perplexité des données de test Novembre 1993 en utilisant le modèle de ANCLM généré en utilisant la mesure de confiance $P(w_i|t_k)$ pour les regroupements durs et mous de n-grammes de fond.

Language Model	20 Topics	40 Topics
Background (B)	83.4	83.4
ANCLM (Hard)	277.3	378.2
ANCLM (Soft)	101.2	109.2
B+ANCLM (Hard)	72.6	72.5
B+ANCLM (Soft)	71.5	70.8

Les résultats WER des expériences sont décrites respectivement dans les figures 12.2 et 12.3 pour la mesures de confiance $P(w_i|t_k)$ et $P(t_k|w_i)$.

12.2.3 Nouveau compte de n-gramme du sujet du LM

Dans cette section, nous présentons un nouveau compte de n-grammes du sujet du modèle de langage (NTNCLM) obtenu à l'aide des probabilités du sujet des documents d'apprentissage et des comptes de n-grammes basés sur des documents. Les probabilités du sujet pour les

Table 12.12 Résultats de la perplexité des données de test Novembre 1993 en utilisant le modèle de ANCLM généré en utilisant la mesure de confiance $P(t_k|w_i)$ pour les regroupements durs et mous de n-grammes de fond.

Language Model	20 Topics	40 Topics
Background (B)	83.4	83.4
ANCLM (Hard)	227.9	287.25
ANCLM (Soft)	92.9	101.45
B+ANCLM (Hard)	71.65	71.5
B+ANCLM (Soft)	70.15	69.9

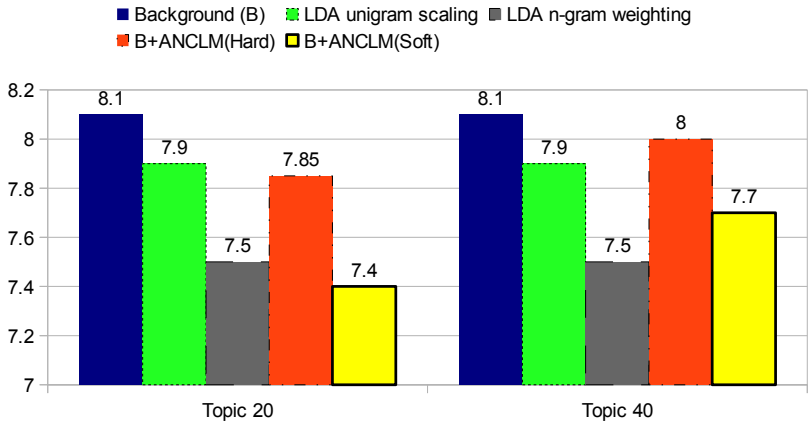

Fig. 12.2 Résultats WER (%) sur les données de test Novembre 1993 pour le modèle de ANCLM développé en utilisant la mesure de la confiance $P(w_i|t_k)$

documents sont calculées en prenant la moyenne des probabilités du sujet des mots observés dans les documents. Les probabilités du sujet des documents sont multipliées par les comptes des n-grammes basés sur des documents. Les produits sont ensuite additionné pour tous les documents d'apprentissage. Les résultats sont utilisés comme les comptes de leurs sujets respectifs pour créer les NTNCLMs. Les NTNCLMs sont adaptés en utilisant les probabilités du sujet d'un ensemble de développement qui sont calculés comme ci-dessus. Nous comparons notre approche avec une autre récemment proposée, nommée TNCLM [42], où les informations extérieurs à distance des événements du n-gramme ne sont pas rencontrées. Notre approche donne une perplexité et un taux d'erreur de mots (WER) significativement réduits par rapport à l'autre approche lorsque testé sur le corpus de Wall Street Journal (WSJ) [48].

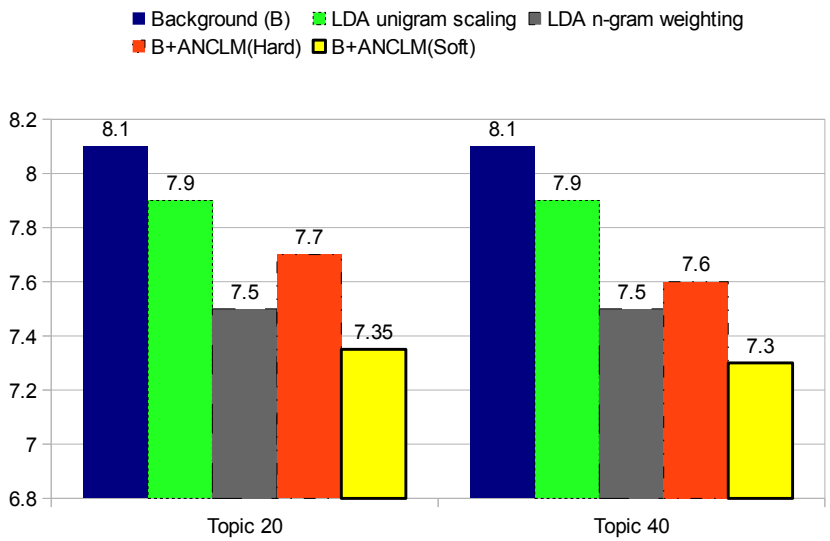

Fig. 12.3 Résultats WER (%) sur les données de test Novembre 1993 pour le modèle de ANCLM développé en utilisant la mesure de la confiance $P(t_k|w_i)$

Données et paramètres

Les données et les autres paramètres sont exactement les mêmes que dans la section 12.2.1, sauf le LDA et les modèles de langage sont entrainés en utilisant le vocabulaire fermé non verbalisé de ponctuation WSJ 5K.

Résultats expérimentaux

Nous avons testé nos approches proposées pour différentes tailles du sujet. Les résultats en termes de la perplexité et du WER sont respectivement présentées par le tableau 12.13 et par la graphique figure 12.4.

L'amélioration significative du WER en utilisant B+ANNCLM est obtenue par l'utilisation d'un de paire assortie du test où les mots mal reconnus dans chaque énoncé du test sont comptés. Nous obtenons les valeurs p de 0,03 et 0,02 par rapport à B+ANCLM [41] respectivement pour les tailles du sujet de 25 et 50. Au niveau 0,05, notre modèle proposé B+ANNCLM a surpassé le modèle B+ANCLM [41].

12.3 Cinq nouveaux modèles probabilistes du sujet

Dans ce section, nous proposons cinq nouveaux modèles probabilistes du sujet qui sont entrainés à l'aide d'algorithme d'espérance-maximisation (EM). Dans ce cas, nous entrainons

Table 12.13 Résultats tels que mesurés par la perplexité obtenus sur les données de test Novembre 1993 en utilisant le trigramme du langage modèles.

Language Model	25 Topics	50 Topics
Background (B)	83.4	83.4
ANCLM	105.5	134.0
ALNCLM	86.5	111.4
ANNCLM	86.2	110.4
B+ANCLM	75.3	75.6
B+ALNCLM	74.6	74.8
B+ANNCLM	74.7	74.9

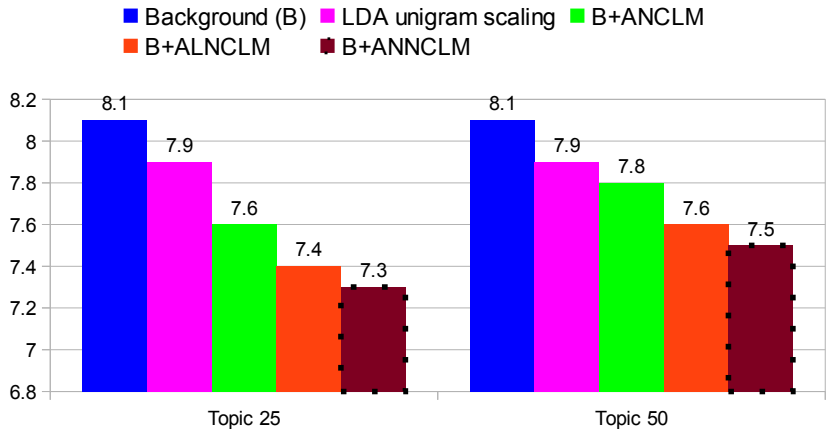

Fig. 12.4 Résultats tels que mesurés par le WER (%) obtenus sur les données de test Novembre 1993 à l'aide des modèles de langage

les paramètres du modèle à l'aide des données d'apprentissage observées. Une procédure de pliage est ensuite appliquée pour calculer les probabilités du sujet des données du test non disponibles. Les modèles n-gramme de langage de l'ensemble du test sont calculés en utilisant les probabilités n-grammes pour les sujets et les probabilités des sujets de l'historique des (n-1) mots de données du test. Les contributions de cette partie sont :

12.3.1 PLSA LM basée sur le contexte

Dans cette section, nous proposons un nouveau modèle de langue à base du contexte pour la reconnaissance vocale nommée l'analyse probabiliste sémantique latente à base du contexte (CPLSA). Dans ce modèle, le sujet est conditionné sur le cadre de l'historique im-

médiat et le document dans le modèle PLSA original [33]. Ce permet le calcul de toutes les probabilités de bi-grammes possibles du l'historique du contexte vu à l'aide du modèle. Il calcule correctement la probabilité du sujet d'un document invisible pour chaque historique du contexte présent, dans le document. Nous comparons notre approche avec un autre récemment proposé, nommée le modèle bi-gramme non lissée PLSA (UBPLSA) [7] où seules les probabilités bi-grammes observées sont calculées, ce qui provoque le calcul de probabilité du sujet incorrect pour l'historique présente du contexte du document non vu. Le modèle de CPLSA proposé nécessite beaucoup moins de temps du calcul et d'espace mémoire que pour le modèle bi-gramme non lissée PLSA [43]. Dans le modèle de CPLSA, les probabilités de mots pour les sujets sont calculées par la somme des événements des bi-grammes dans tous les documents. Toutefois, dans différents documents les mots peuvent apparaître pour décrire les différents sujets. Pour résoudre ce problème, nous introduisons également un Modèle CPLSA à base du documents (DCPLSA) [50]. Ce modèle est similaire au modèle de CPLSA sauf que la probabilité du mot est conditionnée à la fois au sujet et au document. Cependant, il nécessite une plus grande taille de mémoire et du temps du calcul que le modèle de CPLSA.

Données et paramètres

Nous avons choisi au hasard 500 documents du corpus '87-89 WSJ [71] pour l'entrainement de la UBPLSA, la CPLSA et les modèles DCPLSA. Le nombre total de mots dans les documents est de 224,995. Nous avons utilisé 5K de vocabulaire fermé de ponctuation non verbalisé à partir de laquelle nous avons éliminé la liste de mots éliminatoires de MIT [3] et les mots peu fréquents qui se produisent qu'une seule fois dans les documents d'apprentissage. Après ces éliminations, le nombre total de mots du vocabulaire est de 2628 mots. Nous ne pouvions pas envisager plus de documents d'entrainement en raison du coût de calcul plus élevé et besoin énorme en termes de mémoire pour le modèle de UBPLSA [7] et les modèles DCPLSA. Pour la même raison, nous entrainons seulement les modèles bi-gramme UBPLSA, CPLSA et DCPLSA. De plus, nous avons utilisé le même nombre de documents pour les modèles PLSA et CPLSA pour une vraie comparaison. Pour capturer la régularité lexicale locale, les modèles du sujet sont interpolés avec un modèle trigramme *back-off* d'arrière plan. Le modèle trigramme de d'arrière plan est entrainé à partir du corpus le 87-89 WSJ en utilisant la version de *back-off* de lissage de la *Witten-Bell*; 5K de vocabulaire fermé de ponctuation non verbalisé et les seuils de 1 et 3 sont respectivement incorporés sur les comptes de bi-grammes et de tri-gramme. Les coefficients de pondération d'interpolation sont calculés en optimisant sur la lieu de départ des données. Les expéri-

ences sont évaluées sur l'ensemble d'évaluation, qui est un total de 330 énoncés d'essai des données de référence de Novembre 1992 (ARPA CSR) pour les vocabulaires de 5K mots [71, 101].

Résultats expérimentaux

Nous avons testé les approches LM ci-dessus pour différentes tailles de sujets. Nous avons effectué les expériences cinq fois, et les résultats sont moyennés. Les résultats en termes de la perplexité et du WER sont respectivement présentés par le tableau 12.14 et par la graphique figure 12.5.

Table 12.14 Résultats de la perplexité des modèles sujets

Language Model	20 Topics	40 Topics
Background (B)	69.0	69.0
B+PLSA	62.0	61.9
B+UBPLSA	59.0	58.7
B+CPLSA	57.5	55.8
B+DCPLSA	55.5	53.8

Nous avons effectué le test t apparié sur les résultats de la perplexité des modèles ci-dessus avec un un niveau de signification de 0,01. Les valeurs de p pour différentes tailles du sujet sont décrites dans le tableau 12.15.

Table 12.15 p-valeurs obtenues à partir de la t test apparié sur les résultats de la perplexité

Language Model	20 Sujets	40 Sujets
B+UBPLSA and B+CPLSA	$6.0E$-11	$2.8E$-14
B+CPLSA and B+DCPLSA	$6.5E$-12	$3.1E$-13

D'après le tableau 12.15, on peut noter que toutes les valeurs de p sont inférieures à la limite de signification de 0,01. Par conséquent, les améliorations de la perplexité du modèle DCPLSA proposé sur le modèle CPLSA [43] sont statistiquement significatifs. En outre, le modèle de CPLSA [43] est statistiquement meilleur que le modèle de UBPLSA [7].

Nous avons également effectué un test t apparié sur les résultats du WER pour les modèles interpolés avec un niveau de signification de 0,01. Les valeurs p du test sont représentées dans le tableau 12.16.

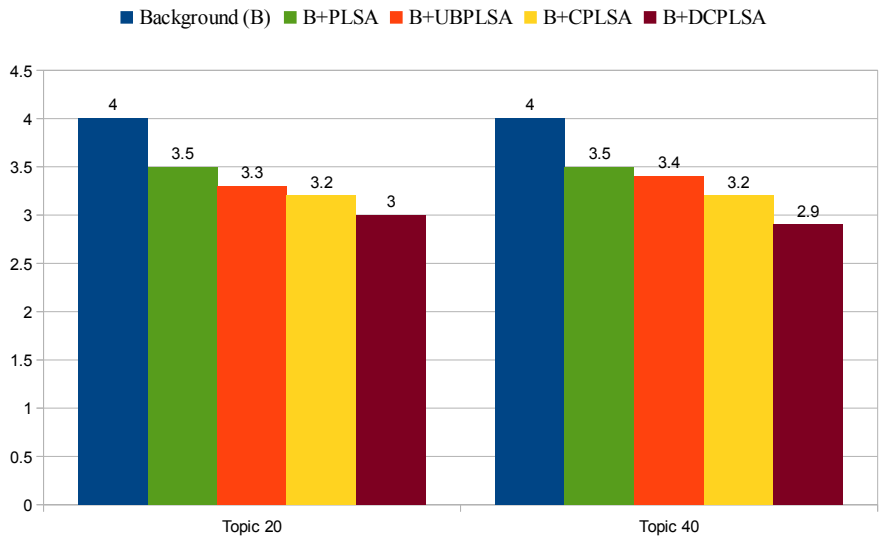

Fig. 12.5 Résultats tels que mesurés par le WER (%) des modèles de langue

Table 12.16 p-valeurs obtenues à partir de la t test apparié sur les résultats WER

Language Model	20 Sujets	40 Sujets
B+UBPLSA and B+CPLSA	4.7E-06	9.3E-06
B+CPLSA and B+DCPLSA	6.9E-06	1.5E-07

D'après le tableau 12.16, nous pouvons voir que les valeurs de p sont inférieures à la limite de signification de 0,01. Par conséquent, les améliorations en terme du WER du modèle proposé DCPLSA sont statistiquement significatives.

12.3.2 La LDLM interpolée

Dans cette section, nous proposons une approche de la modélisation de la langue (LM) utilisant les n-grammes éloignés interpolées dans un modèle de langage de Dirichlet latente (de LDLM) [20] pour la reconnaissance vocale. Le LDLM relaxe l'hypothèse d'ensemble de mots et l'estimation du sujet du document d'Allocation latente de Dirichlet (LDA). Il utilise par défaut les n-grammes d'arrière plan où l'information du sujet est extraite des (n-1) mots d'historique à travers la distribution de Dirichlet dans le calcul des probabilités de n-gramme. Le modèle ne tient pas compte des informations à longue distance à partir de l'extérieur des événements des n-grammes qui peuvent améliorer les performances de la modélisation du langage. Nous présentons une interpolation LDLM (ILDLM) en utilisant différents n-grammes éloignés. Dans ce cas, l'information du sujet est exploitée de (n-1)

mots d'historique à travers la distribution de Dirichlet en utilisant l'interpolation éloignée des *n*-grammes. Les probabilités des n-grammes du modèle sont calculées en utilisant les probabilités des mots éloignés pour les sujets et les informations des sujets interpolées pour les historiques. De plus, nous intégrons un LM à base de caches, qui modélise les mots reproduisant, par l'ajustement une unigramme pour adapter les modèles de LDLM et ILDLM qui modélisent les mots d'actualité [44].

Données et paramètres

Le corpus '87-89 WSJ est utilisé pour entrainer des modèles de langage. Les modèles sont entrainés à l'aide le WSJ 5K de vocabulaire fermé de ponctuation non verbalisé. Un modèle de base de tri-gramme est entrainé en utilisant le lissage *Kneser-Ney* modifié en intégrant les seuils des comptes 1 et 3 respectivement pour les bi-grammes et tri-gramme. Afin de réduire les exigences de calcul et de mémoire à l'aide de MATLAB, nous avons entrainé seulement les modèles bi-gramme LDLM et ILDLM. Pour les modèles de ILDLM, nous avons considéré les bigrammes pour D=1,2. Le paramètre d'apprentissage η est fixé à 0,01. Une taille de mémoire de cache fixe de $F = 400$ est utilisée pour le LM à base de cache. Les poids d'interpolation λ_D, γ et ρ sont calculés en utilisant le programme de *calcul-meilleur-mix* de la boîte à outils SRILM. Ils sont optimisés sur l'ensemble de développement. Les expériences sont évaluées sur l'ensemble d'évaluation, qui est un total de 215 énoncés d'essai des données de référence de Novembre 1993 (ARPA CSR) pour les vocabulaires de 5K mots [71, 101].

Résultats expérimentaux

Nous avons testé les approches proposées pour différentes tailles de sujets. Les résultats en termes de la perplexité et du WER sont respectivement présentés par le tableau 12.17 et par la graphique figure 12.6.

12.3.3 La PLSA améliorée et EPLSA l'interpolée

Dans cette section, nous présentons de la modélisation de la langue (LM) des approches utilisant les *n*-grammes d'arrière plan et les *n*-grammes interpolées éloignés pour la reconnaissance de la parole en utilisant une dérivation d'analyse probabiliste sémantique latente renforcée (EPLSA). PLSA est un modèle d'ensemble de mots qui exploite les informations de sujets au niveau du document, ce qui est incompatible avec la modélisation de la langue en reconnaissance de la parole. Nous considérons la séquence de mots dans la modélisation

Table 12.17 Résultats tels que mesurés par la perplexité des modèles de langage

Language Model	40 Topics	80 Topics
Background (B)	70.3	70.3
PLSA	517.8	514.8
LDLM	251.6	153.6
ILDLM	86.9	65.25
B*PLSA	66.6	66.5
B+LDLM	65.1	62.5
B+ILDLM	53.6	52.7
(B+LDLM)*CACHE	59.9	57.5
(B+ILDLM)*CACHE	49.3	48.5

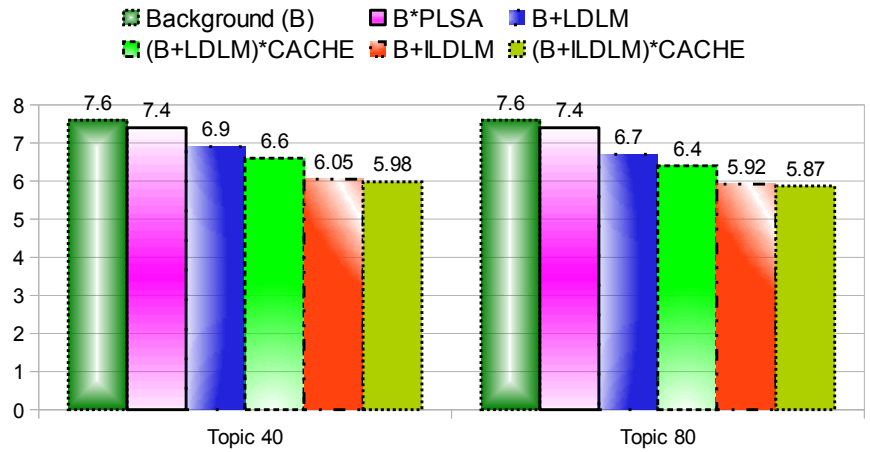

Fig. 12.6 Résultats tels que mesurés par le WER (%) des modèles de langue

de EPLSA. Dans ce cas, le mot prédit d'un événement de n-grammes est tiré d'un sujet qui est choisi de la distribution des sujets des (n-1) mots d'historique. Le modèle de l'EPLSA ne peut pas capturer les informations de sujets à longue distance à partir de l'extérieur de l'événement n-grammes. Les n-grammes éloignés sont incorporés dans la forme interpolée (IEPLSA) pour couvrir l'information à long terme. Un modèle LM à base de caches qui modélise les mots reproduits est également intégré via une unigramme aux modèles EPLSA et IEPLSA, qui modélise les mots dépendant du sujet [45].

Donées et paramètres

Le corpus '87-89 WSJ est utilisé pour entrainer des modèles de langage. Les modèles sont entrainés à l'aide le WSJ 5K de vocabulaire fermé de ponctuation non verbalisé. Un modèle

de base de tri-gramme est entrainé en utilisant le lissage de *Kneser-Ney* modifié intégrant respectivement les seuils des comptes : 1 et 3 des bi-grammes et tri-grammes. Pour réduire les exigences en termes de mémoire et du calcul en utilisant MATLAB, nous avons entrainé seulement les modèles bi-grammes EPLSA et IEPLSA. Pour les modèles IEPLSA, nous avons considéré les bigrammes pour D=1,2. Une taille de cache fixe de F=400 est utilisée pour le LM à base de caches. Les poids d'interpolation λ_D, γ et α sont calculés selon le programme *calcul-meilleur-mix* de la boîte à outils de SRILM. Ils sont optimisés sur l'ensemble du développement. Les expériences sont évaluées sur l'ensemble d'évaluation, qui est un total de 215 énoncés d'essai des données de référence de Novembre 1993 (ARPA CSR) pour les vocabulaires de 5K mots [71, 101].

Résultats expérimentaux

Nous avons testé l'approche proposée pour différentes tailles de sujets. Les résultats en termes de la perplexité et du WER sont respectivement présentées par le tableau 12.18 et par la graphique figure 12.7.

Table 12.18 Résultats tels que mesurés par la perplexité des modèles de langage

Language Model	40 Topics	80 Topics
Background (B)	70.3	70.3
PLSA	517.8	514.8
EPLSA	192.9	123.3
IEPLSA	101.2	93.0
B*PLSA	66.6	66.5
B+EPLSA	62.9	59.7
B+IEPLSA	55.1	55.1
(B+EPLSA)*CACHE	58.0	55.1
(B+IEPLSA)*CACHE	50.7	50.7

12.4 Deux nouvelles approches de DCLM

Dans la dernière partie de la thèse, nous proposons deux nouveaux modèles de langage à base des classes de Dirichlet qui sont entrainés à l'aide d'algorithme EM bayésienne variationnelle (VB-EM). Dans ce cas, nous entrainons les paramètres du modèle en utilisant les données d'apprentissage observés. Ensuite, les probabilités n-grammes pour l'ensemble du

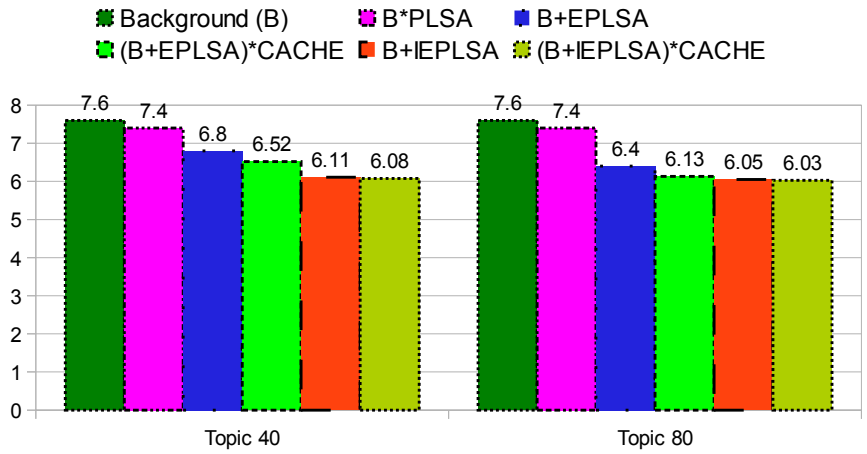

Fig. 12.7 Résultats tels que mesurés par le WER (%) des modèles de langue

test non vus sont calculées à l'aide des paramètres du modèle. Les contributions de cette partie sont les suivants :

12.4.1 La DCLM interpolée

Dans cette section, nous proposons une approche de la modélisation de la langue (LM) en utilisant les *n*-grammes éloignés interpolées dans un modèle de langue de la classe de Dirichlet (DCLM) [21], pour la reconnaissance vocale. Le DCLM relaxe l'hypothèse de l'ensemble-de-mots et l'extraction du document sujet d'Allocation latente de Dirichlet (LDA). La variable latente de DCLM reflète les informations de classe d'un événement *n*-gramme plutôt que le sujet en LDA. Le modèle DCLM utilise les *n*-grammes d'arrière plan par défaut où l'information de la classe est extraite des (*n*-1) mots d'historique via une distribution de Dirichlet durant le calcul des probabilités de *n*-grammes. Le modèle ne tient pas compte des informations à long-terme provenant de l'extérieur de la fenêtre de *n*-gramme qui peut améliorer les performances de la modélisation du langage. Nous présentons une DCLM interpolée (IDCLM) en utilisant différents *n*-grams éloignés. Dans ce cas, l'information de la classe est exploitée à partir de (*n*-1) mots d'historique à travers la distribution de Dirichlet à l'aide des *n*-grammes éloignés interpolées. Une procédure bayésienne variationnelle est introduite pour estimer les paramètres de IDCLM [47].

Données et paramètres

Les approches de LM sont évaluées en utilisant le corpus de Wall Street Journal (WSJ) [71]. Le corpus 87-89 WSJ est utilisé pour entrainer les modèles de langage. Les trigrammes

de base sont entrainées à l'aide la version de *back-off* du lissage de *Witten-Bell* de 5K de vocabulaire fermé de ponctuation non verbalisé. Nous entrainons le modèle trigramme de IDCLM en utilisant R=2 et R=3. Dix itérations EM dans la procédure VB-EM ont été utilisées. Les valeurs initiales des entrées dans la matrice β, β_D ont été fixées à $1/V$ et celles de U, U_I ont été sélectionnées au hasard dans l'intervalle [0,1]. Pour mettre à jour les paramètres variationnels dans l'étape VB-E, une seule itération a été utilisée. L'étape VB-M était exécutée pour mettre à jour les paramètres U, U_I par trois itérations [21]. Pour capturer la régularité lexicale locale, des trigrammes de différentes méthodes sont interpolés avec les trigrammes d'arrière plan. Les poids d'interpolation λ_D et μ sont calculés en optimisant les données détenus selon la métrique de la perplexité. Les expériences sont évaluées sur l'ensemble d'évaluation, qui contient un total de 330 énoncés d'essai des données de Novembre 1992 ARPA CSR test de référence pour des vocabulaires de 5K mots [71, 101].

Résultats expérimentaux

En raison des exigences plus élevées de la mémoire et du temps d'exécution pour l'entrainement du modèle IDCLM, nous avons entrainé les modèles DCLM et IDCLM pour des classes de 10 et 20. Les résultats en termes de la perplexité et du WER sont respectivement présentés par le tableau 12.19 et par la graphique figure 12.8.

Table 12.19 Résultats tels que mesurés par la perplexité des modèles

Language Model	10 Classes	20 Classes
Background (B)	109.4	109.4
B+Class	106.65	107.0
B+DCLM	100.2	100.45
B+IDCLM (L=2)	98.0	97.9
B+IDCLM (L=3)	95.6	95.4

L'amélioration importante en termes du WER est obtenue en utilisant un test de paires assortis où les mots mal reconnus, dans chaque énoncé du test, sont comptés. Les valeurs p sont présentées dans le tableau 12.20.

D'après le tableau 12.20, on peut noter que le IDCLM (R=2) est statistiquement significatif respectivement par rapport au LM à base de classes [16] et au DCLM [21] à un niveau de signification de 0,01 et 0,05. Cependant, le modèle à IDCLM (R = 3) est statistiquement significatif aux modèles ci-dessus à un niveau de signification de 0,01.

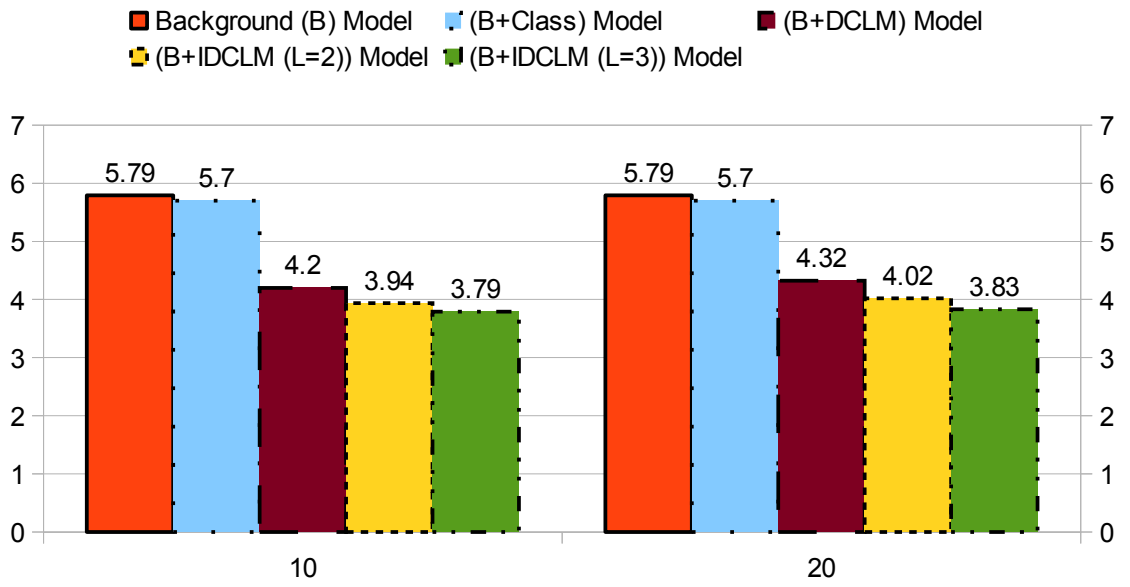

Fig. 12.8 Résultats tels que mesurés par le WER (%) pour différentes tailles des classes

Table 12.20 Valeurs p obtenues à partir des essais des paires-identifiées sur des résultats tels que mesurés par le WER

Language Model	10 Classes	20 Classes
B+Class & B+IDCLM (L=2)	$3.8E$-10	$4.3E$-10
B+Class & B+IDCLM (L=3)	$4.7E$-12	$4.7E$-12
B+DCLM & B+IDCLM (L=2)	0.04	0.01
B+DCLM & B+IDCLM (L=3)	0.004	0.006

12.4.2 DCLM à base du document

Dans cette section, nous proposons un modèle de la langue à base de documents et des classes de Dirichlet (DDCLM) pour la reconnaissance de la parole en utilisant des événements n-grammes à base de documents. Dans ce modèle, la classe est conditionnée sur le l'historique immédiat du context et le document dans le modèle DCLM original [21]. Dans le modèle DCLM, l'information de la classe a été obtenue à partir des (n-1) mots d'historique des événements des n-grammes d'un corpus d'apprentissage. Dans ce cas, le modèle utilise le nombre de n-grammes, qui sont le nombre d'apparitions des n-grammes dans le corpus. Ces chiffres correspondent à la somme de chiffres des n-grammes dans les différents documents où ils pourraient apparaître pour décrire les différents sujets. Par conséquent, les chiffres de n-grammes du corpus peuvent ne pas donner l'information de

la classe appropriée pour les historiques. Nous rencontrons ce problème dans le modèle DCLM et proposons un nouveau modèle DDCLM qui résout le problème ci-dessus en trouvant l'information du contexte historique des événements de *n*-grammes à base de documents [46].

Données et paramètres

Nous avons choisi au hasard 1000 documents des corpus '87-89 WSJ [71] pour l'entrainement des modèles DCLM et DDCLM. Le nombre total de mots dans les documents est 439,212. Nous avons utilisé le 5K de vocabulaire fermé de ponctuation non verbalisé à partir de laquelle nous avons supprimé la liste de MIT de mots à ne pas utiliser [3] et les mots peu fréquents qui se produisent qu'une seule fois dans les documents d'entrainement. Après ces arrangements, le nombre total de mots dans le vocabulaire est de 3169 mots. Nous ne pouvions pas envisager d'utiliser plus de documents d'entrainement en raison du coût de calcul plus élevé et l'énorme taille de la mémoire requise pour le modèle de DDCLM. Cependant, les modèles de trigrammes donnent de meilleurs résultats par rapport aux modèles de bi-grammes lorsque plus de données d'entrainement sont pris en compte. Comme seulement une petite quantité de données d'entrainement peut être pris en compte dans le modèle de DDCLM, la fiabilité des trigrammes diminue plus sévèrement que celui des bi-grammes et les bi-grammes sont plus robustes que les trigrammes [103]. Pour cette raison, nous entrainons les modèles DCLM et DDCLM en utilisant uniquement les bi-grammes. Les modèles sont entrainés en utilisant uniquement les bi-grammes qui contiennent des mots du vocabulaire. Pour capturer la régularité lexicale locale, les modèles sont interpolés avec un trigramme $back\text{-}off$ du modèle d'arrière plan, qui est entrainé sur le corpus '87-89 WSJ en utilisant la version $back\text{-}off$ du lissage de $Witten\text{-}Bell$; 5K de vocabulaire fermé de ponctuation non verbalisé et les seuils des comptes 1 et 3 sont respectivement incorporés sur les bi-grammes et les tri-grammes. Cependant, dix itérations EM dans la procédure VB-EM ont été utilisées. Les valeurs initiales des entrées de la matrice β, β_{d_l} ont été réglés pour être $1/V$ et ceux de U, U_{d_l} ont été sélectionnées au hasard dans l'intervalle [0,1]. Pour mettre à jour les paramètres variationnels dans l'étape VB-E, une seule itération a été utilisée. Trois itérations de l'étape VB-M ont été exécutées pour mettre à jour les paramètres U, U_{d_l} [21]. Le poids d'interpolation λ est calculé en optimisant les données détenues. Les expériences sont évaluées sur l'ensemble d'évaluation, qui est un total de 330 énoncés d'essai des données de référence de Novembre 1992 (ARPA CSR) pour les vocabulaires de 5K mots [71, 101].

Résultats expérimentaux

Les modèles sont entrainés pour différents tailles des classes. Nous avons entrainé cinq fois les modèles DCLM et DDCLM, et les résultats sont moyennés. Les résultats en terme de la perplexité et du WER sont présentés dans le tableau 12.21 et la figure 12.9. D'après le

Table 12.21 Résultats de la perplexité des modèles

Language Model	20 Classes	40 Classes
Background (B)	69.0	69.0
B+DCLM	61.6	61.4
B+DDCLM	59.8	59.9

tableau 12.21, nous pouvons noter que le modèle de DDCLM proposé surpasse les autres modèles pour toutes les tailles des classes.

Nous avons réalisé le test t apparié sur les résultats de la perplexité du DCLM et les modèles de DDCLM avec un niveau de signification de 0,01. Les valeurs de p pour la taille des différentes classes sont présentées dans le tableau 12.22.

Table 12.22 p-valeurs obtenues à partir de la t test apparié sur les résultats de la perplexité

Language Model	20 Classes	40 Classes
B+DCLM & B+DDCLM	$8.58E$-07	$9.24E$-05

D'après le tableau 12.22, on peut noter que tous les p-valeurs sont inférieures à la limite de signification de 0,01. Par conséquent, les améliorations de la perplexité du modèle proposé sur DDCLM et le modèle DCLM [21] sont statistiquement significatives.

Nous avons également effectué un test t apparié sur les résultats WER pour les modèles B+DCLM et B+DDCLM avec un niveau de signification de 0,01. Les valeurs p du test sont présentées dans le tableau 12.23.

Table 12.23 p-valeurs obtenues à partir de la t test apparié sur les résultats WER

Language Model	20 Classes	40 Classes
B+DCLM & B+DDCLM	0.00024	0.00092

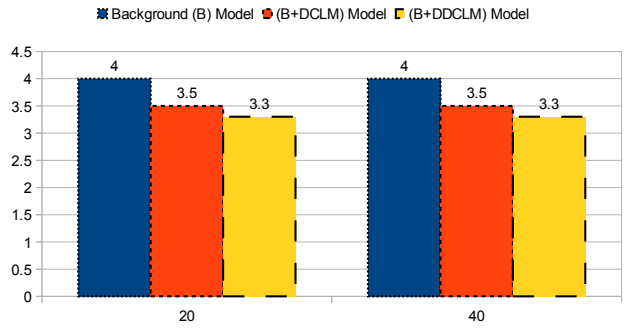

Fig. 12.9 Résultats WER (%) pour la taille des classes différentes

D'après le tableau 12.23, nous pouvons voir que les valeurs de p sont inférieures à la limite de signification de 0,01. Par conséquent, les améliorations de WER du modèle proposé DDCLM sont statistiquement significatives.

12.5 Conclusion

Le LM est une partie très importante de l'ASR, sans laquelle, l'amélioration des performances du système ASR est impossible. Dans cette recherche, nous avons développé des méthodes d'adaptation LM pour améliorer les performances des systèmes ASR. Nous avons intégré le modèle de sujet LDA et introduit de nouveaux algorithmes de modélisation du sujet pour proposer les techniques d'adaptation LM.

12.5.1 Principales contributions de la thèse

Nous décrivons les principales contributions de cette recherche comme suit :

- Nous avons créé des modèles de composants de mélange en employant une méthode du regroupement stricte dans le modèle de LDA. Nous avons proposé une méthode de pondération pour adapter les modèles de composantes. Nous considérons une technique d'adaptation appelée unigramme échelle, qui forme un nouveau modèle adapté en utilisant l'approche d'information discriminante minimal (MDI) [34, 68], qui minimise la divergence de Kullback-Leibler (KL), entre le nouveau modèle adapté et l'autre modèle, avec la contrainte que la distribution marginalisée d'unigramme de nouveau modèle adapté est égale à la LSM. La LSM est la distribution de probabilité d'unigrammes sur des mots qui sont calculés à l'aide des modèles unigramme LDA adaptés [96]. Pour plus de détails, voir [49].

- Nous avons utilisé les caractéristiques du modèle LDA pour créer les modèles de composants de mélange. Comme LDA est un modèle d'ensemble-de-mots, chaque mot a une importance égale dans la détermination de mélanges des sujets. Nous avons calculé les probabilités de sujet des *n*-grammes en faisant la moyenne ses probabilités de sujets pour les mots dans les *n*-grammes et les assigner en tant que le nombre de comptes des *n*-grammes pour les différents sujets. Nous avons créé les modèles de composants en utilisant ces comptes et les adapter en appliquant une approche de pondération, où les poids de mélange sont obtenus par la moyenne des probabilités des sujets pour les mots de l'ensemble de développement. Pour plus de détails, voir [42].

- Nous avons créé les modèles de composants en utilisant les probabilités des sujets à base du documents et les comptes des *n*-grammes à base du document. Les probabilités de sujet des documents d'entrainement sont calculées en faisant la moyenne des probabilités de sujets des mots vus dans les documents. Les probabilités de sujet des documents sont multipliées par les comptes des *n*-grammes à base du document. Ces produits sont ensuite additionnés pour tous les documents d'apprentissage. Les résultats sont utilisés comme les comptes de leurs sujets respectifs pour créer les modèles de composants. Les modèles de composants sont ensuite adaptés en utilisant les probabilités de sujet d'ensemble de développement qui sont calculés comme expliqué ci-dessus. Pour plus de détails, voir [48].

- Nous avons mis en place un modèle PLSA basée sur le contexte (CPLSA) afin de résoudre les problèmes d'un modèle récemment proposé, à savoir, la bigramme PLSA non lissée (UBPLSA) [7]. Les *n*-grammes observés des documents d'apprentissage sont utilisés pour entrainer les modèles. Les probabilités des uni-grammes pour les sujets sont entrainés à l'aide du modèle CPLSA qui permet de calculer les bonnes probabilités, du sujet du documents non vus du test, que le modèle permet de calculer toutes les probabilités possibles des *n*-grammes du contexte historique déjà vue. Pour plus de détails, voir [43].

- Nous avons présenté un modèle CPLSA basée sur des documents (DCPLSA) [50] qui surpasse le modèle de CPLSA. Le modèle DCPLSA peut mieux décrire les mots qui apparaissent dans différents documents pour représenter les différents sujets. Le modèle entraine les probabilités des uni-grammes à base de documents pour des sujets au lieu des probabilités des uni-grammes à base du corpus pour les sujets dans le modèle de CPLSA.

- Pour améliorer le modèle de LDLM [20], nous avons proposé un modèle de langage à base d'interpolation latente de Dirichlet (ILDLM) en utilisant les comptes de *n*-grammes éloignés, où le sujet est tiré du (*n*-1) mots de contexte historique en utilisant la distribution de Dirichlet dans le calcul des probabilités des *n*-grammes. Nous avons calculé les probabilités des *n*-grammes du modèle en utilisant les probabilités des mots éloignés pour les sujets et l'information interpolée du sujet pour les historiques. Voir [44] pour plus de détails.

- Comme pour les approches LDLM et ILDLM, nous avons introduit une modèle PLSA amélioré (EPLSA) et un EPLSA interpolé (IEPLSA) dans le cadre du PLSA. Dans le modèle de EPLSA, le mot prédit des événements des *n*-grammes observées est tiré à partir d'un sujet qui est choisi parmi la répartition d'un sujet des (*n*-1) mots d'historique. Le modèle EPLSA ne peut pas capturer les informations à long-terme à l'extérieur des événements de *n*-grammes. Afin de remédier à ce problème, nous avons présenté un modèle IEPLSA qui utilise les probabilités des mots éloignés pour les sujets et les informations du sujet interpolées pour l'historique [45].

- Nous avons mis en place un DCLM interpolé (IDCLM) incorporant des *n*-grammes interpolés éloignés. Dans ce cas, l'information de classe est exploitée à partir de (*n*-1) mots d'historique par la distribution de Dirichlet utilisant les *n*-grammes interpolés éloignés. Nous avons calculé les probabilités des *n*-grammes du modèle en utilisant les probabilités des mots éloignés pour les classes et les informations de classe interpolées pour les historiques. Les détails peuvent être trouvés dans [47].

- Nous avons présenté un modèle de langue à base du documents et de classe de Dirichlet (DDCLM) [46] en utilisant des événements *n*-grammes basés sur des documents. Dans ce cas, la classe est conditionnée sur l'historique immédiat des (*n*-1) mots et des documents. Le modèle aide à trouver les bonnes informations de la classe pour les *n*-grammes qui sont utilisées pour décrire les différentes catégories dans les différents documents.

References

[1] URL http://www.speech.sri.com/people/anand/771/html/node33.html.

[2] The carnegie mellon university (CMU) pronunciation dictionary, 2013. URL http://www.speech.cs.cmu.edu/cgi-bin/cmudict.

[3] MIT stop word list, 2014. URL http://jmlr.org/papers/volume5/lewis04a/a11-smart-stop-list/english.stop.

[4] G. Adda, M. Jardino, and J.L. Gauvain. Language modeling for broadcast news transcription. In *Proceedings of EUROSPEECH*, volume 4, pages 1759–1762, 1999.

[5] R. B.-Yates and B. R.-Neto. *Modern Information Retrieval*. ACM Press, New York, 1999.

[6] L. R. Bahl, F. Jelinek, and R. L. Mercer. *Readings in Speech Recognition*, chapter A Maximum likelihood approach to continuous speech recognition, pages 308–319. Morgan Kaufmann Publishers Inc., 1990.

[7] M. Bahrani and H. Sameti. A new bigram PLSA language model for speech recognition. *Eurasip Journal on Signal Processing*, pages 1–8, 2010.

[8] N. Bassiou and C. Kotropoulos. Word clustering PLSA enhanced with long distance bigrams. In *Proceedings of International Conference on Pattern Recognition*, pages 4226–4229, 2010.

[9] T. C. Bell, J. G. Cleary, and I. H. Witten. *Text Compression*. Prentice Hall, N. J., 1990.

[10] J. R. Bellegarda. Exploiting latent semantic information in statistical language modeling. *IEEE Transactions on Speech and Audio Processing*, 88 (8):1279–1296, 2000.

[11] J. R. Bellegarda. Statistical language model adaptation: review and perspective. *Speech Communications*, 42:93–108, 2004.

[12] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155, 2003.

[13] D. M. Blei, A. Y. Ng., and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[14] D. M. Blei, T. L. Griffiths, M. Steyvers, and J. B. Tenenbaum. Integrating topics and syntax. In *Advances in Neural Information Processing Systems*, volume 17, pages 537–544, 2004.

[15] S. Broman and M. Kurimo. Methods for combining language models in speech recognition. In *Proceedings of INTERSPEECH*, pages 1317–1320, 2005.

[16] P. Brown, V. D. Pietra, P. D. Souza, J. Lai, and R. Mercer. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479, 1992.

[17] P. F. Brown, J. C., S. A. D. Pietra, V. J. D. Pietra, F. Jelinek, J. D. Lafferty, R. L. Mercer, and P. S. Roossin. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85, 1990.

[18] S. Chen. *Performance prediction for exponential language models*. Technical Report RC 24671, IBM Research, 2008.

[19] S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. *Computer Speech and Language*, 13:359–394, 1999.

[20] J-T. chien and C-H. Chueh. Latent Dirichlet language model for speech recognition. In *Proceedings of the IEEE SLT Workshop*, pages 201–204, 2008.

[21] J-T. Chien and C-H. Chueh. Dirichlet class language models for speech recognition. *IEEE Transactions on Audio, Speech and Language Processing*, 19 (3):482–495, 2011.

[22] C.-H. Chueh and J.-T. Chien. Nonstationary latent Dirichlet allocation for speech recognition. In *Proceedings of INTERSPEECH*, pages 372–375, 2009.

[23] P. R. Clarkson. *Adaptation of Statistical Language Models for Automatic Speech Recognition*. PhD thesis, Cambridge University Engineering Department, 1999.

[24] P. R. Clarkson and A. J. Robinson. Language model adaptation using mixtures and an exponentially decaying cache. In *Proceedings of ICASSP*, pages 799–802, 1997.

[25] K. H. Davies, R. Biddulph, and S. Balashek. Automatic speech recognition of spoken digits. *Acoustical Society America*, 24(6):637–642, 1952.

[26] S. Deerwester, S. Dumais, T. Landauer, G. Furnas, and R. Harsman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41 (6):391–407, 1990.

[27] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.

[28] L. Deng and D. O'Shaughnessy. *Speech Processing: A Dynamic and Optimization-Oriented Approach*. Marcel Dekker Inc., 2003.

[29] P. G. Donnelly, F.J. Smith, E. Silica, and J. Ming. Language modeling with hierarchical domains. In *Proceedings of EUROSPEECH*, volume 4, pages 1575–1578, 1999.

[30] M. Federico and R. D. Mori. *Langauge Model Adaptation*. Springer-Verlag, New York, 1999.

[31] J. D. Ferguson. *Hidden Markov Analysis: An Introduction*. Princeton, NJ, 1980.

[32] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, D. S. Pallett, N. L. Dahlgren, and V. Zue. Timit acoustic-phonetic continuous speech corpus. Linguistic Data Consortium. 1993.

[33] D. Gildea and T. Hofmann. Topic-based language models using EM. In *Proceedings of EUROSPEECH*, pages 2167–2170, 1999.

[34] S. A. D. P. Gildea, V. J. D. Pietra, R. L. Mercer, and S. Roukos. Adaptive language modeling using minimum discriminant estimation. In *Proceedings of ICASSP*, pages 1633–1636, 1992.

[35] W. R. Gilks and S. Richardson. *Markov Chain Monte Carlo in Practice*. Chapman and Hall, Suffolk, 1996.

[36] I. J. Good. The population frequencies of species and the estimation of population parameters. *Biometrika*, 40(3 and 4):237–264, 1953.

[37] T. L. Griffiths and M. Steyvers. Finding scientific topics. *National Academy of Science*, 101 (1):5228–5235, 2004.

[38] R. H.-Umbach and H. Ney. Improvements in time-synchronous beam search for 10000-word continuous speech recognition. In *Proceedings of IEEE Trans Speech and Audio Processing*, volume 2, pages 353–356, 1994.

[39] M. A. Haidar and D. O'Shaughnessy. Novel weighting scheme for unsupervised language model adaptation using latent Dirichlet allocation. In *Proceedings of INTERSPEECH*, pages 2438–2441, 2010.

[40] M. A. Haidar and D. O'Shaughnessy. Unsupervised language model adaptation using n-gram weighting. In *Proceedings of CCECE*, pages 857–860, 2011.

[41] M. A. Haidar and D. O'Shaughnessy. LDA-based LM adaptation using latent semantic marginals and minimum discrimination information. In *Proceedings of EUSIPCO*, pages 2040–2044, 2012.

[42] M. A. Haidar and D. O'Shaughnessy. Topic n-gram count LM adaptation for speech recognition. In *Proceedings of IEEE SLT Workshop*, pages 165–169, 2012.

[43] M. A. Haidar and D. O'Shaughnessy. Comparison of a bigram PLSA and a novel context-based PLSA language model for speech recognition. In *Proceedings of ICASSP*, pages 8440–8444, 2013.

[44] M. A. Haidar and D. O'Shaughnessy. Fitting long-range information using interpolated distanced n-grams and cache models into a latent dirichlet language model for speech recognition. In *Proceedings of INTERSPEECH*, pages 2678–2682, 2013.

[45] M. A. Haidar and D. O'Shaughnessy. PLSA enhanced with a long-distance bigram language model for speech recognition. In *Proceedings of EUSIPCO*, 2013.

[46] M. A. Haidar and D. O'Shaughnessy. Document-based Dirichlet class language model for speech recognition using document-based n-gram events. In *Proceedings of IEEE SLT Workshop*, 2014.

[47] M. A. Haidar and D. O'Shaughnessy. Interpolated Dirichlet class language model for speech recognition using long-distance n-grams. In *Proceedings of COLING*, 2014.

[48] M. A. Haidar and D. O'Shaughnessy. Novel topic n-gram count LM incorporating document-based topic distributions and n-gram counts. In *Proceedings of EUSIPCO*, 2014.

[49] M. A. Haidar and D. O'Shaughnessy. Unsupervised language model adaptation using LDA-based mixture models and latent semantic marginals. *Computer Speech and Language*, 29(1):20–31, 2015.

[50] M. A. Haidar and D. O'Shaughnessy. Document-specific context PLSA language model for speech recognition. *Submitted to ICASSP*, 2015.

[51] A. Heidel and L.-S. Lee. Robust topic inference for latent semantic language model adaptation. In *Proceedings of IEEE ASRU Workshop*, pages 177–182, 2007.

[52] A. Heidel, H. Chang, and L. Lee. Language model adaptation using latent Dirichlet allocation and an efficient topic inference algorithm. In *Proceedings of INTERSPEECH*, pages 2361–2364, 2007.

[53] G. Heinrich. *Parameter estimation for text analysis*. Technical report, Fraunhofer IGD, 2009.

[54] T. Hofmann. Probabilistic latent semantic analysis. In *Proceedings of the Fifteenth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-99)*, pages 289–296, San Francisco, CA, 1999. Morgan Kaufmann.

[55] B.-J. Hsu. *Language Modeling for Limited Data Domains*. PhD thesis, Department of Electrical Engineering and Computer Science, MIT, USA, 2009.

[56] B.-J. Hsu and J. Glass. Style and topic language model adaptation using HMM-LDA. In *Proceedings of EMNLP*, pages 373–381, 2006.

[57] X. Huang, A. Acero, and H.-W. Hon. *Spoken Language Processing: A Guide to Theory, System and Algorithm Development*. Prentice-Hall Inc., 2001.

[58] R. Iyer and M. Ostendorf. Modeling long distance dependence in language: topic mixtures vs dynamic cache models. In *Proceedings of ICSLP*, pages 236–239, 1996.

[59] R. Iyer, M. Ostendorf, and J.R. Rohlicek. Language modeling with sentence level mixtures. In *Proceedings of ARPA Speech and Natural Language Workshop*, pages 82–86, 1994.

[60] F. Jelinek. A fast sequential decoding algorithm using a stack. *IBM Journal of Research and Development*, 13:675–685, 1969.

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

[62] F. Jelinek, L. R. Bahl, and R. L. Mercer. Design of a linguistic statistical decoder for the recognition of continuous speech. *IEEE Transactions on Information Theory*, 21(3):250–256, 1975.

[63] D. Jurafsky and J. H. Martin. *Speech and Language Processing*. Prentice Hall, 2 edition, 2009.

[64] S. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustic, Speech and Signal Processing*, 35(3):400–401, 1987.

[65] P. Kenny. A* admissible heuristics for rapid lexical access. In *Proceedings of ICASSP*, pages 689–692, 1991.

[66] R. Kneser and H. Ney. Improved backing-off for m-gram language modeling. In *Proceedings of ICASSP*, pages 181–184, 1995.

[67] R. Kneser and V. Steinbiss. On the dynamic adaptation of stochastic language models. In *Proceedings of ICASSP*, volume 2, pages 586–589, 1993.

[68] R. Kneser, J. Peters, and D. Klakow. Language model adaptation using dynamic marginals. In *Proceedings of EUROSPEECH*, pages 1971–1974, 1997.

[69] R. Kuhn and R. D. Mori. A cache-based natural language model for speech recognition. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, 12 (6): 570–583, 1990.

[70] G. J. Lidstone. Note on the general case of the bayes-laplace formula for inductive or a posteriori probabilities. *Transactions of the Faculty of Actuaries*, 8:182–192, 1920.

[71] *CSR-II (WSJ1) Complete*. Linguistic Data Consortium, Philadelphia, 1994.

[72] F. Liu and Y. Liu. Unsupervised language model adaptation incorporating named entity information. In *Proceedings of ACL*, pages 672–679, 2007.

[73] F. Liu and Y. Liu. Unsupervised language model adaptation via topic modeling based on named entity hypothesis. In *Proceedings of ICASSP*, pages 4921–4924, 2008.

[74] G. Maltese, P. Bravetti, H. Crepy, B. J. Grainger, M. Herzog, and F. Palou. Combining word and class-based language models: A comparative study in several languages using automatic and manual word-clustering techniques. In *Proceedings of EUROSPEECH*, pages 21–24, 2001.

[75] A. A. Markov. An example of statistical investigation in the text of 'eugene onyegin' illustrating coupling of tests in chains. In *Proceedings of the Academy of Science*, pages 153–162, 1913.

[76] T. Mikolov, M. Karafiat, L. Burget, J. H. Cernocky, and S. Khudanpur. Recurrent neural network based language model. In *Proceedings of INTERSPEECH*, pages 1045–1048, 2010.

[77] T. Mikolov, S. Kombrink, L. Burget, J. H. Cernocky, and S. Khudanpur. Extensions recurrent neural network language model. In *Proceedings of ICASSP*, pages 5528–5531, 2011.

[78] D. Mrva and P. C. Woodland. A PLSA-based language model for conversational telephone speech. In *Proceedings of ICSLP*, pages 2257–2260, 2004.

[79] W. Naptali. *Study on n-gram language models for topic and out-of-vocabulary words*. PhD thesis, Toyohashi University of Technology, 2011.

[80] W. Naptali, M. Tsuchiya, and S. Nakagawa. Topic dependent class-based n-gram language model. *IEEE Transactions of Audio, Speech and Language Processing*, 20 (5):1513–1525, 2012.

[81] H. Ney, U. Essen, and R. Kneser. On structuring probabilistic dependences in stochastic language modeling. *Computer Speech and Language*, 8:1–38, 1994.

[82] J. Nie, R. Li, D. Luo, and X. Wu. Refine bigram PLSA model by assigning latent topics unevenly. In *Proceedings of IEEE ASRU Workshop*, pages 141–146, 2007.

[83] T. R. Niesler and P. C. Woodland. Combination of word-based and category-based language models. In *Proceedings of ICSLP*, pages 1779–1782, 1997.

[84] DB. Paul. The Lincon tied mixture HMM continuous speech recognizer. In *Proceedings of DARPA Speech and Natural Language Workshop*, pages 332–336, 1990.

[85] DB. Paul. Algorithms for an optimal A* search and linearizing the search in the stack decoder. In *Proceedings of ICASSP*, pages 693–696, 1991.

[86] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *Proceedings of SIGIR*, pages 275–281, 1998.

[87] B. Ramabhadran, O. Siohan, and A. Sethy. The IBM 2007 speech transcription system for european parliamentary speeches. In *Proceedings of IEEE ASRU Workshop*, pages 472–477, 2007.

[88] R. Rosenfeld. A maximum entropy approach to adaptive statistical language modeling. *Computer Speech and Language*, 10 (3):187–228, 1996.

[89] G. Salton. *Introduction to Modern Information Retrieval*. McGraw-Hill, 1983.

[90] H. Schwenk. Continuous space language models. *Computer Speech and Language*, 21:492–518, 2007.

[91] A. Sethy and B. Ramabhadran. Bag-of-word normalized n-gram models. In *Proceedings of INTERSPEECH*, pages 1594–1597, 2008.

[92] V. Steinbiss. Sentence hypothesis generation in a continuous speech recognition system. In *Proceedings of European Conf. on Speech Communication and Technology*, volume 2, pages 51–54, 1989.

[93] M. Steyvers. Matlab topic modeling toolbox. URL `http://psiexp.ss.uci.edu/research/programs_data/toolbox.htm`.

[94] A. Stolcke. SRILM-an extensible language modeling toolkit. In *Proceedings of ICSLP*, pages 901–904, 2002.

[95] Y.-C. Tam and T. Schultz. Dynamic language model adaptation using variational bayes inference. In *Proceedings of INTERSPEECH*, pages 5–8, 2005.

[96] Y.-C. Tam and T. Schultz. Unsupervised language model adaptation using latent semantic marginals. In *Proceedings of INTERSPEECH*, pages 2206–2209, 2006.

[97] A. Vaiciunas and G. Raskinis. Cache-based statistical language models of english and highly inflected lithuanian. *Informatica*, 17 (1):111–124, 2006.

[98] K. Vertanen. HTK Wall Street Journal training recipe. URL `http://www.inference.phy.cam.ac.uk/kv227/htk/`.

[99] Y. Wada, N. Kobayashi, and T. Kobayashi. Robust language modeling for a small corpus of target tasks using class-combined word statistics and selective use of a general corpus. *Systems and Computers in Japan*, 34:92–102, 2003.

[100] H. M. Wallach. Topic modeling: Beyond bag-of-words. In *Proceedings of ICML*, pages 977–984, 2006.

[101] P. C. Woodland, J. J. Odell, V. Valtchev, and S. J. Young. Large vocabulary continuous speech recognition using HTK. In *Proceedings of ICASSP*, pages 125–128, 1994.

[102] P. Xu and F. Jelinek. Random forests and the data sparseness problem in language modeling. *Computer Speech and Language*, 21(1):105–152, 2007.

[103] H. Yamamoto, S. Isogai, and Y. Sagisaka. Multi-class composite n-gram language model. *Speech Communications*, 41:369–379, 2003.

[104] S. Young. Large vocabulary continuous speech recognition: A review. Technical report, Cambridge University Engineering Department, 1996.

[105] S. Young, P. Woodland, G. Evermann, and M. Gales. The HTK toolkit 3.4.1., 2013. URL `http://htk.eng.cam.ac.uk/`.

[106] I. Zitouni. Backoff hierarchical class n-gram language models: Effectiveness to model unseen events in speech recognition. *Computer Speech and Language*, 21: 88–104, 2007.

Appendix A

Publication List

Journal Paper

1. M. A. Haidar and D. O'Shaughnessy. Unsupervised language model adaptation using LDA-based mixture models and latent semantic marginals. *Computer Speech and Language*, Volume 29, Issue 1, January 2015, Pages 20–31.

International Conferences

1. M. A. Haidar and D. O'Shaughnessy. Document-specific context PLSA language model for speech recognition. *Submitted to ICASSP*, 2015.

2. M. A. Haidar and D. O'Shaughnessy. Document-based Dirichlet class language model for speech recognition using document-based n-gram events. In *Proceedings of IEEE SLT Workshop* 2014.

3. M. A. Haidar and D. O'Shaughnessy. Interpolated Dirichlet class language model for speech recognition incorporating long-distance n-grams". In *Proceedings of COLING* 2014.

4. M. A. Haidar and D. O'Shaughnessy. Novel topic n-gram count LM incorporating document-based topic distributions and n-gram counts. In *Proceedings of EUSIPCO* 2014.

5. M. A. Haidar and D. O'Shaughnessy. PLSA enhanced with a long-distance bigram

language model for speech recognition. In *Proceedings of EUSIPCO*, 2013.

6. M. A. Haidar and D. O'Shaughnessy. Fitting long-range information using interpolated distanced n-grams and cache models into a latent Dirichlet language model for speech recognition. In *Proceedings of INTERSPEECH*, pages 2678-2682, 2013.

7. M. A. Haidar and D. O'Shaughnessy. Comparison of a bigram PLSA and a novel context-based PLSA language model for speech recognition. *In Proceedings of ICASSP*, pages 8440-8444, 2013.

8. M. A. Haidar and D. O'Shaughnessy. Topic n-gram count LM adaptation for speech recognition. *In Proceedings of IEEE SLT workshop*, pages 165-169, 2012.

9. M. A. Haidar and D. O'Shaughnessy. LDA-based LM adaptation using latent semantic marginals and minimum discriminant information. *In Proceedings of EUSIPCO*, pages 2040-2044, 2012.

10. M. A. Haidar and D. O'Shaughnessy. Unsupervised language model adaptation using latent Dirichlet allocation and dynamic marginals. *In Proceedings of EUSIPCO*, pages 1480-1484, 2011.

11. M. A. Haidar and D. O'Shaughnessy. Unsupervised language model adaptation using n-gram weighting. *In Proceedings of CCECE*, pages 857-860, 2011.

12. M. A. Haidar and D. O'Shaughnessy. Novel weighting scheme for unsupervised language model adaptation using latent Dirichlet allocation. *In Proceedings of INTERSPEECH*, pages 2438-2441, 2010.