NAIST-IS-DD1561024

Doctoral Dissertation

Unsupervised Representation Learning and Acoustic Modeling in the Zero Resource Scenario

Michael Heck

September 12, 2018

Department of Information Science
Graduate School of Information Science
Nara Institute of Science and Technology

A Doctoral Dissertation
submitted to Graduate School of Information Science,
Nara Institute of Science and Technology
in partial fulfillment of the requirements for the degree of
Doctor of ENGINEERING

Michael Heck

Thesis Committee:

Professor Satoshi Nakamura	(Supervisor)
Professor Yuji Matsumoto	(Co-supervisor)
Dr. Sebastian Stüker	(Karlsruhe Institute of Technology)
Associate Professor Sakriani Sakti	(Co-supervisor)

Für meine Familie.

Acknowledgements

I would like to express my heartfelt gratitude to Professor Satoshi Nakamura for giving me the opportunity to conduct the research for this thesis under his supervision at his Augmented Human Communication Laboratory (AHClab) at Nara Institute of Science and Technology (NAIST). From the first days as AHClab intern student in the spring of 2012 to finishing this thesis as doctoral course student, I had his unbroken support to pursue my research goals. I thank NAIST for granting me the International Scholarship to financially support my studies. Thank you to my supervisor Associate Professor Sakriani Sakti for the years that we worked together doing interesting and challenging research. My thanks go to all staff members of AHClab, Associate Professor Katsuhito Sudoh, Associate Professor Yu Suzuki, Assistant Professor Koichiro Yoshino, Assistant Professor Hiroki Tanaka, and former staff members Professor Tomoki Toda and Professor Graham Neubig for their valuable comments and assistance. I would like to thank the thesis committee members Professor Yuji Matsumoto and Dr. Sebastian Stüker for taking their time to read and comment on my thesis. I can not give enough thanks to Manami Matsuda for her unwavering support and her invaluable help at so many occasions, might they be lab related or be the challenges of life in Japan. For my time as intern at IBM Research AI I would like to thank Dr. Gakuto Kurata, Dr. Masayuki Suzuki, and the entire speech team in Tokyo. I learned a great deal and I am grateful for the warm welcome. To Professor Alex Waibel and everyone at the Interactive Systems Labs at Karlsruhe Institute of Technology, thank you for our years of study and research. Without them, I would not be where I am today. Thank you to all the people I met along the way, to the friends I made in Japan and at conferences around the globe. I am deeply grateful for my long-term friends, who walked with me for so many years, no matter how many kilometers between us. I thank my family from the bottom of my heart for enabling me to pursue all my studies wherever they led me, for their unconditional support, their presence and their love. Most grateful and humbled I am for knowing by my side Nunu, the most wondrous person who could have ever crossed my path, sticking with me ever since.

Ich bin meinen lanjährigen Freunden in tiefstem Dank verbunden, die mich all die Jahre auf meinem Weg begleiteten und es immer noch tun, mögen noch so viele Kilometer zwischen uns liegen. Von ganzem Herzen danke ich meiner Familie, die es mir erst ermöglichte, meinen Studien nachzugehen wohin auch immer diese mich brachten, für ihre bedingungslose Unterstützung, ihre Gegenwart und ihre Liebe. Zutiefst dankbar und in Demut bin ich, Nunu an meiner Seite zu wissen, der wundersamste Mensch der je meinen Weg kreuzte und bei mir ist seither.

Unsupervised Representation Learning and Acoustic Modeling in the Zero Resource Scenario[*]

Michael Heck

Abstract

Automatic speech recognition (ASR) has experienced a remarkable development over the decades. Technological advances however have largely been made on a small subset of all human languages that are rich in resources. This has two main consequences: Methods for ASR evolved such that they learn best from massive amounts of data, and the majority of languages, spoken by billions of speakers, has been neglected for a long time. Moreover, most of the world's languages have no written form and are therefore severely under-resourced. If besides raw speech data no other information about a language is available, we speak of a zero resource scenario. Inferring models in such a scenario is a challenging task, which can be compartmentalized into unsupervised learning of lexical units and unsupervised subword modeling.

This thesis addresses the problem of unsupervised subword modeling in the zero resource scenario. The two major challenges of unsupervised subword modeling are representation learning and model design. Representation learning is the task to find speaker independent, robust speech features without prior knowledge that highlight linguistically relevant properties and suppress irrelevant informations. Model design is the task to develop and infer a structure that approximates the true distributions of speech better than previous models.

This thesis approaches the representation learning problem by elaborating a novel framework for unsupervised subword modeling that takes advantage of automatically estimated feature transformations (Chapter 4). The proposed algorithm jointly learns transformations for the speech input without prior category

[*] Doctoral Dissertation, Department of Information Science, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-DD1561024, September 12, 2018.

knowledge and infers a Dirichlet process mixture model (DPMM) that represents sound classes. The incorporation of feature transformations into the unsupervised subword modeling framework considerably supports finding speaker independent, robust representations with high class discrimination properties. The proposed method proved its effectiveness in actual performance evaluations and delivered state-of-the-art performance in the zero resource challenges 2015 and 2017. The construction of a functional acoustic unit tokenizer shows that the found acoustic units carry meaning which can be utilized to solver higher-level problems (Chapter 5).

The model design problem is addressed by the introduction of a novel design for a Dirichlet process mixture of mixtures model (Chapter 6). Speech is inherently complex and requires models of appropriate complexity for proper representation. A long standing assumption in ASR research is that the emission of speech representations is modeled by multimodal distributions. As opposed to the unimodal modeling assumption of a standard DPMM, the novel algorithm proposed in this thesis can infer a mixture of mixtures to discover clusters in raw data that are made up of multimodal distributions. In experiments, the proposed design leads to the inference of fewer classes that represent subword units more consistently and show longer durations, which is a first step towards a fully unsupervisedly learned model for speech that represents units of appropriate length and complexity.

The methods presented in this thesis are ultimately designed towards enabling low and zero resource automatic speech recognition and provide a good basis for further research on the possibilities of learning acoustic units and acoustic features from scratch, without any prior category knowledge or other meta information about the target language.

Keywords:

acoustic unit discovery, Bayesian non-parametrics, Dirichlet process, mixture of mixtures, representation learning, unsupervised subword modeling

Contents

List of Figures

List of Tables

Chapter 1

Introduction

"Language is the archives of history – Language is fossil poetry." [38]
– Ralph Waldo Emerson (1803-1882), *Philosopher*

1.1 Language Acquisition

The early language acquisition of humans is a remarkable process which until today owes the research community satisfactory answers to the questions of how exactly it works. Infants leap into learning their future native language in a pace that is astonishing. In contrast, language learning for adults is a long and difficult process, and major breakthroughs in language acquisition by machines are still long in coming. Why is that? The rapid development of competences in comprehension and speech production in human infants is based on the ability to detect patterns and a statistical approach to learning [91]. This easily explains why statistical methods are the predominant approach to machine based speech processing. Besides – and on the side of – linguists and psychologists, it is the information scientists who likewise look for answers to the mechanics of language acquisition from a machine learning perspective.

1.1.1 Human Language Acquisition

During their first year of life, infants construct for themselves models of acoustic and lexical units in a robust way without any artificial and crafted supervision

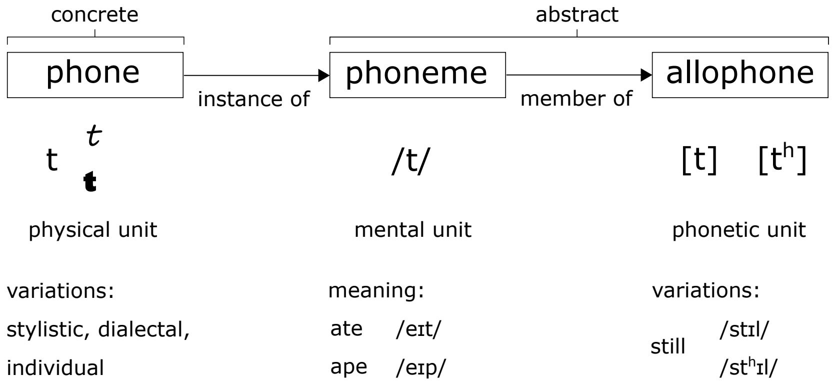


Figure 1.1: The relationship between phones, phonemes and allophones.

such as aligned textual references. Undoubtedly, language acquisition by humans to the point of mastering a language is a multisensory, multimodal learning process, but the bulk of subword modeling which happens very early in the development is resting on the speech input and starts as early as the time in the mother's womb. Equipped with basic pattern recognition abilities, infants learn from exposure and build up a statistical model of language.

Languages are based on a system of sounds, called phones. Sounds that do not change the meaning of words when replacing each other are called allophones. Phonemes as groups of allophones are distinct sound categories that, when replacing each other in a word, change the meaning of a word. For example, the phones */l/* and */r/* are allophones in Japanese, but instances of different phonemes in English. Early language acquisition solves the task of detecting phonemic categories, which is the requirement for building words later on.

Newborn have the ability to perceive the boundaries between phones. This is known as *categorical perception.* Interestingly, this ability is language indifferent, meaning that infants react to category boundaries even in sound systems that differ from their native language [36, 172]. From birth, infants are in fact equipped with the talent to discriminate between sounds in any human language [91], with the constraint that they rely on acoustic cues that are inherent to speech. *Categorization* follows the categorical perception and is the process of grouping, or clustering, of phones into phonemes. Infants naturally discard variance coming from exposure to different speakers, varying style, or context and are able to group sounds into categories [89, 90].

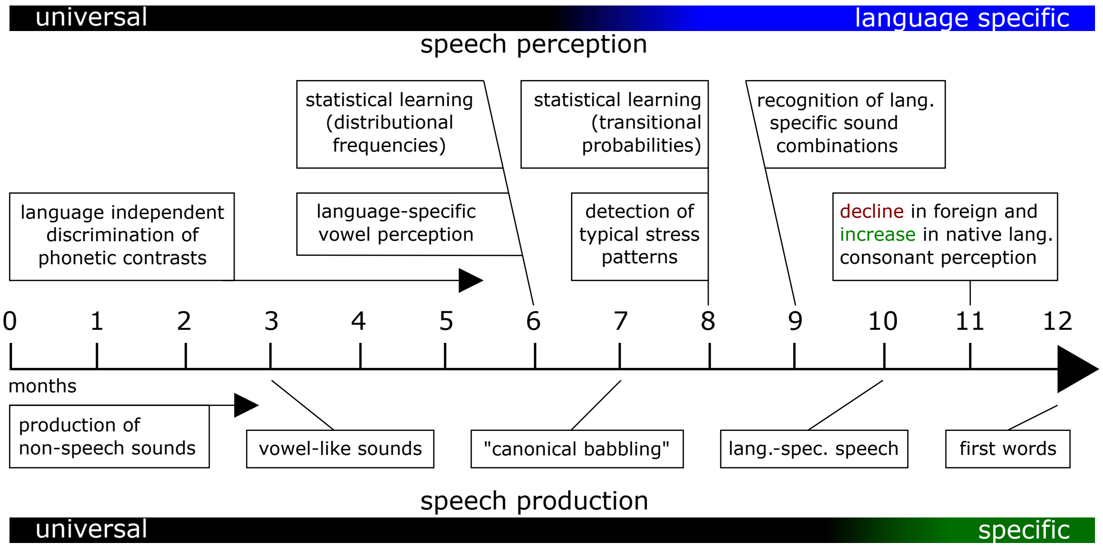

universal
language specific
speech perception
statistical learning (distributional frequencies)
statistical learning (transitional probabilities)
recognition of lang. specific sound combinations
language independent discrimination of phonetic contrasts
language-specific vowel perception
detection of typical stress patterns
decline in foreign and increase in native lang. consonant perception
0
1
2
3
4
5
6
7
8
9
10
11
12
months
production of non-speech sounds
vowel-like sounds
"canonical babbling"
lang.-spec. speech
first words
speech production
universal
specific

Figure 1.2: Universal speech perception and production development timeline, showing the stages of development during human infant's first year of language acquisition [91].

Word discovery and word production starts at the age of around one year. With one and a half years of age, children already understand about 150 words, a third of which they can produce themselves [91]. The word segmentation, like the discovery of sound units, is driven by the statistical learning ability. Infants have a sense for the probabilities of certain sequences, i.e., the phonotactics of their surrounding language. This helps them learn boundaries within and across words of their mother tongue. With the knowledge of words, the learning of meaning comes soon after.

1.1.2 Machine Language Acquisition

The parallels between the tools of human language acquisition and machine language acquisition are apparent. Infants naturally resort to pattern recognition and statistical models [91] without even knowing. Automatic speech processing systems similarly rely on these two basic tools [136]. In the context of automatic speech recognition, the learning of sound units is equivalent to training an *acoustic model*. Phonotactics, words, and linguistic structure is captured by a *language model*. As is the case for infants, the concept of words can not be on its own and presupposes the knowledge of smaller units, i.e., phonemes, phones, or subword

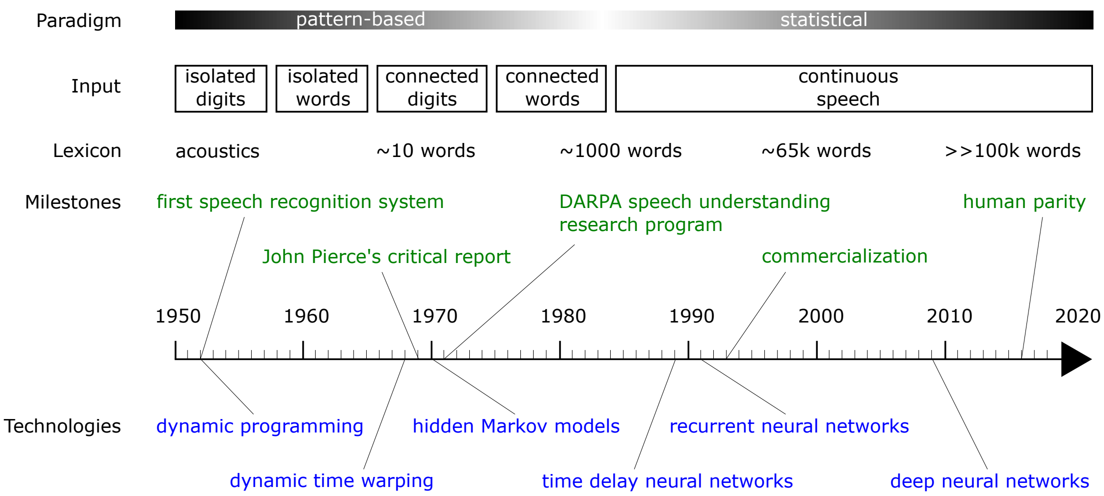


Figure 1.3: The timeline of automatic speech recognition development from the beginnings to today.

units in general.

Over the decades, automatic speech recognition (ASR) technology has experienced a remarkable development. From the first model designs for speech analysis and synthesis in the 1930s [33], the speech recognition research went a long way from simple systems that can identify isolated sounds, then words, to sophisticated systems that recognize continuous, natural speech by modeling variance in human language statistically [136]. In the 1980s, major developments in the field of neural network based methods for ASR emerged [171, 93, 100], which today dominate the field, allowing for extremely high performing systems that scratch on the upper bound of human speech perception capacities [175, 63].

In stark contrast to humans, machine learning approaches to learning models of human speech heavily rely on the availability of labeled training data. Parallel pairs of speech and text is utilized as training data for supervised model training. In the past decades, technological advances have been made mostly on a very small subset of all the human languages. The English language as experimental subject greatly dominates, followed by very few other world languages such as Spanish, German, Japanese and Chinese. What all these languages have in common is that large amounts of data to learn from are available. The development of speech recognition systems largely relies on the access to transcriptions for speech recordings to learn acoustics and large quantities of textual data to model the lexical characteristics of a language. With the rise of statistical methods for machine learning, the maxim has always been “there is no better data than more data”. Today’s ASR giants are trained on tens of thousands of hours of data, and billions of lines of text.

This development has two main consequences: Machine learning methods for ASR evolved such that they learn best from massive amounts of data. Supervised and unsupervised learning methods go hand in hand, but the paradigm is the same; more data (almost) never harms. The other consequence is that the focus has long been fixed on the major languages spoken in the world, to the great disadvantage of billions of speakers of minority languages, or languages that are *under-resourced* or *low-resourced*, i.e., with limited use in analog and digital media. For such languages, large corpora do not exist, which often forbids the use of state-of-the-art methods for ASR system development.

1.2 Under-resourced Languages

The larger part of all human languages do not have a writing system. This not only means that a unified way of transcribing speech is not available, but also that a language does not exist outside its spoken form. There is no literature, no presence in the internet or any other media besides spoken media. Of the roughly estimated 6000 [112] to 7000 [151] languages in the world, about 50% do not have a writing system, i.e., they are kept alive and passed on only orally. Of all the languages, the larger part is spoken by only 10.000 speakers or less each, and an estimated 50% and 90% might go extinct within the next century [59]. According to Ethnologue [151], about 2500 (or 35%) of all living languages are currently categorized as being “threatened” or worse. A threatened language “is used for face-to-face communication within all generations, but it is losing users” [151].

As of now, high-performance real-time ASR systems are only available for the top 20 languages of the world, reliable systems for the top 70. As an example we may look at one of today’s major speech recognition service providers, which currently supports real-time speech recognition for 63 languages. Ethnologue [151] defines 6 major and 135 other language families. Of the major families, 5 are currently covered by this provider with recognizers for 2 or more languages each, 1 family remaining unsupported for the moment. 35 of the supported languages

Table 1.1: Distribution of all living human languages by the number of first-language speakers, sorted by number of speakers per language [151].

Speakers	Living languages			Speakers		
per language	Count	Ratio	Cumulative	Total	Ratio	Cumulative
100M - 1B	8	0.1%	0.1%	2.7B	40.8%	40.8%
10M - 100M	82	1.2%	1.3%	2.6B	39.3%	80.1%
1M - 10M	307	4.3%	5.6%	948M	14.3%	94.4%
100k - 1M	956	13.5%	19.1%	305M	4.6%	99%
10k - 100k	1811	25.5%	44.6%	61M	0.9%	99.9%
1k - 10k	1980	27.9%	72.5%	7.6M	0.1%	99.99%
100 - 1k	1064	15.0%	87.4%	470k	0.007%	99.999%
10 - 100	329	4.6%	92.1%	12.2k	0.0002%	99.99999%
1 - 10	144	2.0%	94.1%	584	0.00001%	100%
0	219	3.1%	97.2%	-	-	-
unknown	199	2.8%	100%	-	-	-
Total:	7099			6.6B		

are Indo-European, i.e., members of only one major language family, accounting for more than half of the provider's language catalogue. Of the other language families, only 9 are covered, mostly with only 1 language per family being supported. This leaves a vast majority not only of languages, but indeed entire language families completely unsupported by speech recognition technology.

From a practical perspective this means that less than 80% of the world's population can theoretically make real use from speech recognition applications, but the other more than 20% remains unattended. This does not only affect societies that are more remote (from a Western perspective), but also a multitude of minority groups in countries whose national languages are well supported. To use a striking example, in the United States of America one can encounter 347 living languages, 220 of which are established, i.e., spoken by the native populations, and 127 are spoken by immigrant groups [151]. There are obvious reasons why one would want to develop speech recognition systems for under-resourced languages or zero resource scenarios. Speech processing technology has the potential to bridge gaps between peoples, and to provide everyone with access to the global community. But even though the consumer/provider perspective of services is not unimportant, there are much more pressing reasons.

A language is the cultural memory of a society. If a language disappears, knowledge disappears. Languages can teach us concepts that do not exist in other language spaces. The two countries with the highest count of living languages are Indonesia (709 languages) and Papua New Guinea (840 languages) [151]. The Papua New Guinean languages are members of the Trans-New Guinea language family, the single group of languages that is not yet covered by the above mentioned exemplary speech recognition service provider. Suffice it to say that invaluable knowledge of a tremendous amount of societies and communities is encapsulated in each and every one of these languages. Sadly, most of them are facing extinction rather sooner than later. Speech recognition technology for under-resourced languages could be a big help in the attempt to preserve languages from going extinct. In the worst case, machine learning methods could help analyze still-living languages before they might disappear forever. Lastly, every language that exists can teach us about human language understanding and language acquisition in general and tell us something about how we – as the only speaking species on this planet – function.

1.2.1 Speech Recognition for Under-resourced Languages

Despite the heavy focus on the few major languages in the world, it is important to point out that a considerable amount of research has been conducted for under-resourced languages across the globe as well [13, 62, 1, 164, 34]. Due to the fact that the major limiting factor for working with such languages is the data sparsity, most research efforts evolve around the efficient use of small amounts of language specific data for model learning. Particularly in recent years research on under-resourced languages flourishes. Topics of interest expanded to fields beyond speech recognition, including for instance the study of human language acquisition from a machine learning perspective, ways to preserve endangered languages by means of automatic analysis and more.

Generally, two polar trends are observable. One line of research aims at the utilization of massively multilingual training corpora for the development of versatile language independent models. Main techniques are bootstrapping and transfer learning from the so-called rich resource languages, as well as multilingual training and adaptation. The other line of research focuses on methods for inference of linguistic knowledge from minimal amounts of language specific data using the paradigms of semi-supervised and unsupervised learning and Bayesian inference.

Bootstrapping is the initialization and training of a system or model given existing systems or models that were trained for different domains, languages, modalities or conditions. Bootstrapping in automatic speech recognition is a popular method to initialize models for handling a new language, if not enough or no data is available for that particular language. This can be done by resorting to existing models that were trained on related [154] or unrelated [102] languages and/or by utilizing multilingual models [145, 169, 168]. Transfer learning is a general term for methods that utilize existing models or sets of parameters to support the training towards a particular language, domain, etc. Popular usages are model pre-training on multilingual data [156] or cross-lingual modeling [37] to compensate for data sparsity.

Multilingual speech recognition systems are systems that rely on language independent modeling. GlobalPhone is among the prominent long-term projects striving towards development of language independent acoustic models based on carefully collected and prepared multilingual training data to support multilingual speech recognition [147, 146, 143, 144]. One of the ideas of the GlobalPhone project is to circumvent the data shortness of under-resourced languages by relying on universal models for rapid adaptation to new languages. The concept of rapid language adaptation has already been successfully transferred to deep learning approaches [119, 114, 113].

The goal of the IARPA Babel program [62] was the development of methods that allow rapid ASR system development for under-resourced languages with minimal amounts of training data. The explored approaches lie between the two above-mentioned research directions and range from investigations of multilingual models [88] to data augmentation and monolingual semi-supervised and unsupervised learning [51, 94, 50]. The BULB project specifically targets unwritten languages and aims at supporting their documentation by bringing together linguists and computer scientists [1]. The goal is to develop tools such as language independent phoneme recognizers to produce universal phonetic transcriptions and pseudo-word alignments for an arbitrary number of languages.

From multilingual and language independent systems to the development of models with as little transcribed data as possible, recent research arrived at the far end of the machine learning spectrum by shedding light on the zero resource scenario, which shall be elaborated in the following section.

1.3 The Zero Resource Scenario

The *zero resource* scenario is an extreme case of resource shortage, where for a given language no other resources are available besides raw speech recordings (or the chance to gather recordings in the wild). Learning from speech without the availability of transcriptions, text corpora, phonetic and other linguistic knowledge is a tough challenge that increasingly draws the attention of the speech processing community, given the pressing reasons as stated in the previous section. The zero resource scenario is not simply a speech research exercise, it is, unfortunately, a fact for the vast majority of all existing languages that has to be approached rather sooner than later. The development of speech processing technologies for zero resource languages is paramount for tackling the manifold issues listed above, since multilingual or cross-lingual approaches are not always applicable. One might think of isolated languages or language groups that stretch less explored regions in the phonetic space. Another important factor to consider is the demand of multi- and cross-lingual approaches for large quantities of clean and balanced data, which binds many resources for acquisition. It is therefore reasonable to assume that methods which do not rely on other out-of-language resources might be a much needed and at the least beneficial contribution to technologies for under-resourced and unwritten languages.

Apart from practical reasons, linguistics and the theory of human language acquisition provide another strong motivation for zero resource research. Learning speech primarily from the auditory input is comparable to the learning challenge of infants. If the raw speech data of a possibly unknown language is the only available data, then the task from a machine learning perspective is to learn meaningful models without any additional knowledge besides the raw speech observations at hand. This is a fundamentally different view on the machine learning challenges for under-resourced languages and naturally invokes unsupervised learning methods. Successful approaches to language acquisition by machines might be able to help in answering fundamental questions about human language acquisition as well.

The challenge of learning linguistic knowledge from raw speech can be generally divided into two major tasks, unsupervised subword modeling, and unsupervised learning of lexical information. Language learning by machines is therefore similar to the task of early language acquisition by humans. Learning of phonotactic models and discovery of lexical units either relies on the availability of some sort of sound unit as elemental building blocks or jointly infers subword and word-like units. If the sound units of a language are unknown or principally unavailable because it is a newly discovered language, for instance, then unsupervised subword modeling might often be first task that needs to be solved. The inferred units might then be utilized for unsupervised learning of lexical information in the next learning stage. Similar to infant learners, unsupervised subword modeling is a critical step in the machine language acquisition process.

1.3.1 Unsupervised Learning of Lexical Information

Unsupervised learning of lexical information from speech, also known as spoken term discovery, is the unsupervised discovery of word-like units, defined as recurring speech signals [164]. A system that performs spoken term discovery takes raw speech as input and delivers a labeled segmentation of the input into recurring speech segments which indicates class memberships.

The general task can itself be broken down into smaller sub-tasks, pair matching, segment clustering and parsing [164]. Pair matching uses similarity measures to find pairs of speech snippets that seem akin. Segment clustering attempts to group matching pairs into larger categories. This step is equivalent to building some form of lexicon for speech data. Parsing is the matching of learned categories to segments in previously unseen data streams. Systems for spoken word discovery may perform all of those steps, or only parts of it. Sometimes the segmentation of speech into coherent fragments is prioritized, and parsing is not required. There has been work for sequential processing as well as joint model learning (examples for both are introduced in the following paragraph). The performance of spoken term discovery can be assessed by evaluating the matching quality, the clustering quality, the parsing quality or by using the learned model to solve a task such as audio document classification, retrieval or recognition.

Unsupervised learning of lexical information from raw speech is a popular theme since more than two decades. [29] introduces an unsupervised learning al-

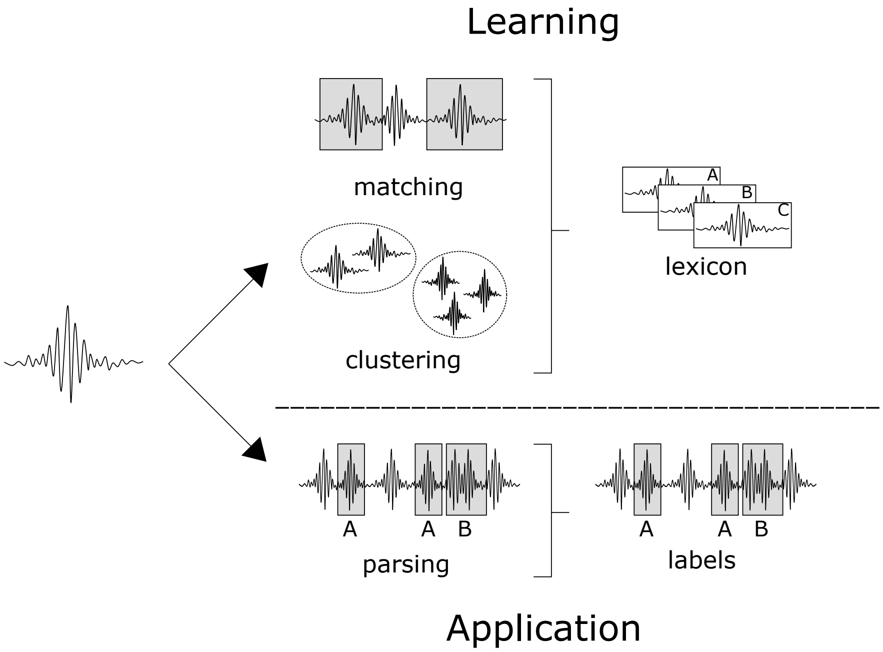


Figure 1.4: Components of a spoken term discovery system.

gorithm that infers a natural-language lexicon from raw speech. Their framework finds a hierarchical representation of language using the principle of minimum description length (MDL). The idea is to maximize the likelihood but also to penalize too complex models. The Bayesian framework by [54] uses Gibbs sampling to infer a word segmentation. They analyze their model on the background of human language acquisition theories. Other works try to infer a more fine-grained segmentation that allows words to be compounds of shorter morphemes [131, 27]. [118, 117] learns an entire language model from speech. They use Bayesian methods to jointly learn word boundaries and an n-gram language model with no prior linguistic knowledge from noisy input. The phonological lexicon discovery approach of [96] combines the unsupervised discovery of phoneme-like units and word-like units from raw speech. [83, 82] proposes a framework that embeds variable-length word segments in a fixed-dimensional acoustic space, in which they perform lexical clustering. The clustering is done by grouping acoustic word tokens in the new space. Best results were achieved using an infinite Gaussian mixture model sampler for model inference. [137] exploits a priori knowledge about the structure of speech signals and perform spoken term discovery based on a segmentation of input speech into syllable-like units. Their focus is on finding high quality segmentations that lead to a good clustering of segments into recurring units. [104] evaluates the performance of a set of graph clustering algorithms to cluster pairwise matches of word-like units into larger classes. [84] uses an embedded segmental k-means model to represent recurring spoken segments of arbitrary length as fixed-dimensional acoustic word embeddings.

1.3.2 Unsupervised Subword Modeling

Unsupervised subword modeling is the task of constructing a representation of speech that is robust to variation within and across speakers and that maximizes class discriminability [164]. A subword model should emphasize linguistically relevant informations contained in the original speech signal, and suppress irrelevant information such as speaker identity, channel characteristics, the influence of emotion and other artifacts that are commonly unused for solving the task of automatic speech recognition by machines.

The output of a system that performs unsupervised subword modeling is not strictly defined. Any representation that serves the discrimination of sounds may be a valid output. The simplest form of representation is a sequence of textual labels that discriminate frames or sequences of frames into distinct units, or members of unit categories. Other possible representations are lattices, vector embeddings or posteriorgrams. Subword models can be evaluated by training a classifier and decoding test data or by solving a higher-level task such as keyword spotting. One downside of these approaches is that the output quality of classifiers typically depends on several components. A good decoder might be able to compensate for weaknesses in the subword models and therefore cover up defects. A more direct way of quality assessment is to perform a discriminability task to measure the discrimination quality of sounds given the respective speech representation. Discriminability tests are invariant to feature dimensionality and sparsity and therefore provide a fair means of evaluating multiple models by comparison as well.

Research on unsupervised subword modeling goes similarly far back than unsupervised lexical learning. Unsupervised subword modeling is itself a two-fold problem. For once, it is a representation learning problem, i.e., the task is to find suitable features to adequately describe speech data. Besides that, the challenge

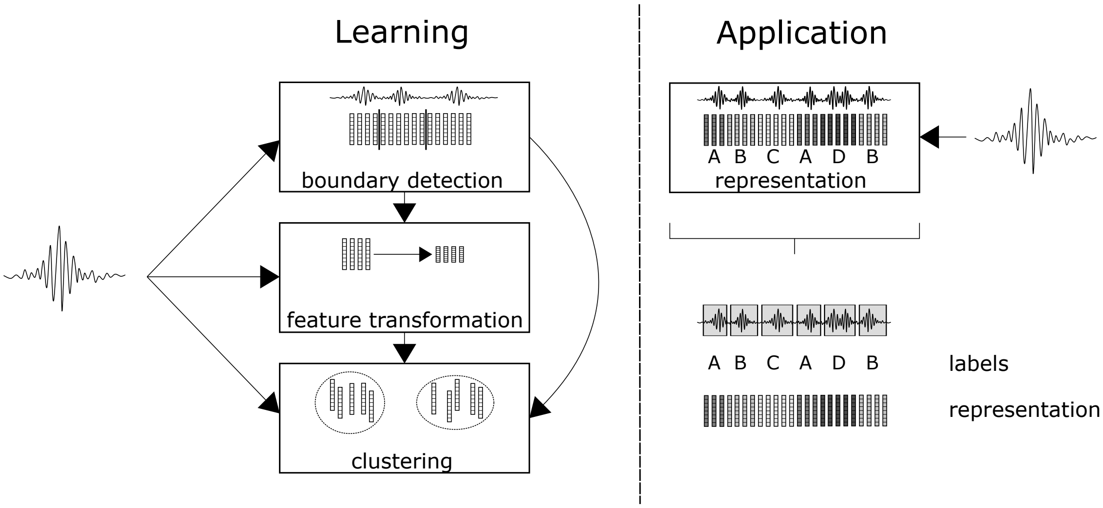


Figure 1.5: Components of an unsupervised subword modeling system.

of unsupervised subword modeling lies in designing a model that can properly represent the underlying subword units of speech. Systems for representation learning try to infer features from raw audio, with top-down constraints [139, 160, 81] or without [8, 183, 22], with no prior knowledge of phonetic categories. [158] describes a method for feature analysis in ASR systems based on locality preserving projections, which can be applied as a linear projection and dimensionality reduction algorithm to standard ASR features such as MFCC. It is argued that this method preserves local relations among input features. Similar to the proposed methods in this thesis, [183] generates new speech representations by computing posteriorgrams given an inferred model. They developed an unsupervised learning framework for spoken keyword detection which labels speech frames with Gaussian posteriorgrams. There are many recent works that utilize some form of weak automatic supervision from unsupervised term discovery (UTD) systems as weak top-down constraints to guide the subword modeling. [157] learns an acoustic model with no direct supervision. Their method however requires at least information about speech segments that are known to be similar and speech segments which are known to be different. An acoustic model is trained with neural networks which also results in a phonetics embedding. Similarly, [160] proposes a Siamese DNN training framework that takes the frames of UTD word pairs as input and minimizes the distance between frames of the same class and maximizes it between frames of different classes. [81] uses a deep auto-encoder neural network to build an unsupervised feature extractor. Pairs of isolated word examples are found using unsupervised term discovery (UTD), and their feature frames are aligned with dynamic programming. These aligned pairs provide weak top-down supervision by their usage as input output pairs to train the auto-encoder. [139] likewise applies a correspondence auto-encoder to learn efficient representations with the help of matched word pairs generated by an unsupervised term discovery system, and [8] makes use of a deep auto-encoder that applies a threshold at the encoding layer to generate a binary representation of speech frames. The Bayesian modeling framework by [22] infers a Dirichlet process Gaussian mixture model (DPGMM) from raw speech. The inference provides a clustering of MFCC speech features into distinct classes, and posteriorgrams over the inferred GMM are used as new speech representation. This approach was outperforming neural net based alternatives by a margin during the zero resource speech challenge (ZeroSpeech) [164] and confirms the modeling powers of Bayesian non-parametric methods as shown by earlier work such as [95]. The submissions for the follow-up challenge introduce various novel or refined approaches, many of which are multilingual. [127] derives new speech representations by estimating cluster centroids for zero component analysis (ZCA) transformed feature vectors using k-means and measuring the distance of each input feature vector to these centroids. [23] makes use of the DPGMM approach of [22] by clustering speech feature vectors into classes, which are then used as targets to train a multilingual multi-task neural network. A new speech representation is extracted from a bottleneck. Being a multilingual approach as well, [5, 6] trains a DNN with bottleneck, using posteriors and labels that come from a universal background model. The bottleneck features are combined with features from an auto-encoder trained on standard speech feature vectors. Following [160, 139], the system of [180] uses STD to match acoustic segments that are represented with multilingual bottleneck features which were trained in the same way as in [23]. A DNN is trained on matched frame pairs and a new speech representation is extracted from a hidden layer. The winning contribution to the latest ZeroSpeech [34] however was again a (monolingual) Bayesian modeling framework, which has been developed by us [69] and which is the central theme of this thesis. The following chapters of this thesis are dedicated to laying out the foundations and details of our Bayesian non-parametric approach to unsupervised subword modeling.

1.4 Scope of this Thesis

This thesis wants to address the task of unsupervised subword modeling in the zero resource scenario. We consciously chose to tackle this extreme case of data sparseness for the manifold reasons stated in the previous sections, the most important of which are that

1. most of the world's languages do not have a written form, which makes it impossible to rely on transcriptions,

2. multilingual or cross-lingual approaches are not always applicable or are simply not desired due to their demand for large quantities of suitable training data, and

3. inferring the acoustic units of a possibly unknown language from raw speech is one of the first steps in human language acquisition and therefore a logical choice for machine learning approaches to language learning as well.

Within the scope of this thesis, the major challenges of unsupervised subword modeling shall be discussed, and solutions be provided.

There are two major problems concerning the unsupervised learning of speech representations from raw speech. Today's speech processing technology is not capable of imitating the extraordinary process of human language acquisition. Infant learners of different languages make use of different features when discriminating sounds. This is known as the dimension learning problem. In the machine learning context this task is also known as *representation learning problem* and raises the question how well discriminative speech features can be learned without prior knowledge, and whether they generalize well across languages. Supervised methods that heavily rely on large amounts of transcribed training data still dominate the field. Many advances in supervised learning are difficult to apply in the absence of supervision, one of which is supervised representation learning by estimating feature transformations with helpful properties. The second problem is the *model design problem.* A model that can be inferred from raw data should ideally represent categories that have some resemblances to subword units as defined by humans, e.g., syllables, phones or sub-phones.

1.4.1 Representation Learning

In linguistics literature, representation learning is also known as dimension learning. Dimension learning as a process of human language acquisition starts in the first year of life and continues into the time of early childhood [173, 92]. Speakers differ cross-linguistically in the cues that they use to discriminate phonemes within their respective languages [35, 45, 103, 179, 101]. It is assumed that the dimension learning is governed by the need to properly discriminate phonemes, and that changes in speech perception are based on developing sound category knowledge [14, 40, 162]. This theory presumes prior knowledge of phonetic categories, at least to some extent, which is acquired during the first year of life. Dimension learning by humans is therefore dependent on the knowledge of sound categories. The alternative hypothesis that dimension learning can happen without category knowledge has support as well [80].

The idea of representation learning is to transform raw speech features into a form that has better properties such as increased sound discriminability and reduced variance in order to be better suited for ASR tasks. Supervisedly trained ASR systems make extensive use of such feature transformations [44, 56, 49, 4, 48], often exploiting various forms of category knowledge. Unsupervised systems for representation learning try to infer good features from raw audio data, with top-down constraints [139, 160, 81] or without [8, 183, 22], but principally without any prior knowledge of phonetic categories.

The challenge for representation learning for low resource and zero resource languages lies in the unavailability of any meta-data. A method for representation learning should be able to generalize across languages, perform comparably well for any arbitrary input, and scale with the data. Inferred representations should be computationally feasible, generalize well across speakers and suppress linguistically irrelevant informations such as channel characteristics, speaker emotion, and other variance.

1.4.2 Model Design

Sounds of previously unexplored languages are cataloged by phonologists. Their expert knowledge about perceptual dimensions and phonetic categories allows them to determine the underlying sound repertoire of a given language. Basic machine learning approaches to this are pattern matching on raw audio data [124, 125] and unsupervised learning of models [163]. These techniques have been successfully applied to solve tasks such as spoken term detection [183], topic segmentation [108] or document classification [32].

However, model complexity usually is not known a priori when dealing with new data sets and where estimation is not possible due to the lack of development data. Bayesian non-parametric models can be a good choice in such cases, as they automatically adjust the model complexity given some data. Bayesian models have already been successfully applied to other speech processing tasks such as unsupervised lexical clustering [83]. Bayesian non-parametric models were already successfully applied to the task of representation learning [22].

There is a discrepancy between the models of supervisedly trained ASR systems and the models that can be inferred for instance by Bayesian non-parametrics. The model assumptions in the latter case are often overly simplified. A typical case is that individual categories are modeled as components in a mixture model, where the components themselves are parametrized as basic probability distributions such as the Gaussian. ASR systems on the other hand have long been made extensive use of hidden Markov models with continuous multimodal emission probabilities to model sounds. One challenge for the design of novel model inference methods is to reduce the size of this gap between model structures for the purpose of representation learning. The problem is how to design a model for inference from data so that it represents units of appropriate length and complexity.

1.5 Contribution

1.5.1 Tackling the Representation Learning Problem

This thesis addresses the *representation learning problem* by elaborating a novel framework for unsupervised subword modeling that takes advantage of methods commonly used in supervised speech recognition systems. The main representation learning method is rooted in Bayesian non-parametrics. A Dirichlet process mixture model (DPMM) is used to infer sound classes from raw speech, but the learning framework is augmented by the aspect of speech feature transformation estimation. The proposed algorithm jointly learns useful feature transformations and an improved class model for the underlying sounds of the input. Feature

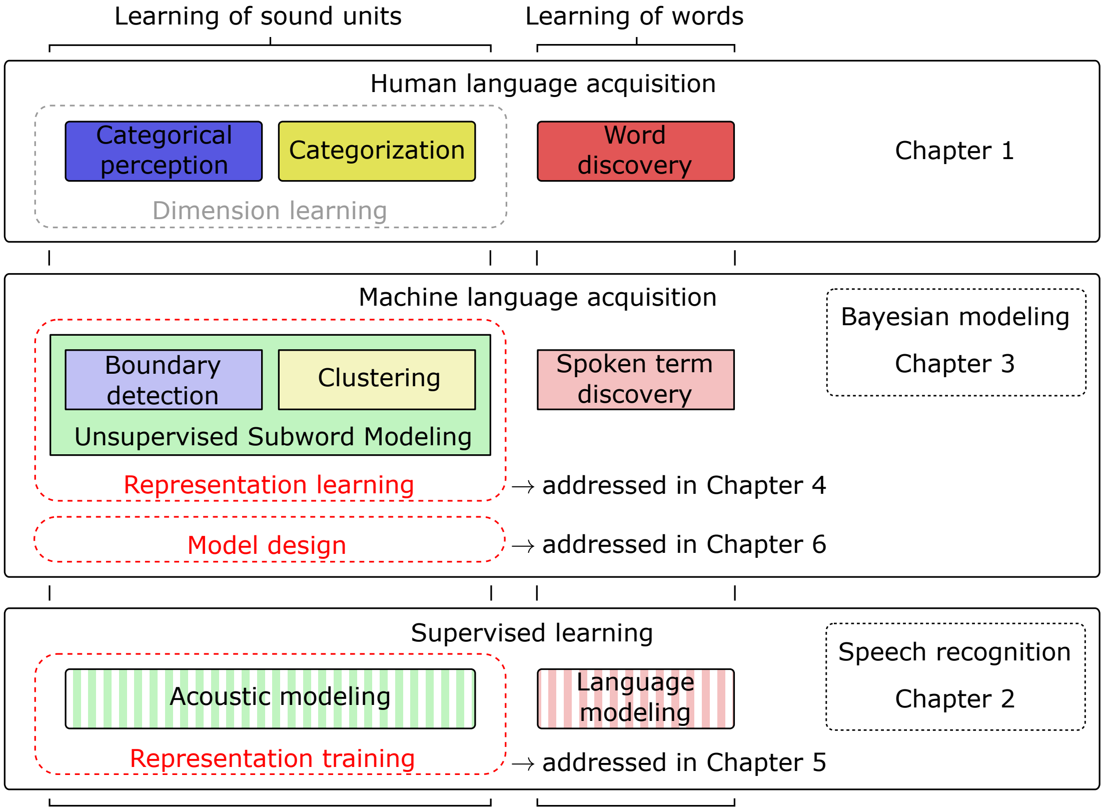


Figure 1.6: Overview of the topics. Matching colors indicate related tasks.

optimization is an integral part of speech processing systems, and methods that improve class discriminability should naturally be beneficial for solving clustering problems. The proposed algorithm is able to utilize popular transformations that are well-established in ASR without supervision. The transformation estimation is performed without any prior category knowledge by utilizing automatically generated labels for the data. The learned transformations in turn support the inference of acoustic units from the data. Posteriorgrams defined over the inferred mixture model then serve as new speech representation. The proposed method proved its effectiveness in actual performance evaluations and delivered state-of-the-art performance on the official data sets for the two most recent ZeroSpeech challenges 2015 and 2017. Compared to similar methods, the proposed framework proves to generalize well across languages, speakers and speech modalities, and also scales well with data. The construction of a functional acoustic unit tokenizer shows that the found acoustic units carry meaning which can be utilized to solver higher-level problems.

The decision to take a Bayesian non-parametric approach is motivated as follows. Although parametric models such as binarized [8] or correspondence auto-encoders (cAE) [139] have shown to be effective for the task of representation learning, the trained models do impose limitations upon the learnable features by pre-defining their shape. By fixing the model complexity, the representation might be inflexible towards differing data sizes or problems of varying modeling difficulty. A model that was defined for one use case can not be easily applied to a novel problem and expected to perform equally well without prior adjustments. Further, the cAE and other such methods that utilize top-down constraints [139, 160, 81] naturally rely on some form of supervision, such as word-like acoustic pairs that are inferred or otherwise proposed by already existing spoken term discovery (STD) systems. Although STD systems can be trained without supervision as well (see Section 1.3.1), the reliance on top-down constraints introduces potential sources of uncertainty and error, which can result in representations that are not optimal given all the available data.

Bayesian non-parametric approaches such as the DPMM on the other hand leave the estimation of the appropriate model complexity to inference given the provided data. The advantage of flexible model complexity that depends on the provided data is that the same model type can be deployed to solve problems that might greatly differ in their difficulty or data size. Not needing to fix the model structure a priori reduces the development overhead and the risk of imposing limiting design decisions. The inference of a descriptive model further holds the advantage of being a tangible representation of the data, whereas above-mentioned models purely serve as feature extractors with little descriptive power. Descriptiveness however might be a desirable trait, especially if the goal is to represent natural phenomena such as speech not only for tackling downstream tasks but also for the purpose of analysis. All this is not to say that Bayesian non-parametric approaches don't demand some thoughts on model design. Foremost, the choice of the prior for a DPMM determines the complexity of the individual model components. For example, a DPMM that utilizes a normal inverse Wishart (NIW) prior will consist of Gaussian components, therefore constitutes a Dirichlet process Gaussian mixture model (DPGMM).

In this thesis, raw speech is processed on the level of frames, and unsupervised subword modeling is approached bottom-up. It is noteworthy that methods such as DPMMs are indifferent to the kind of data being used as input. It would be straightforward to impose top-down constraints similar to the above mentioned works by processing the raw input with UTD systems. Any structured output can be used as input to non-parametric methods, so that whole segments of speech frames could be processed at once. This, however we deem not desirable due to two major reasons. Imposing top-down constraints for instance by UTD (1) introduces new sources of error, and (2) evades the research question whether sensible acoustic units can be inferred from raw data in a bottom-up fashion. The first problem can be handled in various ways, for instance by joint modeling of segments and classes. The second point however is a matter of the chosen research objective. This thesis focuses on exploring the capacities of bottom-up learning approaches towards unsupervised subword modeling.

1.5.2 Tackling the Model Design Problem

Speech – the input modality that this thesis focuses on – is an inherently complex signal type and requires models of appropriate complexity to be represented properly. It is a long standing assumption in ASR research that the emission of speech representations is modeled by multimodal distributions, as opposed to the unimodal modeling assumption of a standard DPMM, for instance. A novel model and inference design for a Dirichlet process mixture of mixtures model (DPMoMM) presented in this thesis addresses the *model design problem*. The DPMoMM allows the distribution of data to be approximated a mixture of mixtures. The model inference therefore enables the discovery of clusters in raw data that are made up of multimodal distributions, a model design that is supposed to approximate the true distribution of real speech observations more reliably. In experiments, the proposed design leads to the inference of fewer classes that represent subword units more consistently and show longer durations, which is a first step towards a fully unsupervisedly learned model for speech that represents units of appropriate length and complexity. The methods presented in this thesis are ultimately designed towards enabling low and zero resource automatic speech recognition and provide a good basis for further research on the possibilities of learning acoustic units and acoustic features from scratch.

1.5.3 Outline

The structure of the rest of this thesis is as follows. Chapter 2 is an introduction to the general ASR framework, the challenges of ASR system development with differing levels of supervision and especially in the zero resource scenario. The chapter closes by putting Bayesian non-parametrics into the context of zero resource research. Chapter 3 covers the topic of Bayesian non-parametrics by guiding through the basics of stochastic processes and providing more detailed informations about the Dirichlet process mixture model as the main technique used throughout this work. The details of how to incorporate feature transformations into the DPMM based unsupervised subword modeling framework is laid out in detail in Chapter 4. The proposed method builds upon the work of [22], which utilizes the DPMM to learn subword units. With the help of the proposed expansions, state-of-the-art performance is established. In Chapter 5, experiments are described which demonstrate that the inferred subword units carry meaning which can be utilized to solve higher-level problems. The chapter explains how an acoustic unit tokenizer can be built from scratch without prior knowledge of sound categories and target language. Evaluation suggests that the tokenizer produces output that is of higher quality than the output of a DPMM sampler. Chapter 6 tackles the model design problem by introducing a novel design for a Dirichlet process mixture of mixtures model (DPMoMM). Inference is done by a parallelizable Markov chain Monte Carlo split and merge sampler. The sampler jointly infers a codebook and clusters, where the codebook is a global collection of components and clusters are mixtures, defined over the codebook. A non-ergodic Gibbs sampler is combined with two layers of split and merge samplers to form a valid ergodic chain. An additional switch sampler is introduced to support convergence. Experimental results show that the proposed model and algorithm infers complex classes from real speech feature vectors that consistently show higher quality on several evaluation metrics. At the same time fewer classes represent subword units more consistently and show longer durations, compared to a DPMM sampler. Finally, Chapter 7 discusses the overall findings and possible future research in the area of unsupervised subword modeling and the prospect of fully unsupervised automatic speech recognition.

Chapter 2

Automatic Speech Recognition

“Opera naturale è ch’uom favella;
ma così o così, natura lascia
poi fare a voi secondo che v’abbella.” [1] *[3]*

– Dante Alighieri (1265-1321), *Poet*

Automatic speech recognition (ASR), also known as speech-to-text (STT) conversion is the task of converting spoken language into written form with the help of a machine. Research interest in ASR is predating the time of personal computer systems. Since then, the technology has found its way into almost every conceivable use case. Smart phones, smart homes, embedded systems, industrial, educational, clinical, office and many other environments are equipped to recognize speech for purposes such as human-machine interaction, communication, dictation, entertainment and more. Client-server architectures make it possible to provide high-performance ASR for any purpose and any place provided with an internet connection. An increasing number of languages is covered by today’s providers of high-performance ASR systems, the focus however lies on the world’s top 50 or top 100 languages, leaving many opportunities for novel developments to serve the so-called under-resourced languages which number in the thousands (see Chapter 1).

This chapter introduces the basics of the ASR framework and guides through

[1] “It is the work of nature man should speak
but, if in this way or in that, nature leaves to you,
allowing you to choose at your own pleasure.”

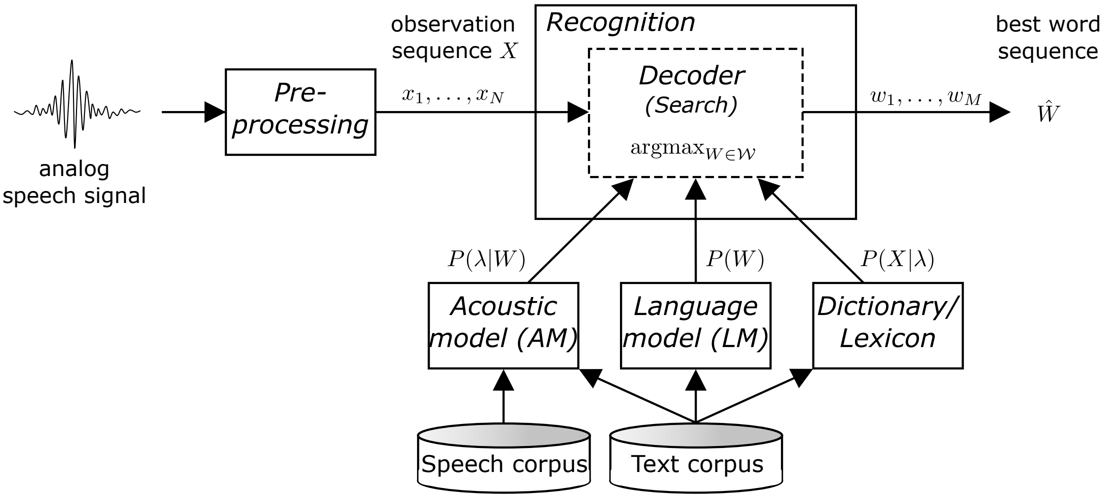


Figure 2.1: The automatic speech recognition framework.

the principal components of a statistical speech recognition system. The main modeling method for the statistical paradigm of speech is the hidden Markov model (HMM) [79, 98, 135, 134, 75], still being highly relevant in developing neural network based systems as proto model or prior. Therefore, large parts of this Chapter are dedicated to the HMM.

2.1 The Statistical Framework

Contrary to the early days where rule-based modeling of human language was the predominant premise, today's state-of-the-art ASR systems almost exclusively define speech as a stochastic process. Modeling and decoding speech rests on methods of probabilistic modeling and statistical pattern recognition [136]. The statistical framework describes the task of automatic speech recognition as a decoding task with the objective to convert an encoded message stream, i.e., a sequence W of spoken words $w_1, \dots, w_M$, represented by a stream X of real valued feature vectors $x_1, \dots, x_N$ into the most likely sequence $\hat{W}$ of written words according to the maximum-likelihood criterion. It is precisely the task of the decoder to find $\hat{W}$ given the sequence of speech representations X of the original spoken sequence of words W. We can decompose this task into several sub-problems with the help of mathematical reformulation. Identifying the best sequence of words $\hat{W}$ out of all possible word sequences $\mathcal{W}$ given the input can be formulated according to Bayes's law as

$$\hat{W} = \underset{W \in \mathcal{W}}{\operatorname{argmax}} P(W|X) \tag{2.1}$$

$$= \underset{W \in \mathcal{W}}{\operatorname{argmax}} \frac{P(X|W)P(W)}{P(X)} \tag{2.2}$$

$$= \underset{W \in \mathcal{W}}{\operatorname{argmax}} P(X|W)P(W), \tag{2.3}$$

where $P(W|X)$ is the probability of words W being observed, given X. $P(X|W)$ is the likelihood of the data given W. $P(X)$ is the a priori probability of observing X. As W is maximized by the search argmax, $P(X)$ is constant and therefore negligible in the classification decision [79]. $P(X|W)$ is the probability that a stream of feature vectors X is observed, given the input sequence W of spoken words. This formulation is commonly known as the *fundamental equation of speech recognition*.

The component $P(X|W)$ can be further decomposed the sub-components $P(X|\lambda)$ and $P(\lambda|W)$, which leads to the following equation:

$$\hat{W} = \underset{W \in \mathcal{W}}{\operatorname{argmax}} P(X|\lambda)P(\lambda|W)P(W). \tag{2.4}$$

Equation (2.4) now models the components of a general ASR system in its entirety:

1. X is the result of the pre-processing, or feature extraction. The pre-processing converts an analog signal into a time discrete digital representation, typically into a stream of multidimensional feature vectors that are designed to preserve the relevant information of human language.

2. $P(X|\lambda)$ is known as the acoustic model and estimates the probability that the stream of observations X was generated by subword models λ.

3. $P(\lambda|W)$ is the dictionary, or lexicon, and estimates the probability of subword models λ given the original word sequence W. The lexicon is the link between the acoustic model and the language model.

4. $P(W)$ is the language model and estimates the prior probability of observing the word sequence W.

5. $\text{argmax}_{W \in \mathcal{W}}$ is the search operation and describes the decoding process which finds the most probable sequence of words $\hat{W}$, given the input X and the above-mentioned statistical models of speech.

Provided that the acoustic model and language model along with the respective dictionary are known, the Bayes formula gives us the optimal mathematical definition for decoding speech [120]. Besides the search itself, the challenge is in finding the probability distributions in Equation (2.4), for which approximations are necessary. The major part in training an ASR system is concerned with computing these approximations.

The acoustic speech signal needs to be transformed into a parametric representation for further processing by an ASR system. The digitization results in a representation of the time domain based continuous wave form as time discrete, quantized digital signal. Further pre-processing results in a stream of multi-dimensional feature vectors over time. This conversion process is described in the following section, before the Chapter proceeds with describing the involved models of speech.

2.2 Pre-processing

The purpose of pre-processing is to convert an analog signal into a digital representation, with the goal to extract relevant information from the speech signal and to discard information that is not helpful for solving the task of speech recognition. A digital representation of speech for the purpose of ASR would ideally be independent of speaker identities and characteristics, invariant to channel characteristics such as changing environments or equipment and contain only such information that is directly related to human speech, as opposed to non-speech events.

For this reason, the pre-processing step is also known as feature extraction, and many works have focused on developing efficient feature designs for ASR. Methods such as linear predictive coding [109] and cepstral analysis [121, 122] provide a solid basis for advanced techniques such as perceptual linear prediction (PLP) [72] and Mel-frequency cepstral coefficients (MFCC) [17, 111], which are the most widely used today.

The source-filter model of speech is a popular approximation to the nature of human language which allows for some simplifications that serve as basis for the principal idea of all pre-processing approaches. Assume that $e[n]$ is the air flow at the vocal cords, i.e., the *source*, and $h[n]$ is the resonance of the vocal tract, i.e., the *filter*. Then the source-filter model of a speech signal $x[n]$ is

$$x[n] = e[n] * h[n], \tag{2.5}$$

where $(*)$ is the operator for convolution. Most of the relevant information encoded in a speech signal is generated by the vocal tract, i.e., the filter $h[n]$, which changes over time. In order to extract the desired information, one has to find a method for de-convoluting source and filter [142]. This is precisely what the above-mentioned methods commonly try to achieve. The general pre-processing performs some kind of spectral analysis, which is then followed by a de-convoluting operation. To track temporal changes in the speech signal, the spectral analysis has to be done on short-term snippets of the speech signal, in which periodicity is assumed. For that, the pre-processor shifts a window over the input signal and processes the data frame-wise. A typical window size for speech recognition would be 25 msec with a shift of 10 msec. This is also known as short-time spectral analysis. The popular and widely used Fourier analysis, which makes use of the Fourier transformation is one such method [26].

2.2.1 Fourier Analysis

The Fourier transformation is a method for spectral analysis and effectively transforms a signal by breaking it up into a sum of complex sine and cosine functions with individual frequencies. The Fourier transformation assumes the input signal to be infinite and periodic, which is given by the windowing process in form of quasi-stationary sound snippets.

In practice, the fast Fourier transform (FFT) [26], an efficient algorithm for the discrete Fourier transform (DFT) is used for the spectral analysis. Applied to the windowed input, frame-wise power spectra are computed. What follows after the spectral analysis defines the type of feature vector that is produced. MFCC and PLP shall be explained exemplarily to lay out the principle idea of the pre-processing.

2.2.2 Mel-frequency Cepstral Coefficients (MFCCs)

MFCCs are the most widespread type of features for ASR. They are spectral features, warped by a non-linear Mel-scale filterbank to approximate the perceptual scale of the human ear, and defined over the cepstral domain. MFCCs provide robustness towards speaker and channel variability by de-emphasizing irrelevant information with respect to general speech processing tasks. Specifically, the goal is to preserve the influence of the vocal tract on the speech signal and to remove the glottis wave form, i.e., the excitation signal [142]. The pipeline for the computation of MFCCs is depicted in Figure 2.2.

After windowing the speech signal, FFT is applied to each frame. The resulting spectral features are piped through a Mel-scale filterbank. The Mel-scale is defined as

$$\mathrm{Mel}(f) = 2595\log(1 + \frac{f}{700}), \tag{2.6}$$

where f stands for frequency. The higher frequency bins have a broader bandwidth, thereby approximating the auditory properties of the cochlear duct of the human ear. The inverse discrete cosine transformation (DCT^{-1}) projects the filterbank output into the cepstral space, where the coefficients are "liftered" by omitting the 0th coefficient and higher-order coefficients. A typical number of coefficients left after these steps is 13.

2.2.3 Perceptual Linear Predictive (PLP) Features

PLP features are based on the concept of psychophysics of hearing [72]. The method is identical to LPC with the difference that the spectral features are transformed to approximate the human auditory system. Similar to the computation of MFCCs, spectral features are warped with a Bark-scale filterbank. The Bark-scale can be expressed as [161]

$$\mathrm{Bark}(f) = \frac{26.81}{1 + (1960/f)} - 0.53. \tag{2.7}$$

The filterbank output is weighted by an equal-loudness pre-emphasizer to account for the human hearing sensitivity. An intensity-loudness pre-emphasizer transforms the coefficients according to Stevens' power law.

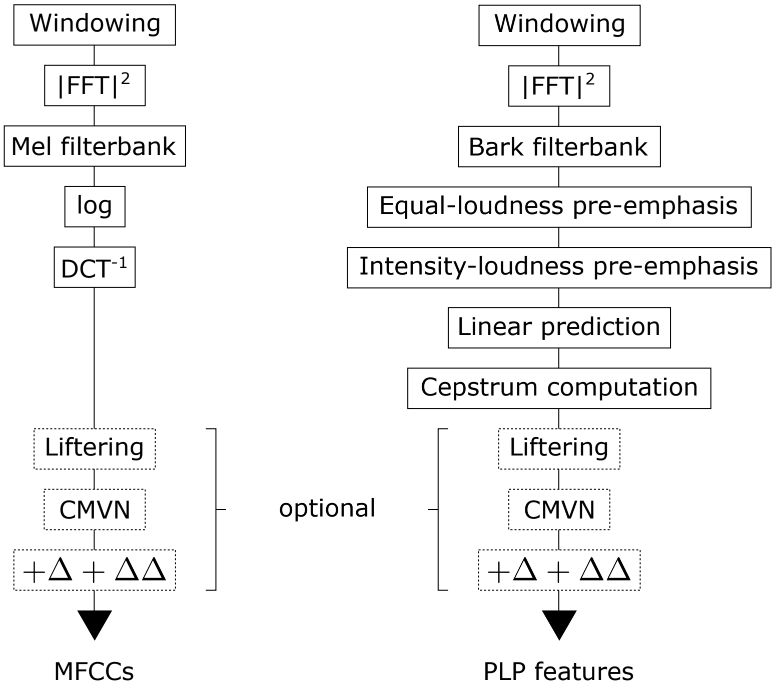


Figure 2.2: Pipelines for calculating Mel-frequency cepstral coefficients (MFCCs) and perceptual linear predictive (PLP) features. The steps marked with dotted frames are not defining for the respective features, but are commonly used for automatic speech processing.

Following these transformations, linear prediction is used to compute predictor coefficients of a (hypothetical) signal that has this warped spectrum as a power spectrum [73]. The last step is the computation of cepstral coefficients. The full pipeline is depicted in Figure 2.2.

MFCC and PLP features alike are often further enhanced by additional processing steps. To reduce the influence of the channel, cepstral mean variance normalization (CMVN) is applied by subtracting the average of the cepstral values. To capture temporal information in form of spectral changes, the first and second order derivatives of a feature vector, denoted as Δ and $\Delta\Delta$, are stacked on top of it [47].

2.3 Acoustic Modeling

The acoustic model is the sum of all structural and parametric knowledge about elementary acoustic units of an ASR system. The purpose of the acoustic model is to provide a method of computing the likelihood of any sequence of feature vectors, given a specific sequence of words. It is impractical for speech recognition systems to model words as a single entity. Therefore, words are usually modeled as a compound of smaller acoustic units such as phones, which themselves are further decomposed into sub-units. The ideal elementary sub-unit should be defined so that it can be modeled acoustically precise and statistically robust. Hidden Markov models (HMMs) are especially useful for modeling dynamic processes that can be structured into discrete states. The emission of speech observations is such a process.

2.3.1 Hidden Markov Models

A hidden Markov model is a five-tuple (S, π, A, V, B), where

- $S = s_1, \cdots, s_n$ is the set of all states of the HMM,
- π is the probability distribution of the start states, where $\pi(i)$ is the probability of s_i being the initial state,
- $A = (a_{ij})$ is the state transition matrix, a_{ij} being the probability of a transition from s_i to s_j,
- V is the feature space of b_i, where in the discrete case $V = v_1, v_2, \cdots \Rightarrow b_i$ is a probability, and in the continuous case $V = (R)^n \Rightarrow b_i$ is a density, and
- $B = b_1, \cdots, b_n$ is the set of emission probabilities for a discrete V, or emission densities for a continuous V, where $b_i(x)$ is the probability of observing x when being in state s_i.

For mathematical correctness the following stochastic constraints must be satisfied. For start probabilities, it must be $\sum_{i=1}^{n} \pi(i) = 1$. A common set-up in practice is $\pi(0) = 1$ and $\pi(i) = 0 \ \forall i > 0$. For transition probabilities, it must be $a_{i,j} \geq 0 \ \forall i, j$ and $\sum_{j=1}^{n} a_{i,j} = 1$, i.e., all outgoing transitions of a state s_i have to be 1.

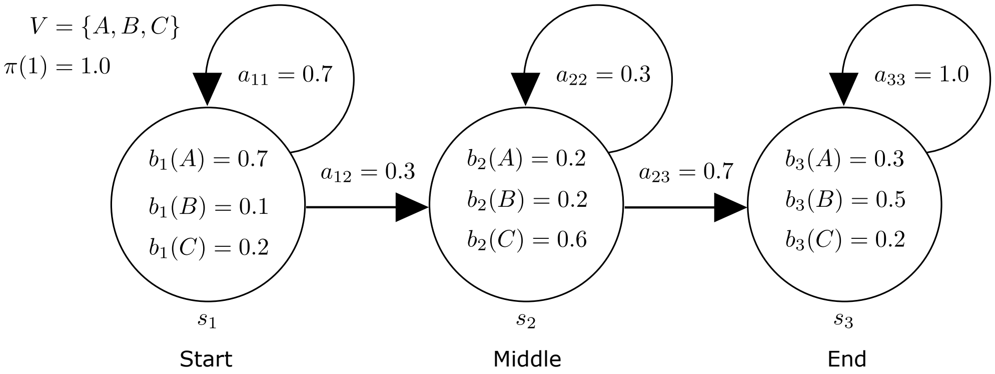


Figure 2.3: A standard HMM with left-to-right topology.

The AM is comprised of HMMs that each model a particular acoustic unit, typically a phone, with multiple states to model temporal dependencies. Each state is considered a sub-unit and is equipped with emission probabilities or densities over possible observations. The AM provides elemental units with which larger entities such as words and sentences can be constructed by concatenation. The basic principle of HMM AMs is to approximate $P(X|W)$ by the concatenation of models in a maximum-likelihood fashion.

The subword based modeling approach, compared to a higher-level modeling scheme, has several advantages:

- Precision: An acoustic unit is specific to it's articulation, i.e, elements of the sound inventory is clearly distinguishable from each other, given appropriate approximations.
- Robustness: Fewer entities require fewer training data, and smaller units require less complex models.
- Modularity: With a finite inventory of acoustic units, one can compose words and sentences of arbitrary length. Ideally, all acts of speech are derivable by proper concatenation of elemental units, which also guarantees scalability.
- Transferability: Previously unseen concepts can be modeled by falling back to elemental units.

For the purpose of acoustic modeling, we make use of the Markov property by computing

$$P(q_{t+1} = j | q_t = i, q_{t-1} = h, \cdots) = P(q_{t+1} = j | q_t = i), \tag{2.8}$$

and

$$a_{ij} = P(q_{t+1} = j | q_t = i), \quad 1 \leq i, j \leq N. \tag{2.9}$$

An HMM can be interpreted as a finite state machine that serves as a generator of vector sequences, where a state $q_t = i$ is changed to $q_{t+1} = j$ once for a particular point t in time, and a feature vector v_t is produced with an emission probability $b_j(v_t)$. The joint probability of a feature vector sequence X and the sequence of visited states S given the HMM λ is calculated as

$$p(X, S | \lambda) = a_{q_0 q_1} \prod_{t=1}^{T} b_{q_t}(x_t) a_{q_t q_{t+1}}. \tag{2.10}$$

The three fundamental problems of HMMs are known as the *evaluation* problem, the *decoding* problem and the *optimization* problem [134]. Given an existing HMM, the evaluation problem is the task of computing the probability of how likely the HMM emits a specific observation. The decoding problem describes how to compute the most probable sequence of visited states to generate a specific observation. The optimization problem is also known as the learning problem and is the task of re-estimating the parameters for a new HMM that emits the given observation with a higher probability than the initial HMM. The main concern of HMM AM training is the optimization problem.

The Optimization Problem

The optimization problem raises the question how to adjust the HMM model parameters (S, π, A, V, B) so that $P(O|\lambda)$ will be maximized. HMMs can be optimized iteratively so that for every point i in time $Q(\lambda_{i+1}) > Q(\lambda_i)$, where Q is a pre-defined optimization function. The predominant training strategy is to maximize the observation probability of the training data, which corresponds with the evaluation problem for HMMs.

Formally, the optimization problem is to find a λ' with

$$p(X|\lambda') > p(X|\lambda) \quad \text{, with given } \lambda, X = x_1, \cdots, x_T. \tag{2.11}$$

There is no known way to analytically solve this training problem, i.e., given any finite observation sequence as training data, there is no optimal way to estimate the model parameters [134]. It is however possible to choose model parameters that locally maximize the probabilities. With the Baum-Welch rules [11, 10] and the Expectation-Maximization (EM) algorithm [31] at hand there exist methods for iterative parameter optimization.

The primary task of training is to optimize all parameters of a state s_i. For that, knowledge about the probability of being in a particular state s_i at time t when making the observation $x_1, \cdots, x_T$ is required, which is defined as

$$\gamma_t(i) = P(q_t = i|X, \lambda) = \frac{P(q_t = i, X|\lambda)}{P(X|\lambda)}. \tag{2.12}$$

The numerator of this term is computed by the Forward-Backward algorithm, which is used to solve the evaluation problem. The probability of being in state s_i at time t and making the full observation X can be described as

$$P(q_t = i, X|\lambda) = P(q_t = i, x_1, \cdots, x_t|\lambda) \cdot P(x_{t+1}, \cdots, x_T|q_t = i, \lambda) = \alpha_t(i) \cdot \beta_t(i), \tag{2.13}$$

where $\alpha_t(i)$ is the probability of being in state s_i after having seen the partial observation $x_1, \cdots, x_t$, and $\beta_t(i)$ is the probability of being in state s_i and making the future partial observation $x_{t+1}, \cdots, x_T$. This implies

$$\gamma_t(i) = \frac{P(q_t = i, X|\lambda)}{P(X|\lambda)} = \frac{\alpha_t(i) \cdot \beta_t(i)}{\sum_j \alpha_t(i) \cdot \beta_t(i)}. \tag{2.14}$$

According to this formulation it is sufficient for the training algorithm to know the observation $X = x_1, \cdots, x_T$ and the corresponding $\gamma_t(i)$ to optimize the emission probabilities of an HMM.

The probability of a transition from s_i to s_j when observing X is defined as

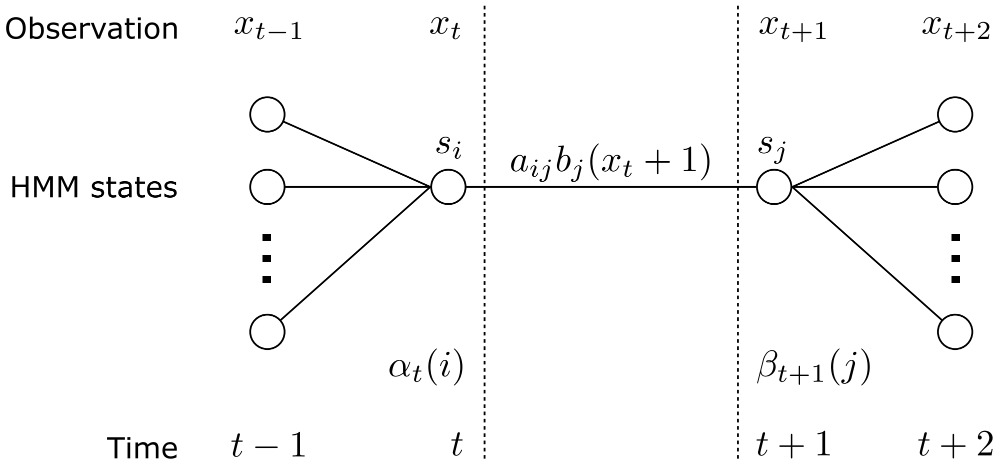


Figure 2.4: Graphical representation of the forward-backward probability computation.

$$\xi_t(i,j) = P(q_t = i, q_{t+1} = j | X, \lambda) \tag{2.15}$$

$$= \frac{P(q_t = i, q_{t+1} = j, X | \lambda)}{P(X|\lambda)} \tag{2.16}$$

$$= \frac{\alpha_t(i) a_{ij} b_j(x_t + 1) \beta_{t+1}(j)}{\sum_l \alpha_t(l) \beta_t(l)}. \tag{2.17}$$

Figure 2.4 provides a graphical representation of the forward-backward probability, i.e., the term in the numerator in Equation (2.17).

By applying the Bayes rule and decomposition by utilization of the α and β terms, this probability can be expressed as

With α, β, γ and ξ at hand, the Baum-Welch rules can be applied for parameter optimization:

$$a'_{i,j} = \frac{\sum_{t=1}^{T} \xi_t(i,j)}{\gamma_t(i)} \tag{2.18}$$

is the updated probability for a transition from s_i to s_j, and

$$\pi'(i) = \gamma_1(i) \tag{2.19}$$

is the updated probability of s_i being the initial state of the HMM.

The update step of the emission probabilities for each state depend on the nature of the emission probability models. In the continuous case, i.e., when using Gaussian mixture models, the EM algorithm is used. In the discrete case, the Baum-Welch rule

$$b'_i(v_k) = \frac{\sum_{t=1}^{T} \gamma_t(i)\delta(x_t, v_k)}{\sum_{t=1}^{T} \gamma_t(i)} \quad \text{, with } \delta(x_t, v_k) = \begin{cases} 0 & \text{for } x_t \neq v_k \\ 1 & \text{for } x_t = v_k \end{cases} \tag{2.20}$$

is applicable. In the case of emission probabilities modeled by neural nets, the Back-Propagation algorithm would be a common choice.

2.3.2 Model Initialization

Several strategies exist for initializing acoustic model training, depending on the available resources. The three common basic approaches are random initialization, initialization by parameter transfer, and initialization with labels.

Following the theoretical formulation of the Baum-Welch rules and the EM algorithm initialization with particular values for parameters is not required. By definition, HMM training converges to a local optimum with every optimization step, in strict accordance with mathematical correctness. It is however recommended to choose some start values that pose a good starting point. There are mainly two reasons for this: The Baum-Welch update rules only guarantee the convergence to a local optimum, and an unfavorable parameter initialization may lead to very long optimization cycles. A popular way to initialize parameters is with a so called flat start, where observations are assigned to HMM states by distributing them evenly in succession.

Parameter transfer from an existing model to a new model is a form of bootstrapping. The complexity of a transfer depends on the divergence between the source and target system. A transfer should be easier if the model structure is similar. If both models differ significantly, certain parameters might have to be discarded or modified to fit the new model, if possible.

Initialization with labels is a popular high-level bootstrapping method, where labels for the training data are typically produced by an existing system for the target language. Labels are assignments of observations to acoustic units which can be used to initialize the model of a new system. Automatic labels are generated by forced-alignment of recordings with matching word based transcriptions. Such transcriptions usually hold a certain level of detail about other speech and non-speech events as well, such as articulatory (smacking, breathing, etc.), linguistic (incomplete words, repetitions, etc.) and environmental noises (background noise, etc.). Forced-alignment is nothing more than applying the Forward-Backward or Viterbi algorithm [165, 46] on the transcribed training data using existing models. Automatic alignments are error-prone or might require mappings from one model space to the other, but are typically still good enough for initializing a new model.

2.3.3 Iterative Optimization

The acoustic model training usually is an iterative process. Phases of parameter estimation and label writing alternate until convergence or until a stop criterion is met, such as a flattened out performance curve on a validation set. The iterative process also involves gradual increase of model complexity, for instance by switching from context independent to context dependent acoustic units or by increasing the mixture size of the emission probability distributions.

The Forward-Backward algorithm computes the probabilities $\gamma_t(i) = P(q_t = i|X, \lambda)$. This allows a training observation to be assigned to multiple HMM states at the same time. One downside of the Forward-Backward algorithm is the increased computational complexity. It is common practice to use the Viterbi algorithm instead, which computes only the most probable sequence of visited states:

$$Q = q_1, \cdots, q_T = \operatorname*{argmax}_Q P(Q|X, \lambda). \tag{2.21}$$

Consequently, the probabilities $\gamma_t(i)$ are approximated by

$$\gamma_t(i) = \begin{cases} 0 & \text{for } i \neq q_t \\ 1 & \text{for } i = q_t. \end{cases} \tag{2.22}$$

The derivation of EM training for HMM parameter optimization is known as Viterbi training, which uses the Baum-Welch rules with constraints

$$\gamma_t(i) = \delta(q_t, s_i) \text{ and } \xi_t(i, j) = \delta(q_t, s_i)\delta(q_{t+1}, s_j). \tag{2.23}$$

With increasing T, both algorithms result in an almost equally effective training, with Viterbi training holding the advantage of much faster training and easier application of search space restrictions.

2.4 Language Modeling

The language model $P(W)$ is introducing linguistic constraints into the speech recognition process. Linguistic constraints are an important factor in recognizing speech, as the ambiguity of the acoustic signal, or the stream of digitized observations, is too big, which makes precise recognition a very difficult task. Language models are estimated from large text corpora written in the target language, and typically reflecting the target domain. If a speech recognizer is used predominantly in the tourism sector, the LM would be trained on tourism related text data. Optimally $P(W)$ will give a relatively high probability to likely word sequences, and a relatively low probability to unlikely or grammatically wrong word sequences.

The joint probability of a sequence of words $(w_1, ..., w_M)$ is defined as

$$P(W) = \prod_{i=1}^{M} P(w_i|w_1, \dots, w_{i-1}), \tag{2.24}$$

i.e., for computing the probability of the current word, we consider the context of the word by going back in time. The issue of this formulation is that seeing the exact same sequence of words is very unlikely if the size of the context remains unconstrained. There would simply be no corpus of sufficient size to model all possible word sequences in a particular language or even just a particular domain. The most conventional type of LM estimates word sequence probabilities with the help of n-grams. An n-gram is a count for the occurrence of a specific sequence of words that has at most a length of n. In other words, n-gram LMs condition the probability of a word on at most $(n-1)$ words, instead of considering the entire context, i.e.,

$$P(W) \approx \prod_{i=1}^{M} P(w_i|w_{i-n+1}, \dots, w_{i-1}), \tag{2.25}$$

where the conditional word probabilities are computed with the help of n-gram counts

$$P(w_i|w_{i-n+1},\ldots,w_{i-1}) = \frac{\#(w_{i-n+1},\ldots,w_i)}{\#(w_{i-n+1},\ldots,w_{i-1})}. \tag{2.26}$$

Even with a comparatively small value for n, data sparseness remains an issue. Typical values for n range from 2 to 5, where the larger values would require significantly more data to provide reliable probability estimates. To mitigate the inevitable data sparseness issue, back-off strategies are used, where the likelihood computation falls back to lower-order n-grams if higher-order n-grams are unavailable [174, 87]. Smoothing techniques are used to free up a small portion of the probability mass for unseen words [55, 24]. This is done by discounting probabilities for seen word sequences.

2.5 Levels of Supervision

Acoustic models, much like any classifier, in practice are trained with varying levels of supervision, depending on the available data and on the task to solve. A multitude of raining strategies that rely on different amounts of labeled training data, on different types of labels or on varying sources. The general scope of machine learning approaches ranges from fully supervised methods, where the entirety training data is usually labeled precisely with non to only few errors, to fully unsupervised methods where no labels of any kind are available a priori. The range of learning paradigms is illustrated by Figure 2.5. Terms used to describe the different scenarios between the two extremes are not used consistently throughout literature, the following overview is therefore subject to some ambiguity.

2.5.1 Supervised Learning

Supervised learning is the learning from data which is properly labeled in its entirety. Most high-performance ASR systems are largely trained with full supervision, which makes their development typically expensive and time consuming, since the producing sufficiently accurate transcriptions requires human workers with expertise. The objective of supervised training is to maximize the probability that the system's models hypothesize the a priori known reference, or ground truth.

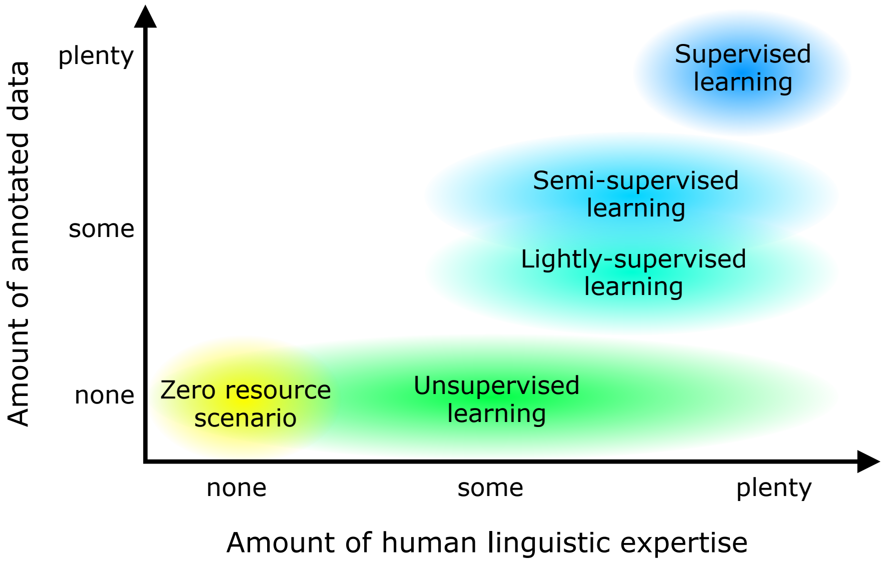


Figure 2.5: The zero resource scenario in the context of levels of supervision. The amount of annotated data refers to the amount of utilized data in the target language. The amount of human linguistic expertise refers to the amount of utilized expert knowledge such as phone set definitions or hand crafted pronunciation dictionaries.

2.5.2 Semi-supervised Learning

Semi-supervised training is used in cases, where references are only available for a subset training data and the remainder of the data is without references. Often, the untranscribed portion of the data is many times larger than the transcribed one. Whereas manual data transcription is usually very expensive, unlabeled data is often available in much higher quantities, especially since the advent of the social web and massive digital online archives. Semi-supervised learning is a form of inductive learning or *self-training* [21]. Models that were trained on a subset of annotated data are used to infer automatic transcriptions of previously untranscribed data, which is then added into the training process. The objective is to make the best possible automatic prediction of what was said in the untranscribed recordings.

2.5.3 Lightly-supervised Learning

Generally, any kind of data that might serve as reference to the training data can be exploited for supervision. In ASR, the most common type of light supervision is inaccurate or fractured transcriptions, which can be useful in combination with methods such as flexible transcription alignment [43]. Related textual data is commonly available on a much larger scale than detailed transcriptions, and loose transcriptions such as closed captions for television broadcasting are produced with considerably less effort. Non-parallel textual corpora may be utilized as constraints on the search space during automatic transcription by training a contextualized language model and dictionary. Multimodal and crossmodal data has also been shown to be a useful source of information in the absence of true labels.

2.5.4 Unsupervised Learning

Methods of unsupervised learning can be used for model training when no labels for the training data are available a priori. The core principle of these methods is to find the latent structure in the unlabeled data and represent them in form of models. In the context of AM training this would be analogous to inferring some kind of label set for the training data without the help of any other supervisedly trained models or systems. One of the challenges lies in the fact that quality estimation is difficult, which on the other hand would be helpful in discarding erroneous data or data which is unsuitable for training. Unsupervised learning often profits from alternating iterations of parameter learning and label re-estimation due to a self-training effect.

2.5.5 Pushing the Limits: The Zero Resource Scenario

The zero resource scenario is a special case of an unsupervised learning problem, where the raw input signal is the only available data to learn from. Unsupervised learning in the context of automatic speech recognition still utilizes a priori knowledge and constraints that are based on such knowledge. For instance, the number and identity of acoustic units for a given target language would usually be known, whereas the assignment of training samples to these acoustic units would be subject to unsupervised learning. In a zero resource scenario, no such prior knowledge is available. The unsupervised learning problem is therefore expanded by the aspect of inferring the model structure and model complexity itself. The zero resource case of machine language acquisition is to a certain extent artificial, since humans do not only rely on audio input only. Human language acquisition is a multimodal process, and as argued in the introductory chapter, humans most likely do not have to start with zero knowledge about the possible forms and appearances of human sounds. One might therefore want to argue that some form of basic information, for instance in form of priors, is sensible to expect as being available. The zero resource scenario as defined here however is still easily justified as an important research challenge, since it provokes the question whether prior category knowledge is actually needed in order to infer sensible, meaningful and robust acoustic units from raw speech only, a question that is also discussed in linguistic circles (see Section 1.4.1.

As discussed in Section 1.3.2, various approaches to inferring acoustic models from raw speech have been explored. To date, the most promising works rely on Bayesian non-parametric modeling techniques such as [95, 22], where Dirichlet process mixture models (DPMMs) are inferred to segment raw speech and to represent the unknown number of classes in the raw data by a dynamically sized set of model components. Bayesian methods such as the DPMM are especially useful in cases where the number of expected classes in data is unknown. This is a frequent case when confronted with unlabeled speech data of a potentially unknown, unfamiliar or unexplored language. Following the arguments in the introductory chapter, this work's proposed solutions to the unsupervised subword modeling problem are built upon a Bayesian non-parametric approach. The following chapter provides an introduction to Bayesian modeling and to the Dirichlet process mixture model in particular.

Chapter 3

The Dirichlet Process Mixture Model

"Der Geist einer Sprache offenbart sich am deutlichsten in ihren unübersetzbaren Worten." [1] *[166]*

– Marie von Ebner-Eschenbach (1830-1916), *Writer*

This chapter introduces the Dirichlet process mixture model, which is the basic model that is used and build upon in this thesis. For understanding the DPMM, the basics of non-parametric Bayesian statistics have to be laid out. The explanations in this Chapter neither focus on discrete, nor on continuous distributions. It shall however be noted that throughout this thesis, heavy use of continuous distributions is made since they are typically used for modeling speech.

3.1 Probabilistic Modeling

Speech as a major example is a complex process that is usually observed with noise, variance and may not be observable in its completeness. Probabilistic modeling is a powerful tool to handle uncertainties like the ones that frequently occur in speech processing and many other machine learning problems.

Let us assume we have a data set $X = \{x_1, \ldots, x_N\}$ that is comprised of N samples. We further assume that these samples are generated from some distri-

[1] "The spirit of a language is most clearly revealed in its untranslatable words."

bution independently. Let $Z = \{z_1, \ldots, z_N\}$ further be a set of latent variables that could for instance determine the membership of samples to classes.

A probabilistic model is then defined as a joint distribution

$$P(x_1, \ldots, x_N, z_1, \ldots, z_N|\theta), \tag{3.1}$$

where θ is a set of parameters that parametrizes the underlying distribution. A common interpretation of this is as a generative model of the data, and the inference of latent variables given the observed data is expressed as

$$P(z_1, \ldots, z_N|x_1, \ldots, x_N, \theta) = \frac{P(x_1, \ldots, x_N, z_1, \ldots, z_N|\theta)}{P(x_1, \ldots, x_N|\theta)}. \tag{3.2}$$

Probabilistic modeling and inference must solve the question of how to estimate the underlying distribution – or more practically the parameters that describe the distribution – that generated the observed data.

The most common way of estimating θ is by maximum likelihood estimation. Here, parameters θ are set to maximize the likelihood of the observed data:

$$\hat{\theta} = \operatorname*{argmax}_{\theta} P(x_1, \ldots, x_N|\theta). \tag{3.3}$$

With a probabilistic model like this, we can do prediction, i.e., we can predict new samples that were not included in the observations used for estimating the model:

$$P(x_{N+1}, z_{N+1}|x_1, \ldots, x_N, \theta). \tag{3.4}$$

We can also classify previously unseen observations:

$$\hat{z}_{N+1} = \operatorname*{argmax}_{z} P(x_{N+1}|\theta_z). \tag{3.5}$$

Estimating the distribution is a *learning* problem, and estimating the latent variables is an *inference*. The ways of estimating them are quite different, as can be seen in Equations (3.3) and (3.5).

Maximum likelihood estimation has two fundamental problems. The first is that by default it does not use any constraints that would prevent the model parameters to take on unreasonable values. This is particularly a problem when data sparseness occurs. Imagine a case where phenomena with a very low probability of occurring were simply never observed and are thus not part of the training data used to estimate the underlying distribution. With maximum likelihood estimation, the probability of observing such a phenomenon would be estimated to be zero, which is unreasonably low and does not reflect the true underlying distribution. The other extreme would be to overestimate the occurrence of a phenomenon that was over-represented in the training data, assigning it a too high probability during maximum likelihood estimation. The theme is the same: The likelihood of the training data will certainly be maximal, but the estimated model might be a poor image of the reality of the true underlying distribution.

The second problem of maximum likelihood estimation is that only a single unique solution of θ is produced, even though we can actually not be certain of the values of θ.

Bayesian modeling handles both problems by doing learning and inference elegantly in the same way. A Bayesian statistical approach does not just use a single estimation of θ but considers the entire distribution over the parameter space of θ, given the data, i.e., $P(\theta|X, Z)$. The learning problem in the Bayesian framework is expressed by the posterior distribution of the parameters and latent variables given the data, which can be decomposed with the help of Bayes's law [12]:

$$P(z_1, \dots, z_N, \theta | x_1, \dots, x_N) = \frac{P(x_1, \dots, x_N, z_1, \dots, z_N | \theta) P(\theta)}{P(x_1, \dots, x_N)}. \tag{3.6}$$

One can see that learning and inference is done in one step. We can calculate the likelihood of the data given the parameters of the distribution as

$$P(x_1, \dots, x_N, z_1, \dots, z_N | \theta) = \prod_{i=1}^{N} P(x_i, z_i = k | \theta). \tag{3.7}$$

The denominator contains the normalization term

$$P(x_1, \dots, x_N) = \int P(x_1, \dots, x_N, z_1, \dots, z_N | \theta) P(\theta) d\theta, \tag{3.8}$$

which is simply the likelihood of the data given all possible parameter values. Calculating this term is not trivial since calculating the integral over a possibly infinite number of distributions is unfeasible. One solution to this problem can be the use of conjugate priors for that distribution, which will be discussed in the following Section.

The formulation according to Equation (3.6) requires a prior distribution over parameters $P(\theta)$, which in the Bayesian framework is an efficient component to regularize learning. The prior probability over parameters can be set according to a certain prior belief about the likelihood of specific parameter values.

Coming back to the two problems of maximum likelihood estimation, $P(\theta)$ is a powerful tool to avoid disadvantageous parameter values. With the prior, we can assign low probabilities to parameter values that seem unreasonable, and higher probabilities to values that are likely or expected to reasonably represent the underlying distribution of a data set, according to the prior knowledge that we might have about said distribution.

In the case of speech data, X might represent the occurrence of acoustic phenomena, i.e., realizations of sounds, and Z might be labels that identify the sounds. We would like to model the probabilities such that all actually possible sounds have a probability larger than zero. We would further want to assign higher probability to sounds that are more likely to occur than others. For instance, English phones are more likely to occur in an English sentence than elements of very unrelated phonetic inventories. We would also want to give higher probabilities to – according to our prior knowledge – the few most frequently occurring sounds, and low probabilities to the larger group of less commonly occurring ones.

With Bayesian modeling, prediction is now expressed as

$$P(x_{N+1}|x_1, \ldots, x_N) = \int P(x_{N+1}|\theta) P(\theta|x_1, \ldots, x_N) d\theta, \tag{3.9}$$

and analogous to the above, classification takes the form

$$P(x_{N+1}|x_1, \ldots, x_N) = \int P(x_{N+1}|\theta_z) P(\theta_z|x_1, \ldots, x_N) d\theta_z. \tag{3.10}$$

3.2 Stochastic Processes

The normalization term in Equation (3.8) in its current form is typically impossible to compute. One way to enable a feasible calculation is by the use of conjugate priors for the distribution that is subject to modeling. A conjugate prior has the

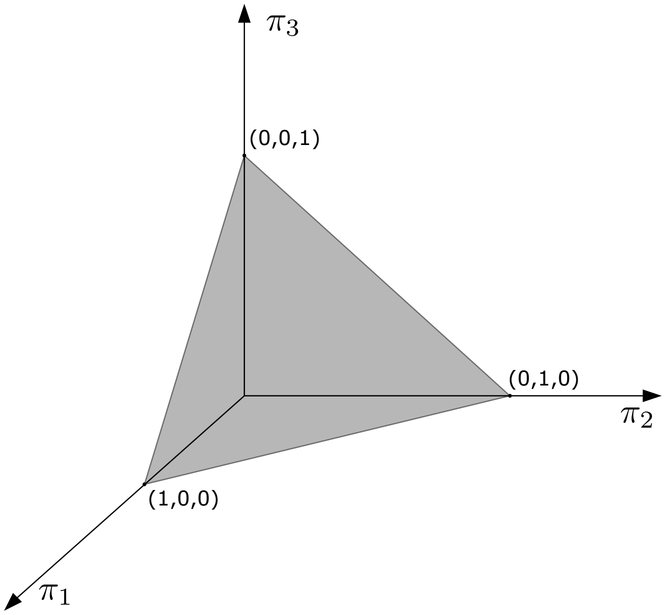


Figure 3.1: Illustration of a probability simplex with $K = 3$.

property that the product of the prior probability with the likelihood is of the same form as the prior itself. This property is very convenient because it allows for analytical calculation of the normalization term, without having to solve the integral. A whole array of probability distributions exists that have with conjugate priors and are therefore commonly used in Bayesian frameworks [42].

3.2.1 The Dirichlet Distribution

The multinomial distribution, which is of relevance in this thesis, has a conjugate prior that is defined by the Dirichlet distribution. Let K be the dimension of the multinomial distribution. The K-dimensional Dirichlet distribution is defined over the probability simplex $\Delta_K = (\pi_1, \ldots, \pi_K)$. The π_i are probabilities with

$$0 \leq \pi_i \leq 1 \qquad \forall i \in [1, N] \tag{3.11}$$

and

$$\sum_{i=1}^{N} \pi_i = 1. \tag{3.12}$$

We then say that $(\pi_1, \dots, \pi_K)$ is *Dirichlet distributed*, or

$$(\pi_1, \dots, \pi_K) \sim \mathrm{Dir}(\alpha_1, \dots, \alpha_K), \tag{3.13}$$

where the parameters $\alpha = (\alpha_1, \dots, \alpha_K)$ are proportional to the expected probabilities of elements in π. The Dirichlet distribution is of the form

$$P(\pi, \alpha) = \frac{1}{B} \prod_{k=1}^{K} \pi_k^{\alpha_k - 1} = \frac{\Gamma(\alpha_0)}{\prod_{k=1}^{K} \Gamma(\alpha_k)} \prod_{k=1}^{K} \pi_k^{\alpha_k - 1}, \tag{3.14}$$

where $\alpha_0 = \sum_{k=1}^{K} \alpha_k$. The normalization term B is calculated in closed form following [41]. $\Gamma(\cdot)$ is the Gamma function, an extension of the factorial function.

Let $c = (c_1, \dots, c_K)$ be the counts of K possibly observable classes. We can easily confirm that the Dirichlet distribution is conjugate to likelihoods of multinomial distributions by multiplying both:

$$\prod_{k=1}^{K} \pi_k^{c_k} \cdot \frac{1}{B} \prod_{k=1}^{K} \pi_k^{\alpha_k - 1} = \frac{1}{B} \prod_{k=1}^{K} \pi_k^{\alpha_k + c_k - 1} \propto \frac{1}{\hat{B}} \prod_{k=1}^{K} \pi_k^{\hat{\alpha}_k - 1}, \tag{3.15}$$

where we update $\hat{\alpha}_k = \alpha_k + c_k$. Therefore, the product is proportional to a Dirichlet distribution with updated parameters $\hat{\alpha}$ of the prior distribution (and updated normalization term $\hat{B}$).

3.2.2 The Dirichlet Process

The previous section described the Dirichlet distribution, which can be used as a conjugate prior for multinomial distributions. So far, the dimensionality K was assumed to be fixed. One can however think of cases where K is not fixed and/or possibly even infinite. Imagine a probability distribution over all possible speech sounds that a human can theoretically produce. While passing through the process of language acquisition, a limited set of phones that the speaker regularly uses to form speech in his or her native languages takes shape. This set is certainly not infinitely large, but its size depends on the speakers exposure to human languages. Since we can not know how many phones a speaker acquired covering what amount of languages, it might be a good idea to assign even just a small amount of the probability mass to every possible sound that a human might be able to produce.

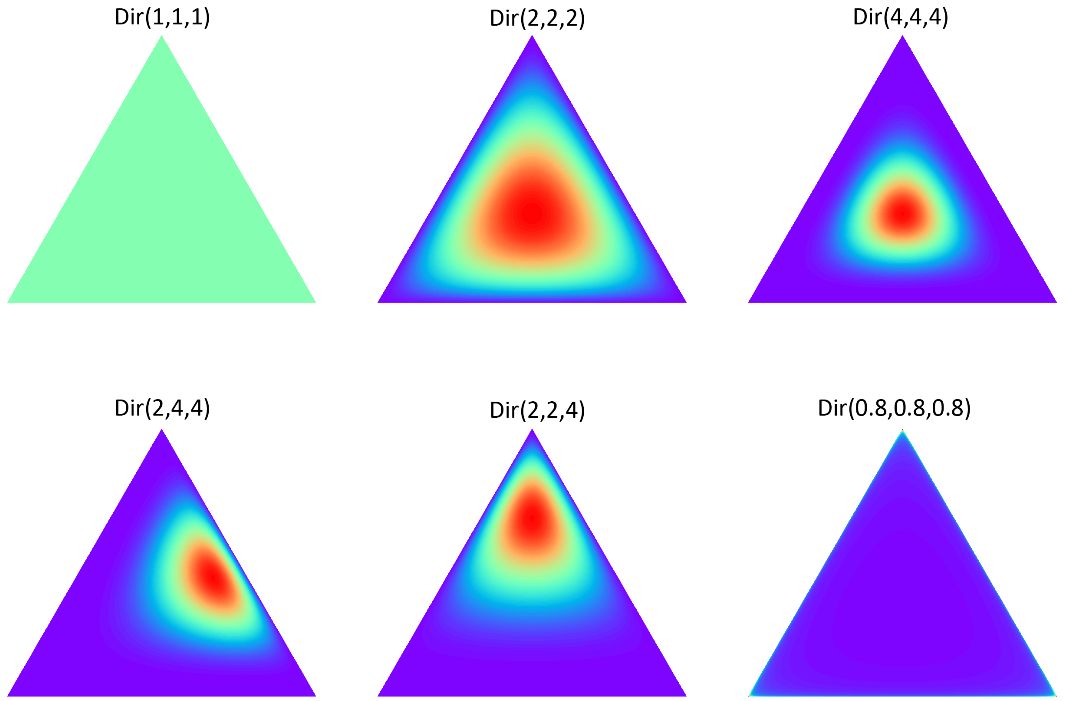


Figure 3.2: Visualization of densities with different Dirichlet priors.

Models that follow this formulation are called *non-parametric*, which refers to the fact that the model complexity is not fixed. The model can theoretically have an infinite number of parameters, because K is not fixed in advance. The Dirichlet process is a framework which helps to model non-parametric distributions. It has been formally defined by [41] as a distribution over probability distributions. The main difference to the standard Dirichlet distribution is the parametrization. Where the Dirichlet distribution is parametrized with a fixed number of parameters $\alpha = (\alpha_1, \ldots, \alpha_K)$, the Dirichlet process uses only a single parameter α_0, called the *concentration parameter*, and the *base measure* or base distribution H.

Let S be a measurable space, and G be a probability distribution over a subset of this measurable space. If G is a distribution over S, then a DP is a distribution over all such distributions. For the standard Dirichlet distribution, the dimensionality K is fixed. The DP however can be used to sample G's that each use a different K. All distributions G are defined over the same, but different subsets of measurable space. We say that G is distributed according to a DP with parameters (α_0, H), or formally

$$G \sim \mathrm{DP}(\alpha_0, H), \tag{3.16}$$

which means that

$$(G(S_1), \ldots, G(S_M)) \sim \mathrm{Dir}(\alpha_0 H(S_1), \ldots, \alpha_0 H(S_M)), \tag{3.17}$$

for every subset $(S_1, \ldots, S_M)$ of S, which means that the probabilities that the G's assign to any subset of S are Dirichlet distributed, according to the Dirichlet distribution parameters $(\alpha_0 H(S_1), \ldots, \alpha_0 H(S_M))$.

Some properties of the DP are important to point out explicitly. The base measure H is the mean of the DP, i.e., $\mathbb{E}[G] = H$, and the concentration parameter α_0 can be considered the inverse variance of the DP:

$$\mathbb{E}[G(S_m)] = H(S_m), \tag{3.18}$$

$$\mathbb{V}[G(S_m)] = \frac{H(S_m)(1 - H(S_m))}{\alpha_0 + 1}, \tag{3.19}$$

which means that on average, distributions drawn from a DP appear like the base measure H. A large α_0 results in a small variance and vice versa. The concentration parameter is also called strength parameter, which refers to the strength of the DP as prior in non-parametric modeling.

The DP is the conjugate prior for arbitrary distributions over the measurable space S. Let $G \sim \mathrm{DP}(\alpha_0, H)$. We can sample independent samples $(z_1, \ldots, z_N) \sim G$, each taking on a value in S, because G is a distribution over a subset of S. Then the posterior distribution over G is a DP as well:

$$G | z_1, \ldots, z_N \sim \mathrm{DP}(\alpha_0 + N, \frac{\alpha_0}{\alpha_0 + N} H + \frac{N}{\alpha_0 + N} \frac{\sum_{i=1}^{N} \delta_{z_i}}{N}). \tag{3.20}$$

As can be seen, the posterior base measure is a weighted average between the prior base measure and the empirical distribution. If $\alpha_0 \to 0$, then the priors influence diminishes and the DP is dominated by the empirical distribution. The same happens with growing number of observations, i.e., $N \gg \alpha_0$. This is known as the consistency property of the DP which says that the posterior DP approaches the true underlying distribution of the data.

The DP has been discovered various times and its model can be derived in different ways as special cases of various stochastic processes [41, 7, 138]. Different representations of the DP lead to different algorithmic solutions for the inference. The following Sections will explain some popular metaphors for the DP, which are the Pólya urn scheme, the Chinese restaurant process and the stick breaking construction.

3.2.3 The Pólya Urn Scheme

The Pólya urn scheme is a metaphor for explaining the posterior probability of the DP. Consider the sampling of distributions and observations

$$G \sim \mathrm{DP}(\alpha_0, H), \tag{3.21}$$

$$(z_1, \dots, z_N) \sim G. \tag{3.22}$$

The conditional distribution for a new observation z_{N+1} could then be expressed as

$$z_{N+1} | z_1, \dots, z_N \sim G_N(z_{N+1}) = \frac{\alpha_0 H(z_{N+1}) + \sum_{i=1}^{N} \delta_{z_{N+1}=z_i}}{\alpha_0 + N}. \tag{3.23}$$

That means the posterior base measure given the observations $(z_1, \dots, z_N)$ is also the posterior predictive distribution of the new observation z_{N+1}. The process of how the observations z_i are sampled is the Pólya urn scheme, or more specifically a generalization of it called Blackwell-MacQueen urn scheme [15].

The analogy reads as follows. Suppose you sample balls x_i with color z_i from an urn G. Each sampled ball then gets replaced in the urn with two copies in same color than the sampled ball. As more and more balls of the same color are sampled, the likelihood of sampling yet another ball of the same color increases, or in other words, the probability is $\propto N$. This is also known as the "rich gets richer" property of the urn scheme. In addition to that, occasionally a ball from a different urn H can be sampled with a probability $\propto \alpha_0$, which has a previously unseen color. The ball is replaced in H and a copy is put into G (note that in the beginning, when G is still empty, z_1 is sampled from H). It is further

$$\lim_{N \to \infty} G_N \to G \sim \mathrm{DP}(\alpha_0, H), \tag{3.24}$$

i.e., if the sampling is continued indefinitely, then G_N will converge to a DP-distributed random distribution G.

The urn scheme has an interesting property, which is the *clustering property.* If we have Equations (3.21) and (3.22), then the variables $(z_1, \dots, z_N)$ can take on $K \leq N$ unique values $(s_1, \dots, s_k)$ of S. The values of $(z_1, \dots, z_N)$ define a partition of the observations $(x_1, \dots, x_N)$ over the values of S, or in other words, x_i is in cluster k if $z_i = s_k$. Phrased as urn and ball analogy this means that balls are grouped by their color. This clustering property is expressed more explicitly in form of the Chinese restaurant process.

3.2.4 The Chinese Restaurant Process

The Chinese restaurant process [2, 128] is another name for the distribution over partitions such as the Pólya urn scheme generates. Its metaphor for the generative process is more explicit. Imagine a Chinese restaurant with a potentially unlimited number of tables and unlimited seating capacity. The first customer that enters the restaurant will sit at the first table. The second customer can decide whether to also sit on the first table, or to pick a new table. Each customer x_i that enters the restaurant can decide whether to take a seat at any of the existing tables $T = (t_1, \dots, t_K)$, i.e., tables at which at least one other customer already took a seat, or to sit at a new table t_{K+1}. The probabilities of each of the possible actions are:

$$\begin{aligned} P(t_k) &= \frac{c_{t_k}}{\alpha_0 + \sum_{m=1}^{K} c_{t_m}}, && (3.25) \\ P(t_{K+1}) &= \frac{\alpha_0}{\alpha_0 + \sum_{m=1}^{K} c_{t_m}}. && (3.26) \end{aligned}$$

with c_{t_k} being the number of customers that sit at table t_k. All customers at a particular table share the same type of dish, and the customer that picks a new table will eat a dish that has not been served to any other table before. This behavior is exactly what Equation (3.23) describes. The assignment of customers to tables defines a partition, where the dish at the table identifies the cluster, and all customers that share a table (that is, meal) are members of the same cluster.

From Equations (3.25) and (3.26) it can be seen that α_0 governs the number of clusters, and that the number follows a logarithmic growth. The larger α_0 is,

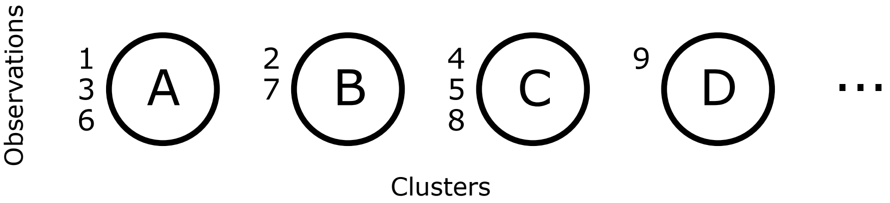


Figure 3.3: Illustration of the Chinese restaurant process. Clusters (“tables”) are gradually generated during the course of the process. Observations (“customers”) are assigned to clusters in order of their generation (their “entering of the restaurant”).

the more clusters are expected. This property is one of the reasons why DPs are interesting as priors on the components of mixture models, a concept that is known as the Dirichlet process mixture model (DPMM). Before we come to DPMMs in Section 3.3, one more important metaphor for the DP called the stick-breaking process shall be explained.

3.2.5 The Stick-breaking Process

The metaphor of the stick-breaking process or stick-breaking construction [149] helps understand how samples from $G \sim \mathrm{DP}(\alpha_0, H)$ look like. As a reminder, if we have Equations (3.21) and (3.22), then the variables $(z_1, \ldots, z_N)$ can take on $K \leq N$ unique values $(s_1, \ldots, s_k)$ of S. Therefore, another way to express a particular $G(\theta)$ is

$$G(z) = \sum_{k=1}^{\infty} \pi_k \delta_{z=\hat{s}_k}, \tag{3.27}$$

where

$$\hat{s}_k \sim H, \tag{3.28}$$

$$\pi_k = \beta_k \prod_{m=1}^{k-1} (1 - \beta_m) \qquad \text{with } \beta_k \sim \mathrm{Beta}(1, \alpha_0). \tag{3.29}$$

With Equation (3.29), we can generate an infinite sequence of *weights* π_k where $\sum_k \pi_k = 1$. According to the stick-breaking process, a DP is comprised of a weighted sum of point masses. The metaphor goes as follows. Imagine a stick of length 1. A part of the stick can be broken off at length β_1. The

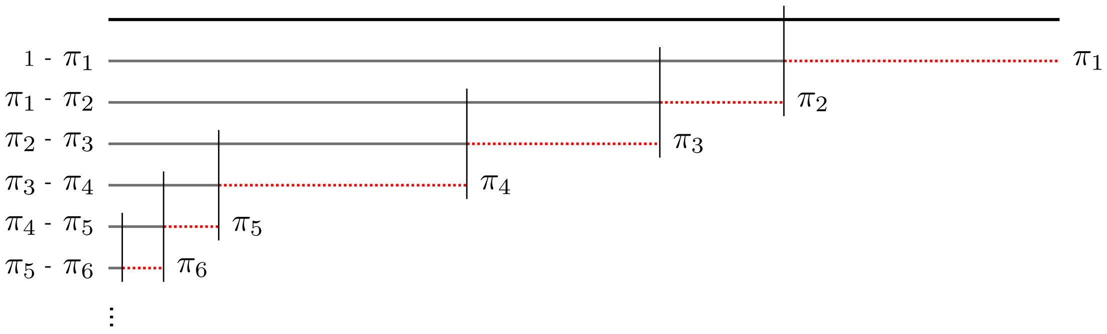


Figure 3.4: Illustration of the stick-breaking process. The sum of all stick lengths π_k (denoted by red dotted lines) is 1, and the π_k are Dirichlet distributed.

length of the broken off part is remembered as π_1. The remaining stick can be broken again, producing more sticks with lengths $(\pi_2, \ldots, \pi_K)$ as $K \to \infty$. As the stick-breaking continues, the stick lengths will become smaller and smaller. The concentration parameter α_0 regulates how the stick lengths are distributed. The larger the value, the flatter the distribution will be, which is also expressed by the expectation $\mathbb{E}[\beta_k] = \frac{1}{1+\alpha_0}$.

A set of weights π is then distributed according to a Griffiths-Engen-McCloskey (GEM) process [129], i.e.,

$$\pi \sim \mathrm{GEM}(\alpha_0), \tag{3.30}$$

which is also known as the stick-breaking distribution. Its simplicity has made the stick-breaking construction a popular concept to design efficient inference methods for the DP [76]. The inference of a DP plays an important role in solving the clustering problem.

3.3 The Dirichlet Process Mixture Model

The most common application of the DP is in fact clustering data by inferring a DPMM [115, 138]. The DPMM uses a DP-distributed (discrete) random measure as prior over the parameters of its mixture components. Formally, the basic DPMM is defined as

$$G \sim \mathrm{DP}(\alpha_0, H), \tag{3.31}$$

$$\theta_i | G \sim G, \tag{3.32}$$

$$x_i | \theta_i \sim F(\theta_i), \tag{3.33}$$

where the θ_i are the parameters of the mixture component that x_i belongs to. x_i has the distribution $F(\theta_i)$. G is the unknown distribution over the parameters, sampled from a DP. Multiple θ_i's can have the same value, and all x_i with the same value θ_i are considered to be in the same cluster. The θ_i can be one parameter or multiple parameters, such as in the case of Gaussian mixtures, where the parameters would be the mean and the covariance of a mixture component.

With the stick-breaking representation, we can express G according to Equation (3.27), and the DPMM can be defined more intuitively and as an alternative to above as

$$\pi | \alpha_0 \sim \mathrm{GEM}(\alpha_0), \tag{3.34}$$

$$\theta_k | H \sim H, \tag{3.35}$$

$$z_i | \pi \sim \mathrm{Discrete}(\pi), \tag{3.36}$$

$$x_i | z_i, \theta_{z_i} \sim F(\theta_{z_i}), \tag{3.37}$$

where z_i is a variable that assigns an observation x_i to a cluster k with probability π_k. The θ_k are now the parameters that model the cluster k, $F(\theta_k)$ is the distribution over the data in cluster k, and π_k is the mixing proportion of cluster k within the mixture. Lastly, the base measure H is the prior over the cluster parameters. As can be seen, the DPMM is a mixture model with a theoretically infinite number of mixture components. Inferring a DPMM is a popular method to cluster data when the number of classes is not known a priori. The notion of an *infinite* mixture model does however not mean that the number of clusters would grow infinitely large. The π_k decrease exponentially, so that the expected number of components is logarithmic in the number of observations. Inference is generally done using Markov chain Monte Carlo (MCMC) sampling algorithms, such as Gibbs sampling [52] (see Section 3.5).

3.4 The Dirichlet Process Gaussian Mixture Model

The Dirichlet process Gaussian mixture model is the infinite extension of finite GMMs, enriched by the aspect of automatic model selection. Let $X = \{x_1, \dots, x_n\}$ be a set of observations. The generative process of X given a DPGMM is as follows:

- Mixing weights $\pi = \{\pi_1, \dots, \pi_k\}$ are generated by a stick-breaking process
- GMM parameters $\theta = \{\theta_1, \dots, \theta_k\}$ are generated according to a Normal-inverse-Wishart (NIW) distribution $\text{NIW}(m_k, S_k, \kappa_k, \nu_k)$ as prior distribution
- A label z_i is assigned to every data point x_i, according to π
- A data point x_i is generated according to the z_i-th GMM component

Now, $\theta_k = \{\mu_k, \Sigma_k\}$ are Gaussian parameters. μ_k is the mean vector, and Σ_k is the covariance matrix of Gaussian component k. The parameter set of the prior NIW distribution consists of a prior m_0 for μ_k, a prior S_0 for Σ_k, the belief-strength κ_0 in m_0 and the belief-strength ν_0 in S_0. Analogous to Equations (3.31) to (3.33), the DPGMM then is defined as

$$\begin{aligned} \pi|\alpha_0 &\sim \text{GEM}(\alpha_0), && (3.38)\\ \theta_k|\lambda &\sim \text{NIW}(\lambda), && (3.39)\\ z_i|\pi &\sim \text{Multi}(\pi), && (3.40)\\ x_i|z_i, \theta_{z_i} &\sim \mathcal{N}(\theta_{z_i}), && (3.41) \end{aligned}$$

where the stick-breaking process is denoted as $\text{GEM}(\cdot)$. $\lambda = \{m_0, S_0, \kappa_0, \nu_0\}$ parametrizes the NIW prior.

3.5 Gibbs Sampling

Until here we described a generative probabilistic process of observations and possible future, yet unseen, data that have some hidden, unobserved structure.

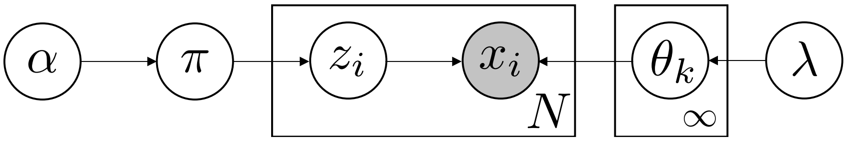


Figure 3.5: The DPMM in plate notation.

Besides the distribution over the model parameters, it is often the hidden structure that is of interest in a Bayesian learning problem. The hidden structure can be inferred by estimating its posterior distribution given the observations. The actual computation of the posterior distribution however is the problem that has to be solved. As the posterior is not available in closed form, an approximation is needed. MCMC methods provide ways of efficient approximation. The general idea is to define a Markov chain on the latent variables which has the posterior distribution as its equilibrium distribution. If samples are drawn for a long enough time from this chain, one eventually draws samples from the real posterior distribution. For DPMMs, a particularly popular choice of an MCMC method is Gibbs sampling [116, 76]. Here, the Markov chain is constructed by considering the conditional distribution of each latent variable given the other variables and the observations.

For explanatory reasons consider a *finite* mixture model with K clusters of which we observed N data points x_i and for which we want to infer the latent cluster assignments z_i. Gibbs sampling will alternatively draw samples from the cluster labels z, the cluster parameters θ and the mixture weights π in an iterative fashion. The conditional posterior distributions of each variable given all other variables are expressed as

$$p(z_i = k|\text{other}) \propto p(z_i = k|\pi)p(x_i|\theta_k) = \pi_k F(x_i|\theta_k), \tag{3.42}$$

$$p(\pi|\text{other}) = p(\pi|z, \alpha_0) = \text{Dir}(n_1 + \frac{\alpha_0}{K}, \dots, n_K + \frac{\alpha_0}{K}), \tag{3.43}$$

$$p(\theta_k|\text{other}) = p(\theta_k|\{x\}_k, \lambda) \propto G_0(\theta_k|\lambda)\mathfrak{L}(\{x\}_k|\theta_k), \tag{3.44}$$

where $F(x_i|\theta_k)$ is the probability that x_i is produced by θ_k, n_k is the number of data points assigned to cluster k, $G_0(\lambda)$ is the base measure parametrized by λ, and $\mathfrak{L}(\{x\}_k|\theta_k)$ is the likelihood of the data currently assigned to cluster k, given the cluster parameters θ_k.

According to Equation (3.42), z_i would be directly sampled from a Dirichlet distribution. This sampling becomes difficult in the case where $K \to \infty$. One can however integrate out π and reformulate the conditional for z_i as

$$p(z_i = k|\text{other}) = \frac{n_{k_{-i}} + \frac{\alpha_0}{K}}{n + \alpha_0 - 1} F(x_i|\theta_k), \tag{3.45}$$

and with $K \to \infty$ the conditional posteriors for $z_i = k$ and $z_i = K + 1$ are expressed as

$$p(z_i = k|\text{other}) = \frac{n_{k_{-i}}}{n + \alpha_0 - 1} F(x_i|\theta_k), \tag{3.46}$$

$$p(z_i = K + 1|\text{other}) = \frac{\alpha_0}{n + \alpha_0 - 1} \int F(x_i|\theta) G_0(\theta|\lambda) d\theta, \tag{3.47}$$

analogous to Equations (3.25) and (3.26).

The Gibbs sampling is presented in detail in Algorithm 1.

Algorithm 1 Gibbs sampling for DPMMs following the CRP representation.

```
Require: $\{\theta_k^{t-1}\}_K$, $\{z_i^{t-1}\}_N$ from previous iteration $t-1$
Ensure: $\{\theta_k^t\}_K$, $\{z_i^t\}_N$
Set $z \leftarrow z^{t-1}$
for $i = 1, \dots, N$ do
    Remove $x_i$ from cluster $z_i$, since a new $z_i$ is going to be sampled
    if Cluster $z_i$ is empty then
        Delete cluster $z_i$, decrease $K$ by 1
    end if
    Draw a new sample for $z_i$ according to

        $p(z_i = k, k \leq K) \propto \frac{n_{k_{-i}}}{n + \alpha_0 - 1} F(x_i|\theta_k^{t-1})$
        $p(z_i = K+1) \propto \frac{\alpha_0}{n + \alpha_0 - 1} \int F(x_i|\theta) G_0(\theta|\lambda) d\theta$

    if $z_i = K+1$ then
        Create cluster $K+1$, increase $K$ by 1
    end if
end for
for $k = 1, \dots, K$ do
    Draw a new sample for $\theta_k^t$ according to

        $\theta_k^t \propto G_0(\theta_k|\lambda) \mathfrak{L}(\{x\}_k^t|\theta_k^{t-1})$

end for
Set $z^t \leftarrow z$
```

Chapter 4

Feature Optimized DPGMM Clustering

"Keine Sprache schreibt, wie sie spricht, sie macht sich Zeichen und Laute selber." [1] *[181]*

– Joseph Stanislaus Zauper (1784-1850), *Premonstratensian*

Systems that tackle speech processing tasks such as ASR utilize methods to improve discriminability by tightly integrating feature optimization techniques. We propose to make use of such methods for supporting DPGMM clustering based unsupervised subword modeling. Features that show improved class discriminability should naturally be beneficial for solving clustering problems in general.

Our approach to improving the quality of DPGMM based speech feature vector clustering is realized by a multi-stage framework. This framework utilizes multiple feature transformations in conjunction to benefit from additive effects. We want to make use of speech feature transformations which are well-established for rich-resource languages to optimize the input features towards discriminability. A wide range of transformations can be applied to features for this purpose, with favorable effects such as dimensional reduction, feature de-correlation or adaptation to certain conditions in order to minimize variability. Because we are situated in a zero-resource scenario, we exploit these transformations in an

[1] "No language writes how it speaks, it makes signs and sounds by itself."

unsupervised fashion.

We utilize a standard pipeline for supervised acoustic model training, where feature transformations are conveniently estimated during the course of the training process to obtain the transformations. The advantage of this is that well-established pipelines already exist and are ready to use. In our case, we specifically decided to utilize the popular Kaldi speech recognition toolkit [132] recipe “s5”, which employs feature transformations for improving class discriminability, reducing variance and counteracting adverse signal properties. The disadvantage of using such pipelines is the requirement of labels for training.

To overcome the issue of not having labels in a zero-resource scenario, we propose using a multi-stage strategy that alternates between feature vector clustering, label generation and transformation estimation via model training. We use the DPGMM sampler to generate initial labels for our untranscribed data by clustering standard feature vectors. These automatic labels serve as basis for feature transformation estimation by unsupervised acoustic model training. The transformations are then applied to the standard feature vectors prior to a second run of DPGMM based clustering. We explain the individual stages of our framework in detail in the following Subsections. A graphical overview is given in Figure 4.1.

4.1 DPGMM Clustering

Dirichlet process Gaussian mixture models – or infinite Gaussian mixture models – extend their finite counterparts by the aspect of automatic model selection, i.e., the model finds its complexity through inference automatically given the data. Model inference is typically sample based using a Markov chain Monte Carlo (MCMC) scheme such as Gibbs sampling. The DPGMM and its sampling are described in Chapter 3. The actual sampler used here combines a restricted Gibbs sampler with a split/merge sampler in an efficient algorithm for fast parallel processing. For the sampling to be efficient, the standard DPGMM is extended by – or augmented – by sub-clusters – The following section briefly outlines the main mechanics of the utilized model and sampler. For more in-depth informations, please refer to [18].

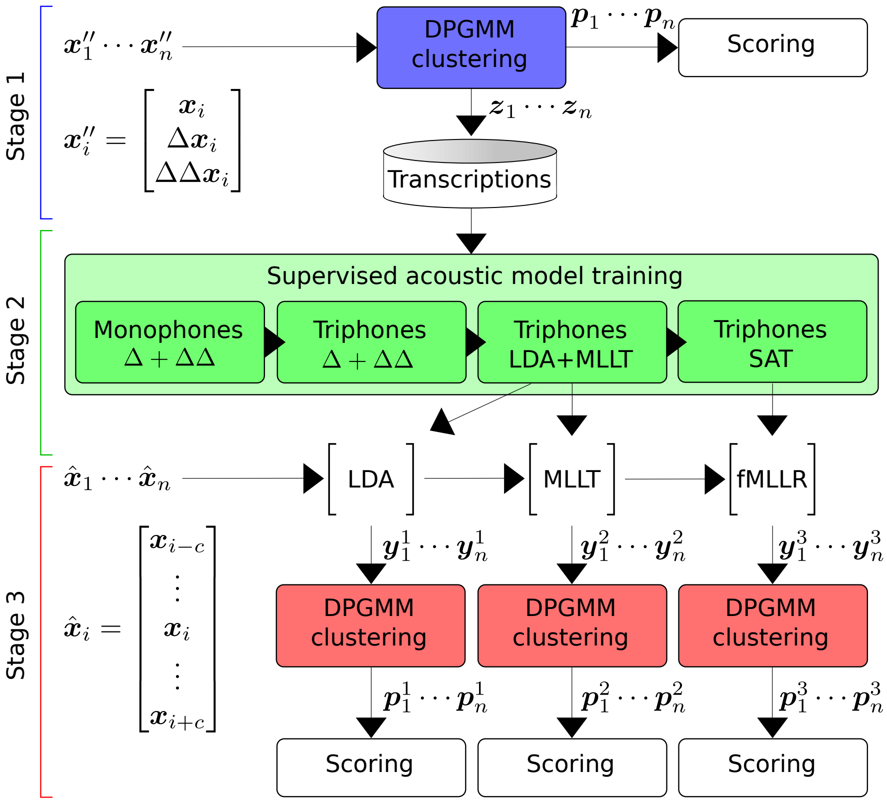

Stage 1
$\boldsymbol{x}_1'' \cdots \boldsymbol{x}_n''$
$\boldsymbol{x}_i'' = \begin{bmatrix} \boldsymbol{x}_i \\ \Delta \boldsymbol{x}_i \\ \Delta\Delta \boldsymbol{x}_i \end{bmatrix}$
DPGMM clustering
$\boldsymbol{p}_1 \cdots \boldsymbol{p}_n$
Scoring
$\boldsymbol{z}_1 \cdots \boldsymbol{z}_n$
Transcriptions
Stage 2
Supervised acoustic model training
Monophones $\Delta + \Delta\Delta$
Triphones $\Delta + \Delta\Delta$
Triphones LDA+MLLT
Triphones SAT
Stage 3
$\hat{\boldsymbol{x}}_1 \cdots \hat{\boldsymbol{x}}_n$
$\hat{\boldsymbol{x}}_i = \begin{bmatrix} \boldsymbol{x}_{i-c} \\ \vdots \\ \boldsymbol{x}_i \\ \vdots \\ \boldsymbol{x}_{i+c} \end{bmatrix}$
LDA
MLLT
fMLLR
$\boldsymbol{y}_1^1 \cdots \boldsymbol{y}_n^1$
$\boldsymbol{y}_1^2 \cdots \boldsymbol{y}_n^2$
$\boldsymbol{y}_1^3 \cdots \boldsymbol{y}_n^3$
DPGMM clustering
DPGMM clustering
DPGMM clustering
$\boldsymbol{p}_1^1 \cdots \boldsymbol{p}_n^1$
$\boldsymbol{p}_1^2 \cdots \boldsymbol{p}_n^2$
$\boldsymbol{p}_1^3 \cdots \boldsymbol{p}_n^3$
Scoring
Scoring
Scoring

Figure 4.1: Scheme of the multi-stage clustering for acoustic unit discovery. In stage 1, standard features are clustered. From frame based class labels, utterance based transcriptions are generated. In stage 2, feature transformations are estimated with the help of an acoustic model training pipeline and the automatic transcriptions. In stage 3, features are transformed with one or more transformations before clustering them by sampling a DPGMM.

4.1.1 Augmented DPGMM

Figure 3.5 is a representation of the general DPGMM in plate notation. Its generative process is explained in Section 3.4. Figure 4.2 shows Chang et al.'s [18] augmented DPMM that uses auxiliary variables to support the split/merge sampling. Each regular cluster is augmented with two explicit sub-clusters, denoted as l for "left" and r for "right". The goal is to design a model that is tailored toward splitting clusters. By picking suitable distributions for these sub-clusters, they can provide good split proposals for their regular parent cluster. Each data point is assigned to either the "left" or "right" sub-cluster with a sub-cluster la-

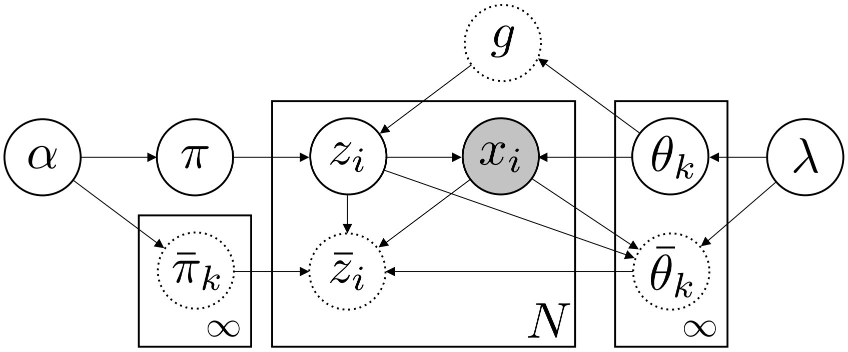


Figure 4.2: Augmented DPMM by Chang et al. [18], utilizing sub-clusters and super-clusters. Auxiliary variables are denoted by dotted circles.

bel $\bar{z}_i \in \{l, r\}$. The naming convention implies that the sub-clusters are designed towards separating the data points into distinct groups within the parent cluster. Sub-clusters have their own weights $\bar{\pi}_k = \{\bar{\pi}_k^l, \bar{\pi}_k^r\}$ and parameters $\bar{\theta}_k = \{\bar{\theta}_k^l, \bar{\theta}_k^r\}$. It is important to note that in this *auxiliary space* the data points x_i generate the labels $\bar{z}_i$, in contrast to the regular space where z_i generate x_i. A super-cluster label g can be used to group clusters, given a similarity measure.

4.1.2 Inference

The parallelizable sampler of [18] alternates between a non-ergodic restricted Gibbs sampler and a split/merge sampler to form an ergodic MCMC sampler which is capable of fast parallel processing.

Restricted Gibbs sampling allows labels z_i to be sampled from a finite set of labels Z. By definition of the DPGMM, the distribution of the mixture weights follows a Dirichlet distribution.

Split/merge sampling performs operations on the existing components. To provide good split candidates, each component is augmented with two sub-clusters with mixing weights $\pi_{k,l}, \pi_{k,r}$ and parameter sets $\theta_{k,l}, \theta_{k,r}$, and each observation of a component is augmented with a sub-cluster label $z_{sub_i} \in l, r$.

The Split/merge sampler proposes split and merge moves in a Metropolis-Hastings fashion. A Hastings ratio H is computed according to the momentary assignment of observations of a component to its sub-clusters, and a move is accepted with a probability $\min(1, H)$. For the merge step, merges of randomly picked components are proposed.

Super-cluster sampling optionally groups similar clusters into super-cluster groups g, given a cluster similarity measure. The merge step of the split/merge sampler can also be conditioned on g to only consider merge candidates within the same super-cluster that the current sample belongs to.

4.2 Unsupervised Audio Segmentation

The data for the challenge's unsupervised subword modeling task is provided without segmentation. Technically, no segmentation is required for our method to work properly. However, we designed an automatic audio segmenter to pre-process long audio recordings for practical reasons. Pre-segmentation guarantees scalability with increasing data size and facilitates the proper termination of our model trainings and decodings. Our segmenter is in essence a silence detector that cuts the audio in the center of places of silence, which results in a lossless segmentation.

The detailed procedure is depicted as flow chart in Figure 4.3. For each recording, we slide over the data with a 0.05 seconds wide sliding window and determine the root mean squared decibel (RMS dB) values of the loudest and the quietest windows as $\mathtt{dB}_{\mathtt{peak}}$ and $\mathtt{dB}_{\mathtt{through}}$. We then calculate an optimistic signal-to-noise ratio (SNR) value as $\widehat{\mathtt{SNR}} = \mathtt{dB}_{\mathtt{peak}} - \mathtt{dB}_{\mathtt{through}}$. We set a threshold for classifying segments in the data as silence so that it lies between $-\widehat{\mathtt{SNR}}$ and the mean RMS dB value $\mathtt{dB}_{\mathtt{lev}}$:

$$\mathtt{dB}_{\mathtt{thresh}} = \frac{\mathtt{dB}_{\mathtt{lev}} + \widehat{\mathtt{SNR}}}{\tau} - \widehat{\mathtt{SNR}}. \tag{4.1}$$

To find segments of silence that are as long as possible, we modify the sensitivity of the threshold by adjusting the parameter τ accordingly. The tool we use for silence detection takes as parameters the silence threshold $\mathtt{dB}_{\mathtt{thresh}}$ and the shortest permitted duration of silence $\mathtt{silLen}_{\mathtt{min}}$. We impose restrictions on the resulting segmentation by defining a desired average segment length $\mathtt{segLen}_{\mathtt{avg}}$ and a maximum segment length $\mathtt{segLen}_{\mathtt{max}}$. We initialize $\mathtt{dB}_{\mathtt{thresh}}$ and $\mathtt{silLen}_{\mathtt{min}}$ and segment the data. If these settings don't lead to a segmentation that meets the requirements, we re-do the segmentation while alternatingly reducing the values of $\mathtt{dB}_{\mathtt{thresh}}$ and $\mathtt{silLen}_{\mathtt{min}}$ until the requirements are met, or until some

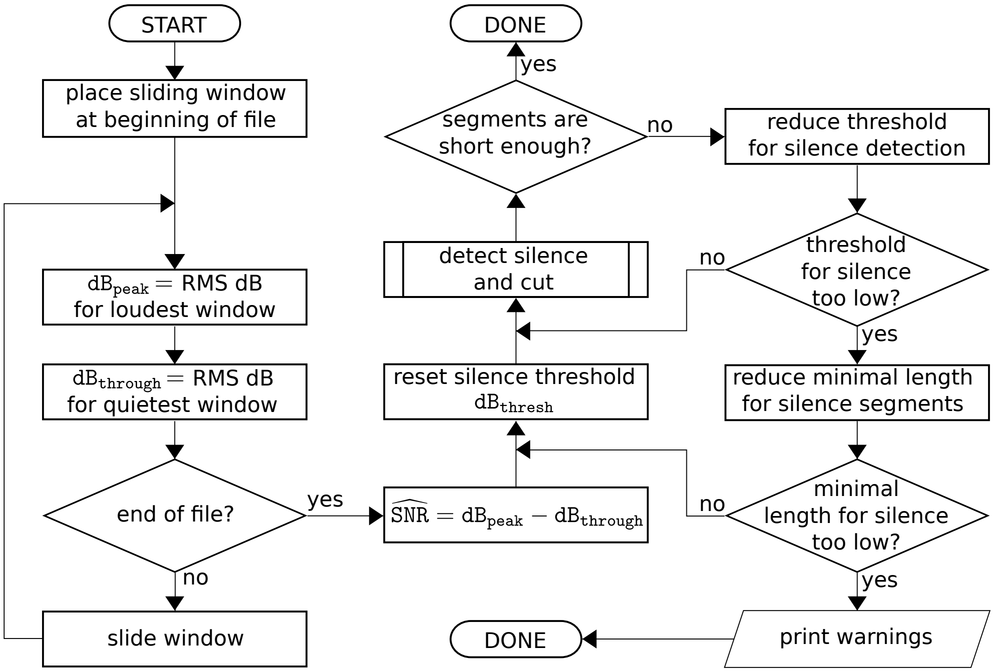

START
place sliding window at beginning of file
$\mathrm{dB}_{\mathrm{peak}}$ = RMS dB for loudest window
$\mathrm{dB}_{\mathrm{through}}$ = RMS dB for quietest window
end of file?
yes
no
slide window
$\widehat{\mathrm{SNR}} = \mathrm{dB}_{\mathrm{peak}} - \mathrm{dB}_{\mathrm{through}}$
reset silence threshold $\mathrm{dB}_{\mathrm{thresh}}$
detect silence and cut
segments are short enough?
yes
DONE
no
reduce threshold for silence detection
threshold for silence too low?
no
yes
reduce minimal length for silence segments
minimal length for silence too low?
no
yes
print warnings
DONE

Figure 4.3: Flow chart of the unsupervised audio segmentation.

minimal values for the parameters are undershot.

4.3 Estimating Feature Transformations

Our framework for unsupervised subword modeling is built around the DPGMM. We utilize the sampler for inferring a DPGMM to automatically generate class labels for unlabeled training data. These labels are used to learn feature transformations for optimizing the input to the DPGMM sampler, which leads to better results in a second DPGMM based clustering. As final step we extract posteriorgrams for test data from the latest DPGMM and use these as new speech representation.

The feature optimizing transformations are unsupervised learned by employing a Kaldi pipeline for acoustic modeling. The argument for this procedure is that the methods used in such pipelines are well established in speech processing. Feature transformations are commonly used to boost the relevant portions of a signal, improve class discriminability, and suppress unwanted channel and speaker variance. For our experiments, we utilize the Kaldi recipe “s5” due to its high popularity particularly in speech recognition setups, which applies the following transformation methods for the respective purposes:

- **Linear discriminant analysis (LDA)** for minimizing intra-class discriminability and maximizing inter-class discriminability of speech features, as well as for dimensional reduction of high-dimensional features spanning larger contexts [44],
- **Maximum likelihood linear transforms (MLLT)** to reduce complexity and to de-correlate features [56, 49],
- **Feature-space maximum likelihood linear regression (fMLLR)** to reduce speaker variability within speech features [4, 48],
- **basis fMLLR** for the same purpose, but in cases the amount of data is insufficient for standard fMLLR [133].

The motivation of using this design in our setup – but also in speech processing in general – is as follows. The LDA transformation is applied early in the pipeline to improve class discriminability and at the same time to reduce the input size by dimensional compression. An arbitrary number of stacked feature vectors can cover any desired temporal context, and dimensional reduction after de-correlation guarantees manageable input vector sizes with acceptable loss of relevant information. As can be seen, there is a trade-off, and therefore the LDA output dimensionality is subject to tuning (in speech recognition, however, some de-facto standard values have been proved to be working well in most scenarios). MLLT is a method used for further de-correlating features given a model, and is applied for instance during the training of GMM-HMM acoustic models. Speaker adaptive methods such as fMLLR are used late in the pipeline to suppress the undesired influence of speaker variance and typically and are estimated given existing models. This particular succession is therefore motivated by practicability and maximizing the positive effects of each step.

Principally, class discriminating properties are critical for clustering methods, and adaptive feature transformations can help to further reduce variability, for instance from various speakers. The above mentioned methods can be applied in conjunction for additive effects. Because we are situated in a zero-resource scenario, we have to estimate these transformations without any prior supervision.

What follows is a guidance through our multi-stage clustering framework that enables us to learn transformations without prior supervision. The explanations are accompanied by a graphical overview given in Figure 4.1.

4.3.1 Automatic Labeling

The DPGMM as a Bayesian non-parametric model has the convenient property to automatically find an optimal number of classes given a set of data during sampling. We use this property and run an initial clustering on standard feature vectors with derivatives (x_i'') to get a set of class labels and the hypothesized class membership z_i for all n speech frames. These classes are simply named with the numeric ID of the Gaussian distribution that most likely produced the respective feature vector.

4.3.2 Transformation Estimation

The output of the previous step is frame-wise class labels for the data. We collapse the labels for each utterance by compressing all subsequent tokens of the same type to a single token, i.e., a sequence of labels 1-1-2-2-2-3-4-4-4 becomes 1-2-3-4. This is done to imitate transcriptions based on phone-like units. We use these transcriptions for transformation estimation by running an out-of-the-box acoustic model training pipeline. We use a 3-state HMM topology with a skip from the first state to the next HMM to guarantee that an alignment is always found, since we cannot guarantee that every label in the transcription covers at least 3 frames. The training is initialized with a flat-start, i.e., by context independent monophone training starting with an equally spaced alignment. Then we subsequently train context dependent triphones, where during training we estimate transformations based on LDA, MLLT and fMLLR.

LDA is a well-known linear transformation that we use to minimize intra-class discriminability and maximize inter-class discriminability of speech features. Estimating the LDA transformation requires the feature vectors themselves and their respective class labels. With our pipeline, we create alignments between utterance HMMs and the automatic textual labels from the previous step and use the HMM states as classes for the LDA.

We compute the LDA for stacked feature vectors ($\hat{x}_i$), where we use a context of c, meaning that we stack the c left and c right feature vectors on top of the current vector, which is the center vector. Context is an important source of information to correctly classify speech features. Feature stacking can cover a much larger context than appending the first and second derivatives, for instance. Dimensional reduction of these high-dimensional vectors is done by omitting lower-ranked coefficients after applying the transformation. Lower dimensional feature vectors encapsulate relevant information more efficiently and help keep the clustering feasible.

MLLT is computed for distributions of speech observations in the HMMs of speech recognizers. The main purpose of MLLT in speech recognition systems is to force the features into a space where diagonal modeling is suitable, which greatly reduces complexity and thus simplifies computing the model parameters [56]. The state-dependent transformations are estimated so that the likelihood of the adaptation data is maximized. Our motivation to use MLLT is to capture correlations between feature vector components.

fMLLR is an algorithm for speaker adaptive training (SAT). The idea of SAT is to capture inter-speaker variability in speaker dependent transformations and to generate speaker independent state distributions instead. Since the transformations are applied in the feature space, the resulting feature vectors are expected to show lower variability across speakers. The transformations are estimated based on alignments with speaker independent features so that the likelihoods are maximized. We apply fMLLR in the zero resource scenario because we expect the transformations to help eliminate variance caused by multiple speakers, which should intuitively aid the clustering process.

4.3.3 Optimized Clustering

We extract the transformations learned in the previous step as transformation matrices, which can be readily applied to the feature vectors $\hat{x}_i$ prior to a second run of DPGMM based clustering. As illustrated in Figure 4.1, we can extract new feature vectors y_i by using one (LDA) up to three transformations (LDA+MLLT+fMLLR) in conjunction. After applying one or more transformations, we perform the frame based DPGMM clustering. We compare the clustering quality using the untransformed features with the clustering quality using each of the transformed features. For that we first extract m sets of GMM posteriorgrams p_i^m for our data given each of the m DPGMMs and then score each of these posteriorgram sets.

4.3.4 Testing

Once the final DPGMMs are sampled in step 3, posteriorgrams p_i can be computed for the test data. We pre-process the input with the same transformations as above. For fMLLR and basis fMLLR, we need to perform a decoding to estimate the transformation parameters. The decoding is done with the existing acoustic model from step 2 and an n-gram language model that has been trained on the automatic labels from step 1. The posteriorgrams are then forwarded to the scoring for evaluating their quality (see Section 4.5).

Basis fMLLR is used for test sets that consist of extremely short recordings. We make the same assumption for the test data as we do for the training data, namely that each recording is uttered by exactly one speaker, and no two recordings come from the same speaker. Standard fMLLR requires a minimum of around 5 seconds of adaptation data per speaker to show positive effects [133]. In cases of extreme data shortage, basis fMLLR can still help to achieve some benefits from speaker adaptation. The proposed framework makes use of the standard fMLLR by default, as shown in experiments described in Section 4.6. We show the positive effect of basis fMLLR in a particular set of experiments that are described in Section 4.7.

4.4 Posteriorgram Combination

In this Section, we describe the method that we developed to combine the output of multiple clusterings. System combination on hypothesis level is a popular method in speech processing to further improve the output quality. Inspired by the idea, we developed a method to combine the output of multiple clusterings on posteriorgram level. The method is formally expressed in Algorithm 2.

The output of each DPGMM m can be represented as a set of posteriorgrams $P_m = \{p_1^m, \ldots, p_n^m\}$ with one posteriorgram for each of the n speech frames (see Equation (4.3)). Generally, combining multiple sets of posteriorgrams $\mathcal{P} = \{P_1, \cdots, P_m\}$ is straightforward. For each frame i, we add together

Algorithm 2 Posteriorgram combination

```
Require: Set $\mathcal{P} = \{P_1, \cdots, P_m\}$ of sets of posteriorgrams
Ensure: Combined set of posteriorgrams $\hat{P}$
 1: $P_{\texttt{tgt}} \leftarrow$ random set from $\mathcal{P}$
 2: $l_{\texttt{tgt}} \leftarrow$ generate labels from posteriorgrams $P_{\texttt{tgt}}$
 3: $\hat{P} \leftarrow P_{\texttt{tgt}}$
 4: for all $P_{\texttt{src}} \in \mathcal{P} \setminus P_{\texttt{tgt}}$ do
 5:     $l_{\texttt{src}} \leftarrow$ generate labels from posteriorgrams $P_{\texttt{src}}$
 6:     $t_{\texttt{cooc}} \leftarrow$ count label co-occurrences in align($l_{\texttt{src}}$, $l_{\texttt{tgt}}$)
 7:     $t_{\texttt{1best}} \leftarrow$ keep 1-best mapping from $t_{\texttt{cooc}}$
 8:     $\hat{P}_{\texttt{src}} \leftarrow \{\}$
 9:     for all $p \in P_{\texttt{src}}$ do
10:         $p_{\texttt{map}} \leftarrow$ map $p$ to space of $P_{\texttt{tgt}}$ using $t_{\texttt{1best}}$
11:         add $p_{\texttt{map}}$ to set $\hat{P}_{\texttt{src}}$
12:     end for
13:     $\hat{P} \leftarrow$ add together pair-wise posteriorgrams in $\hat{P}_{\texttt{src}}$ and $\hat{P}$
14: end for
15: $\hat{P} \leftarrow$ normalize $\hat{P}$
```

the m individual posteriorgrams $\{p_i^1, \dots, p_i^m\}$ (Operation 13) and normalize the new vectors (Operation 15):

$$\hat{p}_i = \frac{1}{m} \sum_{k=1}^{m} p_i^k. \tag{4.2}$$

The result is a new set of posteriorgrams $\hat{P} = \{\hat{p}_1, \dots, \hat{p}_n\}$.

However, for the non-parametric DPGMM, the amount of found classes and thus the dimensionality of posteriorgram vectors differs for each clustering run. Therefore, a mapping between any two sets of posteriorgrams is needed. Given m sets of posteriorgrams, we randomly pick one of these sets as target set $P_{\texttt{tgt}}$ (Operation 1), and consider all other sets as source sets, each denoted as $P_{\texttt{src}}$ (Operation 4).

The mapping from $P_{\texttt{src}}$ to the space of $P_{\texttt{tgt}}$ for any source/target pair works as follows: We first convert all frame-wise posteriorgrams in $P_{\texttt{tgt}}$ into frame-wise labels $l_{\texttt{tgt}}$ by taking the numeric ID of the class with the highest probability as label (Operation 2). Knowing the frame-wise labels, we can represent each

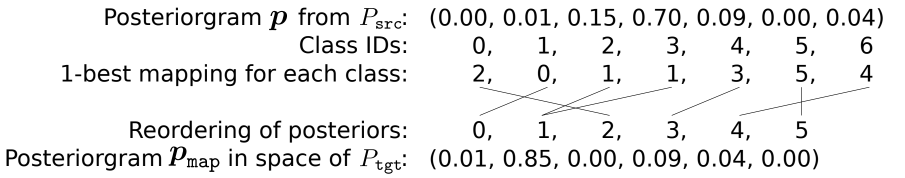

Posteriorgram $\boldsymbol{p}$ from P_{src}: (0.00, 0.01, 0.15, 0.70, 0.09, 0.00, 0.04)
Class IDs: 0, 1, 2, 3, 4, 5, 6
1-best mapping for each class: 2, 0, 1, 1, 3, 5, 4
Reordering of posteriors: 0, 1, 2, 3, 4, 5
Posteriorgram $\boldsymbol{p}_{\mathtt{map}}$ in space of P_{tgt}: (0.01, 0.85, 0.00, 0.09, 0.04, 0.00)

Figure 4.4: Example of mapping one posteriorgram p from the source set $p_{\mathtt{src}}$ to the space of target set $P_{\mathtt{tgt}}$. The 1-best mappings in the mapping table $t_{\mathtt{1best}}$ are used to re-arrange the posteriorgram vector elements to match the posteriorgram vector layout of the target set $P_{\mathtt{tgt}}$. There can be many-to-one mappings in case the posteriorgrams in $P_{\mathtt{src}}$ have higher dimensionality than in $P_{\mathtt{tgt}}$.

utterance in our data as sequences of labels. We do the same given all the posteriorgrams in $P_{\mathtt{src}}$ (Operation 5). For each utterance we now have a pair of label sequences, which we align to count the label co-occuences $t_{\mathtt{cooc}}$ (Operation 6). Given the counts we identify the single most probable “translation” for each class ID, which we keep in a mapping table $t_{\mathtt{1best}}$ (Operation 7). With the mapping table it is possible to re-arrange the posteriorgram vector elements for all $p \in P_{\mathtt{src}}$ to match the posteriorgram vector layout of $P_{\mathtt{tgt}}$ (Operation 10). Note that there can be many-to-one mappings in case the posteriorgrams in $P_{\mathtt{src}}$ have higher dimensionality than the ones in $P_{\mathtt{tgt}}$. For an intuitive example of mapping a single posteriorgram, see Figure 4.4.

4.5 Evaluation Method

The evaluation metric we use to measure the cluster quality is based on the minimal pair ABX phone discriminability between phonemic minimal pairs [141], a method which is related to the ABX task used in psycho-physics [107]. We score GMM posteriorgrams that are computed for each speech frame after clustering, where the posterior probability of the cluster c_k, given an observation x_i is computed as

$$p(c_k|x_i) = \frac{\pi_k \mathcal{N}(x|\theta_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x|\theta_j)}, \tag{4.3}$$

where K is the total number of components in the DPGMM and $p_i = (p(c_1|x_i), \ldots, p(c_K|x_i))$ forms the posteriorgram for observation x_i. θ are the

Gaussian parameters, and π are the mixing weights (see Section 3.4).

Let A and X be two speech representations of sound categories a, B a speech representation of sound category b. The ABX phone discrimination error for categories a and b is

$$c(a,b) = 1 - \frac{1}{|a| \cdot |b| \cdot (|a|-1)} \sum_{A \in a} \sum_{B \in b} \sum_{X \in a \setminus \{A\}} \left(\delta_{d(A,X)<d(B,X)} + \frac{1}{2}\delta_{d(A,X)=d(B,X)}\right), \tag{4.4}$$

where $\delta_{(\cdot)}$ is an indicator function that equals to 1 if the condition $(\cdot)$ holds true and is 0 otherwise, and $d(\cdot,\cdot)$ is the dynamic time warping (DTW) distance defined over sequences of frame based speech representations (in this case posteriorgrams). As in Schatz et al. [141], we use the Kullback-Leibler divergence to compute the DTW distances.

The idea of the ABX test is as follows. Given a phone based reference transcription of the test data, and the posteriorgrams coming from the DPGMM sampler, we can identify sequences of posteriorgrams that represent the same phone-*triplets*, between which we can compute distances. For example, if A and X are two different sequences of posteriorgrams that represent the triplet "b-a-g", and B is another sequence that represents "b-e-g", then the distance between A and X should be smaller than between B and X. If this is not the case, then this counts as a discrimination error. We collect the errors for all possible pairings of central phones. The errors are averaged over all contexts for a given pair of central phones and then over all pairs of central phones. Moreover, we compute the errors within speakers (i.e., the average phone discriminability error for each speaker specific portion of the test data) and across speakers.

The strength of the ABX discriminability task as evaluation method is that the number of classes does not need to be pre-defined. Scores also do not depend on the number of ground-truth classes. As long as the assumed classes show good discriminability from each other, the ABX test will reflect that fact by a low discriminability error. This allows a fair comparison of very different representations of the same data.

4.6 Experimental Evaluation (ZeroSpeech 2015)

4.6.1 The Zero Resource Speech Challenge 2015

In this section we address the unsupervised subword modeling task of ZeroSpeech 2015, where the goal is to find an appropriate and ideally robust speech sounds representation of the underlying language of a dataset [164, 34]. The contributions to the first installment of the challenge in 2015 were diverse. Renshaw et al. [139] apply a correspondence auto-encoder to learn efficient representations with the help of matched word pairs generated by an unsupervised term discovery (UTD) system. Badino et al. [8] make use of a deep auto-encoder that applies a threshold at the encoding layer to generate a binary representation of speech frames. Thiolliere et al. [160] propose a Siamese DNN training framework that takes the frames of UTD word pairs as input and minimizes the distance between frames of the same class and maximizes it between frames of different classes. Chen et al. [22] – performing best in ZeroSpeech 2015 – take a Bayesian non-parametric approach and cluster MFCCs with their derivatives by inferring a Dirichlet process Gaussian mixture model (DPGMM).

4.6.2 Data

The database for all our experiments are the two official data sets of the Interspeech zero resource speech challenge 2015 [164], which greatly vary in size, language and speaking style. One set contains spontaneous, conversational interview-style American English (4h 59min), extracted from the Buckeye corpus [130]. The other set contains carefully uttered, read speech in Xitsonga (2h 29min), a southern African Bantu language. The latter is an excerpt of the NCHLT corpus [30]. All speech segments contain non-overlapping speech of exactly one speaker and are free of non-human noises and pauses. We extract about 1.7M frames for English and 0.8M frames for Xitsonga to cluster.

The references for English and Xitsonga contain 165k and 72k phone-triplet annotations for 39 and 53 unique center phones, respectively. On average, there are 4.2k and 1.3k samples for each English and Xitsonga phone, respectively. This allows reliable discriminability error analyses.

4.6.3 Setup

We use the Kaldi speech recognition toolkit [132] to extract speech feature vectors for a frame length of 25 milliseconds and frame shift of 10 milliseconds. We apply mean variance normalization (MVN) and vocal tract length normalization (VTLN) to all extracted feature vectors.

The used VTLN method is implemented in Kaldi and is of a similar nature than the one presented in [86]. To estimate the transformations, first a universal background model (UBM) is trained using the EM algorithm. We set the size of the UBM to 256 Gaussians and use up to 2M frames for training. The input is pre-segmented via energy based voice activity detection. Given the UBM, for 31 warp factors ranging from 0.85 to 1.15, linear feature transformation matrices are estimated by minimizing the sum-of-squares differences between the original features of some training data and features that are transformed with conventional feature-level VTLN. The latter is a linear warping function similar to the one used in [61] that moves the frequency bins around to ensure that all Mel frequency bins have reasonable widths. Once having learned the feature transformation matrices, warp factors can be estimated on a per-utterance basis.

All AMs used in our framework are likewise trained with Kaldi, following a standard scheme for speaker adaptive training (Kaldi recipe s5). Since we work in a zero resource scenario, all parameters that can be tuned are set to default values. We use the same parameters as Chen et al. [22] during the DPGMM sampling to ensure comparability. The sampling is done for 1500 iterations, and the priors are set so that m_0 is the global mean, S_0 is the global covariance, $\kappa_0 = 1$, and $\alpha = 1$. The value of ν_0 slightly varies and is set to the toolkit's default of $\nu_0 = D + 3$, where D is the dimension of the input feature vectors.

4.6.4 Baseline

The baseline discriminability error rates were produced by clustering 39 dimensional MFCC or PLP vectors with first and second order derivatives (MFCC+Δ+$\Delta\Delta$) with the DPGMM sampler and extracting the GMM posteriorgrams, which is the method of Chen et al. [22] [2].

[2] Despite using the same setup and input feature types, there is a mismatch between the results of Chen et al. [22] and our baseline. We believe this is caused by the fact that Chen et al. reportedly use a custom segmentation of the data, where we use the official segmentation of

For comparison, we computed another set of baseline discriminability error rates by using principal component analysis (PCA) [126, 74] to transform the feature vectors prior to the DPGMM sampling. PCA is an entirely unsupervised method to de-correlate variables with an orthogonal linear transformation and is closely related to LDA, which makes it a fair basis of comparison for the effect of the supervised transformations that we learn without prior supervision. The baseline numbers are found in Table 4.1.

4.6.5 PCA vs. LDA

Figure 4.5 plots discriminability errors of GMM posteriorgrams that were extracted after clustering PCA or LDA transformed feature vectors. The graphs show the performance with regards to the output dimensionality of the transformations, i.e., how many coefficients are used after transforming the features vectors.

Surprisingly, the use of PCA did not show the desired effect of decreasing the discriminability error after DPGMM clustering. In fact, the discriminability error of the GMM posteriorgrams increased on the Xitsonga data. On the English data only little improvement was achieved. The trend is the same whether MFCC or PLP features were transformed.

On the other hand, the LDA transformation produced feature vectors that considerably helped the DPGMM clustering process in finding better clusters, as the discriminability error rates for both data sets decreased greatly, and especially across speakers a strong performance boost is observable. Interestingly, using PLP features for the transformations led to better results than using MFCC features. This is true for both, PCA and LDA transformations.

By using LDA transformed features we already outperform our own baseline and we also beat the numbers of Chen et al. [22]. We take this as proof that the unsupervisedly estimated LDA transformation is the better choice to improve the input to a DPGMM sampler, even when the labels that are used for the estimation are imperfect. The class-discriminating properties of LDA are much more valuable than the simple orthogonalizing that the class-unaware PCA can provide.

the zero resource challenge 2015. Different segmentations can considerably affect the amount of data actually being used for training.

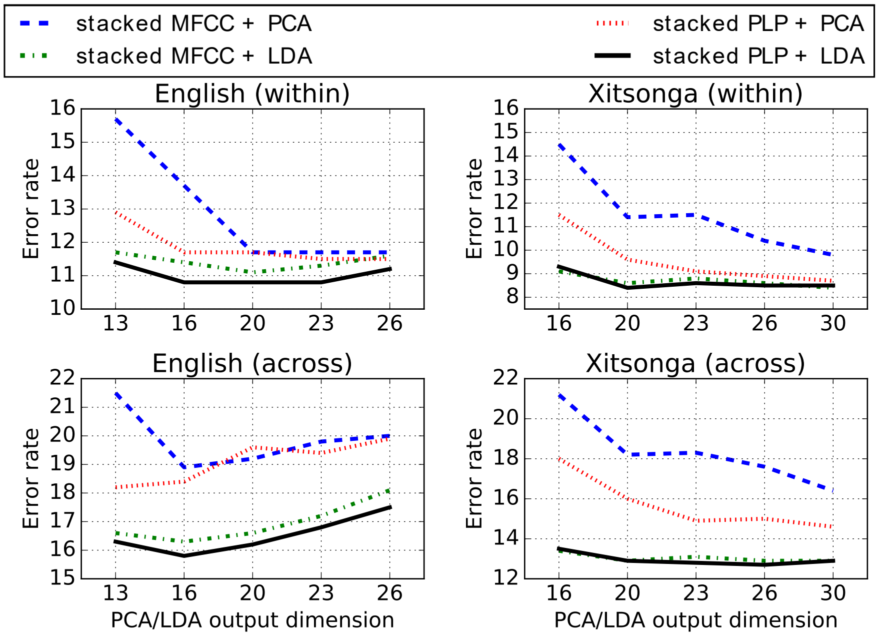


Figure 4.5: Discriminability error rates within and across speakers for DPGMM posteriorgrams after clustering PCA or LDA transformed MFCC or PLP feature vectors. The stacking context size is fixed to $c = 4$. The results are plotted as a function of the output dimensionality d of the transformations. *Left:* Error rates for English. *Right:* Error rates for Xitsonga.

4.6.6 Input and Output Dimensions for LDA

The experiments explained above already show that the choice of the output dimensionality d of transformations influences the clustering performance. We exemplarily conducted a grid search on the parameters c and d LDA transformation of PLP features to find out if this is also true for the input dimensionality. The results of these experiments are visualized in Figure 4.7.

The graphs suggest that the impact of the LDA transformation does not depend on the stacking context size c. In our experiments, any context $c > 2$ was suitable. It seems the largest benefit of the dimensional reduction by LDA transformation lies in the compression of the de-correlated features and not so much in the coverage of a larger context.

We see that $d \leq 20$ works well for the English data, and best results for Xitsonga are achieved with $d \geq 20$. We believe this might be due to the data sets'

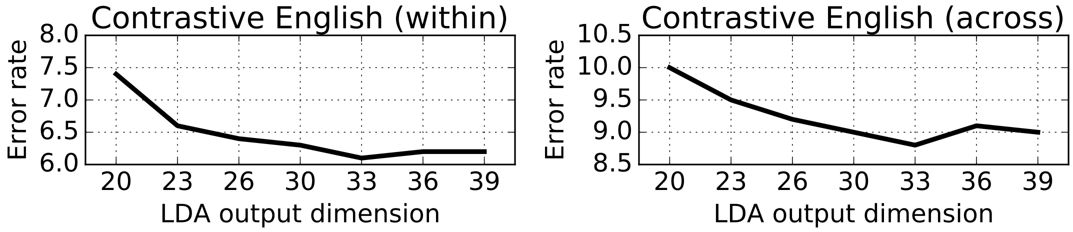


Figure 4.6: Discriminability error rates for the contrastive English data set for DPGMM posteriorgrams clustering LDA transformed PLP feature vectors, plotted as a function of the output dimensionality d of the transformations.

differing speech quality. The English data set consists of conversational speech, and mapping into a lower dimensional space might lead to more stable features for the clustering. The Xitsonga data is read speech, thus the higher dimensions of the transformed feature vectors might still contain distinctive informations.

We conducted additional experiments with a contrastive English data set [34] where we used 38 hours of very clean English read speech to estimate the LDA transformation. Figure 4.6 shows the error rates on this data set as a function of d. The error curve flattens out in a similar range of values than observed for the Xitsonga data set, which shows comparable speech quality. These results might indicate that d is mainly affected by the quality of the speech.

In a real zero resource scenario we don't have the option to tune c and d. One could therefore try and make an informed guess, or more reasonably use values that have been shown to work well for known languages. We take the latter approach and fix the context to $c = 4$ (i.e., we stack 9 feature vectors) for the input to the LDA, and set the output dimensionality to $d = 20$ (i.e., we keep the first 20 coefficients), according to the best overall performance on our English data set. By using the parameters we tuned on English, we achieve a performance on Xitsonga that is only slightly lower than the performance that could be achieved with an optimal parameter set, as can be seen in Figures 4.5 and 4.7.

4.6.7 MLLT

Applying MLLT to LDA transformed features had little to no effect. When we estimate MLLT with our pipeline, the likelihood of the training data is maximized,

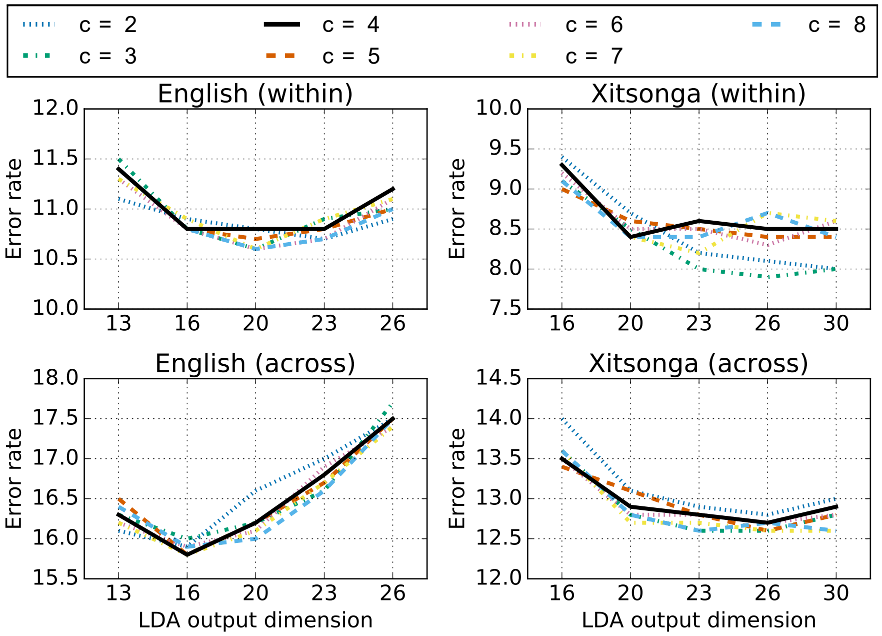


Figure 4.7: Discriminability error rates within and across speakers for DPGMM posteriorgrams after clustering LDA transformed PLP feature vectors with varying stacking context size c. The results are plotted as a function of the output dimensionality d of the transformation. *Left:* Error rates for English. *Right:* Error rates for Xitsonga.

given the acoustic model that we train along. With the DPGMM, a different generative model for the same data is assumed. Intuitively, it is not guaranteed that MLLT works well in such a cross-model scheme, which our results also show (see Table 4.1).

4.6.8 fMLLR

When we applied fMLLR transformations to the feature vector input for the DPGMM sampler, we observed a considerable across speaker discriminability error reduction of the GMM posteriorgrams extracted after the clustering, as seen in Table 4.1. The relative across speaker error reductions range from 3% to 6%, depending on the data set and the type of the transformed features (MFCC or PLP), but the crucial point is that in our experiments the improvements are independent of data amount, language, and feature types.

Besides doing performance tests, we also analyzed the actual effect of the

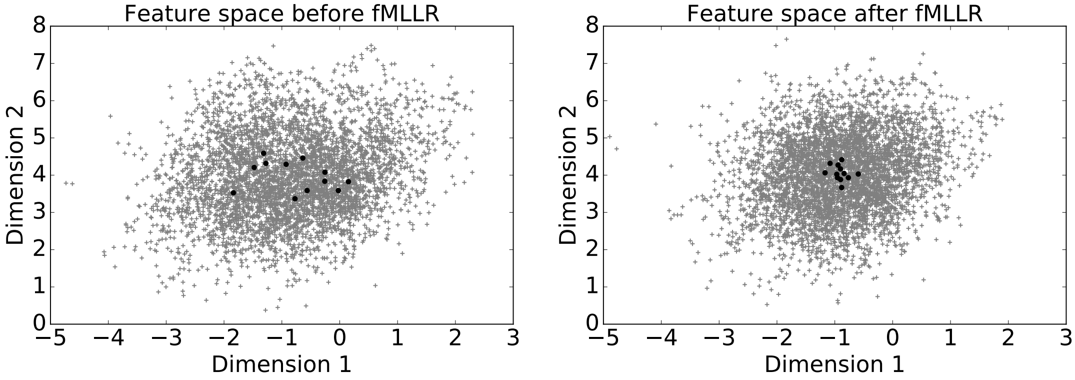


Figure 4.8: The figures exemplarily show the 1st and 2nd dimensions of the feature vectors belonging to an arbitrary English acoustic unit as detected by the DPGMM sampler. *Left* is the feature space before, *right* after applying fMLLR transformations. The speaker dependent means (black dots) now cluster in a much smaller area.

fMLLR in the feature space. With the frame based class labels from the clustering, we computed the means of the feature vectors for each class and calculated their average distance from each other. We compared the distances of the speaker-dependent means for each class before and after applying fMLLR transformations and found an average distance reduction of 19% and 17% relative for English and Xitsonga. This shows that the fMLLR causes the speaker dependent means to move closer together, a direct result of removing speaker variance from the features. Interestingly, the speaker independent means of all the classes moved further away from each other by about 0.7% to 2% relative for English and Xitsonga, and the variance of the features was reduced on average by about 1% relative for both data sets. This means that the fMLLR also helps to increase discriminability between classes. Figure 4.8 shows the effect of fMLLR in the feature space with an example.

4.6.9 Posteriorgram Combination

We were using the DPGMM clustering with various kinds of input features and combined the different results with the method from Section 4.4. The expectation was that GMM posteriorgrams from different DPGMM clusterings contain different kinds of latent information about the data and could complement each

Table 4.1: Summary of the experimental results. The table contrasts Chen et al.'s best performance (row 1), our baseline performance (row 2) (for details about the differences see Section 4.6.4), results using various feature transformations, and the best posteriorgram-level combination (Comb. *V*, see Table 4.2) (row 12). Indices for feature types denote context size, indices for transformations denote output dimensionality.

Features	English		Xitsonga	
	within	across	within	across
MFCC+Δ+$\Delta\Delta$ ([22])	10.8	16.3	9.6	17.2
MFCC_4+Δ+$\Delta\Delta$	12.2	19.5	8.9	14.2
MFCC_4+PCA_{20}	11.7	19.2	9.8	16.4
MFCC_4+LDA_{20}	11.0	16.6	8.7	13.2
MFCC_4+LDA_{20}+MLLT	11.0	16.5	8.7	13.1
MFCC_4+LDA_{20}+MLLT+fMLLR	11.0	16.0	8.6	12.7
PLP_4+Δ+$\Delta\Delta$	11.8	19.6	8.5	13.9
PLP_4+PCA_{20}	11.7	18.4	8.7	14.6
PLP_4+LDA_{20}	10.5	16.1	8.3	12.8
PLP_4+LDA_{20}+MLLT	10.5	16.2	8.4	12.9
PLP_4+LDA_{20}+MLLT+fMLLR	10.5	15.6	8.4	12.2
Posteriorgram combination *V*	**10.0**	**14.9**	**8.1**	**11.7**

other in combination. To produce candidate outputs for combination, we sampled DPGMMs

I multiple times with the same input features,

II for various transformed feature types,

III for transformed MFCC and PLP features,

IV for various LDA output dimensionalities,

V for various LDA input dimensionalities.

Table 4.2 lists the amount of DPGMMs used in each combination, along with their input features. The discriminability errors of the combined posteriorgrams

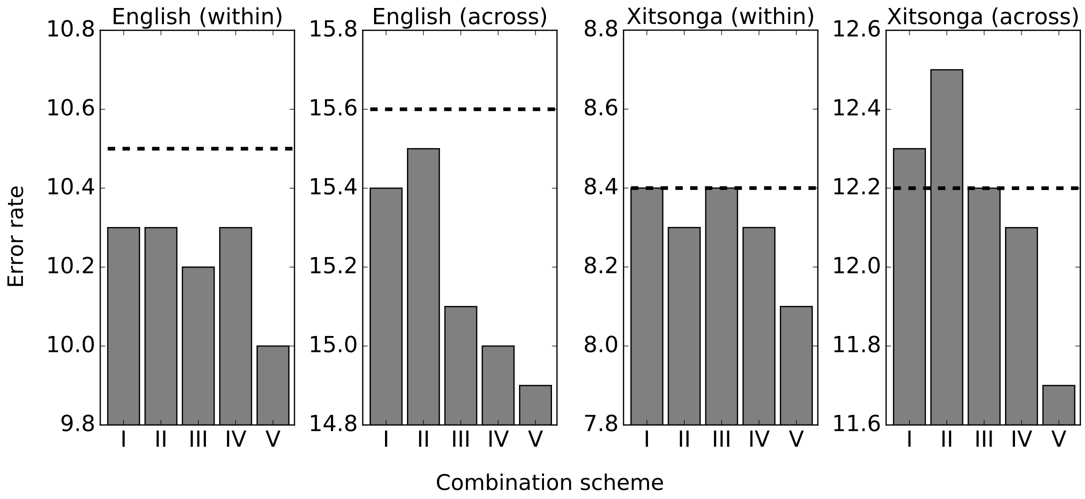


Figure 4.9: Discriminability error rates of various posteriorgram combinations. The dotted line marks the best performance on each data set before combining multiple clustering results.

are plotted in Figure 4.9. For the combination experiments we focused on the transformed PLP features, since they generally showed better performance than transformed MFCC features.

Combining the posteriorgrams of 5 DPGMMs that were sampled on the same input features (*I*) only had a small positive effect on the English data set where the discriminability errors were reduced slightly, compared to the best single DPGMM output. We take this as a sign that the DPGMM sampler generally leads to consistent output, which is why combining results of multiple runs on identical data is particularly helpful. Combination *II* showed similar results.

For combinations *III*, *IV* and *V* we combine the posteriorgrams of DPGMMs that were sampled given more diverse features. The results show that sufficient diversity is critical for the combination to produce better posteriorgrams. In all cases, the combined outputs show lower discriminability errors on the English data set, and can at least match the best single DPGMM output for the Xitsonga data set.

We achieved best results with combination *V*, where we combine posteriorgrams from DPGMMs that were sampled on transformed PLP features with varying context size *c*. The context size governs the stacked PLP feature vector size prior to the LDA transformation. While it seems that an increased context

Table 4.2: DPGMMs used for each posteriorgram combination (*Comb.*). The number in brackets behind *LDA* denotes the used output dimension *d*. For combination *V*, eight models were combined, one for each context size *c* between 1 and 8. The context size governs the stacked PLP feature vector size prior to the LDA transformation. Indices for feature types denote context size, indices for transformations denote output dimensionality.

Combination	#DPGMMs	Clustered features
I	5	PLP_4+LDA_{20}+MLLT+FMLLR
II	3	PLP_4+LDA_{20} PLP_4+LDA_{20}+MLLT PLP_4+LDA_{20}+MLLT+FMLLR
III	2	PLP_4+LDA_{20}+MLLT+FMLLR MFCC_4+LDA_{20}+MLLT+FMLLR
IV	4	PLP_4+LDA_{16}+MLLT+FMLLR PLP_4+LDA_{20}+MLLT+FMLLR PLP_4+LDA_{23}+MLLT+FMLLR PLP_4+LDA_{26}+MLLT+FMLLR
V	8	$\text{PLP}_{1\leq c\leq 8}$+$\text{LDA}_{20}$+MLLT+FMLLR

size does not necessarily help the individual DPGMM sampling in particular (as can be seen in Figure 4.7), we observed considerable improvements by combining the posteriorgrams produced by these models (see Figure 4.9). To ensure that the performance gain is not governed by the choice of the target for the posteriorgram mapping (see Section 4.4), we ran combination *V* multiple times – once for each set of posteriorgrams as target – and averaged the discriminability errors. We found that the average standard deviation across the data is low with 0.05, confirming that the improvements are independent from the choice of the mapping target. The numbers of this best performing combination are found in Table 4.1, which summarizes our experimental results.

4.6.10 Analysis

The improvements we have seen after each step in the pipeline are mostly consistent across the two data sets, with the exception of the improvements by LDA (see Table 4.1). The reductions by fMLLR (0% within and 3.1% to 4.6% across speakers) and by posteriorgram combination (3.5% to 4.7% within and 4% to 4.4% across speakers) are comparable across languages. The improvements by LDA however range from 2.3% to 11% within and 7.8% to 17.8% across speakers, where the larger improvements were observed on English. We believe this is again attributable to the conversational nature of the English data, which provides more room for improvements by LDA. In preliminary experiments on the very clean contrastive English data set mentioned in Section 4.6.6 we observed lower ranges of improvement by LDA (1.5% within and 3% across speakers), which supports our assumption that LDA has more impact on difficult data.

4.7 Experimental Evaluation (ZeroSpeech 2017)

This section describes our contribution to the unsupervised subword modeling task of ZeroSpeech 2017. We generalize our feature optimized DPGMM clustering approach by learning feature transformations and inferring subword models from separate training data and applying these to entirely new data from new speakers. This is done in a multi-stage clustering framework, where we unsupervisedly learn transformations using LDA, MLLT and (basis) fMLLR to reduce variance in the features. The overview of the refined framework is given by Figure 4.10. We show in our experiments that our method generalizes well to many languages and previously unseen data and scales well with increasing (or decreasing) data size. We achieve speaker robustness by blind speaker adaptation even with extremely few adaptation data. Furthermore, our framework has very little need for hyper-parameter adjustment and is entirely unsupervised, i.e., it only takes raw audio recordings as input, without requiring any pre-defined segmentation, explicit speaker IDs or other meta data.

4.7.1 The Zero Resource Speech Challenge 2017

In ZeroSpeech 2017, the newest installment of the challenge, the focus shifts to unsupervised subword modeling for previously unseen data, i.e., representations that were inferred on a training dataset must demonstrate their performance for a test set that contains unseen data from a new set of speakers. Moreover, it was the goal to develop methods that generalize well to any dataset in any language. To tackle this demanding task, we expanded our previous work that we started in the aftermath of the first zero resource speech challenge [68, 67, 66], which in turn drew from the findings of Chen et al. [22]. Specifically, we extended their idea by introducing a way to automatically learn feature transformations for unsupervised feature optimization supporting the DPGMM sampler to find better clusters, which in turn leads to a better subword modeling quality. The motivation for this is that standard features such as MFCC are not explicitly designed to maximize class discriminability and to minimize the effects for instance of speaker variability.

4.7.2 Data

The challenge organizers provided five language datasets, three sets of known languages – English, French and Mandarin – for system development, and two surprise language sets – LANG1 and LANG2 – for testing. The language sets vary in data size so that methods can be tested for scalability. Each set is split into a training portion and three test portions. The training data consists of unsegmented audio recordings. The test data comes as recordings of *120s*, *10s* or *1s* length. All files contain speech of exactly one speaker. The speakers in the test set are unseen, i.e., they are not part of any of the training sets. No other information about speaker identities is provided. The test sets for the development data come with references for the ABX discriminability test. Details of all datasets are provided in [34].

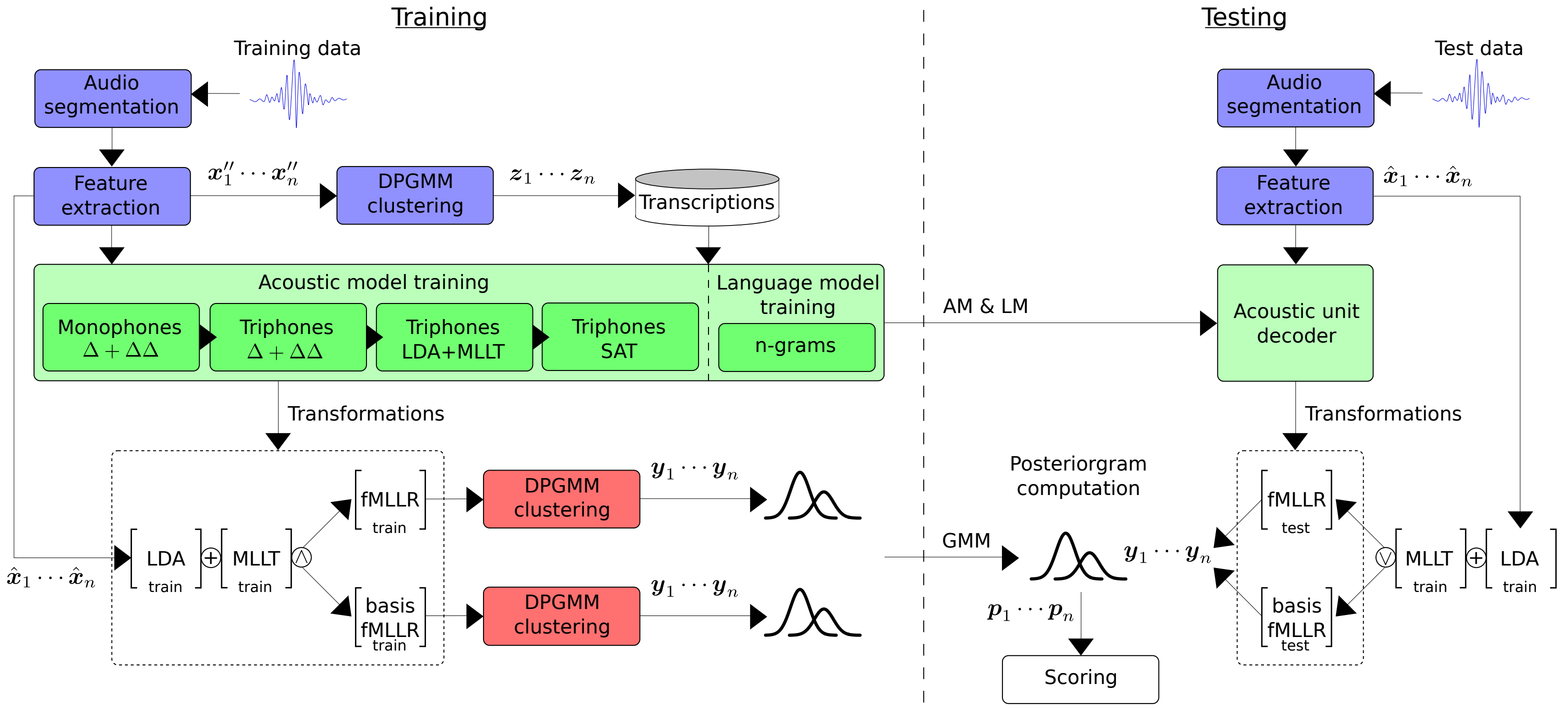


Figure 4.10: Scheme of the multi-stage framework for unsupervised subword modeling in previously unseen data.

4.7.3 Setup

For our experiments we only use freely available tools. To extract the different types of feature vectors, we again use the Kaldi speech recognition toolkit [132]. The specifics are the same as in previous experiments (see Section 4.6.3). For feature vector stacking we use Kaldi's default context size $c = 4$.

The VTLN is done by learning a universal background model on the full training dataset for each language and subsequent training of a model for extracting warp factors. The training provides us with warp factors for the training data. For the test data, we use the trained models to extract warp factors without the overhead of any re-training.

All AMs used in our refined framework are also trained with Kaldi recipe s5, with the modification that we train 1-state HMMs. All parameters of this pipeline are kept at their default values as they come with the recipe. For n-gram LM training, we use the SRILM toolkit [155] with $n = 3$ and Witten-Bell discounting [174].

Our unsupervised audio segmentation method uses SoX and FFmpeg[3]. We set $\texttt{segLen}_{\texttt{avg}}$ and $\texttt{segLen}_{\texttt{max}}$ to 30 seconds. We segment the training portions of each language dataset, and the recordings of the *120s* test sets.

For DPGMM sampling, we use the Dirichlet process mixture sampler by Chang et al.[4], as described in [18, 19]. We use default parameters for the priors and set the concentration parameter α to 1.

4.7.4 Baseline

The baseline and topline discriminability error rates for all datasets were established by the challenge organizers. The former is produced scoring standard 39 dimensional MFCC feature vectors with first and second order derivatives. The latter is produced by extracting and scoring posteriorgrams from a supervisedly trained language specific phone recognition pipeline using Kaldi.

The reference point for evaluating the impact of our proposed feature optimization approach is the performance of posteriorgrams that were extracted from a DPGMM that was sampled given standard 39 dimensional PLP feature vectors

[3] http://sox.sourceforge.net, http://ffmpeg.org

[4] http://people.csail.mit.edu/jchang7/code.php

with first and second order derivatives (PLP+Δ+$\Delta\Delta$), i.e., without using any feature optimization.

All baseline and topline numbers as well as all results from the following experiments are listed in Table 4.3.

4.7.5 Parameter Tuning

Our pipeline has very little need for parameter tuning. For many parameters we expected widely used default values to be sufficiently optimized, such as the ones for the Kaldi model training or the DPGMM sampling. In the case of the concentration parameter α, Chen et al. [22] have demonstrated that with increasing complexity of the input features, the influence of particular values diminishes.

For the feature optimization specifically, we identified the LDA output dimensionality d to be most influential on the performance of the extracted posteriorgrams after clustering transformed feature vectors. For the LDA input dimensionality, which is determined by the feature vector stacking context size c, we found that any value of $c > 2$ produces optimal results already. In comparative tests, results for using c ranging from 3 to 8 were very similar to the default ($c = 4$).

Therefore, the only parameter in our entire framework subject to tuning is the LDA output dimensionality d. We tuned on the development languages by checking the performances for $d \in \{20, 23, 26, 30, 33, 36, 39\}$, where the highest dimensionality is the size of the default PLP+Δ+$\Delta\Delta$. The results of these tests are plotted in Figure 4.11. The graphs show a correlation between increasing dimensionality, ABX discriminability error of the posteriorgrams extracted after clustering and number of found DPGMM classes. The error reductions tend to flatten out with d around 33 for all development sets, which is also when the number of classes flattens out. To keep computational costs as low as possible, but still maintaining optimal performance we fixed the LDA output dimensionality to 33 for all five language datasets.

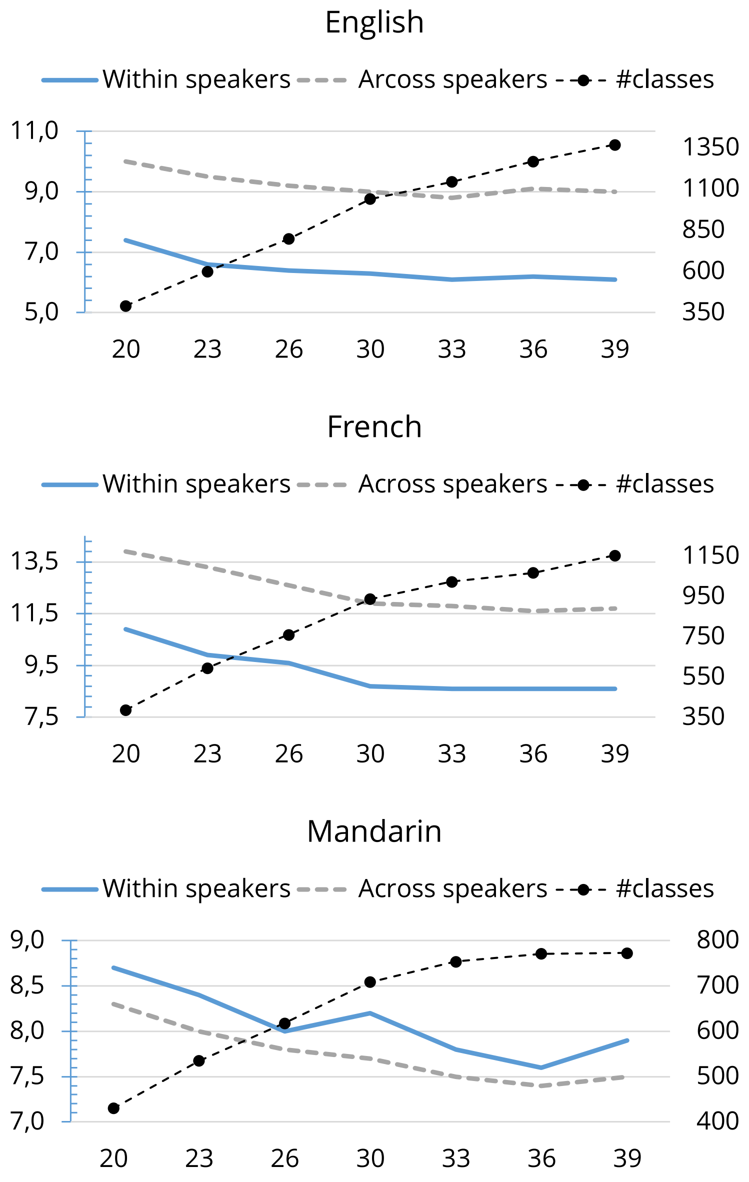


Figure 4.11: Errors (primary y axis) and DPGMM classes (secondary y axis) as a function of feature dimensionality (x axis).

Table 4.3: Summary of the experimental results. *Combination* marks the official results for our contribution to ZeroSpeech 2017. Indices for feature types denote context size, indices for transformations denote output dimensionality.

Systems	English			French			Mandarin			LANG1			LANG2		
	1s	10s	120s	1s	10s	120s	1s	10s	120s	1s	10s	120s	1s	10s	120s
ABX discriminability errors across speakers															
Baseline	23.4	23.4	23.4	25.2	25.5	25.2	21.3	21.3	21.3	23.6	23.2	23.0	30.0	29.5	29.5
Topline	8.6	6.9	6.7	10.6	9.1	8.9	12.0	5.7	5.1	12.8	10.5	10.4	7.1	3.6	4.3
PLP+Δ+$\Delta\Delta$	10.7	9.7	9.8	15.4	13.3	13.2	9.9	8.9	8.9	-	-	-	-	-	-
PLP_4+LDA_{33}	10.3	9.4	9.5	14.4	12.8	12.7	9.6	8.4	8.5	-	-	-	-	-	-
+MLLT	10.3	9.4	9.5	14.1	12.7	12.6	9.5	8.4	8.4	-	-	-	-	-	-
+fMLLR	**10.0**	9.2	8.8	13.9	12.2	11.8	9.2	7.7	7.5	-	-	-	-	-	-
Combination	10.1	**8.7**	**8.5**	**13.5**	**11.6**	**11.3**	**8.8**	**7.4**	**7.3**	**11.8**	**10.0**	**9.7**	**13.0**	**10.0**	**9.8**
ABX discriminability errors within speakers															
Baseline	12.0	12.1	12.1	12.5	12.6	12.6	11.5	11.5	11.5	10.3	9.3	9.4	14.1	14.3	14.1
Topline	6.5	5.3	5.1	8.0	6.8	6.8	9.5	4.2	4.0	8.7	7.1	7.0	6.6	4.6	3.4
PLP+Δ+$\Delta\Delta$	7.3	6.3	6.4	10.6	9.3	9.0	9.1	8.2	8.1	-	-	-	-	-	-
PLP_4+LDA_{33}	7.1	6.3	6.3	10.1	9.2	9.0	9.1	8.2	8.3	-	-	-	-	-	-
+MLLT	7.1	6.4	6.3	10.2	9.2	9.3	9.3	8.1	8.1	-	-	-	-	-	-
+fMLLR	6.9	6.4	6.1	10.3	9.0	8.6	9.0	8.2	7.8	-	-	-	-	-	-
Combination	**6.8**	**6.1**	**6.0**	**9.7**	**8.7**	**8.4**	**8.8**	**7.8**	**7.7**	**6.4**	**5.6**	**5.2**	**10.8**	**8.8**	**8.3**

4.7.6 Handling Very Short Utterances

We found that to reduce the discriminability errors across speakers, the speaker adaptive treatment of the input features to the DPGMM sampler is crucial. While the application of LDA and MLLT already lead to improved performance within as well as across speakers, it is the fMLLR transformations that lead to major performance gains of up to 12% relative.

Since standard fMLLR is not performing well on very short utterances, which is especially critical for the *1s* test sets, our pipeline automatically falls back to basis fMLLR for these datasets. We can see that even though the improvements by basis fMLLR tend to be somewhat smaller, they still contribute considerably to achieve a better performance.

Although the relative error reductions vary between the datasets, the improvements achieved by applying fMLLR are comprehensive and consistent across the board, which proves that the speaker adaptation is absolutely vital for producing high quality results.

4.7.7 Posteriorgram Combination

In previous experiments, we found that the outcomes of different DPGMM clusterings can contain latent information about the data that complement each other in combination. We therefore applied the same combination scheme as in Section 4.6.9. The details are as follows.

To generate candidates for posteriorgram level model combination we sample multiple DPGMMs for input features that vary in the LDA input dimensionality. This choice of candidates is supported by our earlier findings. The dimensionality of the input to the LDA is governed by the stacking context size parameter c. We sample one DPGMM for each c in the range from 2 to 8, compute posteriorgrams for the test sets with each of the DPGMMs and combine the different results. On average, the errors are reduced by 3.3% across speakers and 3.0% within speakers for the development language test sets, compared to using the posteriorgrams for features that use the default context stacking size $c = 4$. The error rates after combination as listed in Table 4.3 are the final and also the official results for our contribution to ZeroSpeech 2017.

4.8 Analysis

This year's challenge also aimed at the ability of methods to scale with increasing amounts data. We conducted a real time factor (RTF) analysis of the DPGMM inference (being the critical component in the framework) and found that the run time increases roughly linearly with the data. The RTFs in ascending order of the training data size are 0.57 for Mandarin 0.49 for French and 0.56 for English. Our results show that the benefits from feature optimization also scale well with the data. Regardless of training data size, test set size and utterance lengths, we observed consistent and considerable improvements especially by applying fMLLR even in the extreme cases of the *1s* test sets.

4.8.1 Conclusions

We presented a novel approach to optimizing the input of a DPGMM sampler to improve acoustic unit discovery. We evaluated the quality of acoustic unit discovery by computing ABX discriminability errors for posteriorgrams that were extracted from DPGMMs. To substantiate the strengths of our method, we demonstrated its effectiveness on two very different data sets that vary in size, language and speech quality. We demonstrated that it is possible to estimate supervised feature transformations without prior supervision, and that these transformations considerably improve clustering performance. Posteriorgrams of DPGMMs that were sampled given transformed features showed drastically reduced discriminability errors. The use of multiple transformations at once produced better results. A method we introduced for combining the results of multiple DPGMM samplings boosts sound class discriminability even further.

The lowest discriminability errors we achieved are 10% within and 14.9% across speakers for English, and 8.1% within and 11.7% across speakers for Xitsonga. Our proposed framework clearly outperforms our own baseline, as well as the previous state-of-the-art [22]. We believe our approach to optimizing feature vectors for clustering is universal and will be helpful for other zero and low resource tasks as well. In future work we will explore the applicability of our method to other tasks beyond improving automatic unit discovery.

We have shown that our approach to unsupervised subword modeling meets all the requirements that were imposed for ZeroSpeech. Our results demonstrate that our method scales well with increasing (or decreasing) amounts of data. We have also shown that our framework generalizes well across various datasets covering many different languages, with very little need for parameter adjustment. Most importantly, the entire framework is completely unsupervised, requiring no labels, segmentations or any other meta data to perform subword modeling for raw speech.

Chapter 5

Training an Acoustic Unit Tokenizer

"Jedem Worte klingt
Der Ursprung nach, wo es sich her bedingt." [1] *[167]*
– Johann Wolfgang von Goethe (1749-1832), *Poet*

In this chapter we further expand our unsupervised learning scheme in the zero resource scenario of our previous studies. We propose to build a full-fledged acoustic unit tokenizer without prior labels. For that, we combine our iterative DPGMM clustering framework with a standard pipeline for supervised GMM-HMM acoustic model (AM) and n-gram language model (LM) training, along with a scheme for iterative model re-training. Specifically, we sample a DPGMM to find a dynamically sized set of acoustic units that are optimized with respect to sound class discriminability. These acoustic units are used to initialize a context dependent speaker adaptive AM and an acoustic unit based n-gram LM. Similar to [152, 25] we follow an iterative approach attempting to gradually improve the trained models by decoding and re-training, but we let the DPGMM sampler decide the amount and structure of the used sounds.

With our proposed framework it is possible to build a DPGMM-HMM acoustic unit tokenizer that is competitive with supervisedly trained phone recognizers,

[1] "In each word there rings
An echo of the source from which it springs."

according to the performance on the ABX sound class discriminability task [141]. The ABX test based evaluation measures class discriminability of posteriorgrams. This allows a direct comparison of the decoding quality with the clustering quality of the DPGMM. We show that our DPGMM-HMM recognizer can beat the baseline set by our previous studies on DPGMMs. We also show that the model re-training helps improve performance even over multiple iterations. Our results indicate that the contextual information encapsulated in the LM considerably helps the sound class discriminability. Useful models can be unsupervisedly learned even on minimal amounts of data. We argue that by utilizing the DPGMM-HMM framework it is possible to build a state-of-the-art acoustic unit tokenizer without any prior supervision.

5.1 Acoustic Unit Recognition

The automatic labels generated with the method described in the previous chapter can be used to train acoustic and language models fit for decoding. This step uses the same standard pipeline for supervised training as for estimating the feature transformation, now with the objective to decode the target data with the resulting model in combination with the LM. The acoustic unit recognition scheme is depicted in Figure 5.1. The data sets we use in this zero resource setting are the only resources we have for training and testing, thus the entire training and decoding pipeline is designed for x-fold cross-validation.

5.2 Training

5.2.1 Acoustic Unit Discovery

To solve the task of acoustic unit discovery, we utilize a DPGMM sampler to cluster extracted speech features into various sound classes. The set size is determined dynamically by the Bayesian approach. Our method is based on [22], but has been modified by us in previous work to incorporate automatically estimated linear feature transformations which proved to be very helpful in constructing good features for boosting the clustering quality (see Chapter 4). Because many useful feature transformations need labels for estimation, we use a multi-staged clustering framework that automatically finds frame-based class labels in a first

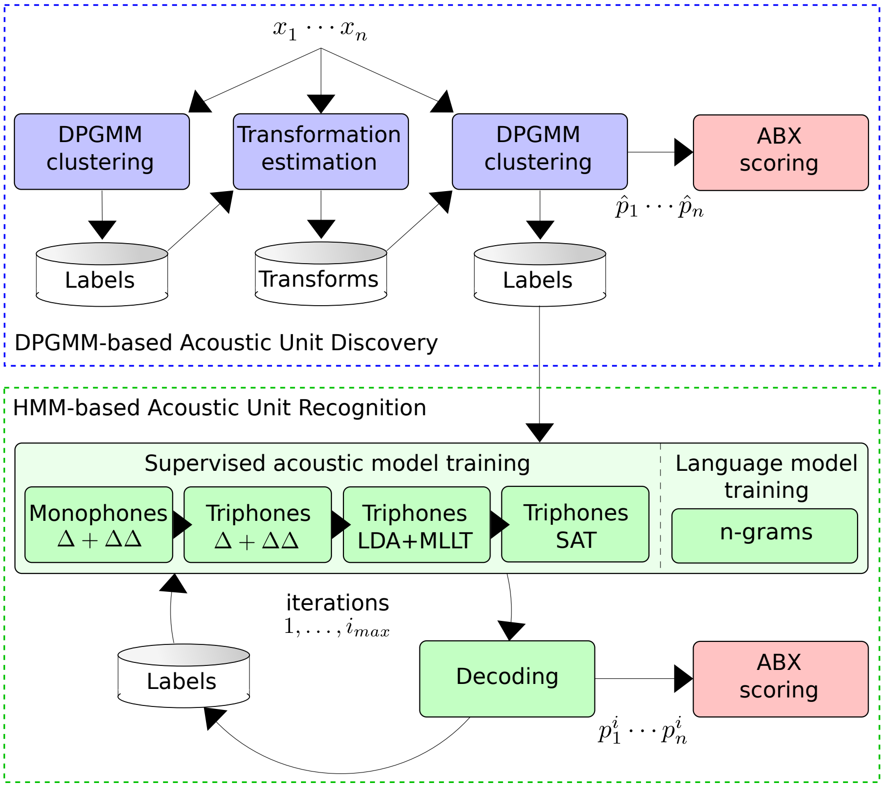

$x_1 \cdots x_n$
DPGMM clustering
Transformation estimation
DPGMM clustering
ABX scoring
$\hat{p}_1 \cdots \hat{p}_n$
Labels
Transforms
Labels
DPGMM-based Acoustic Unit Discovery
HMM-based Acoustic Unit Recognition
Supervised acoustic model training
Language model training
Monophones $\Delta + \Delta\Delta$
Triphones $\Delta + \Delta\Delta$
Triphones LDA+MLLT
Triphones SAT
n-grams
iterations $1, \ldots, i_{max}$
Labels
Decoding
ABX scoring
$p_1^i \cdots p_n^i$

Figure 5.1: Scheme of the DPGMM-HMM acoustic unit recognition framework. $x_1 \cdots x_n$ denotes the input feature vectors. The model training for acoustic unit recognition is iterative, where the models of iteration $i = 1$ are trained on the initial labels from the acoustic unit discovery step, and the models of iteration $i \in \{2, \ldots, i_{max}\}$ are trained on the hypotheses of iteration $i - 1$. $\hat{p}_1 \cdots \hat{p}_n$ denotes the posteriorgrams after DPGMM sampling. $p_1^i \cdots p_n^i$ denotes the posteriorgrams after decoding in iteration i.

run of clustering standard speech features, estimates feature transformations to transform these features and re-clusters the transformed input in a second run. The clustering scheme is depicted in Figure 5.1.

Speech feature transformations used by our framework help to project feature vectors into a more suitable sub-space for sound class discrimination by feature de-correlation and speaker adaptation. To estimate various transformations we train an AM by exploiting a standard pipeline for supervised training. During the course of the training we learn transformations via linear discriminant analysis (LDA), estimating maximum likelihood linear transforms (MLLT) and using feature-space maximum likelihood linear regression (fMLLR). LDA helps to minimize intra-class discriminability and maximize inter-class discriminability of the speech features and to enable dimensional reduction of high-dimensional stacked feature vectors. The state-dependent MLLTs maximize the likelihood of the target data. fMLLR helps to capture inter-speaker variability in speaker dependent transforms and to generate speaker independent state distributions instead.

We produce automatic labels by sampling an initial DPGMM given standard feature vectors with their derivatives. The output consists of generic class labels and the hypothesized class membership of every speech frame. Each class is simply named with the numeric ID of the Gaussian that most likely produces the respective feature vector.

The frame-wise labels serve as basis for the subsequent transformation estimation. We collapse the labels for each utterance to emulate a more natural textual reference by compressing all subsequent tokens of the same type to a single token. We initialize an AM by context independent monophone training. Then we subsequently train context dependent triphones on untransformed standard features. During this model training we automatically learn LDA transformations using the acoustic states as classes. The MLLTs are learned given the initialized HMMs, and fMLLR is based on alignments with speaker independent features.

5.2.2 Acoustic Model Training

The *acoustic model training* makes use of automatic transcriptions that are produced by collapsing the class label output from the multi-stage DPGMM clustering. The transcriptions are used to initialize context and speaker independent GMM-HMM monophone models, see Figure 5.2. Multiple iterations of increasingly complex training followed by label writing result in speaker adaptively trained context dependent triphones. The pre-processing produces LDA+MLLT+fMLLR transformed feature vectors.

A commonly used topology for acoustic modeling is left-to-right 3-state HMMs with or without skip states because of its suitability to model phone inventories crafted by linguists. It is not guaranteed, however, that automatically discovered acoustic units share the temporal properties of phones in the linguistic sense. Thus, our setup is designed to also operate with alternative HMM topologies.

The *language model training* produces an n-gram LM on the same automatic transcriptions, where the transcriptions are used as-is, i.e., no additional filtering

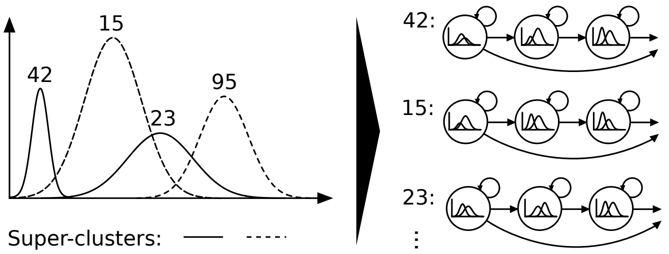


Figure 5.2: *Left:* Scheme of a sampled DPGMM. Super-clusters are visualized with different line styles. Each Gaussian represents one sound class, denoted by a generic ID. *Right:* DGPMM-HMMs trained on the DPGMM label output.

or cleaning is performed prior to training. The LM is based on the class labels, thus captures the phonotactics of the data, given the generic acoustic units.

The DPGMM sampler used to generate the automatic labels can sample labels that group several clusters according to some cluster similarity measure. These super-cluster labels can be used as an alternative to the normal cluster labels, thus effectively reducing the amount of potential acoustic units to be trained. Making use of this reduced set of classes makes sense when the amount of clusters found during DPGMM sampling is considerably higher than the size of commonly used phone or sound inventories.

5.2.3 Decoding

The decoding is performed with the generic acoustic unit based AM and LM, and in turn produces acoustic unit based hypotheses, i.e., essentially resembling a "phone" recognizer. Because naturally we do not have a development data set at hand, we use default values for all parameters that might be subject to tuning, such as beam sizes and model weights.

5.2.4 Iterative Re-training

A first system sys_1 is initialized with the help of the transcriptions that were produced by formatting the DPGMM output. By default, we iteratively re-train AM and LM simultaneously by using the hypotheses produced with system sys_{i-1} to build system sys_i in iteration $i \in \{2, \ldots, i_{max}\}$. The iterations after building system sys_1 can alternatively be restricted to one model type, i.e., either the AM

Table 5.1: The baseline results provided by the DPGMM clustering (*DPGMM*), the top-line result provided by the supervisedly trained phone recognizer, and the optimal results for each condition given our proposed setup (both *DPGMM-HMM*).

Features	English		Xitsonga	
	within	across	within	across
DPGMM ([22])	10.8	16.3	9.6	17.2
DPGMM ([67])	10.6	15.7	8.4	12.2
DPGMM-HMM (supervised)	12.5	16.6	6.7	11.2
DPGMM-HMM	**11.1**	**15.1**	8.2	11.6

or the LM is the sole subject of iterative re-training.

It is straightforward to replace the transcriptions of the previous training step with the hypotheses. After each iteration, we evaluate the system performance by extracting frame-wise acoustic unit posteriorgrams and measuring their ABX sound class discriminability.

5.3 Testing

The evaluation metric we use to measure the cluster quality and the decoding quality is the ABX phone discriminability between phonemic minimal pairs [141], as described in Section 4.5. The provided toolkit allows the easy evaluation of posteriorgrams which we can extract after DPGMM clustering as well as after decoding.

Each acoustic unit being found via DPGMM clustering (and used for acoustic modeling for the decoding approach) is considered a phone in the context of the evaluation. We compute GMM posteriorgrams for each speech frame after clustering as described in Section 5.2.1 and acoustic unit posteriorgrams after decoding as described in Section 5.1, and score them in the same manner. Both types of posteriorgrams share the same structure due to the fact that the sound units of the AM are identical with the DPGMM classes.

5.4 Experimental Evaluation

5.4.1 Data

We use the official test sets of the Interspeech zero resource speech challenge [164] for all our experiments, as described in Section 4.6.2. These are data sets for American English (4h 59min) and Xitsonga (2h 29min).

5.4.2 Setup

For the feature vector clustering via DPGMM sampling, we use the same initialization and parameters than in [68, 67].

We use the Kaldi speech recognition toolkit [132] to extract PLP speech feature vectors for a frame length of 25 milliseconds and frame shift of 10 milliseconds. Mean variance normalization (MVN) and vocal tract length normalization (VTLN) is applied. All AMs used in our framework are likewise trained with Kaldi, following a standard scheme for speaker adaptive training (Kaldi recipe s5). All parameters that can be tuned are set to default values. To form the input for LDA estimation, we stack the standard PLP features with a context of 4, meaning that the 4 left and 4 right feature vectors are stacked on top of the current vector, which is the center vector. The LDA output dimensionality is 20 for feature transformation prior to DPGMM clustering, and set to the default value 40 for the decoding. We use a either a modified 3-state HMM topology with a skip from the first state to the next HMM, or a 1-state HMM topology.

To train the n-gram LMs for our experiments, we use the SRILM toolkit [155] with Witten-Bell discounting [174] and no pruning. We set $n = 4$ for all decoding experiments.

5.4.3 Baseline

The baseline for the DPGMM based feature vector clustering performance was set by Chen et al. [22], which won track one of ZeroSpeech 2015 [164]. This system has been outperformed by our clustering setup using feature transformations as described in Chapter 4. We found that PLP feature vectors are consistently leading to a higher clustering quality than MFCC feature vectors. We also found that the stacking context parameter $c = 4$ prior to LDA transformation and LDA output dimensionality $d = 20$ are good values to work with. With the application of LDA we were able to produce feature vectors that considerably helped the DPGMM clustering process to find better clusters. Further, the transformations learned with fMLLR during the speaker adaptive training helped boost the discrimination capabilities across speakers.The performance of Chen et al.'s and our setup is listed in Table 5.1.

5.4.4 Decoding with Acoustic Units

We trained an AM and a 4-gram LM given the classes discovered during the DPGMM clustering. Because training and test data are identical in our scenario, we use 12-fold cross-validation for training the models for decoding. The cross-validation ensures that the measured performance is an indicator of how well the learned models generalize, besides showing that they are generally capable of representing the training data. The models are used to decode the cross-validation left-out portion of the data. The decoding hypotheses were subsequently used to re-train the models for another iteration of decoding. This was done multiple times to measure a potentially positive effect of iterative unsupervised re-training on the decoder performance.

To get a top-line performance for the decoding with acoustic units, i.e., the kind of performance we can expect if we had an optimal set of acoustic units and (near) perfect transcriptions to learn models, we also trained a normal AM and phone-based 4-gram LM with the same setup given the original references and decoded the target data with 12-fold cross-validation. All results are listed in Table 5.1.

The performance of the DPGMM-HMM acoustic unit recognizers is depicted in Figure 5.3. Even though a general tendency to convergence is not observable, one can see that multiple iterations of model re-training tend to have a positive effect. The error across speakers drops for the recognizers for both languages even after 3 or more iterations, whereas the positive effect diminishes more rapidly within speakers.

The acoustic unit recognizers are competitive when compared to the supervisedly trained phone recognizers. For English, our proposed setup can even beat the supervised system according to ABX discriminability within speakers.

The posteriorgrams after decoding start off with a higher discriminability error than the posteriorgrams after DPGMM sampling, which were used to generate the labels for the decoder training in the first place. In other words, a performance loss is observable by attempting to train more complex models. However, a steady performance improvement is observable for the discriminability across speakers, while the error rate within speakers remains relatively stable. We take this as an indicator that the models do have the capacities of still learning more from the data.

5.4.5 Using Super-cluster Labels

The DPGMM sampler can sample labels that group several clusters according to some cluster similarity measure, in this case the J-Divergence [99]. We used the super-cluster labels as an alternative to the normal cluster labels to effectively reduce the amount of potential acoustic units. The number of clusters found during DPGMM sampling usually is in the hundreds, whereas the sampled super-clusters are in the range of tens, raising the hope that they resemble more phone-like units. As can be seen in Figure 5.3 we indeed observed a performance gain when training the models on super-cluster labels, supporting our assumption that the super-clusters might be more suitable to describe the target data.

5.4.6 Modeling Sounds with Single States

The fact that we were not able to beat the DPGMM clustering in the ABX task led us to assume that the acoustic units we found might not quite resemble phones as defined by linguistics. An analysis has shown that the average length of our automatically inferred units by DPGMM clustering is 2.3 frames, whereas the average length of the ground-truth phone classes for the used test data is around 8 frames. Due to the Gaussian approximation of the mixture model, inferred units resemble relatively short, stationary sound phenomena. Therefore, the found units are potentially too short to be modeled accurately with 3 states, resulting in poor or even failing alignments of audio sequences to 3-state HMM sequences. Thus we also conducted decoding experiments with 1-state HMMs instead of 3-state HMMs. By simplifying the models in this way we observed a considerable performance gain. Apparently, the data can be represented more accurately with chained single state HMMs.

The posteriorgrams after decoding with 1-state HMMs outperform the DPGMM posteriorgrams in all but the within speaker discriminability test for English. It is also noteworthy that the model training seems to saturate after fewer iterations than before, possibly due to the reduced complexity of the AM. We now see optimal performance after the third iteration at the latest. The proposed system also clearly outperforms the supervisedly trained phone recognizer for English by showing a relative improvement of 9% to 11% in sound class discriminability performance. For Xitsonga, the performance of the automatic sound units is fairly close to the performance of the supervisedly trained recognizer.

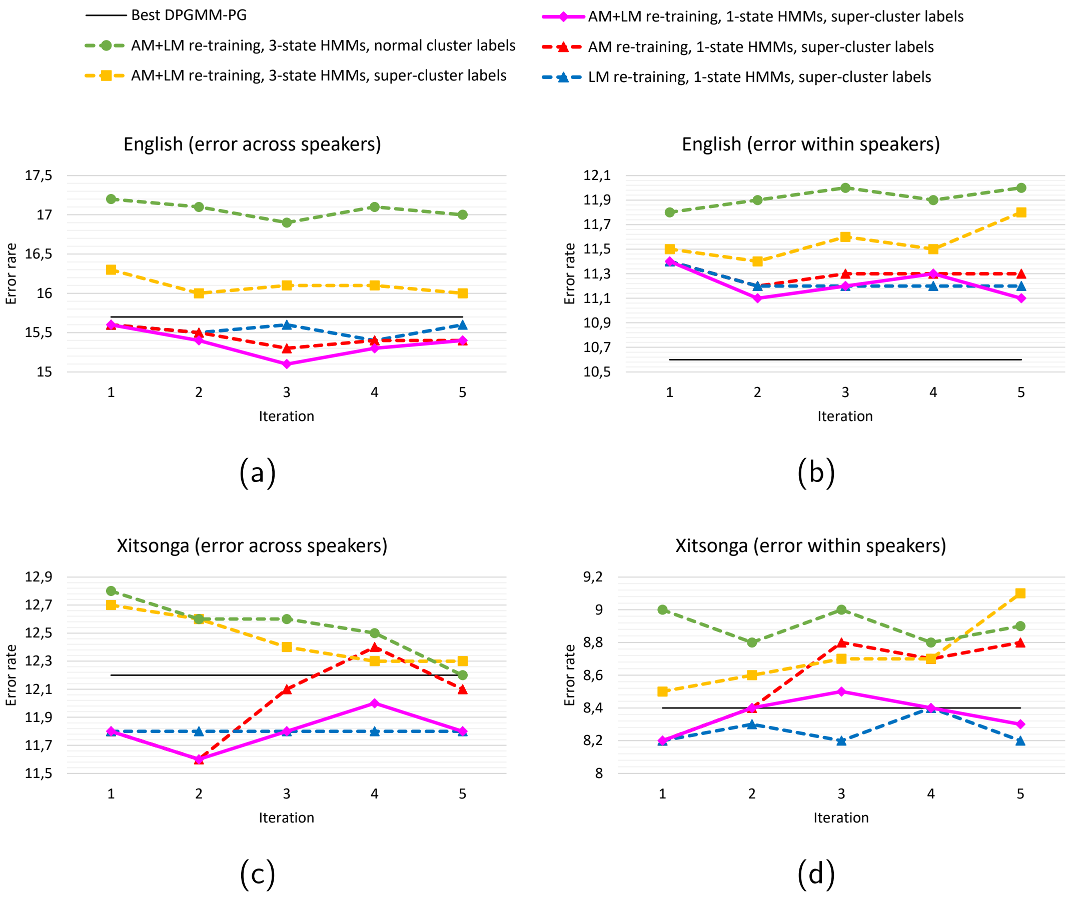
5.4.7 Selective Re-training

We conducted experiments to analyze the isolated effects of AM and LM retraining. In two lines of experiments we only re-trained one of the two model types each. The results that are depicted in Figure 5.3 allow conclusions regarding the importance of the amount of available data: For English we see an improvement when simultaneously re-training AM and LM. If both model types are re-trained exclusively, with the other model kept fix after iteration 1, the performance remains suboptimal. If the same test is done for Xitsonga however, one can see that the AM tends to deteriorate very quickly with new iterations of retraining. This is a strong indicator that the amount of training data is insufficient to reliably estimate models with multiple iterations. The LM re-training seems more robust but also suffers from multiple iterations. The combined re-training of both model types yields suboptimal performance compared to re-training the LM exclusively. The deteriorating AM is simply overpowering the benefits of an LM.

5.5 Conclusions

We proposed to build an acoustic unit recognizer without any provided labels by utilizing a Bayesian DPGMM sampler to unsupervisedly discover acoustic units in the target data for subsequent acoustic and language model training on automatically generated labels. The resulting DPGMM-HMM acoustic unit recognizer was used to solve the ABX sound class discriminability task. Multiple iterations of decoding and model re-training proved to be suitable to boost performance.

We showed that the automatically discovered acoustic units differ from phones in the sense that they seem generally shorter and resemble more stationary sound phenomena, which can be attributed to the Gaussian approximation of clusters by the DPGMM. We demonstrated that the contextual informations modeled by the LM considerably help discriminating sounds and that the sound class discriminability after DGPMM clustering can be outperformed by introducing such contextual knowledge. With our proposed framework it is possible to build a DPGMM-HMM acoustic unit recognizer that is competitive with supervisedly trained phone recognizers. Useful models can be unsupervisedly learned even on minimal amounts of data. A recognizer build in this way without any prior supervision can serve as basis for further and more sophisticated system development. In future work we plan to utilize such initialized systems to also infer lexical knowledge from the data to boost recognition performance and to enable automatic generation of lexica for new languages.



(a)

(b)

(c)

(d)

Figure 5.3: Error rates within and across speakers for both languages in dependency of the model training iteration. The black horizontal line marks the baseline set by the best DPGMM clustering. *AM re-training* and *LM re-training* denote systems with exclusively re-trained AM or LM, respectively. *1-state HMMs* denotes systems that use the single state topology instead of the default. Systems have been trained either on the normal DPGMM label output or on the super-cluster labels.

Chapter 6

Dirichlet Process Mixture of Mixtures Model

"Les langues, comme les arts, n'ont point de bornes connues." [1] *[58]*
– Antoine François Prévost (1697-1763), *Novelist*

Dirichlet process mixture models (DPMMs) [41, 7] are firmly established in pattern recognition and machine learning. Also known as infinite mixture models [138], they elegantly extend finite mixture models by the aspect of automatic model selection. This property makes them a popular tool for solving clustering tasks that are challenging with regards to estimating model complexity a priori. Several extensions to the original concept have been introduced over time, most notably hierarchical models [159, 20] and Dirichlet processes (DPs) with dependencies [105, 106].

DPMMs with *Gaussian* components gained increased interest in the field of low resource automatic speech processing, particularly as method for tackling the task of unsupervised subword modeling. The task is to infer acoustic units from raw audio data that are suitable to reliably represent human speech, i.e., that show low discriminability errors. DPMM samplers were used for subword model inference in an array of works related to the zero resource speech challenge [164, 34, 22, 23]. The idea is that each Gaussian in a mixture model that was inferred from speech data is considered a separate acoustic class. The method introduced in Chapter 4 of this thesis improved unsupervised subword modeling via DPMM

[1] "Languages, like the arts, have no known limits."

sampling by unsupervisedly transforming the sampler's input.

A major impediment for producing better subword models however is the simplicity of the inferred model. It is a long standing modeling assumption that speech observations, i.e., feature vectors that belong to specific sound categories, are multimodally distributed [136]. In practice, Gaussian mixture models (GMMs) are well established to model acoustic units such as phones [148, 97], and methods such as continuous hidden Markov models (HMMs) make use of state-dependent GMMs as multimodal distributions to model emission probabilities of speech observations [170, 136]. It seems therefore an oversimplification to assume single unimodal distributions to be a good model representation for individual sounds. This assumption limits the inferred units to represent generally very short stationary sound phenomena (see Chapter 5).

DPMMs work very well when the clusters in a data set are unimodally distributed. But problems arise when clusters follow more complex, e.g., multimodal, distributions. In such cases, a model that fits unimodal distributions (e.g., single Gaussians) to clusters tends to over-fragment the feature space and to suggest more clusters than actually present. In other words, real clusters tend to be represented by multiple components, i.e., "sub-clusters". However, without dependency modeling in DPMMs, inferred "sub-clusters" are considered independent and the relations between them are lost. The consequence is that the inferred clusters do not reflect the actual structure in the data. To more accurately approximate multimodally distributed clusters, a model that assigns multiple mixture components to each cluster would be required [178].

We consider unsupervised subword modeling to be such a problem where the ability to infer multimodal clusters from a data set can provide models that represent the real underlying data distribution more accurately. The expectation is that a mixture of clusters, where each cluster is a mixture itself, should be a better representation of acoustic units with favorable characteristics. Specifically, one would expect the number of inferred classes to be lowered, the overall model size therefore be reduced. At the same time, average durations when classifying sequences of sounds should be longer. Sound representations should also be more robust across data of different speakers and show higher discriminability due to more natural modeling.

We develop a sampler for a Dirichlet process mixture of mixtures model (DP-MoMM) to overcome the limitations of DPMMs and to enable the inference of a mixture of *multimodal* clusters. In our proposed DPMoMM, each cluster is a mixture of components, and the collection of clusters forms a global mixture of mixtures. Throughout this chapter, we will use the following terms to describe our model. Each mixture in the global mixture of mixtures is called a *cluster*. Each cluster is a mixture of *cluster components*. Cluster components can be shared across clusters, which is why they exist on a global level. We borrow a term from automatic speech recognition and call the global collection of components that can be part of a cluster a *codebook*. When we speak of *codebook components* or simply *components*, we refer to the global components that make up the codebook. A *codebook component* is a cluster component (i.e., a member) in at least one cluster, and each *cluster component* is identical with exactly one codebook component. The difference is that the same component has different weights in different clusters (if it belongs to more than one). Besides that, each codebook component has a global weight. An intuitive illustration of this model is provided in Figure 6.1. The detailed description of the model and its parameters is given in Section 6.2.

We build on the idea of Chang et al. [18] and develop a split, merge and switch sampler with the following characteristics: (1) our sampler jointly infers a global codebook of components, and clusters which are mixtures that are defined over this codebook; (2) split and merge moves modify codebook components; (3) split, merge and switch moves modify cluster components; (4) all sampling steps can be parallelized across clusters and components; (5) the jointly inferred codebook and the mixture of mixtures provide two alternatives to model the same underlying data.

We demonstrate in unsupervised subword modeling use case experiments on real speech data that our DPMoMM sampler is superior to a DPMM in terms of inferring models that are better representations of the underlying data structure. Specifically, with our method we infer mixtures of *Gaussian* mixtures from real speech observations that consistently show higher quality on several subword model evaluation metrics. For that, we extract frame-wise speech feature vectors from a data set and use our proposed sampler to cluster these speech observations into classes. In a standard DPMM, each class is represented by a single Gaussian, whereas with our proposed DPMoMM, each class is represented with its own GMM. This way we infer fewer clusters that represent subword units more consistently across speakers and that show longer average durations. We also show

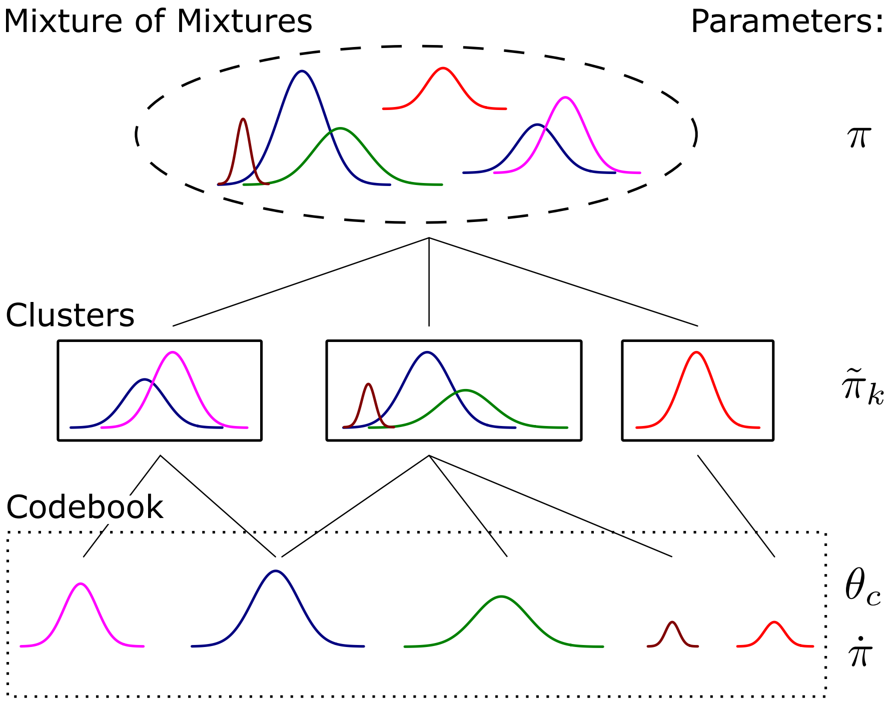


Figure 6.1: Illustration of the mixture of mixtures model. The codebook is a global collection of components. Clusters are mixtures, defined over the codebook.

that an additional switch sampler supports the convergence of the algorithm.

6.1 Related Work

The hierarchical Dirichlet process (HDP) is a well-known method for sampling mixture models that employ a hierarchy [159, 20]. The HDP can be used to infer *topics* that are shared between multiple *documents*, i.e., groups of data. An analogy between HDP and DPMoMM can be drawn to the following extent. A document in the HDP is a mixture of topics, just as a cluster in the DPMoMM is a mixture of components. Documents in the HDP share topics from a global set, just as clusters in the DPMoMM can share components from the codebook. The similarities between the models end at this point, however. The HDP differs greatly from the DPMoMM by assuming that a particular grouping of data into a finite set of documents is known a priori. Topics are assumed to be shared across documents, and each document is assumed to have its own particular distribution of these topics. Topic mixtures that describe individual documents can heavily overlap each other if they have many topics in common. In contrast, our proposed model does not assume a pre-defined grouping of data into document-like clusters. Instead, the DPMoMM sampler infers an unknown number of clusters within ungrouped data, comprised of an unknown number of components each. Our method infers a *mixture of mixtures*, i.e., a group of clusters which itself forms a mixture model with explicit mixture weights. In the DPMoMM, clusters are groups of neighboring components, and clusters do not tend to occupy the same regions in the feature space.

More related to our DPMoMM is the infinite mixture of infinite Gaussian mixtures (I^{2}GMM) by Yerebakan et al. [178]. The I^{2}GMM is a generative model that represents each cluster within a data set with its own mixture model. Here, a top layer DP defines meta-clusters, and lower layer DPs model the cluster data as a mixture of components. The top layer generates cluster parameters according to a base distribution H and defines the number and local expansion of clusters. The cluster parameters in turn define base distributions H_k for the lower layer which control the number and local expansion of cluster components. Covariance matrices are shared across components within the same cluster, leaving components to only differ in their means. In contrast to the I^{2}GMM, our proposed model does not define a prior over cluster appearances in form of meta-clusters. Instead, DPMoMM clusters can take on any structure that the data might inform. DPMoMM components also have their own covariance matrix each, which allows more natural approximations to the data. Further, unlike the I^{2}GMM, the DPMoMM supports the sharing of components across clusters, which enables the sharing of statistical strength [159].

Dependent Dirichlet processes are suitable to capture time dependencies between clusters or samples [184, 57, 16]. Temporal dependencies might in certain cases be reflected by locality in the feature space. With the DPMoMM, we propose a new kind of model sampler that explicitly infers a mixture of multimodal distributions to handle dependencies that are reflected by locality in the feature space.

Split and merge samplers are thoroughly discussed in an array of publications [28, 77, 78]. The DPMM sub-cluster algorithm of Chang et al. [18] addresses several issues that previous approaches coped with. Their sampler combines a non-ergodic restricted Gibbs sampler and split and merge samplers into a valid

Markov chain. The Gibbs sampler is restricted to non-empty clusters. Splits are proposed from sub-clusters that are learned jointly by deferred sampling. Moves are proposed with a Metropolis-Hastings (MH) algorithm. As instantiated-weight (IW) sampler, cluster weights are explicitly represented, as opposed to collapsed-weight (CW) samplers. Like in [123, 39], no finite approximations are used for the Dirichlet process, contrary to [39, 76]. The authors see the advantage of IW samplers in the possibility to parallelize across data points (which they refer to "inter-cluster parallelizable") and propose to use global split and merge moves to counter convergence issues. Inspired by these works, Chang et al. [18] propose moves that rely on jointly learned sub-clusters to reduce computational overhead during the MH steps.

This algorithm was used with success for unsupervised subword modeling in the scope of the zero resource challenge [164]. Chen et al. [22] inferred a DPMM from raw speech, with Gaussians as components, and used the Gaussian posteriorgrams extracted after sampling as new speech representation. In our own work, we extended this approach by developing a framework that unsupervisedly learns feature transformations from inferred classes (see Chapter 4). We showed that these transformations in turn improve the input to the DPMM sampler so that even better classes can be inferred, according to a sound class discriminability measure. We further demonstrated that the inferred classes can be used to model sounds for speech recognition purposes (see Chapter 5). However, we found there is need for a method to find more complex classes that are generalizing across speakers and that cover consistent sequences with longer durations. We show in this Chapter how we enhance the DPMM to be a DPMoMM that jointly learns components and mixtures of components and how we successfully use our model to improve unsupervised subword modeling.

Discriminative (as alternative to generative) non-parametric models such as infinite (structured) support vector machines (i(S)SVMs) [176, 177] have also been successfully applied in the ASR domain to dynamically model speech concepts. The idea of the iSVM and iSSVM is to divide the feature space into regions to be handled by a mixture of experts, i.e., specialized sub-models, where the number of experts is inferred from the data and the mixture underlies a DP prior. However, the number of actual concepts to be modeled is known beforehand, and the SVM training is supervised and relies on labels. In contrast, our DPMoMM infers concepts from data only without prior knowledge of any sort.

6.2 DP Mixture of Mixtures Model

In this section, we develop our DP mixture of mixtures model (DPMoMM). Definitions and mathematical expressions are kept general and are not restricted to a specific type of mixture components. Within the scope of this chapter, we use our sampling algorithm to infer mixtures of Gaussian mixtures in the use case of unsupervised subword modeling, for which we originally developed this method (see Section 6.7). We begin by reviewing the standard DPMM and the augmented DPMM of Chang et al. [18].

6.2.1 Graphical Representation

For the sake of clarity, we repeat the descriptions of the general DPMM and Chang et al.'s augmented DPMM [18] within this section to point out the differences to our proposed DPMoMM. Figure 6.2a is the representation of the general DPMM in plate notation. x_i is an observed data point i out of N data points, and z_i is the corresponding discrete label for that data point. π denotes the theoretically infinite dimensional vector of mixture weights. α is commonly referred to as the concentration parameter for the Dirichlet process, which governs the likelihood for new classes to be generated during sampling, and λ is the hyper-parameter for the Dirichlet process base measure. θ_k denotes the parameters of cluster k, e.g., mean and covariance in the case of Gaussians.

The generative process of the DPMM is expressed as follows:

$$x_i \sim p(x_i|\theta_{z_i}), \qquad \theta_k \sim H(\lambda), \tag{6.1}$$

$$z_i \sim \text{Discrete}(\pi), \qquad \pi \sim \text{GEM}(\alpha), \tag{6.2}$$

where GEM$(\cdot)$ denotes the stick-breaking process, and H is the DP base measure. The generative story of a data point x_i is this. A discrete cluster label z_i is sampled from the set of all possible clusters, which are distributed according to the weights in π. Given the cluster label, x_i is drawn from the cluster with parameters θ_{z_i}.

Figure 6.3a is Chang et al.'s [18] augmented DPMM using auxiliary variables. Each regular cluster is augmented with two explicit sub-clusters, denoted as l for "left" and r for "right". The goal is to design a model that is tailored toward splitting clusters. By picking suitable distributions for these sub-clusters, they can provide good split proposals for their regular parent cluster. Each data point

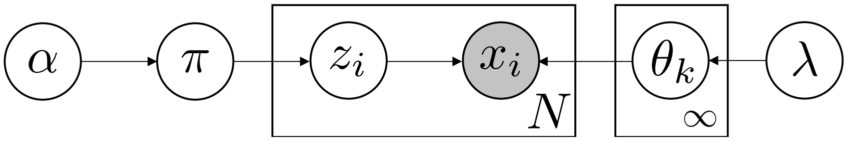


Figure 6.2: Standard DPMM.

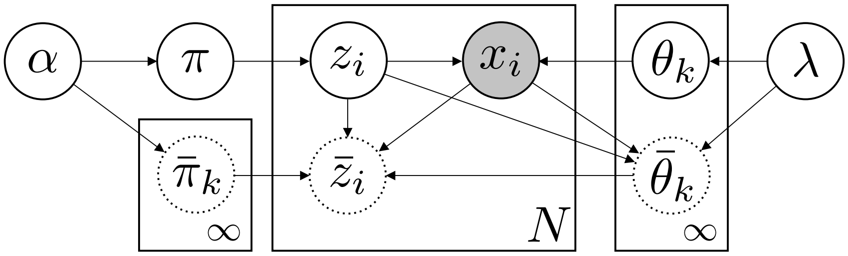


Figure 6.3: Augmented DPMM by Chang et al. [18]. Auxiliary variables are denoted by dotted circles.

is assigned to either the "left" or "right" sub-cluster with a sub-cluster label $\bar{z}_i \in \{l, r\}$. The naming convention implies that the sub-clusters are designed towards separating the data points into distinct groups within the parent cluster. Sub-clusters have their own weights $\bar{\pi}_k = \{\bar{\pi}_k^l, \bar{\pi}_k^r\}$ and parameters $\bar{\theta}_k = \{\bar{\theta}_k^l, \bar{\theta}_k^r\}$. It is important to note that in this *auxiliary space* the data points x_i generate the labels $\bar{z}_i$, in contrast to the regular space where z_i generate x_i.

Our proposed DPMoMM is depicted in Figure 6.4. As before, x_i is an observed data point i out of N data points that belong to a cluster k. β governs the global mixture proportions, and π is the vector of weights for clusters, sampled according to a stick-breaking process. z_i denotes the cluster assignment label for the corresponding data point. In our model, clusters are composed of components, which are represented by the following variables. $\tilde{c}_{ki}$ is the label of the cluster component conditioned on cluster k that the corresponding data point is assigned to. $\tilde{\pi}_k$ are the cluster component weights, governed by β and conditioned on cluster k. θ_c are the parameters for the component that generated x_i. All cluster components are defined globally in the *codebook*. The codebook is the weighted collection of all existing components in the DPMoMM, i.e., it is a global mixture of components. Given the codebook – which is itself a global mixture of components – sampling operations can be performed on component level, independent of their

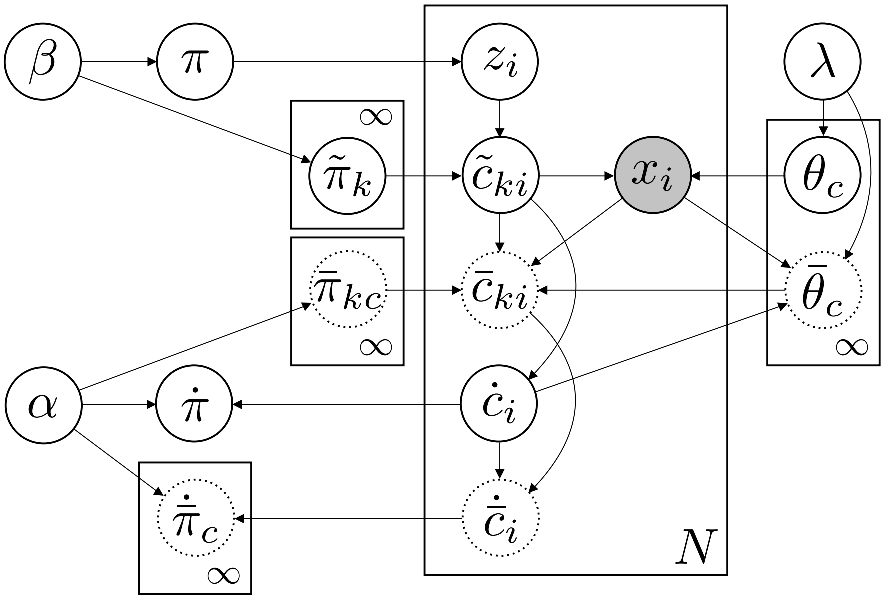


Figure 6.4: Proposed DP mixture of mixtures model (DPMoMM). Auxiliary variables are denoted by dotted circles.

respective cluster memberships. $\dot{c}_i$ assigns a data point i to a codebook component and is derived from the cluster component label $\tilde{c}_{ki}$. Codebook components have global weights $\dot{\pi}$, governed by a separate concentration parameter α. λ is again the hyper-parameter for the DP base measure.

The generative process of the DPMoMM is formally expressed as follows:

$$x_i \sim p(x_i|\theta_{\tilde{c}_{z_i i}}), \qquad \theta_c \sim H(\lambda), \tag{6.3}$$

$$z_i \sim \text{Discrete}(\pi), \qquad \pi \sim \text{GEM}(\beta), \tag{6.4}$$

$$\tilde{c}_{ki} \sim \text{Discrete}(\tilde{\pi}_k), \qquad \tilde{\pi}_k \sim \text{GEM}(\beta), \tag{6.5}$$

$$\dot{c}_i = \tilde{c}_{z_i i}, \qquad \dot{\pi} \sim \text{GEM}(\alpha). \tag{6.6}$$

The generative story for any data point x_i in a DPMoMM is as follows. A cluster label z_i is sampled from the set of all possible clusters, which are distributed according to the weights in π. Then, a component label $\tilde{c}_{z_i i}$ is sampled from the set of components that belong to the cluster with label z_i, which are distributed according to the weights in $\tilde{\pi}_{z_i}$. Given the cluster and cluster component labels, x_i is drawn from the cluster component with parameters $\theta_{\tilde{c}_{z_i i}}$. The codebook is a by-product of this generative process. The codebook component labels are copies of the cluster component labels according to Equation (6.6), and the weights of the codebook components are conditioned on these labels. Figure 6.1 is an illustration of the hierarchy within a DPMoMM. The sampling of the full model is described in detail in Section 6.3.

An intuition of what the DPMoMM represents can be given as follows. Assuming the tackled task is topic clustering, one can view DPMoMM clusters as groups of closely related topics (modeled by cluster components). If for instance there are three components modeling the topics “cars”, “trucks” and “motorbikes”, then a cluster that contains these components would model the meta-topic “personal vehicles”.

Similar to the augmented DPMM, we define an auxiliary space to enable a split and merge sampling approach. Each component in the DPMoMM is augmented with two sub-components, parametrized by $\bar{\theta}_c = \{\bar{\theta}^l_c, \bar{\theta}^r_c\}$, to provide good split candidates for a component split move proposal. $\bar{c}_{ki} \in \{l, r\}$ is the label that assigns the corresponding data point to a sub-component of $\tilde{c}_{ki}$ within cluster k, and $\bar{\pi}_{kc} = \{\bar{\pi}^l_{kc}, \bar{\pi}^r_{kc}\}$ denotes the weights for sub-components of component c within cluster k. On codebook level, sub-component labels $\dot{\bar{c}}_i$ are derived from the cluster sub-component labels $\bar{c}_{ki}$, i.e., $\dot{\bar{c}}_i = \bar{c}_{z_i i}$. $\dot{\bar{\pi}}_c = \{\dot{\bar{\pi}}^l_c, \dot{\bar{\pi}}^r_c\}$ are the sub-component weights for each codebook component, governed by α. The choice of the auxiliary parameter distributions follows Chang et al. [18], which is reflected in the way we sample these variables in Section 6.3.

6.2.2 Sampling Algorithm

Chang et al.’s DPMM sampler [18] is an instantiated-weight sampler that combines non-ergodic Markov chains into an ergodic chain and proposes splits from learned sub-clusters and merges of clusters. Their algorithm runs a Gibbs sampler, which samples the parameters and weights of each cluster and its sub-clusters, followed by sampling the cluster and sub-cluster labels for each data point. The Gibbs sampler is restricted to non-empty clusters. After Gibbs sampling, a split and merge sampler proposes with an MH algorithm to either split or merge clusters into new clusters. Gibbs sampling and split and merge sampling iterate until convergence or until a stop criterion is fulfilled.

Our proposed sampler uses a similar structure. Algorithm 3 is an outline of our algorithm in pseudo-code. We combine a restricted Gibbs sampler with a

Algorithm 3 DPMoMM sampling algorithm

```
Randomly initialize $K_{\text{init}}$ clusters with 1 component each
while stop criterion not met do
    Propose cluster merges and splits                    ▷ Section 6.5
    Propose cluster component switches                   ▷ Section 6.5
    Propose component merges and splits                  ▷ Section 6.4
    for all clusters with split components do
        Duplicate and update                             ▷ Section 6.4
    end for
    Sample parameters and labels                         ▷ Section 6.3
end while
```

split, merge and switch sampler for clusters and a split and merge sampler for components. The Gibbs sampler samples the parameters and weights of each codebook component and its sub-components, the weights of each cluster, and the weights of each cluster component. This is followed by sampling the cluster, component and sub-component labels for each data point. A split, merge and switch sampler proposes to either split or merge clusters or to move a cluster component from one cluster to another. A split and merge sampler for components proposes to either split or merge codebook components. Illustrations of the possible component and cluster moves are given in Figure 6.6. Gibbs sampling, split, merge and switch sampling for clusters, and split and merge sampling for components iterate until convergence or until a stop criterion is fulfilled.

The Gibbs sampling steps and the split and merge moves for the codebook components are equivalent to the original DPMM sampler of Chang et al. [18], and the codebook together with the global component weights is exactly the model that the original sampler would infer.

The following sections explain in detail the individual non-ergodic samplers that make up our proposed algorithm. The non-restricted Gibbs sampler is explained in Section 6.3. Section 6.4 explains the component split and merge sampler which are technically identical with the sampler in [18], but applied to the codebook components. Because changes of the codebook components lead to changes of the clusters, we introduce novel update steps for clusters. These are explained accordingly in the respective subsections. Lastly, we propose cluster split, merge and switch moves in Section 6.5. Figure 6.5 is an illustration of how our proposed DPMoMM algorithm behaves, compared to a DPMM.

6.3 Restricted Gibbs Sampling

In this section we lay out the details of the restricted Gibbs sampler that we employ. The Dirichlet process uses an infinite length prior on the cluster labels z_i, cluster component labels $\tilde{c}_{ki}$ and codebook component labels $\dot{c}_i$. However, any label can only point to a finite number of entities, i.e., the clusters and the components that exist in any current state of the model. Because the restricted Gibbs sampler does not create new clusters and components itself, the dimensions of the infinite vectors π, $\tilde{\pi}_k$, $\dot{\pi}$ and θ are technically finite during Gibbs sampling. Posterior distributions of weights are conditioned on the assignments of data points. The *restricted* conditional distributions of the DPMoMM are

$$p(\pi|z,\beta) = \mathrm{Dir}(N_1,\ldots,N_K,\beta), \tag{6.7}$$

$$p(\tilde{\pi}_k|z,\tilde{c},\alpha,\beta) = \mathrm{Dir}(B_k^1,\ldots,B_k^C,\alpha), \tag{6.8}$$

$$p(\dot{\pi}|\dot{c},\alpha) = \mathrm{Dir}(\dot{N}_1,\ldots,\dot{N}_C,\alpha), \tag{6.9}$$

$$p(\theta_c|x,\dot{c},\lambda) \propto f(\{x\}_c,\theta_c)f(\theta_c,\lambda), \tag{6.10}$$

$$p(z_i = k|x,\pi,\theta) \propto \pi_k \sum_{c=1}^{C} \tilde{\pi}_k^c f(x_i,\theta_c), \tag{6.11}$$

$$p(\tilde{c}_{ki} = c|x,z,\tilde{\pi},\theta) \propto \tilde{\pi}_{z_i}^c f(x_i,\theta_c), \tag{6.12}$$

$$p(\dot{c}_i = c|x,\dot{\pi},\theta) \propto \dot{\pi}_c f(x_i,\theta_c), \tag{6.13}$$

with

$$N_k = \sum_{i=1}^{N} 1_{z_i=k}, \qquad N_k^c = \sum_{i=1}^{N} 1_{\substack{z_i=k\\ \tilde{c}_{ki}=c}}, \tag{6.14}$$

$$\dot{N}_c = \sum_{k=1}^{K} N_k^c, \qquad B_k^c = \begin{cases} \frac{N_k^c+\beta}{C_k} & \text{if } N_k^c > 0,\\ 0 & \text{else,} \end{cases} \tag{6.15}$$

where K is the current number of non-empty clusters, C is the current number of non-empty codebook components and C_k is the current number of non-empty cluster components in cluster k. x is the vector of data points, z is the vector of cluster labels, $\tilde{c}$ is the vector of cluster component labels, $\dot{c}$ is the vector of codebook component labels, and $\tilde{\pi} = \{\tilde{\pi}_1, \ldots, \tilde{\pi}_K\}$. $\{x\}_c$ denotes all data points assigned to the global codebook component c. $f(\cdot)$ is a particular parametrized probability density function. E.g., $f(\{x\}_c, \theta_c)$ is the likelihood of the data subset $\{x\}_c$ given the cluster parameters θ_c, and $f(x_i, \theta_c)$ is the likelihood of the single data point x_i given θ_c. $1_{(\cdot)}$ is an indicator function that equals to 1 if the condition $(\cdot)$ holds true and is 0 otherwise.

Given these probabilities, the sampling is as follows. Conditioned on the labels in the current state, sample the parameters of each codebook component, and all cluster and component weights. Conditioned on all cluster and component parameters in the current state, for each data point, sample a label for a cluster, then sample a label for a component within the cluster. The conditional distribution in Equation (6.11) shows that the generative process described further above prefers cluster components to be neighbors in the feature space. During the model sampling described in this and the following sections, the algorithm will cause the clusters to be groups of nearby components so as to maximize the likelihood of the data.

For our proposed algorithm, we lay out the Gibbs sampler as follows. Given Equations (6.7)-(6.13), the posterior distributions of weights, labels and component parameters are expressed as

$$(\pi_1, \ldots, \pi_K, \pi_{K+1}) \sim \mathrm{Dir}(N_1, \ldots, N_K, \beta), \tag{6.16}$$

$$(\tilde{\pi}_k^1, \ldots, \tilde{\pi}_k^C, \tilde{\pi}_k^{C+1}) \sim \mathrm{Dir}(B_k^1, \ldots, B_k^C, \alpha), \tag{6.17}$$

$$(\dot{\pi}_1, \ldots, \dot{\pi}_C, \dot{\pi}_{C+1}) \sim \mathrm{Dir}(\dot{N}_1, \ldots, \dot{N}_C, \alpha), \tag{6.18}$$

$$\theta_c \overset{\propto}{\sim} f(\{x\}_c, \theta_c) f(\theta_c, \lambda), \tag{6.19}$$

$$z_i \overset{\propto}{\sim} \sum_{k=1}^{K} \pi_k \left(\sum_{c=1}^{C} \pi_k^c f(x_i, \theta_c) \right) 1_{z_i = k}, \tag{6.20}$$

$$\tilde{c}_{ki} \overset{\propto}{\sim} \sum_{c=1}^{C} \pi_k^c f(x_i, \theta_c) 1_{\tilde{c}_{ki} = c}, \tag{6.21}$$

$$\dot{c}_i = \tilde{c}_{z_i i}, \tag{6.22}$$

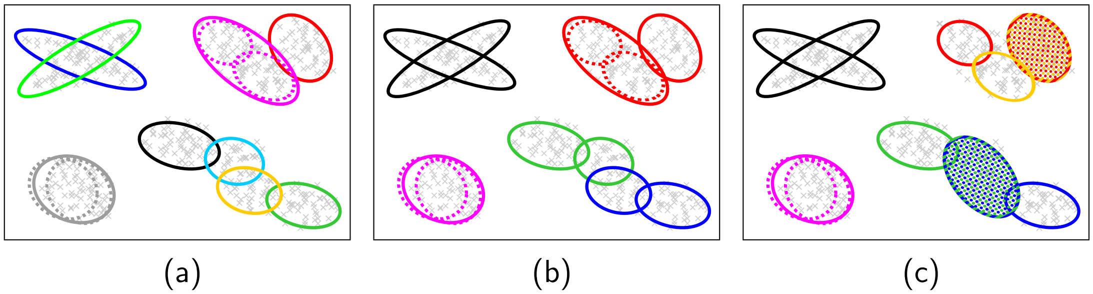


Figure 6.5: Illustration of the algorithm. Components are drawn with solid lines, sub-components with dotted lines. Only some exemplary sub-components are illustrated. *(a)*: Single component clusters inferred by a DPMM. This also corresponds to the codebook of the DPMoMM; *(b)*. Clusters inferred by a DPMoMM, where components of the same color belong to the same cluster; *(c)*: The same DPMoMM after a component split in the upper right cluster and a component merge in the lower right cluster. The original clusters are duplicated, and the bi-colored components are now shared across clusters.

with

$$\pi_{K+1} = 1 - \sum_{k=1}^{K} \pi_k, \tag{6.23}$$

$$\tilde{\pi}_k^{C+1} = 1 - \sum_{c=1}^{C} \tilde{\pi}_k^c, \tag{6.24}$$

$$\dot{\pi}_{C+1} = 1 - \sum_{c=1}^{C} \dot{\pi}_c. \tag{6.25}$$

Note that the codebook component labels $\dot{c}_i$ are not sampled explicitly but derived from the cluster component labels. This is done to not break the assignment of data points to components on the global level; a data point that is assigned to component c within a cluster must also belong to component c within the codebook for the split and merge sampling logic to work properly.

For the split and merge sampling steps described in Section 6.4, we make use of auxiliary variables that are jointly sampled with the regular variables. These auxiliary variables describe the sub-components which augment all the regular components. The sampled sub-components serve as good split candidates for

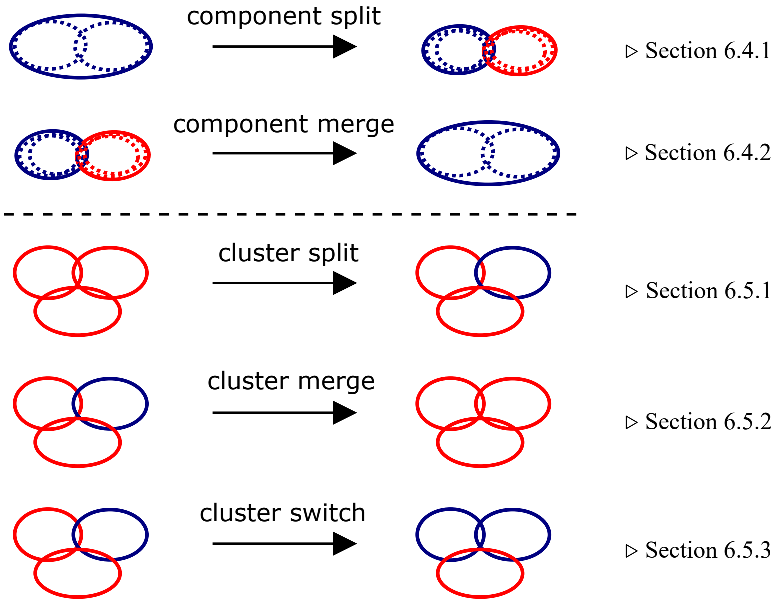


Figure 6.6: Overview of the component moves (top) and cluster moves (bottom).

eventual component splits. The auxiliary variables are sampled as follows:

$$(\bar{\pi}^l_{kc}, \bar{\pi}^r_{kc}) \sim \mathrm{Dir}(\frac{N^{c,l}_k + \alpha}{2}, \frac{N^{c,r}_k + \alpha}{2}), \tag{6.26}$$

$$(\dot{\bar{\pi}}^l_c, \dot{\bar{\pi}}^r_c) \sim \mathrm{Dir}(\frac{\dot{N}^l_c + \alpha}{2}, \frac{\dot{N}^r_c + \alpha}{2}), \tag{6.27}$$

$$\bar{\theta}^l_c \overset{\propto}{\sim} f(\{x\}^l_c, \bar{\theta}^l_c) f(\bar{\theta}^l_c, \lambda), \tag{6.28}$$

$$\bar{\theta}^r_c \overset{\propto}{\sim} f(\{x\}^r_c, \bar{\theta}^r_c) f(\bar{\theta}^r_c, \lambda), \tag{6.29}$$

$$\bar{c}_{ki} \overset{\propto}{\sim} \sum_{s \in \{l,r\}} \bar{\pi}^s_{\tilde{c}_{ki}} f(x_i, \bar{\theta}^s_{\tilde{c}_{ki}}) 1_{\bar{c}_{ki}=s}, \tag{6.30}$$

$$\dot{\bar{c}}_i = \bar{c}_{z_i i}, \tag{6.31}$$

with

$$N^{c,s}_k = \sum_{i=1}^{N} 1_{\substack{z_i=k \\ \tilde{c}_{ki}=c \\ \bar{c}_{ki}=s}}, \qquad \dot{N}^s_c = \sum_{k=1}^{K} N^{c,s}_k, \tag{6.32}$$

where $\{x\}^l_c$ and $\{x\}^r_c$ denote the subsets of data points that are assigned to the left and right sub-components of c. Note that, analogous to Equation (6.22), the labels $\dot{\bar{c}}_i$ are not sampled explicitly to not break the assignment of data points to sub-components on the global level.

6.4 Component Split and Merge Sampler

The split and merge moves for components are performed on the global codebook level and therefore rely on the codebook level variables that are jointly sampled with the other model variables. The moves are designed for efficiency by reducing the overhead of computational costs during the MH step and enabling parallelization across components. Components are equipped with auxiliary variables for sub-components. The parameters of the sub-components are sampled in the same fashion as the parameters for the regular "parent" components. Conveniently, samples for the variables of the regular components can be obtained by sampling the auxiliary variables, since we draw from a joint parameter space.

For performing split and merge moves for components with an MH-MCMC method, candidate moves, or proposals are required. Let $O = \{\dot{\pi}, \theta, \dot{c}\}$ be the set of component variables and $\bar{O} = \{\dot{\bar{\pi}}, \bar{\theta}, \dot{\bar{c}}\}$ be the set of sub-component variables. We propose a new set of random variables $\{\hat{O}, \hat{\bar{O}}\}$ for components and sub-components and compute the Hastings ratio of the form

$$HR = \frac{p(\hat{O}, x) p(\hat{\bar{O}}|x, \hat{\dot{c}})}{p(O, x) p(\bar{O}|x, \dot{c})} \frac{q(O, \bar{O}|\hat{O}, \hat{\bar{O}})}{q(\hat{O}, \hat{\bar{O}}|O, \bar{O})}, \tag{6.33}$$

where q is called the proposal distribution and x denotes the collection of all observations. The Hastings ratio weights the state of the model before and after actually performing a move, with the numerator standing for the post-move state and the denominator the pre-move state. As can be seen requires the Hastings ratio a reverse proposal to the proposed move. In the case of proposing a split move, that would be a merge, and vice versa.

A proposed move is accepted with the probability

$$\min(1, HR). \tag{6.34}$$

6.4.1 Split Moves

The sub-components are utilized as good split move candidates for the MH algorithm. Proposing a split move is typically a non-trivial task. The construction of a prospective move is necessary for an MH framework, where a proposal is weighed against the status quo and accepted or rejected with a certain probability. Theoretically, any kind of split proposal can lead to an ergodic chain [18].

However, proposals with low probability of being accepted unnecessarily increase the computational load, since in case of a rejection, all previous computational efforts are in vain. Iterative fitting of sub-components with the help of the auxiliary variables introduced above circumvents the risk of wasted computational time. The sub-components are sampled jointly with the normal components. During the MH-step, sub-components pose good proposals for split moves. Moreover, split move computations can be parallelized across components.

The proposal distribution for proposing a component split move with the help of the auxiliary variables is defined as follows. First, a split or merge move $Q \in \{Q^c_{\text{c-split}}, Q^{m,n}_{\text{c-merge}}\}$ is selected randomly. $Q^c_{\text{c-split}}$ denotes a move for splitting component c into m, n, and $Q^{m,n}_{\text{c-merge}}$ is a merge of components m, n into c. New sets of model variables are sampled as follows, each conditioned on Q.

If $Q = Q^c_{\text{c-split}}$:

$$(\{\hat{\dot{c}}\}_m, \{\hat{\dot{c}}\}_n) = \text{split}_c(\dot{c}, \dot{\bar{c}}), \tag{6.35}$$

$$(\hat{\dot{\pi}}_m, \hat{\dot{\pi}}_n) = \dot{\pi}_c \cdot (\dot{\bar{\pi}}^l_c, \dot{\bar{\pi}}^r_c), \tag{6.36}$$

$$(\hat{\theta}_m, \hat{\theta}_n) \sim q(\hat{\theta}_m, \hat{\theta}_n | x, \hat{\dot{c}}, \hat{\dot{\bar{c}}}), \tag{6.37}$$

$$(\hat{\bar{O}}_m, \hat{\bar{O}}_n) \sim p(\hat{\bar{O}}_m, \hat{\bar{O}}_n | x, \hat{\dot{c}}), \tag{6.38}$$

with $(\dot{\bar{\pi}}^l_c, \dot{\bar{\pi}}^r_c) \sim \text{Dir}(\hat{\dot{N}}_m, \hat{\dot{N}}_n)$.

If $Q = Q^{m,n}_{\text{c-merge}}$:

$$\{\hat{\dot{c}}\}_c = \text{merge}_{m,n}(\dot{c}), \tag{6.39}$$

$$\hat{\dot{\pi}}_c = \dot{\pi}_m + \dot{\pi}_n, \tag{6.40}$$

$$\hat{\theta}_c \sim q(\hat{\theta}_c | x, \hat{\dot{c}}, \hat{\dot{\bar{c}}}), \tag{6.41}$$

$$\hat{\bar{O}}_c \sim p(\hat{\bar{O}}_c | x, \hat{\dot{c}}). \tag{6.42}$$

The function $\text{split}_c(\cdot)$ splits the label assignments of component c so that labels are assigned to the two new components m and n, whereas the function $\text{merge}_{m,n}(\cdot)$ does the reverse move and merges the label assignments of components m and n so that the respective data points are assigned to a new component c. Sampling new parameters given the new label assignments is done with the restricted Gibbs sampler. The sub-component auxiliary variables are sampled jointly with the variables for the regular components. This specific joint sampler is also called deferred MH sampler [18] and conveniently sets $q(\hat{\bar{O}}|x, \hat{O}) = p(\hat{\bar{O}}|x, \hat{O})$.

Components are marked as "splittable" after the variables show signs of a burn-in, which is the case when the likelihood $f(\{x\}^l_c, \theta^l_c) f(\{x\}^r_c, \theta^r_c)$ for all data points assigned to component c begins to oscillate with the iterations.

It can be shown [18] that the Hastings ratio for a component split can be expressed as

$$\begin{aligned} HR_{\text{c-split}} &= \frac{p(x|\hat{\dot{c}})}{p(x|\dot{c})} \frac{p(\hat{\dot{c}})}{p(\dot{c})} \frac{\alpha \Gamma(\hat{\dot{N}}_m) \Gamma(\hat{\dot{N}}_n)}{\Gamma(\dot{N}_c)} \\ &= \frac{f(x|\hat{\dot{c}}, \hat{\theta}_m) f(x|\hat{\dot{c}}, \hat{\theta}_n)}{f(x|\dot{c}, \theta_c)} \frac{\alpha \Gamma(\hat{\dot{N}}_m) \Gamma(\hat{\dot{N}}_n)}{\Gamma(N_c)}. \end{aligned} \tag{6.43}$$

Splits of codebook components affect the clusters that contain them as cluster components. All clusters that contain a split component require an update of their variables. The update step for clusters is subject to design, as there are various ways to execute it. The naive approach is to split the affected component within each cluster and update the corresponding variables. We opted for an update scheme that keeps the cluster sizes unchanged so as to separate codebook growth and mixture of mixtures growth entirely.

After performing a component split, we proceed as follows. All clusters that contain the respective component are duplicated with a function

$$(\{\hat{\tilde{c}}_k\}_c, \{\hat{\tilde{c}}_{k'}\}_c) = \text{duplicate}^c_k(\tilde{c}, \bar{c}), \tag{6.44}$$

where $\bar{c}$ is the vector of cluster sub-component labels.

The original cluster k keeps all unchanged components and the "left" split result. The duplicate k' also keeps all unchanged components and the "right" split result. Data points with $z_i = k$ that are assigned to the split component can be reassigned so that the ones labeled with $\bar{c}_{ki} = l$ belong to cluster k, and data points with and $\bar{c}_{ki} = r$ now belong to cluster k'. Data points that are assigned to unchanged components within the duplicated cluster however can not be reassigned unambiguously. Because a data point can never belong to two clusters at the same time, the duplication automatically invalidates the labels z_i and weights π. To re-establish a valid state for the sampler, we re-sample all variables. Therefore, the split sampler for components is the last step before the next iteration of the restricted Gibbs sampling in our implementation, as can be seen in Algorithm 3. Figures 6.5b and 6.5c illustrate this step.

6.4.2 Merge Moves

The Hastings ratio for a proposed component merge also requires a reverse proposal. Where there is only one way to merge two sets of label assignments, there are $2^{\dot{N}_c - 1} - 1$ ways to split a set of labels into non-empty partitions. However, since split proposals in this algorithm are determined by the sub-components, the Hastings ratio will be zero if the labels after a proposed reverse split do not match the pre-merge labels. Therefore, the probability for accepting a component merge rapidly diminishes with increasing number of assigned data points. This behavior is approximated by automatically rejecting all component merges.

In order to mitigate for slow convergence in certain situations, a random merge sampler is introduced instead to propose component merge moves whose reverse move is a random split, in contrast to the sub-component based deterministic split proposals. Two random components are sampled and a merge proposal is computed. The reverse split proposal is generated by a random partitioning of the data points assigned to the respective component. The split proposal will generally have a diminishing acceptance probability, whereas the corresponding merge move is much more likely to be sensible.

The Hastings ratio for a random merge proposal is as follows:

$$\begin{aligned} HR_{\text{c-merge}} &= \frac{p(x|\hat{\dot{c}})}{p(x|\dot{c})} \frac{p(\hat{\dot{c}})}{p(\dot{c})} \frac{\Gamma(\alpha)\Gamma(\dot{N}_m)\Gamma(\dot{N}_n)}{\Gamma(\alpha + \hat{\dot{N}}_c)\Gamma(\frac{\alpha}{2})^2} \\ &= \frac{p(x|\hat{\dot{c}})}{p(x|\dot{c})} \frac{\Gamma(\hat{\dot{N}}_c)}{\alpha\Gamma(\dot{N}_m)\Gamma(\dot{N}_n)} \frac{\Gamma(\alpha)\Gamma(\dot{N}_m)\Gamma(\dot{N}_n)}{\Gamma(\alpha + \hat{\dot{N}}_c)\Gamma(\frac{\alpha}{2})^2} \\ &= \frac{f(x|\hat{\dot{c}}, \hat{\theta}_c)}{f(x|\dot{c}, \theta_m) f(x|\dot{c}, \theta_n)} \frac{\Gamma(\alpha)\Gamma(\hat{\dot{N}}_c)}{\alpha\Gamma(\alpha + \hat{\dot{N}}_c)\Gamma(\frac{\alpha}{2})^2}. \end{aligned} \tag{6.45}$$

A random split of component $\hat{\dot{c}}$ is sampled for the reverse split proposal, therefore the weights for the split results are Dirichlet distributed.

Analogous to the split moves, all clusters that contain a merged component need to be updated. Fortunately, the cluster updates after component merges are much simpler. All clusters that contained any of the two merged components replaces the respective component with the merge result. Cluster labels z_i and weights π remain unchanged. Cluster component labels $\tilde{c}_{ki}$ and weights $\tilde{\pi}_k$ are updated if cluster k contains both components involved in the merge:

$$\{\hat{\tilde{c}}_k\}_c = \text{merge}_{m,n}(\tilde{c}_k), \tag{6.46}$$

$$\hat{\tilde{\pi}}_k^c = \hat{\tilde{\pi}}_k^m + \hat{\tilde{\pi}}_k^n. \tag{6.47}$$

New values for cluster sub-component auxiliary variables $\bar{O}_k = \{\bar{\pi}_k, \bar{c}_k\}$ are sampled for all clusters to be updated according to

$$\hat{\bar{O}}_k^c \sim p(\hat{\bar{O}}_k^c | x, \hat{\tilde{c}}_k). \tag{6.48}$$

Again, Figures 6.5b and 6.5c illustrate the outcome of this step.

6.5 Cluster Split, Merge and Switch Sampler

Cluster split, merge and switch moves modify the assignments of components to clusters. A split move splits a cluster – which is a mixture of components – into two smaller mixtures. A merge move merges two clusters into one larger mixture. A switch move moves one component from one cluster to another cluster. In all these cases the components themselves, i.e., their parameters, are not modified. Cluster moves rely on the local, cluster dependent component parameters to produce good move proposals for a MH step. Cluster moves are efficient because they can be easily computed based on the existing data partitions.

Let $M = \{\pi, \tilde{\pi}, z, \tilde{c}\}$ be the set of cluster and cluster component variables and $\bar{M} = \{\bar{\pi}, \bar{c}\}$ be the set of cluster sub-component variables. A new set of random variables $\{\hat{M}, \hat{\bar{M}}\}$ is proposed by any of the possible moves. The Hastings ratio for a move is of the form

$$HR = \frac{p(\hat{M}, x) p(\hat{\bar{M}} | x, \hat{\tilde{c}})}{p(M, x) p(\bar{M} | x, \tilde{c})} \frac{q(M, \bar{M} | \hat{M}, \hat{\bar{M}})}{q(\hat{M}, \hat{\bar{M}} | M, \bar{M})}. \tag{6.49}$$

As before, a proposed move is accepted with the probability defined in Equation (6.34). Note that the component parameters θ_c are not subject to updates during cluster moves. That is because neither assignments of data points to codebook components nor codebook component parameters are modified.

6.5.1 Split Moves

Analogous to component splits, where good split candidates are provided by auxiliary sub-components, we resort to the cluster components themselves as support for producing good proposals. To generate a good split proposal for cluster k, we consider all $2^{C_k-1}-1$ possible non-empty partitions into two separate mixtures and propose the most promising split, according to the Hastings ratio.

The computational overhead grows with the number of components in a cluster. For instance, a cluster with 16 components can be partitioned into two clusters in 32.767 different ways, which would require a same amount of computations for Hastings ratios. In order to control the expected maximum computational load for splits, we introduce a parameter $c_{\max}$ into the algorithm which caps the maximum size of clusters. Setting this parameter accordingly prevents clusters to grow too large. By setting the value arbitrarily high, mixtures may grow to any size. To maintain computation feasible for very large clusters as well, we could also limit the considered partitions to a random subset of possibilities smaller than the Stirling number above.

The cluster split sampler's design essentially follows the considerations of the component split sampler in Section 6.4.1. As is the case for components, we can also parallelize the cluster split move computations. The proposal distribution for a cluster split move is defined as follows. First, we randomly select a split move or a merge move $Q \in \{Q^m_{\text{k-split}}, Q^{a,b}_{\text{k-merge}}\}$, where $Q^m_{\text{k-split}}$ denotes a move for splitting cluster m into a, b, and $Q^{a,b}_{\text{k-merge}}$ is a merge of clusters a, b into m. New sets of model variables are sampled as follows, conditioned on Q. If $Q = Q^m_{\text{k-split}}$:

$$(\{\hat{z}\}_a, \{\hat{z}\}_b) = \text{split}_m(z, \tilde{c}), \tag{6.50}$$

$$(\hat{\pi}_a, \hat{\pi}_b) = \pi_m \cdot (\pi^a_m, \pi^b_m), \tag{6.51}$$

$$(\hat{\tilde{\pi}}^1_a, \dots, \hat{\tilde{\pi}}^C_a) \sim \text{Dir}(\hat{B}^1_a, \dots, \hat{B}^C_a), \tag{6.52}$$

$$(\hat{\tilde{\pi}}^1_b, \dots, \hat{\tilde{\pi}}^C_b) \sim \text{Dir}(\hat{B}^1_b, \dots, \hat{B}^C_b), \tag{6.53}$$

where $(\pi^a_m, \pi^b_n) \sim \text{Dir}(\hat{N}^a_m, \hat{N}^b_m)$. If $Q = Q^{a,b}_{\text{k-merge}}$:

$$\{\hat{z}\}_m = \text{merge}_{a,b}(z, \tilde{c}), \tag{6.54}$$

$$\hat{\pi}_m = \pi_a + \pi_b, \tag{6.55}$$

$$(\hat{\tilde{\pi}}^1_m, \dots, \hat{\tilde{\pi}}^C_m) \sim \text{Dir}(\hat{B}^1_m, \dots, \hat{B}^C_m), \tag{6.56}$$

with

$$\hat{B}_k^c = \begin{cases} \frac{\hat{N}_k^c + \beta}{C_k} & \text{if } \hat{N}_k^c > 0, \\ 0 & \text{else.} \end{cases} \tag{6.57}$$

The function $\text{split}_m(\cdot)$ splits the label assignments of cluster m so that labels are assigned to the two new clusters a and b, whereas the function $\text{merge}_{a,b}(\cdot)$ does the reverse move and merges the label assignments of clusters a and b so that the respective data points are assigned to a new cluster m. The component labels $\tilde{c}_{ki}$ do not require an update since component IDs are valid globally. To promote a more stable splitting behavior, clusters are marked “splittable” if all components within the respective clusters are also marked “splittable” (see Section 6.4.1).

We express the Hastings ratio for a cluster split proposal as follows:

$$\begin{aligned} HR_{\text{k-split}} &= \frac{p(x|\hat{z})}{p(x|\tilde{c})} \frac{p(\hat{z})}{p(z)} \frac{\Gamma(\beta + N_m)\Gamma(\beta \frac{\hat{N}_a}{N_m})\Gamma(\beta \frac{\hat{N}_b}{N_m})}{\Gamma(\beta)\Gamma(\hat{N}_a)\Gamma(\hat{N}_b)} \\ &= \frac{p(x|\hat{z})}{p(x|\tilde{c})} \frac{\beta\Gamma(\hat{N}_a)\Gamma(\hat{N}_b)}{\Gamma(N_m)} \frac{\Gamma(\beta + N_m)\Gamma(\beta \frac{\hat{N}_a}{N_m})\Gamma(\beta \frac{\hat{N}_b}{N_m})}{\Gamma(\beta)\Gamma(\hat{N}_a)\Gamma(\hat{N}_b)} \\ &= \frac{f(x|\hat{\tilde{c}}, \hat{\Theta}_a) f(x|\hat{\tilde{c}}, \hat{\Theta}_b)}{f(x|\tilde{c}, \Theta_m)} \frac{\beta\Gamma(\beta + N_m)\Gamma(\beta \frac{\hat{N}_a}{N_m})\Gamma(\beta \frac{\hat{N}_b}{N_m})}{\Gamma(N_m)\Gamma(\beta)}, \end{aligned} \tag{6.58}$$

with $\hat{\Theta}_k = \{\hat{\pi}_k, \{\theta\}_k\}$, and $\{\theta\}_k$ being the parameters of all codebook components that are also cluster components in k.

6.5.2 Merge Moves

For a prospective merge, two random clusters are sampled and a merge proposal is computed. A cluster merge is only permitted if the component size of the merge result is not exceeding $c_{\max}$. The reverse proposal is a random partition of the cluster components into two separate mixtures.

Analogous to the cluster split proposal above, the Hastings ratio for a cluster merge proposal is expressed as follows:

$$\begin{aligned} HR_{\text{k-merge}} = &\frac{f(x|\hat{z}, \hat{\Theta}_m)}{f(x|z, \Theta_a) f(x|z, \Theta_b)} \\ &\times \frac{\Gamma(N_a + N_b)\Gamma(\beta)}{\beta\Gamma(\beta + N_a + N_b)\Gamma(\beta \frac{N_a}{\hat{N}_m})\Gamma(\beta \frac{N_b}{\hat{N}_m})}. \end{aligned} \tag{6.59}$$

As mentioned earlier, Equation (6.11) suggests that the likelihood of a cluster is higher if the components of the respective cluster are located closer to each other in the feature space. During inference, β controls the importance of proximity for grouping nearby components into clusters. The Hastings ratios in Equations (6.58) and (6.59) suggest that with larger β the probability of accepting a split proposal becomes higher, and the probability of the reverse merge move becomes smaller.

6.5.3 Switch Moves

The component split and merge moves (see Section 6.4) produce clusters with shared components due to the specifics of the cluster update steps. After a component split, all clusters that contain the split component are duplicated. After a component merge, all clusters who contained the previous components now share the new component (unless a cluster already contained both original components).

The number of clusters that share the same component can grow very quickly if the algorithm decides to perform many component splits. Another factor is the maximal cluster size c_{max}. The larger clusters can get, the more likely they duplicate due to eventual component splits. In cases where a large amount of clusters overlap (that is, many clusters share the same components) and therefore cover the same region in the feature space, the algorithm can suffer from convergence issues due to high ambiguity during label sampling.

To mitigate this issue, we developed a switch move sampler that supports algorithm convergence. A switch move is an operation, where all data points that are assigned to component a in cluster m are re-assigned to the same component a in cluster n. Given a pair of clusters, we consider a switch move for a in both directions and propose the direction which is most promising, according to the Hastings ratio. Switch move proposals are sampled by a random switch sampler, where for a prospective move, two random clusters are sampled and the proposal is computed.

New sets of model variables are sampled as follows:

$$
\begin{aligned}
(\{\hat{z}\}_m, \{\hat{z}\}_n, \{\hat{\tilde{c}}\}_m, \{\hat{\tilde{c}}\}_n) &= \mathrm{switch}^a_{m,n}(z, \tilde{c}), && (6.60)\\
(\hat{\pi}_m, \hat{\pi}_n) &= (\pi_m - \pi_m \tilde{\pi}^a_m, \pi_n + \pi_m \tilde{\pi}^a_m), && (6.61)\\
(\hat{\tilde{\pi}}^1_m, \ldots, \hat{\tilde{\pi}}^C_m) &\sim \mathrm{Dir}(\hat{B}^1_m, \ldots, \hat{B}^C_m), && (6.62)\\
(\hat{\tilde{\pi}}^1_n, \ldots, \hat{\tilde{\pi}}^C_n) &\sim \mathrm{Dir}(\hat{B}^1_n, \ldots, \hat{B}^C_n). && (6.63)
\end{aligned}
$$

The function $\mathrm{switch}^a_{m,n}(\cdot)$ re-assigns all data points in component a of cluster m to component a of cluster n by updating the component labels $\tilde{c}_{ki}$ and the cluster labels z_i.

A switch move can be interpreted as splitting one component a off of a cluster m and merging a single-component cluster with a cluster n that already contains a as component. The Hastings ratio for a switch proposal is therefore expressed as follows:

$$
\begin{aligned}
&HR_{\mathrm{switch}}\\
&= \frac{p(x|\hat{z})}{p(x|z)} \frac{\beta\Gamma(N^a_m)\Gamma(\hat{N}_m)\Gamma(N_n)}{\Gamma(N_m)\Gamma(N_n)} \frac{\Gamma(\beta+N_m)\Gamma(\beta+N_n)}{\Gamma(\beta)}\\
&\times \frac{\Gamma(\beta\frac{N^a_m}{N_m+N_n})\Gamma(\beta\frac{\hat{N}_m}{N_m+N_n})\Gamma(\beta\frac{N_n}{N_m+N_n})}{\Gamma(N^a_m)\Gamma(\hat{N}_m)\Gamma(N_n)}\\
&\times \frac{\Gamma(\hat{N}_m)\Gamma(\hat{N}_n)}{\beta\Gamma(N^a_m)\Gamma(\hat{N}_m)\Gamma(N_n)} \frac{\Gamma(\beta)}{\Gamma(\beta+\hat{N}_m)\Gamma(\beta+\hat{N}_n)}\\
&\times \frac{\Gamma(N^a_m)\Gamma(\hat{N}_m)\Gamma(N_n)}{\Gamma(\beta\frac{\hat{N}_n}{N_m+N_n})\Gamma(\beta\frac{\hat{N}_m}{N_m+N_n})}\\
&= \frac{p(x|\hat{z})}{p(x|z)} \frac{\Gamma(\hat{N}_m)\Gamma(\hat{N}_n)}{\Gamma(N_m)\Gamma(N_n)} \frac{\Gamma(\beta+N_m)\Gamma(\beta+N_n)}{\Gamma(\beta+\hat{N}_m)\Gamma(\beta+\hat{N}_n)}\\
&\times \frac{\Gamma(\beta\frac{N^a_m}{N_m+N_n})\Gamma(\beta\frac{N_n}{N_m+N_n})}{\Gamma(\beta\frac{\hat{N}_n}{N_m+N_n})}. \qquad (6.64)
\end{aligned}
$$

The reverse move requires to split the data points that are assigned to component a in cluster n and assign these data points to cluster m. For the same reasons than in the case of normal component merges (see Section 6.4.2), the probability for the reverse proposal quickly approaches zero with increasing data size and the reverse move will be rejected. We approximate this behavior by automatically rejecting all reverse switch moves.

6.6 Use Case: Unsupervised Subword Modeling

In the following section, we will demonstrate the benefits of our DPMoMM sampler on real speech data in the use case of unsupervised subword modeling.

The objective of unsupervised subword modeling is to construct a representation of speech sounds that is robust to variation within and across speakers and that maximizes class discrimination [164]. Previous work of Chen et al. [22, 23] and ourselves [68, 67, 69] already achieved good results using a DPMM sampler to tackle this task in the context of the zero resource speech challenges [164, 34]. The general procedure is to cluster speech feature vectors into classes by sampling a Dirichlet process *Gaussian* mixture model (DPGMM) [20], which is an infinite Gaussian mixture model (IGMM) [138]. Speech would then be represented by frame-level posteriorgrams, for instance, or simply by a sequence of textual class labels, where the classes are the components in the sampled mixture.

The official evaluation metric for the zero resource speech challenge is the minimal pair ABX phone discriminability between phonemic minimal pairs [141] (see Section 4.5). Another popular metric for evaluating speech representations especially with respect to information and sequentiality is the normalized mutual information (NMI). We use both metrics to evaluate the output of our sampler in the following section. The details of the evaluation measures are explained in this section.

6.6.1 ABX Phone Discriminability

The ABX phone discrimination error is computed according to Equation (4.4). As a reminder, $d(\cdot, \cdot)$ in this equation is the DTW distance defined over sequences of frame based speech representations (posteriorgrams, textual labels, etc.). Any proper distance measure can be used for computing the DTW distance. For the experiments in this Chapter, we evaluate two types of representations, textual labels and posteriorgrams. For the comparison of label sequences, we use the Levenshtein distance ($\mathrm{ABX}_{\mathrm{ls}}$). For evaluating posteriorgram sequences, we use the Kullback-Leibler divergence ($\mathrm{ABX}_{\mathrm{kl}}$). We collect discrimination errors for all possible pairings of phone triplets and average them over all contexts for a given pair of central phones, over all pairs of central phones and over all speakers.

In Section 4.5, we argued that the strength of the ABX discriminability task as evaluation method is that any number of inferred classes can be evaluated such that the quality of very different representations of the same underlying data can be compared fairly. In other words, the ABX test is indifferent to the amount of inferred classes and therefore neither rewards reasonably sized, nor penalizes excessively large class inventories. If for instance an inferred speech representation is defined over tens of thousands of classes, the ABX discriminability error will be low as long as these classes are discriminable from each other. In the context of modeling speech, it might however be desired to find a more reasonable number of classes, for instance in the size of phone sets. The symmetric normalized mutual information criterion introduced in the following subsection is an alternative metric that can be used to consider the aspect of vastly different inventory sizes.

6.6.2 Symmetric Normalized Mutual Information

The mutual information of two random variables is a measure for mutual dependence. Normalized mutual information is defined as

$$\mathrm{NMI}(X;Y) = \frac{H(X) - H(X|Y)}{H(X)} \tag{6.65}$$

and gives a measure of how good X can be predicted, given the knowledge about Y. In our use case, X corresponds to the random variable over the "true" distribution of sounds, approximated by the sequence of phones for any target data set. Y is the random variable over the estimated distribution of sounds, given for instance by the label output of a DPMM sampler for the same data. $H(X)$ is the entropy of the true transcription and is used as normalizing factor in the denominator.

NMI$(\cdot)$ is not symmetric, which makes the comparison of random variables that are defined over very different inventory sizes difficult. To guarantee a fair comparison, we suggest to use a symmetric NMI [110] of the form

$$\mathrm{NMI}_{\max}(X;Y) = \frac{H(X) - H(X|Y)}{\max(H(X), H(Y))} \tag{6.66}$$

The inventory sizes of the compared representations directly affect the metric. When comparing sequences defined over any newly defined set of classes to sequences of phone labels, the symmetric NMI score gives an intuition of the quality of the new representations in terms of being "phone-like". With this, we have a fair measure of comparability at hand for the frame-level phone transcriptions of any target data and the frame-level label output of a DPMM or DPMoMM.

6.7 Experimental Evaluation

6.7.1 Data

We use two separate data sets for American English and for Xitsonga known from the zero resource speech challenge [164], as described in Section 4.6.2. Because the sets vary in size, language and speech quality, comparable experimental results should be a good indicator for the robustness of our sampler.

From each of the two data sets, we extract two sets of speech observations. Specifically, from each data set, we extract two types of frame-wise feature vectors using the Kaldi speech recognition toolkit [132]. We can use about 1.7M frames for English and 0.8M frames for Xitsonga as input to the DPMM and DPMoMM samplers. The frame width is 25 milliseconds and the frame shift is 10 milliseconds. The first type of features is perceptual linear predictive (PLP) speech feature vectors [72] with first and second order derivatives (PLP+Δ+$\Delta\Delta$). The second type is stacked PLP vectors that were transformed unsupervisedly by linear discriminant analysis (PLP+LDA) with the method described in Chapter 4. LDA is commonly used in speech recognition to optimize speech features towards discriminability [60]. We conduct experiments for each of the two input feature vector types. Overall, we conduct all our experiments four times, once for each data set and feature vector type.

6.7.2 Procedure

We use our DPMoMM sampler to cluster a set of speech feature vectors into classes. In our use case, our algorithm jointly samples *Gaussian* mixtures as well as a global codebook of Gaussians. This codebook is precisely what the original DPMM sampler of Chang et al. [18] would infer. Due to the joint sampling we can directly compare the modeling quality of inferred Gaussian mixtures and the single Gaussians, or in other words the DPMoMM and the codebook, i.e., a DPGMM/IGMM.

After sampling the DPMoMM, the data can be represented by frame-wise labels. We use the symmetric NMI to compare the quality of label sequences. For that, we compute the symmetric NMI once using the labels for clusters and once using the labels for codebook components and calculate the relative improvement. In the same way, we also compare the ABX phone discriminability using the frame-wise labels as representation and calculate the relative improvement from using cluster labels instead of codebook component labels.

The ABX phone discriminability can also be computed for posteriorgrams as representation for the data. In that case, either a posteriorgram over clusters or over codebook components is computed for each speech frame. The two kinds of posteriorgrams are scored and compared to get a value for the relative improvement by using *cluster posteriorgrams* instead of *component posteriorgrams*.

We run every sampling for 1000 iterations. Each sampling step is parallelized across 30 threads. For all conducted experiments we sample each model 5 times, score each output and average the results. It is known that the influence of α diminishes in very high data regimes [18]. Chen et al. [22] conducted an experience study and confirmed that the value of α does not impact the outcome of sampling a DPMM given high dimensional speech feature vectors. Their samples are extracted from the same data sets that we use for our experiments and are similar in nature. In several informal experiments in we also observed this behavior and therefore set $\alpha = 1$ for all our experiments.

6.7.3 The Impact of β

We compare the use of clusters versus using the codebook components as model for the underlying data. The latter corresponds to output that the original sampler of Chang et al. [18] produces. We test on both data sets, English and Xitsonga, and use either PLP+LDA features or PLP+Δ+$\Delta\Delta$ as input. Figures 6.9 and 6.10 show the relative improvements that our proposed method achieved as contour plots.

We observed that using a very small value for the mixture concentration parameter β tends to result in few sampled mixtures that each contain a maximum amount of components. In other words, the sampler is over-confident in grouping components together based on minimal proximity. In contrast to synthetic data, real data tends to be comprised of overlapping classes. The aggressive grouping is one of the consequences of this fact. With larger β, we observed that fewer

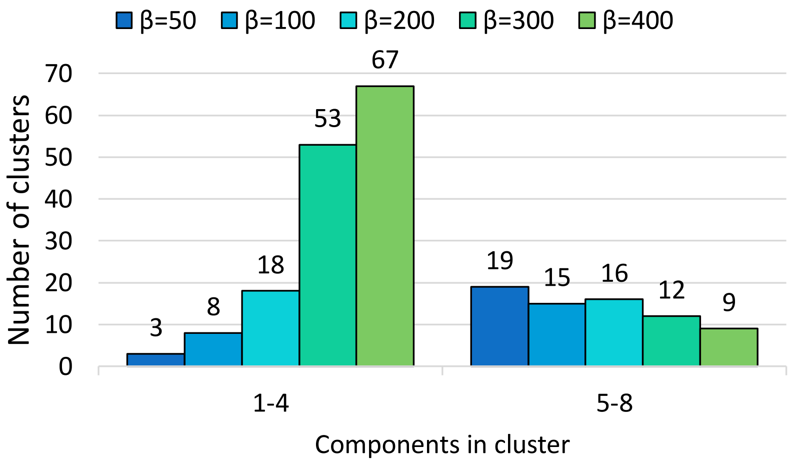


Figure 6.7: Distribution of cluster sizes by the example of sampling Xitsonga data for 500 iterations. Higher β values lead to more clusters with fewer components. The same behavior was observed on the English data set.

components are grouped together to form mixtures, which results in a larger number of clusters that contain fewer components. Figure 6.7 exemplarily plots the distribution of cluster sizes for the Xitsonga data with different values for β. Figures 6.9e, 6.9j and 6.10e, 6.10j show that the number of clusters approximates the number of codebook components as β increases.

The behavior of the cluster inference dependent on β is best explained by analyzing the sampling of weights during restricted Gibbs sampling and the Hastings ratio for cluster split and merge moves. According to Equations (6.4)-(6.5), (6.7)-(6.8) and (6.16)-(6.17), the distributions of cluster and cluster component weights are governed by β, whose impact is twofold. Its value determines the probability mass that is reserved for generating a new cluster by the split sampler, and it regulates the weights of the cluster components. A large β will motivate the generation of more clusters and cause cluster component weights to take on more similar values, therefore keeping more cluster components alive for a longer time. This in turn encourages more cluster splits, which is also reflected in the Hastings ratios for cluster moves. With larger β, Equation (6.58) takes on a larger value, i.e., the probability of accepting a split proposal becomes higher, and Equation (6.59) takes on a smaller value, i.e., the probability of the reverse merge move becomes smaller. Intuitively, the effect is that only closely related components

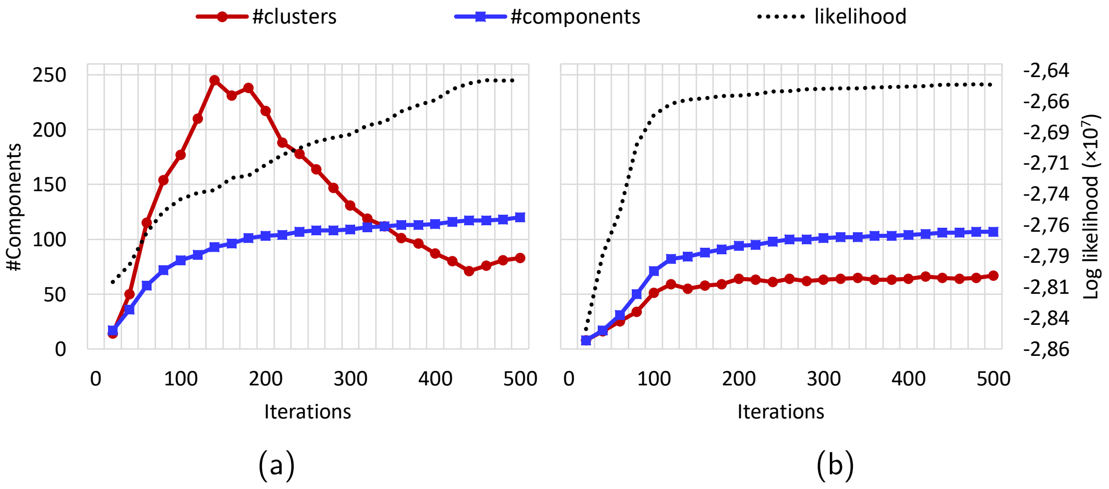


Figure 6.8: Sampling behavior on Xitsonga data when *(a)* not using switch moves for cluster components, and *(b)* using switch moves. Without switch moves, the number of clusters might grow fast, and convergence is slow. With switch moves, this issue does not occur. The same behavior was also observed on the English data set.

remain grouped in form of a cluster, and less dense clusters are likely to be split to form new clusters with less components. The resulting DPMoMM tends to be made up by many clusters with mostly low amounts of components, and only few clusters with higher amounts of components, if the data suggests so. A small value for β has the exact opposite effect and a sampled DPMoMM will have few clusters with mostly large amounts of cluster components.

6.7.4 Convergence and the Switch Sampler

During our experiments, we found that under certain conditions, the number of clusters can grow rapidly and stay large for a long stretch of iterations. This is the case when the allowed cluster size $c_{\max}$ is large and β is set to facilitate many clusters with a large number of components. Under such circumstances, the duplication of clusters during the cluster update step in the component split sampler (see Section 6.4.1) becomes more likely and more frequent, especially if multiple components in the same cluster are subject to splitting.

We developed the switch move sampler to mitigate this issue and support convergence. Figure 6.8 exemplarily shows the sampling behavior of our DPMoMM sampler on Xitsonga data with and without using switch moves for cluster components. As can be seen, the number of clusters might grow fast without switch moves, and convergence is slow. With switch moves, however, this issue does not occur. The same behavior was observable for both of our data sets. Without switch moves, a considerably larger amount of time would be required by the sampler to converge. With switch moves, convergence is fast.

6.7.5 Use Case Performance

Figures 6.9a, 6.9f and 6.10a, 6.10f show the relative improvements of cluster label sequences over codebook component label sequences on the symmetric NMI metric. The best results are always achieved by allowing larger clusters. The optimal value for β seems to lie within a certain range. This range is the same for our two data sets, English and Xitsonga, but it seems to be input feature dependent. Sampling from PLP+Δ+$\Delta\Delta$ features benefits from a β value between 100 and 300, where sampling from PLP+LDA features might even benefit from a β larger than 400, which during our experiments was the highest value that we tested. These observations transfer to the evaluation of cluster and component labels with the ABX phone discriminability test, as can be seen in Figures 6.9b, 6.9g and 6.10b, 6.10g.

Interestingly, we observed considerable performance improvements when we extracted cluster posteriorgrams and compared them to component posteriorgrams with the ABX discriminability test, especially with PLP+Δ+$\Delta\Delta$ as input (see Figures 6.9c, 6.9h and 6.10c, 6.10h). This is noteworthy because despite the much lower dimensionality of cluster posteriorgrams, performance not just equals, but even increases, compared to the component posteriorgrams. This is an indicator that the clusters inferred by our proposed DPMoMM are not just accumulations of related classes, but in fact a better approximation to the real underlying multimodal distributions of more complex classes in the data.

The number of inferred clusters does not necessarily correlate with the average length of sample sequences that use the clusters as classes. Figures 6.9d, 6.9i and 6.10d, 6.10i show that it is the maximum number of allowed components per cluster $c_{\max}$ that governs the average unit length, rather than β. The average unit lengths is higher if the clusters are allowed to grow larger. A too large β somewhat slows down this trend. The increased unit lengths for larger clusters is

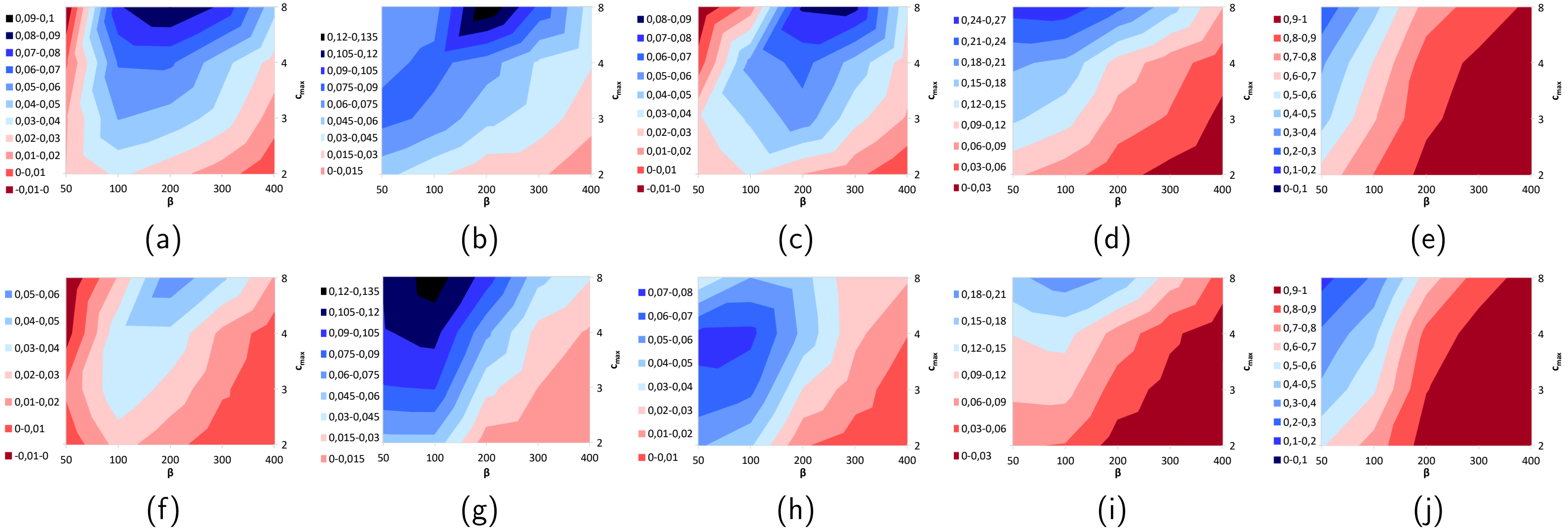


Figure 6.9: Relative performance improvements when interpreting clusters instead of codebook components as acoustic classes. The inputs are 39 dimensional PLP+Δ+ΔΔ speech features. *(a)-(e)*: Results on Xitsonga; *(f)-(j)*: Results on English; *(a),(f)* Relative improvement of $\mathrm{NMI}_{\mathrm{max}}$, *(b),(g)* Relative improvement of $\mathrm{ABX}_{\mathrm{ls}}$, *(c),(h)* Relative improvement of $\mathrm{ABX}_{\mathrm{kl}}$, *(d),(i)* Relative unit length increase, *(e),(j)* Cluster to component amount ratio. Paired t-tests on all ABX task outputs yield $p \ll 0.0001$.

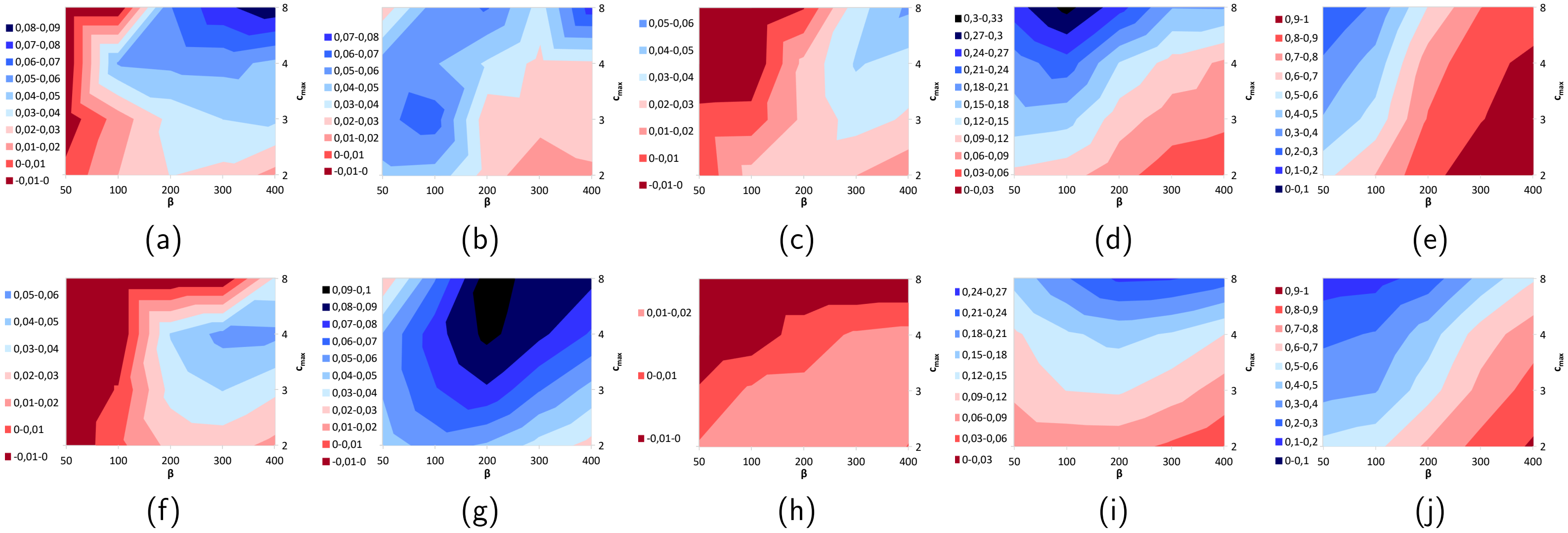


Figure 6.10: Relative performance improvements when interpreting clusters instead of codebook components as acoustic classes. The inputs are 20 dimensional PLP+LDA speech features. *(a)-(e)*: Results on Xitsonga; *(f)-(j)*: Results on English; *(a),(f)* Relative improvement of NMI_{max}, *(b),(g)* Relative improvement of ABX_{ls}, *(c),(h)* Relative improvement of ABX_{kl}, *(d),(i)* Relative unit length increase, *(e),(j)* Cluster to component amount ratio. Paired t-tests on all ABX task outputs yield $p \ll 0.0001$.

an indicator that the grouping of components into mixtures allows the mixture of mixtures model to capture wider acoustic phenomena, an ability that is highly valued in unsupervised subword modeling.

Overall, our results as depicted in Figures 6.9 and 6.10 reveal that with some tuning of β and $c_{\max}$, relative improvements on all evaluation metrics can be as high as 13.5%. The contour plots also suggest that larger mixtures might result in even larger improvements. It is a strong support for our initial motivation for this work that good results can be achieved with settings that lead to the inference of as little as 27% of the amount of clusters than there are codebook components, which considerably reduces model complexity. At the same time, the found clusters can serve as models with average durations of sound instances that are up to 31% longer. Table 6.1 compares the performance of standard DPMMs and good DPMoMMs that we sampled in absolute numbers. During our experiments we sampled models that are best on one metric, but sub-optimal on others. The exemplary DPMoMMs perform reasonably well on all evaluation metrics and give a good impression of what can be expected if the parameters are tuned reasonably. Note that we did not yet exhaust the exploration of the hyper-parameter space. We expect that even better results are possible with larger values for β and especially $c_{\max}$.

6.7.6 State-of-the-art ZeroSpeech Performance

In Chapter 4, we transformed PLP features with LDA, maximum likelihood linear transforms (MLLT) [56, 49] and feature-space maximum likelihood linear regression (fMLLR) [4, 48] – a method commonly used for speaker adaptation – to improve the input to a DPMM sampler. The extracted posteriorgrams achieved the to-date best results on the zero resource speech challenge 2015 Xitsonga and English data sets.

We repeated these experiments with our novel sampler, using the hyper-parameters that we tuned for the PLP+LDA features. Table 6.1 compares the previously published performance of [67] (as in Chapter 4) with using a DPMoMM for sampling instead. We could infer more compact models having fewer clusters and at the same time reduce the discriminability errors – the official evaluation method of the challenge – even further, thus establishing a new state-of-the-art with our proposed method.

Table 6.1: Performance comparison of standard DPMMs and proposed DPMoMMs applied to both data sets. K and C are annotated with the standard deviation over 5 samplings for each model type. It can be seen that the standard deviation correlates with the quality of the speech data. Indices for feature types denote context size, indices for transformations denote output dimensionality. Paired t-tests on all ABX task outputs yield $p \ll 0.0001$.

Features	Sampler	Clusters modeled by	β	$c_{\max}$	K	C	K/C	$\text{NMI}_{\max}$	ABX_{ls}	ABX_{kl}	avg. seg. len.
					Results for the Xitsonga data set						
PLP+Δ+ΔΔ	DPMoMM	Single Gaussians	-	-	154±5	-	1.0	0.275	21.31%	14.03%	2.16 frames
		Gaussian Mixtures	200	8	114±4	154±5	**0.74**	**0.302**	**18.44**%	**12**%	**2.76** frames
$\text{PLP}_4+\text{LDA}_{20}$	DPMoMM	Single Gaussians	-	-	142±5	-	1.0	0.283	20.29%	13.15%	2.22 frames
		Gaussian Mixtures	400	8	120±4	142±5	**0.848**	**0.31**	**18.74**%	**12.47**%	**2.68** frames
$\text{PLP}_4+\text{LDA}_{20}$ +MLLT+fMLLR	DPMM [67]	Single Gaussians	-	-	139	-	-	-	-	12.2%	-
	DPMoMM	Single Gaussians	-	-	144±3	-	1.0	0.293	19.33%	12.44%	2.29 frames
	DPMoMM	Gaussian Mixtures	*400*	*8*	126±4	144±3	**0.876**	**0.315**	**18.06**%	**11.93**%	**2.7** frames
					Results for the English data set						
PLP+Δ+ΔΔ	DPMoMM	Single Gaussians	-	-	232±10	-	1.0	0.232	28.85%	18.99%	2.06 frames
		Gaussian Mixtures	100	4	110±11	232±10	**0.473**	**0.241**	**25.65**%	**17.61**%	**2.37** frames
$\text{PLP}_4+\text{LDA}_{20}$	DPMoMM	Single Gaussians	-	-	152±7	-	1.0	0.248	25.88%	16.18%	2.33 frames
		Gaussian Mixtures	300	4	98±8	152±7	**0.649**	**0.262**	**23.74**%	**15.99**%	**2.73** frames
$\text{PLP}_4+\text{LDA}_{20}$ +MLLT+fMLLR	DPMM [67]	Single Gaussians	-	-	156	-	-	-	-	15.7%	-
	DPMoMM	Single Gaussians	-	-	157±6	-	1.0	0.253	25.28%	15.84%	2.31 frames
	DPMoMM	Gaussian Mixtures	*300*	*4*	93±8	157±6	**0.591**	**0.266**	**23.28**%	**15.47**%	**2.64** frames

6.7.7 Acoustic Unit Recognition Performance

In Chapter 5, we conducted experiments where the output of a DPMM sampler served as labels for training a context dependent "triphone" acoustic unit recognizer. In the proposed framework, the DPMM defines the number and distributions of units dynamically and without any prior supervision, given only extracted speech observations, i.e., frame-based feature vectors. We trained an HMM acoustic model and n-gram language model using collapsed DPMM component labels. The resulting DPMM-HMM acoustic unit recognizer was evaluated by solving the ABX sound class discriminability task. Our results showed that it is possible to build a DPMM-HMM acoustic unit recognizer that is competitive with supervisedly trained phone recognizers.

Here, we conducted experiments with a comparable setup, but using the DPMoMM for subword modeling instead. The labels that we used come from the models that were inferred from PLP+fMLLR-transformed features as listed in Table 6.1. We trained language-dependent acoustic unit recognizers given the DPMoMM cluster labels. We then compared the performance to recognizers that were trained on the DPMoMM component labels instead, which correspond to standard DPMM labels. Since we sampled each DPMoMM five times, we also trained each recognizer five times, scored five times and averaged the results. In addition to the ABX test, we also evaluated the NMI scores. Training and testing was done with 12-fold cross-validation. Table 6.2 compares the results of the trained acoustic unit recognizers using cluster labels vs. using single component labels as training references. The recognizers trained on cluster labels perform significantly better across data sets and across all evaluation metrics.

6.8 Conclusion

We developed a Dirichlet process mixture of mixtures model (DPMoMM) sampler that jointly infers a codebook of global components and a mixture of clusters. The clusters are themselves mixtures that are defined over the codebook components. Split and merge samplers for modifying codebook components were complemented with split and merge samplers for modifying clusters. We introduced an additional switch sampler for cluster components to support and accelerate convergence, which has shown to be effective in experiments on real

Table 6.2: Performance comparison of acoustic unit recognizers trained on DPMM and DPMoMM labels. Paired t-tests on all ABX task outputs yield $p \ll 0.0001$.

Label type	Units	NMI_{max}	ABX_{ls}	ABX_{kl}	avg. seg. len.
Results for the Xitsonga data set					
Gaussians	144	0.319	18.06%	12.09%	2.59 frames
Mixtures	126	**0.327**	**16.45**%	**11.17**%	**2.95** frames
Results for the English data set					
Gaussians	157	0.263	24.48%	16.5%	2.77 frames
Mixtures	93	**0.278**	**22.21**%	**15.76**%	**3.12** frames

data. We demonstrated in the use case of unsupervised subword modeling on two separate data sets, that classes represented by a mixture of mixtures model reliably outperform the output of a standard DPMM. For both data sets and on all our evaluation metrics, the models inferred with our proposed DPMoMM sampler consistently achieved significant performance improvements of up to 13.5% relative. At the same time, considerably fewer classes that show longer average durations were sampled, a behavior that is desired for unsupervised subword modeling. Our experiments suggest that the inferred mixture of mixtures approximates the true underlying distributions of our experimental data much better than a standard DPMM. Lastly, parallelization of sampling steps allows the algorithm to work well even on larger data sets in higher data regimes. Although we originally developed the DPMoMM for improving unsupervised subword modeling, we think that our sampler will be useful not only for handling speech data, but for many tasks where classes are defined by multimodal distributions.

Chapter 7

Conclusion and Future Work

> *"Alle Sprache ist Bezeichnung der Gedanken, und umgekehrt die vorzüglichste Art der Gedankenbezeichnung ist die durch Sprache, dieses größte Mittel, sich selbst und andere zu verstehen."* [1] *[85]*
>
> – Immanuel Kant (1724-1804), *Philosopher*

7.1 Conclusion

This thesis addressed the problem of unsupervised subword modeling in the zero resource scenario. This work has been motivated by the need of novel methods for handling speech data from severely under-resourced languages, such as non-written languages and languages without significant digital presence. The unsupervised subword modeling problem was divided into the sub-tasks of representation learning and model design.

The representation learning problem was addressed by proposing a feature-optimized Bayesian non-parametric clustering method to infer a dynamic set of speech sound classes from raw data. The introduced method utilizes the Dirichlet process Gaussian mixture model and exploits a supervised training framework for learning useful feature transformations without prior supervision. These transformations reduce variance and dimensionality of the raw data and improve class

[1] "All language is expression of thought, and conversely, the most excellent way of expressing thought is through language, this greatest means of understanding oneself and others."

discriminability of the inferred speech sound classes.

The successful construction of an acoustic unit tokenizer showed that the inferred sound classes carry meaning and can be used to solve higher-level tasks. Analyzing the nature of the inferred classes pointed towards the model design problem.

The model design was improved by developing a novel Bayesian nonparametric model and sampler, called the Dirichlet process mixture of mixtures model (DPMoMM). The DPMoMM is a hierarchical model whose sampler infers a mixture of multimodal distributions. In the case of unsupervised subword modeling, this model enables the inference of complex acoustic classes that are represented by multimodal distributions, an approach that is closer to the practice in automatic speech recognition. The proposed DPMoMM is a general method that could also be applied to solve other tasks involving data that is modeled by multimodal distributions.

Posteriorgram representations of speech constructed by the feature optimized DPGMM clustering achieved considerably higher phone discriminability accuracies than all established baselines. The DPMoMM method improved discriminability of the unsupervisedly learned classes even further by inferring more complex models from acoustic data, which results in a better posteriorgram representation. In experiments, the proposed design led to the inference of fewer classes that represent subword units more consistently and show longer durations, which is a first step towards a fully unsupervisedly learned model for speech that represents units of appropriate length and complexity. The proposed methods set the state-of-the-art in the ZeroSpeech challenges 2015 and 2017.

The results presented in this work provide a good basis for further research on the possibilities of learning acoustic units and acoustic features from scratch, without any prior category knowledge or other meta information about the target language. Figure 7.1 illustrates the scope of this thesis with regards to the addressed problems. This larger perspective shows that there are still problems to solve, especially related to context coverage by inferred acoustic units and the explicit handling of sequentiality in speech data.

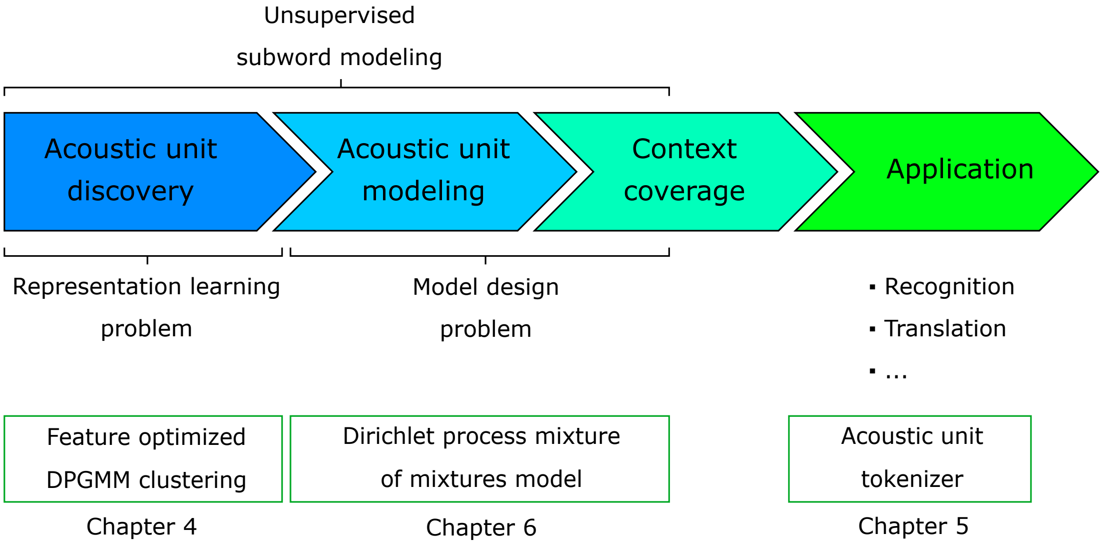


Figure 7.1: Larger perspective on unsupervised subword modeling in the zero resource scenario and its hierarchy of problems. The green boxes denote the contributions of this thesis towards solving the stated problems. Yet unsolved or only partially solved problems provide opportunities for future work.

7.2 Future Work

Although this thesis covers all aspects of the unsupervised subword modeling problem, there are certain sub-tasks that remain challenging and provide many possibilities for further refinement of developed methods. The following three challenges are particularly interesting and their tackling would be a logical continuation of the presented work.

7.2.1 Context Information for Unsupervised Subword Modeling

The research conducted within the frame of this thesis has shown that units which are inferred by Dirichlet process mixture model samplers show very short durations, compared to sound categories that were defined by human experts. This can be attributed to two causes, the model design and the method that is being used for model inference. Using a DPGMM to cluster speech features and interpreting the inferred GMM as modeling a set of acoustic categories is an expression of a fairly simple modeling assumption for speech data. Single Gaussians can not adequately approximate more complex speech sounds in a way that would result in models that cover a larger context in the temporal domain. Because of its simplicity, Dirichlet process mixture models in general tend to overestimate the fragmentation of the feature space. The DPMoMM was proposed to mitigate the over-simplification and managed to find clusters that represent more complex acoustic phenomena that show longer durations in the temporal space.

One unresolved issue that still persists is that neither the models proposed and utilized here, nor their inference consider the sequential nature of speech yet. Methods such as Dirichlet processes with dependencies [105, 106, 184, 57, 16] do account for temporal relations in data, but the problem of the model design still remains. The ability to account for temporal information and to handle sequential input with at the same time sufficient model complexity might help to infer more natural units with more complex distributions and longer durations.

One possible way to enable sequential modeling could be to utilize dependent non-parametric processes as priors. For instance, the dependent processes of [105, 106] generalize the standard DP to allow for a *collection* of non-parametric distributions, where their *realizations* are dependent, and the time-sensitive DP-MMs of [184] define the prior probability of assigning observations to clusters, given the history of previous assignments. Another possibility could be to define sequential models as components in a DPMM or as clusters in a DPMoMM. Lee et al. [96] jointly infer a segmentation of data and HMMs to approximate groups of segments, thereby implicitly modeling sequentiality within classes. Siu at al. [152] propose building ASR systems given raw speech data only by formulating the HMM acoustic model training as optimization problem over the parameter as well as the transcription sequence space. One possible extension to Dirichlet processes one might think of are mixture models with HMMs instead of Gaussians as components, which would similarly enable intra-class sequential modeling.

7.2.2 Context Aware Joint Model for Subword and Word-like Units

Once acoustic units are inferred from raw data, one could proceed by further grouping these units into categories that span larger regions in the feature space or the temporal space. Successive clustering could be used to group acoustic units into more general sound categories (e.g., context dependent units to context independent units) or more complex sound categories (sub-phone like units to phone-like units). Successive clustering however has the disadvantage that it provides an opportunity for new errors to be introduced into the final models. With *joint modeling* however there exists a more elegant method to learn more complex units on the basis of simpler ones.

In light of this work, the next challenge could therefore be to formulate a context aware joint model for subword and word-like units. Work such as [27, 131, 96] already learn word-like units that are made up by shorter morphemes. The goal of these models however usually is to provide a lexicon of word-like tokens, and explicitly modeling subword units remains unaddressed. A promising future direction is to define a joint model that can capture subword units and word-like entities alike. One key property of such joint models would have to be context awareness to the extent that sequential lexical information is captured and modeled jointly with the inferred subword and lexical units, which evades some of the problems that arise in concatenated processing of models that become more complex over time.

7.2.3 Fully Unsupervised Automatic Speech Recognition

The main research question that still stands is, can we teach machines to learn languages from raw speech only, without any supervision at all? The context-aware joint model mentioned above would provide a good starting point for the overarching challenge of building a fully functioning automatic speech recognizer without any prior supervision, nor knowledge (or with extremely little knowledge) about target data, expected sound categories and lexical characteristics. This task lies at the extreme end of unsupervised speech processing and could be considered the best match to the challenge of human language acquisition [53]. Language acquisition by humans is a multimodal learning process, and it would therefore be natural to extend machine language acquisition frameworks by incorporating multiple modalities, i.e., sensory inputs into the learning process, similar to works such as [140, 65, 64].

The zero resource scenario has seen steadily increasing interest in recent years, and first major contributions have laid the foundations for many interesting future works. With new evaluations, workshops and special sessions added to the roster each new year, the zero resource scenario will stay challenging and be the motivation for a manifold of contributions to the speech processing and linguistic research community in the years to come.

Appendix

A Results of the Zero Resource Challenge 2015

This appendix lists the official results for track 1 of the zero resource speech challenge (ZeroSpeech) 2015 [164]. The numbers for the official submissions are complemented by performance evaluations of methods that were published during the post-challenge time, i.e., after conducting the official evaluation as part of Interspeech 2015.

The goal for track 1 of ZeroSpeech 2015 was to discover subword units from raw speech. The provision of unified and open source suite of evaluation metrics as well as data sets supports fair comparisons of unsupervised linguistic unit discovery algorithms.

The metric used for evaluation and comparison is the ABX phone discrimination error rate (see Section 4.5 for details). The used data sets are the ones described in Section 4.6.2. The baseline was provided by raw MFCC speech features as speech representations. The topline was established by extracting posteriorgrams from supervisedly trained language dependent GMM-HMM based ASR systems [164].

Overall, there were 10 officially submitted unsupervised systems. 7 supervised systems were also listed in the leaderboard[1]. The official submissions are by (in alphabetical order) Badino et al. [8], Baljekar et al. [9], Chen et al. [22], Renshaw et al. [139] and Thiolliere et al. [160]. Post-challenge publications are by *Heck et al.* [68, 71, 66], Srivastava et al. [153] and Zeghidour et al. [182].

[1] http://www.zerospeech.com/2015

Table A.1: ABX discriminability errors across and within speakers for the track 1 submissions of ZeroSpeech 2015. Overall, there were 10 officially submitted unsupervised systems. 7 supervised systems are listed separately at the bottom of the table. The table further includes various systems that were published post-challenge (*post-ch.*).

Systems	English		Xitsonga		unsup.	post-ch.
	across	within	across	within		
Baseline	28.1	15.6	33.8	19.1	-	-
Topline	16.0	12.1	4.5	3.5	-	-
Badino et al. [8]	26.3	17.3	23.6	14.1	✓	✗
Badino et al. [8]	26.8	16.7	27.4	16.0	✓	✗
Badino et al. [8]	28.7	19.7	26.4	17.1	✓	✗
Baljekar et al. [9]	29.5	16.7	33.9	19.7	✓	✗
Baljekar et al. [9]	28	17	30.7	19.7	✓	✗
Chen et al. [22]	26.8	17.2	30.8	19.6	✓	✗
Chen et al. [22]	16.3	10.8	17.2	9.6	✓	✗
Heck et al. [68] (Ch. 4)	16.0	10.6	12.6	8.0	✓	✓
Heck et al. [71] (Ch. 4)	15.6	10.5	12.2	8.4	✓	✓
Heck et al. [71] (Ch. 4)	14.9	10.0	11.7	8.1	✓	✓
Heck et al. [66] (Ch. 5)	15.1	11.1	11.6	8.2	✓	✓
Heck et al. [70] (Ch. 6)	15.4	10.5	11.9	8.0	✓	✓
Heck et al. [70] (Ch. 6)	15.7	11.3	11.1	7.8	✓	✓
Renshaw et al. [139]	21.1	13.5	19.3	11.9	✓	✗
Renshaw et al. [139]	N/A	N/A	18.5	11.6	✓	✗
Srivastava et al. [153]	28	15.5	30	19	✓	✓
Srivastava et al. [153]	24	14	30	19	✓	✓
Thiolliere et al. [160]	17.9	12.0	16.6	11.7	✓	✗
Zeghidour et al. [182]	17	11	15.8	12	✓	✓
Baljekar et al. [9]	29.8	18.4	29.7	18.1	✗	✗
Baljekar et al. [9]	N/A	N/A	46.0	42.8	✗	✗
Baljekar et al. [9]	N/A	N/A	46.4	44.1	✗	✗
Chen et al. [22]	15.8	10.4	15.5	12.0	✗	✗
Chen et al. [22]	14.9	9.7	15.0	9.5	✗	✗
Renshaw et al. [139]	18.1	12.8	19.3	14.4	✗	✗
Renshaw et al. [139]	19.3	14.0	18.2	13.0	✗	✗

B Results of the Zero Resource Challenge 2017

This appendix lists the official results for track 1 of the zero resource speech challenge (ZeroSpeech) 2017 [34]. Track 1 of the follow-up challenge to the ZeroSpeech 2015 aimed at constructing systems for subword unit discovery that generalize across languages and adapt to new speakers.

Specifically, the two main innovations in 2017 were that the evaluation tests how well (1) systems and hyper-parameters generalize to new, unseen languages, and (2) how well the trained systems, i.e., their parameters adapt to new, unseen speakers [34]. In addition to that, data sets were designed to reveal whether systems scale well with their sizes.

The metric used for evaluation and comparison is again the ABX phone discrimination error rate (see Section 4.5 for details) to maintain consistency with the previous challenge. The used data sets are the ones described in Section 4.7.2. The challenge data comes as a training/test split, but on a higher level, which is the level of languages; three language sets serve as development sets to train systems and their hyper-parameters, another two surprise language sets serve as test sets. Each language set is split in train and test portions on the speaker level. The surprise language data sets are provided with no meta information at all. Each test set comes in three variants that differ in the length per utterance, which are 1 second, 10 seconds or 120 seconds. The baseline was provided by raw MFCC speech features as speech representations. The topline was established by extracting posteriorgrams from supervisedly trained language dependent GMM-HMM based phone recognizers (using 2-gram language models during decoding) [34].

Overall, there were 11 officially submitted unsupervised systems considered for the final ranking. 3 supervised systems were also listed in the leaderboard[2], but without being officially ranked. The submissions are by (in alphabetical order) Ansari et al. [5], Chen et al. [23], *Heck et al.* [69], Pellegrini et al. [127], Shibata et al. [150] and Yuan et al. [180].

[2] http://www.zerospeech.com/2017

Table B.1: ABX discriminability errors across speakers for the track 1 submissions of ZeroSpeech 2017. Scores are computed for three test file durations for the development languages and surprise languages. Overall, there were 11 unsupervised systems. 3 supervised systems which were excluded from the official ranking are listed separately at the bottom of the table.

Systems	English			French			Mandarin			LANG1			LANG2		
	1s	10s	120s	1s	10s	120s	1s	10s	120s	1s	10s	120s	1s	10s	120s
Baseline	23.4	23.4	23.4	25.2	25.5	25.2	21.3	21.3	21.3	23.6	23.2	23.0	30.0	29.5	29.5
Topline	8.6	6.9	6.7	10.6	9.1	8.9	12.0	5.7	5.1	12.8	10.5	10.4	7.1	3.6	4.3
Ansari et al. [5]	14.5	N/A	13.2	17.8	N/A	16.2	13.2	N/A	12.7	16.9	14.7	14.7	18.8	17.7	17.7
Ansari et al. [5]	13.7	N/A	12.4	17.2	N/A	15.6	12.6	N/A	12.0	16.0	14.0	13.9	17.9	16.9	16.6
Ansari et al. [5]	13.2	12.0	N/A	17.2	N/A	15.4	13.0	12.2	12.3	15.5	13.5	13.4	17.6	16.0	16.0
Ansari et al. [5]	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	15.7	13.7	13.5	17.5	16.1	16.1
Chen et al. [23]	13.7	12.1	12.0	17.6	15.6	14.8	12.3	10.8	10.7	15.5	12.9	12.7	17.6	16.9	16.3
Chen et al. [23]	12.7	11.0	10.8	17.0	14.5	14.1	11.9	10.3	10.1	14.7	11.7	11.6	16.9	14.7	14.4
Heck et al. [69] (Ch. 4)	10.1	8.7	8.5	13.6	11.7	11.3	8.8	7.4	7.3	11.9	10.0	9.7	13.0	10.0	9.9
Pellegrini et al. [127]	17.6	16.3	16.4	20.3	17.6	17.3	14.7	13.5	13.4	19.4	16.2	15.9	22.8	23.1	23.1
Pellegrini et al. [127]	17.6	16.2	16.3	20.1	17.7	17.3	14.7	13.5	13.4	19.2	16.3	16.0	23.3	23.3	23.1
Yuan et al. [180]	14.2	12.1	11.8	18.9	15.8	15.2	12.8	11.1	10.9	16.4	13.3	13.0	19.2	17.3	16.7
Yuan et al. [180]	14.0	11.9	11.7	18.6	15.5	14.9	12.7	10.8	10.7	16.2	12.9	12.6	19.5	17.1	16.6
Shibata et al. [150]	10.1	9.2	8.2	13.7	12.4	10.8	10.4	9.5	8.0	11.6	9.9	8.7	11.5	10.2	8.6
Shibata et al. [150]	7.9	7.4	6.9	11.2	10.8	9.8	7.8	7.5	6.7	9.3	8.6	7.8	8.3	7.9	7.2
Yuan et al. [180]	13.6	11.5	11.3	17.7	14.8	14.4	12.9	10.7	10.5	15.8	12.4	12.3	18.7	17.4	17.0

Table B.2: ABX discriminability errors within speakers for the track 1 submissions of ZeroSpeech 2017. Scores are computed for three test file durations for the development languages and surprise languages. Overall, there were 11 unsupervised systems. 3 supervised systems which were excluded from the official ranking are listed separately at the bottom of the table.

Systems	English			French			Mandarin			LANG1			LANG2		
	1s	10s	120s	1s	10s	120s	1s	10s	120s	1s	10s	120s	1s	10s	120s
Baseline	12.0	12.1	12.1	12.5	12.6	12.6	11.5	11.5	11.5	10.3	9.3	9.4	14.1	14.3	14.1
Topline	6.5	5.3	5.1	8.0	6.8	6.8	9.5	4.2	4.0	8.7	7.1	7.0	6.6	4.6	3.4
Ansari et al. [5]	7.4	N/A	6.6	9.8	N/A	8.5	9.3	N/A	8.3	6.9	6.1	6.0	9.9	9.2	9.1
Ansari et al. [5]	7.4	N/A	6.6	9.8	N/A	8.4	9.2	N/A	8.2	6.8	6.0	6.0	10.1	9.6	9.6
Ansari et al. [5]	7.7	6.8	N/A	10.4	N/A	8.8	10.4	9.3	9.1	7.3	6.2	6.1	11.1	10.3	10.2
Ansari et al. [5]	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	7.6	6.4	6.2	11.6	10.9	10.7
Chen et al. [23]	8.5	7.3	7.2	11.1	9.5	9.4	10.5	8.5	8.4	7.6	6.2	6.3	11.7	9.9	9.8
Chen et al. [23]	8.5	7.3	7.2	11.2	9.4	9.4	10.5	8.7	8.5	7.6	6.2	6.1	11.6	9.8	9.6
Heck et al. [69] (Ch. 4)	6.9	6.2	6.0	9.7	8.7	8.4	8.8	7.9	7.8	6.5	5.6	5.3	10.9	8.8	8.4
Pellegrini et al. [127]	9.9	8.2	8.3	11.8	9.7	9.6	11.0	8.5	8.2	8.9	6.7	6.4	13.3	11.9	11.8
Pellegrini et al. [127]	9.8	8.1	8.2	11.6	9.5	9.3	10.9	8.4	8.1	8.8	6.6	6.3	13.1	11.7	11.7
Yuan et al. [180]	8.9	7.1	7.1	12.2	9.6	9.7	11.3	8.6	8.3	8.2	6.2	6.2	12.7	10.1	9.9
Yuan et al. [180]	9.0	7.1	7.0	11.9	9.5	9.5	11.1	8.5	8.2	8.1	6.0	6.0	12.6	10.0	9.9
Shibata et al. [150]	6.7	6.5	5.7	9.7	9.2	7.9	9.8	9.2	8.2	6.3	5.8	5.0	9.0	8.7	7.2
Shibata et al. [150]	5.5	5.2	4.9	7.9	7.4	6.9	7.9	7.7	7.0	5.2	4.9	4.5	6.9	7.0	6.3
Yuan et al. [180]	8.9	7.1	7.0	12.0	9.3	9.2	11.3	8.6	8.2	8.0	6.0	5.9	12.9	10.8	10.6

C DPMoMM Performance on ZeroSpeech 2017 Data

This appendix describes additional experiments that we conducted using the Dirichlet process mixture of mixtures model (DPMoMM) as introduced in Chapter 6. We perform the same kind of experiments as in Section 6.7, but on the data of the zero resource challenge 2017 [34].

C.1 Data

We sample multiple DPMoMMs separately for each of the ZeroSpeech 2017 development language data sets, which are English, French and Mandarin. The data is described in 4.7.2 and in [34]. Since the references for ABX scoring are not made public for the surprise languages, we refrained from running our experiments for the "LANG1" and "LANG2" data sets, as scoring would not have been possible. Since the data is not accompanied by plain textual references as well, we could not use the NMI criterion (see Section 6.6.2) for evaluation.

We are further only using the test set portions of the data sets for model sampling. For the challenge, each test set came in three versions which differ in their utterance lengths. Utterances are either all cut to 1 second, 10 seconds or 120 seconds of length. We used the test data sets with utterance length of 120 seconds for our experiments.

The details of the used data portions are listed in Table C.1.

Table C.1: Used portions of the language specific data sets.

Language set	#files	duration
English	260	8.6h
French	118	3.9h
Mandarin	262	8.7h

C.2 Procedure

Analogous to Section 6.7.2, for each of the data sets, we separately cluster extracted PLP+LDA+MLLT+fMLLR speech feature vectors by sampling a DP-

MoMM, where the algorithm jointly samples the *Gaussian* mixtures as well as a global codebook of Gaussians, which corresponds to a standard DPMM. We compare the modeling quality of the DPMoMM to the jointly sampled codebook, i.e, the DPMM, by computing the relative performance improvements of extracted posteriorgrams or textual labels on the ABX task. We also compute the ratio of components to clusters, and the relative length increase of the inferred units. The results are presented as contour plots in Figure C.1.

As before, we run every sampling for 1000 iterations. Due to the larger amounts of data, we sampled each model one time for each configuration of hyperparameters. α is fixed to 1, and β can take on the values $\{100, 200, 300, 400\}$. For $c_{\max}$, we tested the values $\{2, 4, 8\}$. Note that the results presented here are not comparable to the performance reported in Appendix B due to differences in the setup. For one, the sampling iterations are fewer in order to handle time constraints. We also applied system combination in previous experiments, which was not utilized here because the focus of the experiments was to compare DPMoMM and DPMM performance. Most importantly, the scoring setup in Appendix B applies a normalization operation during DTW distance computations, which invalidates the comparison of jointly inferred DPMoMM and DPMM. The scoring setup used here is a variant which does not use this normalization. This leads to scores being located in a slightly shifted range of values, but which guarantee proper comparison of DPMoMM and DPMM outputs.

C.3 Analysis

The outcome of our experiments on the new ZeroSpeech 2017 data sets confirm the superior performance of the DPMoMM over the DPMM and the findings that we made before as laid out in detail in Section 6.7. With the proper parametrization, the DPMoMM achieves better performance according to all applied criteria, while at the same time resulting in a more compact model with fewer clusters, which is desired.

Considerable performance improvements are observable for the ABX_{ls} discriminability. Here, the best results are always achieved by allowing larger clusters. The optimal value for β seems to be found at the lower end of the parameter range, with 100 being the smallest value that we tested performing best during our experiments. For the ABX_{kl} discriminability, i.e., the performance of the extracted frame-wise posteriorgrams, we did not expect to see major performance improvements. Interestingly, we made the same observations than in the original experiments in Chapter 6 and witnessed measurable performance improvements. With a slightly larger value for β within the range from 100 to 200, and $c_{\max}$ set to 4, we have seen moderate performance improvements on all test sets. Other parameter settings lead to minor improvements or result in models that are comparable in performance to the DPMM.

Overall, we were able to demonstrate the effectiveness of the DPMoMM on five distinct data sets (two data sets from ZeroSpeech 205, three from ZeroSpeech 2017) that cover very different languages, data sizes and show varying quality, which naturally impacts the difficulty of the clustering problem. We could show that with some tuning of β and $c_{\max}$, considerable relative improvements on all evaluation metrics can be achieved. For the purpose of demonstration, Table C.2 lists the absolute numbers of our performance measures when fixing $\beta = 200$ and $c_{\max} = 4$, values that seem to work well for all of the three new test sets and ABX discriminability tasks. Using a single parameter pair, we would not achieve optimal performance on all data sets, but we see a reasonably good and, more importantly, consistent improvement over DPMM by using DPMoMM.

Table C.2: Performance comparison of standard DPMMs (K modeled by single Gaussians) and proposed DPMoMMs (K modeled by Gaussian mixtures). As a parameter pair that fares reasonably well on all data sets, β is fixed to 200 and $c_{\max}$ is fixed to 4. Paired t-tests on all ABX task outputs yield $p \ll 0.001$.

K modeled by	β	$c_{\max}$	K	C	K/C	ABX_{ls}	ABX_{kl}	avg. seg. len.
Results for the English test set								
Single Gaussians	-	-	443	-	1.0	10.64%	17.77%	2.17 frames
Gaussian Mixtures	200	4	380	443	0.857	10.26%	16.45%	2.36 frames
Results for the French test set								
Single Gaussians	-	-	365	-	1.0	15.47%	22.17%	2.51 frames
Gaussian Mixtures	200	4	347	365	0.95	15.22%	21.47%	2.7 frames
Results for the Mandarin test set								
Single Gaussians	-	-	498	-	1.0	7.94%	13.37%	2.2 frames
Gaussian Mixtures	200	4	458	498	0.919	7.85%	12.83%	2.3 frames

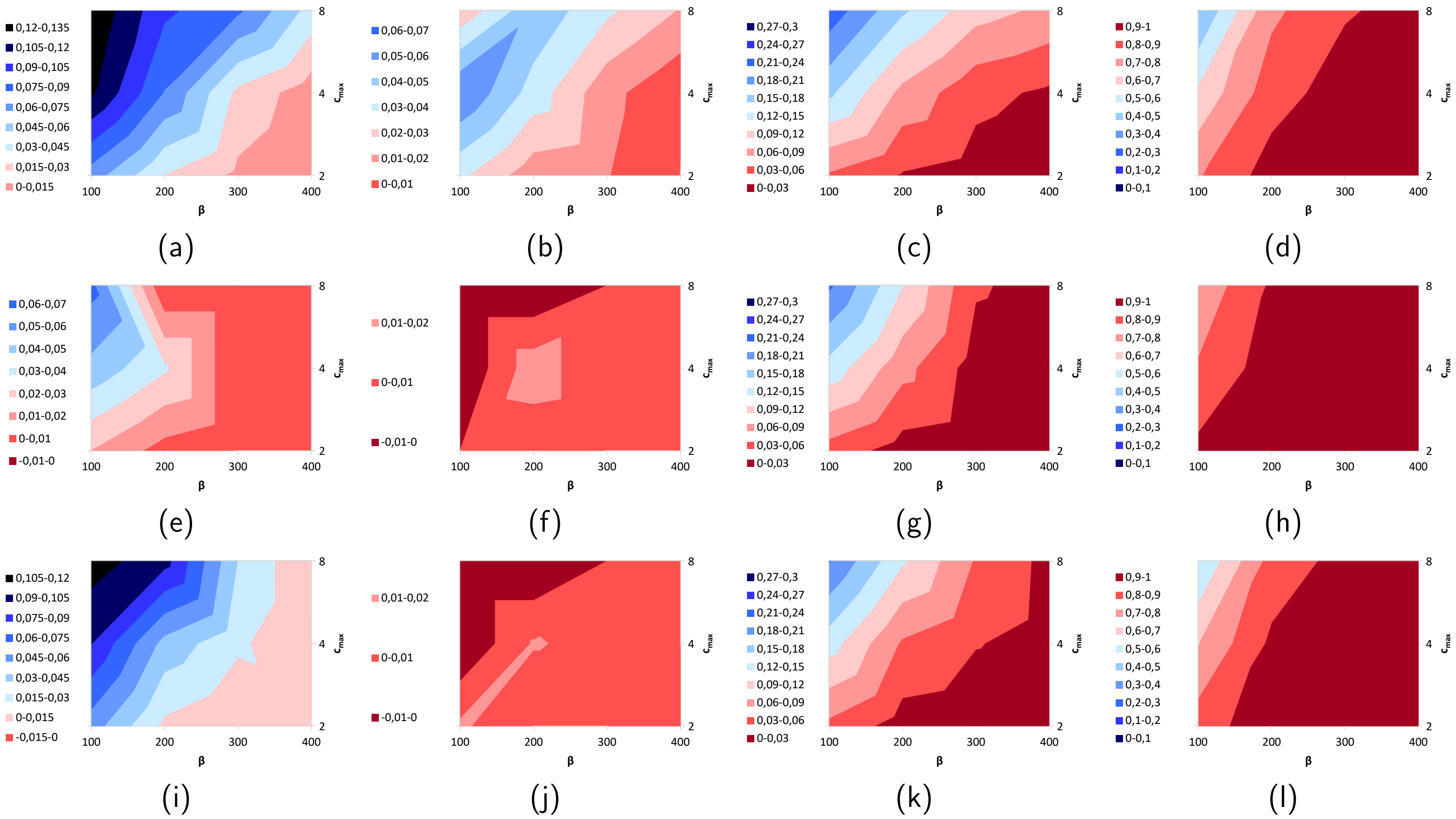


Figure C.1: Relative performance improvements when interpreting clusters instead of codebook components as acoustic classes. The inputs are 33 dimensional PLP+LDA+MLLT+fMLLR speech features. *(a)-(d)*: Results on English; *(e)-(h)*: Results on French; *(i)-(l)*: Results on Mandarin; *(a),(e),(i)* Relative improvement of ABX_{ls}, *(b),(f),(j)* Relative improvement of ABX_{kl}, *(c),(g),(k)* Relative unit length increase, *(d),(h),(l)* Cluster to component amount ratio. Paired t-tests on ABX task outputs with relative improvements > 0 yield $p \ll 0.001$.

References

[1] Gilles Adda, Sebastian Stüker, Martine Adda-Decker, Odette Ambouroue, Laurent Besacier, David Blachon, Hélene Bonneau-Maynard, Pierre Godard, Fatima Hamlaoui, Dmitry Idiatov, Guy-Noël Kouarata, Lori Lamel, Emmanuel-Moselly Makasso, Annie Rialland, Mark van de Velde, François Yvon, and Sabine Zerbian. Breaking the unwritten language barrier: The BULB project. In *Proceedings of the International Workshop on Spoken Language Technologies for Under-resourced Languages*, volume 81 of *Procedia Computer Science*, pages 8–14. International Speech Communication Association, Elsevier, 2016.

[2] David J. Aldous. Exchangeability and related topics. In *École d'Été de Probabilités de Saint-Flour XIII – 1983*, pages 1–198. Springer, 1985.

[3] Dante Alighieri. *Divina Commedia*. Johannes Numeister, 1472.

[4] Tasos Anastasakos, John McDonough, Richard Schwartz, and John Makhoul. A compact model for speaker-adaptive training. In *Proceedings of the International Conference on Spoken Language Processing*, pages 1137–1140. Institute of Electrical and Electronics Engineers, 1996.

[5] T. K. Ansari, Rajath Kumar, Sonali Singh, and Sriram Ganapathy. Deep learning methods for unsupervised acoustic modeling – leap submission to ZeroSpeech challenge 2017. In *Proceedings of the Automatic Speech Recognition and Understanding Workshop*, pages 754–761. Institute of Electrical and Electronics Engineers, 2017.

[6] T. K. Ansari, Rajath Kumar, Sonali Singh, Sriram Ganapathy, and Susheela Devi. Unsupervised HMM posteriograms for language independent acoustic modeling in zero resource conditions. In *Proceedings of the*

Automatic Speech Recognition and Understanding Workshop, pages 762–768. Institute of Electrical and Electronics Engineers, 2017.

[7] Charles E. Antoniak. Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. *Annals of Statistics*, pages 1152–1174, 1974.

[8] Leonardo Badino, Alessio Mereta, and Lorenzo Rosasco. Discovering discrete subword units with binarized autoencoders and hidden-Markov-model encoders. In *Proceedings of Interspeech*, pages 3174–3178. International Speech Communication Association, 2015.

[9] Pallavi Baljekar, Sunayana Sitaram, Prasanna Kumar Muthukumar, and Alan W. Black. Using articulatory features and inferred phonological segments in zero resource speech processing. In *Proceedings of Interspeech*, pages 3194–3198. International Speech Communication Association, 2015.

[10] Leonard E. Baum and John Alonzo Eagon. An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology. *Bulletin of the American Mathematical Society*, 73(3):360–363, 1967.

[11] Leonard E. Baum and Ted Petrie. Statistical inference for probabilistic functions of finite state Markov chains. *The Annals of Mathematical Statistics*, 37(6):1554–1563, 1966.

[12] Thomas Bayes and Richard Price. An essay towards solving a problem in the doctrine of chances. *Philosophical Transactions of the Royal Society of London*, 53:370–418, 1763.

[13] Laurent Besacier, Etienne Barnard, Alexey Karpov, and Tanja Schultz. Automatic speech recognition for under-resourced languages: A survey. *Speech Communication*, 56:85–100, 2014.

[14] Catherine T. Best. The emergence of native-language phonological influences in infants: A perceptual assimilation model. *The Development of Speech Perception: The Transition from Speech Sounds to Spoken Words*, pages 167–224, 1994.

[15] David Blackwell and James B. MacQueen. Ferguson distributions via Pólya urn schemes. *Annals of Statistics*, pages 353–355, 1973.

[16] David M. Blei and Peter I. Frazier. Distance dependent Chinese restaurant processes. *Journal of Machine Learning Research*, 12:2461–2488, 2011.

[17] John S. Bridle and Michael D. Brown. An experimental automatic word recognition system. Technical report, Joint Speech Research Unit, Ruislip, England, 1974.

[18] Jason Chang and John W. Fisher III. Parallel sampling of DP mixture models using sub-cluster splits. In *Advances in Neural Information Processing Systems*, pages 620–628. Neural Information Processing Systems Foundation, 2013.

[19] Jason Chang and John W. Fisher III. Supplemental material for parallel sampling of DP mixture models using sub-cluster splits, 2013.

[20] Jason Chang and John W. Fisher III. Parallel sampling of HDPs using sub-cluster splits. In *Advances in Neural Information Processing Systems*, pages 235–243. Neural Information Processing Systems Foundation, 2014.

[21] Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien. *Semi-supervised learning*. Adaptive Computation and Machine Learning Series. The MIT Press, 2010.

[22] Hongjie Chen, Cheung-Chi Leung, Lei Xie, Bin Ma, and Haizhou Li. Parallel inference of Dirichlet process Gaussian mixture models for unsupervised acoustic modeling: A feasibility study. In *Proceedings of Interspeech*, pages 3189–3193. International Speech Communication Association, 2015.

[23] Hongjie Chen, Cheung-Chi Leung, Lei Xie, Bin Ma, and Haizhou Li. Multilingual bottle-neck feature learning from untranscribed speech. In *Proceedings of the Automatic Speech Recognition and Understanding Workshop*, pages 727–733. Institute of Electrical and Electronics Engineers, 2017.

[24] Stanley F. Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, 13(4):359–394, 1999.

[25] Cheng-Tao Chung, Cheng-Yu Tsai, Hsiang-Hung Lu, Chia-Hsiang Liu, Hung-yi Lee, and Lin-shan Lee. An iterative deep learning framework for unsupervised discovery of speech features and linguistic units with applications on spoken term detection. In *Proceedings of the Automatic Speech Recognition and Understanding Workshop*, pages 245–251. Institute of Electrical and Electronics Engineers, 2015.

[26] James W. Cooley and John W. Tukey. An algorithm for the machine calculation of complex Fourier series. *Mathematics of Computation*, 19(90):297–301, 1965.

[27] Mathias Creutz and Krista Lagus. Unsupervised models for morpheme segmentation and morphology learning. *ACM Transactions on Speech and Language Processing*, 4(1):1–34, 2007.

[28] David B. Dahl. An improved merge-split sampler for conjugate Dirichlet process mixture models. Technical report, Department of Statistics, University of Wisconsin, 2003.

[29] Carl de Marcken. The unsupervised acquisition of a lexicon from continuous speech, 1996.

[30] Nic J. De Vries, Marelie H. Davel, Jaco Badenhorst, Willem D. Basson, Febe de Wet, Etienne Barnard, and Alta de Waal. A smartphone-based ASR data collection tool for under-resourced languages. *Speech communication*, 56:119–131, 2014.

[31] Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.

[32] Mark Dredze, Aren Jansen, Glen Coppersmith, and Ken Church. NLP on spoken documents without ASR. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 460–470. Association for Computational Linguistics, 2010.

[33] Homer Dudley. The carrier nature of speech. *Bell System Technical Journal*, 19(4):495–515, 1940.

[34] Ewan Dunbar, Xuan Nga Cao, Juan Benjumea, Julien Karadyi, Mathieu Bernard, Laurent Besacier, Xavier Anguerra, and Emmanuel Dupoux. The zero resource speech challenge 2017. In *Proceedings of the Automatic Speech Recognition and Understanding Workshop*, pages 323–330. Institute of Electrical and Electronics Engineers, 2017.

[35] Emmanuel Dupoux, Christophe Pallier, Nuria Sebastian, and Jacques Mehler. A destressing "deafness" in French? *Journal of Memory and Language*, 36(3):406–421, 1997.

[36] Peter D. Eimas, Einar R. Siqueland, Peter Juscyk, and James Vigorito. Speech perception in infants. *Science*, 171(3968):303–306, 1971.

[37] Mohamed Elmahdy, Mark Hasegawa-Johnson, and Eiman Mustafawi. A transfer learning approach for under-resourced arabic dialects speech recognition. In *Proceedings of the Workshop on Less Resourced Languages, New Technologies, New Challenges and Opportunities*, pages 60–64, 2013.

[38] Ralph Waldo Emerson. *Essays: Second Series*. James Munroe & Co, 1844.

[39] Stefano Favaro and Yee Whye Teh. MCMC for normalized random measure mixture models. *Statistical Science*, pages 335–359, 2013.

[40] Naomi H. Feldman, Thomas L. Griffiths, Sharon Goldwater, and James L. Morgan. A role for the developing lexicon in phonetic category acquisition. *Psychological Review*, 120(4):751–778, 2013.

[41] Thomas S. Ferguson. A Bayesian analysis of some nonparametric problems. *Annals of Statistics*, pages 209–230, 1973.

[42] Daniel Fink. A compendium of conjugate priors, 1997.

[43] Michael Finke and Alex Waibel. Flexible transcription alignment. In *Proceedings of the Automatic Speech Recognition and Understanding Workshop*, pages 34–40. Institute of Electrical and Electronics Engineers, 1997.

[44] Ronald A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2):179–188, 1936.

[45] James Emil Flege and James Hillenbrand. Differential use of temporal cues to the/s/–/z/contrast by native and non-native speakers of English. *The Journal of the Acoustical Society of America*, 79(2):508–517, 1986.

[46] G. David Forney. The Viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.

[47] Sadaoki Furui. Speaker-independent isolated word recognition using dynamic features of speech spectrum. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 34(1):52–59, 1986.

[48] Mark J. F. Gales. Maximum likelihood linear transformations for HMM-based speech recognition. *Computer Speech and Language*, 12(2):75–98, 1998.

[49] Mark J. F. Gales. Semi-tied covariance matrices for hidden Markov models. *IEEE Transactions on Speech and Audio Processing*, 7(3):272–281, 1999.

[50] Mark J. F. Gales, Kate M. Knill, and Anton Ragni. Unicode-based graphemic systems for limited resource languages. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 5186–5190. Institute of Electrical and Electronics Engineers, 2015.

[51] Mark J. F. Gales, Kate M. Knill, Anton Ragni, and Shakti P. Rath. Speech recognition and keyword spotting for low-resource languages: Babel project research at CUED. In *Proceedings of the International Workshop on Spoken Language Technologies for Under-resourced Languages*, pages 16–23. International Speech Communication Association, 2014.

[52] Stuart Geman and Donald Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

[53] James Glass. Towards unsupervised speech processing. In *Proceedings of the International Conference on Information Science, Signal Processing and their Applications*, pages 1–4. Institute of Electrical and Electronics Engineers, 2012.

[54] Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. A bayesian framework for word segmentation: Exploring the effects of context. *Cognition*, 112(1):21–54, 2009.

[55] Irving J. Good. The population frequencies of species and the estimation of population parameters. *Biometrika*, 40(3-4):237–264, 1953.

[56] Ramesh A. Gopinath. Maximum likelihood modeling with Gaussian distributions for classification. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 661–664. Institute of Electrical and Electronics Engineers, 1998.

[57] Jim E. Griffin and Mark F. J. Steel. Order-based dependent Dirichlet processes. *Journal of the American Statistical Association*, 101(473):179–194, 2006.

[58] Friedrich Melchior Grimm. Janvier 1755. In *Correspondance littéraire, philosophique et critique*. Longchamps & Buisson, 1813.

[59] Collette Grinevald and Michel Bert. *The Cambridge Handbook of Endangered Languages*, chapter Speakers and Communities, pages 45–65. Cambridge University Press, 2011.

[60] Reinhold Haeb-Umbach and Hermann Ney. Linear discriminant analysis for improved large vocabulary continuous speech recognition. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, volume 1, pages 13–16. Institute of Electrical and Electronics Engineers, 1992.

[61] Thomas Hain, Philip C. Woodland, Thomas R. Niesler, and Edward W. D. Whittaker. The 1998 HTK system for transcription of conversational telephone speech. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 57–60. Institute of Electrical and Electronics Engineers, 1999.

[62] Mary Harper. IARPA Babel program. Office of the Director of National Intelligence, Intelligence Advanced Research Projects Activity, 2014. https://www.iarpa.gov/index.php/research-programs/babel (accessed 2018).

[63] William Hartmann, Roger Hsiao, Tim Ng, Jeff Ma, Francis Keith, and Man-Hung Siu. Improved single system conversational telephone speech recognition with VGG bottleneck features. In *Proceedings of Interspeech*, pages 112–116. Institute of Electrical and Electronics Engineers, 2017.

[64] David Harwath and James R. Glass. Learning word-like units from joint audio-visual analysis. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 506–517. Association for Computational Linguistics, 2017.

[65] David Harwath, Antonio Torralba, and James Glass. Unsupervised learning of spoken language with visual context. In *Advances in Neural Information Processing Systems*, pages 1858–1866. Neural Information Processing Systems Foundation, 2016.

[66] Michael Heck, Sakriani Sakti, and Satoshi Nakamura. Iterative training of a DPGMM-HMM acoustic unit recognizer in a zero resource scenario. In *Proceedings of the Spoken Language Technology Workshop*, pages 57–63. Institute of Electrical and Electronics Engineers, 2016.

[67] Michael Heck, Sakriani Sakti, and Satoshi Nakamura. Supervised learning of acoustic models in a zero resource setting to improve DPGMM clustering. In *Proceedings of Interspeech*, pages 1310–1314. International Speech Communication Association, 2016.

[68] Michael Heck, Sakriani Sakti, and Satoshi Nakamura. Unsupervised linear discriminant analysis for supporting DPGMM clustering in the zero resource scenario. In *Proceedings of the International Workshop on Spoken Language Technologies for Under-resourced Languages*, volume 81 of *Procedia Computer Science*, pages 73–79. International Speech Communication Association, Elsevier, 2016.

[69] Michael Heck, Sakriani Sakti, and Satoshi Nakamura. Feature optimized DPGMM clustering for unsupervised subword modeling: A contribution to Zerospeech 2017. In *Proceedings of the Automatic Speech Recognition and Understanding Workshop*, pages 740–746. Institute of Electrical and Electronics Engineers, 2017.

[70] Michael Heck, Sakriani Sakti, and Satoshi Nakamura. Dirichlet process mixture of mixtures model for unsupervised subword modeling. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(11):2027–2042, 2018.

[71] Michael Heck, Sakriani Sakti, and Satoshi Nakamura. Learning supervised feature transformations on zero resources for improved acoustic unit discovery. *IEICE Transactions on Information and Systems*, 101(1):205–214, 2018.

[72] Hynek Hermansky. Perceptual linear predictive (PLP) analysis of speech. *The Journal of the Acoustical Society of America*, 87(4):1738–1752, 1990.

[73] Florian Hönig, Georg Stemmer, Christian Hacker, and Fabio Brugnara. Revising perceptual linear prediction (PLP). In *Proceedings of Interspeech*, pages 2997–3000. International Speech Communication Association, 2005.

[74] Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24(6):417–441, 1933.

[75] Xuedong D. Huang, Yasuo Ariki, and Mervyn A. Jack. *Hidden Markov models for speech recognition*. Edinburgh University Press, 1990.

[76] Hemant Ishwaran and Lancelot F. James. Gibbs sampling methods for stick-breaking priors. *Journal of the American Statistical Association*, 96(453):161–173, 2001.

[77] Sonia Jain and Radford M. Neal. A split-merge Markov chain Monte Carlo procedure for the Dirichlet process mixture model. *Journal of Computational and Graphical Statistics*, 13(1):158–182, 2004.

[78] Sonia Jain and Radford M. Neal. Splitting and merging components of a nonconjugate Dirichlet process mixture model. *Bayesian Analysis*, 2(3):445–472, 2007.

[79] Frederick Jelinek. Continuous speech recognition by statistical methods. *Proceedings of the IEEE*, 64(4):532–556, 1976.

[80] Peter W. Jusczyk. *Phonological development: Models, research, implications*, chapter Developing phonological categories from the speech signal, pages 17–64. York Press, 1992.

[81] Herman Kamper, Micha Elsner, Aren Jansen, and Sharon Goldwater. Unsupervised neural network based feature extraction using weak top-down constraints. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 5818–5822. Institute of Electrical and Electronics Engineers, 2015.

[82] Herman Kamper, Aren Jansen, and Sharon Goldwater. Unsupervised word segmentation and lexicon discovery using acoustic word embeddings. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 24(4):669–679, 2016.

[83] Herman Kamper, Aren Jansen, Simon King, and Sharon Goldwater. Unsupervised lexical clustering of speech segments using fixed-dimensional acoustic embeddings. In *Proceedings of the Spoken Language Technology Workshop*, pages 100–105. Institute of Electrical and Electronics Engineers, 2014.

[84] Herman Kamper, Karen Livescu, and Sharon Goldwater. An embedded segmental k-means model for unsupervised segmentation and clustering of speech. In *Proceedings of the Automatic Speech Recognition and Understanding Workshop*, pages 719–726. Institute of Electrical and Electronics Engineers, 2017.

[85] Immanuel Kant. *Anthropologie in pragmatischer Hinsicht.* Friedrich Nicolovius, 1798.

[86] Do Yeong Kim, S. Umesh, Mark J. F. Gales, Thomas Hain, and Philip C. Woodland. Using VTLN for broadcast news transcription. In *Proceedings of the International Conference on Spoken Language Processing*, pages 1953–1956. Institute of Electrical and Electronics Engineers, 2004.

[87] Reinhard Kneser and Hermann Ney. Improved backing-off for m-gram language modeling. In *Proceedings of the International Conference on Acous-*

tics, Speech and Signal Processing, volume 1, pages 181–184. Institute of Electrical and Electronics Engineers, 1995.

[88] Kate M. Knill, Mark J. F. Gales, Shakti P. Rath, Philip C. Woodland, Chao Zhang, and S.-X. Zhang. Investigation of multilingual deep neural networks for spoken term detection. In *Proceedings of the Automatic Speech Recognition and Understanding Workshop*, pages 138–143. Institute of Electrical and Electronics Engineers, 2013.

[89] Patricia K. Kuhl. Speech perception in early infancy: Perceptual constancy for spectrally dissimilar vowel categories. *The Journal of the Acoustical Society of America*, 66(6):1668–1679, 1979.

[90] Patricia K. Kuhl. Perception of auditory equivalence classes for speech in early infancy. *Infant Behavior and Development*, 6(2):263–285, 1983.

[91] Patricia K. Kuhl. Early language acquisition: Cracking the speech code. *Nature Reviews Neuroscience*, 5(11):831–843, 2004.

[92] Patricia K. Kuhl, Karen A. Williams, Francisco Lacerda, Kenneth N. Stevens, and Björn Lindblom. Linguistic experience alters phonetic perception in infants by 6 months of age. *Science*, 255(5044):606–608, 1992.

[93] Kevin J. Lang, Alex H. Waibel, and Geoffrey E. Hinton. A time-delay neural network architecture for isolated word recognition. *Neural Networks*, 3(1):23–43, 1990.

[94] Viet-Bac Le, Lori Lamel, Abdel Messaoudi, William Hartmann, Jean-Luc Gauvain, Cécile Woehrling, Julien Despres, and Anindya Roy. Developing STT and KWS systems using limited language resources. In *Proceedings of Interspeech*, pages 2484–2488. International Speech Communication Association, 2014.

[95] Chia-ying Lee and James Glass. A nonparametric Bayesian approach to acoustic model discovery. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics: Long Papers*, volume 1, pages 40–49. Association for Computational Linguistics, 2012.

[96] Chia-ying Lee, Timothy J. O'Donnell, and James Glass. Unsupervised lexicon discovery from acoustic input. *Transactions of the Association for Computational Linguistics*, 3:389–403, 2015.

[97] Kai-Fu Lee. On large-vocabulary speaker-independent continuous speech recognition. *Speech Communication*, 7(4):375–379, 1988.

[98] Stephen E. Levinson, Lawrence R. Rabiner, and Man Mohan Sondhi. An introduction to the application of the theory of probabilistic functions of a Markov process to automatic speech recognition. *Bell System Technical Journal*, 62(4):1035–1074, 1983.

[99] Jianhua Lin. Divergence measures based on the Shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151, 1991.

[100] Richard P. Lippmann. Review of neural networks for speech recognition. *Neural Computation*, 1(1):1–38, 1989.

[101] Silvia C. Lipski, Paola Escudero, and Titia Benders. Language experience modulates weighting of acoustic cues for vowel perception: An event-related potential study. *Psychophysiology*, 49(5):638–650, 2012.

[102] Jonas Lööf, Christian Gollan, and Hermann Ney. Cross-language bootstrapping for unsupervised acoustic model training: Rapid development of a Polish speech recognition system. In *Proceedings of Interspeech*, pages 88–91. International Speech Communication Association, 2009.

[103] Andrew J. Lotto, Momoko Sato, and Randy L. Diehl. Mapping the task for the second language learner: The case of Japanese acquisition of /r/ and /l/. In *Proceedings of the From Sound to Sense Conference*, pages C181–C186. Massachusetts Institute of Technology, 2004.

[104] Vince Lyzinski, Gregory Sell, and Aren Jansen. An evaluation of graph clustering methods for unsupervised term discovery. In *Proceedings of Interspeech*, pages 3209–3213. International Speech Communication Association, 2015.

[105] Steven N. MacEachern. Dependent nonparametric processes. In *Proceedings of the Section on Bayesian Statistical Science*, pages 50–55. American Statistical Association, 1999.

[106] Steven N. MacEachern. Dependent Dirichlet processes. Technical report, Department of Statistics, The Ohio State University, 2000.

[107] Neil A. Macmillan and C. Douglas Creelman. *Detection theory: A user's guide*, chapter 9. Psychology Press, 2004.

[108] Igor Malioutov, Alex Park, Regina Barzilay, and James Glass. Making sense of sound: Unsupervised topic segmentation over acoustic input. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 504–511. Association for Computational Linguistics, 2007.

[109] John D. Markel and Augustine H. Gray Jr. *Linear prediction of speech*. Communication and Cybernetics. Springer-Verlag Berlin Heidelberg, 1976.

[110] Aaron F. McDaid, Derek Greene, and Neil Hurley. Normalized mutual information to evaluate overlapping community finding algorithms. *arXiv preprint arXiv:1110.2515*, 2011.

[111] Paul Mermelstein. Distance measures for speech recognition, psychological and instrumental. *Pattern recognition and Artificial Intelligence*, pages 374–388, 1976.

[112] Christopher Moseley. *Atlas of the World's Languages in Danger*. Unesco, 2010.

[113] Markus Müller, Sebastian Stüker, and Alex Waibel. Language adaptive DNNs for improved low resource speech recognition. In *Proceedings of Interspeech*, pages 3878–3882. International Speech Communication Association, 2016.

[114] Markus Müller and Alex Waibel. Using language adaptive deep neural networks for improved multilingual speech recognition. In *Proceedings of the International Workshop on Spoken Language Translation*, pages 167–172, 2015.

[115] Radford M. Neal. Bayesian mixture modeling. In *Maximum Entropy and Bayesian Methods*, pages 197–211. Springer, 1992.

[116] Radford M. Neal. Markov chain sampling methods for Dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 9(2):249–265, 2000.

[117] Graham Neubig. Unsupervised learning of lexical information for language processing systems, 2012.

[118] Graham Neubig, Masato Mimura, Shinsuke Mori, and Tatsuya Kawahara. Learning a language model from continuous speech. In *Proceedings of Interspeech*, pages 1053–1056. International Speech Communication Association, 2010.

[119] Quoc Bao Nguyen, Jonas Gehring, Markus Müller, Sebastian Stüker, and Alex Waibel. Multilingual shifting deep bottleneck features for low-resource ASR. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 5607–5611. Institute of Electrical and Electronics Engineers, 2014.

[120] Heinrich Niemann. *Pattern analysis and understanding.* Springer Series in Information Sciences. Springer, 1990.

[121] Alan V. Oppenheim. *Superposition in a class of nonlinear systems.* MIT Research Laboratory of Electronics, 1965.

[122] Alan V. Oppenheim and Ronald W. Schafer. *Digital Signal Processing.* Prentice Hall international editions. Pearson, 1975.

[123] Omiros Papaspiliopoulos and Gareth O. Roberts. Retrospective Markov chain Monte Carlo methods for Dirichlet process hierarchical models. *Biometrika*, 95(1):169–186, 2008.

[124] Alex Park and James Glass. Towards unsupervised pattern discovery in speech. In *Proceedings of the Automatic Speech Recognition and Understanding Workshop*, pages 53–58. Institute of Electrical and Electronics Engineers, 2005.

[125] Alex Park and James Glass. Unsupervised pattern discovery in speech. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(1):186–197, 2008.

[126] Karl Pearson. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

[127] Thomas Pellegrini, Céline Manenti, and Julien Pinquier. Technical report – the IRIT-UPS system ZeroSpeech 2017 track1: Unsupervised subword modeling. Technical report, Institut de Recherche en Informatique de Toulouse, Université de Toulouse, 2017.

[128] Jim Pitman. Exchangeable and partially exchangeable random partitions. *Probability Theory and Related Fields*, 102(2):145–158, 1995.

[129] Jim Pitman. Combinatorial stochastic processes. Technical report, Department of Statistics, University of California, 2002.

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

[131] Hoifung Poon, Colin Cherry, and Kristina Toutanova. Unsupervised morphological segmentation with log-linear models. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 209–217. Association for Computational Linguistics, 2009.

[132] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, Jan Silovsky, Georg Stemmer, and Karel Vesely. The Kaldi speech recognition toolkit. In *Proceedings of the Automatic Speech Recognition and Understanding Workshop*. Institute of Electrical and Electronics Engineers, 2011.

[133] Daniel Povey and Kaisheng Yao. A basis representation of constrained MLLR transforms for robust adaptation. *Computer Speech & Language*, 26(1):35–51, 2012.

[134] Lawrence R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[135] Lawrence R. Rabiner and Biing-Hwang Juang. An introduction to hidden Markov models. *IEEE ASSP magazine*, 3(1):4–16, 1986.

[136] Lawrence R. Rabiner and Biing-Hwang Juang. *Fundamentals of speech recognition*. Prentice Hall, 1993.

[137] Okko Räsänen, Gabriel Doyle, and Michael C. Frank. Unsupervised word discovery from speech using automatic segmentation into syllable-like units. In *Proceedings of Interspeech*, pages 3204–3208. International Speech Communication Association, 2015.

[138] Carl Edward Rasmussen. The infinite Gaussian mixture model. In *Advances in Neural Information Processing Systems*, pages 554–560. Neural Information Processing Systems Foundation, 2000.

[139] Daniel Renshaw, Herman Kamper, Aren Jansen, and Sharon Goldwater. A comparison of neural network methods for unsupervised representation learning on the zero resource speech challenge. In *Proceedings of Interspeech*, pages 3199–3203. International Speech Communication Association, 2015.

[140] Deb K. Roy and Alex P. Pentland. Learning words from sights and sounds: A computational model. *Cognitive Science*, 26(1):113–146, 2002.

[141] Thomas Schatz, Vijayaditya Peddinti, Francis Bach, Aren Jansen, Hynek Hermansky, and Emmanuel Dupoux. Evaluating speech features with the minimal-pair ABX task: Analysis of the classical MFC/PLP pipeline. In *Proceedings of Interspeech*, pages 1781–1785. International Speech Communication Association, 2013.

[142] Ernst Günter Schukat-Talamazzini. *Automatische Spracherkennung – Grundlagen, statistische Modelle und effiziente Algorithmen.* Künstliche Intelligenz. Springer Vieweg, 1995.

[143] Tanja Schultz. GlobalPhone: A multilingual speech and text database developed at Karlsruhe University. In *Proceedings of the International Conference of Spoken Language Processing*, pages 345–348, 2002.

[144] Tanja Schultz, Ngoc Thang Vu, and Tim Schlippe. GlobalPhone: A multilingual text & speech database in 20 languages. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 8126–8130. Institute of Electrical and Electronics Engineers, 2013.

[145] Tanja Schultz and Alex Waibel. Fast bootstrapping of LVCSR systems with multilingual phoneme sets. In *Proceedings of Eurospeech*, volume 1, pages 371–373. International Speech Communication Association, 1997.

[146] Tanja Schultz and Alex Waibel. Language-independent and language-adaptive acoustic modeling for speech recognition. *Speech Communication*, 35(1-2):31–51, 2001.

[147] Tanja Schultz, Martin Westphal, and Alex Waibel. The GlobalPhone project: Multilingual LVCSR with JANUS-3. In *Proceedings of the 2nd SQEL Workshop*, pages 20–27, 1997.

[148] Richard Schwartz, Y. L. Chow, O. Kimball, S. Roucos, M. Krasner, and John Makhoul. Context-dependent modeling for acoustic-phonetic recognition of continuous speech. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, volume 10, pages 1205–1208. Institute of Electrical and Electronics Engineers, 1985.

[149] Jayaram Sethuraman. A constructive definition of Dirichlet priors. *Statistica Sinica*, 4:639–650, 1994.

[150] Hayato Shibata, Taku Kato, Takahiro Shinozaki, and Shinji Watanabe. Composite embedding systems for ZeroSpeech2017 track1. In *Proceedings of the Automatic Speech Recognition and Understanding Workshop*, pages 747–753. Institute of Electrical and Electronics Engineers, 2017.

[151] Gary F. Simons and Charles D. Fennig. Ethnologue: Languages of the world, twentieth edition. SIL International, Dallas, Texas, 2017. http://www.ethnologue.com (accessed 2018.01.17).

[152] Man-hung Siu, Herbert Gish, Arthur Chan, William Belfield, and Steve Lowe. Unsupervised training of an HMM-based self-organizing unit recognizer with applications to topic classification and keyword discovery. *Computer Speech & Language*, 28(1):210–223, 2014.

[153] Brij Mohan Lal Srivastava and Manish Shrivastava. Articulatory gesture rich representation learning of phonological units in low resource settings. In *Statistical Language and Speech Processing*, pages 80–95. Springer, 2016.

[154] Felix Stahlberg, Tim Schlippe, Stephan Vogel, and Tanja Schultz. Towards automatic speech recognition without pronunciation dictionary, transcribed speech and text resources in the target language using cross-lingual word-to-phoneme alignment. In *Proceedings of the International Workshop on Spoken Language Technologies for Under-resourced Languages*, pages 73–80. International Speech Communication Association, 2014.

[155] Andreas Stolcke. SRILM – an extensible language modeling toolkit. In *Proceedings of the International Conference on Spoken Language Processing*, pages 901–904. Institute of Electrical and Electronics Engineers, 2002.

[156] Pawel Swietojanski, Arnab Ghoshal, and Steve Renals. Unsupervised cross-lingual knowledge transfer in DNN-based LVCSR. In *Proceedings of the Spoken Language Technology Workshop*, pages 246–251. Institute of Electrical and Electronics Engineers, 2012.

[157] Gabriel Synnaeve, Thomas Schatz, and Emmanuel Dupoux. Phonetics embedding learning with side information. In *Proceedings of the Spoken Language Technology Workshop*, pages 106–111. Institute of Electrical and Electronics Engineers, 2014.

[158] Yun Tang and Richard Rose. A study of using locality preserving projections for feature extraction in speech recognition. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 1569–1572. Institute of Electrical and Electronics Engineers, 2008.

[159] Yee W. Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. Sharing clusters among related groups: Hierarchical Dirichlet processes. In *Advances in Neural Information Processing Systems*, pages 1385–1392. Neural Information Processing Systems Foundation, 2005.

[160] Roland Thiolliere, Ewan Dunbar, Gabriel Synnaeve, Maarten Versteegh, and Emmanuel Dupoux. A hybrid dynamic time warping-deep neural network architecture for unsupervised acoustic modeling. In *Proceedings of Interspeech*, pages 3179–3183. International Speech Communication Association, 2015.

[161] Hartmut Traunmüller. Analytical expressions for the tonotopic sensory scale. *The Journal of the Acoustical Society of America*, 88(1):97–100, 1990.

[162] Gautam K. Vallabha, James L. McClelland, Ferran Pons, Janet F. Werker, and Shigeaki Amano. Unsupervised learning of vowel categories from infant-directed speech. *Proceedings of the National Academy of Sciences of the United States of America*, 104(33):13273–13278, 2007.

[163] Balakrishnan Varadarajan, Sanjeev Khudanpur, and Emmanuel Dupoux. Unsupervised learning of acoustic sub-word units. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pages 165–168. Association for Computational Linguistics, 2008.

[164] Maarten Versteegh, Roland Thiolliere, Thomas Schatz, Xuan Nga Cao, Xavier Anguera, Aren Jansen, and Emmanuel Dupoux. The zero resource speech challenge 2015. In *Proceedings of Interspeech*, pages 3169–3173. International Speech Communication Association, 2015.

[165] Andrew Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269, 1967.

[166] Marie von Ebner-Eschenbach. *Gesammelte Schriften.* Gebrüder Paetel, 1893.

[167] Johann Wolfgang von Goethe. Faust. Der Tragödie zweiter Teil. In *Goethe's Werke. Vollständige Ausgabe letzter Hand*, volume 41. J.G. Cotta'sche Buchhandlung, 1832.

[168] Ngoc Thang Vu, Franziska Kraus, and Tanja Schultz. Cross-language bootstrapping based on completely unsupervised training using multilingual A-stabil. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 5000–5003. Institute of Electrical and Electronics Engineers, 2011.

[169] Ngoc Thang Vu, Franziska Kraus, and Tanja Schultz. Rapid building of an ASR system for under-resourced languages based on multilingual unsupervised training. In *Proceedings of Interspeech*, pages 3145–3148. International Speech Communication Association, 2011.

[170] Alex Waibel and Kai-Fu Lee. *Readings in speech recognition.* Morgan Kaufmann, 1990.

[171] Alexander Waibel, Toshiyuki Hanazawa, Geoffrey Hinton, Kiyohiro Shikano, and Kevin J. Lang. Phoneme recognition using time-delay neural networks. In *Readings in Speech Recognition*, pages 393–404. Elsevier, 1990.

[172] Janet F. Werker and Chris E. Lalonde. Cross-language speech perception: Initial capabilities and developmental change. *Developmental Psychology*, 24(5):672, 1988.

[173] Janet F. Werker and Richard C. Tees. Cross-language speech perception: Evidence for perceptual reorganization during the first year of life. *Infant Behavior and Development*, 7(1):49–63, 1984.

[174] Ian H. Witten and Timothy C. Bell. The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. *IEEE Transactions on Information Theory*, 37(4):1085–1094, 1991.

[175] Wayne Xiong, Jasha Droppo, Xuedong Huang, Frank Seide, Mike Seltzer, Andreas Stolcke, Dong Yu, and Geoffrey Zweig. Towards human parity in conversational speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(12):2410–2423, 2017.

[176] Jingzhou Yang, Rogier C. Van Dalen, and Mark J. F. Gales. Infinite support vector machines in speech recognition. In *Proceedings of Interspeech*, pages 3303–3307. International Speech Communication Association, 2013.

[177] Jingzhou Yang, Rogier C. Van Dalen, Shi-Xiong Zhang, and Mark J. F. Gales. Infinite structured support vector machines for speech recognition. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 3320–3324. Institute of Electrical and Electronics Engineers, 2014.

[178] Halid Z. Yerebakan, Bartek Rajwa, and Murat Dundar. The infinite mixture of infinite Gaussian mixtures. In *Advances in Neural Information Processing Systems*, pages 28–36. Neural Information Processing Systems Foundation, 2014.

[179] Sari Ylinen, Maria Uther, Antti Latvala, Sara Vepsäläinen, Paul Iverson, Reiko Akahane-Yamada, and Risto Näätänen. Training the brain to weight speech cues differently: A study of Finnish second-language users of English. *Journal of Cognitive Neuroscience*, 22(6):1319–1332, 2010.

[180] Yougen Yuan, Cheung-Chi Leung, Lei Xie, Hongjie Chen, Bin Ma, and Haizhou Li. Extracting bottleneck features and word-like pairs from untranscribed speech for feature representation. In *Proceedings of the Automatic Speech Recognition and Understanding Workshop*, pages 734–739. Institute of Electrical and Electronics Engineers, 2017.

[181] Joseph Stanislaus Zauper. *Aphorismen moralischen und ästhetischen Inhalts*. Carl Gerold, 1840.

[182] Neil Zeghidour, Gabriel Synnaeve, Maarten Versteegh, and Emmanuel Dupoux. A deep scattering spectrum – deep siamese network pipeline for unsupervised acoustic modeling. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 4965–4969. Institute of Electrical and Electronics Engineers, 2016.

[183] Yaodong Zhang and James Glass. Unsupervised spoken keyword spotting via segmental DTW on Gaussian posteriorgrams. In *Proceedings of the*

Automatic Speech Recognition and Understanding Workshop, pages 398–403. Institute of Electrical and Electronics Engineers, 2009.

[184] Xiaojin Zhu, Zoubin Ghahramani, and John Lafferty. Time-sensitive Dirichlet process mixture models. Technical report, School of Computer Science, Carnegie Mellon University, 2005.

List of Publications

Journals

- Michael Heck, Sakriani Sakti, & Satoshi Nakamura (2018). Learning supervised feature transformations on zero resources for improved acoustic unit discovery. In IEICE transactions on information and systems, vol. E101-D, no. 1 (pp. 205-214).
- Michael Heck, Sakriani Sakti, & Satoshi Nakamura (2018). Dirichlet process mixture of mixtures model for unsupervised subword modeling. In IEEE/ACM transactions on audio, speech and language processing, vol. 26, no. 11 (pp. 2027-2042).

International Conferences (Peer-reviewed)

- Michael Heck, Sakriani Sakti, & Satoshi Nakamura (2017). Feature optimized DPGMM clustering for unsupervised subword modeling: A contribution to ZeroSpeech 2017. In Proceedings of the IEEE workshop on automatic speech recognition and understanding (ASRU) (pp. 740-746).
- Nurul Lubis, Michael Heck, Sakriani Sakti, Koichiro Yoshino, & Satoshi Nakamura (2017). Processing negative emotions through social communication: Multimodal database construction and analysis. In Proceedings of the international conference on affective computing and intelligent interaction (ACII) (pp. 79-85).
- Michael Heck, Masayuki Suzuki, Takashi Fukuda, Gakuto Kurata, &

Satoshi Nakamura (2017). Ensembles of multi-scale VGG acoustic models. In Proceedings of Interspeech (pp. 1616-1620).

- Michael Heck, Sakriani Sakti, & Satoshi Nakamura (2016). Iterative training of a DPGMM-HMM acoustic unit recognizer in a zero resource scenario. In Proceedings of the IEEE workshop on spoken language technology (SLT) (pp. 57-63).

- Michael Heck, Sakriani Sakti, & Satoshi Nakamura (2016). Supervised learning of acoustic models in a zero resource setting to improve DPGMM clustering. In Proceedings of Interspeech (pp. 1310-1314).

- Michael Heck, Sakriani Sakti, & Satoshi Nakamura (2016). Unsupervised linear discriminant analysis for supporting DPGMM clustering in the zero resource scenario. In Proceedings of the international workshop on spoken language technologies for under-resourced Languages (SLTU). Procedia Computer Science, vol. 81 (pp. 73-79).

- Quoc Truong Do, Michael Heck, Sakriani Sakti, Graham Neubig, Tomoki Toda, & Satoshi Nakamura (2015). The NAIST ASR system for the 2015 multi-genre broadcast challenge: On combination of deep learning systems using a rank-score function. In Proceedings of the IEEE workshop on automatic speech recognition and understanding (ASRU) (pp. 654-659).

- Michael Heck, Quoc Truong Do, Sakriani Sakti, Graham Neubig, & Satoshi Nakamura (2015). The NAIST English speech recognition system for IWSLT 2015. In Proceedings of the international workshop on spoken language translation (IWSLT) (pp. 105-111).

- Kevin Kilgour, Michael Heck, Markus Müller, Matthias Sperber, Sebastian Stüker, & Alex Waibel (2014). The 2014 KIT IWSLT speech-to-text systems for English, German and Italian. In Proceedings of the international workshop on spoken language translation (IWSLT) (pp. 73-79).

- Michael Heck, Sebastian Stüker, Sakriani Sakti, Alex Waibel, & Satoshi Nakamura (2013). Incremental unsupervised training for university lecture recognition. In Proceedings of the international workshop for spoken language translation (IWSLT) (pp. 251-255).

- Kevin Kilgour, Christian Mohr, Michael Heck, Quoc Bao Nguyen, Van Huy Nguyen, Evgeniy Shin, Igor Tseyzer, Jonas Gehring, Markus Müller, Matthias Sperber, Sebastian Stüker, & Alex Waibel (2013). The 2013 KIT IWSLT speech-to-text systems for German and English. In Proceedings of the international workshop on spoken language translation (IWSLT) (pp. 107-112).

- Michael Heck, Christian Mohr, Sebastian Stüker, Markus Müller, Kevin Kilgour, Jonas Gehring, Quoc Bao Nguyen, Van Huy Nguyen, & Alex Waibel (2013). Segmentation of telephone speech based on speech and non-speech models. In Proceedings of the international conference on speech and computer (SPECOM) (pp. 286-293).

- Michael Heck, Sebastian Stüker, & Alex Waibel (2012). A hybrid phonotactic language identification system with an SVM back-end for simultaneous lecture translation. In Proceedings of the IEEE international conference on acoustics, speech and signal processing (ICASSP) (pp. 4857-4860).

- Michael Heck, Keigo Kubo, Matthias Sperber, Sakriani Sakti, Sebastian Stüker, Christian Saam, Kevin Kilgour, Christian Mohr, Graham Neubig, Tomoki Toda, Satoshi Nakamura, & Alex Waibel (2012). The KIT-NAIST (contrastive) English ASR system for IWSLT 2012. In Proceedings of the international workshop on spoken language translation (IWSLT) (pp. 91-95).

- Christian Saam, Christian Mohr, Kevin Kilgour, Michael Heck, Matthias Sperber, Keigo Kubo, Sebastian Stüker, Sakriani Sakti, Graham Neubig, Tomoki Toda, Satoshi Nakamura, & Alex Waibel (2012). The 2012 KIT and KIT-NAIST English ASR systems for the IWSLT evaluation. In Proceedings of the international workshop on spoken language translation (IWSLT) (pp. 87-90).

- Jan Niehues, Mohammed Mediani, Teresa Herrmann, Michael Heck, Christian Herff, & Alex Waibel (2010). The KIT Translation system for IWSLT 2010. In Proceedings of the international workshop on spoken language translation (IWSLT) (pp. 93-98).

- Sebastian Stüker, Michael Heck, Katja Renner, & Alex Waibel (2010). Spoken news queries over the world wide web. In Proceedings of the international workshop on Searching spontaneous conversational speech (SSCS) (pp. 61-64).

Domestic Meetings

- Nurul Lubis, Michael Heck, Sakriani Sakti, Koichiro Yoshino, & Satoshi Nakamura (2018). Multimodal database of negative emotion recovery in dyadic interactions: Construction and analysis. In Proceedings of the Acoustical Society of Japan (ASJ) autumn seminar.

- Michael Heck, Masayuki Suzuki, Takashi Fukuda, Gakuto Kurata, & Satoshi Nakamura (2018). Distilling knowledge from a multi-scale deep CNN ensemble for robust and light-weight acoustic modeling. In Proceedings of the Information Processing Society of Japan (IPSJ) special interest group spoken language processing (SIG-SLP) seminar.

- Michael Heck, Masayuki Suzuki, Takashi Fukuda, Gakuto Kurata, & Satoshi Nakamura (2017). Knowledge distillation from a multi-scale VGG ensemble for acoustic modeling. In Proceedings of the Acoustical Society of Japan (ASJ) autumn seminar.

- Michael Heck, Sakriani Sakti, & Satoshi Nakamura (2017). Learning feature transformations without supervision to support DPGMM based representation learning. In Proceedings of the Acoustical Society of Japan (ASJ) spring seminar.

- Michael Heck, Quoc Truong Do, Sakriani Sakti, Graham Neubig, & Satoshi Nakamura (2016). The NAIST ASR for IWSLT: A multi-architecture DNN system combination approach. In Proceedings of the Acoustical Society of Japan (ASJ) spring seminar.

Theses

- Michael Heck (2012). Unsupervised acoustic model training for simultaneous lecture translation in incremental and batch mode. Diploma thesis.
- Michael Heck (2011). Automatic language identification for natural speech processing systems. Student research project thesis.

Talks

- Michael Heck. Supervised learning without supervision: Feature optimized DPGMM clustering for ZeroSpeech 2017. Invited Talk, NTT, Kyoto, Japan, 15th June 2018.
- Michael Heck. Learning feature transformations without supervision to Support DPGMM based representation learning. CLICS Winterschool, Nara Institute of Science and Technology, Nara, Japan, 11th December 2017.
- Michael Heck. Knowledge distillation from a multi-scale VGG ensemble for acoustic modeling. CLICS Winterschool, Nara Institute of Science and Technology, Nara, Japan, 11th December 2017.

Advised Theses

- Ludwig Linhuber (2013). Automatische Segmentierung und Gruppierung natürlicher Sprache anhand verschiedener Sprecher. Master thesis.