**A NONPARAMETRIC BAYESIAN APPROACH FOR
AUTOMATIC DISCOVERY OF A LEXICON AND ACOUSTIC UNITS**

Amir Hossein Harati Nejad Torbati

Jibo Inc.

Redwood City, California, USA

Amir.harati@gmail.com

*Joseph Picone*

College of Engineering, Temple University

Philadelphia, Pennsylvania, USA

joseph.picone@temple.edu

#### Abstract

State of the art speech recognition systems use context-dependent phonemes as acoustic units. However, these approaches do not work well for low resourced languages where large amounts of training data or resources such as a lexicon are not available. For such languages, automatic discovery of acoustic units can be important. In this paper, we demonstrate the application of nonparametric Bayesian models to acoustic unit discovery. We show that the discovered units are linguistically meaningful.

We also present a semi-supervised learning algorithm that uses a nonparametric Bayesian model to learn a mapping between words and acoustic units. We demonstrate that a speech recognition system using these discovered resources can approach the performance of a speech recognizer trained using resources developed by experts. We show that unsupervised discovery of acoustic units combined with semi-supervised discovery of the lexicon achieved performance (9.8% WER) comparable to other published high complexity systems. This nonparametric approach enables the rapid development of speech recognition systems in low resourced languages.

**Index Terms**: acoustic unit discovery, nonparametric Bayesian models, low resource speech recognition

# Introduction

Most state of the art automatic speech recognition (ASR) systems use some type of fundamental linguistically motivated acoustic unit such as a phoneme [1] or a syllable [2]. The challenge in acoustic modeling is training of statistical models for these units. In most high-performance systems, these units are extended by incorporating linguistic context. For example, in the case of phoneme-based systems, context-dependent phones based on three-phone sequences, or triphones as they are known, are employed. As a result, a typical speech recognition system includes tens of thousands of potential models and states, significantly increasing the complexity of the system and the amount of data required for training.

After establishing a set of acoustic units, an ASR system needs a mechanism to map the words into a sequence of these acoustic units. A dictionary or lexicon is often used for this purpose. A lexicon is simply a table that translate a word to several possible phonetic representations reflecting the different ways a word can be pronounced. In spite of its simplicity, a lexicon is often one of the most expensive resources needed in the development of a speech recognition system in a new language (or application). The availability of such linguistic resources often influences our choice of acoustic units since creating a new lexicon manually is not feasible in many situations and requires linguistic expertise not often available to technology developers.

In this paper, we present techniques to automate the development of such resources. We refer to languages for which such resources do not exist as low resources languages (LRLs). A good example of a family of LRLs that are gaining strategic importance are the African click languages. These will challenge our conventional notions of a lexicon and language model. Fortunately, we can utilize machine learning approaches to automatically learn acoustic units, lexicons and language models. Though traditional context-dependent phone models perform well when there is ample data, automatic discovery of acoustic units (ADU) offers the potential to provide good performance for resource deficient languages with complex linguistic structures. Since only a small fraction of the world’s 6,000 languages are currently supported by robust speech technology, this remains a critical problem.

Most approaches to automatic discovery of acoustic units [3]-[5] do this in two steps: segmentation and clustering. Segmentation is accomplished using a heuristic method that detects changes in energy and/or spectrum. Similar segments are then clustered using an agglomerative method such as a decision tree. Advantages of this approach include the potential for higher performance than that obtained using traditional linguistic units, and the ability to automatically discover pronunciation lexicons.

Most of the popular approaches to discovering a lexicon assume the existence of a word transcription [1][6][7]. Some approaches also require additional information such as time alignments of word transcriptions. In this paper, we propose the use of a nonparametric Bayesian (NPB) model for automatically discovering of acoustic units. In our formulation of the problem, the number of acoustic units is unknown.

One brute force approach to this problem is to exhaustively search through a model space consisting of many possible parameterizations. However in an NPB model [8][9], the model complexity can be inferred directly from the data. Segmenting an utterance into acoustic units can be approached in a manner similar to that used in speaker diarization, where the goal is to segment audio into regions that correspond to a specific speaker. Fox et al. ‎ used one state per speaker and demonstrated segmentation without knowing the number of speakers a priori. We have previously reported on the use of a similar model for speech segmentation [11] and spoken term detection by query [12] that achieves state of the art results. This paper is continuation of our previous work and includes the application of nonparametric hidden Markov model (HMM) to the problem of acoustic unit discovery [13][14].

The rest of the paper is organized as follows: in Section 2, some background material related to nonparametric approaches used in the rest of the paper is introduced. In Section 3, the ADU transducer is presented. In Section 4 we introduce an algorithm to learn the lexicon based on the output of an ADU transducer. In Section 5 some experimental results are discussed. These results include a comparison of automatically derived acoustic units and lexicons to manually optimized versions on a standard speech recognition task.

**Relationship to Previous Work:** In [11] we have used an NPB model for speech segmentation that achieves state of the art performance for unsupervised algorithms. In [12] we have used NPB model to learn ADU units and applied it to the problem of spoken term detection by query. The work presented here is a continuation of this work.

Bacchiani and Ostendorf [3] proposed an algorithm to jointly discover the acoustic units and the lexicon. In their algorithm, they have assumed the alignment for words are given and they learn one pronunciation for all examples of a single word. In comparison, we don’t restrict the number of pronunciation variants and let the data speak for itself. This is an extremely important difference in the two works and a critical part of the NPB approach. We also discover the lexicon and acoustic units in two successive steps. One could argue that discovering the lexicon and acoustic units in separate steps results in a suboptimal algorithm. However, we show that our model produces competitive results.

Paliwal [5] also proposed several methods to discover a lexicon for isolated word speech recognition applications. These methods learn multiple pronunciations per word but in their current form can’t be used for continuous speech. Fukadai  [15] proposed a similar model that also needs word alignments. Singh et al.  [6] proposed an approach to estimate the lexicon along with the acoustic units in a probabilistic framework. Their approach involves initializing the lexicon with a heuristic method and then iteratively discovering the lexicon and acoustic units. Our semi-supervised method also needs to be initialized with some approximate word alignments and then iteratively reestimates the lexicon and word alignments.

Finally, Lee [7] proposed a model that discovers the lexicon by first learning a mapping between letters in a word and acoustic units and then generating pronunciations by connecting these mappings for each word. In our approach, we also use letters to initialize our semi-supervised algorithm. However, unlike Lee [7] our algorithm learns the pronunciation directly from examples and is not strongly dependent on using letters.

# Background

A Dirichlet process (DP) [16] is a discrete distribution that consists of a countably infinite number of probability masses. A DP is denoted by DP(α,H), and is defined as:



where α is the concentration parameter, *H* is the base distribution, and  is the unit impulse function at *θk*, often referred to as an atom [17]. The weights *βk* are sampled through a stick-breaking construction [18] and are denoted by *β~GEM(α)*. One of the applications of a DP is to define a nonparametric prior distribution on the components of a mixture model that can be used to define a mixture model with an infinite number of mixture components [17].

An HDP extends a DP to grouped data [19]. In this case there are several related groups and the goal is to model each group using a mixture model. These models can be linked using traditional parameter sharing approaches. One approach is to use a DP to define a mixture model for each group and to use a global DP, DP(γ,H), as the common base distribution for all DPs [19]. An HDP is defined as:



where *H* provides a prior distribution for the factor *θji*, *γ* governs the variability of *G0* around *H* and *α* controls the variability of *Gj*around *G0. H, γ* and *α* are hyperparameters of the HDP. We use a DP to define a mixture model for each group and use a global DP, DP*(γ,H)*, as the common base distribution for all DPs.

An HDPHMM [10] is an HMM with an unbounded number of states. The transition distribution from each state is modeled by an HDP. This lets each state have a different distribution for its transitions while the set of reachable states would be shared amongst all states. The definition for HDPHMM is given by [10]:



The state, mixture components and observations are represented by *zt,* *st* and *xt* respectively. The indices *j* and *k* are indices of the state and mixture components respectively. The base distribution that links all DPs together is represented by *β* and can be interpreted as the expected value of state transition distributions. The transition distribution for state *j* is a DP denoted by *πj* with a concentration parameter α. Another DP, *ψj*, with a concentration parameter *ϭ*, is used to model an infinite mixture model for each state (*zj)*. The distribution *H* is the prior for the parameters *θkj*.

#  An ADU Transducer

The goal in speech segmentation is to map each acoustic observation into a segment and optionally label these segments. Our goal can be expressed as mapping a string of acoustic observations to a string of labels. In speech recognition, observations are vectors of real numbers (instead of symbols in text processing) and segment labels can be replaced with a vector that represents the posterior probability of a set of predefined symbols. This representation is called a posteriorgram [20].

A transducer specifies a binary relationship for a pair of strings [21]. Two strings are related if there is a path in the transducer that maps one string to the other. A weighted transducer also assigns a weight for each pair of strings [21]. Based on this definition our problem is to find a transducer that maps a string of acoustic features onto a string of units. It should be noted that based on this definition any HMM can be considered to be a transducer. We chose the term transducer here to emphasize the operation of converting acoustic observations into acoustic units. The problem can be further divided into two sub-problems: learning a transducer and decoding a string of observations into a string of units (or their equivalent posteriorgram representation).

Let’s assume we already knew the acoustic units (e.g. phonemes) and have trained models for each unit (e.g. HMMs). One way to construct a transducer is to connect all these HMMs using an ergodic network. The final transducer can be some form of ergodic HMM. However, we don’t have the units and the number of units in the data is unknown.

In [11] we used HDPHMM for speech segmentation. In [14] we introduced a DHDPHMM that allows sharing mixture components across states. These models can learn many different structures including ergodic structures. Both of these models are good candidates to train a transducer. A C++ implementation of both algorithms, including DPM and HDP, is available at [22].

In this paper, we use an HDPHMM to train the transducer. Learning an HDPHMM is extensively discussed in [10][14]. Here we train the model in a completely unsupervised fashion. We don’t utilize a speech/non-speech classifier and model everything including silence with one transducer. For read speech, this does not present any problems. However, for other domains such as conversational speech, it might be a problem, and in that case we can employ a speech/non-speech classifier as well. Training is executed by sequentially presenting utterances to the HDPHMM inference algorithm and iterating using Gibbs sampling.

For our transducer, state labels (or their posteriorgrams) are the output string. Since each state is modeled by a Gaussian mixture, the segments defined by this transducer are stationary and the discovered units are sub-phonetic. However, it should be noted that this limitation can be overcome by replacing each state (e.g. mixture model) with an HMM which transforms the model into a hierarchical HMM [23]. The resulting model can model dynamic segments.

Given a transducer and a string of observations the goal of the decoder is to find the most likely path through states of the transducer that implicitly maps the input string to the output string. This objective can be written as:



where *s1*, *s2*, ..., *sM* represent state labels and *o1, o2, ..., oN* represent observations (e.g. *N* observations are mapped to *M* states of HMM)*.* Alternately, we can also estimate the posteriorgram of the state sequence. To optimize  we can utilize the Viterbi algorithm [24]. The resulting transducer is the engine used to convert new acoustic observations into acoustic units.

# Semi-Supervised Lexicon Learning

State of the art ASR systems are based on building statistical models for some form of acoustic units (e.g. phonemes). A lexicon is a table that maps all observable units in an ASR system, typically words, into a sequence of these units. A lexicon is usually developed by an expert linguist and its preparation is a difficult and expensive task. The task of preparing a lexicon becomes even more delicate when we don’t use standard acoustic units like phonemes or other linguistically-defined units. In this section we introduce an algorithm to learn the lexicon given an ADU transducer (which defines the set of ADU units as well) and a corpus of acoustic data with parallel transcriptions.

The algorithm assumes the existence of a parallel transcription with acoustic data but does not assume existence of exact alignments. The algorithm needs to be initialized using a heuristic approach, but this initialization does not need to be accurate and can be easily generated with available resources (e.g. word transcriptions and acoustic data). After aligning the transcription with stream of ADU units, we can generate a mapping between words and AUU units. However, there might be many examples for each word and we need to select a handful that represents each word more accurately. There are many ways to find these representative examples. For example, we can cluster the examples and then select the centroids. In this section, we propose an algorithm that selects at most *R* examples among all instances of a given word that have the average minimum edit distance [25] from other examples. The edit distance is computed using DTW. Posteriorgrams of the states are used to represent each example.

The algorithm is as follows:

1. Generate the posteriorgram representation for all utterances in the dataset using an ADU transducer.
2. Generate an approximate alignment between the words and the output stream of the ADU transducer
3. Use the aligned transcription to extract all examples of each word.
4. Compute the DTW alignment between each two examples *X* and *Y* [26]:



where *X* and *Y* are the two examples, *p* is the warping path that aligns *X* and *Y* and *Cp(X,Y)* is defined as:



1. For all *k*, accumulate the distance between the *k-th* example and all other examples in the data set.
2. Select the *R* examples with a minimum average distance as representatives for that word.
3. Convert the posteriorgram of *R* examples into state labels, remove repetitions and retain the remaining *R* examples (e.g. the number of final examples for each word is less than or equal to *R*).
4. Use the lexicon and associated acoustic units generated in Step 7 to build a speech recognizer.
5. Use the speech recognizer built in Step 8 to force align the transcriptions with the acoustic units.
6. Use the aligned transcriptions to extract all examples of each word.
7. If convergence is not achieved and the number of iterations is less than a specified threshold, go back to Step 4. The convergence criterion can be the WER computed on a small development set.

Note that in this formulation *c(xnl,,yml)* is an element of the cost matrix between *X* and *Y* (row *n* and column *m)*. For our problem, we define the cost between two posteriorgram vectors as a dot product between them [27]:



At the end, this algorithm will find at most *R* pronunciations for each word. This algorithm selects instances with the least average edit distance from the other training examples.

# Experiments

In this section some experimental results are presented. First, ADU units are compared with phonemes. Second, we present the results of semi-supervised lexicon learning algorithm discussed in the previous section. In this section we have trained an ADU transducer on the training subset of TIMIT [28]. However, learning and evaluation of the lexicon is performed using Resource Management (RM) Corpus [29]. The lexicon is trained on a training subset of RM and we have used g2p [30] to augment it with all the words that exist in both the training and evaluation subsets. The evaluation is performed on test subset of RM. The reason for learning the ADU transducer on TIMIT and to learn and evaluate the lexicon on RM was to evaluate the generalization properties of the ADU units. This also allows us to compare our results to other published results.

## Relationship to Phonemes

It is important to explore the relationship between the ADUs and phonemes because we need to determine if the ADUs are linguistically meaningful. The first experiment involves aligning manually transcribed phonemes with ADUs. First, each utterance is passed through the transducer to generate the sequence of ADUs. Then these ADUs are aligned with manual transcriptions using timing information contained in the transcription. Finally, a confusion matrix is calculated.

Table 1- RM lexicon experiments

|  |  |  |  |
| --- | --- | --- | --- |
| Experiment | Context Modeling | Mixture No. | WER (%) |
| CI Baseline | CI | 1 | 24.66 |
| CI Baseline | CI | 16 | 9.17 |
| CD Baseline | CD | 1 | 10.64 |
| CD Baseline | CD | 16 | 4.84 |
| NP algorithm | CI | 1 | 20.34 |
| NP algorithm | CI | 16 | 11.61 |
| CD NP algorithm | CD | 1 | 15.81 |
| CD NP algorithm | CD | 16 | 9.81 |

A confusion matrix between 48 English phonemes and 251 ADU units is shown in Figure 1. A general correlation between ADUs and phonemes can be observed because the diagonal region of the matrix is heavily populated. However, the mapping is not consistently one to one. Some of the ADUs align with multiple phonemes. These phonemes are generally similar phonemes. For example, we can see ADUs that are aligned with “sil” (silence) can also be aligned with “vcl” and “cl” models (both “vcl” and “cl” are special types of silence). ADUs aligned with “z” can also be aligned with “s”. This is not surprising because “z” and “s” are similar acoustically and therefore confusable.

## An Automatically Learned Lexicon

To compare the quality of the lexicon discovered using ADUs and to evaluate the algorithms presented in this paper, we use speech recognition experiments. We compute the word error rate (WER) for both the baseline system that is trained based on a standard lexicon and standard phonetic units to a system that uses an automatically discovered lexicon. We have used the training subset of TIMIT [28] (3,637 utterances) to train the ADU transducer in a completely unsupervised manner.

For evaluation, we have used the RM Corpus [29]. The reason was to examine how the learned ADU units perform on a new dataset. Also other authors have published some results using the RM Corpus, so this allows us to compare our algorithms. All utterances were first passed through the ADU transducer to obtain their posteriorgram representation. Then the lexicon was generated using the lexicon discovery algorithm discussed in Section 4.

Table 1 shows the results obtained for RM. We can see that a CI system trained using an automatically discovered lexicon and 1 mixture component per state works better than a similar system using the reference lexicon (relatively 21%). However, the performance is slightly worse when using more mixture components (9.17% vs. 11.61%). This observation can be explained by considering the fact that ADUs are stationary units (corresponding to single Gaussian distributions). Once we increase the complexity of the system the improvement in their performance would be less than the improvement for more dynamic units such as phonemes (e.g. phonemes are less homogenous). Further, there are more ADUs than phonemes (in this case we have 251 ADUs while 39 phonemes) which means we would have less data per ADU for training. The number of parameters to be estimated increases significantly when the number of mixture components per state increases from 1 to 16, so we expect these models to be more poorly estimated since the amount of data is fixed.



Figure 1- An ADU phoneme confusion matrix

We have present the results for CD-trained systems in . For the baseline system we have used a phonetic decision tree based on linguistic knowledge to generate tied states. For the ADU-based systems we don’t have linguistic knowledge (e.g. similar ADU units can be grouped). In principle it is possible to use a data-driven approach to obtain an approximation of such knowledge (e.g. clustering ADUs), but in this research we chose a simple approach based on singleton questions. Instead of using questions that are categorical in nature (e.g. is the left phoneme/unit a stop?) we let each ADU be a single group with only one member and have questions such as “is the left unit *u1*?”

The result is less powerful than a tree trained by all possible questions. We can see from Table 1 that the gain for the CD ADU system (compare the last two rows to rows 3 and 4) is less than a CD system based on the reference lexicon and phonemes. Part of this is due to the singleton questions and part is a result of the increased number of ADUs compared to phonemes. Nevertheless, the results in Table 1 show unsupervised ADUs and a semi-supervised lexicon trained based on these ADUs provides results comparable to the reference lexicon.

Table 2 shows the results of several competitive algorithms. References [3][6][7] are representative work in this area. The first two approaches can be compared directly on RM. Lee [7] evaluated their approach on the Jupiter Corpus [31] which consisted of spoken queries for weather information. We cannot compare directly to results by Lee [7] because two datasets are different. Nevertheless, their results are presented in Table 2 and show a similar trend to ours. For example, their CI system is 19% worse than their CI baseline while our CI system is 21% worse than our baseline.

Table 2- A comparison of several discovery algorithms

|  |  |  |  |
| --- | --- | --- | --- |
| System | ContextModeling | Corpus | WER (%) |
| Bacchiani CI low-complexity [3] | No | RM | 19.70 |
| Bacchiani High complexity [3] | Word Context | RM | 11.40 |
| Bacchiani CD low-complexity (1mix) [3] | PhonemeContext | RM | 13.70 |
| Bacchiani CD high-complexity [3] | Phoneme +Word Context | RM | 9.90 |
| Singh CI probabilistic framework [6] | No | RM | 20.00 |
| Singh CI phoneme baseline [6] | No | RM | 15.00 |
| CI Nonparametric Bayesian [7] | No | Jupiter | 17.00 |
| CD Nonparametric Bayesian [7] | Phoneme Context | Jupiter | 13.40 |
| CI phoneme baseline Jupiter [7] | No | Jupiter | 13.80 |
| CD phoneme baseline Jupiter [7] | Phoneme Context | Jupiter | 10.00 |

From this table we can see the low-complexity systems of [3] (rows 1 and 3) have similar performance to an ADU-based system with 1 mixture component per state (rows 5 and 8). However, their algorithm is much more complex than the ADU-based system and involves joint discovery of the lexicon and acoustic units in a supervised manner (words alignments are required). The high-complexity system (rows 2 and 4) of [3] performs better than other systems that use *1* mixture component per state.

However, this system is based on another sophisticated algorithm – progressive refinement of a low-complexity system with additional iterations using the K-MEANS and Viterbi algorithms. This high-complexity system implicitly models the context (word context in this case) and therefore should be compared with context dependent systems.

Singh et al. [6] used a slightly different baseline (e.g. semi-continuous HMMs) and a different language model with a higher perplexity. Their result is 25% worse than their baseline while our result for a 1 mixture system is 21% better than our baseline. Their degradation in performance might be a result of the fact that they have used a system with a high-perplexity language model.

Finally, the nonparametric Bayesian approach of Lee [7] produces similar trends compared to our system. For example, their result is 19% worse than their baseline while our result (16 mixture) is 21% worse than our baseline. We must emphasize that all of these systems learned their corresponding acoustic units jointly with the lexicon on the same corpus while we intentionally introduced a mismatch to investigate the generalization performance of the ADU transducer. We also trained the lexicon separately from the ADU units. We expect we would obtain better results if we trained our ADU transducer using the same corpus.

# Conclusions

In this paper we proposed the application of HDPHMM to the problem of learning acoustic units automatically. We have shown discovered ADU units have a meaningful relationship with phonemes. We have also proposed a semi-supervised algorithm to learn the lexicon from a parallel transcription that maps the words into these ADU units. We have shown our system is competitive with other state of the art systems, and that our ADU units can generalize to new datasets.

In the future, we intend to study an NPB model that models nonstationary units. As mentioned above, our current model assumes each unit can be represented with a Gaussian mixture. As a result, our model discovers sub-phonetic units. If we can model each state of the HDPHMM with another HMM then this limitation would be eliminated.

# Acknowledgements

This research was supported in part by the National Science Foundation through Major Research Instrumentation Grant No. CNS-09-58854 and through a Temple University dissertation completion grant.

# References

1. K.-F. Lee, “Context-Independent Phonetic Hidden Markov Models for Speaker-Independent Continuous Speech Recognition,” *IEEE Trans. on ASSP*, vol. 38, no. 4, pp. 599– 609, 1989.
2. A. Ganapathiraju, et al., “Syllable-Based Large Vocabulary Continuous Speech Recognition,” *IEEE Trans. On SAP*, vol. 9, no. 4, pp. 358–366, May 2001.
3. M. Bacchiani and M. Ostendorf, “Joint lexicon, acoustic unit inventory and model design,” *Speech Communication*, vol. 29, no. 2–4, pp. 99–114, 1999.
4. B. Ma, et al., “An Acoustic Segment Modeling Approach to Automatic Language Identification,” in *Proceedings of INTERSPEECH*, 2005, pp. 2829–2832.
5. K. Paliwal, “Lexicon-building methods for an acoustic sub-word based speech recognizer,” in *Proceedings of the IEEE International Conference on ASSP*, 1990, pp. 729– 732.
6. R. Singh, B. Raj, and R. M. Stern, “Automatic Generation of Subword Units for Speech Recognition Systems,” *IEEE Transactions on SAP*, vol. 10, no. 2, pp. 89–99, 2002.
7. C. Lee, “Discovering Linguistic Structures in Speech: Models and Applications,” Massachusetts Institute of Technology, 2014.
8. P. Muller and F. A. Quintana, “Nonparametric Bayesian Data Analysis,” *Statistical Science*, vol. 19, no. 1, pp. 95–110, Feb. 2004.
9. A. Harati, J. Picone, and M. Sobel, “Applications of Dirichlet Process Mixtures to Speaker Adaptation,” in *Proceedings of the International Conference on ASSP*, 2012, pp. 4321–4324.
10. E. Fox, et al., “A Sticky HDP-HMM with Application to Speaker Diarization.,” *The Annalas of Applied Statistics*, vol. 5, no. 2A, pp. 1020–1056, 2011.
11. A. Harati and J. Picone, “Speech Acoustic Unit Segmentation Using Hierarchical Dirichlet Processes,” in *Proceedings of INTERSPEECH*, 2013, pp. 637–641.
12. A. Harati and J. Picone, “A Nonparametric Bayesian Approach for Spoken Term Detection by Example Query,” in *Proceedings of INTERSPEECH*, 2016, (in press).
13. A. Harati Nejad Torbati, J. Picone, and M. Sobel, “A Left-to-Right HDP-HMM with HDPM Emissions,” in *Proceedings of the CISS*, 2014, pp. 1-6.
14. A. Harati and J. Picone, “A Doubly Hierarchical Dirichlet Process Hidden Markov Model with a Non-Ergodic Structure,” *IEEE/ACM Transactions on ASLP*, vol. 24, no. 1, pp. 174–184, 2016.
15. T. Fukadai, M. Bacchiani, K. Paliwal, and Y. Sagisaka, “Speech Recognition Based on Acoustically Derived Segment Units,” in Proceedings of ICSLP, 1996, pp. 1077–1080.
16. Y.-W. Teh, “Dirichlet process,” in *Encyclopedia of Machine Learning*, Springer, 2010, pp. 280–287.
17. C. E. Rasmussen, “The Infinite Gaussian Mixture Model,” in *Proceedings of Advances in Neural Information Processing Systems*, 2000, pp. 554–560.
18. J. Sethuraman, “A constructive definition of Dirichlet priors,” *Statistica Sinica*, vol. 4, no. 2, pp. 639–650, 1994.
19. Y. Teh, M. Jordan, M. Beal, and D. Blei, “Hierarchical Dirichlet Processes,” *Journal of the American Statistical Association*, vol. 101, no. 47, pp. 1566–1581, 2006.
20. Y. Zhang and J. R. Glass, “Unsupervised Spoken Keyword Spotting via Segmental DTW on Gaussian Posteriorgrams,” in *Proceedings of the IEEE Workshop on ASRU*, 2009, pp. 398–403.
21. M. Mohri, F. Pereira, and M. Riley, “Speech Recognition with Weighted Finite-State Transducers,” in *Springer Handbook on Speech Processing and Speech Communication*, 2008, pp. 559–584.
22. Harati Nejad Torbati, “HDPHMM LIB”, Available: *https://github.com/amir1981/hdphmm\_lib*.
23. S. Fine, Y. Singer, and N. Tishby, “The Hierarchical Hidden Markov Model: Analysis and Applications,” *Machine Learning*, vol. 32, no. 1, pp. 41–62, 1998.
24. A. Viterbi, “Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm,” *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 260–269, Apr. 1967.
25. G. Navarro, “A Guided Tour to Approximate String Matching,” *ACM Computing Surveys*, vol. 33, no. 1, pp. 31–88, 2001.
26. M. Müller, “Dynamic Time Warping,” in *Information Retrieval for Music and Motion*, Springer, 2007, pp. 69–82.
27. Y. Zhang and J. R. Glass, “Unsupervised Spoken Keyword Spotting via Segmental DTW on Gaussian Posteriorgrams,” in *Proceedings of ASRU*, 2009, pp. 398–403.
28. J. Garofolo, L. Lamel, W. Fisher, J. Fiscus, D. Pallet, N. Dahlgren, and V. Zue, “TIMIT Acoustic-Phonetic Continuous Speech Corpus,” *The Linguistic Data Consortium Catalog*, 1993.
29. P. Price, W. Fisher, J. Bernstein, and D. Pallett, “The DARPA 1000-Word Resource Management Database for Continuous Speech Recognition,” in *Proceedings of the IEEE International Conference on ASSP*, 1988, pp. 651–654 vol. 1.
30. J. R. Novak, N. Minematsu, and K. Hirose, “WFST-Based Grapheme-to-Phoneme Conversion: Open Source Tools for Alignment, Model-Building and Decoding,” in *International Workshop on Finite State Methods and NLP*, 2012, pp. 45–49.
31. V. Zue, et al., “Jupiter: A Telephone-Based Conversational Interface for Weather Information,” *IEEE Trans. Speech and Audio Processing,* vol. 8, no. 1, pp. 85–96, 2000.