Unsupervised Word Discovery from Phonetic Input Using Nested Pitman-Yor Language Modeling

Oliver Walter*, Reinhold Haeb-Umbach*, Sourish Chaudhuri** and Bhiksha Raj**

Abstract— **In this paper we consider the unsupervised word discovery from phonetic input. We employ a word segmentation algorithm which simultaneously develops a lexicon, i.e., the transcription of a word in terms of a phone sequence, learns a n-gram language model describing word and word sequence probabilities, and carries out the segmentation itself. The underlying statistical model is that of a Pitman-Yor process, a concept known from Bayesian non-parametrics, which allows for an a priori unknown and unlimited number of different words. Using a hierarchy of Pitman-Yor processes, language models of different order can be employed and nesting it with another hierarchy of Pitman-Yor processes on the phone level allows for backing off unknown word unigrams by phone m-grams. We present results on a large-vocabulary task, assuming an error-free phone sequence is given. We finish by discussing options how to cope with noisy phone sequences.**

I. INTRODUCTION

Unsupervised language acquisition is the task of acquiring components of a language without any supervision. Our goal is to acquire these components directly from audio recordings of continuous speech.

This task may be subdivided into two subtasks: a) the discovery of the basic acoustic building blocks of speech (the phones or something alike), and b) the discovery of lexical elements (such as words) which manifest themselves as recurring sequences of the basic acoustic building blocks. While the former is related to finding and clustering acoustically similar patches of speech, the latter is closely related to structure discovery, as language has strong structural elements with few probable words and word sequences and many less probable ones, which can be efficiently described by statistical language models.

Decomposing the unsupervised language acquisition task into this two-layered hierarchical approach not only simplifies the problem but also goes in hand with the two layers of human language: the low-level acoustic and the high-level textual layer.

A practical application for the unsupervised learning of a language is the unsupervised training of an automatic speech recognizer directly from audio recordings, without using labeled data, i.e., without knowing the text that has been spoken [1], [2]. This is of particular interest because obtaining labeled training data can be a very tedious task and is costly. Of particular interest are semi-supervised approaches which employ a little amount of labeled data for initialization and large amounts of unlabeled data for final model traning. An interesting application is the bootstrapping of a recognizer for a low-resource language, where little or no annotated training data is available, from an acoustically similar language, of which labeled training data is abundant. A phoneme recognizer trained on the high-resource language can then be employed to transcribe the unlabeled data of the low-resource language into a phone sequence. Subsequently, with the techniques discussed in this paper a dictionary and a language model can be learned in an unsupervised manner for the low-resource language.

Another example is a machine, for example a robot, which should be able to autonomously learn about it's environment and communicate with humans. By solving tasks a) and b) above, the machine learns while listening to the human interacting with it.

Yet another example is the semantic analysis of audio [3]. Here, the input signal is audio, rather than speech, and the task is to interpret a scene based on the sound events recorded. Low-level audio patterns (e.g., hitting a ball, applause) form high-level acoustic events (scoring of a goal in a soccer match), and the task is to learn this ”acoustic language” to infer the activity on the soccer field.

Note that the unsupervised training of the speech recognizer does not provide semantics for the found patterns. In an application, such as robot teaching, such semantics must come from another knowledge source, e.g., by a human or by another modality (video), such that audio-visual correlates can be defined. Note, however, that only rather high-level labels are required that define an action, rather than a low-level transcription of the spoken input. The grounding of the discoved word sequences, however, is beyond the scope of this paper.

In [4] using very similar statistical modes to those to be used here, human motions were segmented which could then be used to train a robot to mimic these motions.

Concerning task a) above, several solutions can be found in the literature. Variants of dynamic time warping (DTW) and different clustering techniques have been proposed to find recurrent sequence pattern in audio [5], [6]. In [2] we used dynamic time warping and k-means++ clustering to cluster single digit utterances. We also experimented with connected digit sequences and were able to train a speech recognizer for connected digit sequences in an unsupervised manner at a word error rate of about 13% [7]. All techniques,

*Oliver Walter and Reinhold Haeb-Umbach are with the Department of Communications Engineering, University of Paderborn, Germany. {walter,haeb}@nt.uni-paderborn.de

**Sourish Chaudhuri and Bhiksha Raj are with the Language Technologies Institute, Carnegie Mellon University, Pittsburgh, USA. {sourishc,bhiksha}@cs.cmu.edu

The work was in part supported by Deutsche Forschungsgemeinschaft under contract no. Ha 3455/9-1 within the Priority Program SPP1527 ”Autonomous Learning”

however, suffer from a high computational load.

An alternative way of finding basic acoustic buildings blocks of a language is the use of sparse coding approaches. For example, non-negative matrix factorization (NMF) can be employed to decompose an acoustic recording into a set of basic building blocks and their activations [8]. In [9] histogram of acoustic co-occurrences, defined from an NMF analysis are used for unsupervised phone discovery. A shortcoming of the basic NMF is that temporal correlations, which are so typical of speech, are not captured well. In [10] an extension to standard NMF was proposed, where a time series of acoustic vectors is considered as a basic object. One disadvantage of this so-called convolutive NMF is that it cannot model differences in speaking rate well. To overcome this disadvantage [11] proposed the chaining of basic elements in an object using a Markov chain. Using a Markov chain allows to consider variations in speaking rate and duration for the basic objects.

Including task b) above, in [3] a hierarchical approach is described for unsupervised structure discovery for semantic analysis of audio. On the first level basic acoustic patterns, called acoustic unit descriptors (AUDs), representing a basic sound, are discovered using a clustering approach. On the next level patterns consisting of sequences of AUDs, called events, are discovered. An event, which corresponds to the notion of a word in speech, is defined by a characteristic distribution of AUDs. The used model, however, does not capture any temporal information as to how AUD sequences evolve over time to define events. This is an obvious shortcoming, because not only the presence of certain acoustic patterns but, equally important, the temporal succession of these patterns define the units at the higher abstraction level, i.e., the events.

The discovery of recurrent phone sequences that form words is similar to the problem of unsupervised word segmentation. In [12], a Bayesian approach is presented, where both the lexicon (the transcription of a word in terms of characters) and the language model (i.e., the n-gram probability of words) is simultaneously estimated along with the segmentation of the given character string into words. The language model is based on the Pitman-Yor process, which provides a random distribution over discrete probability distributions over infinite sample spaces (in our case the set of words, which may be infinitely large) [13]. This model has a number of advantages. First, the number of words need not be known in advance and can be, in fact, arbitrarily large. Second, the model focuses on the temporal information, i.e. a word is described by its characteristic sequence of acoustic units. Third, a priori information is accounted for. One might wonder which a priori information is available in a completely unsupervised setup. Well, the Zipf's law, which states that the frequency of a word is inversely proportional to its rank in the frequency table, holds universally for many languages, both natural and artificial [14].

We adopt the unsupervised word segmentation approach of [12] here to discover words from phoneme input. For the word segmentation, a hierarchical Pitman-Yor process is employed, where the hierarchy represents different values of the history size $n-1$ of the n-gram language model. A phone language model is nested within the word language model such that the unigram word model is backed of by a m-gram model at the phone (or character level). We will show how progressively more refined language models improve precision and recall of the discovered words. We are not concerned with how to obtain the phone sequence from audio, though, and refer to prior work [2] [7].

The paper is organized as follows. In the next section we give an overview of the unsupervised word segmentation approach of [12] and show how it is applied to the task of phone segmentation. Next, the basic concept of hierarchical Pitman-Yor language models is presented. In section IV we present some results using character and phone sequences as input to the segmentation algorithm. Finally in section V we give an outlook for further developments.

II. UNSUPERVISED WORD SEGMENTATION

In the following we will describe the algorithm used for unsupervised word segmentation of [12].

The problem of word segmentation can be described as follows: Given a character sequence (sentence) c_1^T of T characters c_t

$$c_1^T = [c_1, \dots, c_T]\,, \tag{1}$$

we want to segment this sequence into I words w_i

$$\mathbf{w} = [w_1, \dots, w_I] \tag{2}$$

in which the words don't overlap and completely cover the character sequence. In the experiments reported later c_1^T will be either a character string or a phone sequence.

The word sequence is considered as being generated by a Markov chain of some order $n-1$. This leads to a context of $n-1$ words for the current word and results in the predictive probability

$$w_i \sim \Pr\left(w_i | w_{i-n+1}, \dots, w_{i-1}\right). \tag{3}$$

This is generally called a n-gram language model. The joint probability of a word sequence can then be calculated using

$$\Pr(\mathbf{w}) \approx \prod_{i=1}^{I} \Pr\left(w_i | w_{i-n+1}, \dots, w_{i-1}\right). \tag{4}$$

Given the predictive probability calculated using a n-gram language model and a lexicon mapping words to character or phone sequences, we have a generative model for generating sentences consisting of character or phone sequences making up words. To do the segmentation of a character or phone sequence into words we have to find the word segmentation that maximizes the probability of the word segmentation given a sequence of characters:

$$\hat{\mathbf{w}} = \underset{\mathbf{w}}{\operatorname{argmax}}\ \Pr(\mathbf{w}|c_1^T) \tag{5}$$

This can be done by using a generalized forward backward algorithm.

Let us first consider the unigram case $n = 1$. Let $\alpha[t]$ denote the probability of the character string c_1^t ending at

t. Further, let the variable q_t denote the distance to the last ending word. To be specific, $q_t = k$ denotes that $c_{t-k+1}, \dots, c_t =: c_{t-k+1}^t$ is a word beginning at $t-k+1$ and ending in t. Then we can derive the following recursion:

$$\begin{aligned}
\alpha[t] = \Pr(c_1^t) &= \sum_{k=1}^{t} \Pr(c_1^t, q_t = k) && (6)\\
&= \sum_{k=1}^{t} \Pr(c_1^{t-k}, c_{t-k+1}^t) && (7)\\
&= \sum_{k=1}^{t} \Pr(c_{t-k+1}^t | c_1^{t-k}) \Pr(c_1^{t-k}) && (8)
\end{aligned}$$

where we indicated the word boundary at $t-k$ by the sub- and superscripts of the character strings.

Using the unigram approximation, $\Pr(c_{t-k+1}^t | c_1^{t-k}) = \Pr(c_{t-k+1}^t)$, we arrive at

$$\begin{aligned}
\alpha[t] &= \sum_{k=1}^{t} \Pr(c_{t-k+1}^t) \Pr(c_1^{t-k})\\
&= \sum_{k=1}^{t} \Pr(c_{t-k+1}^t) \alpha[t-k]. && (9)
\end{aligned}$$

which is a recursion on the α variable. The recursion is started with $\alpha[0] = 1$.

The bigram case is more interesting. Let $\alpha[t][k]$ represent the probability of the string c_1^t, with the last k characters being a word. We can develop the following recursion:

$$\begin{aligned}
\alpha[t][k] &= \Pr(c_1^t, q_t = k)\\
&= \sum_{j=1}^{t-k} \Pr(c_1^t, q_t = k, q_{t-k} = j) && (10)\\
&= \sum_{j=1}^{t-k} \Pr(c_1^{t-k-j}, c_{t-k-j+1}^{t-k}, c_{t-k+1}^t) && (11)\\
&= \sum_{j=1}^{t-k} \Pr(c_{t-k+1}^t | c_{t-k-j+1}^{t-k}, c_1^{t-k-j})\\
&\quad \Pr(c_1^{t-k-j}, c_{t-k-j+1}^{t-k}). && (12)
\end{aligned}$$

Using the bigram approximation, we obtain for $k < t$:

$$\begin{aligned}
\alpha[t][k] &= \sum_{j=1}^{t-k} \Pr(c_{t-k+1}^t | c_{t-k-j+1}^{t-k}) \Pr(c_1^{t-k-j}, c_{t-k-j+1}^{t-k})\\
&= \sum_{j=1}^{t-k} \Pr(c_{t-k+1}^t | c_{t-k-j+1}^{t-k}) \Pr(c_1^{t-k}, q_{t-k} = j)\\
&= \sum_{j=1}^{t-k} \Pr(c_{t-k+1}^t | c_{t-k-j+1}^{t-k}) \alpha[t-k][j]. && (13)
\end{aligned}$$

For $k = t$ we use:

$$\alpha[t][t] = \Pr(c_1^t | < s >) \alpha[0][0]. \qquad (14)$$

where the recursion is initialized with $\alpha[0][0] = 1$. We assume a sentence start marker $< s >$ at the beginning of the string.

In the backward step we will use Gibbs sampling. We iteratively resample the word segmentation by drawing the length j of the previous word as shown in Algorithm 1 [12]. We assume a sentence end marker $< /s >$ at the end of the sentence.

Algorithm 1 Backward sampling

```
$t = T + 1, k = 1, i = 0, w_0 =< /s >$
while $t > k$ do
    $j \sim \Pr\left(w_i | c_{t-k-j+1}^{t-k}\right) \alpha[t-k][j]$
    $w_{i+1} = [c_{t-k-j+1}, \dots, c_{t-k}]$
    $i = i + 1$
    $t = t - k$
    $k = j$
end while
```

III. Pitman-Yor Language Model

The presented segmentation algorithm relies on unigram and bigram language model probabilities. However, these are not known and need to be learnt along with the segmentation. Furthermore, the number of words and their definitions in terms of character sequences changes as the hypothesized segmentation changes. Because of this we need a language model which can deal with unknown words, an unknown number of words, and which allows to raise or lower word probabilities according to the hypothesized segementation, including the case that words are completely removed from the lexicon if they are no longer present in the segmentation.

In [12] the so called Nested Pitman-Yor language model (NPYLM) is used. The NPYLM is an extension to the Hierarchical Pitman-Yor language model (HPYLM) presented in [13]. Finally the hierarchical Pitman-Yor language model is a language model based on the Pitman-Yor process [15], a Bayesian nonparametric model.

The Pitman-Yor process provides a random distribution over discrete probability distributions over infinite sample spaces and is a generalization of the Dirichlet process:

$$G \sim \mathrm{PY}\left(d, \theta, G_0\right). \qquad (15)$$

A draw form the Pitman-Yor process delivers a discrete distribution G. The Pitman-Yor process has three parameters: a base distribution G_0, which can be understood as a mean over the distribution of distributions. Additionally a discount parameter $0 \leq d < 1$ and a strength parameter $\theta > -d$ are used, which control the variability around the base distribution. The distribution G delivered by the Pitman-Yor process follows Zipf's power law property, i.e., the elements of G, i.e., the word probabilities, are inversely proportional the word's rank in the probability table.

An extension to the Pitman-Yor process is the Hierarchical Pitman-Yor process which is used for the HPYLM. For language modeling with a known number of words the base distribution G_0 is usually assumed to be a uniform distribution over all words (i.e., a zerogram). Using the Pitman-Yor process a unigram distribution is drawn via $G_\emptyset \sim \mathrm{PY}\left(d_0, \theta_0, G_0\right)$. Higher order n-gram distributions $G_{\mathbf{u}} \sim \mathrm{PY}\left(d_{|\mathbf{u}|}, \theta_{|\mathbf{u}|}, G_{\pi(\mathbf{u})}\right)$ can be generated, where $\mathbf{u} = [w_{i-n+1}, \ldots, w_{i-1}]$ is the context of length $|\mathbf{u}| = n-1$ and $\pi(\mathbf{u})$ is the shorter context with the first word removed.

The predictive probability using the HPYLM is given by [13]

$$\begin{aligned}\Pr(w|\mathbf{u}, S, \Theta) = &\frac{c_{\mathbf{u}w\cdot} - d_{|\mathbf{u}|} t_{\mathbf{u}w}}{\theta_{|\mathbf{u}|} + c_{\mathbf{u}\cdot\cdot}} \\ &+ \frac{\theta_{|\mathbf{u}|} + d_{|\mathbf{u}|} t_{\mathbf{u}\cdot}}{\theta_{|\mathbf{u}|} + c_{\mathbf{u}\cdot\cdot}} \Pr(w|\pi(\mathbf{u}), S, \Theta),\end{aligned} \tag{16}$$

where $S = \{c, t\}$ and $\Theta = \{d, \theta\}$ is the set of parameters for each context in the HPYLM and $G_{\mathbf{u}} := \Pr(w|\mathbf{u}, S, \Theta)$. This predictive probability can be interpreted by the notion of a Chinese restaurant. The counts $c_{\mathbf{u}wk}$ denote the counts for word w in context $\mathbf{u}$ at a so called table k and $t_{\mathbf{u}w}$ the number of tables for word w in context $\mathbf{u}$. The character '$\cdot$' is used as a wildcard in the sense that if a symbol in the subscript is replaced by '$\cdot$', the corresponding symbol can assume any value. For example, $c_{\mathbf{u}w\cdot} = \sum_k c_{\mathbf{u}wk}$.

Due to space limitations we will not describe the Chinese restaurant process in detail. We refer the reader to the literature, e.g., [13], [16], for details.

Gibbs sampling is used to obtain samples for S and Θ. The sampling of the hyper parameters Θ is described in [17]. To sample the parameters S either an existing table k is sampled for the word w whose count is to be increased in context $\mathbf{u}$ according to

$$\Pr(k|\mathbf{u}, S^{-w}, \Theta) \propto \frac{\max\left(0, c_{\mathbf{u}wk}^{-w} - d_{|\mathbf{u}|}\right)}{\theta_{|\mathbf{u}|} + c_{\mathbf{u}\cdot\cdot}^{-w}} \tag{17}$$

or a new table $k = k_{new}$ is sampled according to

$$\begin{aligned}&\Pr(k = k_{new}|\mathbf{u}, S^{-w}, \Theta) \propto \\ &\qquad \frac{\theta_{|\mathbf{u}|} + d_{|\mathbf{u}|} t_{\mathbf{u}\cdot}^{-w}}{\theta_{|\mathbf{u}|} + c_{\mathbf{u}\cdot\cdot}^{-w}} \Pr(w|\pi(\mathbf{u}), S^{-w}, \Theta)\end{aligned} \tag{18}$$

If $k = k_{new}$ is chosen the word counts are recursively increased in the shorter context $\pi(\mathbf{u})$. The table counts $t_{\mathbf{u}w}$ are increased accordingly. The superscript $^{-w}$ denotes the parameters S with the word count for w decreased in the according context. A word count can be decreased by drawing the table k form which the word count is to be decreased according to

$$\Pr(k|\mathbf{u}, S) \propto c_{\mathbf{u}wk} \tag{19}$$

If a table becomes empty the word count is recursively decreased from the shorter contexts. The table counts are adjusted accordingly.

Finally the NPYLM is an extension to the HPYLM which replaces the uniform base distribution G_0 over known words by a distribution over all possible character sequences that can be generated using a finite set of characters. This extension allows to use unknown words in the word level language model. When evaluating the probability of a word at the base distribution the word is expanded into its character sequence and the probability of the character sequence $w_i = (c_1, \ldots, c_k)$ is calculated by

$$\Pr(w_i) \approx \prod_{i=1}^{k} \Pr\left(c_i|c_{i-n+1}, \ldots, c_{i-1}\right). \tag{20}$$

The predictive probability $\Pr\left(c_i|c_{i-n+1}, \ldots, c_{i-1}\right)$ is again estimated using a character level HPYLM in exactly the same manner as for the word level. If a word count is to be increased or decreased at the base distribution of the word HPYLM, the word will be expanded into its character sequence and the counts for the according characters are increased or decreased in the character HPYLM. Finally as the base distribution for the character level HPYLM a uniform distribution over all characters is used.

For unsupervised word segmentation the Algorithm 1 is used together with the NPYLM. Several iterations are done over all sentences until the word segmentation converges. Before segmenting one sentence the word counts for the words of this sentence are decreased in the NPYLM, then the segmentation is done and the word counts for the newly found words are increased in the NPYLM and the word is added to the lexicon. This is also referred to as blocked Gibbs sampling as the parameters of a whole sentence is resampled in one step.

IV. Experimental Results

To evaluate the performance of the segmentation algorithm we performed experiments on the WSJCAM0 [18] acoustic model training data, which is a subset of sentences of the Wall Street Journal (WSJ0) [19] acoustic model training database.

To prepare the input data for the algorithm segmenting characters into words, we removed all word delimiters and punctuations from the text prompts of the WSJCAM0 database and transformed all characters to upper case.

To generate the input for the segmentation of phoneme sequences into words, the BEEP dictionary was employed to transcribe the words in terms of phoneme sequences [20]. For the translation from words to phoneme sequences only one unique phoneme sequence per word was used, i.e., pronunciation variants have not been considered. For the phoneme transcriptions we removed all silences and short pauses. We also removed duplicate sentences from the resulting word and phoneme transcriptions which left 5612 sentences with word transcriptions and 5621 sentences with phoneme transcription.

The phoneme transcriptions were used to evaluate the performance of the word segmentation algorithm because in a next step we want to use the output of a speech recognizer delivering phoneme sequences as input to the segmentation algorithm.

The transcriptions contain an average of 17 words per sentence amounting to a total of 95453 tokens for the words and 95629 tokens for the phoneme sequences (running words). There was an average of 5.3 characters and 4.5 phonemes per word. The number of unique words is 10660

Training data:
POWERFINANCIALISAFINANCIALSERVICESCONCERNTHATISSIXTYNINEPERCENTHELDBYPOWERCORPORATIONOFCANADAAMONTREALBASEDHOLDINGCOMPANY

$m = 4$:
PO WER F INANCIAL IS A FINAN CIAL SERVICES C ONCERN THAT IS SIXTY NINE PERCENT HELD BY PO WER CORP ORATION OF C ANADA A MON TREAL B ASED HOLDING COMPANY

$m = 6$:
POWER FINANCIAL IS A FINANCIAL SERVICES CONCERN THAT IS SIXTY NINE PERCENT HELD BY POWER CORPORATIONOF CANADA A MONTREAL BASED HOLDING COMPANY

$m = 8$:
POWER FINANCIAL IS A FINANCIAL SERVICES CONCERN THAT IS SIXTY NINE PERCENT HELDBY POWER CORPORATION OF CANADA A MONTREALBASED HOLDINGCOMPANY

Fig. 1. Input data and segmentation result with $m \in \{4, 6, 8\}$

and the number of unique phoneme sequences is 10506 in the transcriptions, which we call lexicon.

We let the word segmentation algorithm run for 100 iterations on the word and phoneme sentences and examined the output of the last iteration for the segmentation result. The order n was set to 2 for the word HPYLM, and The order m of the character HPYLM was varied from 4 to 8 to evaluate the influence of the order of the character level language model to the segmentation result.

Fig. 1 shows an example for the segmentation results of one training sentence at the orders $m \in \{4, 6, 8\}$. It can be seen that with $m = 4$ the text is segmented into smaller fragments which don't correspond to meaningful words while with increasing order of the character language model the quality of the segmentation result increases. It can also be seen that with $m \in \{6, 8\}$ the segment boundaries are mostly found at the correct positions resulting in meaningful words being discovered, except of some word boundaries which are skipped. The missing boundaries result in concatenations of word which frequently occur in pairs. Further processing or more data might be needed to split these concatenated words into single words as well.

Tables I and II show the evaluation of the segmentation results on the word level in terms of the token precision (P), token recall (R), token f-score (F), the number of tokens (running words), lexicon precision (LP), lexicon recall (LR), lexicon f-score (LF) and the number of unique words in the lexicon, while the former (token) refers to the total number of running words and the second (lexicon) to the number of unique words. Tables III and IV shows the corresponding evaluation with the phoneme sequences at the input of the segmentation algorithm.

P, R and F are defined as follows.

$$P = \frac{N_c}{N_f},\ R = \frac{N_c}{N} \tag{21}$$

$$F = 2\frac{P \cdot R}{P + R}. \tag{22}$$

Here N is the total number of tokens, i.e., running words in the ground truth transcription, N_f is the overall number of discovered tokens/words and N_c the number of correctly discovered tokens/words. A token is defined by the mapping of the character sequence between two discovered boundaries to a word. A correctly discovered token is a token whose character string and therefore whose boundaries match with the ones in the ground truth transcriptions.

TABLE I

Word token precision, recall and f-score from character input as a function of character language model order $m \in [4, 8]$

m	4	5	6	7	8
P	42.9	59.8	67.7	69.2	71.1
R	48.7	52.5	58.2	58.4	60.5
F	45.6	55.9	62.6	63.4	65.4
Words	108383	84024	81983	80576	81309

TABLE II

Word lexicon precision, recall and f-score from character input as a function of character language model order $m \in [4, 8]$

m	4	5	6	7	8
LP	32.9	36.5	42.7	45.6	48.5
LR	46.6	59.6	62.7	63.8	64.2
LF	38.5	45.3	50.8	53.2	55.2
Words	15096	17405	15662	14901	14104

For the LF, LP and LF scores, N denotes the number of unique words present in the ground truth lexicon, N_f the number of unique words discovered during word segmentation and N_c the number of correctly discovered words. A correctly discovered word is a word that was discovered in the segmentation process and also is contained in the ground truth lexicon.

From the tables it can be seen that all scores increase with increasing order of the character language model and that the number of found words is in the same order as the actual number of words in the data.

There is only a small difference between the results on the phoneme and word level. The number of discovered words also decreases slightly with increasing language model order, except of a peak at $m = 5$ which we cannot explain yet.

This evaluation demonstrates the potential of the segmentation algorithm and the importance of considering the temporal context during the segmentation process.

TABLE III

Word token precision, recall and f-score from phoneme sequence input as a function of phoneme language model order $m \in [4, 8]$

m	4	5	6	7	8
P	54.2	67.6	68.0	72.4	73.3
R	49.9	51.2	52.1	56.8	57.1
F	52.0	58.3	59.0	63.7	64.2
Words	88070	72464	73294	74979	74471

TABLE IV

WORD LEXICON PRECISION, RECALL AND F-SCORE FROM PHONEME SEQUENCE INPUT AS A FUNCTION OF PHONEME LANGUAGE MODEL ORDER $m \in [4, 8]$

m	4	5	6	7	8
LP	33.0	37.1	38.7	43.7	44.8
LR	56.0	65.2	64.0	65.6	66.1
LF	41.5	47.3	48.3	52.5	53.4
Words	17839	18466	17359	15775	15505

V. CONCLUSIONS AND OUTLOOK

In this contribution we have applied a word segmentation algorithm to the unsupervised word discovery from phonetic or character input. The segmentation exploits the fact that certain phoneme (or character) sequences are recurrent thus make up words. Word and character sequence probabilities are learnt alongside the segmentation. Since the segmentation changes in the course of the iterations, words and word probabilities need to be ajusted constantly. This is achieved by using a Nested Pitmal-Yor language model which is able to cope with an unknown and possibly unlimited number of words. By backing off to a character language models unknown words can be accounted for.

We have presented results on the WSJCAM0 large vocabulary database and shown how the quality improves with the order of the character/phoneme language model employed. So far we only used error free phoneme transcriptions and transcriptions without any variations in pronunciation.

As we want to learn a language directly from acoustic data we have to deal with further challenges. The first challenge is that real acoustic recordings, possibly form multiple speakers, will most likely contain variations of pronunciation for a word. In [21] an algorithm was proposed which, given a segmented character string, basically clusters similar words together, exploiting contextual information of the word provided by a language model. Using contextual information helps to cluster similar words as similar words will also most likely appear in similar context.

The second challenge is that we will have to deal with noisy data, especially with insertions, deletions and substitutions in the phoneme transcriptions provided by a phoneme recognizer. In [1] an extension to the word segmentation algorithm of [12] was proposed to do the word segmentation not only over the best recognition result (transcription) of a phoneme recognizer but over a lattice of possible phoneme sequences. Using a lattice will provide the advantages that a) more variants of a recognition result can be considered during the segmentation process and b) that individual variants can be weighted by their likelihood, provided by the recognizer. Given the lattice, the likelihoods and the language model, the algorithm can choose the best phoneme sequence with respect to all other data already seen.

REFERENCES

[1] G. Neubig, M. Mimura, and T. Kawahara, "Bayesian learning of a language model from continuous speech," *IEICE TRANSACTIONS on Information and Systems*, vol. 95, no. 2, pp. 614–625, 2012.

[2] J. Schmalenstroeer, M. Bartek, and R. Haeb-Umbach, "Unsupervised learning of acoustic events using dynamic time warping and hierarchical k-means++ clustering," in *Twelfth Annual Conference of the International Speech Communication Association*, 2011.

[3] S. Chaudhuri and B. Raj, "Unsupervised structure discovery for semantic analysis of audio," in *Advances in Neural Information Processing Systems 25*, 2012, pp. 1187–1195.

[4] T. Taniguchi and S. Nagasaka, "Double articulation analyzer for unsegmented human motion using pitman-yor language model and infinite hidden markov model," in *System Integration (SII), 2011 IEEE/SICE International Symposium on*. IEEE, 2011, pp. 250–255.

[5] A. S. Park and J. R. Glass, "Unsupervised pattern discovery in speech," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 16, no. 1, pp. 186–197, 2008.

[6] R. Flamary, X. Anguera, and N. Oliver, "Spoken wordcloud: Clustering recurrent patterns in speech," in *Content-Based Multimedia Indexing (CBMI), 2011 9th International Workshop on*. IEEE, 2011, pp. 133–138.

[7] O. Walter, J. Schmalenstroeer, and R. Haeb-Umbach, "A novel initialization method for unsupervised learning of acoustic patterns in speech," University of Paderborn Department of Communications Engineering, FGNT Technical Report FGNT-2013-01, Mar. 2013.

[8] D. D. Lee, H. Seung, *et al.*, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, 1999.

[9] M. Van Segbroeck *et al.*, "Unsupervised learning of time–frequency patches as a noise-robust representation of speech," *Speech Communication*, vol. 51, no. 11, pp. 1124–1138, 2009.

[10] P. D. O'Grady and B. A. Pearlmutter, "Discovering speech phones using convolutive non-negative matrix factorisation with a sparseness constraint," *Neurocomputing*, vol. 72, no. 1, pp. 88–101, 2008.

[11] M. Nakano, J. Le Roux, H. Kameoka, Y. Kitano, N. Ono, and S. Sagayama, "Nonnegative matrix factorization with markov-chained bases for modeling time-varying patterns in music spectrograms," *Latent Variable Analysis and Signal Separation*, pp. 149–156, 2010.

[12] D. Mochihashi, T. Yamada, and N. Ueda, "Bayesian unsupervised word segmentation with nested pitman-yor language modeling," in *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*. Association for Computational Linguistics, 2009, pp. 100–108.

[13] Y. W. Teh, "A hierarchical bayesian language model based on pitman-yor processes," in *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2006, pp. 985–992.

[14] C. D. Manning and H. Schütze, *Foundations of statistical natural language processing*. MIT press, 1999.

[15] J. Pitman and M. Yor, "The two-parameter poisson-dirichlet distribution derived from a stable subordinator," *The Annals of Probability*, vol. 25, no. 2, pp. 855–900, 1997.

[16] B. Frigyik, A. Kapila, and M. Gupta, "Introduction to the Dirichlet distribution and related processes," University of Washington Electrical Engineering Department, UWEE Technical Report UWEETR-2010-0006, Dec. 2010.

[17] Y. W. Teh, "A bayesian interpretation of interpolated kneser-ney," 2006.

[18] J. Fransen, D. Pye, T. Robinson, P. Woodland, and S. Young, *WSJ-CAMO corpus and recording description*. Citeseer, 1994.

[19] D. B. Paul and J. M. Baker, "The design for the wall street journal-based csr corpus," in *Proceedings of the workshop on Speech and Natural Language*. Association for Computational Linguistics, 1992, pp. 357–362.

[20] Beep dictionary. [Online]. Available: http://svr-www.eng.cam.ac.uk/comp.speech/Section1/Lexical/beep.html

[21] S. Goldwater, J. Eisenstein, and M. Elsner, "Bootstrapping a unified model of lexical and phonetic acquisition," 2012.