A K-NEAREST NEIGHBOURS APPROACH TO UNSUPERVISED SPOKEN TERM DISCOVERY

Alexis Thual, Corentin Dancette, Julien Karadayi, Juan Benjumea, Emmanuel Dupoux

EHESS, ENS, PSL University, CNRS, INRIA

ABSTRACT

Unsupervised spoken term discovery is the task of finding recurrent acoustic patterns in speech without any annotations. Current approaches consists of two steps: (1) discovering similar patterns in speech, and (2) partitioning those pairs of acoustic tokens using graph clustering methods. We propose a new approach for the first step. Previous systems used various approximation algorithms to make the search tractable on large amounts of data. Our approach is based on an optimized k-nearest neighbours (KNN) search coupled with a fixed word embedding algorithm. The results show that the KNN algorithm is robust across languages, consistently outperforms the DTW-based baseline, and is competitive with current state-of-the-art spoken term discovery systems.

Index Terms— spoken term discovery, word discovery, unsupervised, word segmentation

1. INTRODUCTION

Current automatic speech recognition pipelines require considerable textual resources to train acoustic and language models. Several approaches have been pursued to address the challenge of building useful technology for languages where few textual resources are available. The "zero resource" setting [1] tackles the most extreme case, where no textual resource is available at all, and aims at extracting useful linguistic units from raw speech in an unsupervised way. Spoken term discovery focuses on discovering word-sized units; it can be used for document classification [2], segmentation [3] or retrieval [4, 5], and idiolect recovery [6]. It can also help other zero resource objectives, such as learning phonetic unit or representations [7, 8, 9].

Searching for all matching motifs in a speech corpus scales quadratically and can be impractical for large corpora. Here, we separate spoken term discovery into two subproblems: representing candidate motifs as a fixed-sized vector, and similarity-based search. For the latter, we use a high performance k-nearest neighbours library developed by Facebook [10], which can scale to billion of search terms. We test our systems on datasets from the Zero Ressource Challenges [11, 12], and compare it to the state-of-the-art.

2. RELATED WORK

Spoken term discovery systems can be sorted in two classes. The first class attempts to discover recurring motifs in the speech signal by using a DTW-based distance metric over motifs, and clustering these motifs to form a lexicon [13, 14, 15, 16, 17, 18]. The focus of these algorithms is not to discover all of the words, but to discover a subset of good candidates. This is why they typically set a lower bound on the duration of the motifs, in order to avoid short words, which tend to have many neighbors and be hypo-articulated. The second class attempts to exhaustively parse the input signal into words, thereby leveraging the additional constraints that word candidates cannot occupy the same portion of speech. These algorithms both segment the speech and construct a lexicon at the same time [19, 20]. They are variants of text-based algorithms based on the idea of minimizing the length (or maximizing the probability) of the corpus and its accompanying lexicon [21, 22, 23].

In the present work, we aim at adding a new element to the first class, but simplify the pipeline by getting rid of the DTW step, which requires to compute a very large matrix of frame-wise similarity of the corpus by itself. The system proposed by Jansen and van Durme [16], which was used as baseline of the Zerospeech Challenge 2015 [11], proposes to avoid computing the whole matrix by binarizing it using random projections, an indexation technique yielding a large gain in speed, but a decrement in performance due to the approximations performed by the algorithm [16]. Here, we propose to use another indexation tool, based on product quantization [24], and specifically the FAISS library which can perform efficient exact or approximate search on CPUs or GPUs [10].

3. DESCRIPTION OF THE PIPELINE

Our pipeline is presented in Figure 1. After extracting speech features, we segment the input speech and construct a fixed-length representation of each segmented term. We then use a KNN algorithm to systematically find a set of matching pairs, which we narrow down by a variant of Non Maximal Suppression to avoid overlap. This is followed by an optional step of graph clustering, before the output is evaluated.

To appear in 2018 IEEE Spoken Language Technology (SLT)

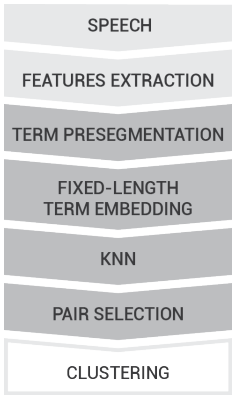


Fig. 1. Steps in our KNN pipeline

3.1. Features extraction

Because we compare our system with Jansen van Durne's DTW system, we use the same input features, Perceptual Linear Prediction (PLP) features [25]. PLP features are designed to compress the input signal in a way that is inspired by human hearing. Thirteen PLP features are computed every 10 ms based on an analysis window of 25ms, and concatenated with first and second order temporal derivatives.

3.2. Term pre-segmentation

Our pipeline extracts every possible segmentation of the input speech into motifs and constructs a fixed length embedding for it. We restrict these motifs to span over the parts of the input data that only contain speech, using the voice activity detection (VAD) temporal alignment in the dataset. In order to avoid unnecessary computations with redundant motifs that only differ in a couple of frames, we only generate a possible segmentation point every a frame, with $a > 1$. Experiments showed that $a = 4$ is a good compromise and corresponds approximately to 2 segmentation points per phoneme.

When it comes to discovering potential words, too short motifs are not useful because difficult to cluster, and too long motifs tend to occur only once in a corpus. Therefore, we only consider terms for which the length is between l_{min} and l_{max}. This yields a total number of presegmented motifs T:

$$T = \frac{1}{a}\left(l_{max} - l_{min}\right) \times \frac{N}{a}$$

Where N is the total number of speech frames in the corpus. For instance, for length bounds between 20 and 50 frames with a step of 4 frames, the number of terms is only 1.8 times the number of frames of the original speech file. In this paper, we test several length bounds.

3.3. Fixed-length term embedding

Each term is encoded as a fixed-length vector in order to enable KNN search. Previous work has devised various frameworks for deriving such embeddings [26].

Here, we use a simple Gaussian smoothed down-sampling method, typically used in [20] and studied in [26]. The idea is to represent each segmented term using only d frames: in order to do so, one averages adjacent frames using a gaussian kernel. Let f_i be the i-th frame of the output, then

$$f_i = \mathcal{N}\left(\frac{i \cdot n_f}{d}; r \cdot n_f + s \cdot g_d(i)\right) \cdot t$$

where $g_d : i \mapsto \frac{d}{2} - |\frac{d}{2} - i|$, t is the feature representation of the term being down-sampled, n_f is the number of frames of t, r and s two parameters which we here set to 0.07 and 0.1 respectively, as recommended in [26]. We set the value of d to 20 for all experiments reported in this paper.
We had tested several values of d (from 10 to 30), r (from 0 to 1) and s (from 0 to 1), but none of these changes had strong positive or negative impact on our results.

3.4. KNN search

The pre-segmented embedded terms are stored in a large index. Then, for each term, we search the entire index for it's k nearest neighbours using FAISS [10], where the similarity is based on the dot product of the embedded terms. In our pipeline, we use an exact search method, parallelized on 10 CPUs, with k set to 200. Searching the index against itself takes 98% of the overall run-time.

We tested several values for k, ranging from 50 to 500. The idea is using a reasonably large value for k (instead of $k = 1$) is to take into account self-overlapping pairs which may also tend to be very self-similar. In addition, if a good match is found between two different parts of the corpus, several overlapping matches will also have a good similarity match. Therefore our strategy was to allow a large number of matches to be returned, and prune this number down in a selection step. Experiments showed that all values above 100 yield the same results for a dataset of the size of a few hours. We set the value of k to 200 in order to handle larger datasets while keeping the run-time reasonable.

3.5. Selection of top pairs

Generating $T \times k$ pairs, where T is the number of previously segmented terms, obviously yields too many pairs, most of which are of poor quality. In addition, as we said above, many pairs are redundant and partially overlapping. In order to reduce the number of pairs, we perform the following 4 steps for each of the input speech files:

1. Sorting: we recover the totality of the KNNs for each segmented term an input file, and sort the resulting pairs of terms by decreasing similarity.

2. Thresholding: we keep only the top δ % of the pairs for a given file; in this paper, δ is the main control parameter and manipulated systematically (typically between 0.5% and 3%).

3. Self-overlapping pairs removal: given a pair $p = (t_1, t_2)$, we discard p if t_1 and t_2 overlap more than a given threshold τ_{self} where τ_{self} equals 50% in our experiments. This selection phase is very efficient and discards a large proportion of the pairs (around 90%).

4. Non Maximal Suppression (NMS): to remove the remaining overlapping pairs, we perform a variant of NMS by sequentially inspecting the pairs starting from the most similar ones. We only keep a new pair if it is not overlapping with an already selected pair. We use a threshold τ_{other} of 50%. NMS helps discarding around 50% of the remaining pairs. Practically, we distribute the selected pairs into slots, one for each by pair of utterances in the original corpus (defined as contiguous stretches of speech) and perform NMS within each slot only. This allows for faster computations in our implementation.

In this paper, we define temporal overlap between two pairs of terms $p = (t_1, t_2)$ and $p' = (t'_1, t'_2)$ as follows:

$$o(p, p') = \frac{l(t_1 \cap t'_1) \cdot l(t_2 \cap t'_2)}{l(t_1) \cdot l(t_2)}$$

where l yields the length of a given term and $t \cap t'$ designates the overlapping part of terms t and t'. It is important to notice that $o(p, p')$ is not necessarily equal to $o(p', p)$ and that the order in which pairs are seen during the NMS actually has an impact. Self-overlapping of a given pair $p = (t_1, t_2)$, with $t_1 = [a_1, b_1]$ and $t_2 = [a_2, b_2]$, is computed using the following formula:

$$so(p) = \frac{\min(b_1, b_2) - \max(a_1, a_2)}{\min\big(l(t_1), l(t_2)\big)}$$

3.6. Clustering

Once pairs of word-units have been selected, one can perform a clustering step in order to group those pairs together. The goal is that each cluster would account for one word-unit of the discovered lexicon, and that each member of a given cluster represents one occurrence of the aforementioned word-unit. Several clustering methods were documented [17]. We did not try to use them here and rather focused on the quality of the output pairs of our pipeline. This can be interpreted as if each pair was a cluster on it's own.

4. EXPERIMENT I

The aim of this Experiment is to test systematically the effect of two important parameters in the KNN pipeline described above: (l_{min}, l_{max}) the range of durations selected in the presegmentation step, and δ, the percentage of pairs that are kept after the KNN search. The range of duration is important because too long terms can lead to useless computations, while too short terms could give rise to very noisy results. To choose the range of durations to study, we had a peek of the distribution of word durations across the languages of the Zero Resource Challenges, and chose the following partition of durations: 100-200ms, 200-300ms, 300-500ms, 500-1000ms, each of which approximately accounts for 25% of the word tokens across these languages. We also used 200-500ms, which covers the middle range, and about 50% of the word tokens. The δ threshold commands a tradeoff between the number of pairs and the quality of these pairs, and we vary this threshold systematically to explore the whole range of this tradeoff.

This is conducted in the Mandarin dataset of the Zero Ressource Speech Challenge 2017 [12], which we considered to be our development dataset to setup and test our KNN pipeline. Previously mentioned parameters remained fixed to the values described in the pipeline description.

4.1. Dataset

The dataset, available in the Zero Resource Speech Challenge 2017, includes 12 Mandarin speakers, each of whom speaks for a duration from 10 to 25 minutes. The overall duration of this dataset is 2 hours and 30 minutes. It has been forced aligned to enable phoneme-based evaluation metrics.

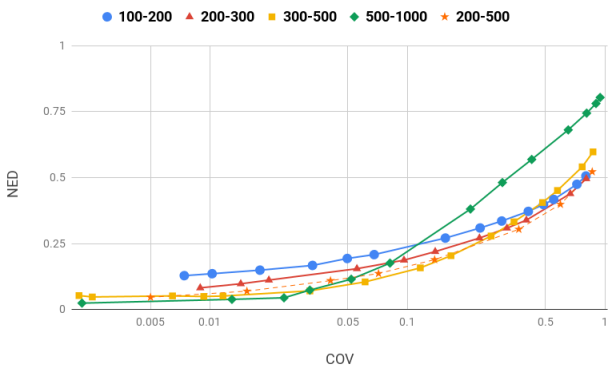
4.2. Evaluation

To study the effect of duration and δ, we only focus on the subset of the Zero Resource Speech Challenge 2015 metric devoted to measuring the quality of the similarity-based search, namely:

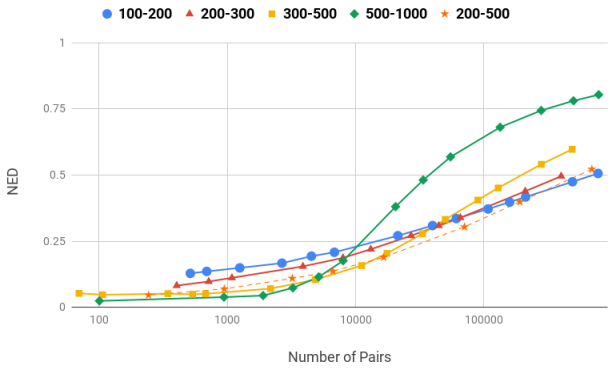
- Normalized Levenshtein Distance (NED): given a pair $p = (t_1, t_2)$, one can read the phonetic transcriptions of t_1 and t_2 using the *gold* temporal alignments; the normalized Levenshtein distance between these two transcriptions yields the NED for this pair
- Coverage (COV): it indicates the proportion of input speech which was discovered; for instance, a coverage of 0.5 indicates that half the input phones were involved in computed pairs of terms
- Number of pairs: simply the number of pairs of terms yielded by the pipeline

4.3. Results and discussion

Results are reported in Figure 2. They confirm the expected effects of duration range on the tradeoff between number and quality of the pairs. Articulation for short words is on average worse than it is for longer terms. This is indeed what we



(a) NED / COV for Mandarin using PLPs with various term lengths



(b) NED / Number of Pairs for Mandarin using PLPs with various term lengths

Fig. 2. Results comparison using various lengths for segmented terms on mandarin using PLPs (labels account for length' range in ms)

observe here: for a small COV or number of pairs, longer terms yield much better results that short ones. However, for larger COV and number of pairs, short terms yield a much better NED than long ones. This is because the frequency of a long term is usually low, thus, when a high number of pairs is required (by setting δ to a permissive value for instance), our structure pairs long words with other long words and the chances of these being different occurrences of the same word get lower. On the contrary, small words' frequency is much higher (words lasting between 100ms and 200ms are almost syllables) and it is still plausible to find theoretically good pairs even when δ is very permissive.

As expected, results for terms of length ranging between 200ms and 500ms approximately behave like terms of length 300-500ms for low COV and number of pairs, and roughly follow terms of length 200-300ms for high COV and number of pairs.

Wider ranges allows for better results but only up to a point. A range between 200ms to 1000ms, for instance, (not shown in the Figure) yields results which are almost comparable to 200-500ms for all COV values, with much longer computation time. This is why in the following, we stick to the 200-500ms range.

5. EXPERIMENT II

The aim of this experiment is to compare our KNN pipeline to the Zero Ressource 2015 DTW-based baseline. To do this, we picked our best system as evaluated in the Mandarin dataset in Experiment 1 and applied it without changing any parameter to the two datasets of the ZR15 Challenge: English and Xitsonga.

We first run the same evaluation as in Experiment 1 on the duration range of 200-500ms, systematically varying the threshold parameters of both algorithms in order to compare their quality/quantity tradeoffs. Next, we pick a value for the threshold parameter δ of the KNN system so that the NED is comparable to that of the DTW-baseline and points with higher NED. We then evaluate the results using the standard metrics of the challenge and compare them to state-of-the-art implementations.

5.1. Datasets

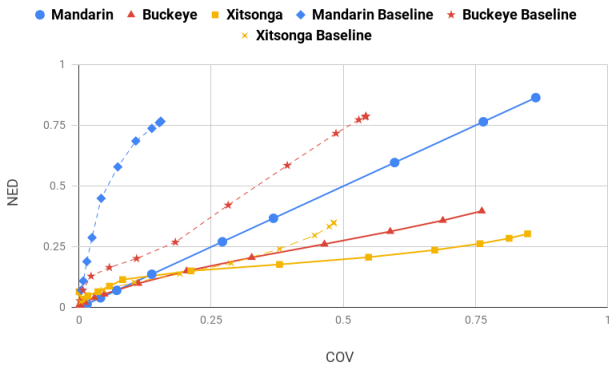
The English dataset comes from the Buckeye corpus and includes 12 speakers speaking between 16 and 30 minutes each, for a total duration of 5 hours. The Xitsonga corpus includes 24 speakers speaking between 2 and 30 minutes each, for a total duration of 2 hours and 30 minutes.

5.2. Evaluation

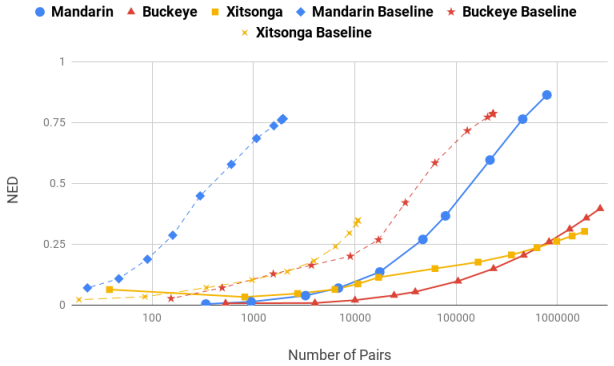
We use the full set of evaluation metrics of the second Track of the Zero Resource Challenge 2017. This includes Ned and Cov (already described above), and a set of 4 precision/recall/F-score metrics each of them dedicated to a particular aspect of spoken term discovery: Grouping, Type, Token and Boundary. The Grouping precision and recall metrics are similar to cluster purity and collocation scores, respectively, except they are computed on a pair by pair basis and do not require a majority decision (each pair is labelled as belonging to the 'same cluster' or 'different cluster' set). The Type metrics compare the discovered clusters to the gold lexicon (also pair-wise). The token metrics compare the set of discovered terms with the set of gold tokens, and the boundary metrics compare the discovered and gold boundaries. The Fscores are the harmonic means of precision and recall.

5.3. Results and discussion

Figure 3 shows the NED / COV and NED / Number of pairs graphics for the DTW baseline system and the KNN system on the two test languages of the Zero Ressource Challenge



(a) NED / COV comparison to baseline for Buckeye, Mandarin and Xitsonga using PLPs



(b) NED / Number of Pairs comparison to baseline for Buckeye, Mandarin and Xitsonga using PLPs

Fig. 3. Results comparison using various lengths for segmented terms on mandarin using PLPs (labels account for length' range in frames)

2015. We also show the results on Mandarin for comparison purposes. The results show that for every language, the KNN system yields better/more numerous pairs than the DTW system. In English, for a NED of 25% for instance, the KNN system finds between 2 and 2.5 more pairs in English and Xitsonga in the COV metrics. The difference is ever larger in the number of pairs metrics (between 20 and 100). This is all the more remarkable that the KNN system was not fine tuned for these two languages, whereas the DTW system was. For Mandarin, the difference between the two systems even larger, mainly due to very poor performance of the DTW system in this language, suggesting an over-fitting of the parameters for the languages of the Zero Resource Challenge 2015.

The interesting discrepancy between COV and number of pairs across the two algorithms suggests that the KNN discover many more pairs per region of speech. This could be because it reports more overlapping pairs (redundant pairs), or because it manages to find more of the pairs related to the same word (less fragmented clusters). Regarding the performance of the KNN algorithm across the three languages, we see similar performances up to 20% coverage, and a divergence thereafter, mandarin giving worst results, followed by English and Xitsonga. Despite these differences, the performance is much more stable than for the DTW algorithm.

Finally, we selected the value of the δ threshold such that the NED value for the KNN algorithm would cover the range of NED found in the the DTW baseline system. The values tested were 15%, 20% and 30%. We then analyzed the results with the suite of metrics of the Challenge as shown in Table 1.

We found that for all three languages, the KNN algorithm was able to beat the DTW baseline on the grouping precision (corresponding to cluster purity), but not on recall (which corresponds to the inverse of cluster fragmentation). This is not surprising because we did not apply a clustering algorithm and only took the raw pairs from the output of the KNN pipeline. Interestingly, the segmentation and lexical F-scores were better for the KNN algorithm, which is probably due to its larger coverage.

The KNN algorithm was competitive with the state-of-the-art DTW system [18], which performs exact DTW using a highly optimized DTW search over a GPU. The GPU-DTW algorithm presented was set on a different NED/COV tradeoff than our KNN algorithm, resulting in better grouping and boundary performance. Interestingly, our KNN still beast the DTW on type and token F-scores.

The final comparison is with exhaustive algorithms. We compare our KNN algorithm with the Bayesian segmental algorithm of [20] and the Kmeans version of the algorithm [27]. Unsurprisingly, these exhaustive algorithms outperform KNN on all segmentation and lexical metrics (except for the Type and Token F-Scores in Mandarin). However, this high coverage and good segmentation comes at a cost in clustering purity (very high NED and poor grouping precision). In light of these tradeoffs, further work would be needed to compare these different algorithms back to back.

6. GENERAL DISCUSSION

We present a new pipeline for spoken term discovery which bypasses the need of performing DTW on the speech data, and rather uses a performant KNN library on fixed-words embeddings. The results, optimized on Mandarin, transfer to two new languages, and beat a DTW-based baseline that had been optimized for these two languages.

The datasets that were used here were small enough that we could used the exact search mode of the KNN library. Using 10 CPUs, our searches run approximatively in real ime (2 to 3 hours for Mandarin - 5 hours for the Buckeye) and necessitate the storage of an index of 1 to 2 GB depending on the dataset's size. For larger datasets, we would have to resort to the approximate mode and/or the parallelized GPU mode

					Grouping			Type			Token			Boundary		
	Clusters	Pairs	NED	Cov	Pr	Re	F	Pr	Re	F	Pr	Re	F	Pr	Re	F
Mandarin																
Topline	1240	1 742 931	0.0	100	100	100	100	29.3	29.3	29.3	28.1	46.1	34.9	66.2	100	79.7
Baseline	156	160	30.7	2.9	30.2	**96.7**	**44.7**	4.5	0.1	0.2	**4.0**	0.1	0.1	**37.5**	0.9	1.8
ES-KM			88.1	**100**	/	/	/	2.5	4.1	3.1	2.5	3.4	2.9	36.	**47.1**	**41.1**
KNN-20	25 488	25 488	**20.9**	17.8	**64.8**	17.6	27.7	**6.5**	3.0	4.1	2.7	3.5	3.0	16.1	13.1	14.4
KNN-30	78 532	78 532	29.8	36.8	50.9	12.5	20.1	6.4	**6.4**	**6.4**	2.8	**8.6**	**4.2**	16.3	28.8	20.9
English - Buckeye																
Topline	/	/	0.0	100	99.5	100	99.7	50.3	56.2	53.1	68.2	60.8	64.3	88.4	86.7	87.5
Baseline	3149	4305	21.9	16.3	21.4	84.6	33.3	6.2	1.9	2.9	5.5	0.4	0.8	44.1	4.7	8.6
O			39.4	92.1	**76.2**	**100**	**82.7**	5.6	5.1	5.3	10.2	1.9	3.2	71.1	22.5	34.3
BES-GMM			56.0	**100**	22.7	29.6	25.5	**14.0**	**28.6**	18.8	**26.6**	2.5	**17.0**	**80.7**	**50.4**	62.0
ES-KM			71.6	**100**	/	/	/	/	/	**18.9**	/	/	18.1	/	/	**62.2**
KNN-20	464 491	464 491	**20.7**	32.7	41.5	15.0	22.1	5.5	17.4	8.4	2.6	5.9	3.6	25.4	22.3	23.7
KNN-30	1 319 411	1 319 411	31.4	58.9	27.0	13.6	18.1	4.1	29.7	7.3	2.2	**10.2**	3.6	25.4	40.0	31.1
Xitsonga																
Topline	/	/	0.0	100	100	100	100	15.1	18.1	16.5	34.1	49.7	40.4	66.6	91.9	77.2
Baseline	1782	1818	12.0	16.2	52.1	77.4	62.2	3.2	1.4	2.0	2.6	0.5	0.8	22.3	5.6	8.9
O			39.6	95.5	19.1	**100**	**31.7**	1.6	2.2	1.9	1.5	0.5	0.8	**49.9**	27.6	35.5
BES-GMM			58.6	**100**	8.3	10.3	9.2	3.8	9.8	5.5	**4.3**	4.0	**4.1**	44.5	42.0	**43.2**
ES-KM			80.3	**100**	/	/	/	/	/	4.9	/	/	3.7	/	/	42.1
KNN-15	61 818	61 818	**15.2**	21.2	**66.3**	5.9	10.8	3.6	3.8	3.7	1.2	4.1	1.9	13.0	16.7	14.6
KNN-20	351 899	351 899	20.8	54.8	53.0	2.9	5.5	3.3	8.3	4.7	1.3	16.8	2.4	13.0	47.6	20.4
KNN-30	1 852 520	1 852 520	30.4	84.8	36.9	1.7	3.3	4.1	**17.2**	**6.7**	1.2	**42.5**	2.4	12.8	**80.5**	22.1

Table 1. Zero Resource Track 2 evaluation results for the Topline (gold phonemes plus Adaptor Grammar), the DTW Baseline system, O: parallelized DTW system [18], ES-KM: Embedded Segmental Kmeans [27], BES-GMM: BayesSegMinDur's model of [20], and KNN (our system) with 15%, 20%, or 30% of NED.

of the KNN library. What's important, though, is that because we have separated the problem of representing the speech motifs from the problem of search, the scaling up problems can be addressed by generic systems without having to make particular tweaks about speech.

The pre-segmentation and representation steps are of course critical for the quality and efficiency of the algorithm. Further work would need to explore other pre-segmentation ideas, in particular the idea of using syllabic boundaries as in Rasanen et al.[28]. If this turns out to be reliable, we could expect a speed up factor of about 10-20. As for the representation, down-sampling is not the most efficient way to represent spoken terms. Other work has shown that recurrent NNs can achieve even better results, on some occasions overtaking DTW methods [26] and would need to be tested in this framework.

7. ACKNOWLEDGMENTS

This work was funded by the European Research Council (ERC-2011-AdG-295810 BOOTPHON), the Agence Nationale pour la Recherche (ANR-10-LABX-0087 IEC, ANR-10-IDEX-0001-02 PSL*), Almerys (industrial chair Data Science and Security), Microsoft Research (joint MSR-INRIA center), Amazon Web Service (AWS Research Credits) and a Google Faculty Award.

8. REFERENCES

[1] James Glass, "Towards unsupervised speech processing," in *Information Science, Signal Processing and their Applications (ISSPA), 2012 11th International Conference on*. IEEE, 2012, pp. 1–4.

[2] Mark Dredze, Aren Jansen, Glen Coppersmith, and Ken Church, "Nlp on spoken documents without asr," in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2010, pp. 460–470.

[3] Lilei Zheng, Cheung-Chi Leung, Lei Xie, Bin Ma, and Haizhou Li, "Acoustic texttiling for story segmentation of spoken documents.," in *ICASSP*, 2012, pp. 5121–5124.

[4] Jerome White, Douglas Oard, Aren Jansen, Jiaul Paik, and Rashmi Sankepally, "Using zero-resource spoken term discovery for ranked retrieval," in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2015, pp. 588–597.

[5] Lin-shan Lee, James Glass, Hung-yi Lee, and Chun-an Chan, "Spoken content retrievalbeyond cascading speech recognition with text retrieval," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 9, pp. 1389–1420, 2015.

[6] Aren Jansen, Daniel Garcia-Romero, Pascal Clark, and

Jaime Hernandez-Cordero, "Unsupervised idiolect discovery for speaker recognition.," .

[7] Aren Jansen and Kenneth Church, "Towards unsupervised training of speaker independent acoustic models," in *Twelfth Annual Conference of the International Speech Communication Association*, 2011.

[8] Herman Kamper, Micha Elsner, Aren Jansen, and Sharon Goldwater, "Unsupervised neural network based feature extraction using weak top-down constraints," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. 2015, pp. 5818–5822, IEEE.

[9] Gabriel Synnaeve, Thomas Schatz, and Emmanuel Dupoux, "Phonetics embedding learning with side information," in *2014 IEEE Workshop on Spoken Language Technology, SLT 2014 - Proceedings*, 12 2014.

[10] Jeff Johnson, Matthijs Douze, and Hervé Jégou, "Billion-scale similarity search with gpus," *arXiv preprint arXiv:1702.08734*, 2017.

[11] Maarten Versteegh, Roland Thiolliere, Thomas Schatz, Xuan Nga Cao, Xavier Anguera, Aren Jansen, and Emmanuel Dupoux, "The zero resource speech challenge 2015," in *Proc. of Interspeech*, 2015.

[12] Ewan Dunbar, Xuan Nga Cao, Juan Benjumea, Julien Karadayi, Mathieu Bernard, Laurent Besacier, Xavier Anguera, and Emmanuel Dupoux, "The zero resource speech challenge 2017," in *Automatic Speech Recognition and Understanding Workshop (ASRU), 2017 IEEE*. 2017, pp. 323–330, IEEE.

[13] Alex S. Park and James R. Glass, "Unsupervised Pattern Discovery in Speech," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 1, pp. 186–197, Jan. 2008.

[14] Armando Muscariello, Guillaume Gravier, and Frdric Bimbot, "Audio keyword extraction by unsupervised word discovery," in *INTERSPEECH 2009: 10th Annual Conference of the International Speech Communication Association*, 2009.

[15] Armando Muscariello, Guillaume Gravier, and Frdric Bimbot, "Unsupervised Motif Acquisition in Speech via Seeded Discovery and Template Matching Combination," in *IEEE TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING*, 2012, vol. 20,7, pp. 2031–2044.

[16] Aren Jansen and Benjamin Van Durme, "Efficient spoken term discovery using randomized algorithms," *2011 IEEE Workshop on Automatic Speech Recognition & Understanding*, pp. 401–406, 2011.

[17] Vince Lyzinski, Gregory Sell, and Aren Jansen, "An evaluation of graph clustering methods for unsupervised term discovery," in *INTERSPEECH*, 2015.

[18] Bradley Oosterveld, Richard Veale, and Matthias Scheutz, "A parallelized dynamic programming approach to zero resource spoken term discovery," in *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*. 2017, pp. 5800–5804, IEEE.

[19] Herman Kamper, Aren Jansen, and Sharon Goldwater, "Fully unsupervised small-vocabulary speech recognition using a segmental bayesian model," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[20] Herman Kamper, Aren Jansen, and Sharon Goldwater, "A segmental framework for fully-unsupervised large-vocabulary speech recognition," *Computer Speech & Language*, vol. 46, pp. 154–174, 2017.

[21] Michael R. Brent, "An efficient, probabilistically sound algorithm for segmentation and word discovery," *Machine Learning*, vol. 34, no. 1-3, pp. 71–105, 1999.

[22] Sharon J. Goldwater, *Nonparametric Bayesian models of lexical acquisition*, Ph.D. thesis, Brown, 2007.

[23] Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson, "A Bayesian framework for word segmentation: Exploring the effects of context," *Cognition*, vol. 112, no. 1, pp. 21–54, 2009.

[24] Mohammad Norouzi and David J Fleet, "Cartesian k-means," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 3017–3024.

[25] H. Hermansky, "Perceptual linear predictive (PLP) analysis of speech," *J. Acoust. Soc. Am*, vol. 87, no. 4, pp. 1738–1752, 1990.

[26] Nils Holzenberger, Mingxig Du, Julien Karadayi, Rachid Riad, and Emmanuel Dupoux, "Learning word embeddings: unsupervised methods for fixed-size representations of variable-length speech segments," in *Interspeech-2018*, 2018.

[27] Herman Kamper, Karen Livescu, and Sharon Goldwater, "An embedded segmental k-means model for unsupervised segmentation and clustering of speech," in *Automatic Speech Recognition and Understanding Workshop (ASRU), 2017 IEEE*. 2017, pp. 719–726, IEEE.

[28] Okko Rasanen, Gabriel Doyle, and Michael C Frank, "Unsupervised word discovery from speech using automatic segmentation into syllable-like units," in *INTERSPEECH*, 2015.