AUTOMATIC TOOLS FOR ANALYZING SPOKEN HEBREW

Adiel Ben-Shalom[1], *Joseph Keshet*[2], *Doron Modan*[3], *and Asher Laufer*[3]

[1]Musicology Department, The Hebrew University, Jerusalem, Israel
[2]Department of Computer Science, Bar-Ilan University, Ramat-Gan, Israel
[3]The Phonetic Lab, Hebrew Language Dept., The Hebrew University, Jerusalem, Israel
joseph.keshet@biu.ac.il

ABSTRACT

This work summarizes our project to propose a set of automatic tools for analyzing the phonetic and phonological content of spoken Hebrew. The goal of the project is to provide a set of resources to scientists and engineers who work on research and engineering problems related to the acoustics and linguistics of the modern Hebrew language. The set of tools includes: (i) a transcribed corpus of Modern Hebrew; (ii) a phonetic classifier; (iii) a forced aligner, and (iv) an automatic procedure for converting dotted (vowelized) orthographic transcription into phonemic transcription. This tool along with the forced aligner can be used to automatically align words and their phonemes to the speech signal. We are not familiar with any such publicly available corpus or tools for the Hebrew language.

1. INTRODUCTION

In recent years, one of the main area of the speech recognition community focused on developing engines for low resource languages (i.e., the IARPA Babel Program). This paper presents our efforts on providing resources and tools for Hebrew, that can be used for the speech recognition community as well as for phoneticians and linguists. First, we present a corpus that was collected and transcribed at the phonetic lab in the Hebrew language department of the Hebrew University. The speech files in the corpus were manually annotated for their orthographic and phonetic content by graduate and undergraduate students of the lab and verified by an expert.

Then, we present a set of automatic tools for analyzing Hebrew. The set of tools includes a phonetic classifier, a forced aligner and a tool that convert dotted written Hebrew (orthography) into phonemic transcription. This tool can be used along with the forced aligner to align dotted Hebrew text into speech. All of those tools were trained on very small number of examples (roughly 10 min of Hebrew), with large-margin and kernel-based machine learning algorithms, and and propose reasonable performance.

2. CORPUS

One of the main goals of this project was to develop an open corpus for the research community. Our corpus in an ongoing process currently contains 589 seconds of recordings gathered between January 2011 and January 2013. The corpus contains radio broadcasts: a snippet from the daily news, a read story, a lecture in economy class, and a speech by prime minister Netanyahu. The primary motivation for this collection is to provide data for a research study on phone durations of Modern Hebrew. The corpus is composed of 76 utterances, which contain a total of 6227 phones; each utterance corresponds to one sentence. For each utterance we provide time-aligned phonetic transcription as well as a 16-bit 16kHz speech waveform. The transcriptions were performed by graduate and undergraduate students in the phonetic lab of the Hebrew University in Jerusalem, and verified and corrected by an expert. A list of the phones and some statistics about their duration is given in Table 1. At the bottom of the table there are 5 special phonemes: the phonemes /Z/, /tS/, and /dS/ express Hebrew pronunciation of foreign words and the phonemes /X\/ and /?\/ express dialectical production of /X/ an /?/. Those phonemes rarely occur in our collection. The corpus will be available on the web and is planned to be extended with time.

We labeled all the files by their recording source and by their transcriber. The corpus contains four types of sources (radio broadcastings: a news snippet, a read story, a lecture in economy class, and a speech by prime minister Netanyahu), respectively labeled as B01-B04. There were 7 transcribers labeled as T01-T07.

3. PHONEME CLASSIFIER

A basic tool in speech and language processing is a phoneme classifier. Given a fixed frame of speech, the phoneme classifier returns the most probable phoneme that was produced in that frame. More specifically, it returns a vector of phonemes scores. The highest score in this vector is the prediction of the most probable score.

Formally, let $\mathbf{x} \in \mathcal{X}$ represents the acoustic features of

symbol	occurrences	duration [10 ms]	
(sampa)		mean	std
p	42	8.2	3.0
b	173	7.6	2.9
t	362	8.7	3.8
d	158	7.2	2.2
k	180	8.5	2.4
g	70	7.3	2.3
?	46	4.2	2.7
f	60	10.0	3.1
v	168	6.6	2.6
s	134	11.6	3.7
z	65	8.1	2.8
S	231	10.0	5.3
X	183	9.9	3.1
h	65	3.9	2.5
ts	73	11.4	3.4
m	398	7.1	3.1
n	228	6.0	2.5
l	283	5.4	3.1
R	296	5.2	2.5
j	137	6.4	4.0
i	473	6.7	3.0
e	686	6.4	3.2
a	1109	7.3	3.8
o	294	7.8	3.7
u	182	7.9	3.5
Z	3	11.2	3.3
X\	0	0.0	0.0
tS	1	10.2	0.0
dZ	0	0.0	0.0
?\	6	5.3	2.1
sil	197	71.8	51.8

Table 1. Phonemes in corpus and their durations.

a single frame of speech, where $\mathcal{X} \subset \mathbb{R}^d$ is the the domain of the acoustics feature vectors (such as MFCCs). Denote by $p \in \mathcal{P}$, a phoneme symbol, where $\mathcal{P}$ is the set of Hebrew phoneme symbols (sampa), and there are $L = |\mathcal{P}|$ phonemes. A phoneme classier is a function $f_{\mathbf{w}} : \mathcal{X} \to \mathbb{R}^L$ from the set of acoustic feature vectors to a vector of scores of length L, and $\mathbf{w} \in \mathbb{R}^d$ are the parameters of this function that should be estimated from the corpus.

We used multiclass Passive-Aggressive algorithm [2], which is aimed at minimizing the misclassification error rate. We trained it on the corpus described in Sec. 2. We extracted standard 12 MFCC features and log energy with their deltas and double deltas to form 39-dimensional acoustic feature vectors. The window size and the frame size were 25 ms and 10 ms, respectively. We normalized the feature to have zero mean and standard deviation of 1. Then, we applied RBF kernel approximation as in [3] with a parameter that was chosen on a held-out set to be $\sigma^2 = 19$.

We ran two experiments with two types of cross validation: i) *leave-one-recording-out*: we left one recording type as a test set and the other types are used as a training set; and ii) *leave-one-transcriber-out*: we left one transcriber set of files as a test set and all other files to be used as a training set. In each round we use the training set of files to train the phoneme classifier model and then use this model to evaluate the test set. Results for these two experiments are in Table 2 and Table 3.

	Error rate
B01	35.6
B02	32.7
B03	50.5
B04	31.4

Table 2. Percentage of correctly classified phonemes for *leave-one-recording-out* task

	Error rate
B01T01	23.5
B01T02	21.5
B01T03	23.8
B01T04	24.5
B02T05	32.3
B03T06	51.6
B04T07	29.9

Table 3. Percentage of correctly classified phonemes for *leave-one-transcriber-out* task

The results show that the error rate is between 30-35% except for one recording, B03, a lecture in economy class (The performance of B03 is probably low due to the stammering of this speaker). The error rate of 30-35% is within the range of system trained on hours of examples.

4. PHONEME ALIGNER

Phoneme alignment is the task of proper positioning of a sequence of phonemes in relation to a corresponding continuous speech signal. An accurate and fast alignment procedure is a necessary tool for developing speech recognition and text-to-speech systems, but also important tool for phoneticians and linguists in automatic phonetic analysis of speech.

In the alignment problem, we are given speech utterances along with phonetic representations of the utterances. Our goal is to generate an alignment between each utterance and its phonetic representation. As before, denote the domain of the acoustic feature vectors by $\mathcal{X} \subset \mathbb{R}^d$. The acoustic feature representation of a speech signal is therefore a sequence of vectors $\bar{\mathbf{x}} = (\mathbf{x}_1, \ldots, \mathbf{x}_T)$, where $\mathbf{x}_t \in \mathcal{X}$ for all

$1 \leq t \leq T$. A phonetic representation of an utterance is defined as a string of phoneme symbols. Formally, we denote each phoneme by $p \in \mathcal{P}$, where $\mathcal{P}$ is the set of Hebrew phoneme symbols (sampa). Therefore, a phonetic representation of a speech utterance consists of a sequence of phoneme values $\bar{p} = (p_1, \ldots, p_k)$. Note that the number of phonemes clearly varies from one utterance to another and thus k is not fixed. We denote by $\mathcal{P}^\star$ (and similarly $\mathcal{X}^\star$) the set of all finite-length sequences over $\mathcal{P}$. In summary, an alignment input is a pair $(\bar{\mathbf{x}}, \bar{p})$ where $\bar{\mathbf{x}}$ is an acoustic representation of the speech signal and $\bar{p}$ is a phonetic representation of the same signal. An alignment between the acoustic and phonetic representations of a spoken utterance is a sequence of start-times $\bar{y} = (y_1, \ldots, y_k)$ where $y_i \in \mathbb{N}$ is the start-time (measured as frame number) of phoneme i in the acoustic signal. Each phoneme i therefore starts at frame y_i and ends at frame $y_{i+1} - 1$.

We denote by $\bar{y}' = f_{\mathbf{w}}(\bar{\mathbf{x}}, \bar{p})$ the alignment function that get as input the acoustic feature vectors $\bar{\mathbf{x}}$ and the phonemes $\bar{p}$ and return the start times of the phoneme in the speech, $\bar{y}'$. The function f has parameters $\mathbf{w} \in \mathbb{R}^n$ which is a vector that must be learned from training set of examples. Each example is composed of an acoustic and a phonetic representation of an utterance $(\bar{\mathbf{x}}, \bar{p})$ together with the true alignment between them, $\bar{y}$.

The quality of phoneme aligners usually assessed by a cost function. This cost function compares the predicted alignment $\bar{y}'$ with the manually labeled alignment $\bar{y}$. Often this function measures the average number of times the absolute difference between the predicted alignment sequence and the manual alignment sequence is greater than τ, were τ is set to 10 ms, 20 ms, 30 ms and 40 ms. Formally, denote by $\gamma(\bar{y}, \bar{y}')$ the cost of predicting the alignment $\bar{y}'$ where the true alignment is $\bar{y}$. This cost function is defined as follows:

$$\gamma(\bar{y}, \bar{y}') = \frac{1}{|\bar{y}|} \left| \{k : |y_k - y'_k| > \tau\} \right| . \qquad (1)$$

We used the algorithm described in [4] and its improvement in [5] to find $\mathbf{w}$. The algorithm is aimed at minimizing the cost function γ in expectation, and produces the best known results on the TIMIT dataset for American English [5]. The set of feature functions we used here is the same set used in [4], except for replacing the frame-based classifier with the one described in Sec. 3. For test purposes, we use the same cross validation settings as we did for the phoneme classifier validation. We executed two leave-one-out cross validation experiments: leave-one-recording-out, and leave-one-transcriber-out, as in the previous section. Results are listed in Table 5 and Table 4.

The results show performance similar to HMM-based systems that are trained on hours of speech (compare with [1]). Recordings B03 and B04 have lower results on average, because of the lack of data and the stuttering of the speaker B03.

	t ≤10ms	t ≤20ms	t ≤30ms	t ≤40ms
B01	72.7	84.3	89.3	91.7
B02	68.1	77.0	82.9	84.8
B03	64.7	73.4	77.2	81.2
B04	58.0	65.5	68.8	70.9

Table 4. Percentage of correctly positioned boundaries for *leave-one-recording-out* alignment task

	t ≤10ms	t ≤20ms	t ≤30ms	t ≤40ms
B01T01	74.8	86.7	91.8	94.0
B01T02	72.1	83.4	88.4	90.0
B01T03	75.1	86.2	91.2	93.8
B01T04	75.3	85.5	88.6	90.1
B02T05	71.9	81.7	87.1	89.9
B03T06	70.7	79.1	83.1	85.1
B04T07	53.4	61.0	63.9	65.8

Table 5. Percentage of correctly positioned boundaries for *leave-one-transcriber-out* alignment task

5. AUTOMATIC CONVERSION OF ORTHOGRAPHIC TRANSCRIPTION TO PHONETIC TRANSCRIPTION

Hebrew is written without vowels. The word *yeled* (child) is written with the Hebrew equivalent of *y*, *l* and *d*. It is impossible to know how to pronounce a word from the way it is written, and actually the word witten as *y*, *l* and *d* can also be read as *yalad* (gave birth) or as *yiled* (to deliver a baby). Around the 9th century, a system for indicating vowels by using small dashes and dots placed below or above the consonant letters was developed, and is still used today.

We developed a procedure for the automatic conversion of dotted Hebrew text into its phonetic content. This conversion procedure is based on a set of rules that were manually compiled. Consonants and vowels are mapped to their phonemic realization either instantly or by looking ahead or back to the previous letter or vocalization. Special rules based on the sonority level of consonants and vowels were created to predict whether the vocalization Sheva (Shva) at the beginning of a word is realized as quiescent Sheva (Shva Nah) or mobile Sheva (Shva Na'). Preliminary results shows that the phonemic transcriptions from this automatic procedure are on a par with the manual phonemic transcription of the words in the corpus. This tool allows us to use automatically aligned Hebrew dotted orthographic text with the speech. This system and its performance are described in a journal publication (under submission).

6. REFERENCES

[1] F. Brugnara, D. Falavigna, and M. Omologo. Automatic segmentation and labeling of speech based on hidden markov models. *Speech Communication*, 12:357–370, 1993.

[2] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. Online passive aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585, 2006.

[3] J. Keshet, D. McAllester, and T. Hazan. PAC-Bayesian approach for minimization of phoneme error rate. In *IEEE International Conference on Audio, Speech and Signal Processing (ICASSP)*, 2011.

[4] J. Keshet, S. Shalev-Shwartz, Y. Singer, and D. Chazan. A large margin algorithm for speech and audio segmentation. *IEEE Trans. on Audio, Speech and Language Processing*, 15(8):2373–2382, Nov. 2007.

[5] D. McAllester, T. Hazan, and J. Keshet. Direct loss minimization for structured prediction. In *Advances in Neural Information Processing Systems (NIPS) 24*, 2010.