INTERSPEECH 2017
August 20–24, 2017, Stockholm, Sweden

A relevance score estimation for spoken term detection based on RNN-generated pronunciation embeddings

Jan Švec[1], *Josef V. Psutka*[2], *Luboš Šmídl*[2], *Jan Trmal*[3]

[1]NTIS, University of West Bohemia, Czechia
[2]Department of Cybernetics, University of West Bohemia, Czechia
[3]Center for Language and Speech Processing, Johns Hopkins University, USA
[honzas,psutka_j,smidl]@kky.zcu.cz, jtrmal@gmail.com

Abstract

In this paper, we present a novel method for term score estimation. The method is primarily designed for scoring the out-of-vocabulary terms, however it could also estimate scores for in-vocabulary results. The term score is computed as a cosine distance of two pronunciation embeddings. The first one is generated from the grapheme representation of the searched term, while the second one is computed from the recognized phoneme confusion network. The embeddings are generated by specifically trained recurrent neural network built on the idea of Siamese neural networks. The RNN is trained from recognition results on word- and phone-level in an unsupervised fashion without need of any hand-labeled data. The method is evaluated on the MALACH data in two languages, English and Czech. The results are compared with two baseline methods for OOV term detection.

Index Terms: spoken term detection, recurrent neural networks, pronunciation embeddings

1. Introduction

A typical spoken term detection (STD) system is based on a large vocabulary continuous speech recognition (LVCSR) system. The in-vocabulary (IV) terms could be directly searched in a word index created from the LVCSR word lattices. The disadvantage of this approach is that we are not able to directly search the out-of-vocabulary (OOV) terms. There are two basic approaches handling this problem [1]: (1) the use of sub-word units (phonemes, syllables or word fragments) in language model [2], (2) the substitution of OOV terms with proxy words [3] or syllable-like units [4] that are acoustically similar to OOV terms. There are several techniques to normalize the posterior probabilities of different term to be on the same scale [5, 6]. By using OOV substitutions consisting of multiple units (such as proxy words or sub-word units, and generally for searching any multi-word terms), the problem how to combine the acoustic and language score of such units into a single relevance score must be solved. Many methods have been proposed, recently a recurrent neural network (RNN) based keyword verification system [1], which assigns a new relevance score to every proxy word occurrence. Another approach to the relevance score estimation is based on training a linear logistic regression model on the predicted scores from multiple STD systems to estimate a new combined relevance score [7].

The presented method is based on the concept of Siamese neural networks. Such networks find the similarity of comparable inputs. In speech processing, they have been proposed for the query-by-example task [8] or for the sentence pairs modelling task [9].

Our method is designed for the use in the STD task with large spoken archives in mind. It is a machine learning based extension of an empirical method described in [10]. The spoken archive is processed off-line, i.e. the speech is first recognized and then the index is built to be able quickly search for the user supplied query term. The search process could be implemented in two phases: (1) in the first phase, the term occurrence candidates (putative hits) are identified. In this phase, the sub-word units or proxy words could be used. (2) the second phase scores the candidates and the final results are presented to the user [11].

The application of Siamese neural network for similarity estimation in the STD task is proposed in [12]. This approach of acoustic word embeddings uses long-short term memory (LSTM) convolutional neural networks to compute the embeddings from the source acoustic signal. In large archive, such approach is not very practical, because the acoustic signal (or sequence of corresponding feature vectors) must be retrieved for each term occurrence candidate and subsequently processed in the neural network. Another task, not solved in [12], is how to find the term occurrence candidates.

For this reason, we therefore propose the *pronunciation word embeddings* which are derived from the recognized phoneme hypothesis instead of the original acoustic signal. Our approach is able to intrinsically learn the grapheme-to-phoneme mapping. This mapping is not directly observable, it decomposes into the pair of mapping: grapheme-to-pronunciation embedding and phoneme-to-pronunciation embedding. In addition, we are using multiple phoneme hypotheses instead of the one-best hypothesis. We use the phoneme sausages[1] as the input, because such representation could be easily mapped to a sequence of segments. For each segment, the feature vector is extracted and subsequently processed in the standard RNN. We used the unsupervised approach of recognizing the input audio on the word- and the phoneme-level. The stream of recognized words is then iterated over and if the word confidence is higher than a threshold, the word and the corresponding part of the phoneme sausage is used for training.

2. A Siamese network architecture

The Siamese architecture of neural networks is beneficial in tasks, where the goal is to compute similarity of two input samples. In this case, it is not necessary to train the network in fully supervised manner. It is sufficient to train the network with samples labelled as *the same* or *different*. In the original application of Siamese architecture to the query-by-example task [8], each training sample consisted of the triplet (x_a, x_s, x_d),

[1]To avoid confusion with neural networks, we use the term phoneme sausage instead of phoneme confusion network.

http://dx.doi.org/10.21437/Interspeech.2017-1087

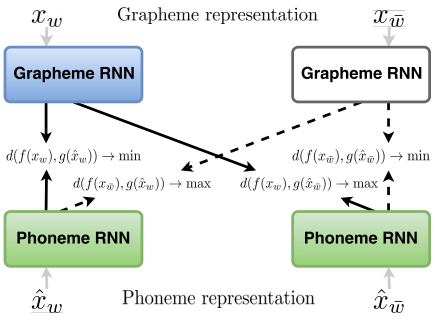

Figure 1: *Illustration of the loss function. Excluding the white node results in the original Siamese loss (Eq. 2), including it represents the symmetrized loss (Eq. 3).*

where x_a is a reference input for word w (an anchor), x_s is an input containing different realisation of the *same* word w (a positive example) and x_d is an input for a word *different* from w (a negative example). During the network training, the RNN is applied to x_a and produces an output embedding y_a. The same network could be applied to x_s to obtain y_s and x_d to obtain y_d. The goal of network training is to minimize the "relative distance" between output embeddings y_a and y_s and maximize the "relative distance" between y_a and y_d. This could be formulated as minimizing the hinge loss [13] in the form:

$$l(y_a, y_s, y_d) = \max\{0, m + d(y_a, y_s) - d(y_a, y_d)\} \quad (1)$$

where $d(y_1, y_2) = 1 - \cos(y_1, y_2)$ is cosine distance and $\cos(y_1, y_2) = \frac{y_1 \cdot y_2}{||y_1|| \cdot ||y_2||}$ is cosine similarity of two vectors y_1 and y_2. The meta-parameter m represents the margin between positive and negative examples. The training process is only lightly supervised, it requires just the correct sample pairs. Common stochastic gradient descent (SGD) algorithms could be used to optimize RNN parameters.

Previous applications of Siamese networks suppose, that all inputs (the anchor, the positive and the negative example) are from the same domain, e.g. the sequences of MFCC coefficients in the query-by-example task [13]. For the use as scoring algorithm, we changed this paradigm. The output embeddings are computed from two different representations of the same word w – the grapheme sequence x_w and the phoneme confusion network $\hat{x}_w$. The first RNN $f(x_w)$ computes the output embedding y_w from the graphemes and the second RNN $g(\hat{x}_w)$ computes the output embedding $\hat{y}_w$ from the phonemes of the word w. The loss has the same structure as Eq. 1:

$$l(w, \bar{w}) = \max\{0, m + d(f(x_w), g(\hat{x}_w)) - d(f(x_w), g(\hat{x}_{\bar{w}})\} \quad (2)$$

where $\hat{x}_{\bar{w}}$ is a phoneme confusion network of word $\bar{w}$ which is different from word w. Note that the grapheme RNN $f(\cdot)$ occur in the equation only once for word w, but the phoneme RNN $g(\cdot)$ obtains gradient updates from two different words – w and $\bar{w}$. We could therefore symmetrize the hinge loss to include both $f(x_w)$ and $f(x_{\bar{w}})$:

$$\begin{aligned} l(w, \bar{w}) &= \max\{0, m + d(f(x_w), g(\hat{x}_w)) - d(f(x_w), g(\hat{x}_{\bar{w}})\} \\ &+ \max\{0, m + d(f(x_{\bar{w}}), g(\hat{x}_{\bar{w}})) - d(f(x_{\bar{w}}), g(\hat{x}_w))\} \end{aligned} \quad (3)$$

Note: At this point it is good to compare the symmetrized criterion (Eq. 3) with the work [12] where the authors independently studied multiple choices of a symmetrization and their results are consistent with our definition of loss function.

The main difference between the Siamese networks and the classification networks is in the degree of supervision needed during training. The classification networks require the exact target label, while the Siamese networks use just the pairs of similar and dissimilar examples. The trained Siamese network does not directly predict the degree of similarity, they rather project the network inputs into an embedding space where the similarity is computed as the mutual distance of the embeddings (see Fig. 2).

3. Relevance score estimation

For the purpose of this paper, the relevance score estimation is a task of assigning a score to term occurrence candidates. This score should correlate with the probability of the given occurrence candidate being a true positive.

Since the set of audio records could be very large, the term occurrence candidates are identified using the index over sub-word units (for detail see Sec. 4). To quickly compute the term relevance score, we use a phoneme recognizer to obtain a phoneme lattice which is subsequently converted into a phoneme sausage. In this way we are able to store the phoneme sausage into the database and quickly retrieve the part corresponding to the occurrence candidate. Also, the goal of the relevance score estimation is to assign a relevance score to some segment of the input audio (represented by corresponding part of the phoneme sausage $\hat{x}_w$) given a searched term w (represented by grapheme sequence x_w).

The training data for the neural network consists of a set of pairs $(x_w, \hat{x}_w)$ extracted from the input data recognized in the unsupervised fashion. The x_w is the recognized word and $\hat{x}_w$ the corresponding phoneme sausage. We use the word x_w only if its word confidence is higher than a threshold. During training the Siamese neural network, first the pair of two different words $(w, \bar{w})$ must be sampled from the training data. To model the variations in pronunciation of words, the corresponding phoneme sausages $\hat{x}_w$ and $\hat{x}_{\bar{w}}$ are sampled from the training set of pairs. Then, the neural network is trained to optimize the criterion given by Eq. 3 using $(x_w, \hat{x}_w, x_{\bar{w}}, \hat{x}_{\bar{w}})$ as the input data.

The grapheme-to-embedding mapping $f(\cdot)$ is implemented in a straightforward way as a one layer bidirectional RNN with one maxout layer [14]. The input graphemes are encoded as one-hot vectors and projected into a space of higher dimension using the embedding layer. The phoneme-to-embedding mapping $g(\cdot)$ is also the one layer bidirectional RNN with one maxout layer. The input of the RNN $g(\cdot)$ consists of sequence of feature vectors extracted from the phoneme sausage in each time segment. A time segment of the sausage is the part between two subsequent nodes of the sausage consisting only from parallel hypothesis (phonemes and corresponding posterior probabilities).

Finally, the relevance score is estimated as a cosine similarity of the pronunciation embedding $f(x)$ obtained from the graphemes of the query x and the pronunciation embedding $g(\hat{x})$ computed from the phoneme sausage of the occurrence candidate $\hat{x}$ (see Fig. 2), formally as $\cos(f(x), g(\hat{x}))$.

Sampling of pairs of different words. During the experiments, we observed that the sampling method for selecting $(w, \bar{w})$ affects the performance of the trained networks. For the networks, it is easy to differentiate between two completely dissimilar words, and it becomes harder if the two words share a subsequence in the corresponding phoneme sausage. Since we use sub-word units to identify the term occurrence candidates, we select such pairs $(w, \bar{w})$ that share at least one common sub-word unit. More specifically we use such words $(w, \bar{w})$

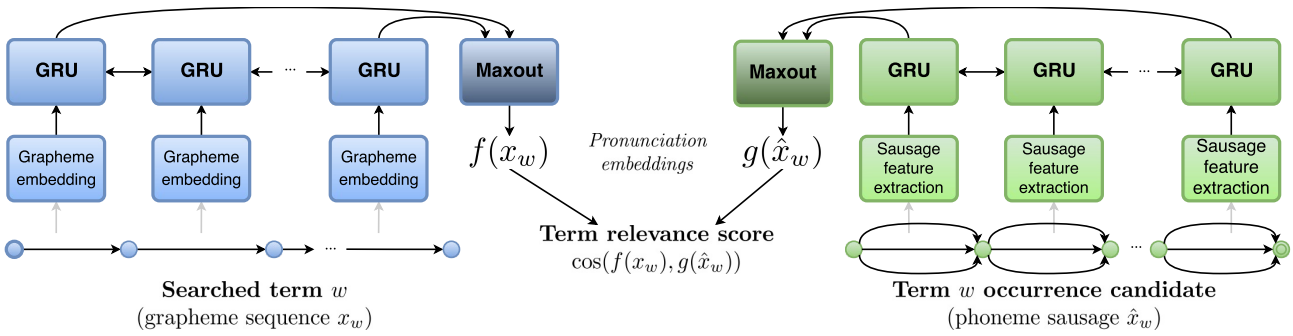

Figure 2: *Schema of term relevance score estimation.*

that share at least three subsequent phonemes from the one-best hypotheses decoded from phoneme sausages $(\hat{x}_w, \hat{x}_{\bar{w}})$.

Phoneme sausage feature extraction. To process the phoneme sausage, each time segments of the sausage (containing variable number of parallel hypotheses) must be converted to a fixed vector of features. There are several possibilities how to perform such feature extraction, e.g. bag-of-phonemes approach, the use of pooling operations or carefully designed features [15, 16]. In this paper we used the RNN to extract the features. The alternate hypotheses in a given segment are sorted according to the posterior probability. The phonemes are encoded as one-hot vectors (its dimension is equal to the number of distinct phonemes) and projected using an embedding layer. To simplify the implementation, the posterior probabilities are quantized into 32 bins, encoded as one-hot vectors and projected using another embedding layer. The outputs of phoneme- and posterior- embeddings are concatenated and processed in RNN. The feature vector describing the time segment of the sausage is then the RNN output at the last step.

4. Term occurrence candidates

In order to compute the term relevance scores, the term occurrence candidates must be determined. The term occurrence candidate is a segment of an input audio record that could be potentially a match with a searched term. In the phase of determining the occurrence candidates, only the recall is important, because it represents the upper bound on the system's performance. On the opposite side, a high number of occurrence candidates negatively affects the processing speed of the search system.

In this paper, we use the index-based method for determining the occurrence candidates. The method is based on the empirical approach [10], but it is used only to determine the candidates and no relevance score estimation is performed. To index the archive we use the sub-word units, particularly the triplets of phonemes. The phoneme triplets are extracted from the recognized phoneme lattices by converting the lattice into a factor automaton and composition of such automaton with the "filter" which selects only paths of length 3. In this way, we obtain set of overlapping phoneme triplets with their time alignment and posterior probabilities. Such set is indexed in the database so that for a given triplet we are able to find all occurrences in the recognized phoneme lattices.

In the search phase, the searched term is first transcribed into a sequence of phonemes. Then, this sequence is converted into a sequence of overlapping phoneme triplets and these triplets are looked up in the index. The found triplets are then sorted according to the time of the occurrence and clustered so that the clusters are separated by at least 0.3 seconds. Then, each such cluster is declared as an occurrence candidate and subsequently the term relevance score is estimated.

Note that this approach still relies on the availability of the phonetically transcribed searched term. In our case we used the same grapheme-to-phoneme mapping which was used for generating the pronunciation lexicon of the speech recognizer. In the future research we would like to focus on the possibility of generating the searched triplets directly from the pronunciation embedding.

5. Experiments

The presented method was evaluated on the data from a USC-SFI MALACH archive in two languages – English [17] and Czech [18]. The archive for each language was recognized using the word- and phoneme-level recognizers. Then, the time aligned transcripts of words with confidence score (generated by a recognition engine) higher than 0.95 were processed and the corresponding phoneme sausages were generated. For each word x_w, we used at most 50 examples $\hat{x}_w$. The pairs were generated for English and Czech from 2,037 and 997 hours of audio, the resulting training set contained 277k and 672k pairs $(x_w, \hat{x}_w)$ and 21k and 83k distinct words x_w, respectively. Then, the RNNs using criterion from Eq. 3 were trained for each language.

In our experiments, we used the gated recurrent units (GRUs) [19] implemented in Keras [20]. We also experimented with LSTM units, but the results were similar to GRUs. In the experiments we used the following RNN structure: grapheme and phoneme embedding width, number of recurrent units and number of maxout neurons were 256, the width of the posterior embedding was 32.

For Czech, during one training epoch the network saw each word w exactly once with a counterexample $\bar{w}$ sampled using procedure described in Sec. 3. For English, the word w was used five times with five different counterexamples during one epoch. The training data for each epoch were randomly shuffled and the ADAM optimization [21] was used to train the network parameters for 30 epochs, with the dropout 0.2 and the margin width $m = 0.5$ (Eq. 3).

Speech recognition. We followed a typical Kaldi [13] training recipe for a deep neural network (DNN) acoustic model training. This recipe supports layer-wise RBM pre-training, stochastic gradient descent training supported by GPUs and sequence-discriminative training optimizing sMBR criterion. We applied the standard 6 layers topology (5 hidden layers, each with 2048 neurons) with a softmax layer. The output dimension was equal to the number of context-dependent states (4521 for English, 4557 for Czech). We used features based on standard 12-dimensional Cepstral Mean Normalized (CMN) PLP coefficients with first and second derivatives. In total, the English

Table 1: *Statistics of development and test sets.*

	English		Czech	
	Dev	Test	Dev	Test
LVCSR vocabulary	243,699		252,082	
#speakers	10	10	10	10
OOV rate	0.5%	3.2%	0.3%	2.6%
LVCSR WER	24.10	19.66	23.98	19.11
#IV terms	597	601	1680	1673
#OOV terms	31	6	1145	948
dataset length [hours]	11.1	11.3	20.4	19.4

Table 2: *Results on the development dataset (MTWV)*

		English	Czech
Empirical	IV terms, phonetically	0.4729	0.6471
	OOV terms, phonetically	0.3350	0.5864
	All terms, phonetically	0.4636	0.6225
	All terms, LVCSR+phonemes	0.6935	0.6734
Proxies	IV terms, LVCSR	0.7899	0.9015
	OOV terms, proxy	0.1543	0.4201
	All terms, LVCSR only	0.7509	0.5361
	All terms, LVCSR+proxy	0.7586	0.6842
Siamese	IV terms, phonetically	0.5115	0.6752
	OOV terms, phonetically	0.3350	0.6255
	All terms, phonetically	0.5012	0.6547
	All terms, combination	0.7650	0.7758

acoustic model was trained from 217 hours and the Czech from 84 hours of signal. We used our in-house real-time decoder both for the word- and phoneme recognition with trigram word- and 5-gram phoneme language model.

Evaluation of STD. To evaluate this approach, we used the development and test dataset used to train and tune the speech recognizer. The Tab. 1 shows important characteristics of these datasets. The sets of terms used in the evaluation were generated automatically. First, all terms in the dataset were selected and filtered to satisfy the following conditions: (1) it has more than three phonemes, (2) its phonetic transcription is not a subsequence or near-subsequence of phonetic transcription of another term. We used the ATWV metric to evaluate the performance [22]. The decision threshold was determined on the development set and therefore Tab. 2 virtually shows the MTWV metric as reported by Kaldi [23, 24]. Then the optimal thresholds were applied to the test set and the results are reported in Tab. 3.

First of all, we used the baseline empirical approach to phoneme-based search described in [10]. The first three rows of Tab. 2 show the results for searching the IV, OOV and all terms phonetically. The difference between IV and OOV terms is caused by the influence of the phoneme language model, since the 5-grams are able to model whole words and therefore higher ATWV is expected for IV terms. The row *LVCSR+phonemes* shows the performance of the combined system, where IV terms are searched in word lattices and OOV terms are searched phonetically.

The second group of results were obtained using proxy words for OOV terms [3]. In this case, all term relevance scores were computed form LVCSR confidence scores. The ATWV for IV terms is rather high for this case, because the LVCSR hits contained only a few false-alarms. The search of

Table 3: *Results on the test dataset (ATWV)*

All terms	English	Czech
phonetically (empirical)	0.4435	0.6564
phonetically (Siamese)	**0.4956**	**0.6873**
LVCSR+phonemes (empirical)	0.7191	0.6840
LVCSR+proxy words	0.7209	0.7013
combination (Siamese)	**0.7263**	**0.7703**

OOV terms using proxy words provided lower ATWV score than the empirical method. The higher ATWV values (row *LVCSR+proxy*) for the combined system (compare with the row *Empirical/LVCSR+phonemes*) were caused by the fact that the term relevance score for IV and OOV terms respectively were generated from LVCSR confidence scores and therefore were more homogeneous. Obviously, the capability of searching OOV terms improved the ATWV in comparison with the case, where only IV terms are searched and OOV terms are ignored (row *LVCSR only*).

The last group contains results for the method based on Siamese networks. It was first evaluated as pure phoneme-based search. The method overcome the baseline empirical method for both the IV and OOV terms and their union. To obtain the combined results, we took the IV term occurrences generated by LVCSR and OOV term occurrences from the phoneme-based search. The resulting list was scored using the method of Siamese networks, which resulted in calibrated scores with higher ATWV than both baselines.

The results on the test data (Tab. 3) shows that the phoneme-based search outperforms the baseline empirical method and also the combined system involving the use of LVCSR outperforms the baselines. The results are reproducible using another dataset which differs in the speakers and the searched terms. The table also shows the stability of the optimal threshold, which was estimated on the development data for each method.

6. Conclusions

The method of searching the OOV terms based on Siamese networks clearly outperforms the baseline methods, both on the OOV terms and also on the task where all terms are searched phonetically. The method naturally generates the relevance scores both for IV and OOV terms and therefore such results could be easily merged without any loss in the ATWV metric. The method implicitly models different pronunciation variants of terms and also the phoneme recognition errors. The recognized phoneme lattices could be easily indexed using just phoneme triplets. To estimate the relevance score, only the time-aligned phoneme sausages must be stored. The increased computation requirements caused by the use of RNNs could be compensated by pre-computing the feature vectors for phoneme sausage segments and storing them instead of the sausages itself.

In the future work, we would like to focus on the reduction of required training data by using the pronunciation vocabulary to pre-train the RNNs and subsequently adopting the RNNs using the unsupervised data from word and phoneme recognizers.

7. Acknowledgements

This research was supported by the Grant Agency of the Czech Republic, project No. GAČR GBP103/12/G084. Jan Trmal was supported by the NSF grant No CRI-1513128.

8. References

[1] L. Zhiqiang, J. Kang, W.-Q. Zhang, and J. Liu, "An LSTM-CTC based verification system for proxy-word based OOV keyword search," in *Proceedings of ICASSP 2017*, 2017, pp. 5655–5659.

[2] J. Mamou, B. Ramabhadran, and O. Siohan, "Vocabulary Independent Spoken Term Detection," *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 615–622, 2007.

[3] G. Chen, O. Yilmaz, J. Trmal, D. Povey, and S. Khudanpur, "Using proxies for OOV keywords in the keyword search task," in *2013 IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2013 - Proceedings*, 2013, pp. 416–421.

[4] B. Logan, J. M. Van Thong, and P. J. Moreno, "Approaches to reduce the effects of OOV queries on indexed spoken audio," *IEEE Transactions on Multimedia*, vol. 7, no. 5, pp. 899–906, 2005.

[5] D. Karakos, R. Schwartz, S. Tsakalidis, L. Zhang, S. Ranjan, T. Ng, R. Hsiao, G. Saikumar, I. Bulyko, L. Nguyen, J. Makhoul, F. Grezl, M. Hannemann, M. Karafiat, I. Szoke, K. Vesely, L. Lamel, and V. B. Le, "Score normalization and system combination for improved keyword spotting," in *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, vol. 26, no. 3, 2013, pp. 210–215.

[6] J. Mamou, J. Cui, X. Cui, M. J. F. Gales, B. Kingsbury, K. Knill, L. Mangu, D. Nolden, M. Picheny, B. Ramabhadran, R. Schlter, A. Sethy, and P. C. Woodland, "System combination and score normalization for spoken term detection," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, May 2013, pp. 8272–8276.

[7] J. van Hout, L. Ferrer, D. Vergyri, N. Scheffer, Y. Lei, V. Mitra, and S. Wegmann, "Calibration and multiple system fusion for spoken term detection using linear logistic regression," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, no. February, 2014, pp. 7138–7142.

[8] S. Settle and K. Livescu, "Discriminative acoustic word embeddings: Tecurrent neural network-based approaches," in *2016 IEEE Spoken Language Technology Workshop (SLT)*, Dec 2016, pp. 503–510.

[9] W. Yin, H. Schtze, B. Xiang, and B. Zhou, "ABCNN: Attention-Based Convolutional Neural Network for Modeling Sentence Pairs," *Transactions of the Association for Computational Linguistics*, vol. 4, pp. 259–272, 2016.

[10] J. Psutka, J. Švec, J. V. Psutka, J. Vaněk, A. Pražák, L. Šmídl, and P. Ircing, "System for Fast Lexical and Phonetic Spoken Term Detection in a Czech Cultural Heritage Archive," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2011, no. 1, p. 10, 2011.

[11] P. Stanislav, J. Švec, and P. Ircing, "An engine for online video search in large archives of the holocaust testimonies," in *Proceedings of Interspeech 2016*. International Speech Communication Association, 2016.

[12] W. He, W. Wang, and K. Livescu, "Multi-view Recurrent Neural Acoustic Word Embeddings," *Appearing in ICLR 2017*, pp. 1–12, nov 2017. [Online]. Available: http://arxiv.org/abs/1611.04496v2

[13] H. Kamper, W. Wang, and K. Livescu, "Deep convolutional acoustic word embeddings using word-pair side information," in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2016-May, 2016, pp. 4950–4954.

[14] I. J. Goodfellow, D. Warde-farley, M. Mirza, A. Courville, and Y. Bengio, "Maxout networks," in *Proceedings of International Conference on Machine Learning*, 2013.

[15] N. Henderson, M. Gašić, B. Thomson, P. Tsiakoulis, K. Yu, and S. Young, "Discriminative Spoken Language Understanding Using Word Confusion Networks," in *Spoken Language Technology Workshop (SLT), 2012 IEEE*, 2012, pp. 176–181.

[16] V. Soto, E. Cooper, L. Mangu, A. Rosenberg, and J. Hirschberg, "Rescoring Confusion Networks for Keyword Search," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2014*, 2014, pp. 7138–7142.

[17] B. Ramabhadran, S. Gustman, W. Byrne, J. Hajič, D. Oard, J. S. Olsson, M. Picheny, and J. Psutka, "USC-SFI MALACH Interviews and Transcripts English LDC2012S05," 2012. [Online]. Available: https://catalog.ldc.upenn.edu/LDC2012s05

[18] J. Psutka, V. Radová, P. Ircing, J. Matoušek, and L. Müller, "USC-SFI MALACH Interviews and Transcripts Czech LDC2014S04," 2014. [Online]. Available: https://catalog.ldc.upenn.edu/LDC2014S04

[19] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling," *Presented at the Deep Learning workshop at NIPS2014*, pp. 1–9, 2014.

[20] F. Chollet, "Keras," https://github.com/fchollet/keras, 2015.

[21] D. P. Kingma, "ADAM: A Method for Stochastic Optimization," *Proceedings of 3rd International Conference for Learning Representations*, pp. 1–15, 2015.

[22] J. G. Fiscus, J. Ajot, J. S. Garofolo, and G. Doddington, "Results of the 2006 spoken term detection evaluation," in *Proceedings of the ACM SIGIR Conference*, vol. 7, 2007, pp. 51–57.

[23] D. Povey, A. Ghoshal, N. Goel, M. Hannemann, Y. Qian, P. Schwarz, and G. Stemmer, "The Kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. Big Island, Hawaii: IEEE, 2011.

[24] S. Wegmann, A. Faria, A. Janin, K. Riedhammer, and N. Morgan, "The TAO of ATWV: Probing the mysteries of keyword search performance," in *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, Dec 2013, pp. 192–197.