Interspeech 2018
2-6 September 2018, Hyderabad

Speech2Vec: A Sequence-to-Sequence Framework for Learning Word Embeddings from Speech

Yu-An Chung *James Glass*

Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139, USA
{andyyuan,glass}@mit.edu

Abstract

In this paper, we propose a novel deep neural network architecture, Speech2Vec, for learning fixed-length vector representations of audio segments excised from a speech corpus, where the vectors contain semantic information pertaining to the underlying spoken words, and are close to other vectors in the embedding space if their corresponding underlying spoken words are semantically similar. The proposed model can be viewed as a speech version of Word2Vec [1]. Its design is based on a RNN Encoder-Decoder framework, and borrows the methodology of skipgrams or continuous bag-of-words for training. Learning word embeddings directly from speech enables Speech2Vec to make use of the semantic information carried by speech that does not exist in plain text. The learned word embeddings are evaluated and analyzed on 13 widely used word similarity benchmarks, and outperform word embeddings learned by Word2Vec from the transcriptions.

Index Terms: word embeddings, recurrent neural networks, sequence-to-sequence learning, skipgrams, bag-of-words

1. Introduction

Natural language processing (NLP) techniques such as Word2Vec [1, 2] and GloVe [3] transform words into fixed dimensional vectors, or word embeddings. The embeddings are obtained via unsupervised learning from co-occurrence information in text, and contain semantic information about the word which are useful for many NLP tasks [4, 5, 6, 7, 8].

Researchers have also explored the concept of learning vector representations from speech [9, 10, 11, 12, 13, 14]. These approaches are based on notions of acoustic-phonetic (rather than *semantic*) similarity, so that different instances of the same underlying word would map to the same point in a latent embedding space. Our work, highly inspired by Word2Vec [1], uses a skipgrams or continuous bag-of-words formulation to focus on *neighboring* acoustic regions, rather than the acoustic segment associated with the word itself. We show that the resulting acoustic embedding space is more semantic in nature.

Recent research by [15, 16, 17] has presented a deep neural network model capable of rudimentary spoken language acquisition using raw speech training data paired with contextually relevant images. Using this contextual grounding, the model learned a latent semantic audio-visual embedding space. In this paper, we propose a deep neural network architecture capable of learning embeddings of audio segments corresponding to words from *raw* speech without any other modalities. The proposed model, called Speech2Vec, integrates an RNN Encoder-Decoder framework [18, 19] with the concept of skipgrams or continuous bag-of-words, and can handle arbitrary length audio segments. The resulting word embeddings contain information pertaining to the meaning of the underlying spoken words such that semantically similar words produce vector representations that are nearby in the embedding space.

Speech2Vec can be viewed as a speech version of Word2Vec. Traditionally, when we want to learn word embeddings from speech, we need to first transcribe the speech into text by an ASR system, then apply a textual word embedding method on the transcripts. The motivations for this work are that learning word embeddings directly from speech surmounts the recognition errors caused by the process of transcribing. Moreover, speech contains richer information than text such as prosody, and a machine should be able to make use of this information in order to learn better semantic representations.

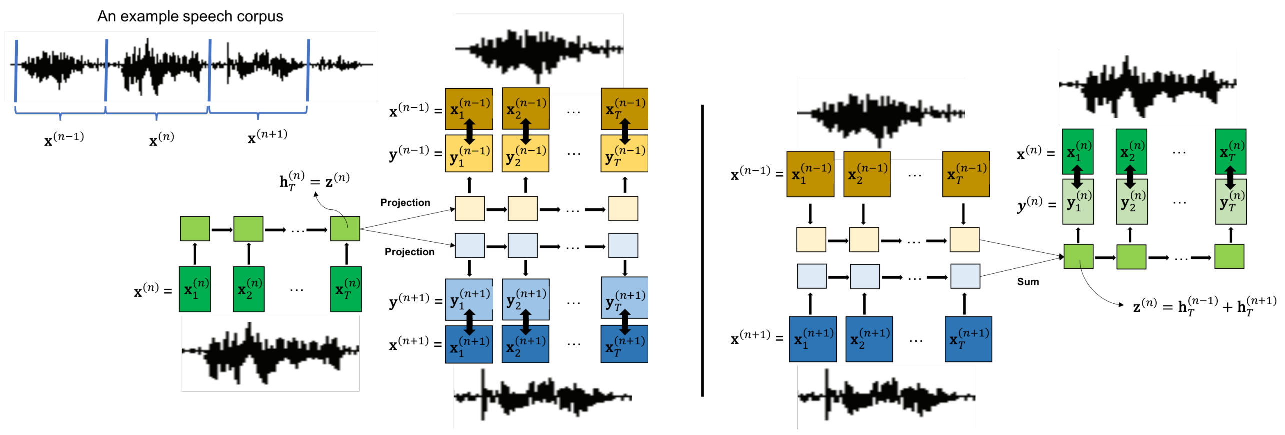
In this paper, we build on a preliminary version of the Speech2Vec model [20] by introducing additional methodologies and details for training the model, comparing the model with additional baseline approaches, and providing systematic analysis and visualizations of the learned word embeddings.

2. Proposed Approach

Our goal is to learn a fixed-length embedding of a audio segment corresponding to a word that is represented by a variable-length sequence of acoustic features such as Mel-Frequency Cepstral Coefficients (MFCCs), $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T)$, where $\mathbf{x}_t$ is the acoustic feature at time t and T is the length of the sequence. We desire that this word embedding is able to describe the semantics of the original audio segment to some degree. Here we first review the RNN Encoder-Decoder framework, followed by formally proposing the Speech2Vec model.

2.1. RNN Encoder-Decoder Framework

A Recurrent Neural Network (RNN) Encoder-Decoder consists of an Encoder RNN and a Decoder RNN [18, 19]. For an input sequence $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T)$, the Encoder reads each of its symbol $\mathbf{x}_i$ sequentially, and the hidden state $\mathbf{h}_t$ of the RNN is updated accordingly. After the last symbol $\mathbf{x}_T$ is processed, the corresponding hidden state $\mathbf{h}_T$ is interpreted as the learned representation of the entire input sequence. Subsequently, by initializing its hidden state using $\mathbf{h}_T$, the Decoder generates an output sequence $\mathbf{y} = (\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_{T'})$ sequentially, where T and T' can be different. Such a sequence-to-sequence framework does not constrain the input or target sequences, and has been successfully applied to tasks such as machine translation [21, 22], video caption generation [23], abstract meaning representation parsing and generation [24], and acoustic word embeddings acquisition [11].



(a) Speech2Vec with skipgrams (b) Speech2Vec with cbow

Figure 1: *The structures of Speech2Vec trained with skipgrams and cbow, respectively. All audio segments were padded by zero vectors into the same length T. Note that when training Speech2Vec with skipgrams, it is the same Decoder RNN that generates all the output audio segments; when training Speech2Vec with cbow, it is the same Encoder RNN that encodes all the input audio segments.*

2.2. Speech2Vec

The backbone of Speech2Vec is the RNN Encoder-Decoder framework. Inspired by Word2Vec, here we propose two methodologies for training Speech2Vec: skipgrams and continuous bag-of-words (cbow). The two variants are depicted in Figure 1(a) and Figure 1(b), respectively.

2.2.1. Training Speech2Vec with skipgrams

The idea of training Speech2Vec with skipgrams is that for each audio segment $\mathbf{x}^{(n)}$ (corresponding to a word) in a speech corpus, the model is trained to predict the audio segments $\{\mathbf{x}^{(n-k)}, ..., \mathbf{x}^{(n-1)}, \mathbf{x}^{(n+1)}, ..., \mathbf{x}^{(n+k)}\}$ (corresponding to nearby words) within a certain range k before and after $\mathbf{x}^{(n)}$. During training, the Encoder first takes $\mathbf{x}^{(n)}$ as input and encodes it into a vector representation of fixed dimensionality $\mathbf{z}^{(n)}$. The Decoder then maps $\mathbf{z}^{(n)}$ to several output sequences $\mathbf{y}^{(i)}, i \in \{n-k, ..., n-1, n+1, ..., n+k\}$. The model is trained by minimizing the gap between the output sequences and their corresponding nearby audio segments, measured by the general mean squared error $\sum_i \left\|\mathbf{x}^{(i)} - \mathbf{y}^{(i)}\right\|^2$. The intuition behind the this approach is that, in order to successfully decode nearby audio segments, the encoded vector representation $\mathbf{z}^{(n)}$ should contain sufficient semantic information about the current audio segment $\mathbf{x}^{(n)}$. After training, $\mathbf{z}^{(n)}$ is taken as the word embedding of $\mathbf{x}^{(n)}$.

2.2.2. Training Speech2Vec with cbow

In contrast to training Speech2Vec with skipgrams that aim to predict nearby audio segments from $\mathbf{z}^{(n)}$, training Speech2Vec with cbow sets $\mathbf{x}^{(n)}$ as the target and aims to infer it from nearby audio segments. During training, all nearby audio segments are encoded by a shared Encoder into $\mathbf{h}^{(i)}, i \in \{n-k, ..., n-1, n+1, ..., n+k\}$, and their sum $\mathbf{z}^{(n)} = \sum_i \mathbf{h}^{(i)}$ is then used by the Decoder to generate $\mathbf{x}^{(n)}$. After training, $\mathbf{z}^{(n)}$ is taken as the word embedding for $\mathbf{x}^{(n)}$. In our experiments, we found that Speech2Vec trained with skipgrams consistently outperforms that trained with cbow.

2.3. Differences between Speech2Vec and Word2Vec

The proposed Speech2Vec aims to learn a fixed-length embedding of an audio segment that captures the semantic information of the spoken word directly from audio. It can be viewed as a speech version of Word2Vec. Although they have many properties in common, such as sharing the same training methodologies (skipgrams and cbow), and learning word embeddings that capture semantic information from their respective modalities, it is important to identify two fundamental differences. First, the architecture of a Word2Vec model is a two-layered fully-connected neural network with one-hot encoded vectors as input and output. In contrast, the Speech2Vec model is composed of Encoder and Decoder RNNs, in order to handle variable-length input and output sequences of acoustic features. Second, in a Word2Vec model, the embedding for a particular word is deterministic. Every instance of the same word will be represented by one, and only one, embedding vector. In contrast, in the Speech2Vec model, due to the fact that every instance of a spoken word will be different (due to speaker, channel, and other contextual differences etc.), every instance of the same underlying word will be represented by a *different* (though hopefully similar) embedding vector. For experimental purposes, in this work, all vectors representing instances of the same spoken word are averaged to obtain a single word embedding. The effect of this averaging operation is discussed in Section 3.

3. Experiments

3.1. Dataset

For our experiments we used LibriSpeech [25], a corpus of read English speech, to learn Speech2Vec embeddings. In particular, we used a 500 hour subset of broadband speech produced by 1,252 speakers. Speech features consisting of 13 dimensional Mel Frequency Cepstral Coefficients (MFCCs) were produced every 10ms. The speech was segmented according to word boundaries obtained by forced alignment with respect to the reference transcriptions such that each audio segment corresponds to a spoken word. This resulted in a large set of audio segments $\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, ..., \mathbf{x}^{(|C|)}\}$, where $|C|$ denotes the total number of audio segments (words) in the corpus.

Table 1: *The relationship between the embedding size and the performance. For each of the four models, the highest ρ of each benchmark is marked in bold. The highest ρ of each benchmark among all four models is further colored in red.*

Model	Speech2Vec								Word2Vec							
	cbow				skipgrams				cbow				skipgrams			
Vector dim.	10	50	100	200	10	50	100	200	10	50	100	200	10	50	100	200
Verb-143	0.182	**0.223**	0.203	0.205	0.263	**0.315**	0.276	0.222	0.296	0.380	0.383	**0.385**	0.307	0.378	**0.384**	0.365
SimLex-999	0.183	0.235	**0.238**	0.237	0.200	0.292	0.317	**0.335**	0.118	**0.146**	0.142	0.140	0.202	0.280	0.298	**0.300**
MC-30	0.680	**0.716**	0.688	0.684	0.701	**0.846**	0.815	0.787	0.524	**0.539**	0.532	0.521	0.726	**0.762**	0.746	0.713
WS-353	0.305	**0.343**	0.336	0.335	0.370	**0.508**	0.502	0.498	0.198	**0.234**	0.228	0.233	0.334	0.452	0.455	**0.471**
WS-353-SIM	0.461	**0.484**	0.474	0.471	0.533	**0.663**	0.653	0.636	0.313	**0.335**	0.330	0.334	0.491	0.602	0.599	**0.605**
WS-353-REL	0.122	**0.192**	0.189	0.186	0.207	**0.346**	0.332	0.331	0.051	**0.106**	0.095	0.100	0.172	0.308	0.308	**0.327**
RG-65	0.676	**0.705**	0.699	0.697	0.702	**0.790**	0.756	0.740	0.421	0.425	0.424	**0.428**	0.666	**0.752**	0.749	0.724
MEN	0.476	**0.509**	0.501	0.498	0.543	**0.619**	0.606	0.573	0.427	**0.465**	0.461	0.459	0.563	0.642	**0.646**	0.632
MTurk-287	0.346	**0.349**	0.336	0.331	0.426	**0.468**	0.442	0.398	0.368	0.387	**0.390**	0.389	0.430	**0.504**	0.503	0.469
MTurk-771	0.356	**0.391**	0.380	0.377	0.445	**0.521**	0.503	0.463	0.246	**0.290**	0.289	0.288	0.413	0.499	**0.504**	0.479
SimVerb-3500	0.098	0.122	**0.126**	0.125	0.100	0.157	0.183	**0.204**	0.049	**0.075**	0.072	0.069	0.090	0.149	0.176	**0.193**
Rare-Word	0.240	0.273	**0.275**	0.269	0.249	**0.323**	0.321	0.317	0.230	0.307	0.309	**0.310**	0.286	0.408	0.419	**0.431**
YP-130	0.198	**0.216**	0.211	0.214	0.322	0.321	**0.334**	0.302	0.231	**0.261**	0.257	0.253	0.345	0.391	0.431	**0.448**

3.2. Implementation and Training Details

We implemented the Speech2Vec model with PyTorch [26]. The Encoder RNN is a single-layered bidirectional LSTM [27], and the Decoder RNN is another single-layered unidirectional LSTM. To facilitate the learning process, we also adopted the attention mechanism that enables the Decoder to condition every decoding step on the last hidden state of the Encoder [28], in other words, the Decoder can refer to $\mathbf{h}_T$ when generating every symbol $\mathbf{y}_t$ of the output sequence $\mathbf{y}$. The window size k for training the model with skipgrams and cbow is set to three. The model was trained by stochastic gradient descent (SGD) without momentum, with a fixed learning rate of $1e-3$ and 500 epochs. We experimented with hyperparameter combinations for training the Speech2Vec model, including the depths of the Encoder and Decoder RNNs, which memory cell (LSTM or GRU) to use, and bidirectional or unidirectional RNNs. We conducted experiments using the specified architecture since it produced the most stable and satisfactory results.

3.3. Evaluation

Existing schemes for evaluating methods for word embeddings fall into two major categories: extrinsic and intrinsic [29]. With the extrinsic method, the learned word embeddings are used as input features to a downstream task [4, 5, 6, 7, 8], and the performance metric varies from task to task. The intrinsic method directly tests for semantic or syntactic relationships between words, and includes the tasks of word similarity and word analogy [1]. In this paper, we focus on the intrinsic method, especially the word similarity task, for evaluating and analyzing the Speech2Vec word embeddings.

We used 13 benchmarks [30] to measure word similarity, including **WS-353** [31], **WS-353-REL** [32], **WS-353-SIM**, **MC-30** [33], **RG-65** [34], **Rare-Word** [35], **MEN** [36], **MTurk-287** [37], **MTurk-771** [38], **YP-130** [31], **SimLex-999** [39], **Verb-143** [40], and **SimVerb-3500** [41]. These 13 benchmarks contain different numbers of pairs of English words that have been assigned similarity ratings by humans, and each of them evaluates the word embeddings in terms of different aspects. For example, **RG-65** and **MC-30** focus on nouns, **YC-130** and **SimVerb-3500** focus on verbs, and **Rare-Word** focuses on rare-words. The similarity between a given pair of words was calculated by computing the cosine similarity between their corresponding word embeddings. We then reported the Spearman's rank correlation coefficient ρ between the rankings produced by each model against the human rankings [42].

We compared Speech2Vec trained with skipgrams or cbow with its Word2Vec counterpart trained on the transcriptions of the LibriSpeech corpus using the fastText implementation [2]. For convenience, we refer to these four models as skipgrams Speech2Vec, cbow Speech2Vec, skipgrams Word2Vec, and cbow Word2Vec, respectively.

3.4. Results and Discussions

We trained the four models with different embedding sizes to understand how large the embedding size should be to capture sufficient semantic information about the word. The results are shown in Table 1. We also varied the size of the corpus used for training the four models and report the results in Table 2.[1] The numbers in both tables are the average of running the experiment 10 times and the standard deviations are negligible. From Table 1 and Table 2, we have the following observations.

Embedding size impact on performance. We found that increasing the embedding size does not always result in improved performance. For cbow Speech2Vec, skipgrams Speech2Vec, and cbow Word2Vec, word embeddings of 50-dimensions are able to capture enough semantic information of the words, as the best performance (highest ρ) of each benchmark is mostly achieved by them. For skipgrams Word2Vec, although the best performance of 7 out of 13 benchmarks is achieved by word embeddings of 200-dims, there are 6 benchmarks whose best performance is achieved by word embeddings of other sizes.

Comparing Speech2Vec to Word2Vec. From Table 1 we see that skipgrams Speech2Vec achieves the highest ρ in 8 out of 13 benchmarks, outperforming cbow and skipgrams Word2Vec in combination. We believe a possible reason for such results is due to skipgrams Speech2Vec's ability to capture semantic information present in speech such as prosody that is not in text.

Comparing skipgrams to cbow Speech2Vec. From Table 1 we observe that skipgrams Speech2Vec consistently outperforms cbow Speech2Vec on all benchmarks for all embedding sizes. This result aligns with the empirical fact that skipgrams Word2Vec is likely to work better than cbow Word2Vec with small training corpus size [1].

[1] For Table 2, we randomly picked six benchmarks to report due to page limits, but the conclusion remains the same.

Table 2: *The relationship between the size of the training corpus and the performance. The percentage denotes the proportion of the entire corpus that was used for training the models. The reported results are based on the word embeddings of 50-dim.*

Model	Speech2Vec								Word2Vec							
	cbow				skipgrams				cbow				skipgrams			
Training size	10%	40%	70%	100%	10%	40%	70%	100%	10%	40%	70%	100%	10%	40%	70%	100%
Verb-143	0.090	0.071	0.116	0.223	0.098	0.152	0.220	0.315	0.196	0.257	0.331	0.380	0.148	0.259	0.328	0.378
WS-353	-0.101	0.211	0.319	0.343	0.066	0.392	0.459	0.508	0.045	0.091	0.167	0.234	0.129	0.377	0.412	0.452
RG-65	0.024	0.199	0.593	0.705	0.020	0.605	0.661	0.790	0.196	0.192	0.333	0.425	0.330	0.416	0.642	0.752
MEN	0.033	0.311	0.451	0.509	0.283	0.506	0.585	0.619	0.016	0.258	0.403	0.465	0.247	0.541	0.621	0.642
MTurk-771	0.098	0.246	0.321	0.391	0.186	0.416	0.462	0.521	0.094	0.148	0.223	0.290	0.182	0.392	0.474	0.499
YP-130	-0.027	0.067	0.181	0.216	0.097	0.196	0.311	0.321	0.064	0.085	0.182	0.256	0.403	0.216	0.365	0.391

Impact of training corpus size. From Table 2 we observe that when 10% of the corpus was used for training, the resulting word embeddings perform poorly. Unsurprisingly, the performance continues to improve as training size increases.

3.5. Variance Study

In Section 2.3 we mention that in Speech2Vec, every instance of a spoken word will produce a different embedding vector. Here we try to understand how the vectors for a given word vary. i.e., are they similar, or is there considerable variance that the averaging operation adopted in this paper smooths out?

To study this, we partitioned all words into four sub-groups based on the number of times, N, that they appeared in the corpus, ranging from $5 \sim 99, 100 \sim 999, 1000 \sim 9999$, and $\geq 10k$. Then, for all vector representations $\{\mathbf{w}^1, \mathbf{w}^2, ..., \mathbf{w}^N\}$ of a given word w that appeared N times, we computed the mean of the standard deviations of each dimensions $m_w = \frac{1}{d}\sum_{i=1}^{d} \texttt{std}(\mathbf{w}^1, \mathbf{w}^2, ..., \mathbf{w}^N)$, where d denotes the embedding size. Finally, we averaged m_w for every word w that belongs to the same sub-group and reported the results in Figure 2.

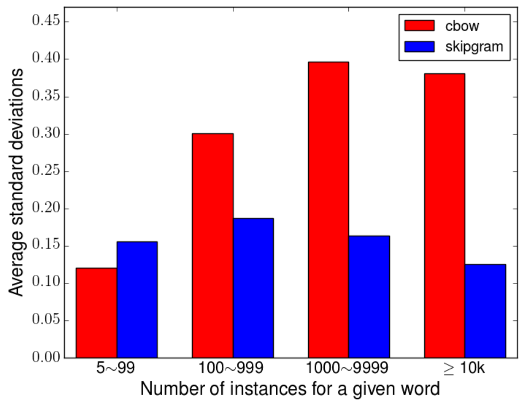


Figure 2: *How the vector representations for a given word vary with respect to the times it appears in the corpus.*

From Figure 2 we observe that when N falls in $5 \sim 99$, the variances of the vectors generated by cbow Speech2Vec are smaller than those generated by skipgrams Speech2Vec. However, when N becomes bigger, variances of the vectors generated by skipgrams Speech2Vec become smaller than those generated by cbow Speech2Vec, and the gap continues to grow as N increases. We suspect the lower variation of the skipgrams model relative to the cbow model is related to the overall superior performance of the skipgrams Speech2Vec model. We are encouraged that the deviation of the skipgrams model gets smaller as N increases, as it suggests stability in the model.

3.6. Visualizations

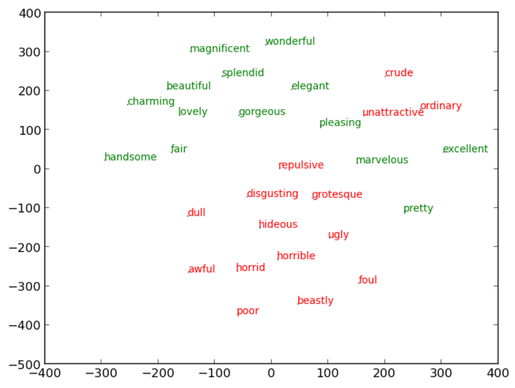


Figure 3: *t-SNE projection of the word embeddings learned by skipgrams Speech2Vec. Words with positive and negative meanings were colored in green and red, respectively.*

We visualized the word embeddings learned by skipgrams Speech2Vec with t-SNE [43] using `http://www.wordvectors.org/` in Figure 3. We see that words with positive meanings (colored in green) are mainly located at the upper part of the figure, while words with negative meanings (colored in red) are mostly located at the bottom. Such distribution suggests that the learned word embeddings do capture notions of antonym and synonyms to some degree.

4. Conclusions and Future Work

Speech2Vec, which integrates a RNN Encoder-Decoder framework with skipgrams or cbow for training, extends the text-based Word2Vec [1] model to learn word embeddings directly from speech. Speech2Vec has access to richer information in the speech signal that does not exist in plain text. In our experiments, the learned word embeddings outperform those produced by Word2Vec from the transcriptions. In the future, we plan to evaluate the word embeddings on speech-related extrinsic tasks such as machine listening comprehension [44, 45] and speech-based visual question answering [46] by initializing the embedding layers of the neural network models. Finally, in this work, some supervision was incorporated into the learning by using forced alignment segmentations as the basis for audio segments. It would be interesting to explore less supervised segmentations to learn word boundaries [47, 48].

5. References

[1] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *NIPS*, 2013.

[2] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017.

[3] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *EMNLP*, 2014.

[4] X. Yu and N. T. Vu, "Character composition model with convolutional neural networks for dependency parsing on morphologically rich languages," in *ACL*, 2017.

[5] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, "Neural architectures for named entity recognition," in *NAACL HLT*, 2016.

[6] B. Plank, A. Søgaard, and Y. Goldberg, "Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss," in *ACL*, 2016.

[7] Y. Kim, Y. Jernite, D. Sontag, and A. M. Rush, "Character-aware neural language models," in *AAAI*, 2016.

[8] M. Ballesteros, C. Dyer, and N. A. Smith, "Improved transition-based parsing by modeling characters instead of words with LSTMs," in *EMNLP*, 2015.

[9] W. He, W. Wang, and K. Livescu, "Multi-view recurrent neural acoustic word embeddings," in *ICLR*, 2017.

[10] S. Settle and K. Livescu, "Discriminative acoustic word embeddings: Recurrent neural network-based approaches," in *SLT*, 2016.

[11] Y.-A. Chung, C.-C. Wu, C.-H. Shen, H.-Y. Lee, and L.-S. Lee, "Audio word2vec: Unsupervised learning of audio segment representations using sequence-to-sequence autoencoder," in *INTERSPEECH*, 2016.

[12] H. Kamper, W. Wang, and K. Livescu, "Deep convolutional acoustic word embeddings using word-pair side information," in *ICASSP*, 2016.

[13] S. Bengio and G. Heigold, "Word embeddings for speech recognition," in *INTERSPEECH*, 2014.

[14] K. Levin, K. Henry, A. Jansen, and K. Livescu, "Fixed-dimensional acoustic embeddings of variable-length segments in low-resource settings," in *ASRU*, 2013.

[15] D. Harwath and J. Glass, "Learning word-like units from joint audio-visual analysis," in *ACL*, 2017.

[16] D. Harwath, A. Torralba, and J. Glass, "Unsupervised learning of spoken language with visual context," in *NIPS*, 2016.

[17] D. Harwath and J. Glass, "Deep multimodal semantic embeddings for speech and images," in *ASRU*, 2015.

[18] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *NIPS*, 2014.

[19] K. Cho, B. van Merriënboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder–decoder for statistical machine translation," in *EMNLP*, 2014.

[20] Y.-A. Chung and J. Glass, "Learning word embeddings from speech," in *NIPS ML4Audio Workshop*, 2017.

[21] T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," in *EMNLP*, 2015.

[22] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *ICLR*, 2014.

[23] S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, and K. Saenko, "Sequence to sequence-video to text," in *CVPR*, 2015.

[24] I. Konstas, S. Iyer, M. Yatskar, Y. Choi, and L. Zettlemoyer, "Neural amr: Sequence-to-sequence models for parsing and generation," in *ACL*, 2017.

[25] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an ASR corpus based on public domain audio books," in *ICASSP*, 2015.

[26] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *NIPS Autodiff Workshop*, 2017.

[27] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[28] S. Subramanian, A. Trischler, Y. Bengio, and C. Pal, "Learning general purpose distributed sentence representations via large scale multi-task learning," in *ICLR*, 2018.

[29] T. Schnabel, I. Labutov, D. Mimno, and T. Joachims, "Evaluation methods for unsupervised word embeddings," in *EMNLP*, 2015.

[30] M. Faruqui and C. Dyer, "Community evaluation and exchange of word vectors at wordvectors.org," in *ACL System Demonstrations*, 2014.

[31] D. Yang and D. M. Powers, "Verb similarity on the taxonomy of wordnet," in *GWC*, 2006.

[32] E. Agirre, E. Alfonseca, K. Hall, J. Kravalova, M. Paşca, and A. Soroa, "A study on similarity and relatedness using distributional and wordnet-based approaches," in *NAACL HLT*, 2009.

[33] G. A. Miller and W. G. Charles, "Contextual correlates of semantic similarity," *Language and Cognitive Processes*, vol. 6, no. 1, pp. 1–28, 1991.

[34] H. Rubenstein and J. B. Goodenough, "Contextual correlates of synonymy," *Communications of the ACM*, vol. 8, no. 10, pp. 627–633, 1965.

[35] M.-T. Luong, R. Socher, and C. D. Manning, "Better word representations with recursive neural networks for morphology," in *CoNLL*, 2013.

[36] E. Bruni, G. Boleda, M. Baroni, and N.-K. Tran, "Distributional semantics in technicolor," in *ACL*, 2012.

[37] K. Radinsky, E. Agichtein, E. Gabrilovich, and S. Markovitch, "A word at a time: computing word relatedness using temporal semantic analysis," in *WWW*, 2011.

[38] G. Halawi, G. Dror, E. Gabrilovich, and Y. Koren, "Large-scale learning of word relatedness with constraints," in *KDD*, 2012.

[39] F. Hill, R. Reichart, and A. Korhonen, "Simlex-999: Evaluating semantic models with (genuine) similarity estimation," *Computational Linguistics*, vol. 41, no. 4, pp. 665–695, 2015.

[40] S. Baker, R. Reichart, and A. Korhonen, "An unsupervised model for instance level subcategorization acquisition," in *EMNLP*, 2014.

[41] D. Gerz, I. Vulić, F. Hill, R. Reichart, and A. Korhonen, "Simverb-3500: A large-scale evaluation set of verb similarity," in *EMNLP*, 2016.

[42] J. L. Myers and A. D. Well, *Research design and statistical analysis*, 1st ed. Routledge, 6 1995.

[43] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[44] Y.-A. Chung, H.-Y. Lee, and J. Glass, "Supervised and unsupervised transfer learning for question answering," in *NAACL HLT*, 2018.

[45] B.-H. Tseng, S.-S. Shen, H.-Y. Lee, and L.-S. Lee, "Towards machine comprehension of spoken content: Initial TOEFL listening comprehension test by machine," in *INTERSPEECH*, 2016.

[46] T. Zhang, D. Dai, T. Tuytelaars, M.-F. Moens, and L. Van Gool, "Speech-based visual question answering," *CoRR*, vol. abs/1705.00464, 2017.

[47] H. Kamper, K. Livescu, and S. Goldwater, "An embedded segmental k-means model for unsupervised segmentation and clustering of speech," in *ASRU*, 2017.

[48] H. Kamper, A. Jansen, and S. Goldwater, "A segmental framework for fully-unsupervised large-vocabulary speech recognition," *Computer Speech and Language*, vol. 46, no. C, pp. 154–174, 2017.