INTERSPEECH 2019
September 15–19, 2019, Graz, Austria

Attention-based word vector prediction with LSTMs and its application to the OOV problem in ASR

Alejandro Coucheiro-Limeres[1], *Fernando Fernández-Martínez*[1], *Rubén San-Segundo*[1], *Javier Ferreiros-López*[1]

[1]Speech Technology Group, Universidad Politécnica de Madrid, Spain

alejandro.coucheiro@upm.es, fernando.fernandezm@upm.es, ruben.sansegundo@upm.es, javier.ferreiros@upm.es

Abstract

We propose three architectures for a word vector prediction system (WVPS) built with LSTMs that consider both past and future contexts of a word for predicting a vector in an embedded space where its surrounding area is semantically related to the considered word. We introduce an attention mechanism in one of the architectures so the system is able to assess the specific contribution of each context word to the prediction. All the architectures are trained under the same conditions and the same training material, following a curricular-learning fashion in the presentation of the data. For the inputs, we employ pretrained word embeddings. We evaluate the systems after the same number of training steps, over two different corpora composed of ground-truth speech transcriptions in Spanish language from TCSTAR and TV recordings used in the Search on Speech Challenge of IberSPEECH 2018. The results show that we are able to reach significant differences between the architectures, consistently across both corpora. The attention-based architecture achieves the best results, suggesting its adequacy for the task. Also, we illustrate the usefulness of the systems for resolving out-of-vocabulary (OOV) regions marked by an ASR system capable of detecting OOV occurrences.

Index Terms: word vector prediction, speech recognition, out-of-vocabulary terms, curricular learning, neural network, language model

1. Introduction

Traditional language models (LMs) predict a word w by employing its past context. This appears natural in tasks like ASR, since the future context of w may not be available in an online decoding. The current maturity of ASR algorithms allows to achieve a good performance in WER terms without the need of taking into account the future context of the words (also thanks to the acoustic evidence that these systems handle in parallel).

However, the future context of a word can contribute notably to its prediction. Due to the extra computational cost involved, we may consider the use of both past and future contexts only to certain words that are difficult to predict. In ASR, we can think of OOV terms as such type of words that could benefit from this two-side prediction. Moreover, instead of predicting a word w, we can predict a vector in a word embedded space which is semantically related to w, without the need of having w in the vocabulary of the system. That is the purpose of a WVPS. Thus, with a WVPS, we find a new way of dealing with the OOV problem in ASR, following the next approach:

- First, a portion of an utterance is decoded by an ASR system, which in principle has a limited vocabulary, so the appearance of OOV terms is possible (typically resulting in errors while substituting each OOV by two or more in-vocabulary (INV) terms).
- Second, some OOV detection algorithm is applied to the decoding result, making use for example of confidence measures or some mechanism that explicitly indicate OOV regions, like those found in Term Discovery (TD) strategies, as in [1] or [2]. This results in the determination of potential OOV regions in the decoded portion of an input utterance.
- Third, a WVPS is employed to carry out a two-side prediction for the previous OOV regions (ignoring the transcribed content inside the OOV regions). A vector in a word embedding space is thus yielded, to which the missing word is semantically related. In the simplest use case, the most appropriate words for each region that exist in the vocabulary should appear as the nearest neighbors of this vector. In our intended use case, the vector would trigger searches in some knowledge sources (e.g. Wikipedia) for collecting word candidates in a similar way as in TD strategies like those found in [1] or [3].
- Finally, a decision mechanism should be employed in order to approve or reject the substitution of the original content of an OOV region by a retrieved OOV term.

In this work, we present the architecture of a WVPS capable of carrying out two-side word vector predictions, based on LSTMs and attention mechanisms, comparing its performance to another two architectures also based in neural networks (NNs). Although our interest for such WVPS resides in the previously described approach for OOV resolution, it can be employed in any task where a two-side word vector prediction might appear useful.

2. Related Work

Traditional LMs are based on n-gram statistics, typically with an n of order 3. The enduring use of these models has lead to numerous improvements through smoothing techniques that are able to leverage the counts of the tokens and also to take into consideration unseen words by setting apart some probability [4].

In fact, n-gram LMs are still used in ASR, specially for the first speech decoding which would lead to a lattice generation. In this field, it is usual that this first LM is constrained to an operative reasonable size, and compiled in a Weighted FST together with the lexicon and a HMM containing the acoustic states related to the acoustic model, as in the Kaldi toolkit [5]. Then, a second, bigger LM would be employed for the rescor-

http://dx.doi.org/10.21437/Interspeech.2019-2347

ing of the lattice, which can be based also on n-grams (perhaps of a higher order) or, more recently, on NNs.

LMs based on NNs have proven to achieve a good performance in perplexity terms [6], and also their suitability for different tasks including ASR [7]. In recent years, multiple variants have been proposed, where various regularization and optimization techniques have also been explored, as shown in [8]. When employing RNNs like LSTMs, the context that these models can regard is arbitrarily large for the same amount of parameters (although usually it is limited to some span), in contrast with FF networks, whose number of parameters increases as longer spans are required. For the input to this kind of LMs, we can employ pretrained word embeddings of the context words instead of an index-codification (like 1-hot) [9].

A recent improvement in the area of NNs is the attention mechanism [10], employed typically in seq2seq tasks, like machine translation [11] or abstractive summarization [12], but also for improving DNN classifiers like those found in more general NLP tasks (e.g. sentiment analysis [13]). Attention allows to focus on different parts of the input sequence, which is useful both for single decision problems (because usually not all the input elements are equally important) and for seq2seq problems (because each of the output elements can be decoded more precisely focussing on different input elements). Machine translation is a clear example where the resulting alignment of both sequences via attention is inherently advantageous.

We find some works that attempt to introduce some attention mechanism into a NN LM (the most similar system to a WVPS). In [14], a cache is added to the network so recent words acquire a higher probability of reappearance, while in [15] a pointer sentinel is appended in order to add the possibility of selecting one of the previous words as the output instead of a prediction.

In our work, the attention mechanism proposed is applied to both sides of the handled word, in contrast to [14] and [15]. Also, the motivation for our framework is to perform a robust estimation of a difficult word vector to predict where the surrounding terms might contribute in different proportions to its insight (like in an OOV resolution scenario), rather than considering whether some recent word is to be repeated at some moment (more oriented to a long-range dependency modeling in LMs).

3. Proposed Architectures

We propose three different architectures of WVPS's that employ the context at both sides of the word to predict. The first one would act as a baseline architecture, due to its simplicity. Then, the second one will present some refinements, while the third one expands the refinements by employing an attention mechanism. Despite their different levels of complexity, they will be designed with a similar number of parameters so their evaluation is comparable, as explained in the next section.

For all the upcoming descriptions, we use the following notation. The letters L and R refers to the left (past) and right (future) context of the word to predict, respectively. We take the same span $C/2$ at both sides, for a total context of C (we made C=20 in all our experiments). The sub-indices of each context word are numbered as their order of entrance to the front LSTMs, so the string of words that could be find in a text when predicting a word w would be $\{L_1, L_2, ..., L_{\frac{C}{2}}, w, R_{\frac{C}{2}}, ..., R_2, R_1\}$.

In the depictions, an LSTM block would receive its inputs from the bottom, in the direction of the arrow drawn inside; the outputs would leave the block from the top; and its inner state might be initialized through one of the sides. The same applies to the Feed-Forward (FF) block, except for the inner state, since they are memory-less.

The inputs to every architecture are the normalized word embeddings associated to the context words L_i and R_j, of dimension d_{embed}. Their output, $\overline{predicted}$, is a normalized vector in the same embedded space, so it can be interpreted as a point in that space with which the surrounding words have some semantic relation, and consequently this point should be close to the embedding of the actual w (the golden standard solution).

The *baseline architecture* is depicted in figure 1. It consists of one LSTM per context side and a FF block of two hidden layers. The FF is fed with the concatenation of the last outputs of the two LSTMs, so each LSTM can contribute with the information extracted from all of its inputs, and in the proper feeding direction towards the word to predict.

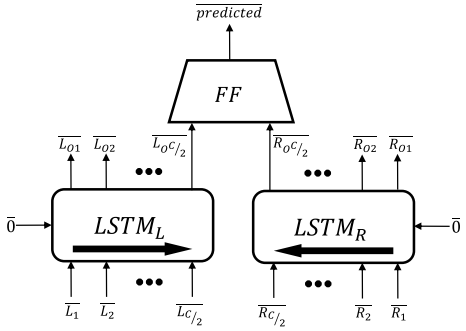

Figure 1: *Baseline architecture of WVPS.*

The second architecture, that we call *stacked architecture*, can be seen in figure 2. We introduced a second layer of LSTMs, one per side of the context, each one receiving the pertinent outputs of the preceding LSTMs. With the purpose of harnessing information gathered by both LSTMs of the first layer, we add their last outputs (i.e. $\overline{L1_{O\frac{C}{2}}} + \overline{R1_{O\frac{C}{2}}}$) for initializing the states of both LSTMs of the second layer. Then, the FF is fed with the concatenation of the last outputs of the top LSTMs.

Finally, the third architecture, or *attention architecture*, is drawn in figure 3. Now, the previous architecture is extended with an attention mechanism, applied to the outputs of the LSTMs of the second layer. In order to generate a query vector that is able to asses the importance of each of these outputs, we make use of an additional FF block, FF_1, fed with the concatenation of the last outputs of the first layer of LSTMs. This way, the query should carry information of both sides of the context, so it can be employed to compute the attention weights of the outputs of both LSTMs of the second layer. These weights are computed with a dot product between the query and each of those outputs, followed by a global softmax transformation of all the dot products, so all the attention weights together total one. As it seems reasonable, the query is also employed to initialize the states of the top LSTMs. When all the attention weights are computed, we perform a weighted sum of the attended vectors to feed the last FF block, FF_2, in charge of producing the output of the network. We changed the feeding direction of the top LSTMs, because, apart from that now the attention mechanism should select the adequate output combi-

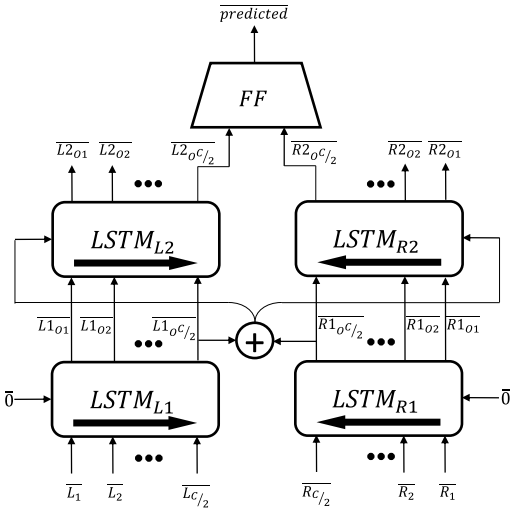

Figure 2: *Stacked architecture of WVPS.*

nation, we are more sure that the information placed in their initial states (i.e. $\overline{query}$) does not vanish when reaching the most immediate context words to the word to predict, which are often more important than the other words.

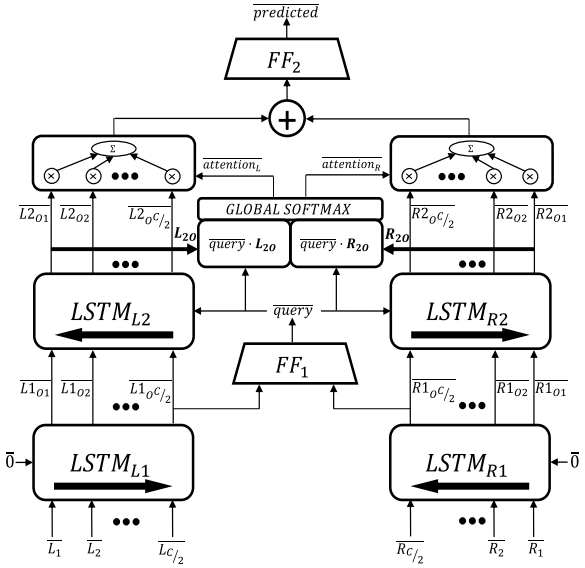

Figure 3: *Attention architecture of WVPS.*

4. Experiments

We conducted a series of experiments to asses the performance of the three architectures proposed. We start by describing the word embeddings that all the architectures employed. Then, we present the training and evaluation materials and schemes followed. Both training and evaluation designs are common among architectures, including a similar number of parameters (about 46M), so their results are comparable.

4.1. Word embeddings

The word embeddings employed were trained following the Skip-gram version of Word2Vec [16], choosing a dimension d_{embed}=100, and contexts of four words at each side. They were kept untouched during WVPS training. The involved entities can be composed of one, two or three words (e.g. *'Madrid'*, *'San Francisco'*, *'old bay seasoning'*, respectively), totaling 248K, 25K and 60K tokens, respectively. They were selected as the most frequent tokens (until some threshold) found in the training texts, described below. For the case of multi-word tokens, we also required that they had their own Wikipedia article, paying attention to discard those entities that are not specially interesting to a general WVPS (e.g. scientific names of plants).

The corpora used for their training belong to different sources in Spanish language: literature, Wikipedia and political records, whose sizes are indicated in table 1 (62M of lines of text in total). The literature set includes novels and essays collected from different public Internet sites. For the Wikipedia articles, we used the dump from the Spanish Wikipedia on date 2017-04-01. The political records are composed by the Spanish portions of the Europarl corpus [17] and the United Nation documents that belong to the OPUS Project Corpora [18]. With such a variety of texts, we aimed to train quite universal word embeddings for the Spanish language, shuffling their lines.

Table 1: *Texts employed for the training of word embeddings. Vocabulary: 248K one-words, 25K two-words, 60K three-words*

Source	# of lines	# of words
Literature	32M	1.2G
Wikipedia (Spanish)	19M	621M
Political records	11M	382M

4.2. Training scheme

The training data for the proposed WVPS's comes from the training material of the word embeddings (table 1). Now, the data presentation to the systems was done in a curricular-learning fashion, for what we performed the next preprocessing:

- First, we trained an *attention architecture* WVPS with the shuffled corpora, for a quarter of an epoch, when a reasonable level of convergence was obtained.
- Second, we computed with the previous system the average cosine similarity of the predicted vectors and the K words of every line *l* in the corpora in vector form:

$$dist_l = \frac{1}{K}\sum_{j=1}^{K}(\overline{w_j} \cdot \overline{predicted}) \quad (1)$$

- Finally, we sorted all the lines by their similarity and we removed the lines with a cosine similarity inferior to 0.2 (mostly very short sentences or with unusual syntax and/or rare terms). We also removed the best 2M lines (similarities close to 1.0), since they were mostly template sentences used in Wikipedia or in political proceedings that do not appear valuable for our purposes.

With this preprocessed text, we firstly presented to our systems the M lines with the best cosine similarity, restarting the process to present again the M best lines plus the next N best ones, in an iterative manner so after each restart we present an

additional bunch of N unseen lines, as in the following progression: $\{M, M+N, M+2N, ...\}$. We made M and N to be both equal to 8M lines.

4.3. Evaluation scheme

We evaluated the systems after a certain amount of training batches were completed for all of them. The material employed for that is composed of speech transcriptions (ground truth) of two different corpora. The first one is the Spanish partition of the TCSTAR corpus [19], from which we took a third of the audio transcriptions (equivalent to 27 hours of speech). The second corpus collects audio recordings (and their human transcriptions) of political programs of the National Spanish TV, between 2015 and 2016, which were used in the Search on Speech Challenge of IberSPEECH 2018 [20]. We also took the transcription of 27 hours of speech. The reason of evaluating over speech transcriptions is to asses the usefulness of the proposed architectures to be used as an OOV resolver in ASR systems.

The evaluation metric employed, $dist_C$, is the average cosine similarity of the predicted vectors and every word in an evaluation corpus C in vector form. It can be computed with equation 1 by considering a corpus as a single long line with K words. Since most of the individual similarities will be comprised between 0 and 1.0, the metric can be interpreted as the average probability that a predicted vector for a word w is in the semantic area of the embedded space that is most related to w.

In order to provide the results with approximated confidence intervals of 95%, we would like to employ the next equation, adapted from chapter 8 of [21]:

$$dist_C\ (95\%) \approx dist_C \pm 1.96 \cdot \sqrt{\frac{dist_C \cdot (1 - dist_C)}{N}} \quad (2)$$

where N is the total number of words in corpus C.

However, to employ this equation, a gaussian distribution is presumed over all the items under evaluation (words in our case). If we study the cosine similarities of the words in an evaluation corpus with any of the systems, we would observe the distribution showed in figure 4, which appears to be gaussian except for the peak on the right, as the dashed line shows (a fitted gaussian to the central part). The peak on the right is due to the stopwords, much easily to predict than normal words. Thus, if we ignore for our evaluation metric the 250 most frequent words in the WVPS's vocabulary, we can get rid of most of this peak and so we can apply equation 2, with an updated value of N (smaller, so the confidence intervals would get wider).

4.4. Results

The evaluation results of the three architectures are presented in table 2, averaging the results of two random initializations per system (though their differences were non-significant). All the systems were evaluated after the same number of training batches (with also the same training data), at the moment in the curricular-learning when $M+3N$ (∼32M) different lines of text were showed and significant differences between the systems were already maintained over time.

As can be seen, all results show significant differences among architectures, preserving this for both evaluation corpora. The *attention architecture* achieves the best performance, with a relative improvement over the *baseline architecture* of 25.6% and 18.2% for the TCSTAR and TV corpora, respectively; and with a relative improvement over the *stacked architecture* of 1.7% and 1.8% for the TCSTAR and TV corpora, respectively. These results confirms our initial intuition about the usefulness of the proposed attention mechanism applied to this scenario.

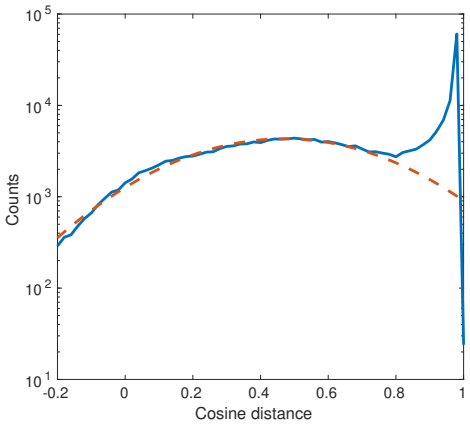

Figure 4: *Histogram of cosine similarities applied to the TCSTAR evaluation corpus, for the attention architecture of WVPS. In dashed line, gaussian approximation of the central part.*

Table 2: *Evaluation results of the WVPS architectures proposed, excluding the 250 most frequent words in the vocabulary*

Architecture	$dist_{TCSTAR}$	$dist_{TV}$
Baseline	0.4043 ± 0.0031	0.3536 ± 0.0028
Stacked	0.4996 ± 0.0031	0.4104 ± 0.0029
Attention	0.5079 ± 0.0031	0.4179 ± 0.0029

The general performance between TCSTAR and TV can be understood due to the characteristics of each corpus. While TCSTAR has a clear political domain with speakers maintaining a strict formality, in the TV we find more spontaneous speech and more variety of topics, complicating the task.

5. Conclusions

The three proposed WVPS architectures have been compared in a realistic scenario involving speech transcriptions of two distinct corpora. The refinements applied to a baseline architecture have demonstrated notable improvements. Specifically, the attention mechanism introduced in the third architecture outperforms the other two with significant differences. Therefore, their interest for a word vector prediction task is evidenced, where we can consider both the past and future contexts of a word, obtaining a vector in an embedded space semantically related to this word. This is specially useful when aiming to predict vectors for difficult words like OOV terms present in the speech decoded by an ASR system, for which the system has to previously detect the pertinent OOV regions.

6. Acknowledgements

The work leading to these results has been supported by AMIC (TIN2017-85854-C4) project (Ministerio de Economía, Industria y Competitividad). We would also like to acknowledge all the other members of the Speech Technology Group at Universidad Politécnica de Madrid for the continuous and fruitful discussion on these topics.

7. References

[1] A. Coucheiro-Limeres, J. Ferreiros-Lopez, R. San-Segundo, and R. Cordoba, "Resource2Vec: Linked Data distributed representations for term discovery in automatic speech recognition," *Expert Systems with Applications*, vol. 112, pp. 301–320, 2018.

[2] C. Parada, M. Dredze, D. Filimonov, and F. Jelinek, "Contextual information improves OOV detection in speech," in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 2010, pp. 216–224.

[3] I. Sheikh, D. Fohr, I. Illina, and G. Linares, "Modelling Semantic Context of OOV Words in Large Vocabulary Continuous Speech Recognition," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 25, no. 3, pp. 598–610, 2017.

[4] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," *Computer Speech & Language*, vol. 13, no. 4, pp. 359–394, 1999.

[5] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The kaldi speech recognition toolkit," IEEE Signal Processing Society, Tech. Rep., 2011.

[6] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of machine learning research*, vol. 3, no. Feb, pp. 1137–1155, 2003.

[7] T. Mikolov, M. Karafiát, L. Burget, J. Černockỳ, and S. Khudanpur, "Recurrent neural network based language model," in *Eleventh annual conference of the international speech communication association*, 2010.

[8] S. Merity, N. S. Keskar, and R. Socher, "An analysis of neural language modeling at multiple scales," *arXiv preprint arXiv:1803.08240*, 2018.

[9] K. Audhkhasi, A. Sethy, and B. Ramabhadran, "Semantic word embedding neural network language models for automatic speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 5995–5999.

[10] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, pp. 5998–6008.

[12] A. Celikyilmaz, A. Bosselut, X. He, and Y. Choi, "Deep communicating agents for abstractive summarization," *arXiv preprint arXiv:1803.10357*, 2018.

[13] P. Chen, Z. Sun, L. Bing, and W. Yang, "Recurrent attention network on memory for aspect sentiment analysis," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 452–461.

[14] E. Grave, A. Joulin, and N. Usunier, "Improving neural language models with a continuous cache," *arXiv preprint arXiv:1612.04426*, 2016.

[15] S. Merity, C. Xiong, J. Bradbury, and R. Socher, "Pointer sentinel mixture models," *arXiv preprint arXiv:1609.07843*, 2016.

[16] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[17] P. Koehn, "Europarl: A parallel corpus for statistical machine translation," in *MT summit*, vol. 5, 2005, pp. 79–86.

[18] J. Tiedemann, "Parallel Data, Tools and Interfaces in OPUS," in *LREC*, vol. 2012, 2012, pp. 2214–2218.

[19] D. Mostefa, O. Hamon, N. Moreau, and K. Choukri, "Evaluation report for the technology and corpora for speech to speech translation. TC-STAR Project. Deliverable N. 30," 2007.

[20] J. Tejedor and D. T. Toledano, "The ALBAYZIN 2018 Search on Speech Evaluation Plan," in *IberSPEECH'18*, 2018.

[21] N. A. Weiss, *Introductory statistics*. Pearson, 2017.