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

Mining Polysemous Triplets with Recurrent Neural Networks for Spoken Language Understanding

Vedran Vukotić[1], *Christian Raymond*[2]

[1]LAMARK / IMATAG, Rennes, France
[2]INSA Rennes & INRIA/IRISA, Rennes, France

name.surname@lamark.fr, name.surname@irisa.fr

Abstract

The typical RNN (Recurrent Neural Network) pipeline in SLU (Spoken Language Understanding), and specifically in the slot-filling task, consists of three stages: word embedding, context window representation, and label prediction. Label prediction, as a classification task, is the one that creates a sensible context window representation during learning through backpropagation. However, due to natural variations of the data, differences in two same-labeled samples can lead to dissimilar representations, whereas similarities in two differently-labeled samples can lead to them having close representations. In computer vision applications, specifically in face recognition and person re-identification, this problem has recently been successfully tackled by introducing data triplets and a triplet loss function.

In SLU, each word can be mapped to one or multiple labels depending on small variations of its context. We exploit this fact to construct data triplets consisting of the same words with different contexts that form a pair of datapoints with matching target labels and an another pair with non-matching labels. By using these triplets and an additional loss function, we update the context window representation in order to improve it, make dissimilar samples more distant and similar samples closer, leading to better classification results and an improved rate of convergence.

Index Terms: spoken language understanding, recurrent neural networks, RNN, triplets, triplet loss, triplet mining, hard triplets, long short-term memory, LSTM, gated recurrent units, GRU, ATIS, SNIPS, MEDIA, deep learning

1. Introduction

In all generality, in typical deep learning applications, samples are often represented in a high-dimensional latent space, where drawing a decision boundary is easier. There, similar samples are often clustered and dissimilar ones reside further apart. However, sometimes samples from different target classes that resemble each-other in the input space may end up being similar in the latent space and samples from the same target class may end up being dissimilar in the latent space due to natural variations in the input space [1]. This is especially true for cases where not many samples are at disposal and there is a high variability in the input space. To address this issue, in the task of face recognition and person re-identification, an architecture consisting of 3 instances of a siamese network [2] (a network containing two or more identical subnetwork components sharing the same weights) was used to perform learning by minimizing a, at the time unusual, triplet loss function [3, 4].

Similarly, introducing a triplet loss function was later shown to be successful for computing text similarity [5] with a character-based bidirectional LSTM network, for learning a metric in speaker diarization [6, 7, 8] with an end-to-end to end architecture, and for recognizing speech emotion [9] by adding it to an LSTM network. In speaker verification, a triplet loss [10, 11] was proven successful on short inputs both with an end-to-end CNN architecture [12, 13] and when used with an Euclidean mining metric on a bidirectional LSTM-based architecture. A triplet loss was also successfully deployed in speaker recogntion [14], spoken language identification [15] and even in multispeaker spech synthesis [16].

In the task of slot filling a similar, but less performing, approach introduced a ranking loss [17] in order to achieve a similar effect. We propose to exploit the polysemy present in the dataset by introducing a triplet loss and a mining strategy to a bidirectional RNN architecture, in order to generate a more discriminative latent representation such that there is a more effective separation of dissimilar samples.

2. Datasets

In the experimental part of this work, three datasets are used: ATIS (Air Travel Information Services), SNIPS and MEDIA (*Méthodologie d'Evaluation automatique de la compréhension hors et en contexte du DIAlogue* - evaluation of man-machine dialogue systems). These three datasets vary in their language, complexity, size and degree of exploitability of the polysemous nature of the words found therein. In ATIS and MEDIA, the inputs that we use consists of three values: a word, a class to which the word belongs — if feasible, and a target label. Words are the original words, as transcribed from the source. Word classes are database entries of known words that belong to the same cluster. (To illustrate words that are undoubtedly a city name, e.g. *"London"*, *"Paris"* and so on, can form a cluster named *"CITY"* that replaces them). Finally, target labels are the labels to which the words belong to, given their current context. SNIPS, however, consists solely of words and labels.

2.1. ATIS

The ATIS [18] datasets is dedicated to systems providing flight information. It consists of a training set of 4978 utterances, a testing set of 893 utterances and contains a total of 1117 possible words. Polysemy-wise, ATIS is not very complex. Its average number of possible labels per word is only 1.41, with 860 words having the median value of only one possible label. Words having the highest number of possible labels are cities, typically with 5 possible labels (*"null"*, *"city_name"*, and *"toloc.city_name"*, *"fromloc.city_name"*, *"stoploc.city_name"*), with the only exception of *"Washington"* that has 6 possible labels, since it can also be a state (*"toloc.state_name"*).

Figure 1 illustrates the histogram of possible labels per

http://dx.doi.org/10.21437/Interspeech.2019-2977

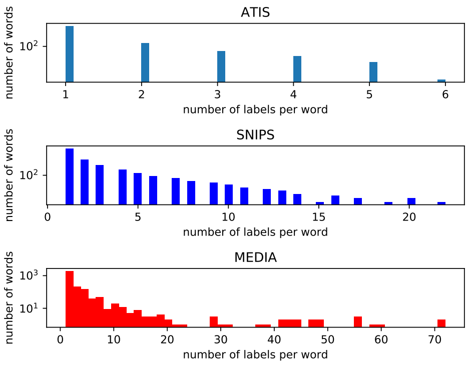

Figure 1: *Histograms illustrating the distribution of the number of possible labels for each word (on a logarithmic scale) for the ATIS, SNIPS and MEDIA datasets.*

word. For ATIS, it also indicates the simplicity of the dataset and the low amount of polysemy found therein. This is a potentially compelling observation reinforcing the arguments regarding the excessive simplicity of the dataset, already postulated by the authors of [19] and [20].

2.2. SNIPS

SNIPS [21] is a new dataset developed as part of the *Snips Voice Platform*, a framework focusing on SLU on IoT devices. It consist of a training and a testing set with a total of 12582 different words and 72 possible target labels, distributed within 13784 sentences present in the training set and 700 sentences of the testing set. Regarding polysemy, words have from 1 to 22 possible labels, with the average number of 1.48 possible labels per word. 9853 words have the median value of only one possible label and 9 words ("in", "my", "of", "the", *etc.*) have more than 15 labels.

2.3. MEDIA

The MEDIA [22] dataset was created by ELDA following a *Wizard of Oz Protocol* where a total of 250 speakers went through 5 different hotel reservation scenarios each. It consists of a training set with 720 dialogues (12K messages), a development / cross-validation set of 200 dialogues (3K messages) and a testing set with 79 dialogues (3K messages). The dataset consists of a total of 2428 possible words and a total of 138 possible labels. Polysemy-wise, words in MEDIA have an average of 2.55 possible labels, varying from the median of only one label up to a total of 72 possible labels per word, and a vast portion of available words have up to 20 possible labels, as nicely illustrated by the histogram in Figure 1. The words with an unusually high number of possible labels (more than 50) are: *"la"*, *"de"*, *"à"*, *"pour"*, *"le"*, *"un"*, and *"les"*, while 1418 words have the median value of only one possible label.

3. Polysemous Triplets and Mining

In the context of spoken language understanding and specifically slot tagging, we define polysemy as the coexistence of many possible target labels for a given word within the dataset. We do not analyze the general coexistence of many possible meanings of each world and we are only interested in the possible labels occurring within the dataset. More specifically, only the labels occurring for a given word (or word class) within the training set of the dataset are taken into consideration. Given any word w_i occurring within a phrase in the dataset, we define its target label y_i and its context window (of size c), that forms an input sample $x_i = \left\{ w_i^{n-\lfloor \frac{c}{2} \rfloor}, \cdots, w_i^{n-1}, w_i^{n}, w_i^{n+1}, \cdots, w_i^{n+\lfloor \frac{c}{2} \rfloor} \right\}$.

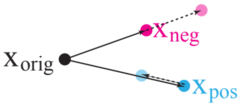

Figure 2: *The triplet loss aims at altering the representation space to make the representation of same-labeled samples more similar and the representation of different-labeled samples more dissimilar.*

A triplet is then defined as a set of 3 input samples x_{orig}, x_{pos} and x_{neg} (a random original sample that is treated as anchor, a positive sample, and a negative sample) such as that all three of them have the same central word ($w_{orig}^n = w_{pos}^n = w_{neg}^n$) but only two of them have the same target label ($y_{orig} = y_{pos}$) while the third one has a different target label ($y_{orig} \neq y_{pos}$ and, by transitivity, $y_{pos} \neq y_{neg}$).

In practice, it is only possible to form triplets for words that have at least two possible target labels occurring within the dataset and at least two different sentences (context windows) per label. (Although it would be possible to use labels with only one sentence as negative samples, we keep the dataset balanced and impose a minimum requirement of at least two sample sentences per label.) This makes the number of available sentences / context windows smaller but, given the number of combinations of possible target labels per word and the number of combination of sentences / possible context windows per target label, it does not have a negative impact.

3.1. Triplet Loss

The seminal idea of a triplet loss is to make the representations of inputs that lead to the same target label similar (even if they are initially dissimilar due to very different context windows surrounding the same central word) and representations of inputs that lead to different target labels dissimilar (even if they are originally similar do to very similar context windows surrounding the same central word), as illustrated in Figure 2.

When training a classification network, the neural architecture consists of a word embedding layer that takes one input consisting of a context window with c words. The output of the embedding layer is then given to a bidirectional GRU (Gated Recurrent Unit) layer [23] (or any other RNN network for that matter). This is then followed by a fully-connected dense layer and an activation layer to predict the classes of the given central word of the context window, as illustrated in the top part of Figure 3. Learning is performed by minimizing a classification loss such as categorical cross-entropy.

Training the network with a triplet loss reuses the first part of the classification architecture, namely the embedding and bidirectional GRU layers that form a siamese network sharing the architecture and weights across all its instances. Three instances are used, one for each element of the input triplet. This time, a loss is attached directly to the output of the bidirectional GRU networks that represent the context windows representation space. Here, the loss will alter directly the representation space by making the representations of x_{orig} and x_{pos} more

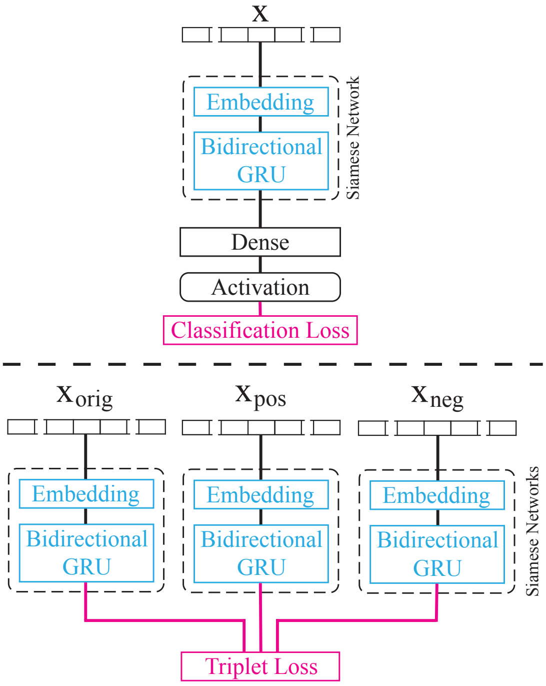

Figure 3: *Siamese networks consisting of an embedding and a bidirectional GRU layer shared across the architecture. One instance is used to perform and learn classification (top image) and three instances, with no additional layers, are used when training with a triplet loss.*

similar and the representations of x_{orig} and x_{neg} more dissimilar. Formally, the triplet loss function is defined as follows:

$$L(x_o, x_p, x_n) = \sum_{i}^{N} max\left\{\|f(x_{o_i}) - f(x_{p_i})\|^2 - \|f(x_{o_i}) - f(x_{n_i})\|^2 + \alpha, 0\right\}$$

Let $f(x)$ indicate the output of a siamese network for a given input x. The equation minimizes the distance between the representations of x_{orig} and x_{pos} ($f(x_{orig})$ and $f(x_{pos})$ respectively) and maximizes the distance between the representations of x_{orig} and x_{neg} ($f(x_{orig})$ and $f(x_{neg})$ respectively). A scalar, α is introduced to enforce a minimal margin between the two representations and, to prevent accidental negative values for the loss function, all the values below zero are removed. The loss is then aggregated over all the N samples of the current input batch.

Although the target labels are not directly presented during training, they are implicitly given to the network through the selection of input triplets, thus making learning with a triplet loss fall still under the supervised learning category. It is also important to note that both losses have to be minimized during training (both architectures have to be trained) since the final fully-connected dense layer of the classifier would remain uninitialized if only the triplet loss is minimized. Additionally, the dense layer of the classifier needs to be updated and fine-tuned as the representation outputted by the siamese networks changes over time.

3.2. Mining Strategies

When forming a batch of triplets for training, it is possible to use different strategies to select the labels and their corresponding input context windows for each of the selected labels. The simplest approach would be to simply randomly select two labels (out of all the possible labels) for each word and then select two inputs that support the chosen labels for the given word. We denote this as a random mining strategy, although it does not really perform any mining and it consists solely of a random selection of appropriate datapoints to form triplets. This makes it the fastest selection method.

Instead of selecting samples randomly to create triplets, it is possible to first compute a similarity between all the possible labels for a given word and select the most similar samples (that given that they support different labels should instead be dissimilar). We refer to this method as hard triplet mining. Given that computing the similarity of all the pairs of possible target labels for each word can be quite expensive, this can be performed once every epoch or less frequently. After the similarities between all the possible label pairs for each word are computed, only the top most difficult can be kept. Additionally different similarity measures can be used for mining hard triplets. In this work we evaluate both Euclidean and cosine distances as possible similarity measures for hard triplet selection, as they are both compatible with the previously stated triplet loss function.

4. Experimental Evaluation

All the experiments, regardless of the loss function used, were performed with a context window of 11 words (5 words before and 5 words after the current word), a word embedding space with a dimensionality of 200, a batch size of 32 and a dropout following the siamese networks of 50%. Different context window sizes and embedding sizes either decreased the performance or did not further improve it [24]. Implementation-wise, the described architectures were implemented with the *Keras* [25] API of the *TensorFlow* framework and each experiment was run 4 times in order to compute averaged results and their respective standard deviations, as illustrated in Table 1. When mining hard triplets, after computing the similarity scores of all the possible different label combinations for the given words, only the top 2% “hardest” (most similar ones that should be dissimilar) were kept and made available for selection during the formations of the batches.

4.1. Overall Performance

Table 1 illustrates the averaged results obtained over multiple runs of each experimental setup and their respective standard deviations. When comparing the different setups we use a single-sided t-test to determine the intervals of significance of our hypotheses.

4.1.1. ATIS

Introducing a triplet loss and hard mining during training outperforms training performed solely with a classification loss with 95.66% (Euclidean distance) and 95.58% (cosine distance), compared to 95.53% in F1. This can be stated with a significance of 90% and 80% respectively, according to a single-sided t-test. Although hard mining with an Euclidean distance appears to outperform hard mining with a cosine distance, which then outperforms random mining — on ATIS, this can be stated only with 70% significance.

Table 1: *Performances of the various recurrent architectures on ATIS, SNIPS and MEDIA: averaged (over 4 runs) accuracy, precision, recall, F1 measure (%) and their respective standard deviations (in parenthesis) as computed by the conlleval evaluation tool. The levels of significance, for each pair compared, are discussed within the text.*

Strategy	Mining Sim. Measure	Accuracy	Precision	Recall	F1
ATIS					
classification only	-	98.00 (0.06)	94.86 (0.15)	96.21 (0.19)	95.53 (0.17)
class. & triplet - random mining	-	98.02 (0.08)	94.99 (0.24)	96.19 (0.21)	95.58 (0.19)
class. & triplet - hard mining	**Euclidean**	98.07 (0.04)	94.99 (0.15)	96.34 (0.12)	**95.66** (0.13)
class. & triplet - hard mining	cosine	98.06 (0.04)	94.99 (0.10)	96.25 (0.14)	95.61 (0.10)
SNIPS					
classification only	-	94.22 (0.19)	84.54 (0.60)	89.62 (0.52)	87.00 (0.41)
class. & triplet - random mining	-	94.63 (0.18)	85.16 (0.31)	90.18 (0.37)	87.59 (0.31)
class. & triplet - hard mining	**Euclidean**	94.59 (0.41)	85.60 (0.65)	90.57 (0.57)	**88.01** (0.56)
class. & triplet - hard mining	cosine	94.68 (0.23)	85.36 (0.47)	90.55 (0.47)	87.88 (0.25)
MEDIA					
classification only	-	88.81 (0.09)	82.93 (0.42)	84.34 (0.33)	83.63 (0.16)
class. & triplet - random mining	-	89.19 (0.11)	82.76 (0.34)	85.16 (0.26)	83.94 (0.06)
class. & triplet - hard mining	**Euclidean**	89.08 (0.18)	82.83 (0.55)	85.39 (0.52)	**84.08** (0.32)
class. & triplet - hard mining	cosine	89.00 (0.06)	83.36 (1.43)	85.26 (0.26)	84.01 (0.19)

4.1.2. SNIPS

A simple classification loss based training achieves 87.00% in terms of F1, which is comparable to the performance of *Snips NLU* [21]. All triplet loss based evaluations outperform that with 87.59% (for random triplet selection), 87.88% (for hard mining with a cosine distance) and finally, with 88.01% (for hard triplet mining with an Euclidean distance). The significances of those statements are 95%, 98% and 99% respectively, according to a single-sided t-test.

4.1.3. MEDIA

The empirical results show that introducing a triplet loss is beneficial and outperforms (regardless of the mining method used) training that is performed only with a classification loss with a $p = 98\%$ interval of significance. Strictly speaking, the best performing method appears to be the variant with a triplet loss and hard mining with an Euclidean similarity measure that achieves 84.08% in F1, compared to the 83.63% of a classification only architecture. When comparing solely the hard mining similarity measures, no significant hypothesis can be accepted and all the method perform equally well from a realistic standpoint. Nevertheless, with an slightly less-acceptable significance of $p = 86\%$, we can state that hard mining methods outperform random mining in terms of F1 score.

4.2. Rate of Convergence

The rate of convergence (cross-validation on the development set of the MEDIA dataset) during training is illustrated on Figure 4. Clearly, introducing a triplet loss increases the speed of convergence significantly (75.97% in F1 after only one epoch of training when a triplet loss and hard mining are introduced, compared to 20.78% after one epoch when using exclusively a classification loss). However, it is worth noting that the difference in final performance is small and not very significant (75.08% in F1 for hard mining vs 73.08% for random mining after one epoch compared to 84.08% vs 83.94% for the final F1 score) thus making the choice of the mining strategy a cost-benefit based one. The same boost in convergence speed was present in ATIS (82.50% *vs.* 66.40% after the first epoch) and SNIPS (82.88% *vs.* 5.42% after the first epoch).

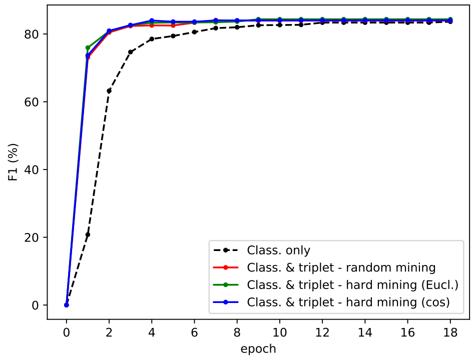

Figure 4: *Rate of convergence on MEDIA (validation set). Introducing a triplet loss improves the speed of convergence and the overall performance in comparison to using only a classification loss. However, there is only a small difference in the final F1 score between the different mining methods thus making the choice of a mining strategy a cost-benefit based one.*

5. Conclusions

We proposed the introduction of a triplet loss, in addition to the classification loss, in recurrent neural network approaches to solving the SLU task of slot filling, and we evaluated different triplet mining techniques on three datasets: ATIS, SNIPS and MEDIA. The experimental evaluation clearly demonstrated that introducing a triplet loss significantly ($p = 98\%$ according to a single-sided t-test of significance) improves the classification performance of recurrent neural networks and increases their convergence speed consistently on all datasets. Moreover, we showed that hard triplet mining additionally improves the performance (with an Euclidean distance based hard triplet mining as the best performing strategy), although the improvement significance is slightly smaller so the choice of its use can thus be formulated through a cost-benefit analysis between the improved performance and slightly increased overhead of selecting hard triplets as compared to the random mining strategy.

6. References

[1] M. Dinarelli, V. Vukotic, and C. Raymond, “Label-dependency coding in simple recurrent networks for spoken language understanding,” in *Interspeech*, 2017.

[2] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, “Signature verification using a” siamese” time delay neural network,” in *Advances in neural information processing systems*, 1994, pp. 737–744.

[3] F. Schroff, D. Kalenichenko, and J. Philbin, “Facenet: A unified embedding for face recognition and clustering,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.

[4] A. Hermans, L. Beyer, and B. Leibe, “In defense of the triplet loss for person re-identification,” *arXiv preprint arXiv:1703.07737*, 2017.

[5] P. Neculoiu, M. Versteegh, and M. Rotaru, “Learning text similarity with siamese recurrent networks,” in *Proceedings of the 1st Workshop on Representation Learning for NLP*, 2016, pp. 148–157.

[6] H. Song, M. Willi, J. J. Thiagarajan, V. Berisha, and A. Spanias, “Triplet network with attention for speaker diarization,” *arXiv preprint arXiv:1808.01535*, 2018.

[7] R. Yin, H. Bredin, and C. Barras, “Neural speech turn segmentation and affinity propagation for speaker diarization,” in *Annual Conference of the International Speech Communication Association*, 2018.

[8] G. Le Lan, D. Charlet, A. Larcher, and S. Meignier, “A triplet ranking-based neural network for speaker diarization and linking.” in *INTERSPEECH*, 2017, pp. 3572–3576.

[9] J. Huang, Y. Li, J. Tao, and Z. Lian, “Speech emotion recognition from variable-length inputs with triplet loss function,” in *Proc. Interspeech 2018*, 2018, pp. 3673–3677. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2018-1432

[10] L. Wan, Q. Wang, A. Papir, and I. L. Moreno, “Generalized end-to-end loss for speaker verification,” in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4879–4883.

[11] W. Ding and L. He, “Mtgan: Speaker verification through multitasking triplet generative adversarial networks,” *arXiv preprint arXiv:1803.09059*, 2018.

[12] C. Zhang and K. Koishida, “End-to-end text-independent speaker verification with triplet loss on short utterances,” in *Proc. Interspeech 2017*, 2017, pp. 1487–1491. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2017-1608

[13] C. Zhang, K. Koishida, and J. H. Hansen, “Text-independent speaker verification based on triplet convolutional neural network embeddings,” *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 26, no. 9, pp. 1633–1644, 2018.

[14] S. Novoselov, V. Shchemelinin, A. Shulipa, A. Kozlov, and I. Kremnev, “Triplet loss based cosine similarity metric learning for text-independent speaker recognition,” *Proc. Interspeech 2018*, pp. 2242–2246, 2018.

[15] G. Gelly and J.-L. Gauvain, “Spoken language identification using lstm-based angular proximity.” in *INTERSPEECH*, 2017, pp. 2566–2570.

[16] Y. Jia, Y. Zhang, R. Weiss, Q. Wang, J. Shen, F. Ren, P. Nguyen, R. Pang, I. L. Moreno, Y. Wu *et al.*, “Transfer learning from speaker verification to multispeaker text-to-speech synthesis,” in *Advances in Neural Information Processing Systems*, 2018, pp. 4480–4490.

[17] N. T. Vu, P. Gupta, H. Adel, and H. Schütze, “Bi-directional recurrent neural network with ranking loss for spoken language understanding,” in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Ieee, 2016, pp. 6060–6064.

[18] D. A. Dahl, M. Bates, M. Brown, W. Fisher, K. Hunicke-Smith, D. Pallett, C. Pao, A. Rudnicky, and E. Shriberg, “Expanding the scope of the ATIS task: the ATIS-3 corpus,” in *HLT*, 1994, pp. 43–48.

[19] F. Béchet and C. Raymond, “Is ATIS too shallow to go deeper for benchmarking Spoken Language Understanding models?” in *InterSpeech 2018*, Hyderabad, India, Sep. 2018, pp. 1–5. [Online]. Available: https://hal.inria.fr/hal-01835425

[20] V. Vukotić, C. Raymond, and G. Gravier, “Is it time to switch to Word Embedding and Recurrent Neural Networks for Spoken Language Understanding?” in *InterSpeech*, Dresden, Germany, Sep. 2015. [Online]. Available: https://hal.inria.fr/hal-01196915

[21] A. Coucke, A. Saade, A. Ball, T. Bluche, A. Caulier, D. Leroy, C. Doumouro, T. Gisselbrecht, F. Caltagirone, T. Lavril *et al.*, “Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces,” *arXiv preprint arXiv:1805.10190*, pp. 12–16, 2018.

[22] H. Bonneau-Maynard, S. Rosset, C. Ayache, A. Kuhn, and D. Mostefa, “Semantic Annotation of the French Media Dialog Corpus,” in *InterSpeech*, Lisbon, September 2005.

[23] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using RNN encoder-decoder for statistical machine translation,” *arXiv preprint arXiv:1406.1078*, 2014.

[24] V. Vukotić, C. Raymond, and G. Gravier, “A step beyond local observations with a dialog aware bidirectional gru network for spoken language understanding,” in *Interspeech*, 2016.

[25] F. Chollet, “Keras,” https://github.com/fchollet/keras, 2015.