Exploring the use of Attention-Based Recurrent Neural Networks For Spoken Language Understanding

Edwin Simonnet **Paul Deléglise** **Nathalie Camelin** **Yannick Estève**

LIUM - University of Le Mans, France
firstname.lastname@univ-lemans.fr

Abstract

This study explores the use of a bidirectional recurrent neural network (RNN) encoder/decoder based on a mechanism of attention for a Spoken Language Understanding (SLU) task. First experiments carried on the ATIS corpus confirm the quality of the RNN baseline system used in this paper, by comparing its results on the ATIS corpus to the results recently published in the literature. Additional experiments show that RNN based on a mechanism of attention performs better than RNN architectures recently proposed for a slot filling task. On the French MEDIA corpus, a French state-of-the-art corpus for SLU dedicated to hotel reservation and tourist information, experiments show that a bidirectionnal RNN reaches a f-measure value of 79.51 while the use of a mechanism of attention allows us to reach a f-measure value of 80.27.

1 Introduction

Spoken Language Understanding (SLU) can be defined as the interpretation of signs conveyed by a speech signal [1]. This interpretation is usually understood as the extraction and the representation of the *meanings* supported by the words within an uttered sentence.

1.1 Slot filling task

Nowadays, extracting meaning from speech is still a very complex process and, for an application, SLU is often reduced to the construction of a task-specific semantic representation. This representation consists classically in the use of *frames* describing general concepts and their specific instances. A frame is composed by a data structure which represents a predefined concept by associating to the concept name a set of roles: these roles are represented by *slots*.

In this framework, SLU corresponds to a slot filling task and, classically, this task can be defined as a concept tagging process, which is the extraction of a sequence of concepts out of a given word sequence [2]. In the past, several sequence tagging methods have been proposed to extract such sequences of concepts and Conditional Random Fields [3] (CRFs) were considered as the state-of-the-art approach [2] until two years ago.

1.2 Purpose

Recently, it was shown in [4, 5] that recurrent neural networks (RNNs) could get better performances for a SLU slot filling task than CRFs. These works were conducted on the ATIS corpus [6] but were

not confirmed by a very recent work presented in [7] made on the MEDIA corpus [8]: in this last study, CRF got significantly better results than RNN. This may be explained by the fact that the MEDIA task seems more difficult to process than the ATIS one, while the size of the vocabulary and the proportion of words in the corpus having a concept are greater in the MEDIA corpus than in the ATIS one.

In this paper, we do not try to arbitrate if CRF or deep neural network (DNN) architectures are the current state-of-the-art for SLU. We are convinced of the potentials of DNN architectures and we aim to explore the use of attention-based recurrent neural networks [9] initially dedicated for handwriting recognition and successfully used for speech recognition [10]. Since the MEDIA corpus seems more challenging for SLU than the ATIS corpus, we have focused on this corpus to evaluate the consequences for the SLU task of the use of the attention-based mechanism proposed in [11].

1.3 Attention-based RNN general principles

Very good descriptions of the principles of attention-based RNNs can be found in [11, 10, 12]. The attention mechanism was intuitively designed in order to take care about the positions of input elements when encoding an input sequence in an RNN encoder-decoder approach. In this paper, the attention-based RNN was largely inspired from the architecture proposed in [11] for machine translation, depicted in figure 1 : we consider the concept tagging process as a translation problem from words (source language) to semantic concepts (target language).

This architecture is based on a bidirectional RNN as an encoder. This bidirectional RNN computes an annotation h_i for each word w_i from the input sequence $\{w_1, ... ,w_I\}$. This annotation is the concatenation of the matching forward hidden layer state and the backward hidden layer state obtained respectively by the forward RNN and the backward RNN comprising the bidirectional RNN. Each annotation contains the summaries of both the preceding words and the following words. Since hidden layers of RNNs tend to better represent recent inputs, each annotation h_i will be focused on the words around w_i.

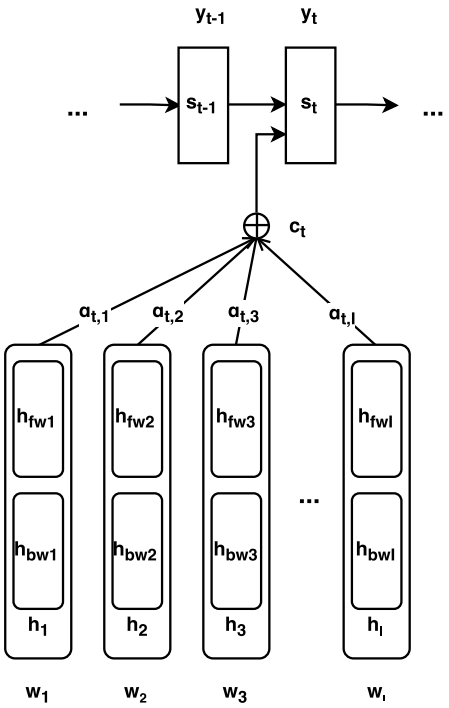

Figure 1: Illustration of Attention-based RNN from [11]

As a result of applying the bidirectional encoder, for each word within the input sequence an annotation is computed: this sequence of annotations $\{h_1, ... ,h_I\}$ will be used by the decoder to compute a context vector c_t. A context vector is recomputed after each emission of an output label. This computation takes into account a weighted sum of all the annotations computed by the encoder. This weighting depends on the current output target, and is the core of the attention mechanism: a good estimation of these weights α_{ti} allows the decoder to choose parts of the input sequence to pay attention to. This context vector will be used by the decoder in conjunction with the previous emitted label output y_{t-1} and the current state s_t of the hidden layer of a RNN to take a decision about the current label output y_t.

2 Implementation

In order to compare RNN and attention-based RNN for an SLU task, implementations provided by the authors of [4] and [11] were used to make our experiments. The RNN implementation coming from [4] was not entirely the one used to make their experiments: only a forward RNN implementation was available while this study used a bidirectional neural network. In order to validate our implementation of this bidirectional neural network, first experiments were carried on the ATIS corpus to compare our results with the ones presented in [4].

2.1 RNN implementation

The RNN implementation is based on [4] and on the implementation also proposed by the first author in [13] which gives an implementation of an Elman/Jordan-type forward RNN with T (previous time steps vectors from the output layer as in the Jordan-type network or the hidden layer as in the Elman-type network) fixed to 1. From this implementation, we used the Elman-type RNN (the operation would be the same for a Jordan except the output layer is given back to the hidden layer at t+1).

$$hidden\ layer : h(t) = sigmoid(W_x.x(t) + W_h.h(t-1) + b_h))$$
$$output\ layer : s(t) = softmax(W.h(t) + b))$$

where $x(t)$ is the input word of the RNN (embedding) at t and $h(t-1)$ the output of the hidden layer at $t-1$. The parameters of the RNN are W_x, W_h and W the weight matrices, b_h and b the bias, and h_0 the initial hidden layer of the last step for the first word of the sentence for which nothing has been calculated yet. The parameters are adjusted through training epochs with a gradient descent performed on mini-batches.

The backward version was implemented from the forward distributed in [13]. A backward RNN is similar to a forward one except that the prediction is from the future to the past. The sentence is given backward to simulate this. W_h represents the weight matrix between the next step hidden layer and the current one. h_0 is the initial hidden layer of the next step for the last word of the sentence (*i.e.* first word given to the RNN).

$$hidden\ layer : h(t) = sigmoid(W_x.x(t) + W_h.h(t+1) + b_h))$$
$$output\ layer : s(t) = softmax(W.h(t) + b))$$

With a backward RNN acquired, the bidirectional one can be implemented. The bidirectional RNN makes predictions taking into account the past (as a forward) and the future (as a backward). Therefore already trained RNNs forward and backward are used jointly. There are two W_h weight matrices: W_h_fw between the last step hidden layer and the current one and W_h_bw between the next step hidden layer and the current one. It goes the same for b_h_fw et b_h_bw. Finally there is no initial hidden layer h_0 since those are recovered from the already trained forward and backward RNNs.

$$hidden\ layer : h(t) = sigmoid(W_x.x(t) + W_h\_fw.h(t-1) + b_h\_fw + W_h\_bw.h(t+1) + b_h\_bw)$$
$$output\ layer : s(t) = softmax(W.h(t) + b)$$

Our objective was to implement long-term dependencies as described in [4] by feeding the network with the sum of the previous/next steps:

$$h_{bidirectional}(t) = f(W_x.x(t)+\sum_{k=1}^{T}(W_h\_bw_k.h\_bw(t+k)+b_h\_bw)+\sum_{k=1}^{T}(W_h\_fw_k.h\_fw(t-k)+b_h\_fw))$$

[13] give an implementation with T fixed to 1. Different values of T were experimented on the ATIS corpus for the forward and backward RNNs to see if adding context improves the system, but bidirectional RNN reached better results with a forward and backward RNNs having both T=1.

The forward and backward RNNs used for the bidirectional training or classification are trained beforehand. We experimented with different ways to train those RNNs. First, the *parallel train* learning, which consists on training forward, backward, and then bidirectional RNNs at each epoch. Second, the *get best* learning, in which the training of the bidirectional RNN is based on the best parameters of both forward and backward RNNs already trained separately before. Last, the *train best* learning which combines the approaches described above: at each epoch the forward and the backward RNNs are trained as in the *parallel train*. Then the bidirectional RNN uses only the parameters of the last best epoch for the forward and the backward respectively as in *get best*.

Experiments have shown, that the best training is *parallel train* approach, followed by *train best* and *get best*. This might, be because the bidirectional RNN learn more with forward and backward RNNs, which have a variability through epochs even if they do not always give the best results. The learning is more diversified. Indeed the *get best* approach using forward and backward parameters fixed from their best epoch is the one giving the worst results.

2.2 Attention-based bidirectional RNN implementation

The attention-based RNN implementation used in our study was derived from the one used in [11] and available at `https://github.com/kyunghyuncho/GroundHog`. This implementation was made for a machine translation task. In this task, input and output sequences have often different lengths. The RNN encoder-decoder approach is particularly well-fitted for such a case. For the focused SLU task, it is important to get a very precise alignment between inputs (words) and outputs (semantic concepts labeling words). To get a so precise alignment, we have modified the decoder process of the bidirectional RNN in order to enforce an output label sequence to get the same length of the input word sequence. This is the only modification we made on the implementation coming from [11].

3 Experiments

As seen before, in order to validate our implementation of this bidirectional neural network, first experiments were carried on the ATIS corpus to be able to compare our results with the ones presented in [4]. At last, we will compare the RNN approach and the attention-based RNN approach on the MEDIA corpus.

3.1 Validation of the RNN implementation on the ATIS corpus

The corpus used by [4] is the ATIS corpus (Airline Travel Information System) specialized in airline ticket reservation requests. It is composed of 4978/893 (learning/testing) annotated sentences. There are 128 semantic labels. The training corpus is divided as follows: 80% for learning and 20% for validation.

In order to help the classifier to delimit the sequences of words having the same label, a common way is to attach a $B/I/O$ suffix to the semantic labels, respectively for *Beginning*, *Inside* and *Outside* of a sequence, as an added information. Only the B and I suffixes are used here. O is represented by the $NULL$ label which is associated to words that do not convey any semantic information within the specific task.

The evaluation is made by computing the f-measure which uses recall and precision metrics to calculate a score taking into account the presence or the absence of a concept in a sentence without the notion of sequentiality.

$$f - measure = \frac{2(precision.recall)}{precision + recall}$$

In [4] precision and the recall are defined[1] as:

$$recall = \frac{number\ of\ correct\ segments}{number\ of\ segments\ in\ the\ reference}$$

$$precision = \frac{number\ of\ correct\ segments}{number\ of\ segments\ in\ the\ hypothesis}$$

A segment of concepts is correct if it begins and ends with the same words for the hypothesis and the reference. The f-measure is maximized on the validation corpus during the training process.

Table 1 presents the results obtained in [4] and the results obtained by our implementation used by using the following hyper-parameters: number of epochs=100 ; window=5 ; unit number in the hidden layer=200 ; embeddings dimension=50.

Experience	Architecture	Type	f-measure
[Mesnil et al. 2013]	Jordan	bidirectional	93.98
RNN baseline	Elman	bidirectional	94.13

Table 1: Comparison between the performances of the RNN presented in [4] and our implementation of a RNN baseline on the ATIS corpus.

As shown in Table 1, our bidirectional RNN baseline system reaches similar results to the bidirectional RNN presented in [4]. This validates our implementation and allow us to explore the use for a SLU slot filling task of an attention-based RNN in comparison to a classical RNN.

3.2 Comparison between RNN and attention-based RNN performances on the MEDIA corpus

The MEDIA corpus [8] is a French state-of-the-art dialog corpus. It contains 1257 dialogs between users and a simulated system (Wizard of Oz protocol) in the domain of hotel reservation and tourist information. Only the user's turns are considered for the training and the classification. This set of turns is divided into three corpora : the TRAIN set contains 17,6k utterances, the DEV set contains 1,3k utterances and finally the TEST set is composed of 3,5k utterances.

Each utterance has been manually transcribed and annotated according to 74 concept labels from general simple response (*e.g.* the word "yes" is associated with the concept *response*) to specific-task requirements (*e.g.* the words "with bath" are associated with the concept *room_equipment*). A much richer annotation is available in MEDIA including modals, specifiers and values but as a first step we choose to only evaluate semantic concept labels.

In MEDIA, the purpose of the dialog for the user is to obtain information that is stored in a database. As a consequence, names of the streets, cities or hotels, lists of room equipments, food type, *etc.* are known. Furthemore, more general words representing figures, days, months are also known. All these words (specific to the SLU task or general) have been gathered in a semantic lexicon that enables to associate a word to a semantic class.

A user's utterance is represented by the sequence of words and semantic classes. If it exists, the word is substituted by its semantic class. An example of the utterance representation is shown in Table 2. As in ATIS, suffixes I/O are associated to label concepts.

Table 3 presents the performances measured in terms of f-measure of different RNN architectures on the MEDIA corpus.

[1] It is calculated with the conlleval.pl script provided by [13]

Words	please I would like to book a hotel for the **first three** days of **May** in **Marseille**.
Words+Sem. Class	please I would like to book a hotel for the **ORDINAL UNIT** days of **MONTH** in **CITY**.

Table 2: Representation of a user's utterance by words or words+semantic classes.

Architecture	f-measure
RNN forward	74.04
RNN backward	77.42
RNN bidirectional	79.51
Attention-based mechanism	80.27
RNN Encoder-Decoder without attention mechanism	38.25

Table 3: Results on MEDIA with semantic classes

As expected, the results are not as good as on the ATIS corpus. The bidirectional RNN architecture gets better results in comparison to a backward or forward RNN. This emphasizes the utility of using information from past and future context together.

At last, results presented in Table 3 show that the bidirectional RNN encoder-decoder based on an attention-based mechanism performs better than a more classical bidirectional RNN.

Also it is shown that a RNN Encoder-Decoder performs very poorly without the attention mechanism in order to produce an output label sequence having the same length that the input word sequence.

4 Conclusion

This study aims to explore the use of a bidirectional RNN based on an mechanism of attention for a SLU slot filling task. Our experiments show that this architecture reaches better results than a more classical bidirectional RNN approach on a complex SLU corpus. This former bidirectional RNN approach had been introduced and presented as a state-of-the-art approach for SLU two years ago by [4] on the ATIS corpus. Even if [7] has shown that CRF still performs better than this bidirectional RNN on more complex data like the MEDIA corpus, our results show that promising approaches like the mechanism of attention can still improve RNN approach for SLU.

Acknowledgments

Thanks to the ANR agency for funding through the CHIST-ERA ERA-Net JOKER under the contract number ANR-13-CHR2-0003-05.

In addition, the authors thank Sahar Ghannay for her constructive help during the writing of this paper.

References

[1] R. De Mori, F. Bechet, D. Hakkani-Tür, M. McTear, G. Riccardi, and G. Tur, "Spoken language understanding," *Signal Processing Magazine, IEEE*, vol. 25, no. 3, pp. 50–58, 2008.

[2] S. Hahn, M. Dinarelli, C. Raymond, F. Lefevre, P. Lehnen, R. De Mori, A. Moschitti, H. Ney, and G. Riccardi, "Comparing stochastic approaches to spoken language understanding in multiple languages," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 19, no. 6, pp. 1569–1583, 2011.

[3] J. D. Lafferty, J. D. Mccallum, and F. C. N. Pereira, "Conditional random fields: probabilistic models for segmenting and labeling sequence data," in *Proceedings of the 18th International Conference on Machine Learning (ICML-2001)*, (San Francisco, CA, USA), 2001.

[4] G. Mesnil, X. He, L. Deng, and Y. Bengio, "Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding.," in *INTERSPEECH*, pp. 3771–3775, 2013.

[5] G. Mesnil, Y. Dauphin, K. Yao, Y. Bengio, L. Deng, D. Hakkani-Tur, X. He, L. Heck, G. Tur, D. Yu, *et al.*, "Using recurrent neural networks for slot filling in spoken language understanding," *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, vol. 23, no. 3, pp. 530–539, 2015.

[6] C. T. Hemphill, J. J. Godfrey, and G. R. Doddington, "The atis spoken language systems pilot corpus," in *Proceedings of the DARPA speech and natural language workshop*, pp. 96–101, 1990.

[7] V. Vukotic, C. Raymond, and G. Gravier, "Is it time to switch to word embedding and recurrent neural networks for spoken language understanding?," in *InterSpeech*, 2015.

[8] H. Bonneau-Maynard, M. Quignard, and A. Denis, "Media: a semantically annotated corpus of task oriented dialogs in french," *Language Resources and Evaluation*, vol. 43, no. 4, pp. 329–354, 2009.

[9] A. Graves, "Generating sequences with recurrent neural networks," *arXiv preprint arXiv:1308.0850*, 2013.

[10] J. Chorowski, D. Bahdanau, K. Cho, and Y. Bengio, "End-to-end continuous speech recognition using attention-based recurrent nn: First results," *arXiv preprint arXiv:1412.1602*, 2014.

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

[12] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," *arXiv preprint arXiv:1506.07503*, 2015.

[13] G. Mesnil, "Recurrent Neural Networks with Word Embeddings DeepLearning 0.1 documentation." `http://www.deeplearning.net/tutorial/rnnslu.html#rnnslu`.