Experimental Research on Encoder-Decoder Architectures with Attention for Chatbots

Marta R. Costa-jussà[1], Álvaro Nuez[1], Carlos Segura[2]

[1] Universitat Politècnica de Catalunya, TALP Research Center, Barcelona, Spain

[2] Telefònica I+D, Barcelona, Spain

marta.ruiz@upc.edu, alvaronuez.eis@gmail.com, carlos.seguraperales@telefonica.com

Abstract. Chatbots aim at automatically offering a conversation between a human and a computer. While there is a long track of research in rule-based and retrieval-based approaches, the generation-based approaches are promisingly emerging solving issues like responding to queries in inference that were not previously seen in development or training time. In this paper, we offer an experimental view of how recent advances in close areas as machine translation can be adopted for chatbots. In particular, we compare how alternative encoder-decoder deep learning architectures perform in the context of chatbots. Our research concludes that a fully attention-based architecture is able to outperform the recurrent neural network baseline system.

Keywords. Chatbot, encoder-decoder, attention mechanisms.

1 Introduction

A chatbot stands for the short version of *chat* plus *robot* and it is a computer program that conducts a human-machine conversation in any topic.

One of the very first chatbots was rule-based. It was proposed in 1966 by Joseph Weizenbaum's program ELIZA [13]. Input sentences were analyzed using several predefined decomposition rules, and after that key words were used to generate responses to them. The Artificial Intelligence Markup Language (AIML) is an evolution of these first rule-based chatbots. This AIML follows the idea of defining written *patterns* and the corresponding *templates* which are responses to the patterns. Then, in inference, if the robot identifies a pattern in a sentence from a user, the robot is able to reply taking the corresponding template [11].

To reduce the amount of work that developing these patterns and templates requires, alternative chatbots, no longer rule-based, but retrieval-based were proposed. These systems use different dialogue databases to train an information retrieval system [2]. The big advantage of these retrieval-based systems is that their training requires little human dedication. However, these systems still rely on giving the most appropriate response from a set of sentences, which limits their performance in the case of unseen events.

Thanks to the emergent deep learning techniques, the novel generative-based approaches have arisen offering chatbots that are capable, for the first time, to respond to non-predefined sentences. The first successful approach is based on the popular encoder-decoder architecture, which has been effectively used in quite a few natural language applications, and, moreover, it has been extended to image and speech processing [10, 12]. One successful implementation of this encoder-decoder architecture in natural language processing has been the recent concatenation of recurrent neural networks [7, 3]. In fact, this architecture builds on top of recurrent neural language models [6] by adding an encoder

Computación y Sistemas, Vol. 22, No. 4, 2018, pp. 1233–1239
ISSN 1405-5546
doi: 10.13053/CyS-22-4-3060

step and a decoder step. In the encoder step, a recurrent neural network converts an input sequence into a fixed representation (called thought vector). This representation is fed in the recurrent neural network from the decoder step which allows the decoder model to output more intelligent predictions given the context from the encoding. While this implementation has shown some results in chatbots [10], the main drawback is that long sequences are not well codified into a single vector. This challenge is faced through the recent attention-based mechanisms [1, 9] recently proposed for machine translation.

The main contribution of this paper is the application of the experimentation of these attention-based mechanisms [1, 9] to chatbots. Taking [10] as starting point, we compare the encoder-decoder architecture with attention [1] and the transformer [9]. A manually performed evaluation shows that the latter is able to outperform the encoder-decoder with attention which is already better than the encoder-decoder baseline architecture.

The rest of the paper is organized as follows. Section 2 briefly introduces the deep learning architectures used in this work, which basically are encoder-decoder based on recurrent neural networks (with or without attention mechanism) and the transformer which uses a fully attention-based encoder-decoder without recurrent neural networks. Section 3 details the experimental framework, particularly, data statistics and parameters from systems. Section 4 describes the manual evaluation. Section 5 discusses insights of results and contributions of this study.

2 Encoder-Decoder Architectures and Attention-based Mechanisms

An autoencoder is a type of neural network that aims at learning a representation of the input while allowing for a decoding of this representation by minimizing the recovering error. A generalization of this architecture is the encoder-decoder which allows for inputs and outputs to be different. This architecture, see a diagram of it in Figure 1 (left), has emerged as an effective paradigm for dealing with variable-length inputs and outputs. Although this effectiveness, its simplicity limits their performance. For example, it seems unrealistic that the input information has to fit in a fixed length vector of the internal representation. It seems more reasonable to take input information gradually while we are generating the output. That is why, attention-based mechanisms have arisen as a solution to this matter.

In this section, we briefly provide a high-level description of this encoder-decoder with recurrent neural networks plus two successful encoder-decoder implementations that use attention-based mechanisms. Among these two architectures we first describe the encoder-decoder based on recurrent neural networks with attention based on multi-layer perceptron. And, we second describe the transformer encoder-decoder architecture that uses only a combination of feed-forward neural networks with more sophisticated attention based on multiple heads.

2.1 Encoder-Decoder with Recurrent Neural Networks

Given an input sentence, the encoder iteratively computes for each word a hidden state vector using the word and previous hidden state of the recurrent neural network (RNN). Once the whole sentence has been analyzed, the relevant information of the input sentence is contained in the last hidden state of the RNN, known as context or thought vector. The decoder computes, word by word, an output in the original representation space using the information contained in the context vector and previous decoded words.

The architecture implementation can vary depending on the type of RNN cell used (genuine RNN cell, a LSTM cell [4] or a GRU cell [3]), number of cells per layer or the number of hidden layers among other parameters. Figure 1 (left) shows a diagram of this architecture.

One of the main drawbacks of this architecture resides in the fact that as the size of the input sentence increases, the encoder needs to compress a large quantity of information into a fixed-length vector.

This is lousy compressing process that may yield to a poor performance of the chatbot.

2.2 Encoder-Decoder with Recurrent Neural Networks with Attention

To overcome the aforementioned drawback of the basic RNN-based encoder-decoder approach, an attention mechanism is commonly used in the decoder [1]. In this case, for each generated word, the decoder computes a context vector composed of the weighted sum of all hidden state vectors of the encoder instead of relying on the ability of the encoder to compress the whole input sequence into the thought vector.

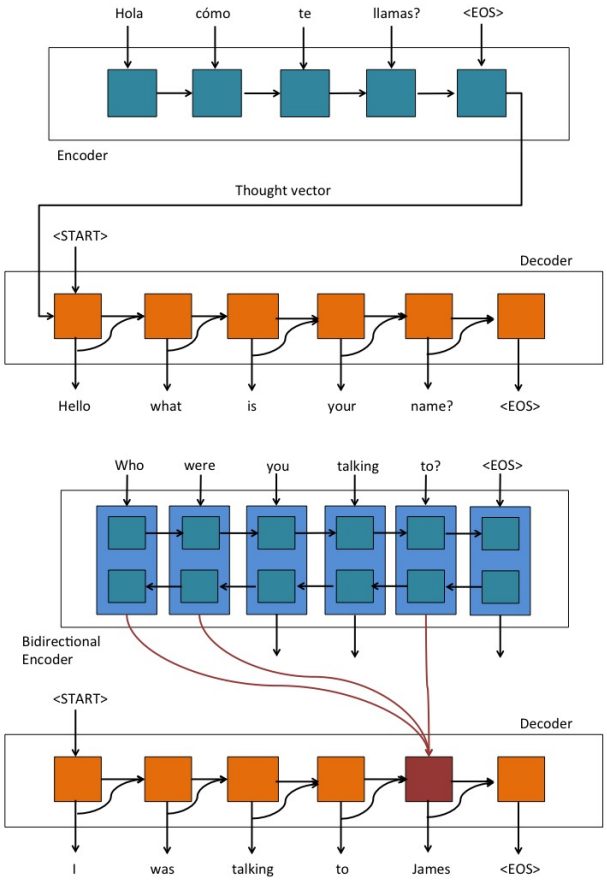

Fig. 1. (Left) Encoder-decoder with RNNs; (Right) Encoder-decoder with RNNS and attention

Weights are computed by an alignment model and normalized over all values to get a percentage of how relevant the word from the input sentence is, in relation to the word to be decoded, see figure 1 (right) showing the diagram of this architecture. For further technical explanation of how weights are computed see [1].

2.3 Transformer

While previous architecture has been successfully applied to machine translation, there are still some issues to solve. The architecture in practice can be really slow to train and given the way RNNs deal with sequences, it is not easy to parallelize the algorithm and take advantage of recent computational resources such as Tensor Processing Units (TPUs). Motivated by this issue, the Transformer model [9] has been proposed and it has been proven to be competitive in the task of machine translation. The Transformer model is able to improve state-of-the-art results in a couple of academic benchmarks while speeding up training by an order of magnitude in comparison to RNN-based encoder-decoder with attention shown in previous section.

The Transformer architecture is basically an encoder-decoder which concatenates attention-based mechanisms allowing to model relationships between words without requiring recurrence. More specifically, there are three main stages in the encoder (see Figure 2). The first one is where input words are projected into a vector representation space by an embedding matrix and then, given that there is no information of the order and position of words in the input sentence, a positional encoding is added to the embedded input vectors. Note that in previous RNN encoder/decoder models, due to their sequential nature, no positional information is required.

The second stage is a multi-head attention block (of Self-Attention in this first case) that linearly projects the input information into different space representations and performs attention over all of them. This method allows the model to identify different semantic, morphological and lexical characteristics of the input sequence and attend them separately at the decoding process.

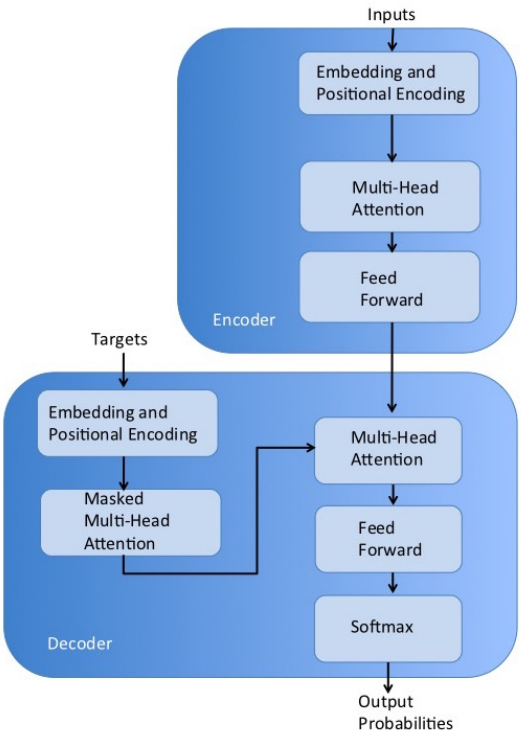

Fig. 2. Transformer

Finally, a position-wise feed-forward network is used, which applies two linear transformations to each position separately.

The decoder has five stages, the first two only used at the training phase: an output embedding and positional encoding (similar to the one used in the encoder but for target sentences in the training phase), a masked multi-head attention (also Self-Attention), a multi-head attention, a feed forward network and finally a softmax layer to compute the output probabilities.

Given that at the decoding process we can not know the future words, the attention can only be applied to previous ones. This is what the masked multi-head attention does, which is a multi-head attention block with a mask that restricts the attention only to past words. For a deeper technical explanation of the architecture see [9].

3 Experimental Framework

This section reports the data, preprocessing and parameters that we used to build our chatbot systems.

3.1 Data and Preprocessing

Models were tested on the OpenSubtitles dataset [8]. The Open Subtitles Corpus is composed by a wide range of movie and TV series scripts translated to multiple languages. It is generally used by video platforms to show subtitles of their movies/TV series.

The subtitles do not contain identity nor turn information. Therefore, similarly to [10], we assumed that consecutive sentences were uttered by different characters. We constructed a dataset consisting of pairs of consecutive utterances, using every sentence twice as context and as target. Due to computing and memory constrains, we extracted a subset of the first 10 million sentences for training.

Preprocessing of the database consisted on removing XML tags, limiting the sentence size and removing strange symbols (i.e., #, ”, *, -, musical notes, etc.). For evaluation we used the same 200 sentences that were used in [10], which covers different styles of conversation (i.e. basic, philosophical, personality and general knowledge). Details on training and evaluation split are reported on Table 1.

3.2 Parameters

At the experiments in [10], the architectures had 4096 unit cells for the entire OpenSubtitles database. Due to computational limitations, our model had to be simpler both by limiting the database (as we explained) and also by using a two layered LSTM model with 512 unit cells per layer. Additionally, the model with attention uses a sampled softmax loss function with 512 samples. All three models have a 64 dense size for the embedding matrix. To ensure that we cover the 99% of the dataset, we have limited the vocabulary size to 72,827 words and the length of sentences to 24 words. All words that are used only once are discarded.

Table 1. Size of the training database

Set	Role	Segments	Words	Vocab
Training	Context	20,000,000	64,192,197	180,368
	Target		48,174,044	182,404
Evaluation	Context	200	1,446	399

For training, we used ADAM [5] with a learning rate of 0.002, an exponential decay rate for the first moment estimates (β_1) equal to 0.9, an exponential decay rate for the second moment estimates (β_2) equal to 0.999 and $\epsilon = 10^{-8}$ (offset to prevent any division by zero); a batch size of 256 and a dropout rate of 0.1.

The transformer architecture has 8 attention heads and 6 hidden layers with 512 units. For training, we have also used ADAM with a learning rate of 0.2, exponential decay rates for the first and second moment estimates $\beta_1 = 0.9$ and $\beta_2 = 0.98$. and $\epsilon = 10^{-9}$; a batch size of 4096 and a drop out and attention dropout rates of 0.1. For better responses, we have added to the three models a beam search algorithm with a beam size of 4 and a penalty length factor of 0.6.

4 Evaluation

There is no automatic measure established for chatbots. Therefore, we carried out the evaluation manually. In order to compare our three conversational models, we asked three evaluators to rank chatbot responses among: good, bad or neutral. Additionally, we annotated when the three evaluators agreed in rankings.

Results are shown in Table 2 showing ranking from Evaluator 1, Evaluator 2, Evaluator 3 and agreement between them. Results show clear improvement of both attention mechanisms over the baseline system. In fact, all evaluators agree that the transformer architecture is good or neutral in 34.5% of the questions whereas the baseline architecture is only good or neutral in 23.5% of the questions and the baseline plus attention is good or neutral in 24.5% of the questions.

In addition to the results, we report several examples on different styles of conversations present in the test set: basic, philosophical, personality and general knowledge. These examples show the type of improvement that attention is capable to bring to the system which in general include more adequate responses in the context of the user input.

Conversation 1: Basic

Context	what is your name ?
RNN	Anouk.
+Attention	Adriano faustino sotolongo.
Transformer	i don't know .
Context	what time is it ?
RNN	I do n't know.
+Attention	6:30.
Transformer	i don't know.

Conversation 2: Philosophical

Context	I think you are a machine
RNN	I'm sorry
+Attention	Vlanne:
Transformer	I'm not a machine .
Context	Life is sad .
RNN	I'm not a monster.
+Attention	I do n't want to go to jail.
Transformer	It's not sad .
Context	Is there a god ?
RNN	No.
+Attention	Surprisingly, yes.
Transformer	no .

Conversation 3: Personality

Context	How old were you when you were able to speak ?
RNN	Eighteen.
+Attention	Seventeen.
Transformer	six months.

Conversation 4: General knowledge

Table 2. Manual Evaluation. Best results in bold

System	Ranking	Evaluator 1	Evaluator 2	Evaluator 3	Agreement
RNN	Good	41	47	54	32
+Attention		61	51	71	43
Transformer		74	57	70	**51**
RNN	Bad	75	123	53	46
+Attention		90	116	57	46
Transformer		25	110	25	**20**
RNN	Neutral	84	30	93	15
+Attention		49	33	72	6
Transformer		101	33	105	**18**

Context	ronald reagan is ...
RNN	I'm not a good person.
+Attention	And i'm not a spy..
Transformer	he's in the hospital.
Context	Is five plus five equal to ten ?
RNN	No, no.
+Attention	That's exactly what i'm gon na do.
Transformer	yes .

5 Conclusions

Attention-based mechanisms are revolutionizing natural language, speech and image processing applications. In this paper, we are implementing a couple of recently proposed attention mechanisms into the chatbot application.

Experiments trained on an open-domain database show that a fully attention-based architecture performs significantly better in a variety of contexts including basic, philosophical, personality and general knowledge. Three evaluators agreed on rating the responses of the fully attention-based mechanism 34.5% of the time either good or neutral, while the responses of the baseline architecture with attention was rated in that terms on a 24.5% of the time and the responses of the baseline system were only 23.5% of the time either good or neutral.

Taking advantage of this generic encoder-decoder architecture, among further research, we plan to introduce further contexts while training the system so as to allow the system to keep coherence in longer dialogues and to train our system on multiple languages.

Acknowledgments

This study has been funded by the Spanish Ministerio de Economía y Competitividad, the European Regional Development Fund and the Agencia Estatal de Investigación, through the post-doctoral senior grant Ramón y Cajal, the contract TEC2015-69266-P (MINECO/FEDER,EU) and the contract PCIN-2017-079 (AEI/MINECO).

References

1. **Bahdanau, D., Cho, K., & Bengio, Y. (2014).** Neural machine translation by jointly learning to align and translate. *CoRR*, Vol. abs/1409.0473.
2. **Banchs, R. E. & Li, H. (2012).** IRIS: a chat-oriented dialogue system based on the vector space model. *The 50th Annual Meeting of the Association for Computational Linguistics, Proceedings of the System Demonstrations, Jeju Island, Korea*, pp. 37–42.
3. **Cho, K., van Merrienboer, B., Gülçehre, Ç., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014).** Learning phrase representations using RNN encoder-decoder for statistical machine translation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pp. 1724–1734.
4. **Hochreiter, S. & Schmidhuber, J. (1997).** Long short-term memory. *Neural Comput.*, Vol. 9, No. 8, pp. 1735–1780.
5. **Kingma, D. P. & Ba, J. (2014).** Adam: A method for stochastic optimization. *CoRR*, Vol. abs/1412.6980.

6. **Mikolov, T., Karafiát, M., Burget, L., Cernocký, J., & Khudanpur, S. (2010).** Recurrent neural network based language model. *INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan*, pp. 1045–1048.
7. **Sutskever, I., Vinyals, O., & Le, Q. V. (2014).** Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems, Montreal, Quebec, Canada*, pp. 3104–3112.
8. **Tiedemann, J. (2009).** News from OPUS: A collection of multilingual parallel corpora with tools and interfaces. In *Recent Advances in Natural Language Processing*, volume V. John Benjamins, pp. 237–248.
9. **Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017).** Attention is all you need. *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems, Long Beach, CA, USA*, pp. 6000–6010.
10. **Vinyals, O. & Le, Q. V. (2015).** A neural conversational model. *CoRR*, Vol. abs/1506.05869.
11. **Wallace, R. (2003).** The elements of aiml style.
12. **Weiss, R. J., Chorowski, J., Jaitly, N., Wu, Y., & Chen, Z. (2017).** Sequence-to-sequence models can directly transcribe foreign speech. *CoRR*, Vol. abs/1703.08581.
13. **Weizenbaum, J. (1966).** ELIZA: a computer program for the study of natural language communication between man and machine. *Commun. ACM*, Vol. 9, No. 1, pp. 36–45.

Article received on 21/12/2017; accepted on 15/02/2018.
Corresponding author is Marta R. Costa-jussà.