ISSN 1870-4069

On Sequence Encodings for Positional Reasoning Task with Deep Neural Networks

Ari Reyes, Hiram Calvo

National Polytechnic Institute, Center for Computing Research, México

a160848@sagitario.cic.ipn.mx,hcalvo@cic.ipn.mx

http://www.cic.ipn.mx

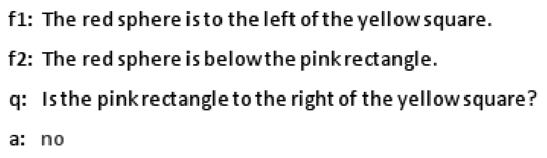
Abstract. This paper addresses the problem of encoding natural language in neural networks for the task of question answering on positional facts. Current state of the art works use many different ways to encode their inputs in natural language. Most of them separate each fact and their interaction with the question is independent. Another common issue is that, when encoding is not based on bags of words, sequence of words is considered, but the effect of alignment has not been particularly studied. In this paper we propose representing the intermediate states of a Recurrent Neural Network (Particularly a Long Short Term Memory network) as a matrix, and then using a convolutional layer on it. This architecture allows to experiment with different strategies of word alignment, as well as different modes of interaction between facts and questions, including a 3D convolution to combine word alignments and interaction of all facts and the question to be answered. We apply this model to the Positional Reasoning Task of bAbI to evaluate our proposed models. We found that alignment does not play a very important role in this task, but allowing interaction between all facts and question simultaneously is important to improve performance.

Keywords. Question answering, LSTM, CNN, deep learning, sequence encodings.

1 Introduction

Question answering (QA) is a complex problem within natural language processing that involves understanding a question and reasoning about provided facts presented in order to give an answer the question. Many efforts have been focused on building rule-based solutions [4]; however, due to the great flexibility and versatility of natural language, coupled with the fragility of these solutions, one of the major challenges for these systems has been the encoding of natural language into a formal language in such a way that it allows finding an answer using rules of inference.

On the other hand, recently different models based on neural networks, such as Memory Network [8], Dynamic Memory Network [3] and Neural Reasoner [4],


f1: The red sphere is to the left of the yellow square.

f2: The red sphere is below the pink rectangle.

q: Is the pink rectangle to the right of the yellow square?

a: no

Fig. 1. Example facts, question and answer from bAbI Task 17: Positional Reasoning.

have been proposed. All of them use deep neural networks with the goal of answering factual questions.

Notwithstanding, the problem of encoding natural language is still a gist for these systems. Every model uses different ways of encoding their inputs. Most of them isolate the interaction of the question to be answered with each fact separately; additionally, when not considering bags of words, sequence of words is taken into account, but, to our knowledge, the effect of alignment has not been particularly studied. Our hypothesis is that, by allowing all the intermediate states of a Recurrent Neural Network (Particularly a Long Short Term Memory network—LSTM) to be represented as a matrix, and then using a convolutional layer on it, the effect of misalignment could be undertaken. The same strategy can be used to allow interaction between question and all facts, when they are represented as a matrix as well. In this paper we present different models in order to cover these issues, and finally we propose using a 3D convolution to allow both kinds of interaction: different alignments along with all facts and the question to be answered.

One of the datasets used to evaluate neural network model for QA is the bAbI tasks dataset [7]. The bAbI tasks were created to measure the progress in the development of an intelligent dialog agent, which allow altogether to evaluate the reading comprehension of a system using Question Answering [7]. Particularly, in this paper we use the task Positional Reasoning—See Figure 1 for an example of the problems from this task—in order to verify our hypothesis.

The rest of this paper is organized as follows. In Section 2 we discuss related works, then in Section 3 we present our model with four variations proposed to answer the aforementioned questions. In Section 4 we give details of our experiments and results, and finally in Section 5 we draw our conclusions and outline some possibilities for future work.

2 Related Work

As far as we know, a study of the impact of using different encodings in the sequence of words to obtain a semantic representation of a sentence for this task (bAbI Task 17: Positional Reasoning) has not been conducted. As described in the previous section, we are interested on determining the convenience of modeling facts and question using the interaction of the words contained in them, and afterwards applying a sequential flattening; or, applying this sequential flattening first and then allowing the interaction of these sequential representations.

In general, works dealing with bAbI tasks are mainly based on neural networks with peculiar modifications to the classical models handled by these algorithms. These works enrich the wide panorama in which these models are developed allowing flexible and versatile solutions for the problems presented in question answering. One of these works introduces Memory Networks [8] whose main contribution is the addition of memory to a RNN.

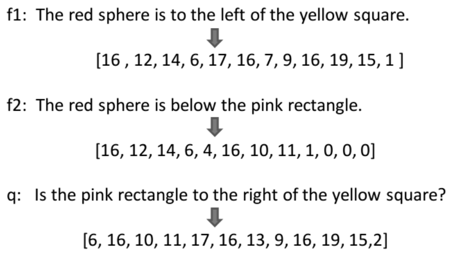
A modification to the previous work is presented in the End-to-End Memory Network [6] model, that can be seen as a kind of recurrent neural network that allows the handling of a memory that only produces an output after a fixed number of time steps with intermediate steps that update the internal state of the memory.

Another work brings in Dynamic Memory Networks (DMM) [3], that use an episodic memory with which it is possible to link the facts that relate more directly to the question leaving aside those that have no relevance to the answer.

A work even closer to the models proposed in this article is presented in [4]. The Neural Reasoner has a first layer that, using recurrent neural networks, encodes the facts and question separately, and then applies several layers of reasoning based on DMNs to obtain a solution to tasks 17 and 19 of the bAbI dataset.

Each one of these works tackles the problem of encoding natural language in different ways. For example, [8] encodes facts in a separate memory that is combined with the stream of words, [7] experiment with adding a special ”stop” class to separate facts and questions, using bags of n-grams instead of bags of words, and a multilinear map, i.e., a linear map of each word depending on its position. Finally, the Neural Reasoner [4] proposes using an encoding layer (based on Recurrent Neural Networks – RNNs) before all reasoning layers. This encoding layer later allows the question to interact separately with each fact through a Deep Neural Network (DNN), and finally, outputs of each DNN are combined to form an answer in the last layer by means of different pooling strategies.

In rough terms, the first two models propose a separate memory, while the last one, permits interaction between facts only after they have been compounded with the question. Several research interrogations arise from these works. For example, given that facts sometimes refer to relations between the same objects, would it not be advantageous to allow all of them to interact with the question, as well as between them? Another issue is related to the influence word position has in the encodings. If we model each fact and the question as sequences of words, then they must be aligned. How much does this affect the final result? And finally, sequential networks change their states as new inputs are considered. If we keep these changes as a matrix, then it would be possible for past states to interact with new states, rendering the problem of alignment as not important. We propose a model with different variations aiming to have a better panorama in the realm of these questions.


f1: The red sphere is to the left of the yellow square.

[16 , 12, 14, 6, 17, 16, 7, 9, 16, 19, 15, 1]

f2: The red sphere is below the pink rectangle.

[16, 12, 14, 6, 4, 16, 10, 11, 1, 0, 0, 0]

q: Is the pink rectangle to the right of the yellow square?

[6, 16, 10, 11, 17, 16, 13, 9, 16, 19, 15,2]

Fig. 2. Example left alignment in the numeric representation.

3 Proposed Model and Variations

In order to find answers to questions posed in the last section, we propose a neural model with different variations, so that we can verify: (1) the effect of allowing all facts and questions to interact before reasoning is carried out; (2) allowing past states of the sequential parts of the model to be stored in a matrix, so that the problem of misalignment is minimized; and (3) incorporating both variations in a single model.

Below is a general overview of the 4 models proposed for the solution of the Positional Reasoning task.

Within this task we have positional facts, see Figure 1, which will be subsequently named $f_1, f_2, ..., f_\mathrm{k}$; and a question, labeled as q.

For all models, a numerical representation is made first for each word from each fact f_i and question q according to a dictionary obtaining vectors of the same dimension n, which is based on the largest number of words found in the facts or questions. For those sentences with a smaller number of words than n, empty spaces are padded with zeros. This alignment can be to the right or to the left. For example, in Figure 2, $n = 12$ and f_2 is aligned to the left.

Subsequently, a vector representation for each word from each fact and question is obtained from the pre-trained 50-dimensional word vectors generated by GloVe [5].

In this way, we obtain matrices of size $n \times 50$ for the representation of each fact, which we will refer now as $F_1, F_2, ..., F_\mathrm{k}$ as well as the question Q.

In all models we use layers of *dropout* in order to avoid the co-adaptation of feature detectors; in other words, to avoid overfitting, and with this, to make the models more robust by reducing the adaptation to noise, following [1].

In the final part of all models there is a layer called Dense. This layer is a completely connected layer to all the vocabulary found in facts, questions and answers whose function is to convert the numerical answer to a word for each given question. See Figures 9 and 12.

3.1 Model 1

For this model we propose a scheme in which matrices obtained for each event $(F_1, F_2, ..., F_{\mathrm{k}})$ and Q are added. See equation 1:

$$S = Q + F_1 + F_2 + ... + F_k. \tag{1}$$

This resulting matrix S is passed to a recurrent neuronal network, specifically to an LSTM [2], which allows to encode the interaction of words in a sequential form.

Then, the values of each cell of the recurring network are obtained by forming an array with these interactions. Next, a convolutional network is applied to perform an analysis on these sequential representations. This process is shown in detail in Figure 4.

Once Matrix C is obtained, a convolutional network is applied. The purpose of this layer is to blend information of the obtained sequential representations. This layer uses 100 filters, and it is connected to a flattening layer that is connected to all known vocabulary. Then, an answer is selected as shown in Figure 9. This last flattening layer is labeled as *Dense*.

This model's block diagram can be found in Figure 3. Dropout layers were added as they were used in the experiments.

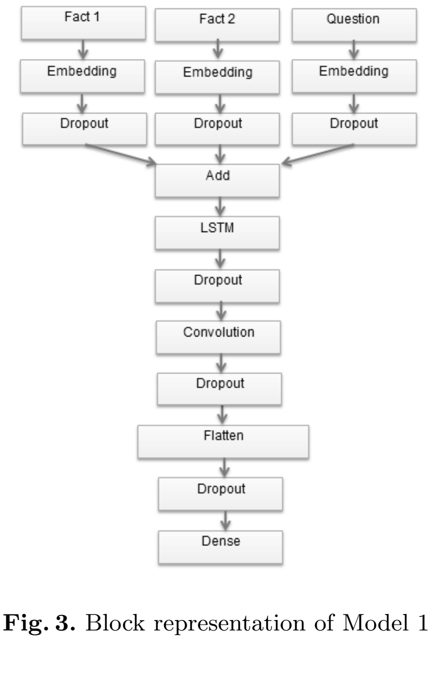


Fig. 3. Block representation of Model 1.

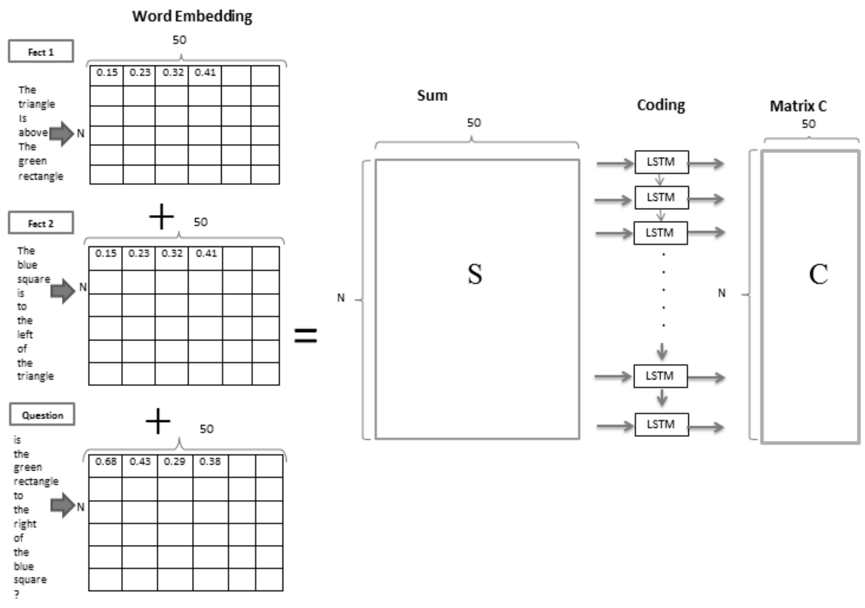

Word Embedding
50
Fact 1
The triangle Is above The green rectangle
N
0.15 0.23 0.32 0.41
Fact 2
The blue square is to the left of the triangle
Question
is the green rectangle to the right of the blue square ?
0.68 0.43 0.29 0.38
Sum
S
Coding
LSTM
Matrix C
C

Fig. 4. Obention of Matrix C after LSTM coding, based on the S matrix from Model 1.

3.2 Model 2

Model 2 consists in concatenating instead of adding up matrices. That is, $F_1, F_2, ..., F_\mathrm{k}$ and Q are concatenated. This is done by creating a final matrix of $((n \times m)+1) \times 50$, where m represents the number of positional facts and n the largest number of words contained in a fact or question. This process is depicted in Figure 6, showing the way in which matrices are concatenated and then entered into the recurring network LSTM.

In the same way as in Model 1, after the previous step we obtain a matrix C that continues with the same procedure of Model 1 that was described in Figure 9. Model 2 is presented in Figure 5.

3.3 Model 3

For Model 3, each matrix $(F_1, F_2, ..., F_\mathrm{k})$ and Q, is, word by word (in its vectorial representation), passed into a LSTM recurrent neuronal network. This change allows first to codify sentences in a sequential way to obtain separate matrices representing the semantics of each fact and the question. These new matrices are denominated $C_1, C_2, ..., C_k, C_{k+1}$ respectively. Subsequently the matrices obtained by this last step are added to obtain a final matrix C. This process is detailed graphically in Figure 8.

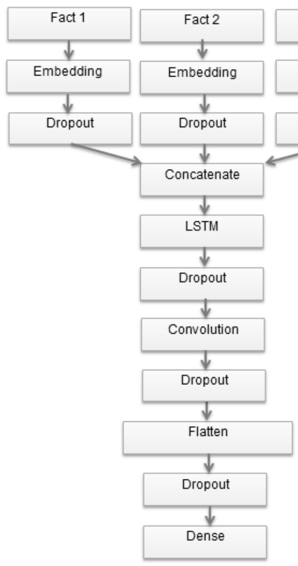


Fig. 5. Block representation of Model 2.

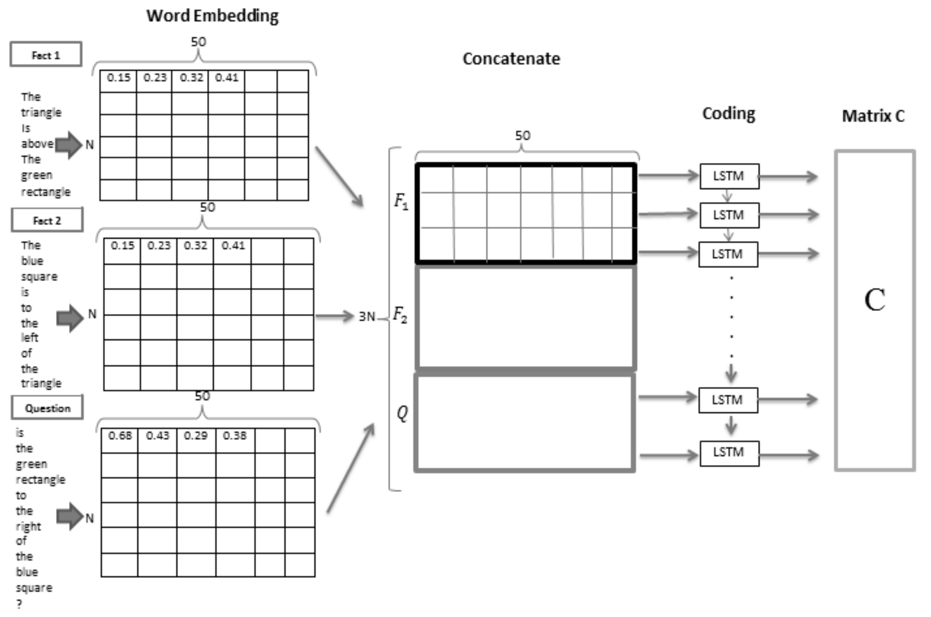


Fig. 6. Concatenation of matrices before LSTM as part of Model 2.

Again, to this point a matrix C is obtained on which a convolution is applied in the same way as in previous models (See Figure 9). Block diagram for this model is presented in Figure 7.

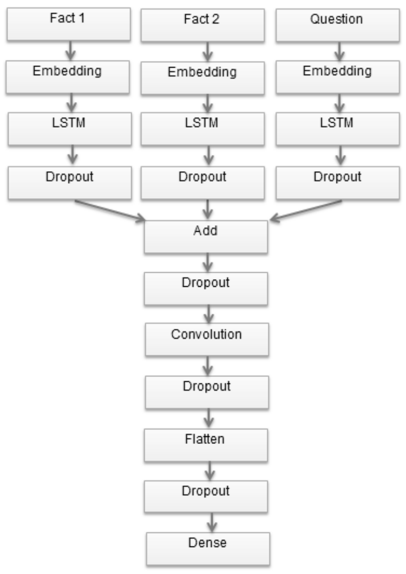


Fig. 7. Block representation of Model 3.

3.4 Model 4

Finally, in Model 4 a modification is made to Model 3 where, instead of adding the matrices obtained by each LSTM, a concatenation of these matrices is done in such a way that a three-dimensional matrix is obtained where each channel contains the sequential representation of a fact and of the question, allowing to make a 3D Convolution on all channels that compose this matrix (see Figure 10).

The intuition behind this model can be seen in Figure 11. This figure shows how matrices obtained from the sequential coding of sentences, now referred to as $C1, C_2, ..., C_k, C_{k+1}$, are concatenated to obtain a representation in a new matrix C.

Obtaining this matrix C, which represents a matrix with 3 channels, allows to do a convolution on these channels. The procedure for this is very similar to the previous models except for the convolution mask, that now resembles a cube that travels through matrix C. This the last process of model 4. See Figure 12.

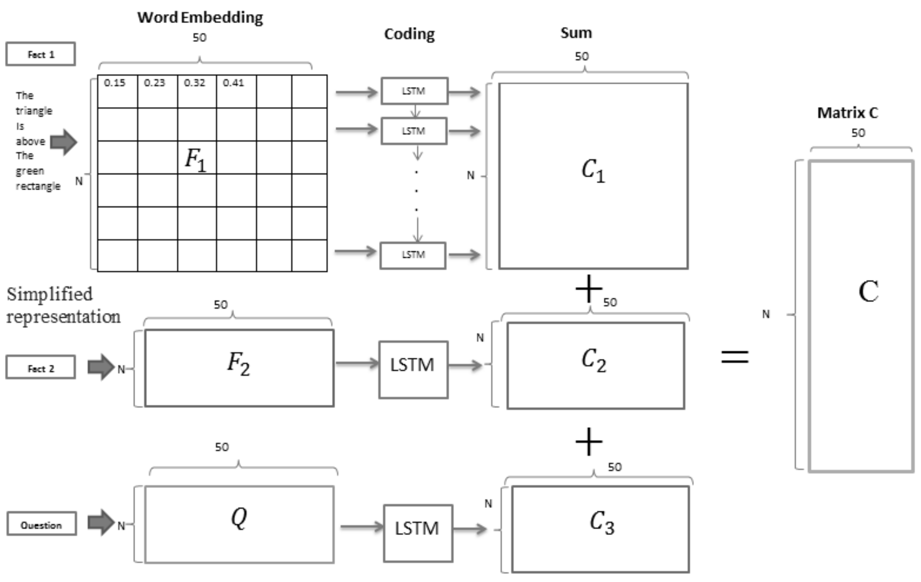


Fig. 8. Addition of matrices from the LSTM output as part of Model 3.

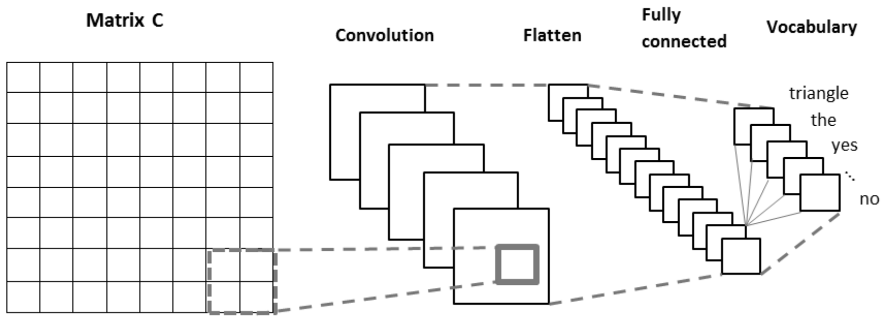


Fig. 9. Part 2 of the detailed Model 1,2 & 3 process.

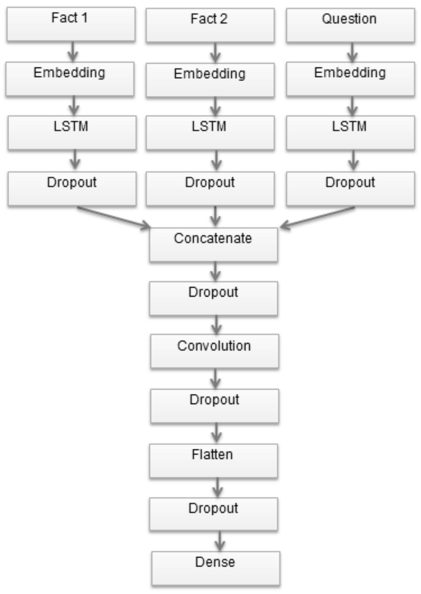


Fig. 10. Block representation of Model 4.

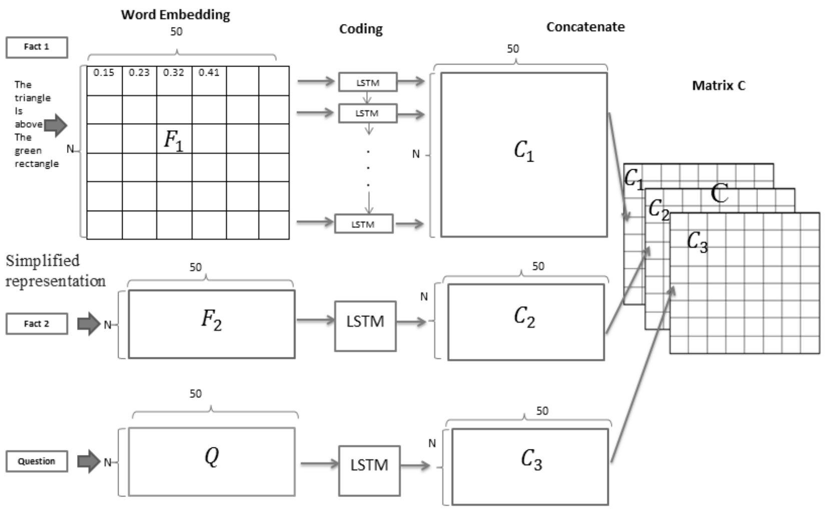


Fig. 11. Concatenation of Matrices in first part of Model 4.

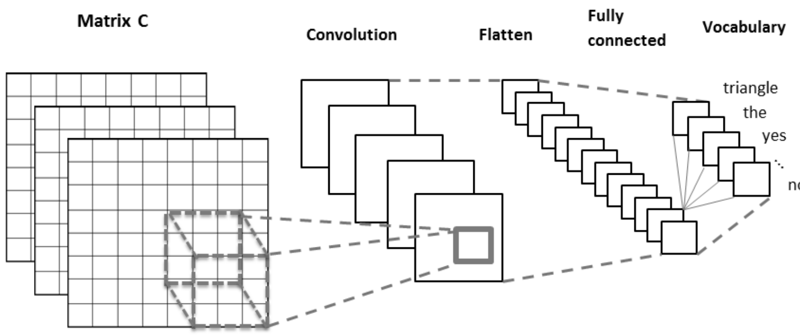


Fig. 12. Three dimensional convolution in Model 4.

4 Experiments and Results

For these experiments, the subset 1k from bAbI Task 17 was used. 1,000 instances were used for the training, and 1,000 instances for testing. 200 epochs and a batch size of 32 were used. Alignment to the right and to the left in the numerical representation of the facts and questions was made to obtain conclusions about the effect of alignment of sentences in each model. For the convolutional networks the ReLu activation function was used and the number of filters used was 100. The experiment was carried out with different mask sizes in the convolution, see Section 4.2. In dropout layers, a value of 0.3 was used.

4.1 Efect of Alignment

We experimented with an alignment to the left (see Figure 2) and alignment to the right (by placing zeros on the opposite side) in the numerical representation of the sentences in order to attest its effect in the models. By this means, we can verify if there is any influence of the sequential coding of sentences using LSTM recurrent neural networks. Results of this experiment can be seen in table 1.

4.2 Variation of the Size of Masks in Convolution

We experimented with varying the size of the mask in the filters in convolutional networks. Particularly, 2x2, 3x3 and 4x4 masks were used allowing different interactions between the sequential representations of the models. The idea behind the variation of these masks is to allow the interaction between a greater or a lower number of words, aiming to obtain better characteristics on the sequential representations of each sentence. For this task, the convolution (also called 3D convolution in Model 4), has a maximum number of channels of 3, because there are at most three channels: one for the first fact, one for the second fact, and a third one for the question.

Table 1. Results on the models using right alignment, left alignment and different kernel sizes.

Model	Kernel size	Alignment	Accuracy on training	Accuracy on test
Model 1	2x2	right	0.782	0.617
Model 1	3x3	right	0.776	0.606
Model 1	4x4	right	0.799	**0.624**
Model 1	2x2	left	0.771	0.619
Model 1	3x3	left	0.766	0.587
Model 1	4x4	left	0.732	0.623
Model 2	2x2	right	0.812	0.558
Model 2	3x3	right	0.78	0.555
Model 2	4x4	right	0.758	0.587
Model 2	2x2	left	0.791	0.559
Model 2	3x3	left	0.785	0.599
Model 2	4x4	left	0.754	0.582
Model 3	2x2	right	0.97	0.579
Model 3	3x3	right	0.952	0.549
Model 3	4x4	right	0.949	0.583
Model 3	2x2	left	0.946	0.579
Model 3	3x3	left	0.933	0.568
Model 3	4x4	left	0.93	0.578
Model 4	2x2x2	right	0.792	0.507
Model 4	3x3x3	right	0.793	0.532
Model 4	4x4x3	right	0.827	0.545
Model 4	2x2x2	left	0.772	0.526
Model 4	3x3x3	left	0.793	0.534
Model 4	4x4x3	left	0.805	0.509

4.3 Comparison with the State of the Art

Results of Table 2 were obtained from related works ([7], [6],[3],[4]), considering the best results achieved by each algorithm. We include the best result obtained by Model 1 within this comparison.

It can be observed that Model 1 is above algorithms like the Dynamic Memory Networks and End-to-End Memory Networks; however its performance is below the Neural Reasoner. We did not consider the version of Neural Reasoner with auxiliary information, because external resources were not used, as with plain Neural Reasoner.

5 Conclusions and Future Work

We found that concatenation of matrices was not useful within the interactions of the sequential representations of the words, since Models 2 and 4 have a lower performance than Models 1 and 3 that add matrices, instead of concatenating them. Because models adding matrices performed better, we can support the

Table 2. Accuracy on the task 17 Positional reasoning for different algorithms.

Algorithm	Positional Reasoning (1K)
N-gram Classifier	46.0%
LSTM	51.0%
MemN2N	59.6%
Dynamic Memory Networks	59.6%
Structured SVM	61.0%
Model 1 (4×4)	*62.4*%
Neural Reasoner	66.4%

part of our hypothesis that blending facts and questions yields better results than restricting them to interact separately.

It was also observed that applying a layer of RNNs before adding or concatenating is not favorable to the results, probably because of the interaction between the linear substructures of the vectors generated from GloVe.

It is also possible to conclude that the effect of the alignment in the sentences is not so important due perhaps to the effect of the LSTM, which allows flexibility in this aspect.

Apparently larger masks in convolution yield better results. Part of our future work is to experiment with new shapes and larger sizes in these masks.

Despite the hypothesis that allowing interaction between all words in facts and questions seemed logical, doing a 3D convolution on the channels composed by matrices of all words from positional facts and the question had the lowest performance, as we attested with results of Model 4.

As future work, a parser can be implemented to obtain another a structured representation of sentences. In addition to this, other models that involve more convolution layers or pooling layers could be implemented in order to obtain better characteristics on the sequential representations.

Additionally, proposed models will be tested in the other bAbI tasks to obtain a better evaluation of the performance of these algorithms.

Acknowledgments Authors wish to thank the support of Mexican Government (SNI, SIP-IPN, COFAA-IPN, and BEIFI-IPN). This work was partially funded by CONACYT under the Thematic Networks program (Language Technologies Thematic Network project 281795).

References

1. Baldi, P., Sadowski, P.: Understanding dropout. In: Burges, C.J.C., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 26, pp. 2814–2822. Curran Associates, Inc. (2013), `http://papers.nips.cc/paper/4878-understanding-dropout.pdf`

2. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation 9(8), 1735–1780 (1997)
3. Kumar, A., Irsoy, O., Ondruska, P., Iyyer, M., Bradbury, J., Gulrajani, I., Zhong, V., Paulus, R., Socher, R.: Ask me anything: Dynamic memory networks for natural language processing. In: International Conference on Machine Learning. pp. 1378–1387 (2016)
4. Peng, B., Lu, Z., Li, H., Wong, K.: Towards neural network-based reasoning. CoRR abs/1508.05508 (2015), `http://arxiv.org/abs/1508.05508`
5. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: Empirical Methods in Natural Language Processing (EMNLP). pp. 1532–1543 (2014), `http://www.aclweb.org/anthology/D14-1162`
6. Sukhbaatar, S., Szlam, A., Weston, J., Fergus, R.: End-to-end memory networks. In: Cortes, C., Lawrence, N.D., Lee, D.D., Sugiyama, M., Garnett, R. (eds.) Advances in Neural Information Processing Systems 28, pp. 2440–2448. Curran Associates, Inc. (2015), `http://papers.nips.cc/paper/5846-end-to-end-memory-networks.pdf`
7. Weston, J., Bordes, A., Chopra, S., Rush, A.M., van Merriënboer, B., Joulin, A., Mikolov, T.: Towards ai-complete question answering: A set of prerequisite toy tasks. arXiv preprint arXiv:1502.05698 (2015)
8. Weston, J., Chopra, S., Bordes, A.: Memory networks. CoRR abs/1410.3916 (2014), `http://arxiv.org/abs/1410.3916`