Question Dependent Recurrent Entity Network for Question Answering

Andrea Madotto and Giuseppe Attardi

OpenEdition
Journals

Question Dependent Recurrent Entity Network for Question Answering

Andrea Madotto[* **]
The Hong Kong University of Science
and Technology

Giuseppe Attardi[†]
University of Pisa

Question Answering is a task which requires building models capable of providing answers to questions expressed in human language. Full question answering involves some form of reasoning ability. We introduce a neural network architecture for this task, which is a form of Memory Network, that recognizes entities and their relations to answers through a focus attention mechanism. Our model is named Question Dependent Recurrent Entity Network and extends the Recurrent Entity Network by exploiting aspects of the question during the memorization process. We validate the model on both synthetic and real datasets: the bAbI question answering dataset and the CNN & Daily News reading comprehension dataset. In our experiments, our models improved the existing Recurrent Entity Network and achieved competitive results in both dataset.

1. Introduction

Question Answering is a task that requires capabilities beyond simple Natural Language Processing since it involves both linguistic techniques and inference abilities. Both the document sources and the questions are expressed in natural language, which is ambiguous and complex to understand. To perform such a task, a model needs in fact to understand the underlying meaning of a text. Achieving this ability is quite challenging for a machine since it requires a reasoning phase (chaining facts, basic deductions, etc.) over knowledge extracted from the plain input data. In this article, we focus on two Question Answering tasks: a *Reasoning Question Answering* (RQA) and a *Reading Comprehension* (RC). These tasks are tested by submitting questions to be answered directly after reading a piece of text (e.g. a document or a paragraph).

Recent progress in the field has been possible thanks to machine learning algorithms which automatically learn from large collections of data. Deep Learning (LeCun, Bengio, and Hinton 2015) algorithms achieve the current State-of-The-Art in our tasks of interest. A particularly promising approach is based on *Memory Augmented Neural Networks*. These networks are also known as *Memory Networks* (Weston, Chopra, and Bordes 2015) or *Neural Turing Machines* (Graves, Wayne, and Danihelka 2014). In the literature the RQA and RC tasks are typically solved by different models. However, the two tasks share a similar scope and structure. We propose to tackle both with a model

---

* Human Language Technology Center, Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong.
  E-mail: eeandreamad@ust.hk
** Work done while at University of Pisa
† Dipartimento di Informatica, University of Pisa, Largo B. Pontecorvo, 3. E-mail: attardi@di.unipi.it

called *Question Dependent Recurrent Entity Network* (QDREN), which improves over the model called *Recurrent Entity Network* (Henaff et al. 2017).

Our major contributions are: 1) exploiting knowledge of the question for storing relevant facts in memory, 2) adding a tighter regularization scheme, and 3) changing the activation functions. We test and compare our model on two datasets, bAbI (Weston et al. 2016) and CNN & Daily News (Hermann et al. 2015), which are standard benchmark for both tasks. The rest of the paper is organized as follows: section *Related* outlines the models used in QA tasks, while section *Model* discusses the proposed QDREN model. Section *Experiments and Results* shows training details and performance achieved by our model. The section *Analysis* reports a visualization with the aim to explain the obtained results. Finally, section *Conclusions* summarizes the work done.

2. Related Work

2.1 Reasoning Question Answering

A set of synthetic tasks, called bAbI (Weston et al. 2016), has been proposed for testing the ability of a machine in chaining facts, performing simple inductions or deductions. These tasks became a standard benchmark for measuring reasoning QA, several examples are shown in Table 1 [1]. The dataset is available in two sizes, 1K and 10K training samples, and in two settings, i.e. with and without supporting facts. The latter allows knowing which facts in the input are needed for answering the question (i.e. a stronger supervision). Obviously, the 1K sample setting with no supporting facts is quite hard to handle, and it is still an open research problem. *Memory Network* (Weston, Chopra, and Bordes 2015) was one of the first models to provide the ability to explicitly store facts in memory, achieving good results on the bAbI dataset. An evolution of this model is the *End to End Memory Network* (Sukhbaatar et al. 2015), which allows for end-to-end training. This model represents the State-of-The-Art in the bAbI task with 1K training samples. Several other models have been tested in the bAbI tasks achieving competitive results, such as *Neural Turing Machine* (Graves, Wayne, and Danihelka 2014), *Differentiable Neural Computer* (Graves et al. 2016) and *Dynamic Memory Network* (Kumar et al. 2015, Xiong, Merity, and Socher 2016). Several other baselines have also been proposed (Weston et al. 2016), such as: an n-gram (Richardson, Burges, and Renshaw 2013) models, an LSTM reader and an SVM model. However, some of them still required strong supervision by means of the supporting facts.

2.2 Reading Comprehension

Reading Comprehension is defined as the ability to read some text, process it, and understand its meaning. A impending issue for tackling this task was to find suitably large datasets with human annotated samples. This shortcoming has been addressed by collecting documents which contain easy recognizable short summary, e.g. news articles, which contain a number of bullet points, summarizing aspects of the information contained in the article. Each of these short summaries is turned into a fill-in question template, by selecting an entity and replacing it with an anonymized placeholder.

Three datasets follows this style of annotation: *Children's Text Books* (Hill et al. 2016), *CNN & Daily Mail news articles* (Hermann et al. 2015), and *Who did What* (Onishi et al.

---

1 Interested readers can find all the tasks examples in (Weston et al. 2016)

Table 1
bAbI dataset examples.

Task	Counting	Lists/Sets
Story	Daniel picked up the football. Daniel dropped the football. Daniel got the milk. Daniel took the apple.	Daniel picks up the football. Daniel drops the newspaper. Daniel picks up the milk. John took the apple.
Question	How many objects is Daniel holding?	What is Daniel holding?
Answer	Two	Milk, football
Task	*Three Argument Relations*	*Yes/No Questions*
Story	Mary gave the cake to Fred. Fred gave the cake to Bill. Jeff was given the milk by Bill.	John moved to the playground. Daniel went to the bathroom. John went back to the hallway.
Question	Who gave the cake to Fred?	Is John in the playground?
Answer	Mary	No

2016). It is also worth to mention *Squad* (Rajpurkar et al. 2016), a human annotated dataset from Stanford NLP group. *Memory Networks*, described in the previous subsection, has been tested (Hill et al. 2016) on both the CNN and CBT datasets, achieving good results. The *Attentive and Impatient Reader* (Hermann et al. 2015) was the first model proposed for the *CNN & Daily Mail* dataset, and it is therefore often used as a baseline. While this model achieved good initial results, shortly later a small variation to such model, called *Standford Attentive Reader* (Chen, Bolton, and Manning 2016), increased its accuracy by 10%. Another group of models are based on an Artificial Neural Network architecture called *Pointer Network* (Vinyals, Fortunato, and Jaitly 2015). *Attentive Sum Reader* (Kadlec et al. 2016) and *Attention over Attention* (Cui et al. 2017) use a similar idea for solving different reading comprehension tasks. *EpiReader* (Trischler et al. 2016) and *Dynamic Entity Representation* (Kobayashi et al. 2016), partially follow the *Pointer Network* framework but they also achieve impressive results in the RC tasks. Also for this task several baselines, both learning and non-learning, have been proposed. The most commonly used are: *Frame-Semantics*, Word distance, and *LSTM Reader* (Hermann et al. 2015) and its variation (windowing etc.).

3. Proposed Model

Our model is based on the *Recurrent Entity Network* (REN) (Henaff et al. 2017) model. The latter is the only model able to pass all the 20 bAbI tasks using the 10K sample size and without any supporting facts. However, this model fails many tasks with the 1K setting, and it has not been tried on more challenging RC datasets, like the CNN news articles. Thus, we propose a variant to the original model called *Question Dependent Recurrent Entity Network* ($QDREN$)[2]. This model tries to overcome the limitations of

---

2 An implementation is available at https://github.com/andreamad8/QDREN

the previous approach. The model consists in three main components: *Input Encoder*, *Dynamic Memory*, and *Output Module*.

The training data consists of tuples $\{(x_i, y_i)\}_{i=1}^{n}$, with n equal to the sample size, where: x_i is composed by a tuple (T, q), where T is a set of sentences $\{s_1, \ldots, s_t\}$, each of which has a maximum of m words, and q a single sentence with k words representing the question. Instead, y_i is a single word that represents the answer.

The Input Encoder transforms the set of words of a sentence s_t and the question q into a single vector representation by using a multiplicative mask. Let's define $E \in \mathbb{R}^{|V| \times d}$ the embedding matrix[3], that is used to convert words to vectors, i.e. $E(w) = e \in \mathbb{R}^d$. Hence, $\{e_1, \ldots, e_m\}$ are the word embedding of each word in the sentence s_t and $\{e_1, \ldots, e_k\}$ the embedding of the question's words. The multiplicative masks for the sentences are defined as $f^{(s)} = \{f_1^{(s)}, \ldots, f_k^{(s)}\}$ and $f^{(q)} = \{f_1^{(q)}, \ldots, f_k^{(q)}\}$ for the question, where each $f_i \in \mathbb{R}^d$. The encoded vector of a sentence is defined as:

$$s_t = \sum_{r=1}^{m} e_r \odot f_r^{(s)} \qquad q = \sum_{r=1}^{k} e_r \odot f_r^{(q)}$$

Dynamic Memory stores information of entities present in T. This module is very similar to a Gated Recurrent Unit (GRU) (Cho et al. 2014) with a hidden state divided into blocks. Moreover, each block ideally represents an entity (i.e. person, location etc.), and it stores relevant facts about it. Different datasets may require different number of blocks, in the experiment section we will further discuss this issue. Each block i is made of a hidden state $h_i \in \mathbb{R}^d$ and a key $k_i \in \mathbb{R}^d$, where d is the embedding size. The Dynamic Memory module is made of a set of blocks, which can be represent with a set of hidden states $\{h_1, \ldots, h_z\}$ and their correspondent set of keys $\{k_1, \ldots, k_z\}$. The equation used to update a generic block i are the following:

$$g_i^{(t)} = \sigma(s_t^T h_i^{(t-1)} + s_t^T k_i^{(t-1)} + s_t^T q) \qquad \text{(Gating Function)}$$

$$\hat{h}_i^{(t)} = \phi(U h_i^{(t-1)} + V k_i^{(t-1)} + W s_t) \qquad \text{(Candidate Memory)}$$

$$h_i^{(t)} = h_i^{(t-1)} + g_i^{(t)} \odot \hat{h}_i^{(t)} \qquad \text{(New Memory)}$$

$$h_i^{(t)} = h_i^{(t)} / \|h_i^{(t)}\| \qquad \text{(Reset Memory)}$$

where σ represents the sigmoid function, ϕ a generic activation function which can be chosen among a set (e.g. sigmoid, ReLU, etc.). $g_i^{(t)}$ is the gating function which determines how much the ith memory should be updated, and $\hat{h}_i^{(t)}$ is the new candidate value of the memory to be combined with the existing one $h_i^{(t-1)}$. The matrix $U \in \mathbb{R}^{d \times d}$, $V \in \mathbb{R}^{d \times d}$, $W \in \mathbb{R}^{d \times d}$ are **shared** among different blocks, and are trained together with the key vectors. The addition of the $s_t^T q$ term in the gating function is our main contribution. We add such term with the assumption that the question can be useful to focus the attention of the model while analyzing the input sentences.

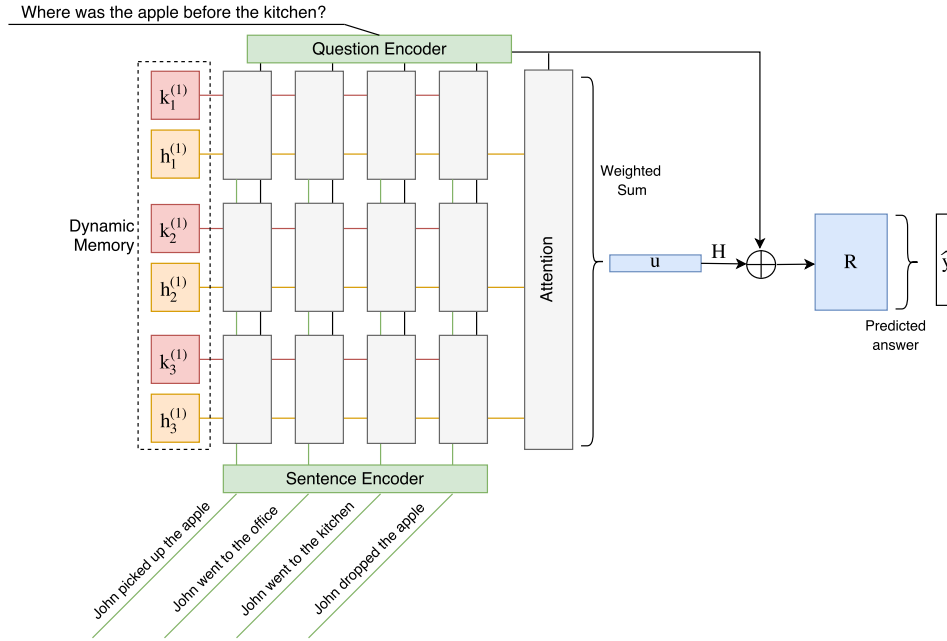3 Where $|V|$ is the vocabulary size and d the embedding dimension.

Figure 1
Conceptual schema of the QDREN model, with three memory blocks. In input a sample taken from bAbI task dataset.

The Output Module creates a probability distribution over the memories' hidden states using the question q. Thus, the hidden states are summed up, using the probability as weight, to obtain a single state representing all the input. Finally, the network output is obtained by combining the final state with the question. Let us define $R \in \mathbb{R}^{|V| \times d}$, $H \in \mathbb{R}^{d \times d}$, $\hat{y} \in \mathbb{R}^{|V|}$, z is the number of blocks, and ϕ can be chosen among different activation functions. Then, we have:

$$p_i = \text{Softmax}(q^T h_i)$$

$$u = \sum_{j=1}^{z} p_j h_j$$

$$\hat{y} = R\phi(q + Hu)$$

The model is trained using a cross entropy loss $H(\hat{y}, y)$ plus L2 regularisation term, where y is the one hot encoding of the correct answer. The sigmoid function and the L2 term are two novelty added to the original REN. Overall, the trainable parameters are:

$$\Theta = [E, f^{(s)}, f^{(q)}, U, V, W, k_1, \dots, k_z, R, H]$$

where $f^{(s)}$ refers to the sentence multiplicative masks, $f^{(q)}$ to the question multiplicative masks, and each k_i to the key of a generic block i. The number of parameters is dominated by E and R, since they depend on the vocabulary size. However, R is normally

15

is much smaller than E like in the CNN dataset, in which the prediction is made on a restricted number of entities[4]. All the parameters are learned using the Backpropagation Through Time (BPTT) algorithm. A schematic representation of the model is shown in Figure 1.

4. Experiments and Results

Our model has been implemented using TensorFlow v1.1 (Abadi et al. 2015) and the experiments have been run on a Linux server with 4 Nvidia P100 GPUs. As mentioned earlier, we tested our model in two datasets: the bAbI 1k sample and the CNN news articles. The first dataset have 20 separate tasks, each of which has 900/100/1000 training, validation, and test samples. Instead, the second one has 380298/3924/3198 training, validation and test samples. We kept the original splitting to compare our results with the existing ones.

bAbI:. in these tasks, we fixed the batch size to 32, we did not use any pre-trained word embedding, and we used Adam (Kingma and Ba 2015) optimizer. In all the experiment we used 20 blocks of memory since it is equal to the maximum number of entity in each story. We have also clipped the gradient to a maximum of 40 (to avoid gradient explosion), and we set the word embedding size to 100, as it has also been suggested in the original paper. We have also implemented an early stopping method, which stop the training ones the validation accuracy does not improve after 50 epochs. Several values for the hyper-parameter have been tried and, for each task, we selected the setting that achieved the highest accuracy in validation. Once we selected the best model, we estimate its generalization error using the provided Test set. Table 2 shows an example of the dataset and the used hyper-parameters. We compared our results

Table 2
On the left an example of the bAbI task, and on the right the selected model hyper-parameters.

Story	Question
John picked up the apple	Where was the apple before the kitchen?
John went to the office	
John went to the kitchen	**Answer**
John dropped the apple	office

Parameter	Values
Learning Rate (α)	0.01,0.001,0.0001
Number of Blocks	20,30,40,50
L2 reg. (λ)	0,0.001,0.0001
Dropout (**Dr**)	0.3,0.5,0.7

with four models: n-gram model, LSTM, original REN (with no question in the gating function) and *End To End Memory Network* (MemN2N) (Sukhbaatar et al. 2015), which is currently the State-Of-The-Art in this setting. To the best of our knowledge we achieved the lowest number of failed tasks, failing just 8 tasks compared with the previous State-Of-The-Art which was 11. Comparing our QDREN with the original *Recurrent Entity Network* (REN) we achieved, on average, an improvement of 11% in the average error rate and we passed 7 tasks more. Table 3 shows the error rate[5] in the test set obtained using each compared model, and the hyper-parameter setting used in each task. We improve the mean error compared to the original REN, however we still do know reach the error rate achieved by the *End To End Memory Network* (even if we passed more

---

4 Therefore $R \in \mathbb{R}^{|entities| \times d}$
5 The error is the percentage of wrong answers.

Table 3
Test set error rate comparison between n-gram, LSTM, QDREN, REN and End To End Memory Network (MemN2N). All the results have been taken from the original articles. In bold we highlight the task in which we greatly outperform the other models. On the right the hyper-parameters used in QDREN.

Task	n-gram	LSTM	MemN2N	REN	QDREN	Blk	λ	α	Dr
1	64	50	0	0.7	0	20	0	0.001	0.5
2	98	80	8.3	56.4	67.6	30	0	0.001	0.5
3	93	80	40.3	69.7	60.8	40	0	0.001	0.5
4	50	39	2.8	1.4	0	20	0	0.001	0.5
5	80	30	13.1	4.6	**2.0**	50	0	0.001	0.2
6	51	52	7.6	30	29	30	0	0.001	0.5
7	48	51	17.3	22.3	**0.7**	30	0	0.001	0.5
8	60	55	10	19.2	**2.5**	20	0.001	0.001	0.7
9	38	36	13.2	31.5	**4.8**	40	0.0001	0.001	0.5
10	55	56	15.1	15.6	**3.8**	20	0	0.001	0.5
11	71	28	0.9	8	**0.6**	20	0	0.001	0.5
12	91	26	0.2	0.8	0	20	0	0.0001	0.5
13	74	6	0.4	9	**0.0**	40	0.001	0.001	0.7
14	81	73	1.7	62.9	15.8	30	0.0001	0.001	0.5
15	80	79	0	57.8	**0.3**	20	0	0.001	0.5
16	57	77	1.3	53.2	52	20	0.001	0.001	0.5
17	54	49	51	46.4	37.4	40	0.001	0.001	0.5
18	48	48	11.1	8.8	10.1	30	0.0001	0.001	0.5
19	10	92	82.8	90.4	85	20	0	0.001	0.5
20	24	9	0	2.6	0.2	20	0	0.001	0.5
Failed Tasks (>5%):	20	20	11	15	8				
Mean Error:	65.9	50.8	13.9	29.6	18.6				

tasks). It is worth to notice the following two facts: first, in task 14 and 18 the error is very close to the threshold for passing the task (5%); second, in task 2, we achieved a slightly worse result (10% error more) with respect to the original REN.

CNN news articles:. in this dataset, the entities in the original paragraph are replaced by an ID, making the task even more challenging. The CNN dataset is already tokenized and cleaned, therefore we did not apply any text pre-processing. As it was done in other models, the set of possible answers is restricted to the set of hidden entities in the text, that are much less, around 500, compared to all the words (120K) in the vocabulary. Compared to the model used for bAbI, we changed the activation function of the output layer, using a sigmoid instead of parametric ReLU, since after several experiments we noticed that such activation was hurting the model performance. Moreover, the input was not split into sentences, thus we divided the text into sentences using the dot token ("."). sentence splitting in general is itself a challenging task, but in this case the input was already cleaned and normalised. However, the sentence may be very long, thus we introduced a windowing mechanism. The same approach has been used in the *End To End Memory Network* (Sukhbaatar et al. 2015) as a way to encode the input sentence. This method takes each entity marker ($@entity_i$) and it creates a window of b words around it. Formally, $\{w_{i-\frac{(b-1)}{2}}, \ldots, w_i, \ldots, w_{i+\frac{(b-1)}{2}}\}$, where w_i represent the entity of interest. For the question, a single window is created around the placeholder marker (the word to predict). Moreover, we add $2(b-1)$ tokens for the entities at the beginning and at the end of the text. To check whether our QDREN could improve the existent REN and whether the window-based approach makes any difference in comparison with plain sentences, we separately trained four different models: REN+SENT, REN+WIND, QDREN+SENT and QDREN+WIND. Where SENT represent simple input sentences,

and WIND the window as a input. For each of this model, we conduct a separated model selection using a various number of hyper-parameters. Table 4 shows an example of the dataset and the used hyper-parameters. As for the bAbI task, we used early

Table 4
On the left, an example from CNN news article, and on the right, the model selection Hyper-parameters.

Story	Question
(@entity1) @entity0 may be @entity2 in the popular @entity4 superhero films but he recently dealt in some advanced bionic technology ...	"@placeholder" star @entity0 presents a young child
	Answer
	@entity2

Parameter	Values
Learning Rate (α)	0.1,0.01,0.001,0.0001
Window	2,3,4,5
Number of Blocks	10,20,50,70,90
L2 reg. (λ)	0.0,0.001,0.0001,0.00001
Optimizer	Adam,RMSProp
Batch Size	128,64,32
Dropout (**Dr**)	0.2,0.5,0.7,0.9

stopping, ending the training once the validation accuracy does not improve for 20 epochs. Since each training required a large amounts of time (using a batch size of 64 an epoch takes around 7 hours), we opted for a random search technique (Bergstra and Bengio 2012), and we used just a sub-sample of the training set, i.e. 10K sample, for the model selection, but we still keep the validation set as it was. Obviously, this is not an optimal parameter tuning, since the model is selected on just 10K samples. Indeed, we noticed that the selected model, which is trained using all the samples (380K), tends to under-fit. However, it was the only way to try different parameters in a reasonable amount of time. Moreover, we also limited the vocabulary size to the most common 50K words, and we initialize the embedding matrix using Glove (Pennington, Socher, and Manning 2014) pre-trained word embedding of size 100. As for bAbI, we used 20 blocks of memory. As before, we selected the models that achieved the highest accuracy

Table 5
Test set accuracy comparison between **REN+SENT**, **QDREN+SENT**, **REN+WIND** and **QDREN+WIND**. We show the best hyper-parameters picked by the model selection, and the accuracy values.

	REN+SENT	QDREN+SENT	REN+WIND	QDREN+WIND
Number of Blocks	20	10	50	20
Window	-	-	5	4
Learning Rate	0.001	0.001	0.0001	0.01
Optimizer	Adam	Adam	RMSProp	RMSProp
Dropout	0.7	0.2	0.5	0.5
Batch Size	128	64	64	64
λ	0.0001	0.001	0.0001	0.0001
Loss Training	2.235	2.682	2.598	2.216
Loss Validation	2.204	2.481	2.427	1.885
Loss Test	2.135	2.417	2.319	1.724
Accuracy Training	0.418	0.349	0.348	0.499
Accuracy Validation	0.420	0.399	0.380	0.591
Accuracy Test	0.420	0.397	0.401	**0.628**

in the validation set, and then we estimate its generalization error using the provided test set. The selected models, with their hyper-parameters, are shown in Table 5. The

Table 6
Validation/Test accuracy (%) on CNN dataset. All the reported results are taken from the original articles: Max Freq.,Frame-semantic, Attentive Reader, Word distance, Impatient Reader, LSTM Reader from (Hermann et al. 2015), Attentive Reader(Chen, Bolton, and Manning 2016), AS for Attentive Sum(Kadlec et al. 2016), AoA for Attention over Attention (Cui et al. 2017), and DER for Dynamic Entity Representation(Kobayashi et al. 2016).

	Val	Test		Val	Test		Val	Test
Max Freq.	30.5	33.2	MemN2N	63.4	66.8	AS Reader	68.6	69.5
Frame-semantic	36.3	40.2	Attentive Reader	61.6	63	AoA	73.1	**74.4**
Word distance	50.5	50.9	Impatient Reader	61.8	63.8	EpiReader	73.4	74
LSTM Reader	55	57	Stanford (AR)	72.5	72.7	DER	71.3	72.9

best accuracy[6] is achieved by QDREN+WIND with a value of 0.628, while all other models could not achieve an accuracy greater than 0.42. The window-based version without question supervision could not achieve an accuracy higher than 0.401. Indeed, saving only facts relative to the question seems to be the key to achieving a good score in this task. We also noticed that using plain sentences, even with QDREN, we cannot achieve a higher accuracy. This might be due to the sentence encoder, since just using the multiplicative masks does not provide enough expressive power for getting key features of the sentence. Moreover, we notice that the accuracy achieved in the training set is always lower than that in the validation and test set. The same phenomenon is present also in other models, in our particular case this might be due to the strong regularization term used in our models. Our model achieves an accuracy comparable to the *Attentive and Impatient Reader* (Hermann et al. 2015), but not yet State-Of-The-Art model (i.e. *Attention over Attention* (AoA)). It is worth noting though that our model is much simpler and it goes through each paragraph just once. A summary of the other models' results are shown in Table 6.

5. Analysis

To better understand how our proposed model (i.e. QDREN) works and how it improves the accuracy of the existing REN, we studied the gating function behavior. Indeed, the output of this function decides how much and what we store in each memory cell, and it is also where our model differs from the original one. Moreover, we trained the QDREN and the original REN using the bAbI task number 1 using 20 memory blocks. The latter mention number of blocks has been selected heuristically by knowing that the maximum number of entity in the facts is 20. We pick up this task since both models pass it, and it is one of the simplest, which also allows to better understand and visualize the results. Indeed, we have tried to visualize other tasks but the result was difficult to understand since there were too many sentences in input and it was difficult to understand how the gate opened. The visualization result is shown in Figure 2, where we plotted the activation matrix for both models, using a sample of the validation set. In these plots, we can notice how the two models learn which information to store.

In Figure 2 (a), we notice that the QDREN is opening the gates just when in the

---

6 Percentage of correct answers.

sentence appears the entity named Mary. This because such entity is also present in the question (i.e., "where is Mary?"). Even though the model is focusing on the right entity, its gates are opening all at the same times. In fact, we guess that a sparser activation would be better since it may have modeled which other entities are relevant for the final answer. Instead, the gaiting activation of the original REN is sparse, which is good if we would like to learn all the relevant facts in the text. Indeed, the model effectively assigns a block to each entity and it opens the gates just ones such entity appears in the input sentences. For example, in Figure 2 (b) the block cell number 13 supposedly
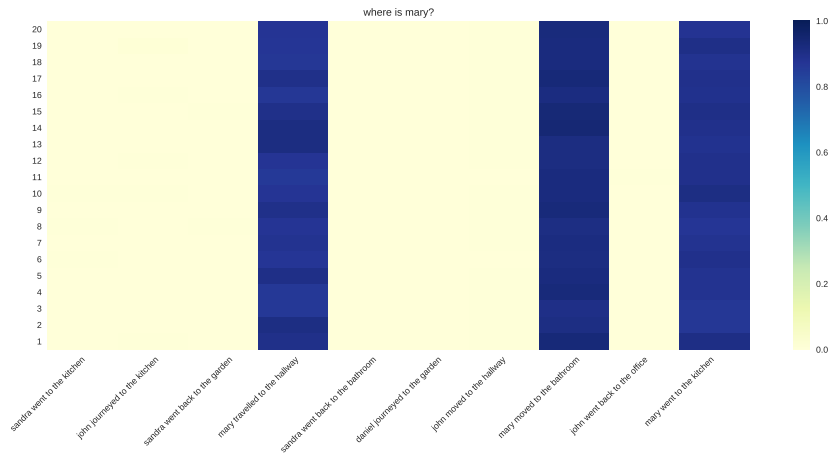
(a)
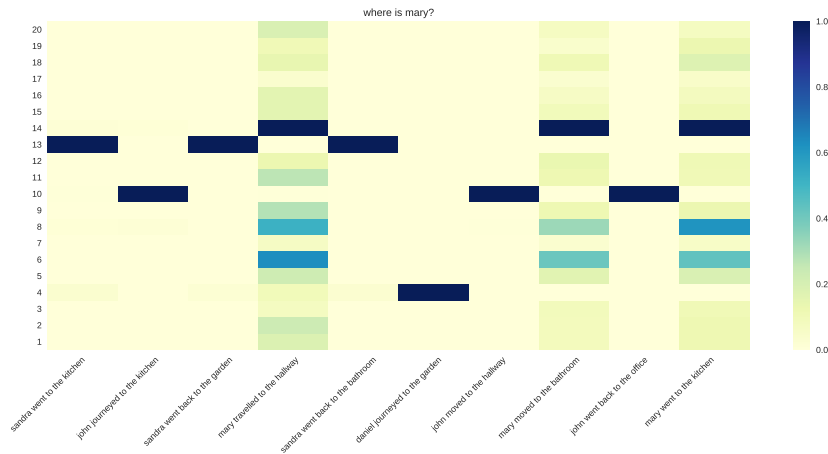
(b)

Figure 2
Heatmap representing the gating function result for each memory block. In the y-axes represents the memory block number (20 in this example), in the x-axes, there are the sentences in the input divided into time steps, and at the top, there is the question to be answered. Darker color means a gate more open (values close to 1) and lighter colour means the gate less open. (a) shows QDREN and (b) shows REN.

represents the entity Sandra, since each sentence in which this name appears the gate function of the block fully opens (value almost 1). Futher, we can notice the same phenomenon with the entity John (cell 10), Daniel (cell 4), and Mary (cell 14). Other entities (e.g., kitchen, bathroom, etc.) are more difficult to recognize in the plot since their activation is less strong and probably distributes this information among blocks.

6. Conclusion

In this paper we presented the *Question Dependent Recurrent Entity Network*, used for reasoning and reading comprehension tasks. This model uses a particular RNN cell in order to store just relevant information about the given question. In this way, in combination with the original *Recurrent Entity Network* (keys and memory), we improved the success rate in the bAbI 1k task and achieved promising results in the Reading comprehension task on the *CNN & Daily news* dataset. However, we believe that there are still margins for improving the behavior for the proposed cell. Indeed, the cell has not enough expressive power to make a selective activation among different memory blocks (notice in Figure 2 (a) the gates open for all the memories).

Acknowledgments

References

Abadi, Martín, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Bergstra, James and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb):281–305.

Chen, Danqi, Jason Bolton, and Christopher D. Manning. 2016. A thorough examination of the cnn/daily mail reading comprehension task. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2358–2367, Berlin, Germany. Association for Computational Linguistics.

Cho, Kyunghyun, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

Cui, Yiming, Zhipeng Chen, Si Wei, Shijin Wang, Ting Liu, and Guoping Hu. 2017. Attention-over-attention neural networks for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 593–602, Vancouver, Canada. Association for Computational Linguistics.

Graves, Alex, Greg Wayne, and Ivo Danihelka. 2014. Neural turing machines. *CoRR*.

Graves, Alex, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwińska, Sergio Gómez Colmenarejo, Edward Grefenstette, Tiago Ramalho, John Agapiou, et al. 2016. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538(7626):471–476.

Henaff, Mikael, Jason Weston, Arthur Szlam, Antoine Bordes, and Yann LeCun. 2017. Tracking the world state with recurrent entity networks. *International Conference on Learning*

Representations.

Hermann, Karl Moritz, Tomáš Kočiský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, pages 1693–1701, Cambridge, MA, USA. MIT Press.

Hill, Felix, Antoine Bordes, Sumit Chopra, and Jason Weston. 2016. The goldilocks principle: Reading children's books with explicit memory representations. *International Conference on Learning Representations*.

Kadlec, Rudolf, Martin Schmid, Ondřej Bajgar, and Jan Kleindienst. 2016. Text understanding with the attention sum reader network. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 908–918, Berlin, Germany. Association for Computational Linguistics.

Kingma, Diederik and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *International Conference for Learning Representations*.

Kobayashi, Sosuke, Ran Tian, Naoaki Okazaki, and Kentaro Inui. 2016. Dynamic entity representation with max-pooling improves machine reading. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 850–855, San Diego, California, USA. Association for Computational Linguistics.

Kumar, Ankit, Ozan Irsoy, Jonathan Su, James Bradbury, Robert English, Brian Pierce, Peter Ondruska, Ishaan Gulrajani, and Richard Socher. 2015. Ask me anything: Dynamic memory networks for natural language processing. *CoRR, abs/1506.07285*.

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature*, 521(7553):436–444.

Onishi, Takeshi, Hai Wang, Mohit Bansal, Kevin Gimpel, and David McAllester. 2016. Who did what: A large-scale person-centered cloze dataset. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2230–2235, Austin, Texas, USA. Association for Computational Linguistics.

Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar.

Rajpurkar, Pranav, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas, USA. Association for Computational Linguistics.

Richardson, Matthew, Christopher J.C. Burges, and Erin Renshaw. 2013. Mctest: A challenge dataset for the open-domain machine comprehension of text. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 193–203, Seattle, USA. Association for Computational Linguistics.

Sukhbaatar, Sainbayar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. End-to-end memory networks. In *Advances in neural information processing systems*, pages 2440–2448.

Trischler, Adam, Zheng Ye, Xingdi Yuan, Philip Bachman, Alessandro Sordoni, and Kaheer Suleman. 2016. Natural language comprehension with the epireader. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 128–137, Austin, Texas, USA. Association for Computational Linguistics.

Vinyals, Oriol, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28*. Curran Associates, Inc., pages 2692–2700.

Weston, Jason, Antoine Bordes, Sumit Chopra, Alexander M. Rush, Bart van Merriënboer, Armand Joulin, and Tomas Mikolov. 2016. Towards ai-complete question answering: A set of prerequisite toy tasks. *International Conference on Learning Representations*.

Weston, Jason, Sumit Chopra, and Antoine Bordes. 2015. Memory networks. *International Conference on Learning Representations*.

Xiong, Caiming, Stephen Merity, and Richard Socher. 2016. Dynamic memory networks for visual and textual question answering. *CoRR*, 1603.