Gated End-to-End Memory Networks

Fei Liu [*] [†]
The University of Melbourne
Victoria, Australia
fliu3@student.unimelb.edu.au

Julien Perez [†]
Xerox Research Centre Europe
Grenoble, France
julien.perez@xrce.xerox.com

Abstract

Machine reading using differentiable reasoning models has recently shown remarkable progress. In this context, End-to-End trainable Memory Networks (MemN2N) have demonstrated promising performance on simple natural language based reasoning tasks such as factual reasoning and basic deduction. However, other tasks, namely multi-fact question-answering, positional reasoning or dialog related tasks, remain challenging particularly due to the necessity of more complex interactions between the memory and controller modules composing this family of models. In this paper, we introduce a novel end-to-end memory access regulation mechanism inspired by the current progress on the connection short-cutting principle in the field of computer vision. Concretely, we develop a Gated End-to-End trainable Memory Network architecture (GMemN2N). From the machine learning perspective, this new capability is learned in an end-to-end fashion without the use of any additional supervision signal which is, as far as our knowledge goes, the first of its kind. Our experiments show significant improvements on the most challenging tasks in the 20 bAbI dataset, without the use of any domain knowledge. Then, we show improvements on the Dialog bAbI tasks including the real human-bot conversion-based Dialog State Tracking Challenge (DSTC-2) dataset. On these two datasets, our model sets the new state of the art.

[*] Work carried out as an intern at XRCE
[†] Equal contribution

1 Introduction

Deeper Neural Network models are more difficult to train and recurrency tends to complexify this optimization problem (Srivastava et al., 2015b). While Deep Neural Network architectures have shown superior performance in numerous areas, such as image, speech recognition and more recently text, the complexity of optimizing such large and non-convex parameter sets remains a challenge. Indeed, the so-called vanishing/exploding gradient problem has been mainly addressed using: 1. algorithmic responses, e.g., normalized initialization strategies (LeCun et al., 1998; Glorot and Bengio, 2010); 2. architectural ones, e.g., intermediate normalization layers which facilitate the convergence of networks composed of tens of hidden layers (He et al., 2015; Saxe et al., 2014). Another problem of memory-enhanced neural models is the necessity of regulating memory access at the controller level. Memory access operations can be supervised (Kumar et al., 2016) and the number of times they are performed tends to be fixed apriori (Sukhbaatar et al., 2015), a design choice which tends to be based on the presumed degree of difficulty of the task in question. Inspired by the recent success of object recognition in the field of computer vision (Srivastava et al., 2015a; Srivastava et al., 2015b), we investigate the use of a *gating* mechanism in the context of End-to-End Memory Networks (MemN2N) (Sukhbaatar et al., 2015) in order to regulate the access to the memory blocks in a differentiable fashion. The formulation is realized by gated connections between the memory access layers and the controller stack of a MemN2N. As a result, the model is able to dynamically determine how and when to skip its memory-based reasoning process.

Roadmap: Section 2 reviews state-of-the-art Memory Network models, connection short-

Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers, pages 1–10,
Valencia, Spain, April 3-7, 2017.

cutting in neural networks and memory dynamics. In Section 3, we propose a differentiable gating mechanism in MemN2N. Section 4 and 5 present a set of experiments on the 20 bAbI reasoning tasks and the Dialog bAbI dataset. We report new state-of-the-art results on several of the most challenging tasks of the set, namely positional reasoning, 3-argument relation and the DSTC-2 task while maintaining equally competitive performance on the rest.

2 Related Work

This section starts with an introduction of the primary components of MemN2N. Then, we review two key elements relevant to this work, namely shortcut connections in neural networks and memory dynamics in such models.

2.1 End-to-End Memory Networks

The MemN2N architecture, introduced by Sukhbaatar et al. (2015), consists of two main components: supporting memories and final answer prediction. Supporting memories are in turn comprised of a set of input and output memory representations with memory cells. The input and output memory cells, denoted by $\boldsymbol{m}_i$ and $\boldsymbol{c}_i$, are obtained by transforming the input context $x_1, \ldots, x_n$ (or stories) using two embedding matrices $\boldsymbol{A}$ and $\boldsymbol{C}$ (both of size $d \times |V|$ where d is the embedding size and $|V|$ the vocabulary size) such that $\boldsymbol{m}_i = \boldsymbol{A}\Phi(x_i)$ and $\boldsymbol{c}_i = \boldsymbol{C}\Phi(x_i)$ where $\Phi(\cdot)$ is a function that maps the input into a bag of dimension $|V|$. Similarly, the question q is encoded using another embedding matrix $\boldsymbol{B} \in \mathbb{R}^{d \times |V|}$, resulting in a question embedding $\boldsymbol{u} = \boldsymbol{B}\Phi(q)$. The input memories $\{\boldsymbol{m}_i\}$, together with the embedding of the question $\boldsymbol{u}$, are utilized to determine the relevance of each of the stories in the context, yielding a vector of attention weights

$$p_i = \text{softmax}(\boldsymbol{u}^\top \boldsymbol{m}_i) \tag{1}$$

where $\text{softmax}(a_i) = \frac{e^{a_i}}{\sum_j e^{a_j}}$. Subsequently, the response $\boldsymbol{o}$ from the output memory is constructed by the weighted sum:

$$\boldsymbol{o} = \sum_i p_i \boldsymbol{c}_i \tag{2}$$

For more difficult tasks demanding multiple supporting memories, the model can be extended to include more than one set of input/output memories by stacking a number of memory layers. In this setting, each memory layer is named a hop and the $(k+1)^{\text{th}}$ hop takes as input the output of the k^{th} hop:

$$\boldsymbol{u}^{k+1} = \boldsymbol{o}^k + \boldsymbol{u}^k \tag{3}$$

Lastly, the final step, the prediction of the answer to the question q, is performed by

$$\hat{\boldsymbol{a}} = \text{softmax}(\boldsymbol{W}(\boldsymbol{o}^K + \boldsymbol{u}^K)) \tag{4}$$

where $\hat{\boldsymbol{a}}$ is the predicted answer distribution, $\boldsymbol{W} \in \mathbb{R}^{|V| \times d}$ is a parameter matrix for the model to learn and K the total number of hops.

2.2 Shortcut Connections

Shortcut connections have been studied from both the theoretical and practical point of view in the general context of neural network architectures (Bishop, 1995; Ripley, 2007). More recently Residual Networks (He et al., 2016) and Highway Networks (Srivastava et al., 2015a; Srivastava et al., 2015b) have been almost simultaneously proposed. While the former utilizes a residual calculus, the latter formulates a differentiable gateway mechanism as proposed in Long-Short Terms Memory Networks (Hochreiter and Schmidhuber, 1997) in order to cope with long-term dependency issues in the dataset in an end-to-end trainable manner. These two mechanisms were proposed as a structural solution to the so-called vanishing gradient problem by allowing the model to shortcut its layered transformation structure when necessary.

2.3 Memory Dynamics

The necessity of dynamically regulating the interaction between the so-called controller and the memory blocks of a Memory Network model has been studied in (Kumar et al., 2016; Xiong et al., 2016). In these works, the number of exchanges between the controller stack and the memory module of the network is either monitored in a hard supervised manner in the former or fixed apriori in the latter.

In this paper, we propose an end-to-end supervised model, with an automatically learned gating mechanism, to perform dynamic regulation of memory interaction. The next section presents the formulation of this new Gated End-to-End Memory Networks (GMemN2N). This contribution can be placed in parallel to the recent transition from Memory Networks with hard attention mechanism

(Weston et al., 2015) to MemN2N with attention values obtained by a softmax function and end-to-end supervised (Sukhbaatar et al., 2015).

3 Gated End-to-End Memory Network

In this section, the elements behind residual learning and highway neural models are given. Then, we introduce the proposed model of memory access gating in a MemN2N.

3.1 Highway and Residual Networks

Highway Networks, first introduced by Srivastava et al. (2015a), include a transform gate T and a carry gate C, allowing the network to learn how much information it should transform or carry to form the input to the next layer. Suppose the original network is a plain feed-forward neural network:

$$\boldsymbol{y} = \mathrm{H}(\boldsymbol{x}) \tag{5}$$

where $\mathrm{H}(\boldsymbol{x})$ is a non-linear transformation of its input $\boldsymbol{x}$. The generic form of Highway Networks is formulated as:

$$\boldsymbol{y} = \mathrm{H}(\boldsymbol{x}) \odot \mathrm{T}(\boldsymbol{x}) + \boldsymbol{x} \odot \mathrm{C}(\boldsymbol{x}) \tag{6}$$

where the transform and carry gates, $\mathrm{T}(\boldsymbol{x})$ and $\mathrm{C}(\boldsymbol{x})$, are defined as non-linear transformation functions of the input $\boldsymbol{x}$ and $\odot$ the Hadamard product. As suggested in (Srivastava et al., 2015a; Srivastava et al., 2015b), we choose to focus, in the following of this paper, on a simplified version of Highway Networks where the carry gate is replaced by $1 - \mathrm{T}(\boldsymbol{x})$:

$$\boldsymbol{y} = \mathrm{H}(\boldsymbol{x}) \odot \mathrm{T}(\boldsymbol{x}) + \boldsymbol{x} \odot (\mathbf{1} - \mathrm{T}(\boldsymbol{x})) \tag{7}$$

where $\mathrm{T}(x) = \sigma(\boldsymbol{W}_T\boldsymbol{x} + \boldsymbol{b}_T)$ and σ is the sigmoid function. In fact, Residual Networks can be viewed as a special case of Highway Networks where both the transform and carry gates are substituted by the identity mapping function:

$$\boldsymbol{y} = \mathrm{H}(\boldsymbol{x}) + \boldsymbol{x} \tag{8}$$

thereby forming a hard-wired shortcut connection $\boldsymbol{x}$.

3.2 Gated End-to-End Memory Networks

Arguably, Equation (3) can be considered as a form of residuality with $\boldsymbol{o}^k$ working as the residual function and $\boldsymbol{u}^k$ the shortcut connection. However, as discussed in (Srivastava et al., 2015b), in contrast to the hard-wired skip connection in Residual Networks, one of the advantages of Highway Networks is the adaptive gating mechanism, capable of learning to dynamically control the information flow based on the current input. Therefore, we adopt the idea of the adaptive gating mechanism of Highway Networks and integrate it into MemN2N. The resulting model, named *Gated End-to-End Memory Networks* (GMemN2N) and illustrated in Figure 1, is capable of dynamically conditioning the memory reading operation on the controller state $\boldsymbol{u}^k$ at each hop. Concretely, we reformulate Equation (3) into:

$$\mathrm{T}^k(\boldsymbol{u}^k) = \sigma(\boldsymbol{W}_T^k\boldsymbol{u}^k + \boldsymbol{b}_T^k) \tag{9}$$

$$\boldsymbol{u}^{k+1} = \boldsymbol{o}^k \odot \mathrm{T}^k(\boldsymbol{u}^k) + \boldsymbol{u}^k \odot (\mathbf{1} - \mathrm{T}^k(\boldsymbol{u}^k)) \tag{10}$$

where $\boldsymbol{W}_T^k$ and $\boldsymbol{b}^k$ are the hop-specific parameter matrix and bias term for the k^{th} hop and $\mathrm{T}^k(x)$ the transform gate for the k^{th} hop. Similar to the two weight tying schemes of the embedding matrices introduced in (Sukhbaatar et al., 2015), we also explore two types of constraints on $\boldsymbol{W}_T^k$ and $\boldsymbol{b}_T^k$:

1. **Global:** all the weight matrices $\boldsymbol{W}_T^k$ and bias terms $\boldsymbol{b}_T^k$ are shared across different hops, i.e., $\boldsymbol{W}_T^1 = \boldsymbol{W}_T^2 = \ldots = \boldsymbol{W}_T^K$ and $\boldsymbol{b}_T^1 = \boldsymbol{b}_T^2 = \ldots = \boldsymbol{b}_T^K$.
2. **Hop-specific:** each hop has its specific weight matrix $\boldsymbol{W}_T^k$ and bias term $\boldsymbol{b}_T^k$ for $k \in [1, K]$ and they are optimized independently.

4 QA bAbI Experiments

In this section, we first describe the natural language reasoning dataset we use in our experiments. Then, the experimental setup is detailed. Lastly, we present the results and analyses.

4.1 Dataset and Data Preprocessing

The 20 bAbI tasks (Weston et al., 2016) have been employed for the experiments (using v1.2 of the dataset). In this synthetically generated dataset, a given QA task consists of a set of statements, followed by a question whose answer is typically a single word (in a few tasks, answers are a set of words). The answer is available to the model at training time but must be predicted at test time. The dataset consists of 20 different tasks with various emphases on different forms of reasoning. For each question, only a certain subset of the statements contains information needed for the answer, and the rest are essentially irrelevant distractors.

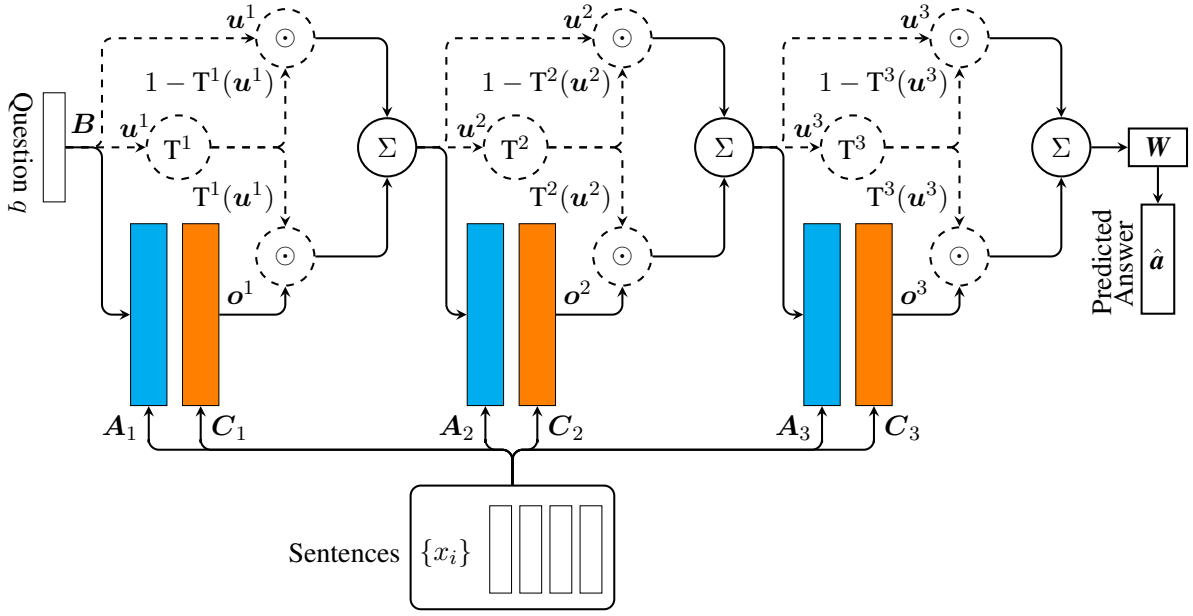

Figure 1: Illustration of the proposed GMemN2N model with 3 hops. Dashed lines indicate elements different from MemN2N (Sukhbaatar et al., 2015).

As in (Sukhbaatar et al., 2015), our model is fully end-to-end trained without any additional supervision other than the answers themselves. Formally, for one of the 20 QA tasks, we are given example problems, each having a set of I sentences $\{x_i\}$ (where $I \leq 320$), a question sentence q and answer a. Let the j^{th} word of sentence i be x_{ij}, represented by a one-hot vector of length $|V|$. The same representation is used for the question q and answer a. Two versions of the data are used, one that has 1,000 training problems per task and the other with 10,000 per task.

4.2 Training Details

As suggested in (Sukhbaatar et al., 2015), 10% of the bAbI training set was held-out to form a validation set for hyperparameter tuning. Moreover, we use the so-called position encoding, adjacent weight tying, and temporal encoding with 10% random noise. Stochastic gradient descent is used for training and the learning rate η is initially assigned a value of 0.005 with exponential decay applied every 25 epochs by $\eta/2$ until 100 epochs are reached. Linear start is used in all our experiments as proposed by Sukhbaatar et al. (2015). With linear start, the softmax in each memory layer is removed and re-inserted after 20 epochs. Batch size is set to 32 and gradients with an ℓ_2 norm larger than 40 are divided by a scalar to have norm 40. All weights are initialized randomly from a Gaussian distribution with zero mean and $\sigma = 0.1$ except for the transform gate bias $\boldsymbol{b}_T^k$ which we empirically set the mean to 0.5. Only the most recent 50 sentences are fed into the model as the memory and the number of memory hops is 3. In all our experiments, we use the embedding size $d = 20$. Note that we re-use the same hyperparameter configuration as in (Sukhbaatar et al., 2015) and no grid search is performed.

As a large variance in the performance of the model can be observed on some tasks, we follow (Sukhbaatar et al., 2015) and repeat each training 100 times with different random initializations and select the best system based on the validation performance. On the 10k dataset, we repeat each training 30 times due to time constraints. Concerning the model implementation, while there are minor differences between the results of our implementation of MemN2N and those reported in (Sukhbaatar et al., 2015), the overall performance is equally competitive and, in some cases, better. It should be noted that v1.1 of the dataset was used whereas in this work, we employ the latest v1.2. It is therefore deemed necessary that we present the performance results of our implementation of MemN2N on the v1.2 dataset. To facilitate fair comparison, we select our implementation of MemN2N as the baseline as we believe that it is indicative of the true performance of MemN2N on v1.2 of the dataset.

Task	1k				10k			
	MemN2N	Our MemN2N	GMemN2N +global	GMemN2N +hop	MemN2N	Our MemN2N	GMemN2N +global	GMemN2N +hop
1: 1 supporting fact	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
2: 2 supporting facts	91.7	89.9	88.7	**91.9**	99.7	99.7	**100.0**	**100.0**
3: 3 supporting facts	59.7	58.5	53.2	**61.2**	90.7	89.1	94.7	**95.5**
4: 2 argument relations	97.2	99.0	99.3	**99.6**	**100.0**	**100.0**	**100.0**	**100.0**
5: 3 argument relations	86.9	86.6	98.1	**99.0**	99.4	99.4	**99.9**	99.8
6: yes/no questions	**92.4**	92.1	92.0	91.6	**100.0**	**100.0**	96.7	**100.0**
7: counting	82.7	83.3	**83.8**	82.2	96.3	96.8	96.7	**98.2**
8: lists/sets	**90.0**	89.0	87.8	87.5	99.2	98.1	**99.9**	99.7
9: simple negation	86.8	**90.3**	88.2	89.3	99.2	99.1	**100.0**	**100.0**
10: indefinite knowledge	**84.9**	84.6	80.1	83.5	97.6	98.0	**99.9**	99.8
11: basic coreference	99.1	99.7	99.8	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
12: conjunction	99.8	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
13: compound coreference	99.6	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
14: time reasoning	98.3	**99.6**	98.5	98.8	**100.0**	**100.0**	**100.0**	**100.0**
15: basic deduction	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
16: basic induction	98.7	**99.9**	99.8	**99.9**	99.6	**100.0**	**100.0**	**100.0**
17: positional reasoning	49.0	48.1	**60.2**	58.3	59.3	62.1	68.8	**72.2**
18: size reasoning	88.9	89.7	**91.8**	90.8	93.3	**93.4**	92.0	91.5
19: path finding	**17.2**	11.3	10.3	11.5	33.5	47.2	54.8	**69.0**
20: agent's motivation	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
Average	86.1	86.1	86.6	**87.3**	93.4	94.1	95.2	**96.3**

Table 1: Accuracy (%) on the 20 QA tasks for models using 1k and 10k training examples. MemN2N:(Sukhbaatar et al., 2015). Our MemN2N: our implementation of MemN2N. GMemN2N +global: GMemN2N with global weight tying. GMemN2N +hop: GMemN2N with hop-specific weight tying. **Bold** highlights best performance. Note that in (Sukhbaatar et al., 2015), v1.1 of the dataset was used.

4.3 Results

Performance results on the 20 bAbI QA dataset are presented in Table 1. For comparison purposes, we still present MemN2N (Sukhbaatar et al., 2015) in Table 1 but accompany it with the accuracy obtained by our implementation of the same model with the same experimental setup on v1.2 of the dataset in the column "Our MemN2N" for both the 1k and 10k versions of the dataset. In contrast, we also list the results achieved by GMemN2N with global and hop-specific weight constraints in the GMemN2N columns.

GMemN2N achieves substantial improvements on task 5 and 17. The performance of GMemN2N is greatly improved, a substantial gain of more than 10 in absolute accuracy.

Global vs. hop-specific weight tying. Compared with the global weight tying scheme on the weight matrices of the gating mechanism, applying weight constraints in a hop-specific fashion generates a further boost in performance consistently on both the 1k and 10k datasets.

State-of-the-art performance on both the 1k and 10k dataset. The best performing GMemN2N model achieves state-of-the-art performance, an average accuracy of 87.3 on the 1k dataset and 96.3 on the 10k variant. This is a solid improvement compared to MemN2N and a step closer to the strongly supervised models described in (Weston et al., 2015). Notice that the highest average accuracy of the original MemN2N model on the 10k dataset is 95.8. However, it was attained by a model with layer-wise weight tying, not adjacent weight tying as adopted in this work, and, more importantly, a much larger embedding size $d = 100$ (therefore not shown in Table 1). In comparison, it is worth noting that the proposed GMemN2N model, a much smaller model with embeddings of size 20, is capable of achieving better accuracy.

5 Dialog bAbI Experiments

In addition to the text understanding and reasoning tasks presented in Section 4, we further examine the effectiveness of the proposed GMemN2N model on a collection of goal-oriented dialog tasks (Bordes and Weston, 2016). First, we briefly describe the dataset. Next, we outline the training details. Finally, experimental results are presented with analyses.

5.1 Dataset and Data Preprocessing

In this work, we adopt the goal-oriented dialog dataset developed by Bordes and Weston (2016) organized as a set of tasks. The tasks in this dataset can be divided into 6 categories with each group focusing on a specific objective: 1. issuing API calls, 2. updating API calls, 3. displaying options, 4. providing extra-information, 5. conducting full dialogs (the aggregation of the first 4 tasks), 6. Dialog State Tracking Challenge 2 corpus (DSTC-2). The first 5 tasks are synthetically generated based on a knowledge base consisting of facts which define all the restaurants and their associated properties (7 types, such as location and price range). The generated texts are in the form of conversation between a user and a bot, each of which is designed with a clear yet different objective (all involved in a restaurant reservation scenario). This dataset essentially tests the capacity of end-to-end dialog systems to conduct dialog with various goals. Each dialog starts with a user request with subsequent alternating user-bot utterances and it is the duty of a model to understand the intention of the user and respond accordingly. In order to test the capability of a system to cope with entities not appearing in the training set, a different set of test sets, named out-of-vocabulary (OOV) test sets, are constructed separately. In addition, a supplementary dataset, task 6, is provided with real human-bot conversations, also in the restaurant domain, which is derived from the second Dialog State Tracking Challenge (Henderson et al., 2014). It is important to notice that the answers in this dataset may no longer be a single word but can be comprised of multiple ones.

5.2 Training Details

At a certain given time t, a memory-based model takes the sequence of utterances $c_1^u, c_1^r, c_2^u, c_2^r, \ldots, c_{t-1}^u, c_{t-1}^r$ (alternating between the user c_i^u and the system response c_i^r) as the stories and c_t^u as the question. The goal of the model is to predict the response c_t^r.

As answers may be composed of multiple words, following (Bordes and Weston, 2016), we replace the final prediction step in Equation (4) with:

$$\hat{\boldsymbol{a}} = \text{softmax}(\boldsymbol{u}^\top \boldsymbol{W}' \Phi(\boldsymbol{y}_1), \ldots, \boldsymbol{u}^\top \boldsymbol{W}' \Phi(\boldsymbol{y}_{|C|}))$$

where $\boldsymbol{W}' \in \mathbb{R}^{d \times |V|}$ is the weight parameter matrix for the model to learn, $\boldsymbol{u} = \boldsymbol{o}^K + \boldsymbol{u}^K$ (K is the total number of hops), $\boldsymbol{y}_i$ is the i^{th} response in the candidate set C such that $\boldsymbol{y}_i \in C$, $|C|$ the size of the candidate set, and $\Phi(\cdot)$ a function which maps the input text into a bag of dimension $|V|$.

As in (Bordes and Weston, 2016), we extend Φ by several key additional features. First, two features marking the identity of the speaker of a particular utterance (user or model) are added to each of the memory slots. Second, we expand the feature representation function Φ of candidate responses with 7 additional features, each, focusing on one of the 7 properties associated with any restaurants, indicating whether there are any exact matches between words occurring in the candidate and those in the question or memory. These 7 features are referred to as the *match* features.

Apart from the modifications described above, we carry out the experiments using the same experimental setup described in Section 4.2. We also constrain ourselves to the hop-specific weight tying scheme in all our experiments since GMemN2N benefits more from it than global weight tying as shown in Section 4.3. As in (Sukhbaatar et al., 2015), since the memory-based models are sensitive to parameter initialization, we repeat each training 10 times and choose the best system based on the performance on the validation set.

5.3 Results

Performance results on the Dialog bAbI dataset are shown in Table 2, measured using both per-response accuracy and per-dialog accuracy (given in parentheses). While per-response accuracy calculates the percentage of correct responses, per-dialog accuracy, where a dialog is considered to be correct if and only if every response within it is correct, counts the percentage of correct dialogs. Task 1-5 are presented in the upper half of the table while the same tasks in the OOV setting are in the lower half with the dialog state tracking task as task 6 at the bottom. We

Task	MemN2N	GMemN2N	MemN2N +match	GMemN2N +match
T1: Issuing API calls	99.9 (99.6)	**100.0** (100.0)	**100.0** (100.0)	**100.0** (100.0)
T2: Updating API calls	**100.0** (100.0)	**100.0** (100.0)	98.3 (83.9)	**100.0** (100.0)
T3: Displaying options	**74.9** (2.0)	**74.9** (0.0)	**74.9** (0.0)	**74.9** (0.0)
T4: Providing information	59.5 (3.0)	57.2 (0.0)	**100.0** (100.0)	**100.0** (100.0)
T5: Full dialogs	96.1 (49.4)	96.3 (52.5)	93.4 (19.7)	**98.0** (72.5)
Average	86.1 (50.8)	85.7 (50.5)	93.3 (60.7)	**94.6** (74.5)
T1 (OOV): Issuing API calls	72.3 (0.0)	82.4 (0.0)	96.5 (82.7)	**100.0** (100.0)
T2 (OOV): Updating API calls	78.9 (0.0)	78.9 (0.0)	**94.5** (48.4)	94.2 (47.1)
T3 (OOV): Displaying options	74.4 (0.0)	**75.3** (0.0)	75.2 (0.0)	75.1 (0.0)
T4 (OOV): Providing information	57.6 (0.0)	57.0 (0.0)	**100.0** (100.0)	**100.0** (100.0)
T5 (OOV): Full dialogs	65.5 (0.0)	66.7 (0.0)	77.7 (0.0)	**79.4** (0.0)
Average	69.7 (0.0)	72.1 (0.0)	88.8 (46.2)	**89.7** (49.4)
T6: Dialog state tracking 2	41.1 (0.0)	47.4 (1.4)	41.0 (0.0)	**48.7** (1.4)

Table 2: Per-response accuracy and per-dialog accuracy (in parentheses) on the Dialog bAbI tasks. MemN2N: (Bordes and Weston, 2016). +match indicates the use of the match features in Section 5.2.

choose (Bordes and Weston, 2016) as the baseline which achieves the current state of the art on these tasks.

GMemN2N with the match features sets a new state of the art on most of the tasks. Other than on task T2 (OOV) and T3 (OOV), GMemN2N with the match features scores the best per-response and per-dialog accuracy. Even on T2 (OOV) and T3 (OOV), the model generates rather competitive results and remains within 0.3% of the best performance. Overall, the best average per-response accuracy in both the OOV and non-OOV categories is attained by GMemN2N.

GMemN2N with the match features significantly improves per-dialog accuracy on T5. A breakthrough in per-dialog accuracy on T5 from less than 20% to over 70%.

GMemN2N succeeds in improving the performance on the more practical task T6. With or without the match features, GMemN2N achieves a substantial boost in per-response accuracy on T6. Given that T6 is derived from a dataset based on real human-bot conversations, not synthetically generated, the performance gain, although far from perfect, highlights the effectiveness of GMemN2N in practical scenarios and constitutes an encouraging starting point towards end-to-end dialog system learning.

The effectiveness of GMemN2N is more pronounced on the more challenging tasks. The performance gains on T5, T5 (OOV) and T6, compared with the rest of the tasks, are more pronounced. Regarding the performance of MemN2N, these tasks are relatively more challenging than the rest, suggesting that the adaptive gating mechanism in GMemN2N is capable of managing complex information flow while doing little damage on easier tasks.

6 Visualization and Analysis

In addition to the quantitative results, we further look into the memory regulation mechanism learned by the GMemN2N model. Figure 2 presents the three most frequently observed patterns of the $\mathrm{T}^k(\boldsymbol{u}^k)$ vectors for each of the 3 hops in a model trained on T6 of the Dialog bAbI dataset with an embedding dimension of 20. Each row corresponds to the gate values at a specific hop whereas each column represents a given embedding dimension. The pattern on the top indicates that the model tends to only access memory in the first and third hop. In contrast, the middle and bottom patterns only focus on the memory in either the first or last hop respectively. Figure 3 is a t-SNE projection (Maaten and Hinton, 2008) of the flattened $[\mathrm{T}^1(\boldsymbol{u}^1); \mathrm{T}^2(\boldsymbol{u}^2); \mathrm{T}^3(\boldsymbol{u}^3)]$ vectors obtained on the test set of the same dialog task with points corresponding to the correct and incorrect responses in red and blue respectively. Despite the relative uniform distribution of the wrong answer points, the correct ones tend to form clusters that suggest the frequently observed behavior of a successful

Story	Support	MemN2N			GMemN2N		
		Hop 1	Hop 2	Hop 3	Hop 1	Hop 2	Hop 3
Fred took the football there.		0.05	0.10	0.07	0.06	0.00	0.00
Fred journeyed to the hallway.		0.45	0.09	0.01	0.00	0.00	0.00
Fred passed the football to Mary.	yes	0.10	0.64	0.93	0.29	1.00	1.00
Mary dropped the football.		0.40	0.17	0.00	0.64	0.00	0.00
Avg. transform gate cell values, $\sum_i \mathrm{T}^k(\boldsymbol{u}^k)_i/d$		N/A	N/A	N/A	0.22	0.23	0.45
Question: Who gave the football? Answer: Fred, MemN2N: Mary, GMemN2N: Fred							

Table 3: MemN2N vs. GMemN2N- bAbI dataset - Task 5 - 3 argument relations

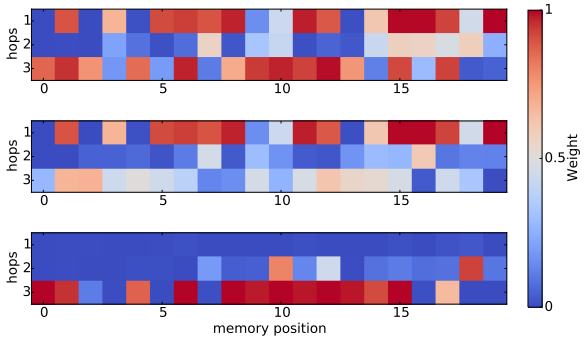

Figure 2: 3 most frequently observed gate value $\mathrm{T}^k(\boldsymbol{u}^k)$ patterns on T6 of the Dialog bAbI dataset

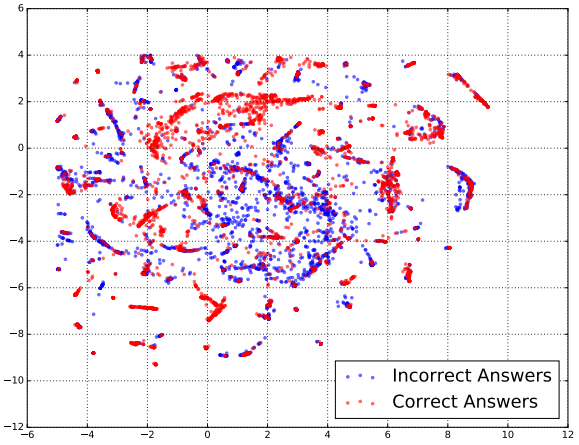

Figure 3: t-SNE scatter plot of the flattened gate values

inference. Lastly, Table 3 shows the comparison of the attention shifting process between MemN2N and GMemN2N on a story on bAbI task 5 (3 argument relations). Not only does GMemN2N manage to focus more accurately on the supporting fact than MemN2N, it has also learned to rely less in this case on hop 1 and 2 by assigning smaller transform gate values. In contrast, MemN2N carries false and misguiding information (caused by the distracting attention mechanism) accumulated from the previous hops, which eventually led to the wrong prediction of the answer.

7 Related Reading Tasks

Apart from the datasets adopted in our experiments, the CNN/Daily Mail (Hermann et al., 2015) has been used for the task of machine reading formalized as a problem of text extraction from a source conditioned on a given question. However, as pointed out in (Chen et al., 2016), this dataset not only is noisy but also requires little reasoning and inference, which is evidenced by a manual analysis of a randomly selected subset of the questions, showing that only 2% of the examples call for multi-sentence inference. Richardson et al. (2013) constructed an open-domain reading comprehension task, named MCTest. Although this corpus demands various degrees of reasoning capabilities from multiple sentences, its rather limited size (660 paragraphs, each associated with 4 questions) renders training statistical models infeasible (Chen et al., 2016). Children's Book Test (CBT) (Hill et al., 2015) was designed to measure the ability of models to exploit a wide range of linguistic context. Despite the claim in (Sukhbaatar et al., 2015) that increasing the number of hops is crucial for the performance improvements on some tasks, which can be seen as enabling MemN2N to accommodate more supporting facts, making such performance boost particularly more pronounced on those tasks requiring complex reasoning, Hill et al. (2015) admittedly reported little improvement in performance by stacking more hops and chose a single-hop MemN2N. This suggests that the necessity of multi-sentence based reasoning in this dataset is not mandatory. In the future, we plan to investigate into larger dialog datasets such as (Lowe et al., 2015).

8 Conclusion and Future Work

In this paper, we have proposed and developed what is, as far as our knowledge goes, the first attempt at incorporating an iterative memory access control to an end-to-end trainable memory-enhanced neural network architecture. We showed the added value of our proposition on a set of, natural language based, state-of-the-art reasoning tasks. Then, we offered a first interpretation of the resulting capability by analyzing the attention shifting mechanism and connection short-cutting behavior of the proposed model. In future work, we will investigate the use of such mechanism in the field of language modeling and more generally on the paradigm of sequential prediction and predictive learning. Furthermore, we plan to look into the impact of this method on the recently introduced Key-Value Memory Networks (Miller et al., 2016) on larger and semi-structured corpus.

References

Christopher M. Bishop. 1995. *Neural Networks for Pattern Recognition*. Oxford University Press.

Antoine Bordes and Jason Weston. 2016. Learning end-to-end goal-oriented dialog. *arXiv preprint arXiv:1605.07683*.

Danqi Chen, 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 (ACL 2016)*, pages 2358–2367, Berlin, Germany.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS 2010)*, pages 249–256, Sardinia, Italy.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV 2015)*, pages 1026–1034, Santiago, Chile.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *the 29th IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016)*, Las Vegas, USA.

Matthew Henderson, Blaise Thomson, and Jason D. Williams. 2014. The second dialog state tracking challenge. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL 2014)*, pages 263–272, Philadelphia, USA.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Proceedings of Advances in Neural Information Processing Systems (NIPS 2015)*, pages 1684–1692, Barcelona, Spain.

Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. 2015. The goldilocks principle: Reading children's books with explicit memory representations. In *Proceedings of the 4th International Conference on Learning Representations (ICLR 2016)*, San Juan, Puerto Rico.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Ankit Kumar, Ozan Irsoy, Jonathan Su, James Bradbury, Robert English, Brian Pierce, Peter Ondruska, Ishaan Gulrajani, and Richard Socher. 2016. Ask me anything: Dynamic memory networks for natural language processing. In *Proceedings of the 33rd International Conference on Machine Learning (ICML 2016)*, New York, USA.

Yann LeCun, Leon Bottou, Genevieve B. Orr, and Klaus Robert Müller. 1998. Efficient backprop. *Neural Networks: Tricks of the Trade*, pages 9–50.

Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015. The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. In *Proceedings of the 16th Annual SIGdial Meeting on Discourse and Dialogue (SIGDIAL 2015)*, Prague, Czech Republic.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605.

Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016)*, Austin, USA.

Matthew Richardson, 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 (EMNLP 2013)*, pages 193–203, Seattle, USA.

Brian D. Ripley. 2007. *Pattern recognition and neural networks*. Cambridge university press.

Andrew M Saxe, James L McClelland, and Surya Ganguli. 2014. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In *Proceedings of the 2nd International Conference on Learning Representations (ICLR 2014)*, Banff, Canada.

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015a. Highway networks. *arXiv preprint arXiv:1505.00387*.

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015b. Training very deep networks. In *Proceedings of Advances in Neural Information Processing Systems (NIPS 2015)*, pages 2377–2385, Montréal, Canada.

Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. End-to-end memory networks. In *Proceedings of Advances in Neural Information Processing Systems (NIPS 2015)*, pages 2440–2448, Montréal, Canada.

Jason Weston, Sumit Chopra, and Antoine Bordes. 2015. Memory networks. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015)*, San Diego, USA.

Jason Weston, 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. In *Proceedings of the 4th International Conference on Learning Representations (ICLR 2016)*, San Juan, Puerto Rico.

Caiming Xiong, Stephen Merity, and Richard Socher. 2016. Dynamic memory networks for visual and textual question answering. In *Proceedings of the 33rd International Conference on Machine Learning (ICML 2016)*, pages 2397–2406, New York, USA.