HOW TO LEARN (AND HOW NOT TO LEARN) MULTI-HOP REASONING WITH MEMORY NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Answering questions about a text frequently requires aggregating information from multiple places in that text. End-to-end neural network models, the dominant approach in the current literature, can theoretically learn how to distill and manipulate representations of the text without explicit supervision about how to do so. We investigate a canonical architecture for this task, the memory network, and analyze how effective it really is in the context of three multi-hop reasoning settings. In a simple synthetic setting, the path-finding task of the bAbI dataset (Weston et al., 2015), the model fails to learn the correct reasoning without additional supervision of its attention mechanism. However, with this supervision, it can perform well. On a real text dataset, WikiHop (Welbl et al., 2017), the memory network gives nearly state-of-the-art performance, but does so without using its multi-hop capabilities. A tougher anonymized version of the WikiHop dataset is qualitatively similar to bAbI: the model fails to perform well unless it has additional supervision. We hypothesize that many "multi-hop" architectures do not truly learn this reasoning as advertised, though they *could* learn this reasoning if appropriately supervised.[1]

1 INTRODUCTION

Question answering from text is a key challenge problem for NLP that tests whether models can extract information based on a query. Recent new datasets (Richardson et al., 2013; Hill et al., 2015; Hermann et al., 2015; Rajpurkar et al., 2016) and new models (Seo et al., 2016; Shen et al., 2017; Yu et al., 2018) have dramatically advanced the state-of-the-art in this area. However, some QA tasks, such as SQuAD, only simple pattern matching to solve (Weissenborn et al., 2017). One thread of recent work has emphasized multi-hop reasoning in particular (Kumar et al., 2016; Joshi et al., 2017; Welbl et al., 2017), particularly work on memory networks (Weston et al., 2015; Sukhbaatar et al., 2015; Kumar et al., 2016). Memory networks define a generic model class that attends to a text passage using the question and a memory cell iteratively to gather information in the different parts of the passage. Many existing reading comprehension models use memory net-like structures and iterative attention over the document, showing improvement in a variety of tasks and settings (Hermann et al., 2015; Peng et al., 2015; Sordoni et al., 2016; Dhingra et al., 2016; Shen et al., 2017; Raison et al., 2018).

We tackle two main questions in this paper. First, are memory networks effective? Second, do they behave as advertised (selecting a sequence of relevant passage excerpts through their computation)? We examine the behavior of memory network-like models across three different tasks. These include one purely synthetic setting, the bAbI path-finding task (Weston et al., 2015), and two forms of a more realistic multi-hop reasoning dataset constructed from Wikipedia (Welbl et al., 2017). In each case, we apply memory networks to the problem, and can observe their performance and behavior. Exploiting the properties of these particular datasets, we can use heuristics capturing how humans might solve this problem to derive a pseudogold "reasoning chain." We then compare the model's reasoning chain with this pseudogold to see whether the model is following a similar chain of reasoning.

Our results show that memory networks generally do not learn to do reasoning in the right way, but can do well when using additional supervision to guide how they reason. On bAbI and in a

[1] Code and auxiliary supervision will be available on release.

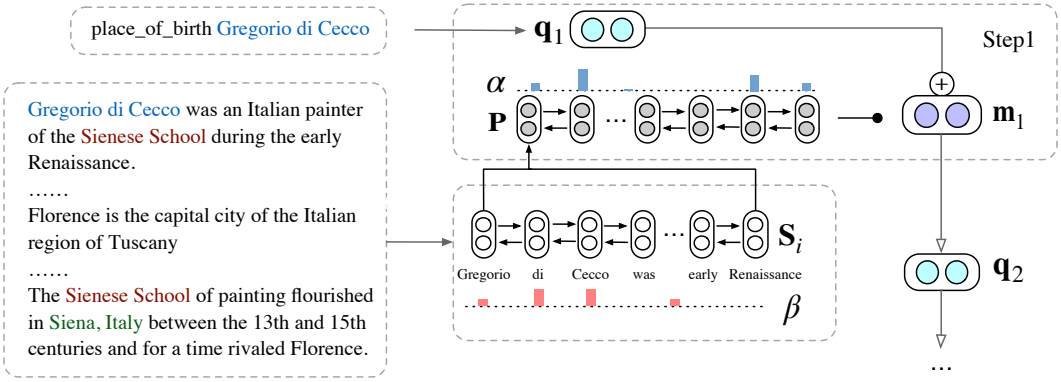

Figure 1: Computation flow of our hierarchical memory network on an example from WikiHop (Welbl et al., 2017). The question is encoded to produce a query $\boldsymbol{q}_1$, which produces sentence-level attention $\boldsymbol{\alpha}$ and word-level attention $\boldsymbol{\beta}$ for each sentence. This attention computes a passage representation $\boldsymbol{m}_1$ from which we form the query $\boldsymbol{q}_2$ for the next step of inference.

"masked" form of the WikiHop task (where entities are anonymized), the memory network performs badly when applied in the standard way. However, when we explicitly supervise the model with pseudogold chains, the model can perform dramatically better with no other changes to its structure or parameterization. On the standard WikHop dataset, our memory network model can achieve nearly state-of-the-art performance, but we show that this is not due to multi-hop reasoning: it barely outperforms a baseline that does not make use of the text at all, calling into question what is being learned. However, additional supervision on the attention can still yield improvement, making our final system close in performance to much more sophisticated state-of-the-art models.

Our observations can be summarized as follows:

- In both synthetic and more realistic settings, memory networks fail to learn multi-hop reasoning from task supervision alone. This is true even though there exist settings of the parameters that *do* fit the data, as we can see by their success when more heavily supervised.
- When the attention of memory networks is additionally supervised during training, they can do well at text question answering. This supervision qualitatively changes the model's performance with respect to multi-hop reasoning.
- When memory networks and related models perform well on multi-hop reasoning tasks, they may be doing so through other means and not actually performing multi-hop reasoning, as we see on the standard WikiHop setting.

2 Model

In this paper, we use a hierarchical version of the original memory network, shown in Figure 1. Given a passage P containing n sentences and a query Q, we first map all the words in P and Q to d-dimensional vectors by an embedding matrix $\boldsymbol{E} \in \mathbb{R}^{|V| \times D}$, where $|V|$ is the size of vocabulary. Next, we use word-level bidirectional GRUs (Chung et al., 2014) with hidden size h to compute contextual representations of each token in the query and each token in the sentences $S_i = w_1, w_2, ..., w_m$ in P. We then have $\boldsymbol{S_i} = \boldsymbol{h}_1^w, \boldsymbol{h}_2^w, ..., \boldsymbol{h}_m^w$, $\boldsymbol{Q} = \boldsymbol{h}_1^q, \boldsymbol{h}_2^q, ..., \boldsymbol{h}_k^q$. We then use another bidirectional GRU with the same hidden size h on the sentence level to form sentence-level representations for the passage: $\boldsymbol{P} = \boldsymbol{h}_1^S, \boldsymbol{h}_2^S, ..., \boldsymbol{h}_n^S$.

For a single layer reasoning model, we take the encoded query $\boldsymbol{q}_1 = \boldsymbol{h}_k^q$ and use it to compute sentence-level attention α as well as word-level attention β_i for each sentence i, which then are used to compute our summarized passage representation (memory):

$$\boldsymbol{\alpha} = \text{softmax}(\boldsymbol{q}_1 \boldsymbol{W}_\alpha \boldsymbol{P}), \quad \boldsymbol{\beta_i} = \text{softmax}(\boldsymbol{q}_1 \boldsymbol{W}_\beta \boldsymbol{S_i}), \quad \boldsymbol{m} = \sum_i \alpha_i \left(\sum_j \beta_j^i \boldsymbol{h}_j^w \right) \tag{1}$$

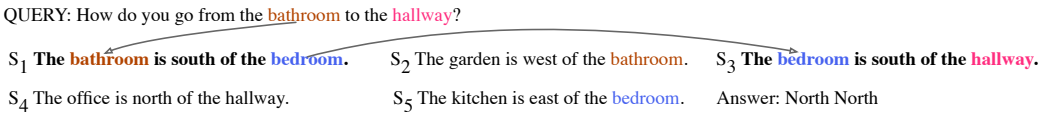

Figure 2: An example from the Path Finding task (Weston et al., 2015). The bold sentences denote the gold reasoning chain. The arrows illustrates the reasoning process.

The product of $\alpha_i \beta_j$ can be viewed as a hierarchical way of achieving a word-level attention which normalizes over the passage. When using multiple steps of reasoning, shared GRU encoders are used but different attention weight matrices $\boldsymbol{W}_\alpha$ and $\boldsymbol{W}_\beta$ are used for each step. At timestep $t > 1$, the query q_t is a combination of the memory $\boldsymbol{m}_{(t-1)}$ of last time step and the original encoded query $\boldsymbol{q}_1$, computed as:

$$\boldsymbol{q}_t = \text{ReLU}(\boldsymbol{W}_t(\boldsymbol{q}_1 + \boldsymbol{m}_{(t-1)}), \tag{2}$$

Then, based on Equation 1, we can compute the sentence-level attention $\boldsymbol{\alpha}$, word-level attention $\boldsymbol{\beta}$, and memory for each step.

We can either predict the answer from the last memory cell $m_{t_{\text{final}}}$ or use a combination of multiple memory cell values; this is a hyperparameter we vary for different datasets. We denote the model proposed here as MemNet in the following sections.

3 Attention Supervision

With the attention weights $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, our model exposes an explicit representation of what part of the passage it consults at each timestep. Past work typically treats this as a purely latent variable to be learned end-to-end. However, it can also be treated as a variable we can explicitly supervise (Liu et al., 2016; Mi et al., 2016; Das et al., 2017). This allows us to "teach" the model the right mode of reasoning at training time so it can generalize correctly at test time.

Suppose we want to supervise the reasoning chain at training time. We assume access to a series of subjectively "correct" sentence-level attention targets $\boldsymbol{\alpha}^* = (\boldsymbol{\alpha}_1^*, \ldots, \boldsymbol{\alpha}_n^*)$ for the n steps of inference. We can encourage the model to attend to these targets at each step by supervising the attention weight; this supervision can be accomplished by incorporating a loss term $L = -\sum_t \log(\boldsymbol{\alpha}_t^*)$

We train with this extra supervision for the first several epochs during training, remove it after the model performance converges on the development set, then keep training only with the downstream supervision. The model trained with extra supervision is denoted as "MemNet+Sup".

4 Synthetic Task: Path-finding

To see how well our model learns multi-hop reasoning in a simple setting, we first conduct experiments on the synthetic bAbI dataset (Sukhbaatar et al., 2015). The passages and questions in this dataset are generated using templates, removing many complexities inherent in natural language. The original memory network achieves fairly strong performance on most of the sub-tasks of bAbI, but does poorly on the task of Path Finding (task 19). Figure 2 shows an example from this dataset. Because examples are constructed synthetically with a completely random distribution, the model

	1k examples	10k examples
MemNet from Sukhbaatar et al. (2015)	10.1%	19.2%
MemNet	17.4%	36.1%
MemNet+Sup	90.0%	100.0%

Table 1: The accuracy of MemNet on the task of Path Finding with 1k and 10k examples. Supervising the attention layers helps substantially. We also report the performance of the 2-hop memory network in Sukhbaatar et al. (2015) for reference.

	Step 1			Step 2		
	> 0.5	> 0.8	AvgMax	> 0.5	> 0.8	AvgMax
MemNet	11.1	7.8	0.88	42.8	30.0	0.76
MemNet+Sup	89.7	87.7	0.98	98.5	97.7	0.99

Table 2: Sentence-level attention weights placed on gold chains on both first and second step on the bAbI development set. Models are trained on 1k examples. Here, "$> a$ " denotes the percentage of samples which place more than an a-fraction of the weight on the gold sentence at that step. AvgMax denotes the average maximum attention value over the whole development set. Statistics on models trained on 10k examples look similar.

must consult multiple parts of the text in order to identify the correct path: there are no spurious correlations or surface regularities it can use to achieve high performance.

Gold reasoning chains (the relevant sentences describing the path) are provided naturally with the dataset, which provide a point of comparison for our model's behavior. We focus on the Path Finding task to test the limitations of our proposed model and see whether attention supervision helps.

Implementation Details Since the Path Finding task only requires a two-step reasoning, we fix the number of MemNet hops to 2. The word embedding dimension is set to 100 and the GRU hidden size is 128. We apply a dropout of rate 0.5 to both embedding layer and GRU output layer. Batch size is set to 32 and we use Adam (Kingma & Ba, 2014) as the optimization method with an initial learning rate 0.0001.

4.1 RESULTS

The results on Path Finding are shown in Table 1. MemNet trained on 1k examples performs poorly, similar to results from the original memory network in previous work. However, once the gold chains are provided, the performance improves significantly. Even in the larger 10k examples setting, the memory network *still* fails to generalize, while extra supervision enables it to achieve perfect accuracy.

We can probe our model further and compare its learned sentence-level attention values with the gold reasoning chain to see whether the model's attention aligns with a human's. The statistics are shown in Table 2. The average max weight tells us that the attention distribution is somewhat peaked even for MemNet – but while the model always attends to something, it does not attend to the gold chain. The MemNet model can fit the training set, but it does not generalize when only using the downstream supervision. While the model may work better with more training data, requiring more than 10k examples for a synthetic task suggests that the model will not scale to real natural language data. By adding the extra supervision, we can see a big jump both in the performance and the number of sentences with the correct attention. The attention supervision helps the model figure out the right pattern of reasoning, which it can apply at test time.

Overall, we see that even in a simple synthetic setting, the MemNet fails to learn multi-hop reasoning. Critically, we see that this does not appear to be a failing of the model: it is *within our model's capacity* to learn to do the right thing, but not without additional supervision. This calls into question whether memory networks are doing the right thing in more complex, real-world problems, and whether we might be able to improve them in those settings as well, as we explore in the next section.

5 WIKIHOP

WikiHop (Welbl et al., 2017) is a recently released reading comprehension dataset specially designed for multi-hop reasoning over multiple documents. Disjoint pieces of text over multiple documents are required to answer the question. Each instance of the dataset contains several documents $d_1, d_2, ..., d_n$. Questions are posed as a query of a relation r followed by a head entity h, with the task being to find the tail entity t from a set of entity candidates E.

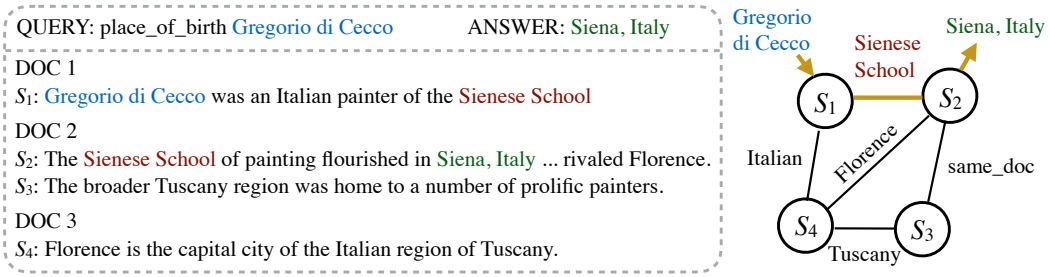

Figure 3: An example of graph search. A graph is formed based on entity relationships between the document sentences. The bold path is the path from query to answer obtained by breadth-first search.

5.1 Pseudogold Chain Extraction

Unlike bAbI, a gold reasoning chain is not provided in WikiHop. However, we can derive a pseudo-gold chain based on heuristic reasoning. We first concatenate all the documents, and run the Stanford Named Entity Recognition system (Finkel et al., 2005) to identify all entities in the documents. We then construct a graph over the sentences of the documents as shown in Figure 3. Each sentence s_i is represented as a node i in the graph. If sentence i and sentence j contain the same entity, we add an edge between node i and j. Also, since sentences in the same document d are likely to be linked through coreference or bridging anaphora, or at least relevant to the same entity, we add an edge between all pairs of sentence within the same document.

We then do a search over the constructed graph to get a pseudogold reasoning chain. Specifically, we first find the sentences containing the head entity in the query, then do a breadth-first search to find the answer. This process returns some shortest path, which we take as the reasoning chain. More sophisticated strategies are possible, including treating the choice of correct path as a latent variable, but we found that this simple approach returns *some* sensible path in most cases, likely due to how it mirrors the process used to construct the dataset.

By conducting the graph search, we found that most of pseudogold chain needs two (47%) or three (25%) steps' reasoning, indicating that multi-hop reasoning is truly necessary. Only 2% and 10% examples need one and more than three steps' reasoning respectively. We are not able to find the pseudogold chain for the remaining examples because the NER system fails to recognize some entities in the document.

5.2 Implementation Details

We fix the number of reasoning steps to 3. When supervising examples with pseudogold chains of length less than 3, we duplicate the final attention target for the remaining steps. When no pseudogold chain can be found, we do not supervised that example. When the pseudogold chain is longer than 3 entries, we set the first and last steps of the supervision and add all other sentences as possible intermediate targets for the second step. We combine the memory cell computed in the second and third step and do a dot product with each option to get the probabilities over options.

We only keep the most frequent 50k words as our vocabulary, and map all the other tokens to *unk*. The word embedding is initialized with 100-dimensional pre-trained Glove embedding (Pennington et al., 2014). We use a hidden size of 128 for GRU, and apply a dropout of rate 0.5 to both the embedding layer and the GRU output layer. All the attention weights used are initialized using Glorot initialization (Glorot & Bengio, 2010). The batch size is set to 24, and we use Adam (Kingma & Ba, 2014) as the optimization method with a initial learning rate 0.0001.

5.3 Results

Our model's performance on WikiHop is shown in Table 3. We compare against several state-of-the-art systems, including the coreference-aware system of Dhingra et al. (2018), the deep co-encoding model of Raison et al. (2018), and two models that use entity graph structure at test time (Song et al., 2018; Raison et al., 2018). Our basic MemNet already achieves strong performance on this dataset, but supervising the attention (MemNet+Sup) improves performance by around 1%, outper-

	Standard		Masked	
Method	Dev	Test	Dev	Test
Majority-cand (Welbl et al., 2017)	–	38.8	–	12.0
BiDAF (Welbl et al., 2017)	–	42.9	–	54.5
Coref-GRU (Dhingra et al., 2018)	56.0	59.3	–	–
Facebook Jenga (Raison et al., 2018)	–	65.3	–	–
MHQA-GRN (Song et al., 2018)	62.8	65.4	–	–
Entity-GCN (De Cao et al., 2018)	**64.8**	**67.6**	70.5	–
NoText	59.7	–	–	–
MemNet	61.8	–	14.2	–
MemNet+Sup	**62.7**	**66.9**	48.7	–

Table 3: The performance of different models on development and test set. Our simple memory-network based model outperforms recent prior work and nearly equals the performance of the Entity-GCN (De Cao et al., 2018).

forming all prior systems except that of Raison et al. (2018). In particular, our model outperforms several other models with more sophisticated test-time preprocessing such as coreference resolution (Dhingra et al., 2018), apparently indicating that the memory network can learn to compensate for this.

There is another explanation for the strong performance on this task, which is that the model actually does something much more straightforward than it appears. We implement one additional "no text" baseline: we encode the query and the options using a bi-GRU and do a dot product between them to get the probabilities over options, making no reference to the document text. This baseline to our knowledge is unexplored in prior work, yet achieves 59.7% performance on development set.

We conclude that this task is actually possible to solve reasonably well *without using the document at all*. Our MemNet model therefore may not really be relying on multi-hop reasoning to attain its high performance. We correct this problem with the dataset using the masked version, as we discuss in the next section.

5.4 Masked WikiHop

From the high performance of the NoText model, we see that the model can pick up on correlations between questions and options in a way that does not require multi-hop reasoning over the text. We can use an alternative form of the dataset described in (Welbl et al., 2017) that removes the model's ability to capture these correlations. The dataset is *masked* as follows: each answer is replaced with a special indexed mask token $mask_1, \ldots, mask_n$, and its occurrences in the text are replaced as well. Now, the model must use multiple hops to find the answer: the NoText baseline cannot do better than random chance (around 11%).

Table 3 shows the performance of the masked system. In this setting, the basic memory network fails to learn generalizable reasoning, just as in the case of bAbI. We will quantify the behavior of its attention in the next section. With supervision, however, our model can achieve 48.7% accuracy, which at least reflects some ability to answer questions better than the baseline methods. Capturing the correct reasoning is therefore in model capacity, but supervision is necessary to learn it in the context of this model.

6 Attention Behavior

We have observed in the previous section that additional supervision is critical for the model to learn in the masked setting and can still lead to performance improvements in the unmasked setting, despite that setting being somewhat easy. To understanding the attention mechanism's behavior, we conduct two additional experiments. First, as we do for bAbI in section 4.1, we compute the fraction of attention on the pseudogold sentences. Second, we compute the model's accuracy with examples stratified by the attention weight placed on the third (final) step of the pseudogold chain, which always contains the answer. The results are shown in Table 4 and Table 5. From the tables we have some general observations: (1) Adding attention supervision causes the model's attention

1st step gold weight					
	≤0.1	>0.1	>0.5	>0.8	AvgMax
MemNet	87.4	12.8	3.9	2.9	0.41
MemNet+Sup	16.3	83.7	79.8	75.1	0.93
MemNet Masked	95.2	4.8	3.4	3.1	0.42
MemNet Masked+Sup	12.7	87.3	77.9	70.0	0.85
3rd step gold weight					
	≤0.1	>0.1	>0.5	>0.8	AvgMax
MemNet	77.3	22.7	5.4	5.4	0.18
MemNet+Sup	47.6	52.4	20.0	10.4	0.41
MemNet Masked	65.5	34.5	6.1	6.1	0.16
MemNet Masked+Sup	61.1	38.9	21.1	13.3	0.56

Table 4: Percentages of samples with attention above or below the given threshold. AvgMax denotes the average of the max weight over sentence of the whole development set.

Model accuracy regarding 3rd step gold weight				
	≤0.1	>0.1	>0.5	>0.8
MemNet	58.0	74.3	75.8	75.8
MemNet+Sup	52.2	71.2	80.9	81.4
MemNet Masked	12.5	23.1	31.1	31.1
MemNet Masked + Sup	35.3	69.6	83.3	82.1

Table 5: Accuracy of models with attention weight above or below the given threshold. Here we only pick the the attention weight on the third step as an illustration since it has similar behaviors on all three steps.

distribution to become dramatically more peaked, and also much more concentrated on the correct chain. (2) Attending to the right place with higher weights yields consistently better performance. Beyond this, we observe a few key phenomena.

MemNet does not match the pseudogold reasoning From the average max values in table 4, we see that the attention distribution of MemNet is much flatter than it is in bAbI, and most of the attention weight on the pseudo gold path is less than 0.1 for all of the three steps. However, MemNet can still achieve an accuracy of 58.0% even these examples with less than 0.1 attention mass on the third step of the pseudogold chain. The fact that performance is independent of attention identifying the right sentence indicates that rather than learning the right reasoning pattern, MemNet is likely learning lexical overlap between the query, option, and the document.

Correct reasoning is hard to learn Comparing MemNet+Sup and MemNet Masked+Sup, we observe a correlation between attention concentrated on the pseudogold and strong model performance. In fact, the masked model can achieve performance comparable to the unmasked model on examples for which its attention is very confident about the correct answer. The difference is primarily the fact that the model's attention is not "correct" as frequently in the masked setting. However, when the model is not confident, MemNet+Sup performs much better, indicating that these samples are being answered by some kind of lexical matching.

7 Error Analysis

Our results have generally established that it is difficult for MemNet to learn the correct reasoning directly from the data with no additional supervision. However, even our best model, MemNet+Sup, still makes a substantial number of errors. We randomly pick 50 examples with wrong predictions and roughly identify a few major categories of errors. The first category is **attention split (40%)** (top of Table 6). If the current sentence contains too many entities or some common entities like

Category	Attended Sentences
Attention Split	Query: award_received_by WGBH (ID: dev_660) Step 1: The WGBH Educational Foundation, ...is a **nonprofit organization** (0.11) WGBH is a public radio station located in Boston ... (0.53) Step 2,3: The WGBH Educational Foundation, ...is a **nonprofit organization** (0.29) It won a **Peabody Award** in 2007 ... (0.12) Predict: **Nonprofit organization** Answer: **Peabody Award**
Wrong Tail Entity	Query: located_in_the_administrative_territorial USS Bowfin (ID: dev_1049) Step 1: USS Bowfin (SS / AGSS - 287) , a Balao - class submarine ... (0.99) Step 2: She has been open to public tours ,..., and Park in Pearl Harbor. (0.64) Step 3: Pearl Harbor is a harbor on the island of Oahu, **Hawaii**, west of **Honolulu**. (0.26) Predict: **Hawaii** Answer: **Honolulu**

Table 6: Some representative examples from two error categories. Here, we picked up the sentence with the highest attention weight at each step. ID denotes the actual example ID in the development set of WikiHop. The number at the end of each sentence denotes the aggregate attention weight for the sentence.

country names, the model tends to split its attention weight over all "successor" sentences containing such entities. Another major source of errors is **wrong tail entity (30%)** (second in Table 6). In this case, the model's attention follows the pseudogold chain, but the final sentence contains several entities, e.g., a country and a city, and the model fails to choose the right one which fits the query. In both of these cases, attention is behaving as expected, but the model is simply not powerful enough to do the correct reasoning. There may be difficult ambiguity in the natural language or, particularly to handle attention split, the model may need a greater ability to plan and do global reasoning. The remaining errors are largely due to **wrong attention (12%)**, where the model simply attends to the wrong sentences for unknown reasons, or **other (18%)**, which include cases where unknown words have confused the model or the selected answer seems unrelated to the attention.

8 Related Work

Memory networks (Weston et al., 2015; Sukhbaatar et al., 2015; Kumar et al., 2016) define a generic model class to deal with multi-hop reasoning over sentences, paragraphs and passages. Such models use the question and the memory cell iteratively to gather information from different parts of the passage to answer the query. A large number of models (Peng et al., 2015; Hermann et al., 2015; Hill et al., 2015; Sordoni et al., 2016; Dhingra et al., 2016; Shen et al., 2017; Hu et al., 2017), regardless of whether they appear to need multi-hop reasoning or not, have incorporated the memory and multi-hop computation into their works, showing improvement in a variety of tasks and settings.

Attention mechanisms have been widely used across many NLP tasks such as machine translation (Bahdanau et al., 2015), question answering (Seo et al., 2016), and summarization (See et al., 2017). Past work typically treats this as a purely latent variable to be learned end-to-end. However, a line of work in machine translation finds gains from supervising attention (Liu et al., 2016; Mi et al., 2016). In visual question answering, past work has also focused on understanding and exploiting how the attention aligns with what humans do (Das et al., 2017; Qiao et al., 2017).

Beyond bAbI and WikiHop, other reading comprehension datasets like McTest (Richardson et al., 2013), CNN/Daily Mail (Hermann et al., 2015), SQuAD (Rajpurkar et al., 2016), RACE (Lai et al., 2017), TriviaQA (Joshi et al., 2017) contain questions related to multi-hop reasoning, but do not focus on it explicitly.

9 Conclusion

In this paper, we explore how the memory network behaves on the task of multi-hop reasoning. Experimental results on bAbI and WikiHop show that additional supervision beyond the downstream answers to the questions is needed to learn generalizable multi-hop reasoning. However, when incorporating this supervision, our memory network model can learn to do this and achieves strong results on the WikiHop dataset.

REFERENCES

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *ICLR*, 2015.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Abhishek Das, Harsh Agrawal, Larry Zitnick, Devi Parikh, and Dhruv Batra. Human attention in visual question answering: Do humans and deep networks look at the same regions? *Computer Vision and Image Understanding*, 163:90–100, 2017.

Nicola De Cao, Wilker Aziz, and Ivan Titov. Question answering by reasoning across documents with graph convolutional networks. *EMNLP*, 2018.

Bhuwan Dhingra, Hanxiao Liu, Zhilin Yang, William W Cohen, and Ruslan Salakhutdinov. Gated-attention readers for text comprehension. *ACL*, 2016.

Bhuwan Dhingra, Qiao Jin, Zhilin Yang, William W Cohen, and Ruslan Salakhutdinov. Neural models for reasoning over multiple mentions using coreference. *NAACL*, 2018.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pp. 363–370. Association for Computational Linguistics, 2005.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256, 2010.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pp. 1693–1701, 2015.

Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. The Goldilocks Principle: Reading Children's Books with Explicit Memory Representations. *arXiv preprint arXiv:1511.02301*, 2015.

Minghao Hu, Yuxing Peng, and Xipeng Qiu. Reinforced mnemonic reader for machine comprehension. *CoRR, abs/1705.02798*, 2017.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. *ACL*, 2017.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. Ask me anything: Dynamic memory networks for natural language processing. In *International Conference on Machine Learning*, pp. 1378–1387, 2016.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. RACE: Large-scale ReAding Comprehension Dataset From Examinations. *arXiv preprint arXiv:1704.04683*, 2017.

Lemao Liu, Masao Utiyama, Andrew Finch, and Eiichiro Sumita. Neural machine translation with supervised attention. *COLING*, 2016.

Haitao Mi, Zhiguo Wang, and Abe Ittycheriah. Supervised attentions for neural machine translation. *EMNLP*, 2016.

Baolin Peng, Zhengdong Lu, Hang Li, and Kam-Fai Wong. Towards neural network-based reasoning. *arXiv preprint arXiv:1508.05508*, 2015.

Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

Tingting Qiao, Jianfeng Dong, and Duanqing Xu. Exploring human-like attention supervision in visual question answering. *arXiv preprint arXiv:1709.06308*, 2017.

Martin Raison, Pierre-Emmanuel Mazaré, Rajarshi Das, and Antoine Bordes. Weaver: Deep Co-Encoding of Questions and Documents for Machine Reading. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2018.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

Matthew Richardson, Christopher JC Burges, and Erin Renshaw. MCTest: A Challenge Dataset for the Open-Domain Machine Comprehension of Text. In *EMNLP*, volume 3, pp. 4, 2013.

Abigail See, Peter J Liu, and Christopher D Manning. Get to the point: Summarization with pointer-generator networks. *ACL*, 2017.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*, 2016.

Yelong Shen, Po-Sen Huang, Jianfeng Gao, and Weizhu Chen. Reasonet: Learning to stop reading in machine comprehension. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1047–1055. ACM, 2017.

Linfeng Song, Zhiguo Wang, Mo Yu, Yue Zhang, Radu Florian, and Daniel Gildea. Exploring graph-structured passage representation for multi-hop reading comprehension with graph neural networks. *arXiv preprint arXiv:1809.02040*, 2018.

Alessandro Sordoni, Philip Bachman, Adam Trischler, and Yoshua Bengio. Iterative alternating neural attention for machine reading. *arXiv preprint arXiv:1606.02245*, 2016.

Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. End-to-end memory networks. In *Advances in neural information processing systems*, pp. 2440–2448, 2015.

Dirk Weissenborn, Georg Wiese, and Laura Seiffe. Making neural qa as simple as possible but not simpler. *CoNLL*, 2017.

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. Constructing datasets for multi-hop reading comprehension across documents. *arXiv preprint arXiv:1710.06481*, 2017.

Jason Weston, Antoine Bordes, Sumit Chopra, Alexander M Rush, Bart van Merriënboer, Armand Joulin, and Tomas Mikolov. Towards AI-Complete Question Answering: A set of prerequisite toy tasks. *arXiv preprint arXiv:1502.05698*, 2015.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V Le. QANet: Combining Local Convolution with Global Self-Attention for Reading Comprehension. *arXiv preprint arXiv:1804.09541*, 2018.