Every Answer begins with a Question - Ask me Another!
Dynamic Memory Network for Question Answering

Anup Jha
anup.jha@berkeley.edu

Chitra S. Agastya
chitra.agastya@berkeley.edu

Abstract

Question Answering (QA) is a difficult problem to solve. Neural network models with memory and attention have made some strides in exhibiting reasoning capabilities required for QA. One such architecture, the Dynamic Memory Network (DMN)[1] achieved high levels of accuracy on several language tasks. In this paper, we assess the extent to which the DMN can answer questions for a semi-closed[2] QA domain i.e. predict if it is possible to answer a question in a given context after the machine has analyzed the question-context pair; and where an answer is possible, generate the answer to the question.

1 Introduction

In the NLP world any task can be thought of as a question answer pair (Kumar et al., 2015[1]). For example, if we want a machine model to summarize a document, it can be thought of as the question ”What is the summary of this passage?” or if we want to classify an email as spam, it can be thought of as the question ”Is this email spam?”.

Answering questions from the context of a passage means a machine has to learn the correlation between the question and the passage, pay attention to specific parts of the passage and have transitive reasoning ability. This is challenging, as the model has to relate the facts presented in the passage within the context of the given question and create a meaningful answer. This task is particularly difficult with the SQuAD 2.0 dataset which is a mix of answerable and unanswerable questions.

We compare and evaluate models that use general principles from DMN. Our models have two heads: (1) to predict the feasibility of an answer for the given question and context and is trained on raw inputs from the question-context-unanswerable[3] triplets from the data (2) to generate the answer to the given question, and is trained on the raw inputs from the question-context-answer triplets from the data. We use pretrained GloVe vectors for word embeddings.

2 Data

We use the SQuAD 2.0 dataset (Rajpurkar et al., 2018[4]) for training and evaluation of our model. This is a reading comprehension dataset, and contains questions posed by crowdworkers on a set of Wikipedia articles. Every question in the dataset, if answerable, has a span of text from the context marked as the answer. If such an answer is not possible, then the question is marked unanswerable. In this report, we refer to this as a semi-closed domain, as the dataset[5] combines around 100,000 question-context-answer triplets with around 30,000 unanswerable questions written by humans to look similar to answerable ones.

3 Model Design

There are five modules that make up our DMN: question module, input context module, episodic memory module, answer module and answer feasibility module. Figure 1 shows the common architecture of our DMN depicting all five modules and the interactions between them.

3.1 Modules

The question module encodes the given sequence of word embeddings using a deep bi-directional

[1] Ask me Anything: Dynamic Memory Networks for Natural Language Processing
- https://arxiv.org/pdf/1506.07285.pdf

[2] Not all questions have answers in the given context

[3] SQuAD2.0 has a field is-impossible that indicates if a question is answerable or not by the given context

[4] Know What you Don’t Know: Unanswerable Questions for SQuAD - https://arxiv.org/abs/1806.03822

[5] https://rajpurkar.github.io/SQuAD-explorer/

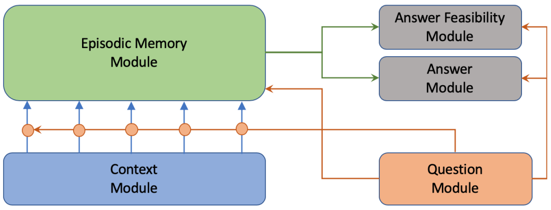


Figure 1: DMN architecture for our models showing interactions indicated by arrows

recurrent neural network (RNN). It emits the final hidden state of the question embeddings as a question vector.

The input context is a paragraph of sentences each made up of a sequence of words. We concatenate input sentences into a list of word tokens. This module encodes the word embeddings of the given paragraph using a deep bidirectional RNN. We emit the hidden state for every word in the input context.

The episodic memory module at time t performs an attention on the context and question vectors, while also taking into account the previous internal episodic memory m_{t-1} and it updates its internal memory state m_t. The initial hidden state of the memory m_0 is assigned to the question vector itself. The attention mechanism[6] is a gating function. It takes as inputs context c, question q and memory m and produces a score $z(c, q, m)$ given by:

$$[c, m, q, c * q, c * m, |c - q|, |c - m|, c^T W q, c^T W m] \quad (1)$$

where * is the hadamard product between the vectors. The gating function is calculated using feed forward network

$$g_t^i = tanh(W^{(1)} z(c, q, m) + b^{(1)}) \quad (2)$$

The episodic memory for iteration i is calculated as

$$e^i = \sum_{t=1}^{T} softmax(g_t^i) c_t \quad (3)$$

where the $softmax(g_t^i)$ is calculated as

$$softmax(g_t^i) = \frac{exp(g_t^i)}{\sum_{j=1}^{T} exp(g_j^i)} \quad (4)$$

The answer module takes as inputs vectors from one or more of the above three modules and predicts the answer to the question-context pair.

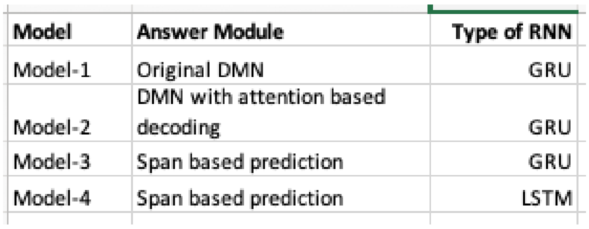
The answer feasibility module takes the question and episodic memory vectors through a dense feed forward neural network followed by an output layer with a sigmoid classifier to predict if the question is answerable or not.

3.2 Embeddings

We use pre-trained GloVe embeddings of dimension 100 for each token in our vocabulary. We also add special tokens $<unk>$ for unknown words, $<s>$ for the start of a sentence and $</s>$ for the end of a sentence. The vector value of $<unk>$ is the average of all tokens from GloVe while the vector values of $<s>$ and $</s>$ are averages of the first and second halves respectively of the word embeddings. The data shows us that 95% of passages, questions and answers are of length less than 250, 20 and 15 tokens respectively. So we take these length values to create the padded sequences.

3.3 Models

We evaluate four different neural models, all implementing DMN in combination with one or more of the language Lego blocks. Our goal is to evaluate which model is most effective in predicting the correct answers for our dataset. Table 1 gives a snapshot of the models. We do not leverage transfer learning and instead build all of our models from scratch.



Model	Answer Module	Type of RNN
Model-1	Original DMN	GRU
Model-2	DMN with attention based decoding	GRU
Model-3	Span based prediction	GRU
Model-4	Span based prediction	LSTM

Table 1: Models Overview

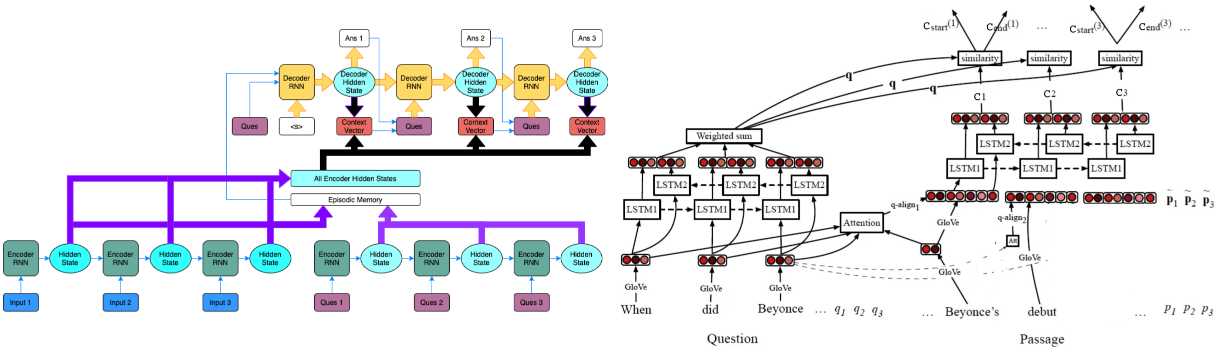
3.3.1 Model-1

Model-1 is our baseline model and is an implementation of DMN as described in the Ask Me anything paper[1]. The answer module takes as input the vectors from the episodic memory and question modules and generates the answer much like the seq2seq model. The answer module's first input token is $<s>$.

3.3.2 Model-2

Model-2 is Model-1 with an attention mechanism in the answer module. We compute an attention vector on the input context. We then concatenate it with the question vector and feed it as an input into the answer module. There are two steps that are performed in the answer module in any iteration

[6] gating and scoring functions are both taken from the original DMN paper



(a) Attention based Architecture

(b) Span based Architecture

Figure 2: Model Architectures

i: (1) compute the attention vector v_i using the previous answer hidden state a_{i-1}; and (2) decode the next answer hidden state a_i using the attention vector v_i and question q.

The decoder uses the episodic memory as its initial state. Figure 2a shows the working mechanism of the answer module. We use the Teacher Forcing method for quickly and efficiently training the RNN in our answer module. During training time, for every prediction of answer token, we feed the correct previous token from the actual answer while at prediction time we feed the previous token from the predicted answer.

3.3.3 Model-3

Since answers to the questions in our dataset are a span of token sequences from the passage, we incorporate the answer spans where available, into our training. Inspired by the neural design provided by Chen et al., 2017[7] we implement a neural network model[8] to predict the begin and end of a span of tokens that is most likely to be the correct answer for our question-context pair.

We generate the context vectors $c_1, c_2 ... c_n$ from a deep layer RNN. Each input token embedding is concatenated with alignment of question embedding and fed as input to the RNN. The alignment of question to input tokens is calculated as $f_{align}(c_i) = \sum_j a_{i,j} E(q_j)$ where the attention score $a_{i,j}$ captures the similarity between input token c_i and question token q_j. Specifically, $a_{i,j}$ is computed by the dot product between nonlinear mappings of word embedding as:

$$a_{i,j} = \frac{exp(\alpha(E(c_i)).\alpha(E(q_j)))}{\sum_k exp(\alpha(E(c_i)).\alpha(E(q_k)))} \tag{5}$$

where $\alpha()$ is a single dense layer with ReLU activation and $E()$ is the embedding vector.

We take the weighted average of question vectors $q_1, q_2 ... q_n$ and each context vector to train two classifiers to predict the probability of each context token being the start and end of an answer span. We compute the probability of any context c_i token being start or end using:

$$C_{start(i)} \propto exp(c_i W_s q) \tag{6}$$

$$C_{end(i)} \propto exp(c_i W_e q) \tag{7}$$

During prediction, we choose the best span from token i to token i' such that $i \leq i' \leq i + 15$ and $C_{start(i)} \times C_{end(i)}$ is maximized. Figure 2b gives an architectural overview of how the span based prediction works.

3.3.4 Model-4

Input contexts have a temporal aspect. A sentence embedding, therefore, has to be analyzed along a time sequence both in forward and reverse directions. We rely on a bidirectional RNN to figure out what information is important enough to carry to the next state. We considered both LSTM and GRU as our potential candidates. Models 1, 2 and 3 use GRU. Model-4 is a span based model just like Model-3 with LSTM in place of GRU.

4 Experiments

Our dataset has over 130,000 questions. We did a 80/20 stratified split of the training dataset and used it for training and validation respectively. The split ensured that there was a good representation of answerable and unanswerable questions in the training and validation datasets. We reserved the SQuAD 2.0 dev dataset with over 11800 questions for evaluation of our best model.

[7] Reading Wikipedia to Answer Open-Domain Questions - https://arxiv.org/pdf/1704.00051.pdf

[8] used in Model-3 and Model-4

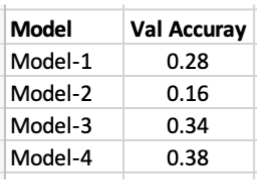
4.1 Hyperparameter Selection

We used a randomized search for optimizing our hyperparameter estimation. Our hyperparameters include: L1 regularization, L2 regularization, depth of bidirectional RNN layers, number of units in RNN layers, number of neurons and layers in feasibility layer, learning rate, number of episodes in the episodic memory and dropout rate.

We ran 8 experiments for each model (listed in table 4, appendix A) with the hyperparameters estimated by our random search and selected results of the best performing model for evaluation and analysis.

4.2 Results

The results of our answer module are shown in table 2. The feasibility module accuracy was around 0.65 and was similar across all models.



Model	Val Accuray
Model-1	0.28
Model-2	0.16
Model-3	0.34
Model-4	0.38

Table 2: Performance of Answer Module

Model-1 gave us 28% accuracy. A closer look at the answer predictions showed that the decoder always predicted the answer as $</s>$ indicating the question was unanswerable (Figure 8, appendix A). A third of our data comprised of unanswerable questions, thus accounting for the accuracy we observed. We concluded that the model was unable to learn and was not worth pursuing. Figure 3 shows the performance graphs for the answer module of Model-1.

Even though Model-2 showed a lower accuracy on the answer module than Model-1, it was able to get much better answer predictions and in many cases, the predicted answers matched the actual answers for the training set. As training progressed, however, this model exhibited signs of over fitting as evidenced by Figure 4 . While Model-2 learnt really well on the training dataset, it was not able to generalize well and performed poorly on the hold out data. Figure 9 in appendix A demonstrates an example of nearly perfect predictions on training data and imperfect prediction on the hold out data.

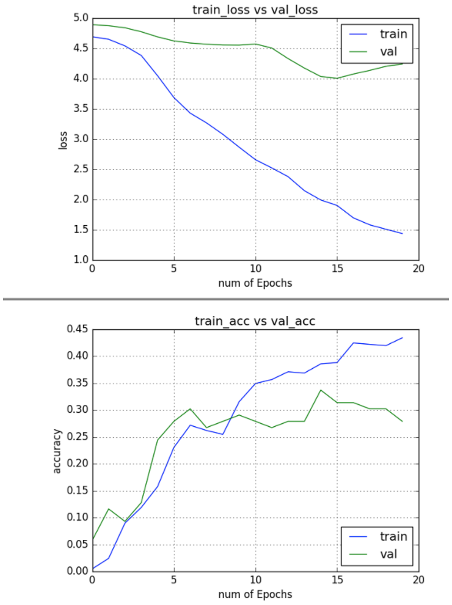


Figure 3: Loss and Accuracy: Model-1 (baseline)

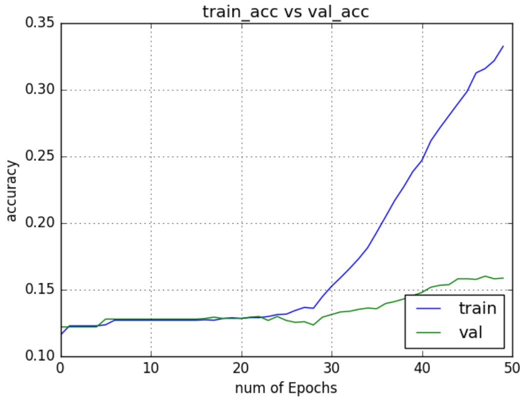


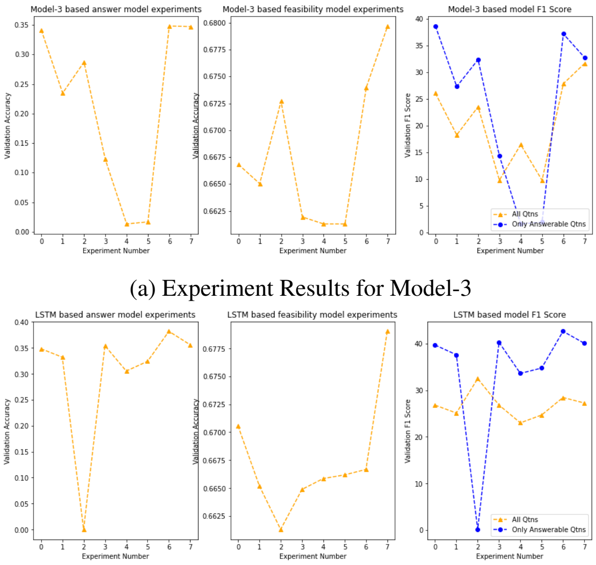
Figure 4: Model-2 accuracy

The span based models Model-3 and Model-4 gave us much better results than Model-1 and Model-2. The accuracy of the answer module for both more than doubled compared to Model-2 with best accuracy of 0.38. Figure 5 shows the results of both models for each experiment. Experiment6 gave us the best performance on both models. Furthermore, our experiments indicate that the LSTM span model i.e. Model-4 is more effective[9] in predicting answers than its GRU counterparts.

4.3 Evaluation

We evaluate our best GRU and LSTM models on two scores: F1 score and exact match (EM) score between the predicted answer and actual answer. The evaluation scores are shown in Table 3. Model-3 gives better scores on predictions for unanswerable questions and Model-4 does better

[9] an improvement of 4 points

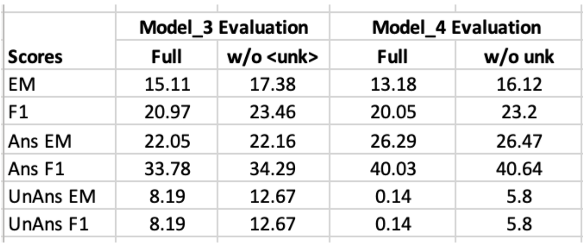


(a) Experiment Results for Model-3

(b) Experiment Results for Model-4

Figure 5: Experiment results for Span based models

in predictions for answerable questions[10]. The difference between them on the overall score is marginal. In our evaluation of the test data, we have not mixed and matched the feasibility modules with the answer modules because the difference in performance of the feasibility modules for the different experiments was marginal and could just be an effect of the initialization parameters chosen for the experiments.



	Model_3 Evaluation		Model_4 Evaluation	
Scores	Full	w/o <unk>	Full	w/o unk
EM	15.11	17.38	13.18	16.12
F1	20.97	23.46	20.05	23.2
Ans EM	22.05	22.16	26.29	26.47
Ans F1	33.78	34.29	40.03	40.64
UnAns EM	8.19	12.67	0.14	5.8
UnAns F1	8.19	12.67	0.14	5.8

Table 3: Evaluation Results

5 Analysis

As evidenced by our experiments, we observe that DMN on its own is not quite effective in predicting answers to questions from the SQuAD 2.0 dataset. While DMN gave high accuracy results with the bAbI project[11], the input contexts in bAbI were much simpler clauses and the answers were just one word from within the context. Moreover, every question/context pair in the dataset had an answer. SQuAD 2.0 on the other hand has answers spanning more than one word, and not every question has an answer. Figure 11 in appendix A gives a sample input from both datasets to demonstrate the difference in complexity.

Additionally, while the episodic memory was effective in capturing the relevance between questions and simple contexts in bAbI to predict answers, it seems to fall short in capturing the relevance between questions and longer, more complex input contexts in SQuAD. We also observe that the episodic memory is not able to do an effective job in determining the feasibility of answering a question. We hypothesize that it is because an attention mechanism can help only in extracting the answer but not predicting whether it is possible to answer a question. To predict whether the question is answerable or not might require other features such as Parts Of Speech tagging (POS), Named Entity Recognition (NER) and other Natural Language Understanding (NLU) features.

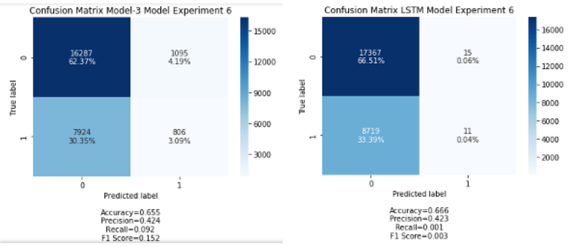


Figure 6: Confusion Matrix: Feasibility Module (Models 3 and 4)

Figure 6 shows the confusion matrices for Model-3 and Model-4. The figure indicates that both models predict almost all questions as answerable[12] even though 30% of the questions are unanswerable. In our evaluation, we first take the result from the feasibility module and if the question is answerable, then generate the answer using the answer module. The feasibility module is ineffective in identifying the unanswerable questions, impacting the overall F1 and EM scores. A more accurate classifier for identifying the feasibility of answering a question could potentially result in higher evaluation scores.

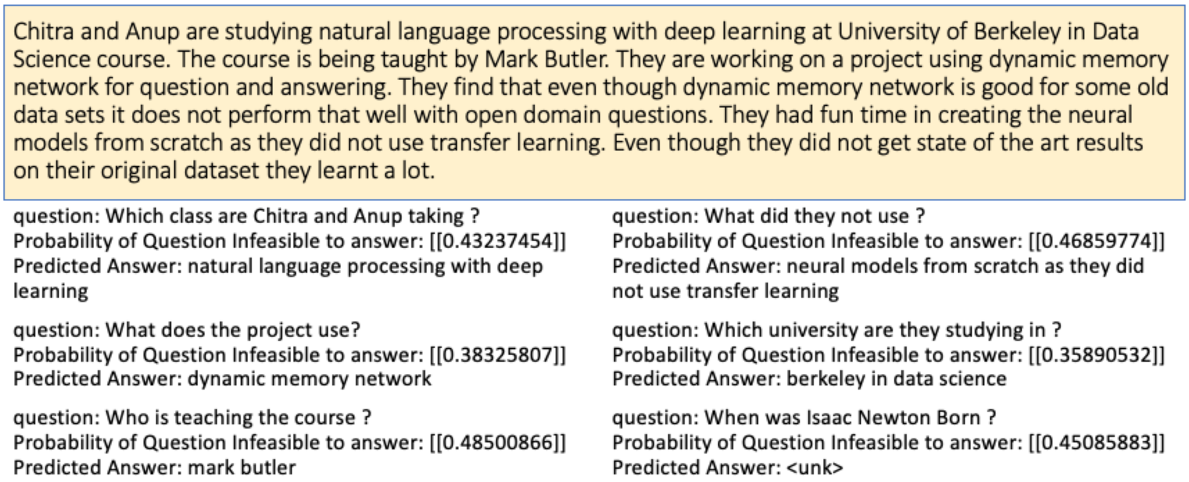
Adding attention mechanisms to our models significantly helped improve the answer prediction in the answer module. We also observe that a deeper RNN in the model design resulted in better performance[13] than a wider RNN. The LSTM based span model was better at predicting answers

[10] When the span model is not able to answer a question from the context it generates $< unk >$ as the answer

[11] https://research.fb.com/downloads/babi/

[12] true label 0

[13] Experiment6 had the greatest depth and smallest width RNN of all our experiments



Chitra and Anup are studying natural language processing with deep learning at University of Berkeley in Data Science course. The course is being taught by Mark Butler. They are working on a project using dynamic memory network for question and answering. They find that even though dynamic memory network is good for some old data sets it does not perform that well with open domain questions. They had fun time in creating the neural models from scratch as they did not use transfer learning. Even though they did not get state of the art results on their original dataset they learnt a lot.

```
question: Which class are Chitra and Anup taking ?
Probability of Question Infeasible to answer: [[0.43237454]]
Predicted Answer: natural language processing with deep
learning

question: What does the project use?
Probability of Question Infeasible to answer: [[0.38325807]]
Predicted Answer: dynamic memory network

question: Who is teaching the course ?
Probability of Question Infeasible to answer: [[0.48500866]]
Predicted Answer: mark butler

question: What did they not use ?
Probability of Question Infeasible to answer: [[0.46859774]]
Predicted Answer: neural models from scratch as they did
not use transfer learning

question: Which university are they studying in ?
Probability of Question Infeasible to answer: [[0.35890532]]
Predicted Answer: berkeley in data science

question: When was Isaac Newton Born ?
Probability of Question Infeasible to answer: [[0.45085883]]
Predicted Answer: <unk>
```

Figure 7: Predicting answers for a non Wikipedia passage

than the GRU based span model as evidenced by the samples in Figure 10 in appendix A. But it does not matter for the overall score in our results. This is attributed to the poor performance of the feasibility module. With a more accurate feasibility module, it would benefit to have a LSTM based answer predicting module to generate the answers to given question/context pairs.

Our model generalized well to non Wikipedia passages. We hypothesize that this might be because we did not rely on transfer learning using a pre-trained model like BERT which is trained on Wikipedia articles. Figure 7 shows our model's prediction for a question/context pair that we created and the answers are quite accurate.

6 Conclusion

Our choice of training from scratch using principles of neural networks, and not relying on available transfer learning models, was a deliberate attempt to understand and get a feel for the complexity that is involved with the language task of QA. Through our experiments and analysis we have demonstrated that the original DMN that performed well on bAbI tasks, is not effective in answering the SQuAD 2.0 questions.

The complexity of the SQuAD dataset required the following modifications to our version of DMN: (1) emit the hidden state of every word in the input context to generate an answer span as compared to the original DMN which only emitted hidden states of every sentence; (2) employ explicit attention mechanisms in the question, input and/or answer modules[14] while the original DMN relied on the built in attention mechanism of the episodic memory; (3) use bidirectional RNN to determine context/question relevance compared to the original DMN which only uses RNN in the forward direction; and (4) add a fifth module i.e. the answer feasibility module to predict the feasibility of answering a given question from the context.

6.1 Summary

Our extensive modeling efforts on the semi-closed SQuAD 2.0 dataset proved to us that QA is indeed a difficult problem to solve. While the DMN by itself is not effective on the SQuAD dataset, with help from attentions and span based modeling, we have demonstrated that we can improve the machine's comprehension of a question in relevance to an input context. While we did not achieve state of the art results with our models, we are optimistic that an improved feasibility module and additional feature engineering on the input context could potentially give us better results.

6.2 Future Work

Given that we have used GloVe embedding in our modeling, it would help to see if context friendly embeddings like ElMO and ULMFiT can potentially result in better performance. We also believe enhancing the input context vectors with additional feature engineering like NER, POS tagging etc. could potentially lead to improved performance.

[14] Models 2,3 and 4

7 References

1. Ask Me Anything: Dynamic Memory Networks for Natural Language Processing, https://arxiv.org/pdf/1506.07285.pdf .

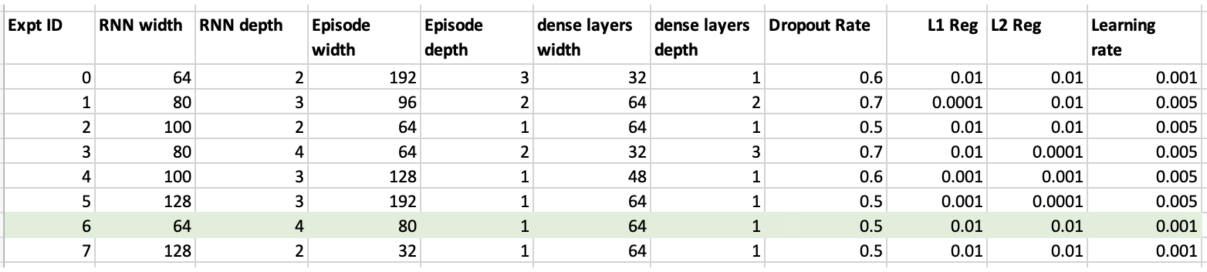
2. A Multi-Stage Memory Augmented Neural Network for Machine Reading Comprehension, https://www.aclweb.org/anthology/W18-2603.pdf.

3. Exploiting Contextual Information via Dynamic Memory Network for Event Detection, https://www.aclweb.org/anthology/D18-1127.pdf.

4. Episodic Memory Reader: Learning What to Remember for Question Answering from Streaming Data, https://www.aclweb.org/anthology/P19-1434.pdf.

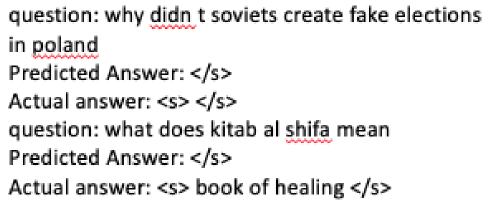
5. Reading Wikipedia to Answer Open Domain Questions - https://arxiv.org/pdf/1704.00051.pdf.

A Appendix: Additional Figures



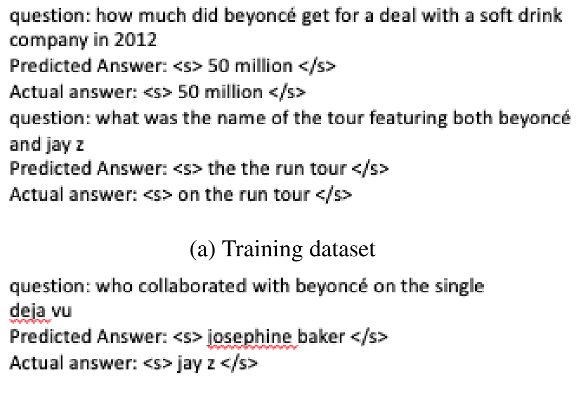
Expt ID	RNN width	RNN depth	Episode width	Episode depth	dense layers width	dense layers depth	Dropout Rate	L1 Reg	L2 Reg	Learning rate
0	64	2	192	3	32	1	0.6	0.01	0.01	0.001
1	80	3	96	2	64	2	0.7	0.0001	0.01	0.005
2	100	2	64	1	64	1	0.5	0.01	0.01	0.005
3	80	4	64	2	32	3	0.7	0.01	0.0001	0.005
4	100	3	128	1	48	1	0.6	0.001	0.001	0.005
5	128	3	192	1	64	1	0.5	0.001	0.0001	0.005
6	64	4	80	1	64	1	0.5	0.01	0.01	0.001
7	128	2	32	1	64	1	0.5	0.01	0.01	0.001

Table 4: Experiment Details



question: why didn t soviets create fake elections in poland
Predicted Answer: </s>
Actual answer: <s> </s>
question: what does kitab al shifa mean
Predicted Answer: </s>
Actual answer: <s> book of healing </s>

Figure 8: Sample Predictions: Model-1 (baseline)



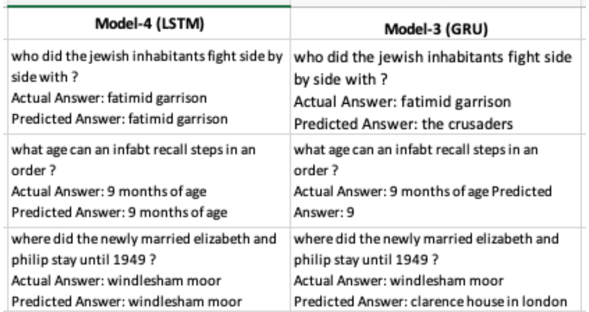
question: how much did beyoncé get for a deal with a soft drink company in 2012
Predicted Answer: <s> 50 million </s>
Actual answer: <s> 50 million </s>
question: what was the name of the tour featuring both beyoncé and jay z
Predicted Answer: <s> the the run tour </s>
Actual answer: <s> on the run tour </s>

(a) Training dataset

question: who collaborated with beyoncé on the single deja vu
Predicted Answer: <s> josephine baker </s>
Actual answer: <s> jay z </s>

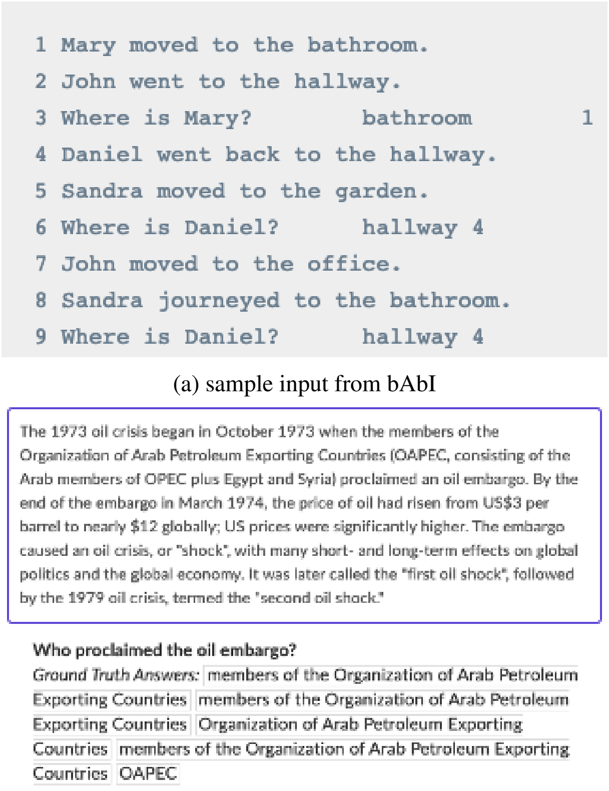
(b) Validation dataset

Figure 9: Sample Predictions: Model-2



Model-4 (LSTM)	Model-3 (GRU)
who did the jewish inhabitants fight side by side with ? Actual Answer: fatimid garrison Predicted Answer: fatimid garrison	who did the jewish inhabitants fight side by side with ? Actual Answer: fatimid garrison Predicted Answer: the crusaders
what age can an infabt recall steps in an order ? Actual Answer: 9 months of age Predicted Answer: 9 months of age	what age can an infabt recall steps in an order ? Actual Answer: 9 months of age Predicted Answer: 9
where did the newly married elizabeth and philip stay until 1949 ? Actual Answer: windlesham moor Predicted Answer: windlesham moor	where did the newly married elizabeth and philip stay until 1949 ? Actual Answer: windlesham moor Predicted Answer: clarence house in london

Figure 10: Comparison of sample predictions from GRU and LSTM based span models



```
1 Mary moved to the bathroom.
2 John went to the hallway.
3 Where is Mary?        bathroom        1
4 Daniel went back to the hallway.
5 Sandra moved to the garden.
6 Where is Daniel?      hallway 4
7 John moved to the office.
8 Sandra journeyed to the bathroom.
9 Where is Daniel?      hallway 4
```

(a) sample input from bAbI

The 1973 oil crisis began in October 1973 when the members of the Organization of Arab Petroleum Exporting Countries (OAPEC, consisting of the Arab members of OPEC plus Egypt and Syria) proclaimed an oil embargo. By the end of the embargo in March 1974, the price of oil had risen from US$3 per barrel to nearly $12 globally; US prices were significantly higher. The embargo caused an oil crisis, or "shock", with many short- and long-term effects on global politics and the global economy. It was later called the "first oil shock", followed by the 1979 oil crisis, termed the "second oil shock."

Who proclaimed the oil embargo?
Ground Truth Answers: members of the Organization of Arab Petroleum Exporting Countries members of the Organization of Arab Petroleum Exporting Countries Organization of Arab Petroleum Exporting Countries members of the Organization of Arab Petroleum Exporting Countries OAPEC

(b) sample input from SQuAD 2.0

Figure 11: Sample Inputs showing complexity of bAbI vs SQuAD

B Appendix: Module Representations

B.1 Question Module

Question is a sequence of words. The question module encodes the given sequence of words using a bi-directional recurrent network. Word embeddings are given as input to the RNN. If the question has T_Q words: $w_1^Q, w_2^Q ... w_T^Q$, at each time step t, the network updates the hidden state of the question q_t as:

$$q_t = RNN(V[w_t^Q], q_{t-1}) \tag{8}$$

where V is the word embedding matrix, w_t^Q is the word index of the t^{th} word in the question.

B.2 Input Context Module

The input context is a paragraph of sentences each made of a sequence of words. We concatenate input sentences into a long list of word tokens. We emit the hidden state for every word in the input context. If the input context has T_C words: $w_1^C, w_2^C ... w_T^C$, at each time step t, the network updates the hidden state of the context c_t as:

$$c_t = RNN(V[w_t^C], c_{t-1}) \tag{9}$$

where V is the word embedding matrix, w_t^C is the word index of the t^{th} word in the input context.

B.3 Answer Feasibility Module

Prediction for the feasibility module can be written as:

$$= \sigma(W^{(2)} ReLU(W^{(1)} f + b^{(1)}) + b^{(2)}) \tag{10}$$

where f is the input to the feasibility module.