**Literature Review**

Deep Learning in Question Answering – Towards Goal-aware Learning

Student ID: 26250365

Name: Chenyang Wang

Course: Bachelor of Information Technology (Honours)

Date: 9/06/2017

Supervisor: Reza Haffari

# Abstract

Question answering in the context of text comprehension has been an active research field in recent years, especially with neural network-based approaches. While there have been several influential researches on external memory-based, iterative neural reasoning models, most of them focus on building efficient static multi-layer networks to improve performance. Our goal is to explore the possibility of building a memory-based, iterative network that can change the number of iterations used according to the question difficulty, i.e. having adaptive flexible depth. We call such a network a “goal-aware” network for its ability to recognise the completion status of a reasoning process. In this review, we explore the background of question answering and analyse the relevant research leading up to this question, as well as outline the potential challenges and contributions of a goal-aware network.

Contents

[Abstract 2](#_Toc484777857)

[1. Introduction 4](#_Toc484777858)

[2. Problem Formulation 4](#_Toc484777859)

[3. Question Answering in Traditional NLP 5](#_Toc484777860)

[4. Datasets and Benchmarks 6](#_Toc484777861)

[CNN / Daily Mail 6](#_Toc484777862)

[bAbI 6](#_Toc484777863)

[TriviaQA 6](#_Toc484777864)

[5. Deep Learning in NLP 7](#_Toc484777865)

[Recurrent Network Architectures 7](#_Toc484777866)

[Word Embeddings 7](#_Toc484777867)

[Phrase and Sentence Representations 7](#_Toc484777868)

[6. Question Answering with Neural Networks 8](#_Toc484777869)

[Comparison of Traditional Approaches with Neural Network Methods 8](#_Toc484777870)

[Memory Network 9](#_Toc484777871)

[Dynamic Memory Network 10](#_Toc484777872)

[Neural Reasoner 10](#_Toc484777873)

[Differentiable Neural Computer 11](#_Toc484777874)

[7. Information Flow Analysis of Above Models 12](#_Toc484777875)

[8. Reinforcement Learning 14](#_Toc484777876)

[Deep Q Network and other Deep Learning Approaches 14](#_Toc484777877)

[Reinforcement Learning in QA Tasks 14](#_Toc484777878)

[9. Goal-Aware Reasoning: How to Get There 15](#_Toc484777879)

[10. Other Potential Future Research Areas 16](#_Toc484777880)

[Choice of Natural Language Representations as Network Input 16](#_Toc484777881)

[Human-Readable Layer Visualisation 16](#_Toc484777882)

[11. Conclusion 16](#_Toc484777883)

[References 17](#_Toc484777884)

# Introduction

Text understanding and question answering have always been among some of the most challenging tasks in natural language processing. With the rise of deep learning and DNN-based learning algorithms, as well as the increasing availability of large training datasets, neural network-based QA has been steadily gaining traction. Many influential models are proposed in recent years, such as Memory Networks (MemNN) (Weston, et al., 2014), Dynamic Memory Networks (DMN) (Kumar, et al., 2016) and Differentiable Neural Computer (DNC) (Graves, et al., 2016). Many of these models adopt a few common basic building blocks: an external memory, an attention mechanism over the inputs and an iteratively-applied reasoning layer for updating memory / extracting answer. In models with such structures, like MemNN and DMN, more reasoning layers (iterations) generally imply better reasoning performance and better resolution of transitive reasoning. However, there is a trade-off between having more layers and encountering overfitting or training convergence problems. It is difficult to find a sweet spot for the number of iterations which work well for both simple questions with short reasoning and more complex questions with longer chains of reasoning. Inspired by prepositional logic, where there is a clearly defined condition for terminating a deduction process when the goal is reached, we intend to explore the possibility of building a flexible-iteration network capable of determining a condition to terminate reasoning and output the answer. We call such a network “goal-aware network”.

In the beginning of this review (section 2-3), we provide a definition of the problem and a brief history of QA in general.

In section 4, we introduce some of the most widely used as well as latest benchmarking datasets for QA algorithms. Analysis of these datasets help us understand the strengths and weaknesses of each dataset, enabling us to put performance claims in later sections in perspective.

Section 5 is a brief overview of the most QA / NLP-relevant neural network architectures and techniques.

Section 6 and 7 are the critical analysis of existing neural QA models. In section 8, an additional model using reinforcement learning is introduced due to its novelty and potential relevance.

Section 9 focuses on the justification and potential research directions for goal-aware reasoning.

Section 10 briefly discusses other potential future research areas.

# Problem Formulation

Natural language question answering has been an active research field since the 1960s (Hirschman & Gaizauskas, 2001). Over the years, the goal and scope of the problem has changed several times depending on the target use case and technical capabilities of the time. Hirschman and Gaizauskas (2001) define a question answering system as one that “allows a user to ask a question in everyday language and receive an answer quickly and succinctly, with sufficient context to validate the answer.”. Andrenucci and Sneiders (2005) define the problem as “the process of retrieving precise answers to natural language (NL) questions”. These definitions fit several different sub-problems in QA research, such as:

1. Natural language QA frontend for databases / knowledge bases, which focuses on the processing of natural language questions and retrieval of answers stored in structured data
2. Information retrieval, which focuses on searching for relevant documents from a large collection of documents (such as the task of a web search engine)
3. Text comprehension, which focuses on answering questions based on facts presented in a natural language form

Etc.

The focus of this review is question answering in the context of text comprehension. We formulate the definition of the problem as follows:

Given one or more **natural language documents** containing a number of **facts** and a **natural language question**, **find** relevant facts in the input documents, perform necessary **reasoning** over the facts, and **present the answer** in either structured or natural language format.

The above definition requires a QA system to be able to take natural language information sources and queries as input and perform at least simple logical inferences to extract the answer desired. Unlike information retrieval tasks, in text comprehension tasks, we usually restrict the allowed information sources to the input documents only.

In the following chapters, we mainly discuss about text comprehension-related research, however certain historically significant researches from other sub-fields of QA are also mentioned.

# Question Answering in Traditional NLP

Some well-known early research on natural language question answering were conducted in the 1960-70s, with limited success on providing a natural language frontend to structured knowledge bases within a narrow domain (Hirschman & Gaizauskas, 2001).

Early work on text comprehension started in the late 1970s, such as Lehnert’s theory of question answering (1977).

Prior to the rapid improvement of neural network-based NLP solutions, most question answering solutions can be categorised into three groups: NLP-based QA, information retrieval QA and template-based QA (Andrenucci & Sneiders, 2005). The comparison of these techniques, along with earlier database NL frontends and deep learning approaches are shown in table 1.

QA systems based on traditional NLP workflow usually rely on an stack of NLP processor modules, each specialising in one NLP task. A typical NL frontend to a knowledge base has a structure shown in figure 1.

Table 1 (based on Andrenucci & Sneiders, 2005)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Input | Knowledge Source | Output | Domain |
| Early database NL frontends | Semi-structured | Highly structured, limited | Accurate | Narrow |
| Traditional NLP QA systems | Natural | Structured, limited | Accurate | Narrow |
| Information Retrieval Techniques | Natural | Unstructured, large, redundant | Low accuracy | Broad |
| Templates | Structured | Structured | Low accuracy | Narrow |
| Deep Learning | Natural | Structured or unstructured | Accurate | Data dependent |

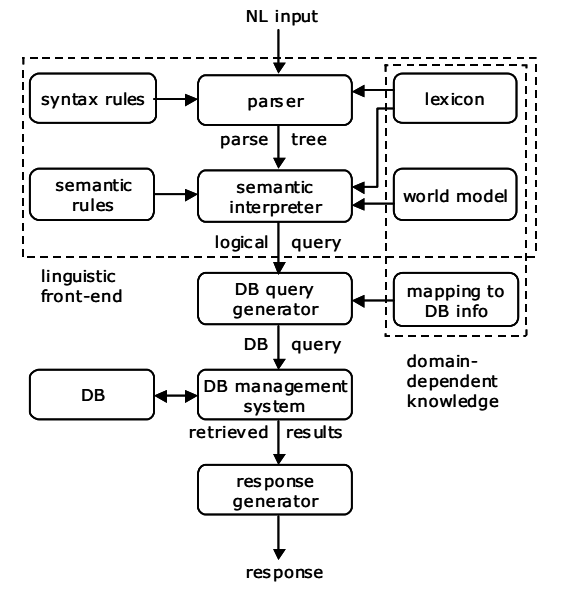


Figure 1 Traditional NLP stack (Andrenucci & Sneiders, 2005)

There are several limitations to the traditional NLP workflow for question answering, such as:

1. Each NLP module is usually designed individually for their specific roles, not for working together with other modules in a system. For instance, a module at the front of the workflow cannot easily adjust its output to provide more useful output for a module later in the workflow.
2. The architecture of the system must be designed by experts in linguistics and NLP, yet one system cannot be easily adapted to perform a different task. This limits the viability of such systems in production use.
3. Such a system mostly derives its language-related knowledge from pre-defined rulesets and language models rather than discovering the structure in the input documents.

# Datasets and Benchmarks

To compare different question answering techniques as well as to validate the effectiveness of new models, many benchmarking datasets and tools have been developed and adopted by researchers over the years. For instance, the TREC (Text Retrieval Conference) datasets are widely used for information retrieval benchmarking. For text comprehension and reasoning over text data, traditionally there was a lack of large, high quality datasets suitable for the task. In recent years, several new datasets have been proposed to meet these demands.

1 Bill went back to the cinema yesterday.

2 Julie went to the school this morning.

3 Fred went to the park yesterday.

4 Yesterday Julie went to the office.

5 Where was Julie before the school? office

## CNN / Daily Mail

The CNN / Daily Mail QA dataset was collected by Hermann, et al. (2015) for developing their deep neural network-based question answerers. The dataset consists of more than 300k articles taken from CNN and Daily Mail websites. Questions on each article are built from the bullet points for these articles. In order to truly test an algorithm for its ability to extract information from the text itself rather than relying on “common sense knowledge” (such as those deduced from word co-occurrence), Hermann et al. also anonymised the named entities in the corpora.

Figure 2 Sample bAbI data

In Chen, et al. (2016) this dataset is studied for its effectiveness in evaluating text comprehension models. The authors conclude that this dataset is valuable for training QA models, however it has several limitations, such as noisy data, relatively simple reasoning tasks and limited room of improvement for future models.

## bAbI

The bAbI dataset was constructed by Weston, et al. (2015) specifically for evaluating a model’s ability to reason over natural language evidences. The dataset generally consists of “stories” in which a set of simple statements are followed by a question based on previous statements. There are a total of 20 different categories of tasks, varying in the number of evidences needed for each question and the type of reasoning required. A sample of the bAbI data is given in figure 2.

Compared to the CNN / Daily Mail dataset, the vocabulary size and syntax complexity of the bAbI dataset is significantly lower. However, due to the bAbI dataset being designed around challenging the model’s ability to perform certain types of reasoning, the inference difficulty of it is higher compared to the CNN / Daily Mail dataset. Several notable reasoning models developed later used the bAbI set as the standard benchmarking tool, such as Weston et al.’s own Memory Networks, Dynamic Memory Networks (Kumar, et al., 2016) and Differentiable Neural Computers (Graves, et al., 2016).

## TriviaQA

TriviaQA (Joshi, et al., 2017) is a new dataset for question answering designed to overcome many of the shortcomings of older datasets, such as dataset scale, evidence type (i.e. type of reasoning required), syntactical variation, vocabulary size, etc. It is constructed by combining trivia questions with supplementary evidence documents collected from web searches and wiki pages. The quality and benchmarking effectiveness of this dataset are yet to be further tested.

# Deep Learning in NLP

The rise of deep neural networks sparked a whole new round of research into natural language processing. Specifically, the effectiveness of recurrent neural networks at sequence processing and the surprising usefulness of embedded word vectors enabled the direct (sometimes even end-to-end) application of neural network-based models in relatively complex NLP tasks such as sentence parsing, transcription, translating as well as question answering. In this section, we discuss the key concepts and techniques used in neural network NLP relevant to text comprehension.

## Recurrent Network Architectures

Recurrent neural networks are neural networks with cycles in its connections. Conceptually, the circular connections are usually unrolled and represented as a connection from the network in time step t-1 to t. This allows the network to pass state representations between time steps and therefore capture long-distance relationships within the input sequence. This is essential for NLP, as medium to long-distance dependencies frequently exist in phrases, sentences, and documents. A few modifications to the basic architecture have been designed to decrease the training difficulty and increase the representation power of RNN, such as LSTM (Hochreiter & Schmidhuber, 1997) and GRU (Cho, et al., 2014). Frequently, a bidirectional RNN is used to allow information to flow from the end of the sequence back to the beginning. In QA tasks, these designs are commonly used to encode sentences and questions, perform reasoning over facts and generate answer sequences.

## Word Embeddings

The development of techniques to embed words in a dense lower-dimensional vector space is crucial to almost all types of NLP tasks. Prior to the adoption of these techniques, most NLP processes use one-hot word vectors to represent individual words in a document. This approach has the obvious drawback of being extremely sparse and unable to capture relationships between words. Word embedding generation algorithms such as word2vec (Mikolov, et al., 2013) and GloVe (Pennington, et al., 2014) provide us with generic means to create semantically meaningful embeddings for various types of NLP tasks. These embedding techniques exploit the context similarities of words and generate embeddings that usually place semantically or functionally related words close together in the embedded space.

In addition to these general embedding algorithms, it is also possible to train a task-specific embedding by having a neural network find an optimal embedding for the training objective of the network. This is sometimes preferred when the vocabulary size is relatively small and the training examples are abundant. It is also possible to initialise word embeddings in a neural network with a pre-trained generic embedding such as GloVe and fine-tune the embedding with gradients from task objectives.

In QA tasks, words are normally considered as the most basic unit of the documents (character-level models are rare as far as we know), therefore a word embedding layer is usually the first layer of a neural network model for QA. In tasks where the vocabulary size is relatively small (such as bAbI), the usefulness of pre-trained word embeddings are limited, as it is easy for the network to learn the function and relationship of the vocabulary in its own embedding layer(s). However, for tasks with a larger vocabulary, especially when certain words may not appear or only appear a handful of times in the training data, and when the training data size is limited, an expressive word embedding layer might be crucial for the generalisation power of the network.

## Phrase and Sentence Representations

It is often not sufficient to obtain vector representations of natural languages at word level. In almost all NL QA task settings, facts are presented in sentences and paragraphs. There are usually two strategies to convert sentences into vector inputs that can be accepted by a neural network: treating the whole document as a word sequence with separators (including both natural punctuations and artificially inserted dividers), or representing each sentence as a single vector. Both strategies have been used in notable works on text comprehension. For the second strategy, there are more options in how to encode a sentence as a single vector. Weston, et al. (2014) explored two of the most common techniques for combining words into a single sentence, namely weighted average of word vectors and RNN output over a word vector sequence (more details in the next section).

Another technique of interest is to exploit the recursive structure of a sentence and apply a tree-CNN on a sentence to recursively encode words into phrases then into sentences, such as used in Kalchbrenner, et al. (2014). There lacks a systematic analysis of whether this type of sentence embedding is capable of improving the performance of text comprehension algorithms, but we suspect that the additional incorporated syntactical information could potentially be useful for tasks with long complex sentences where learning about the syntactical structure would have taken up a significant portion of the network’s capacity.

# Question Answering with Neural Networks

## Comparison of Traditional Approaches with Neural Network Methods

Collobert, et al. (2011) proposed an multi-purpose neural network model for four different NLP tasks. Although not engineered carefully to utilise elaborate linguistic features, the model compares competitively with the state-of-the-art non-neural network NLP systems at the time, with some networks reaching within 1% of the best model. Even though not a question answering model, this demonstrates that neural network models have great potential in natural language modelling and understanding.

In Hermann, et al. (2015) a comparison is made between the performance of traditional symbolic matching models and neural network models in CNN / Daily Mail reading comprehension tasks. The benchmarking shows an average 10%+ improvement in accuracy of the best neural network models over the best symbolic matching models.

With more sophisticated model design (such as the architectures mentioned below), deep learning models have managed to exceed the performance of traditional NLP approaches in multiple domains. Apart from the improvement in methodology, the availability of large training data and the increase in computation power has contributed greatly to the rise of neual NLP models, as with other sub-fields of deep learning.

## Memory Network

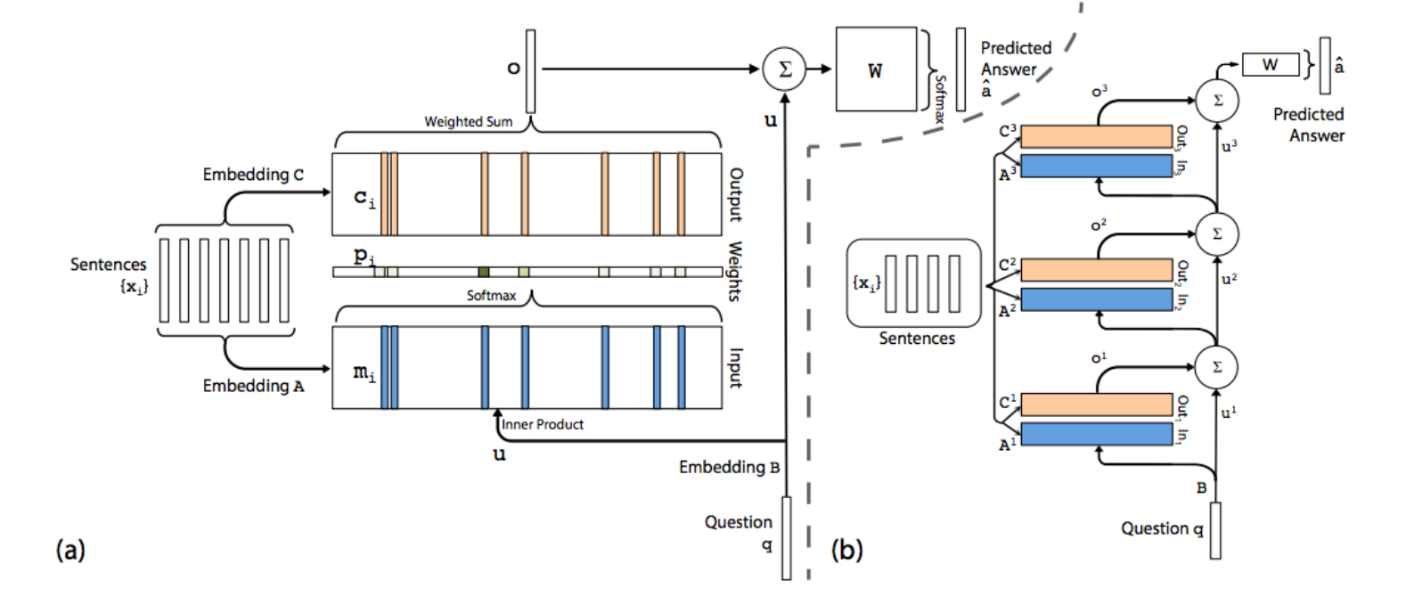
One of the most notable works in question answering with reasoning and inference is the Memory Network (MemNN) (Weston, et al., 2014). The main contribution of this research over earlier deep models such as deep LSTM is the introduction of an explicit long-term memory module, allowing information in the network to flow not only from the start of one layer to the end (as enabled by RNN layers), but also from one “scan” of the document to the next, allowing previous states (representing the intermediate result of reasoning) to direct and affect the re-interpretation of the facts at the next time step (via an attention mechanism), thus making multi-step reasoning and fact extraction more viable.

Figure 3 Memory Network (Sukhbaatar, et al., 2015)

The memory network model is further improved in Sukhbaatar, et al. (2015) (figure 3), allowing it to be trained end-to-end and to be easily appliable to different tasks. In this version of the network, it is able to be trained to map a couple (facts, question) to an answer, which is usually represented as a probability vector over a limited vocabulary. The results on the bAbI dataset from the above two research demonstrate that the Memory Network is capable of performing exceptionally well on multiple types of reasoning tasks, approaching or even reaching zero error rate in some cases, but are still struggling with certain types of tasks such as “positional reasoning” and “path finding” tasks (Weston, et al., 2014) (Sukhbaatar, et al., 2015).

There are still many potential areas of improvement for the Memory Network model, some of which are addressed in later research works (such as the DMN meontioned below). The representation of input sentences and the interaction of questions and facts are achieved with weighed averaging and inner product respectively, which are relatively simple approaches. The memory size of the network determines the maximum number of facts the model is able to process at the same time, which is not efficient when the fact input is long and irrelevant facts have to stay in memory for the entire duration. Despite a lack of further investigation, we suspect that the number of “rescans” (or reasoning layers) in the network is related to the network’s ability to perform multi-step reasoning (it is observed in the results of Sukhbaatar et al. that more “hops” or reasoning layers generally increase the performance on multi-step reasoning tasks). It is difficult to estimate for a given task, how many reasoning layers is optimal, and it is unknown whether the model can trained with variable number of layers.

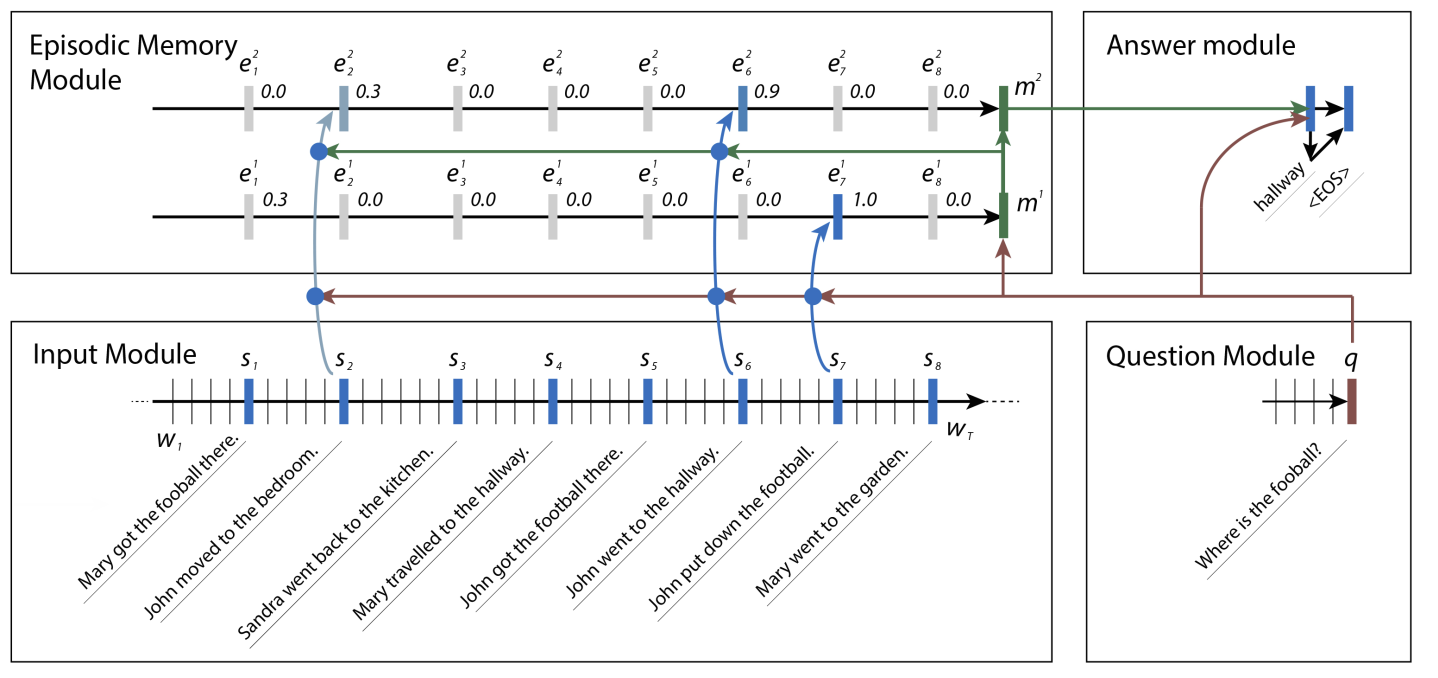
A variant of the Memory Network model, the Key-Value Memory Network (Miller, et al., 2016) is developed to tackle the limitations of memory manipulation and content addressing in MemNN. In this model, the memory is broken down into a key and a value. The key is encoded to facilitate question-based addressing whereas the value is used for reading during answer-seeking. This model performs significantly better than MemNN in answering questions based on Wikipedia articles, and performs competitively against previous best results (Miller, et al., 2016), displaying the effectiveness of the key-value approach in dealing with longer documents with more sparse information. The idea of content-addressable memory expands the flexibility of an external memory module, making it more like a hybrid of neural network cells and classical computer memories. This idea is further explored in the DNC model (Graves, et al., 2016) (discussed below).

Figure 4 Dynamic memory Network (Kumar, et al., 2016)

## Dynamic Memory Network

The Dynamic Memory Network (DMN, figure 4) (Kumar, et al., 2016) is a model similar in structure to MemNN, but can tackle more flexible input or output types in QA problems. The main difference between DMN and MemNN are: 1. Instead of requiring the inputs to the memory module to be vector representations of sentences as in MemNN, it can be either sentence sequences or word sequences, depending on the required granularity of the answer (an RNN is used to encode sentences); 2. The DMN uses a large set of similarity measures as features for the attention mechanism (for selecting memory locations to update), rather than MemNN’s dot product alone; 3. The DMN also generates an “episodic memory” at each time step from the previous episodic memory and the “episode” (memory of individual inputs) via a GRU, which is absent in MemNN. The episodic memory is later used as input to the answer module. Overall, the DMN is identical in information flow structure to MemNN, but has some adjustments to increase its representation power and IO flexibility.

Experiments on the bAbI dataset shows an overall increase of performance over MemNN, although in certain tasks the accuracy is slightly lower. It still does not perform well in two specific tasks (“positional reasoning” and “path finding”). However, in addition to bAbI QA tasks, the DMN model is also capable of sentiment analysis and PoS tagging, with respectable results. (Kumar, et al., 2016)

The most relevant contributions in this work of Kumar et al. are: 1. The introduction of the episodic memory as a state representation of current reasoning; 2. A detailed analysis on the effect of number of passes (similar to the number of “hops” / reasoning layers in MemNN) and task performances. It is found that more passes generally increase answer accuracy. The effect is more pronounced in complicated tasks. Also, the attention foci of the network generally shift to more contextually important parts of the input in later episodes. However, Kumar et al. also noted that having more passes will cause overfitting problems in some tasks.

## Neural Reasoner

The Neural Reasoner (Peng, et al., 2015) is a conceptually simple stacked DNN model for QA tasks (figure 5). The model takes RNN-encoded question and facts as input, combines the question and each fact, then outputs a new question and fact representations at each reasoning layer, and at the final answering layer, the last question representation is used as input to generate the answer. Intuitively, each reasoning layer “extracts” relevant information from the facts into the question representation, and outputs a re-assessed version of the facts after the extraction. At the final layer, all useful information to answer the question should have been incorporated in the question representation.

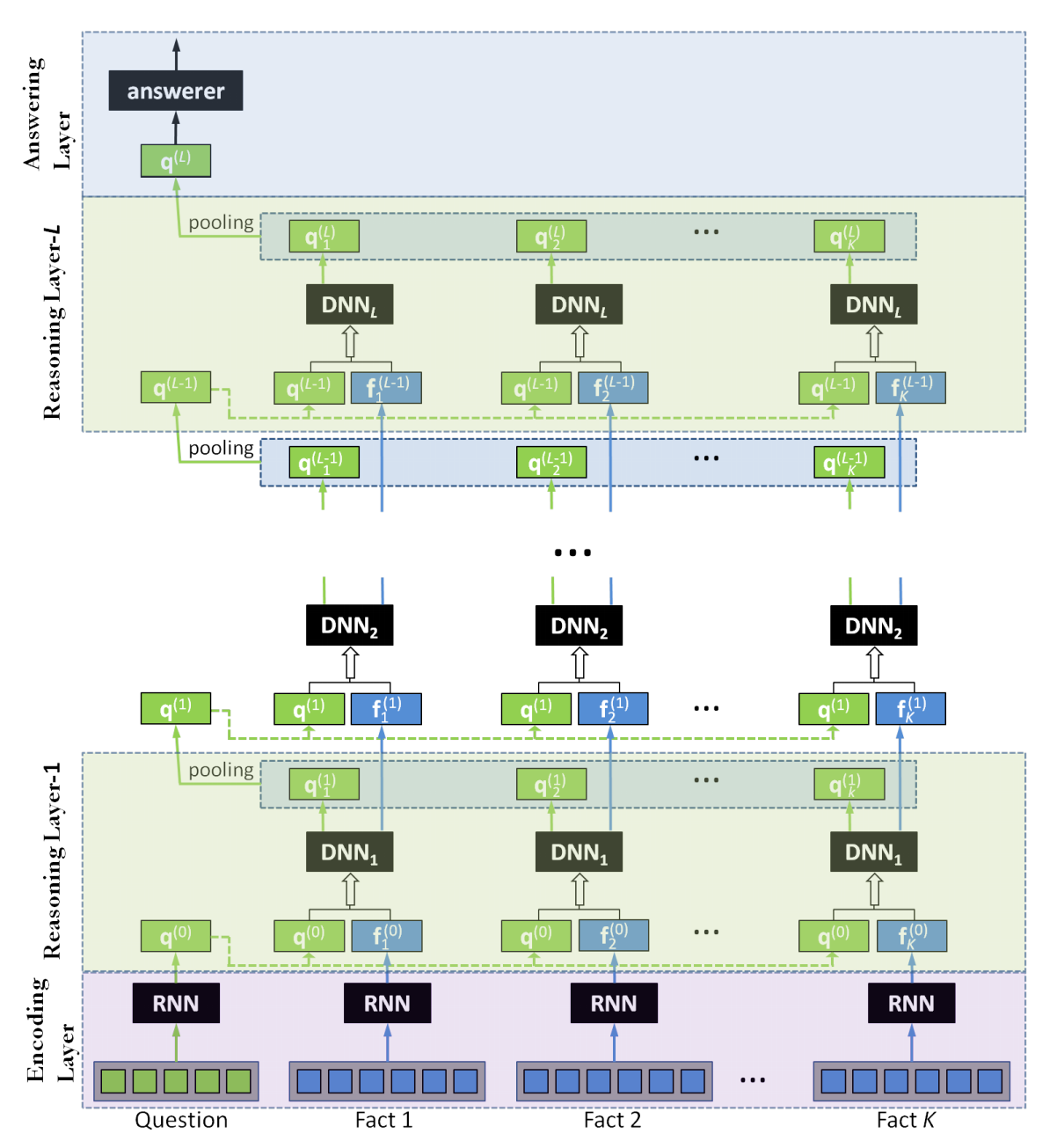


Figure 5 Neural Reasoner (Peng, et al., 2015)

According to Peng et al., the Neural Reasoner performs exceptionally well on two of the most difficult tasks in the bAbI dataset (“positional reasoning” and “path finding”) when using a large (10K) training set, vastly outperforming both MemNN and DMN. However, there is a lack of benchmarking results for other bAbI tasks or on additional datasets. We suspect that the neural reasoner is not an ideal model for temporal reasoning due to lack of internal design to cope with it (such as the memory-summarisation RNN in the DMN model that runs along the temporal direction of the inputs). Also, due to the use of stacked DNNs, the Neural Reasoner model is much deeper than either MemNN or DMN, making training and parameter tuning a big challenge (as can be seen from the poor performance of the network with the 1K examples dataset).

One notable contribution of Peng et al. is the introduction of an auxiliary task to jointly train the embedding layer of the network. Because of the depth of the network, sentence embedding learning (which is the first layer) is difficult on its own. For this reason, reconstructing the stem of input sentences from their vector representations is used as a secondary goal to assist the learning of sentence representations. This is analogous to the unsupervised pre-training a network with autoencoders, and is also related to the idea of using pre-trained word embeddings for NLP networks. Generally, pre-training is useful as a feature extractor when labelled training examples are limited but unlabelled data is abundant, it provides better generalisation for the final model, and it eases the training of deep networks (Erhan, et al., 2010). As NLP tasks usually have far more unlabelled data than labelled data, and new models proposed are getting deeper, unsupervised pre-training / joint training for representation layers can become more relevant for future models.

## Differentiable Neural Computer

The differentiable neural computer (DNC, figure 6) (Graves, et al., 2016) is an external memory-augmented neural network model built to deal with multiple types of tasks in a way more akin to conventional computers. Instead of having the network learn a mapping from inputs to outputs directly, it trains the network to learn a set of operations to manipulate the memory and ultimately to generate the desired output. The network acts as a controller that not only handles input and output, but also issues and receives operations to read and write the memory. The main novelty of this research is the use of differentiable functions in all I/O, operation generation and operation execution steps which enables the training of the network as a whole.

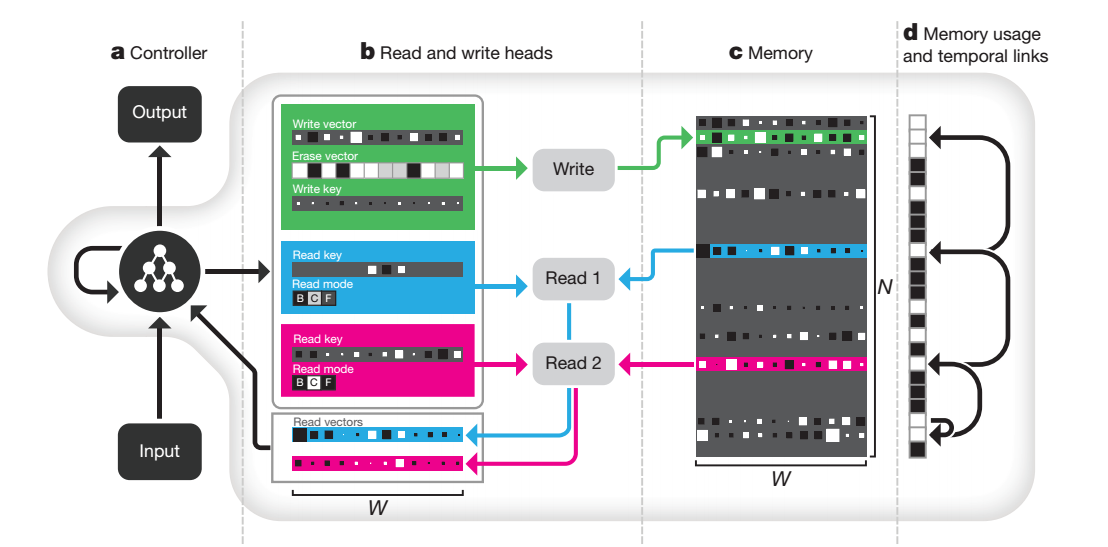
Graves et al. have shown in their work that the DNC is capable of learning text comprehension tasks, in this case using the bAbI dataset. The network takes individual word tokens as input (rather than whole sentence representations as in MemNN, DMN or NR) and outputs an answer when a “question end” token is encountered. An interesting difference between the DNC and previously mentioned models is that there are no explicit reasoning steps or revisiting of facts in the DNC model. The network learns the operations to store memory about the facts in the memory and to construct answers from the memories directly through supervised learning. Intuitively, it works like an active note-taker, who reads the facts, take “notes” about important information in the facts, reads the question, then piece together the answer from previous notes. The DNC is able to achieve lower error rates and task failure rates on bAbI tasks than previous models with only relatively weak performance in two of the tasks.

Figure 6 DNC (Graves, et al., 2016)

The DNC is an interesting idea to combine the advantages of conventional and neural computing, and has shown great promise in its ability to solve a diverse set of tasks. It is worth further investigation to explore its application in question answering.

# Information Flow Analysis of Above Models

In this section, we analyse the direction of information flow in the notable models discussed above to investigate their differences in reasoning mechanism and whether the model adjusts its behaviour based on the question or reasoning state (level of goal-awareness).

Below is the basic flow of information for the models analysed in the previous sections. Arrows are used to indicate the dependencies between modules or variables.

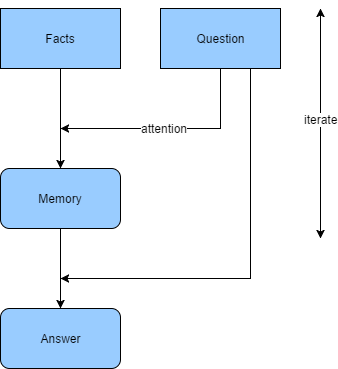


Figure 7 MemNN & DMN

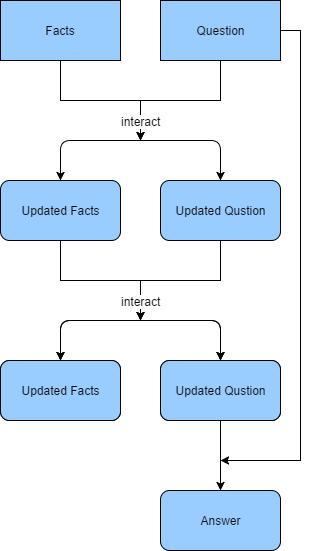


Figure 8 Neural Reasoner

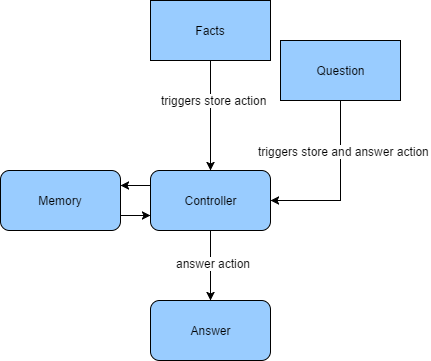


Figure 9 DNC

As we see from the information flow, both the MemNN and DMN iteratively extract information from the facts and store them in the memory for the answering task. The iterations are “guided” by the question, and the answering module is also question-driven. The neural reasoner has a similar flow of information from facts and question to answer, yet it lacks the memory component, using “updated representations” to pass information from layer to layer. MemNN and DMN read inputs multiple times to update the memory, whereas NR only reads the input once. The memory module can be seen as a highway between reasoning layers, and when coupled with attention, it allows the network to integrate reasoning outcomes with different focus on the facts at different time steps (i.e. reasoning layers). For the neural reasoner, all information relevant to question answering must be passed on through each reasoning layer. We speculate that this is a reason that MemNN / DMN is easier to train than NR (which requires auxiliary tasks to initialise the embedding layer).

None of MemNN, DMN or NR is goal-aware despite having question-driven fact representation. Goal-aware learning requires using the question and the memory state to assess if the current reasoning state is ready for answer extraction. A possible goal-aware network will have the following information flow:

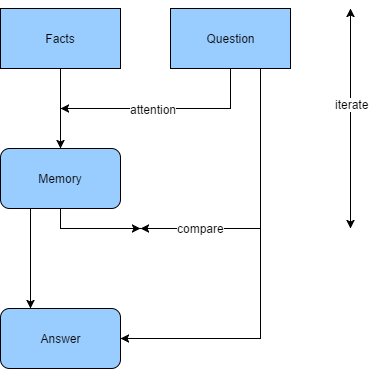


Figure 10 (Proposed) Goal-Aware Network

The DNC also only reads the facts and question once before answering. Moreover, it reads the question only after all the facts have already been read. Therefore, to answer a future unknown question, the DNC must learn to store all meaningful information in the facts in the memory rather than using question-guided attention to selectively focus on parts of the facts. In this sense, the DNC uses its memory in a very different way from MemNN and DMN. While the latter use their memory to store the state of reasoning, the DNC uses it to memorise the entire story. In fact, the entire process of reading the story and question into memory can be perceived as a text embedding process, and only the final learned operation to output the answer corresponds to the reasoning steps in other models.[[1]](#footnote-1) The DNC is ideal for tasks that require “one-shot reading”.

The input reading stage of the DNC is not goal-aware; however, we suspect that the learned operations to extract the answer from memory can be trained to be goal-aware.

# Reinforcement Learning

Compared to supervised or unsupervised learning, there are fewer works on applying reinforcement learning to NLP or QA in particular. However, recently, deep learning-powered reinforcement learning has gained traction in various domains including game playing, robot controls, self-driving vehicles etc., and there are a few successful attempts on using RL techniques to solve NLP problems, making it a promising direction for future research.

## Deep Q Network and other Deep Learning Approaches

A highly impactful work on RL in recent years is the work on Deep Q-networks (DQN) (Mnih, et al., 2013), which shows that:

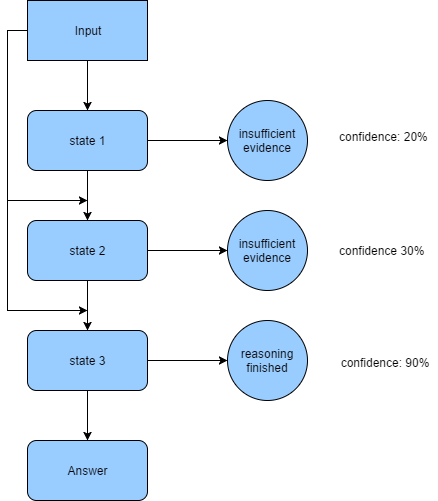
1. The value function in a Markov decision process (MDP) can be reliably learned by a deep neural network with proper training measures.
2. The trained agent can solve a wide variety of tasks (in this case, seven different Atari games), and it is possible to train one single agent for multiple tasks.

Additional research further improves the feasibility and lowers the difficulty of applying reinforcement learning to more challenging tasks. Notable techniques in deep reinforcement learning include double Q-learning (Van Hasselt, et al., 2016), asynchronous methods (Mnih, et al., 2016), etc.

## Reinforcement Learning in QA Tasks

Celikyilmaz et al. (2017) proposed an unique approach to the text comprehension problem called the Scaffolding Network (SN). In their model, there are two competing network modules: one Q-network for generating answers and one question generator network for generating simulated questions. Without the question generator, the structure of the model is simular to that of the DMN. Insead of having the answering network to directly output answers, it is first challenged with simulated questions from the question generator before answering the actual question. The Q-network is trained to maximise payoff for successfully answering simulated (low score) and actual (high score) questions whereas the question generator is trained to minimise the success rate of the Q-network i.e. challenge it with difficult questions. The novelty here are:

1. Use adversarial training in the question generator to generate more “valuable” questions and force the Q-network to try harder in answering simulated questions. The idea of having the network question itself is very different from previous approaches, which focus less on consolidating existing knowledge about the facts. However, the question simulator on its own is not necessarily better than having more genuine training examples.
2. Use reinforcement learning to let the Q-network base its decision for the actual question on the feedbacks from simulated questions. We believe this is what sets this approach apart from having more training examples, as the results from simulated questions, right or wrong, can be valuable information in answering the actual question afterwards. This cannot be achieved by expanding the training dataset.

Celikyilmaz et al. reports that the SN is not yet outperforming previous models on the bAbI dataset, yet it is able to achieve respectable results. Nevertheless, this has proven that there is potential in applying reinforcement learning and adversarial training to text comprehension in future research.

# Goal-Aware Reasoning: How to Get There

The basic idea of goal-aware reasoning we propose is that the network should be able to keep track of the completion status of reasoning, and depending on the status, employ a variable-depth network (in other words, run for a variable number of iterations) to extract the answer. This idea is not restricted to one single neural network architecture, but can be applied to all reasoning networks with multiple similar layers or iterative structures. The intuition behind this is that each pass-through of the reasoning layer will push the reasoning state (can be memory states or intermediate layer outputs) closer to what is required to answer the question, and if we can restrict the reasoning layers to perform a single type of operation on the previous reasoning state (by means such as weight-tying), questions of different difficulty should require different number of reasoning steps to get close enough to its desired state. We will be designing a mechanism to estimate the distance of current reasoning state versus the desired reasoning state, or the confidence of the network to answer the question if it were to do so at current reasoning step. Figure 11 shows a potential episode of goal-aware reasoning process.

Figure 11 Goal-Aware Reasoing Process

The base model with which to incorporate goal-aware reasoning can be MemNN, DMN, NR or other similar models. However, there are some challenges in designing and implementing the goal-aware approach:

1. The reasoning layers in existing models do not always have the same weights, allowing them to perform different types of calculations. This is problematic for a variable-depth model, as it will result in a theoretically unbounded number of weights to train. Weight-tying will be necessary for the reasoning layers. Fortunately, Sukhbaatar et al. (2015) has shown that employing wieght-tying in their MemNN model do not have significant negative performance impacts.
2. The training of a veriable-depth network can prove difficult and unstable. Although there is no technical difficulty in applying back-propagation in variable-depth networks, it will make the common form of minibatch training impossible (due to each training instance requiring different network depths) and risk not being able to find an optimum that works for all depths.
3. The confidence measure is yet to be defined here. So far, we do not have a trainable measure to determine how “ready” a reasoning state is for answering a given question. Joint-training and reinforcement learning techniques will be explored in finding a reliable measure of answer confidence.

Despite the difficulties, goal-aware learning will enable the creation of a more flexible neural reasoning model that can adapt to a large variety of question difficulty and type of reasoning. We expect a well-tuned goal-aware network to obtain better multi-task performance over the corresponding goal-agnostic variants in benchmarks such as bAbI.

# Other Potential Future Research Areas

Despite active research on text comprehension and neural reasoning, there still exist several additional gaps in current research that are not well-addressed by existing works. Below are two of the examples:

## Choice of Natural Language Representations as Network Input

In section 5, we discussed a multitude of ways to represent words and sentences as dense vectors. In neural network models, these embedding techniques are often used to encode natural languages as provide inputs to the networks. The quality of the embeddings is crucial to the performance of the networks upon which all following calculations depend. Most of the existing models use either word weight averaging or bidirectional RNNs to encode sentences based on individual word embeddings, which is sufficient for benchmarks with low syntactical complexity, such as bAbI. However, this approach can become problematic when there are more syntactical variations in the data, and the reasoning task depends on unwinding the syntax. Language modelling and sentence parsing are relatively difficult tasks on their own, therefore it is doubtful that a network can simultaneously learn to perform syntactical understanding and logical reasoning. We speculate that in such cases, incorporating word dependency information in the input or using recursive embedding schemes (such as applying tree-LSTM to the parse tree) might help performance by utilising the structure of sentences.

It would also benefit the research community to develop a similar synthetic benchmark to bAbI but with multiple adjectives and simple clauses.

## Human-Readable Layer Visualisation

One of the challenges of studying reasoning networks is the difficulty in understanding the role of each layer in answer inference. Unlike image processing networks, it is much harder to generate a human-understandable visualisation of layer weights in a reasoning network with which we can theorise the function of the layers. Some previously used techniques include visualisation of attention focus (Sukhbaatar, et al., 2015) (Kumar, et al., 2016), direct visualisation of memory cell values (Sukhbaatar, et al., 2015) or visualisation of neuron activation over time (Graves, et al., 2016). It would help the research community to develop a generalisable visualisation method to explain the function of individual reasoning steps in a stacked / iterative reasoning network.

# Conclusion

In this review, we proposed the concept of goal-aware reasoning, with which a QA model is able to adjust its reasoning steps based on the completion state of reasoning. We briefly expanded on the background of question answering and text comprehension, as well as the important breakthroughs in neural NLP that facilitates its usage in question answering. We focused our attention on recent neural QA models, discussed about their contributions and analysed how they are relevant to goal-aware reasoning. We outlined the potential difficulties and contributions of developing a goal-aware reasoning model. In addition, we proposed two other potential areas of future research.

Through this review, we have found goal-aware reasoning to be a natural extension of current research, with plenty of related works to draw inspirations from. We also believe the potential contributions of goal-aware reasoning will be worthwhile.

# References

Andrenucci, A. & Sneiders, E., 2005. Automated question answering: Review of the main approaches. *Third International Conference on Information Technology and Applications,* Volume 1, pp. 514-519.

Celikyilmaz, A., Deng, L., Li, L. & Wang, C., 2017. Scaffolding Networks for Teaching and Learning to Comprehend. *eprint arXiv:1702.08653.*

Chen, D., Bolton, J. & Manning, C., 2016. A thorough examination of the cnn/daily mail reading comprehension task. *eprint arXiv:1606.02858.*

Cho, K. et al., 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *eprint arXiv:1406.1078.*

Collobert, R. et al., 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research,* 12(Aug), pp. 2493-2537.

Erhan, D. et al., 2010. Why does unsupervised pre-training help deep learning?. *Journal of Machine Learning Research,* 11(Feb), pp. 625-660.

Graves, A. et al., 2016. Hybrid computing using a neural network with dynamic external memory. *Nature,* 528(7626), pp. 471-476.

Hermann, K. et al., 2015. Teaching machines to read and comprehend. *Advances in Neural Information Processing Systems,* pp. 1693-1701.

Hirschman, L. & Gaizauskas, R., 2001. Natural language question answering: the view from here. *Natural Language Engineering,* 7(4), pp. 275-300.

Hochreiter, S. & Schmidhuber, J., 1997. Long short-term memory. *Neural computation,* 9(8), pp. 1735-1780.

Joshi, M., Choi, E., Weld, D. & Zettlemoyer, L., 2017. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. *eprint arXiv:1705.03551.*

Kalchbrenner, N., Grefenstette, E. & Blunsom, P., 2014. A convolutional neural network for modelling sentences. *eprint arXiv:1404.2188.*

Kumar, A. et al., 2016. Ask me anything: Dynamic memory networks for natural language processing. *International Conference on Machine Learning,* pp. 1378-1387.

Lehnert, W., 1977. A conceptual theory of question answering. *Proceedings of the 5th international joint conference on Artificial intelligence,* Volume 1, pp. 158-164.

Mikolov, T., Chen, K., Corrado, G. & Dean, J., 2013. Efficient estimation of word representations in vector space. *eprint arXiv:1301.3781.*

Miller, A. F. et al., 2016. Key-value memory networks for directly reading documents. *eprint arXiv:1606.03126.*

Mnih, V. et al., 2016. Asynchronous methods for deep reinforcement learning. *International Conference on Machine Learning,* pp. 1928-1937.

Mnih, V. et al., 2013. Playing atari with deep reinforcement learning. *eprint arXiv:1312.5602.*

Peng, B., Lu, Z., Li, H. & Wong, K., 2015. Towards neural network-based reasoning. *eprint arXiv:1508.05508.*

Pennington, J., Socher, R. & Manning, C., 2014. Glove: Global Vectors for Word Representation. *EMNLP ,* Volume 14, pp. 1532-1543.

Sukhbaatar, S., Weston, J. & Fergus, R., 2015. End-to-end memory networks. *Advances in neural information processing systems,* pp. 2440-2448.

Van Hasselt, H., Guez, A. & Silver, D., 2016. Deep Reinforcement Learning with Double Q-Learning. *AAAI ,* pp. 2094-2100.

Weston, J. et al., 2015. Towards ai-complete question answering: A set of prerequisite toy tasks. *eprint arXiv:1502.05698.*

Weston, J., Chopra, S. & Bordes, A., 2014. Memory networks. *eprint arXiv:1410.3916.*

1. There is some oversimplification here. Strictly speaking, while reading a story, the reading of earlier facts can affect the reading of later facts through the controller LSTM’s hidden states. However, according to Graves et al. (2016), the DNC usually learns to separate memory from processing. In this case, it will learn to use different memory locations to store individual facts. Further investigation on the memory usage behaviour of the DNC is needed. [↑](#footnote-ref-1)