CS224n: Natural Language Processing with Deep Learning [1]

Lecture Notes: Part VII

Question Answering [2]

Winter 2019

[1] Course Instructors: Christopher Manning, Richard Socher

[2] Authors: Francois Chaubard, Richard Socher

1 *Dynamic Memory Networks for Question Answering over Text and Images*

The idea of a QA system is to extract information (sometimes passages, or spans of words) directly from documents, conversations, online searches, etc., that will meet user's information needs. Rather than make the user read through an entire document, QA system prefers to give a short and concise answer. Nowadays, a QA system can combine very easily with other NLP systems like chatbots, and some QA systems even go beyond the search of text documents and can extract information from a collection of pictures.

There are many types of questions, and the simplest of them is factoid question answering. It contains questions that look like "The symbol for mercuric oxide is?" "Which NFL team represented the AFC at Super Bowl 50?". There are of course other types such as mathematical questions ("2+3=?"), logical questions that require extensive reasoning (and no background information). However, we can argue that the information-seeking factoid questions are the most common questions in people's daily life.

In fact, most of the NLP problems can be considered as a question-answering problem, the paradigm is simple: we issue a query, and the machine provides a response. By reading through a document, or a set of instructions, an intelligent system should be able to answer a wide variety of questions. We can ask the POS tags of a sentence, we can ask the system to respond in a different language. So naturally, we would like to design a model that can be used for general QA.

In order to achieve this goal, we face two major obstacles. Many NLP tasks use different architectures, such as TreeLSTM (Tai et al., 2015) for sentiment analysis, Memory Network (Weston et al., 2015) for question answering, and Bi-directional LSTM-CRF (Huang et al., 2015) for part-of-speech tagging. The second problem is full multi-task learning tends to be very difficult, and transfer-learning remains to be a major obstacle for current neural network architectures across artificial intelligence domains (computer vision, reinforcement learning, etc.).

We can tackle the first problem with a shared architecture for NLP: Dynamic Memory Network (DMN), an architecture designed for

general QA tasks. QA is difficult, partially because reading a long paragraph is difficult. Even for humans, we are not able to store a long document in your working memory.

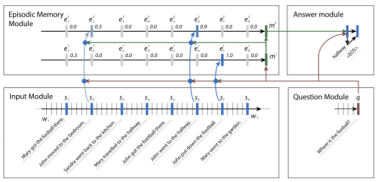

Figure 1: A graphical illustration of the Dynamic Memory Network.

1.1 Input Module

Dynamic Memory Network is divided into modules. First we look at input module. The input module takes as input a sequence of T_I words and outputs a sequence of T_C fact representations. If the output is a list of words, we have $T_C = T_I$ and if the output is a list of sentences, we have T_C as the number of sentences and T_I as the number of words in the sentences. We use a simple GRU to read the sentences in, i.e. the hidden state $h_t = \text{GRU}(x_t, h_{t-1})$ where $x_t = L[w_t]$, where L is the embedding matrix and w_t is the word at time t. We further improve it by using Bi-GRU as shown in Figure 2.

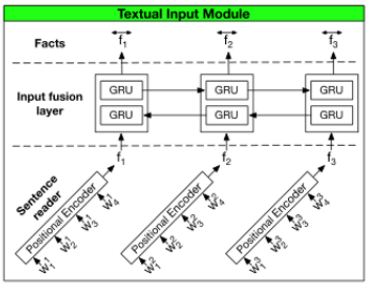

Figure 2: A graphical illustration of the Dynamic Memory Network.

1.2 Question Module

We also use a standard GRU to read in the question (using embedding matrix L: $q_t = \text{GRU}(L[w_t^Q], q_{t-1})$), but the output of the question module is an encoded representation of the question.

1.3 Episodic Memory Module

One of the distinctive features of the dynamic memory network is the episodic memory module which runs over the input sequence multiple times, each time paying attention to a different subset of facts from the input.

It accomplishes this using a Bi-GRU that takes input of the sentence-level representation passed in from the input module, and produces an episodic memory representation.

We denote the episodic memory representation as m^i and the episode representation (output by the attention mechanism) as e^i. The episodic memory representation is initialized using $m^0 = q$, and proceeds using the GRU: $m^i = \text{GRU}(e^i, m^{i-1})$. The episode representation is updated using the hidden state outputs from the input module as follows where g is the attention mechanism:

$$h_t^i = g_t^i \text{GRU}(c_t, h_{t-1}^i) + (1 - g_t^i) h_{t-1}^i$$
$$e_i = h_{T_C}^i$$

The attention vector g may be computed in a number of ways, but in the original DMN paper (Kumar et al. 2016), the following formulation was found to work best:

$$g_t^i = G(c_t, m^{i-1}, q)$$

$$G(c, m, q) = \sigma(W^{(2)} \tanh(W^{(1)} z(c, m, q) + b^{(1)}) + b^{(2)})$$
$$z(c, m, q) = [c, m, q, c \circ q, c \circ m, |c - q|, |c - m|, c^T W^{(b)} q, c^T W^{(b)} m]$$

In this way, gates in this module are activated if the sentence is relevant to the question or memory. In the ith pass, if the summary is not sufficient to answer the question, we can repeat sequence over input in the $(i + 1)$th pass. For example, consider the question "Where is the football?" and input sequences “John kicked the football" and “John was in the field." In this example, *John* and *football* could be linked in one pass and then *John* and *field* could be linked in the second pass, allowing for the network to perform a transitive inference based on the two pieces of information.

1.4 Answer Module

The answer module is a simple GRU decoder that takes in the output of question module, and episodic memory module, and output a word (or in general a computational result). It works as follows:

$$y_t = \text{softmax}(W^{(a)} a_t)$$
$$a_t = \text{GRU}([y_{t-1}, q], a_{t-1})$$

1.5 Experiments

Through the experiments we can see DMN is able to outperform MemNN in babl question answering tasks, and it can outperform other architectures for sentiment analysis and part-of-speech tagging. How many episodes are needed in the episodic memory? The answer is that the harder the task is, the more passes are required. Multiple passes also allows the network to truly comprehend the sentence by paying attention to only relevant parts for the final task, instead of reacting to just the information from the word embedding.

The key idea is to modularize the system, and you can allow different types of input by change the input module. For example, if we replace the input module with a convolutional neural network-based module, then this architecture can handle a task called visual question answering (VQA). It is also able to outperform other models in this task.

1.6 Summary

The zeal to search for a general architecture that would solve all problems has slightly faded since 2015, but the desire to train on one domain and generalize to other domains has increased. To comprehend more advanced modules for question answering, readers can refer to the dynamic coattention network (DCN).