Paper Summary

David Miller
CIS 5930: Social Network Mining

January 24, 2018

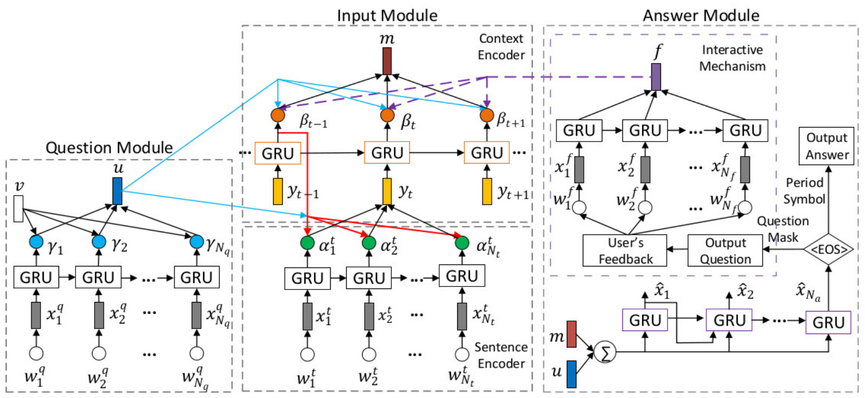

Figure 1

Question Answering is a discipline in Computer Science, specifically in the realm of natural language processing, that aims at automatically answering questions from user input. Current implementations of QA systems are based on statement-question-answer triplets where each input sentence is mapped to an input representation space regarded as a memory component. The output representation is calculated by summarizing over input representations with different attention weights. This single-layer memory is extended to multi-layer memory by reasoning the statements and the question with multiple hops [1]. This approach presents several limitations. The first problem is not determing the context of the word(s) in a sentence. Second is when the input-question pair fails to provide enough information regarding the question. An example of this is the triplet ({The master bedroom is north of the kitchen. The guest bedroom is east of the office}, What is the bedroom east of?, Unknown) [1]. The ambiguity of which bedroom leads to this failure. In contrast, the Context aware Attention Network (CAN) presented in this paper learns proper word contextual representation and interacts with the user if ambiguity is present in the answer processing phase. Figure 1 illustrates what goes on in the CAN implementation. To understand what is going on let us discuss in detail what happens at each step:

Question Module

1. Words $w_1^q, \ldots, w_{N_q}^q$ are mapped to one-hot vectors $x_1^q, \ldots, x_{N_q}^q$.

2. Vectors $x_1^q, \ldots, x_{N_q}^q$ are fed into GRUs that generate importance weights $\gamma_1, \ldots, \gamma_{N_q}$ based on vocabulary v and context.

3. The question representation u is then generated by a sum of $x_1^q, \ldots, x_{N_q}^q$ weighted by $\gamma_1, \ldots, \gamma_{N_q}$ and passed to the Input Module.

Input Module

1. For each input sentence l_t each word w_i is embedded into a word space x_i^t through the shared learned embedding matrix, and a recurrent neural network is used to capture the context information from the words in the same sentence.

2. GRUs compute each word annotation vector $h_1^t, \dots, h_{N_t}^t$ (this is a hidden state).

3. Sentence representation y_t is generated by aggregating $h_1^t, \dots, h_{N_t}^t$, computed with the same GRUs in the previous hidden state, with weights $\alpha_1^t, \dots, \alpha_{N_t}^t$.

4. Weights β_i are calculated in the GRUs.

5. Context representation vector m is calculated by summing over $y_1, \dots, y_t$ weighted by their corresponding β_i weights.

Answer Module

1. Vectors u and m are summed together and passed to GRUs to compute predicted word vectors $\hat{x}_1, \dots, \hat{x}_{N_\alpha}$.

2. If enough information is present for the answer, return output and terminate, else go to step 3.

3. Ask the user a question to get more information; user responds with a sequence of words $w_1^f, \dots, w_{N_f}^f$ and mapped to vectors $x_1^f, \dots, x_{N_f}^f$.

4. Corresponding annotation vectors $g_1^f, \dots, g_d^f$ and a representation f is constructed via aggregation of annotation vectors.

5. Representation f is then used to produce an answer based on the user's feedback.

Future work for the CAN implementation can include applications in commercial QA systems, such as Amazon echo or Google home. As we progress further, smart technology is becoming ever more so present in our homes and I feel that this will be the primary application of intelligent QA systems.

Three strengths of the paper I found were

1. The CAN QA system implements contextual awareness and user feedback during the answer module, allowing almost perfect responses to user input.
2. It is based upon previous work [2] making it an easy system to learn by those already involved in this field of research.
3. The framework of the CAN QA system makes it easily applicable to commercial systems, such as the Amazon echo, due to it's reliance on user input for ambiguity rather than highest likely match.

Three weaknesses I found with the paper are

1. The additional computational complexity of the CAN QA system seems like it might add time to user's answers, making it inefficient for commercial products. The fact that the authors did not really talk over computational time complexity hints at this fact.

2. User feedback for the answer module can actually confuse the CAN QA system if the feedback is incorrect since there does not seem to be any error checking at this stage.

3. The framework of the CAN QA system does not seem to allow for parallelization making scalability hard for the future.

Questions for the presenter

1. What happens if the user asks a question totally unrelated to the input that requires some prior knowledge from the user?

2. What is the systems response to input with no logical sequence or context?

References

[1] Huayu Li , Martin Renqiang Min , Yong Ge , Asim Kadav *A Context-aware Attention Network for Interactive Question Answering*, KDD17, August 1317, 2017, Halifax, NS, Canada.

[2] Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. 2016. Ask Me Anything: Dynamic Memory Networks for Natural Language Processing. In ICML. 13781387.