Squadbots and Deception

Problem and Background

- We consider the problem of Question Answering (QA) on the Stanford Question Answering Dataset (SQuAD 2.0).
- The objective is to design a system that answers a question using the provided context information.
- Formally, given a question of $N$ words, $q_1, \ldots, q_N$, and a context paragraph of $M$ words, $c_1, \ldots, c_M$, the QA system should return a span of context words $[\hat{c}_1, \ldots, \hat{c}_k]$ as the answer or an empty span if unanswerable.
- The QA problem is relevant to many modern-day technologies ranging from digital assistants like Siri and Alexa to the handling of Google search queries.
- This problem involves addressing many open challenges in Natural Language Processing (NLP) such as text comprehension, sequence modeling, and information retrieval.

Methods

- We trained a deep neural network that is adapted from the provided BiDAF model implementation.
- We preserved the BiDAF model structure but investigated the impact of various design choices on F1-EM performance metrics including:
  - Introducing character embeddings,
  - Replacing LSTM layers with Transformer blocks for improved global context modeling,
  - Introducing convolution layers for improved local context modeling, and
  - Model pretraining.
- Pretraining was performed using the SQuAD 2.0 dataset.
- We computed the context with random word and character vectors and train the model to reproduce the true context.
- We used an adaptive softmax layer to output the context without the corruption.

- Remarks:
  - Introducing character embeddings produces the largest performance improvement.
  - Transformers perform similarly to LSTMs for the hidden sizes permitted by our hardware memory constraints.
  - Convolution layers after the Attention Flow Layer provide small performance improvements.
  - Convolution layers before the Attention Flow Layer appear to smear per-word information, hurting performance.
  - Pretraining also yields a minor performance improvement.

Best Model

- Our best performing model (Fig. 1) uses word and character embeddings and a convolution layer between the Attention Flow and Modeling Layers.
- We found using Transformers for the Contextual Embed and Modeling Layers performed similarly.
- This model was pre-trained for 12 epochs on the corrupted input and fine-tuned on the QA task for 18 epochs.
- Training was performed with a batch size of 64. We use AdamW as the optimizer with a fixed learning rate of 0.5. Training took approximately 3 hours on an Nvidia GeForce RTX 2080 Ti.

Fig. 1: Model architecture for the best performing model. This model takes both word and character embeddings as inputs and introduces a convolution layer following the Attention Flow Layer. This model was pretrained to reconstruct corrupted context data from SQuAD 2.0.

Analysis

- Character Embedding
  - Question: What is a ligand on the cell surface that is upregulated after helper T cell activation?
  - Context: "... helper T cell activation causes an upregulation of molecules expressed on the T cell’s surface, such as CD40 ligand,..."
  - Character-level representation was required to figure out CD40, since it is a rare word.

- Understanding vs. Word Finding
  - Question: What kind of experiment did Huxley perform to test for the welfare of the group?
  - Context: "... Henry N. Huxley performed a reversion to Catholicism, who had protected Protestants through the Edict of Nantes."
  - Prediction: Henry N. Huxley
  - The model understood similarities between “reversion for the welfare of the group” and “protected.”
  - It figured out that Henry N. Huxley was a king based on other context, despite never using the word King.

- Trouble with Models such as E-HR
  - Question: What sort of motion did Newcomen’s steam engine continuously produce?
  - Context: "... James Watt patented a steam engine that produced continuous rotary motion..."
  - Prediction: rotary motion
  - It understood that “rotary motion” is linked to “steam engine” but incorrectly credited Newcomen.
  - In another example where the question asked about Watt, the model gave the correct answer.

Conclusions

- Subword modeling is crucial for questions pertaining to specialized terminology, numerical entities, or obscure words.
- Transformers seem to require significantly more parameters than LSTMs to see performance benefits.
- Convolution layers before the Attention Flow Layer appear to smear information where per-word information seems important.
- After Attention Flow, convolutions help aggregate local context for answers.
- Pretraining yields a minor performance improvement, but would likely be more useful with a larger unlabeled data set.