Exploring Subword Level Embeddings for the SQuAD Dataset

Theo Culhane tculhane@stanford.edu

Stanford
CS 229

Problem

Since I did the default final project, I am working on solving the SQuAD problem, which is answering reading comprehension questions for a paragraph of context or determining if the question cannot be answered given the context. Though SQuAD is very well studied, and so it was unlikely I would be able to make any significant contributions, it serves as a good framework for considering issues in question answering as a whole.

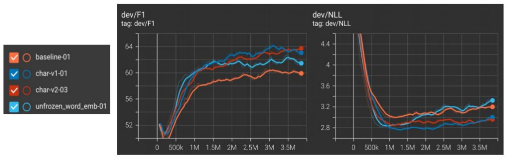
Background

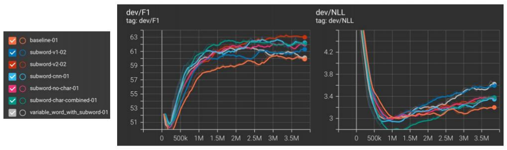
- SQuAD is very well studied, and the best models are consistently massive pretrained models that would be far larger than the scope of a single class project
- Using a slightly outdated model, BiDAF, that uses LSTMs instead of the current state-of-the-art, which is transformers

Methods

- The base BiDAF model we were given only includes word embeddings in the input layer
- I added in character level embeddings and subword level embeddings
- For character level embeddings, I used a CNN to process the embeddings for all experiments
  - Two different ways of using the CNN were tested
  - In one method, all embeddings were concatenated into one long embedding, and then a CNN was run over the result
  - In the other method, a CNN was run over each embedding separately, and then the result was max-pooled
- For subword, two methods were tried
  - In the first, all subword embeddings were concatenated and used directly
  - In the second, I used a similar CNN structure as the second character embedding methd
- I also tried combining the character and subword embeddings, after they were each processed by a CNN, using a multi-layer perceptron
- Finally, I experimented with using an unfrozen version of the word embeddings concatenated with a frozen version

Experiments





Analysis

- Subword level embeddings appear to have slightly overfitted the training data, as can be seen by how NLL on the dev set begins to increase after around 1M iterations, whereas with the baseline and character level embeddings NLL on the dev set was mostly stable once it reached its minimum
- Unfreezing the word level embeddings also appears to have induced some overfitting, which is apparent in the shape of the NLL on the dev set with unfrozen word embeddings in both graphs
- The models with no subword embeddings ended up performing slightly better on average on the dev set than the ones with subword embeddings, possibly due to the overfitting
- The experiment that included no character embeddings and only subword embeddings overfit the least out of all of the subword level embedding experiments, which seems to indicate that the issue of overfitting at least partially stems from subword and character level embeddings duplicating some information
- The model that is about tied for the worst overfitting was one in which subword embeddings were used directly, instead of processed through a CNN, which indicates that the CNN alleviates some of the overfitting/forces some sort of summarization.

Conclusion

- The more complex models ended up suffering quite a bit from overfitting, with an almost perfectly consistent correlation between complexity and overfitting
- Finding a way to more efficiently summarize all of the embeddings together early on may help to stop some of the overfitting, which would probably make the model stronger

References

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*, 2016.

Yoon Kim. Convolutional neural networks for sentence classification. *CoRR*, abs/1408.5882, 2014.