Self-Attention QA System on SQuAD 2.0

Kevin Yang kevvyang@stanford.edu
Aryan Chaudhary achaud@Stanford.edu
Affiliated with Stanford University

Abstract

In this project, we built a neural network model that can answer questions for the Stanford Question Anwering Dataset (SQuAD) 2.0. From the baseline BiDAF model, we included two key changes: R-NET self-attention, and included character embeddings. We expected these two changes to display improvements from the baseline (EM = 52.19, F1 = 55.69), and after including character embeddings, we saw a slight increase to EM = 54.51, F1 = 57.92. Our scores further improved after adding self-attention to EM = 56.15, F1 = 59.96.

Introduction

The goal of this project is to give a model the ability to pick out the important pieces of a passage and return the correct answer given a question. Obviously, there is a plethora of work in this area, even if we limit the scope to just the world-renowned SQuAD dataset. As students, we cannot hope to compete with the work of distinguished researchers who have dedicated their lives to natural language processing, but we can aspire to implement their ideas and possibly find methods to improve their baselines.

We first ran our code with the initial BiDAF model and classifier as provided in the starter code, then ran it again with additional character-level embeddings. We optimized the model using Adam, setting the learning rate to 0.1, and epsilon to 1e-6. Our dropout layer has a dropout rate of 0.2, and our hidden vector length is 75 for all purposes (attention, linear layers, etc.).

The EM score and F1 score for the baseline in the starter code were 52.19 and 55.69, respectively. After adding the character embeddings, our new scores became 54.51 and 857.92. Evidently, the implementation of character embeddings for this model improved the performance, which is the result we expected. Once we added self-attention, we saw additional jumps to 56.15 and 59.96.

Approach

Our ultimate goal was to incorporate both self-attention as well as character embeddings in order to improve the results from the baseline code.

We first approached character embeddings. The starter code had word level embeddings already implemented, and it wasn't too difficult to extrapolate from that implementation a rough picture of what character-level embeddings was going to look like. The implementation step was a little harder, as matching sizes and understanding pytorch functionalities such as max-pooling proved to be difficult at times.

In the end, we had a fully functioning character and word level embedding neural net, an improvement from the baseline, and our results of F1 and EM showed it.

To approach R-NET's self attention, we started by first extensively studying the R-NET self-attention equations, shown here:

$$s_j^t = \mathrm{v}^{\mathrm{T}} \tanh(W_v^P v_j^P + W_v^{\tilde{P}} v_t^P)$$
$$a_i^t = \exp(s_i^t) / \Sigma_{j=1}^n \exp(s_j^t)$$
$$c_t = \Sigma_{i=1}^n a_i^t v_i^P$$

After getting a firm grasp of each and every variable, we used the initial BiDAF Attention layer as our starter code and made changes until it achieved the same effect that R-NET's self attention did.

Table 1. EM/FM levels for each model on SQuAD 2.0 Dataset

	EM	FM	Average
BiDAF Starter	52.19	55.69	53.94
Char Encodings	54.51	57.92	56.22
Self-Attention	56.15	59.96	58.06
R-NET Initial	72.3	80.6	76.45

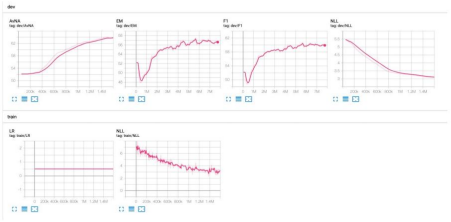

Chart 1. Character Embeddings Results

Results

] We are on the PCE leaderboard. Our final F1 and EM scores were 56.15 and 59.96. This is a nice improvement from the baseline we got of 52.19 and 55.69.

We expected a little better than 56.15 and 59.96. Within R-NET, a huge portion of the implementation they had was their self-attention neural network, which we followed pretty closely and should have an exact implementation. With that being said, they achieved way higher scores of 72.3 and 80.6, so we were expecting at least to break 60s in EM and even break 63s in F1.

The reason this may be is because we never incorporated any sorts of self-matching or question-answer matching which was incorporated within R-NET. They also most likely had more resources than us as well resulting in better

Conclusions

Throughout our project, we can see even small improvements can result in significantly better performing test results such as seen from our improvements after implementing both character encoding and self attention on top of the baseline model that only incorporates BiDAF attention and word encoding.

Our model ended up being fairly successful, improving the baseline by more than 4\% each. With more optimizations, we can expect this jump to skyrocket even more. The biggest limitation of our work has been the time constraint caused by 5 weeks within this class, and also our lack of NLP experience. Although this class has taught us a lot about NLP basics, I would say self implementation and thoroughly understanding how the initial BiDAF model worked is another level above what we've done prior in class.
In the future we would love to see where the possibilities of creating multiple layers, maybe implementing the transformer-XL or coattention. Another possible realm of exploration is how minor changes - non artitectural changes can play a role in improving F1 and EM scores of our project.

Contact

Kevin Yang
Stanford University
kevvyang@stanford.edu

Aryan Chaudhary
Stanford University
achaud@stanford.edu

References

1. R-NET: Machine Reading Comprehension with Self-Matching Networks. Natural Language Computing Group, Microsoft Research Asia, May 2017, https://www.microsoft.com/en-us/research/wp-content/uploads/2017/05/r-net.pdf.
2. https://pytorch.org/docs/stable/generated/torch.argsa.html
3. Bansal, Aakash, et al. "A Neural Question Answering System for Basic Questions about Subroutines." 2021 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER), 2021, https://doi.org/10.1109/saner50967.2021.00015.
4. Robillard, Martin P., et al. "On-Demand Developer Documentation." 2017 IEEE International Conference on Software Maintenance and Evolution (ICSME), 2017, https://doi.org/10.1109/icsme.2017.17.
5.
6.
7.
8.
9.
10.