Replicating Advances in Question-Answering with Deep Learning and Complex Attention

Stuart Cornuelle
CS224n | Winter 2018
Stanford University
stuartcc@stanford.edu

Abstract

Question answering is a canonical test of machine reading comprehension. The availability of large labeled datasets has now opened the way for end-to-end training of machine learning models in this space; successful approaches apply novel combinations of recurrent neural networks, deep learning and mechanisms of attention. In this project we seek to replicate certain of these modern approaches to building a powerful question-answering model, starting from a functional baseline and taking special guidance from the work of Seo et al. in their 2017 paper, *Bi-Directional Attention Flow for Machine Comprehension*. [1]

1 Introduction + Background

Extractive question answering is a machine reading comprehension task. It requires a system to identify, though not generate, the minimal best answer to a given query. The answer must be a contiguous span of text contained within a context paragraph against which the query is posed. Question answering of this sort is a cornerstone in information retrieval systems, with applications ranging from smart virtual assistants and dialog agents to the auto-summarization and synthesis of source documents.

The question-answering problem takes as input a question and a context paragraph. It outputs an uninterrupted span within the context that answers the question posed. The answer should be free of extraneous information; correct answers are both accurate and minimal. Accuracy is assessed through two metrics of interest, described below.

Recurrent neural networks in general, and attention mechanisms in particular, have been successful in capturing more of the nuanced relationship between query and context in machine comprehension. In this project we adopt techniques from one such success, the Bi-Directional Attention Flow (BiDAF) network.

2 Approach

Our approach is one of iterative improvement to a provided neural baseline model, which we here describe. The baseline is a three-stage system consisting of encoder, attention and output layers; input features are pre-trained 50- to 300-dimensional GloVe word vector representations of both the context and question text.

The encoder layer consists of a single-layer bidirectional RNN outputting past- and future-conditioned representations of the context and question. Forward and backward hidden states are concatenated. Vanishing gradients are managed through the use of gated recurrent unit (GRU) cells. The encoder produces hidden states fed next through the attention layer.

Attention in the baseline is unidirectional dot product attention; context hidden states attend to question hidden states, yielding an attention output that captures which elements of the query are most relevant to each token of the context. The output layer, finally, applies a rectified linear unit (ReLU) activation following a linear transformation of the blended representations produced by the attention layer. Answer spans are constructed by maximizing greedily over the distributions produced by two additional softmax layers that follow the output layer.

Our approach includes three significant architectural updates to the baseline, followed by extensive hyperparameter tuning.

We begin by substituting a Bidirectional Attention Flow (BiDAF) layer for the baselines dot product attention. [2] Whereas the baseline applies attention in one direction only, computing an attention distribution over the question tokens for each context word, the BiDAF model introduces an additional mechanism by which elementwise similarity between context and query words is assessed to measure contextual relevance to specific regions of the query. The result is a second attention distribution, this over context hidden states, that is combined with the first for a more sophisticated sequence of blended representations for processing in the output layer.

In the parlance of Seo et al.s work on this model, we say specifically that context-to-question (C2Q) attention outputs are concatenated with their question-to-context (Q2C) counterparts. Each constituent of the input thereby attends to the other, allowing the BiDAF attention layer to more robustly model relationships among the question and context than can the simple baseline.

Our second architectural improvement is also inherited from the BiDAF network. Between attention and output layers we insert a modeling layer consisting of an RNN that, as Seo et al. explain, learns the interaction within the query-aware context representation (the output of the attention layer). The authors cite as motivation a useful division of learning by the addition of the modeling layer, in their case a two-layer Long Short-Term Memory (LSTM) network. We implement a simpler single-layer version that nonetheless displays notable performance improvements on dev and test sets.

Third, we address the naivete of the baseline models prediction mechanism. That mechanism selects start and end indices of the predicted answer span by independently maximizing over distributions output by two softmax layers. This approach, while simple, makes no use of available knowledge about the distribution of ground truth answer lengths, which should bias the model against returning extreme spans. The baseline is also free to predict spans of nonsensically negative length. Both problems are rectified by instead predicting spans via joint probability maximization over the softmax distributions, where start and end indices are constrained based on the known composition of test set answers. This approach is taken from Chen et al. [2]

We experiment lastly with iterative tuning of hyperparameters, namely GloVe embedding and hidden state dimensionalities, context and question length, learning rate, and dropout and L2 norm regularization parameters. We explore alternative methods of combining encoder outputs and of gated recurrent unit cells (GRU and LSTM) in the modeling layer.

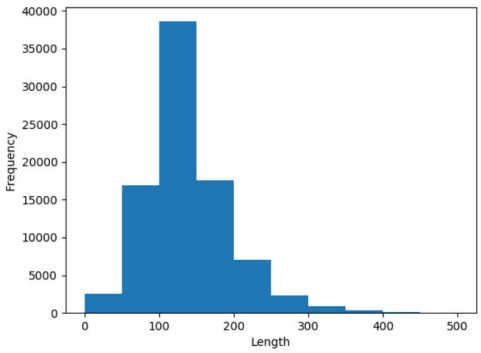
3 Experiments

3.1 Dataset

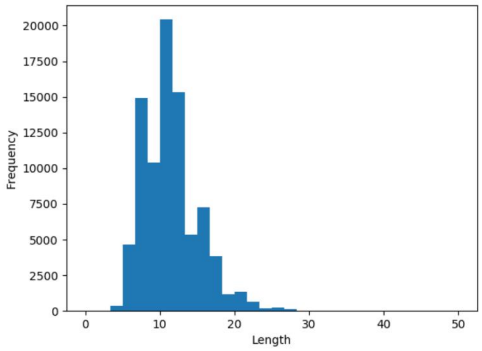
Our data source is the Stanford Question Answering Dataset (SQuAD), a collection of over 107,000 question-context-answer triples designed for the supervised learning of machine reading comprehension systems. [3] Questions are posed and answered by human readers; contexts are extracted from over 500 Wikipedia entries, ensuring broad topical and lexical coverage in the dataset.

3.2 Evaluation

Evaluation is twofold. We measure both F1 and Exact Match (EM) scores for each example, implicitly seeking to maximize each by minimizing the summed cross entropy loss over start and end index predictions. The F1 metric is



(a) Context lengths in training set.

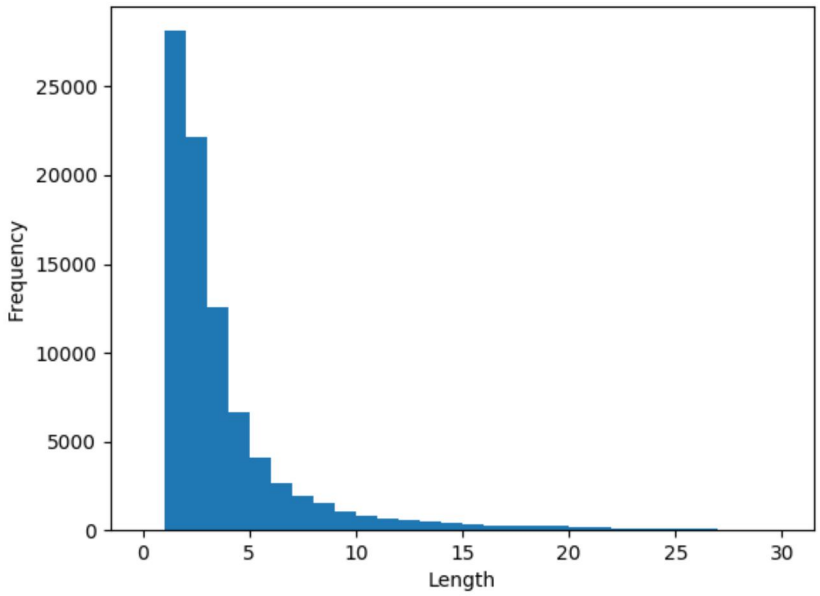


(b) Question lengths in training set.

a combination by harmonic mean of precision (number of predicted answer words actually in the answer span) and recall (number of words in the answer span correctly predicted). The EM score is a binary measure reflecting whether the systems predicted answer span perfectly matches that of the human reader.

3.3 Model Configurations

Our first step is to explore the dataset by visualizing context, question and answer length distributions in the training data via histogram. We further visualize the distribution of start indices in the datas answer spans. This step allows for an immediate reduction in `context_len` and `question_len` hyperparameters without fear of meaningful degradations in accuracy. Combined with a modest reduction in minibatch size, this step prevents memory overflows during the high-dimensional tensor manipulations required in parts of the BiDAF implementation.



Answer lengths in training set.

We next inspect baseline model outputs over a random selection of training examples to look for qualitative opportunities for improvement. We see instances of run-on or null predicted answer spans, motivating an update to the index prediction scheme post-output layer. We see at least one apparent example of insufficient attention paid to a query word, with a material impact on the resulting prediction, which motivates a more complex attention layer. This we next undertake to construct.

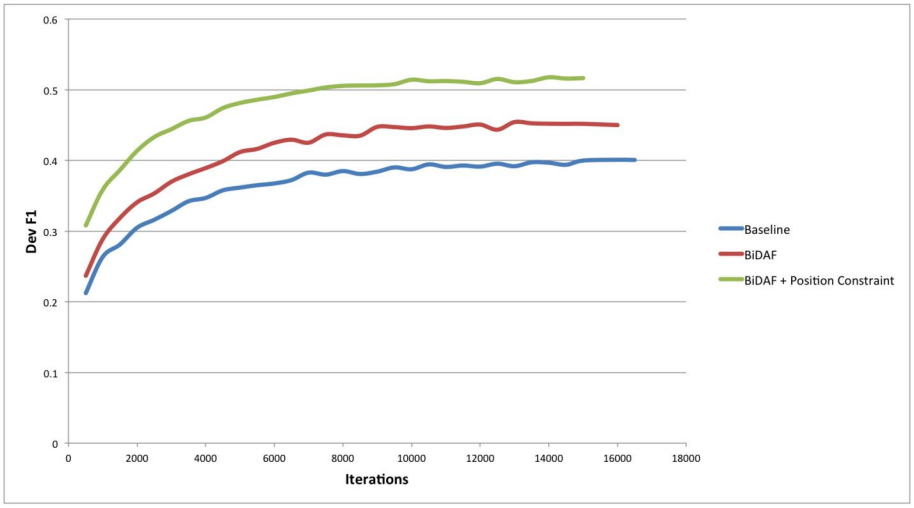
```
CONTEXT: (green text is true answer, magenta background is predicted start, red background is predicted end, _underscores_ are unknown
tokens). Length: 39
southern california is home to many major business districts . central business districts ( cbd ) include downtown los angeles ,
downtown san diego , downtown san bernardino , downtown bakersfield , south coast metro and downtown riverside .
          QUESTION: what is the only district in the cbd to not have " downtown " in it 's name ?
       TRUE ANSWER: south coast metro
    PREDICTED ANSWER:
     F1 SCORE ANSWER: 0.000
          EM SCORE: False
```

Naive span prediction wherein end index precedes start index.

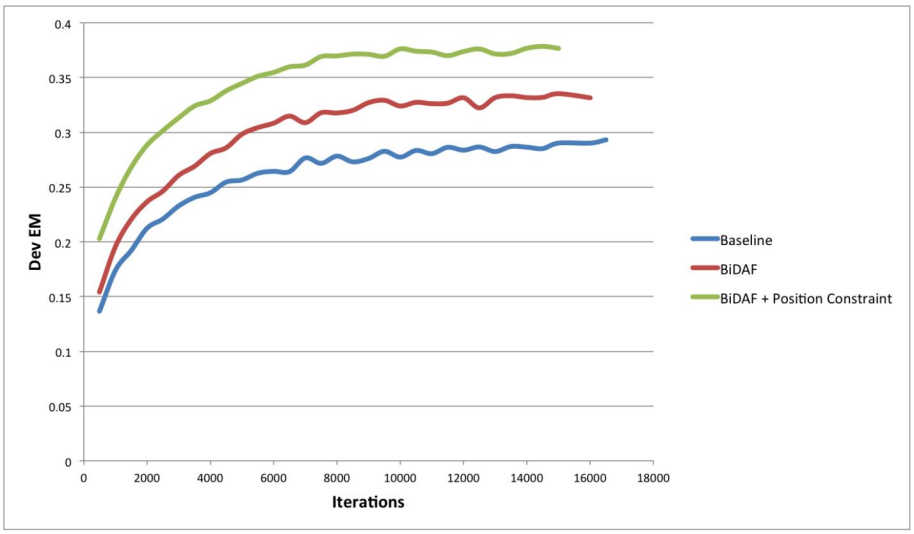
```
CONTEXT: (green text is true answer, magenta background is predicted start, red background is predicted end, _underscores_ are unknown
tokens). Length: 39
southern california is home to many major business districts . central business districts ( cbd ) include downtown los angeles ,
downtown san diego , downtown san bernardino , downtown bakersfield , south coast metro and downtown riverside .
          QUESTION: what is the only district in the cbd to not have " downtown " in it 's name ?
       TRUE ANSWER: south coast metro
    PREDICTED ANSWER:
     F1 SCORE ANSWER: 0.000
          EM SCORE: False
```

Query word “not” ignored pre-BiDAF implementation.

Once the BiDAF attention mechanism is functional, we measure performance against the baseline and find a 14.3% improvement in EM and 13.3% increase in F1 score on the dev set over roughly 15,000 training iterations, at which point improvements level off. We pair this updated attention with the smarter span index selection protocol from [2], yielding an additional 12.8% increase in EM (29.1% in aggregate over the baseline) and 14.0% improvement in F1 (also 29.1% higher than the baseline).

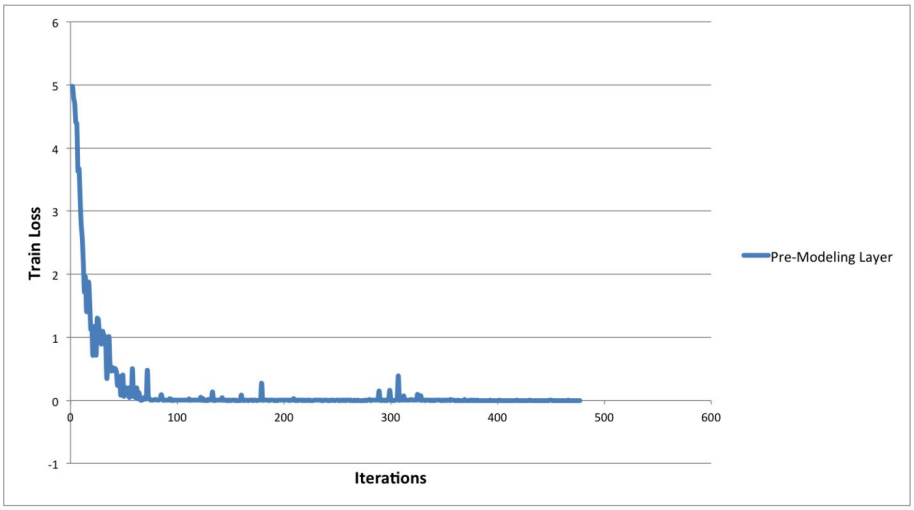


Dev F1 following first two model improvements.



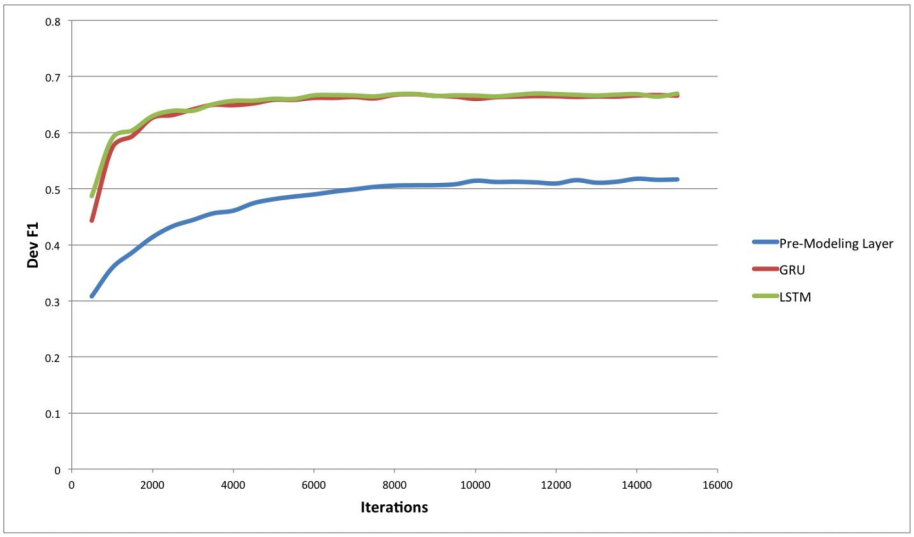
Dev EM following first two model improvements.

Before proceeding to the addition of the modeling layer, we construct a small, randomized dataset from training examples to ensure the current model successfully overfits as expected. Training loss indeed collapses toward zero within 100 iterations on this small dataset.

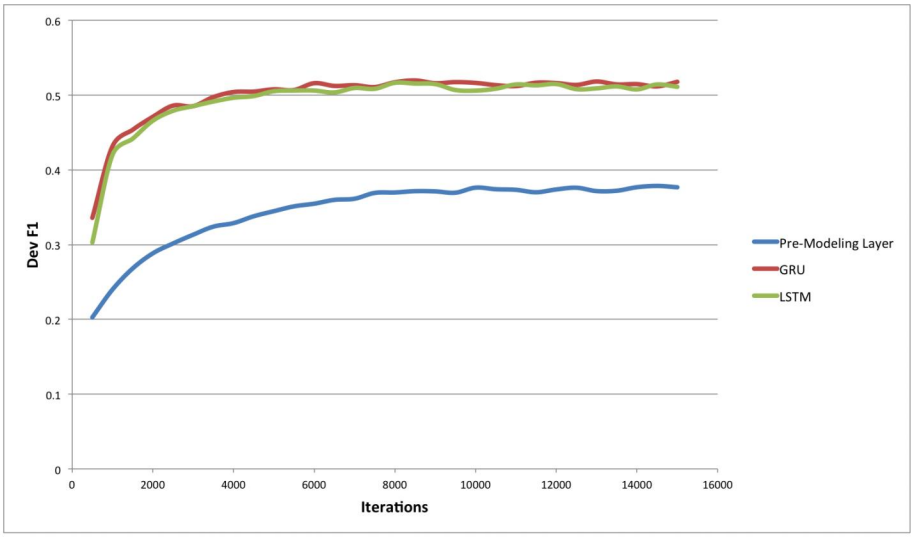


Training loss on reduced-size dataset.

We proceed to implement a bidirectional RNN modeling layer with GRU cells stacked atop the BiDAF attention module. (We find that LSTM cells perform comparably and elect to maintain the GRU implementation in further experiments.) The result is a further improvement in dev F1 from 0.518 to 0.670 and in dev EM from 0.379 to 0.520. We also review outputs to scan for qualitative changes and note that, in the example identified earlier, the model no longer ignores the relevant query word not as a qualifier of which context words are appropriate to include in the answer. While perhaps spurious this seems to represent an explicit change due to the BiDAF attention layers introduction of, as Seo et al. report, query-aware context representations.

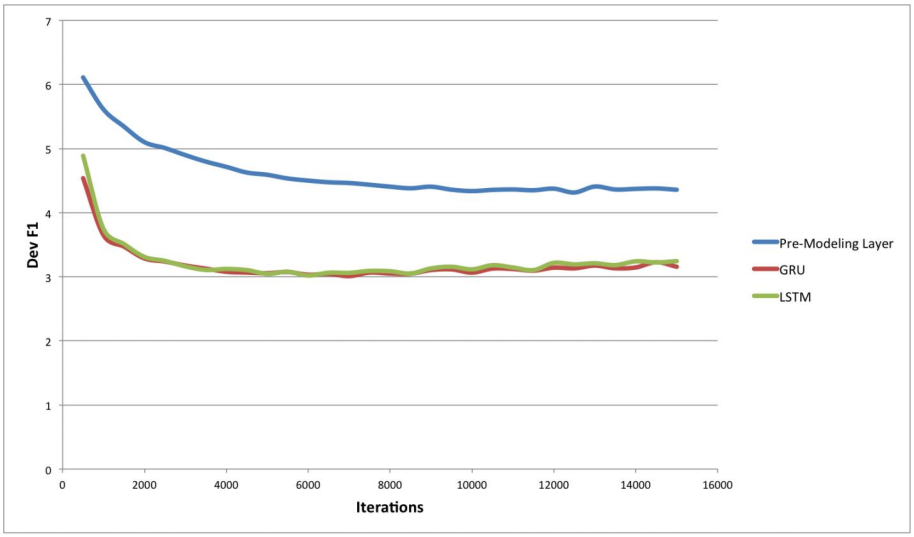


Dev F1 following addition of modeling layer.

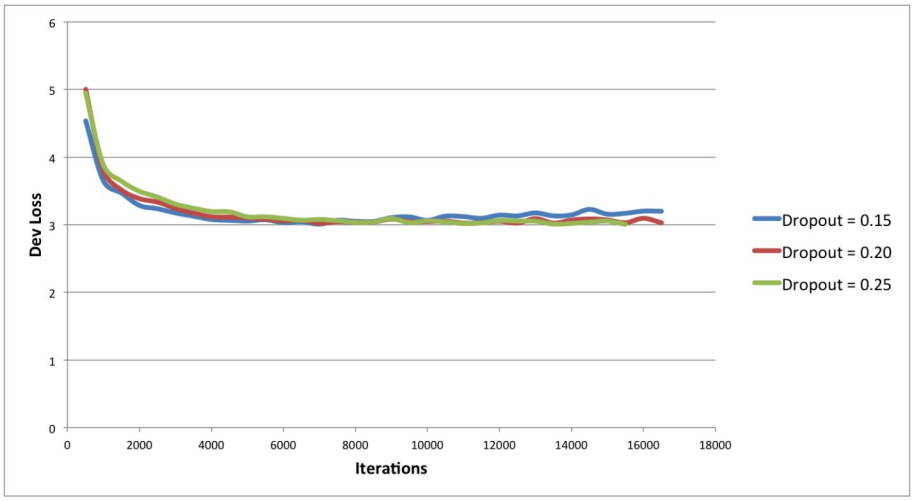


Dev EM following addition of modeling layer.

We note at this point not only an increase in batch training time but some visible overfitting to the training set; dev loss begins a slow but sustained increase after approximately 6,000 training iterations. We experiment with penalizing parameter norms using L2 regularization and with dropout, in both cases over a range of hyperparameter values. We find that a modest increase in dropout rate to 0.20 from its baseline default of 0.15 best remedies overfitting without compromised accuracy.



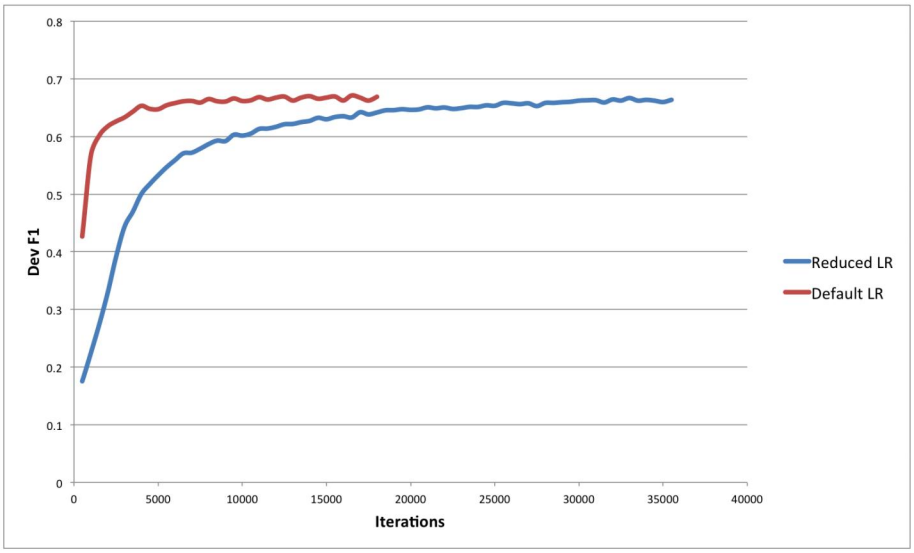
Dev loss following addition of modeling layer; note visible evidence of overfitting.



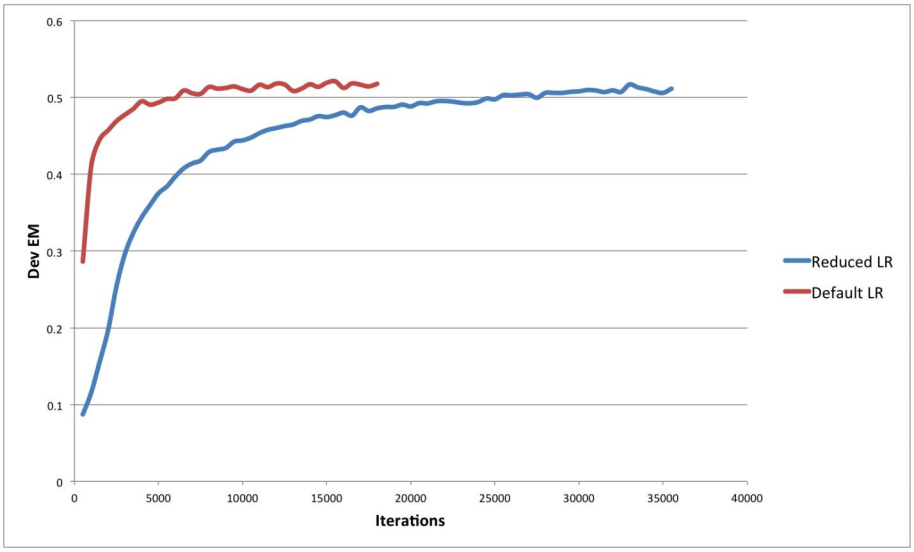
Tuning of dropout hyperparameter to counter modeling layer overfitting.

At this point we pursue several further attempts at tuning, each performed one at a time, while visualizing the resulting model runs in TensorBoard. We explore averaging and adding context and question hidden states, rather than concatenating them, following the encoder layer. We increase model size via two updates to the hidden state dimensionality. We increase the GloVe embedding size twice. None of these attempts improves our performance metric without amplified overfitting.

Finally, we observe the relatively sharp increase in dev F1 and dev EM (and corresponding sharp decrease in dev loss) early in the runs of every model trained to date. We hypothesize a too-coarse step size for the optimizer to converge on a better minimum, and experiment with reducing the learning rate twice by orders of magnitude. The first attempt shows a smoother learning curve but arrives at the same performance levels after roughly 35,000 iterations; the second trains too slowly to complete the experiment.



Result of reducing learning rate on dev F1; we observe slower learning but no performance improvement.



Dev EM after reducing learning rate.

4 Conclusion

Ultimately no tuning is able to improve on our results achieved from addition of the GRU modeling layer: a dev F1 score of 0.670 and dev EM score of 0.520. We see that the substitution of complex attention and the modeling layer do yield meaningful improvements but that, lacking the full suite of architecture components incorporated into Seo et al.'s work, the model remains limited. Given more time and perhaps intelligence we would pursue a number of different attention mechanisms—coattention and self-attention chiefly—in order to construct an ensemble of the various models with an averaging or heuristic best-prediction scheme; would add a second modeling layer, as in the BiDAF paper; would implement the same paper's character-level CNN feature; and would conduct a more thorough and rigorous exploration of the hyperparameter space in the interest of full optimization.

5 References

[1] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi & Hannaneh Hajishirzi (2016) *Bi-Directional Attention Flow for Machine Comprehension*. arXiv preprint arXiv:1611.01603.

[2] Danqi Chen, Adam Fisch, Jason Weston & Antoine Bordes (2017) *Reading Wikipedia to Answer Open-Domain Questions*. arXiv preprint arXiv:1704.00051.

[3] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev & Percy Liang (2016) *Squad: 100,000+ Questions for Machine Comprehension of Text.* CoRR, abs/1606.05250.