Accelerated and Accurate Question Answering

CS 224N Final Project | Winter 2019 Kuangcong Liu , Haoshen Hong , Xiao Wang {cecilia4, haoshen, xiao1105}@stanfrod.edu
Computer Science Department, Electrical Engineering Department

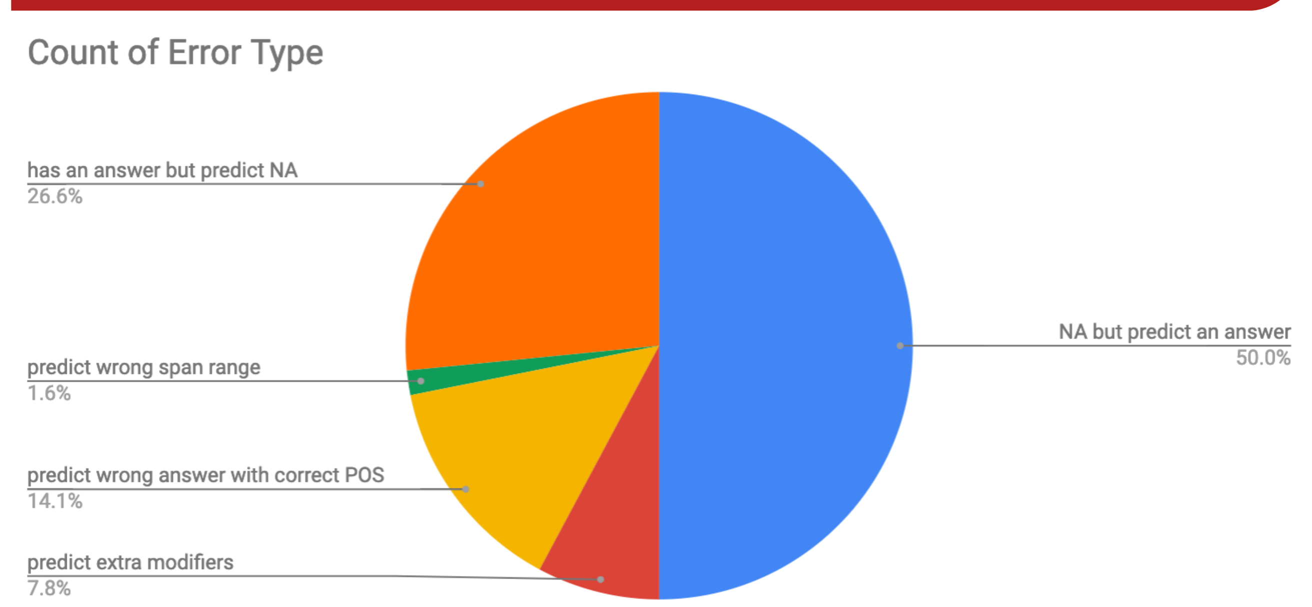
Problem

- ❖ The task of question answering on SQuAD 2.0 reading comprehension dataset has led many significant breakthrough models in building the machine comprehension system, important models like pre-trained contextual embeddings (PCE) model BERT and non-PCE model BiDAF.
- ❖ Limitation of PCE models like BERT:
  - ➢ need pretraining weights on a large-scale language modeling task
  - ➢ expensive to finetune due to the calculation resources we have.
- ❖ Non-PCE approaches are likely to underperform the PCE models by a large margin.
- ❖ Our project will focus on developing non-PCE model based on BiDAF and measure how well our machine comprehension system could understand the context.

Dataset

We will use SQuAD 2.0 dataset for this work, which comprises 129,941 training examples, 6078 dev examples and 5921 test examples, where each example is presented as context - question pairs.
The target is whether the question is answerable from the given context, and if so, we need to predict the answer span from the given context.

Analysis

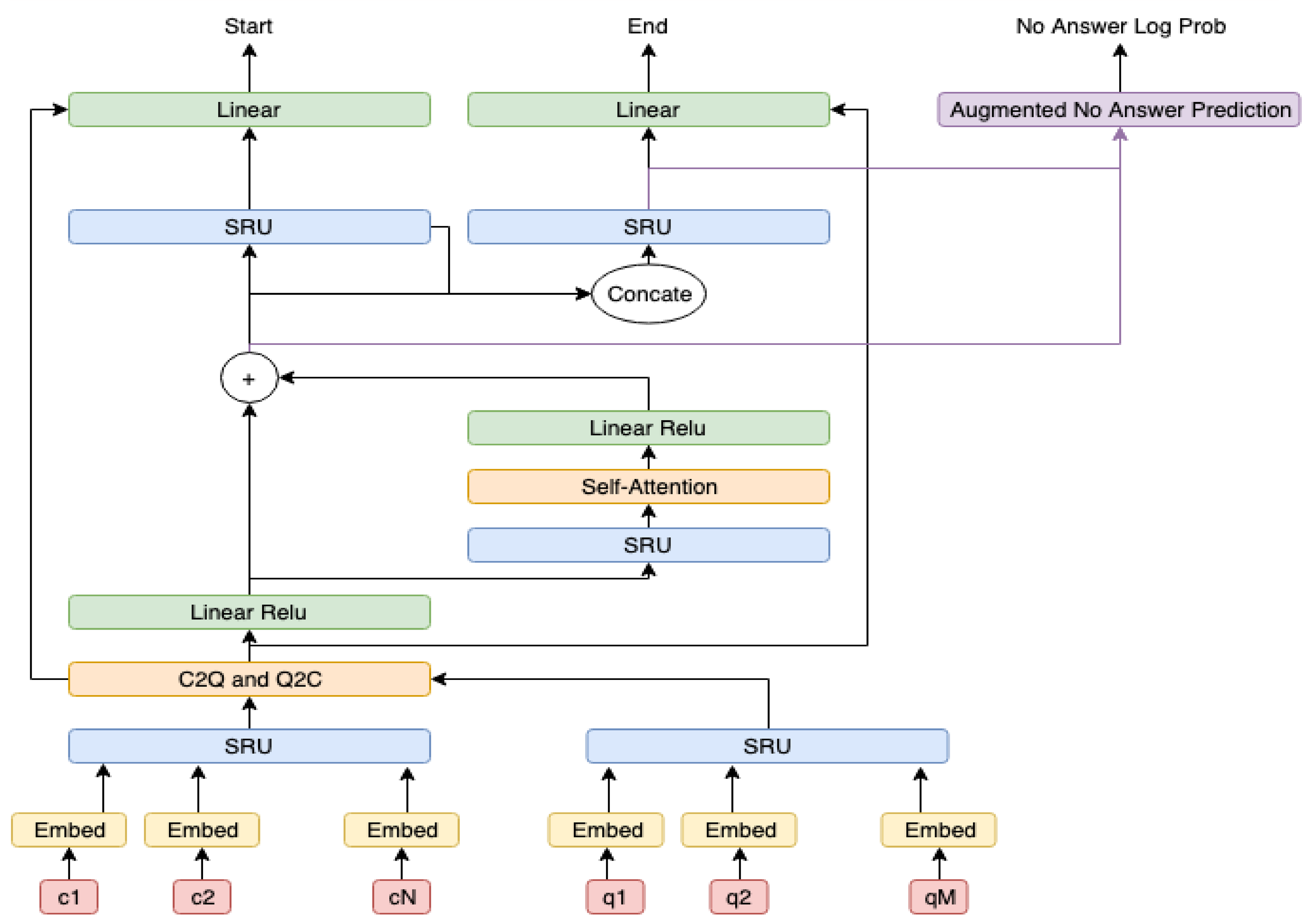


Examples:

Error type	Ratio(%)	Example
has an answer but predict NA	26.6	**Context:** They even lent their ethnicity to the name of their castle: Afranji, meaning "Franks." **Question:** What was the name of the Norman castle? **Prediction:** [] **Answer:** ['Afranji', 'Afranji', 'Afranji']

- ❖ Randomly select 64 incorrect questions (based on EM) and categorize them into 5 categories
- ❖ NA prediction error comprises 76.6% of total errors, would focus on reducing NA errors.

Approach



1. Char-CNN:
- utilize subword information and effective in modeling OOV (out-of-vocabulary) words.
- Combine with word level embedding from Glove.

2. BiDAF with Self-Attention:
- look at other positions in the input sequence
- find other words with strong relation of the current word

3. Prediction Layer:
- apply Bi-RNN and concatenate the output of RNN to attention output for end index calculation.

4. SRU:
- preserves the performance of RNN-like structure
- enable the computation to run in parallel to promote the training efficiency

5. Regularization:
- Embedding Dropout: perform dropout on the embedding matrix at a word level.
- Activation Regularization(AR): penalize activations that are significantly larger than 0.
- Temporal Activation Regularization(TAR): penalize the model from producing large changes in the hidden state from time t to time t+1.

$$AR = \alpha L_2(m \odot h_t)$$
$$TAR = \beta L_2(h_t - h_{t+1})$$

6. Augmented No-answer Prediction:
- output the probability of whether this question can be answered

Results

1. Most Improvements: Self Attention + SRU + Char CNN
- SRU could accelerate our training process by 2x and improve EM by 1.8, F1 by 2.5.
- Char CNN concat with word embedding has significant impact in EM and F1.

Model	EM	F1	Time(Hours)
BiDAF	57.64	60.80	12.98
BiDAF + Self Attention	59.20	62.45	13.63
BiDAF + Self Attention + SRU	60.04	63.66	6.08
BiDAF + Self Attention + SRU + Char CNN	65.20	68.48	12.80

2. Other Parameters:
- Several structures of the augmented no-answer inputs by trying some combinations of attention results, self attention results and start/end raw representation and logit

Model	EM	F1
Base + AR (alpha = 0.1) + TAR (beta = 0.05)	61.12	64.40
Base+ NA Pred[Att, End rep]	60.09	63.70
Base+ NA Pred[Att, Start rep, End rep, Max logits]	59.49	62.24
Base+ NA Pred[Att, Start rep, End rep]	60.12	62.96

3. Fine-tuning:
- We perform an extensive parameter tuning on tunable parameters
- Below graph shows our final training graph compared with initial training
- Blue - final parameters, Red - initial parameters