### Problem

The problem we are trying to solve is to improve the F1 and EM scores for a QA system for SQuAD. Challenges with the baseline BDEAF model include lack of data augmentation, as well as simplicity of embedding layer and coattention layer. Our work is meaningful in that it explores three new areas of improvements and improves the baseline by 11.6% in accuracy.

### Background

QA systems have two levels of significance. From a research perspective, it serves as a measure for how well systems can "understand" text. From a practical perspective, they are useful for better understanding any piece of text. To improve the QA system for SQuAD, we start from a baseline Bi-directional Attention Flow (BDEAF) model [4]. Since the baseline model only includes word level embedding, the first approach we used is adding character level embedding. Then, we explored coattention [7], self-attention [6], span corruption [3] and back-translation [2].

### Methods

We explored three main areas of improvements: data augmentation, embedding layer, and attention layer. This is an overview of our structure:

- **Data Augmentation**
  - We use two techniques for data augmentation.
    - **Span Corruption**: Each example has a 30% chance of having part of the context replaced by a span of OOV tokens [3]. If part of the answer was corrupted, then the example is marked as one with no answer. Otherwise, the original answer is kept.
    - **Back Translation**: For each example, we first attempted translating both the query and context into German and then back into English. Then we updated the answer [2]. Due to compute limitations, we back translated only the query on 25% of the data.

- **Embedding Layer**
  - We added a character-level embedding layer to the BDEAF model. This layer maps each word to a vector space using character-level Convolutional Neural Networks (CNNs). The vectors are 1D inputs to the CNN. The CNN outputs are max-pooled over the entire width to obtain a fixed-size vector for each word [4].

- **Attention Layer**
  - We implemented both co-attention and self-attention.
    - **Coattention** refers to attending over representations that are themselves attention outputs. We implement the Coattention Layer following Xiong (2015)'s approach for Dynamic Coattention Networks [7].
    - **Self-attention** measures the hidden state $h_t$ to attend to all the previous hidden states so far $h_1, \ldots, h_{t-1}$. We implement the Self-Matching Attention Layer from R-Net [6].

### Experiments

- **Data**: We use the official SQuAD 2.0 dataset with three splits: train, dev, and test as described in the default SQuAD handout. The three splits in detail:
  - *train (120,941 examples)*: All taken from the official SQuAD 2.0 training set.
  - *dev (6,786 examples)*: Roughly half of the official dev set, randomly selected.
  - *test (1,155 examples)*: Remaining examples from official dev set, plus held-out examples.

- **Evaluation Method**: We will evaluate based on two key metrics:
  - F1: calculates the harmonic mean of precision and recall. Harmonic mean: the reciprocal of the arithmetic mean of the reciprocals of the pairs set of observations.
  - Exact Match: a binary measure. It is equal to 1 if exactly matches the ground truth.

- **Evaluation Model**: The model is evaluated using the dev set.

- **Experimental Details**: For the baseline model, we follow the provided configurations, where:
  - Setting B: 0:1
  - Training: 30 epochs, or about 3 hrs.

- **Results**:
  - We note that all models perform well on "Who" questions.
  - Coattention is particularly bad at "when" and "what" questions but stands out on all the when questions.

### Analysis

We categorized the questions to analyze the model's performance in different categories:

- **Models performed best on** When questions, which is intuitive based on the simplicity of date finding.
- **Attention-based models exhibited more variable performance across question categories.**
- **Coattention was poor at questions where great at when questions, while embeddings based models gave a boost across the board.**

### Conclusions

- **Character embeddings**, implemented alone, produce the greatest improvement in baseline.
- **Improvement methods, when combined, did not always lead to an improvement.**
- **Character embedding did not train well with data augmentation or our two implementation designs of attention, coattention or self-matching attention.** We did, however, see a 3-point lift in F1 using self-matching attention alone.
- **Model performance varies across different question categories.**
  - All models performed the best on "Who" questions.
  - Coattention is particularly bad at "when" and "what" questions but stands out on all the when questions.

### References (Part)

[3] Cai Li, Huiyu Nan, Reuter, Alan Wexler, Katherine Lee, Shawn Narang, Michael Mathe, Yang Zou, Wei Li, and...