Semi-Supervised Question Answering

Niki Agrawal, nikhar@stanford.edu **Mayuka Sarukkai,** mayuka@stanford.edu

Abstract

Creating high-performing models in the context of "low-resource" domains with limited labelled training data is a particularly salient challenge for today's question-answering systems. We believe that creating and evaluating question-answering models that perform effectively on largely unlabelled training datasets is an important contribution to the robustness of SQuAD solutions. Given an input consisting of a passage, and a query about the passage, the goal of this model is to perform the question-answering task of returning the correct answer to the question by selecting the span of the passage correctly answering the question. The proposed model improves upon the Bidirectional Attention Flow model by 1) implementing adjustments such as advanced embedding (character-level and part-of-speech embedding) to the BiDAF model to improve overall performance, and 2) cloze fill-in-the-blank question-answer generation to pre-train BiDAF model and thus boost performance in a semi-supervised setting with limited labelled training data.

Our primary findings are the following: 1) Character-level and part-of-speech embeddings improve question-answering performance on our model, for both large and small training sets. (F1 = 52.04, EM = 48.34 for 25% split of SQuAD) 2) Pre-training our model on cloze question-answer pairs before fine-tuning improves performance on 25% of SQuAD training data (F1 = 52.66, EM = 48.35), but worsens performance on 10% of SQuAD training data.

1 Introduction

With the Internet making vast quantities of text available at our fingertips, Question Answering systems have become critical for us to better understand and quickly extract key information from text.

The creation of the Stanford Question Answering Dataset (SQuAD) [10]– a large corpus of Wikipedia articles annotated by crowdsourced workers – has motivated many research efforts to build advanced reading comprehension systems. In many domains, however, gathering a large labeled training dataset would not be feasible due to limits on time and resources, leading to our exploration of semi-supervised question answering. Our goal is to create and evaluate question-answering models that perform effectively on largely unlabelled training datasets, which we believe is an important contribution to the robustness of SQuAD solutions. Ultimately, we envision the creation of a reading comprehension system that can answer questions based on any piece of text, from a children's story to an advanced neuroscience textbook. In order to achieve this vision, we must address the challenges of creating question-answering systems in low-resource domains.

Existing research in the Question-Answering space explores a variety of models for building QA systems, from Bidirectional Attention Flow (BiDAF) to ELMO and BERT. These efforts primarily focus on building models that perform effectively given the entire SQuAD training corpus. In this project, however, we present a method for semi-supervised question answering [4] that can be applied to contexts where limited question-answer pairs are available. This model requires 1) a base, unlabeled document, and 2) a small set of labeled QA pairs. First, we automatically generate cloze (fill-in-the-blank) style question answer pairs from the unlabeled data set. Then, we pre-train a

BiDAF (Bidirectional Attention Flow) QA model with complex embeddings on the automatically generated dataset. Finally, the pretrained model is fine-tuned on a small labeled corpus.

We were inspired to pursue this approach by research in which Dhingra, et. al generate cloze question-answer pairs, use these to train a BiDAF + Self Attention model, and fine-tune on a small labeled corpus [4]. They achieved promising results, despite the format of fill-in-the-blank questions differing significantly from natural-language, human-style questions. Additionally, the time-efficiency of their question-answer generation method (as opposed to a neural network method) makes it a compelling solution for low-resource domains.

2 Related Work

Several studies have explored methods for semi-supervised or low-resource comprehension learning. Most approaches involve generative models for generating questions from an unlabeled text. Yang et al. employ a generative domain-adaptive net (GDAN) to generate questions from the unlabeled text, and then adapt these questions using a reinforcement learning method for domain adaptation to improve quality of generated questions [12]. In contrast to this text-based approach, Oh et al. use a language model to generate questions from previously compiled causality relations, and then match questions with paraphrases from the unlabeled text to identify close paraphrases and generate question-answer pairs[9]. Dhingra et al. find that a non-neural model for generating fill-in-the-blank (or cloze) style questions by finding phrase matches across multiple locations in an unlabelled document is highly effective in boosting performance in low-resource settings; this strategy receives a significant boost in accuracy after fine-tuning on a small subset of labelled SQuAD data [4].

Cloze questions (CQ) have been widely investigated particularly for their widespread applications in educational settings. Strategies for automated cloze generation include identification of important sentences followed by selection of key phrases [8], and simplification and pronoun resolution of sample sentences preceding answer demaracation [6]. An additional concern for CQ generation in the educational setting is the creation of "distractors," or relevant alternate answers, for a given CQ [7] – however, within the context of this question-answering project, we believe a non-neural approach to cloze generation can be equally effective. Thus, this paper employs a method similar to Dhingra's of generating cloze questions through exact phrase matching between question and context, with fine-grained phrase tagging to identify key phrases and concepts [4].

In addition to optimizing the semi-supervised context through automated generation of additional training data, this work also considers simple improvements to the baseline BiDAF model that may boost performance in low-resource settings. In particular, character-level embeddings [11] and part-of-speech embeddings [3] may boost performance across the board.

3 Approach

The proposed system of semi-supervised question-answering employs three stages: 1) unsupervised generation of cloze-style questions (in fill-in-the-blank style) from the target domain, 2) pre-training of a custom Bi-Directional Attention Flow (BiDAF) model using cloze dataset, and 3) fine-tuning of model on a small set of supervised QA pairs from an available labeled dataset. We also make improvements to the baseline supervised BiDAF model through character-level and part-of-speech embeddings.

3.1 Baseline

As a baseline model, we assess official SQuAD F1 and EM scores for the provided model, which implements a state-of-the-art BiDAF model [11] with the exclusion of the character-level embedding. In order to establish a baseline for the semi-supervised model, we also assess F1 and EM scores for 10% and 25% fractional subsets of the train data.

3.2 Character-level and Part-of-Speech Embeddings

In order to add another level of granularity to the hierarchical BiDAF model, the character embedding layer maps the character-level indices corresponding to each question and context word set onto an embedding vector space. Character-level embeddings are passed through a Convolutional Neural Network (CNN), and the resulting embeddings are then concatenated with word-level embeddings to form a longer word embedding [11]. Part-of-speech embeddings were created by mapping question and context words to parts of speech, indexing parts of speech, and mapping indices to an embedding vector space. The resulting embedding vector is also concatenated with word- and character- level embeddings. Finally, the joint embedding is passed through a dropout layer, linear projection layer, and Highway network (Srivatsava et al., 2015).

The final model consists of six layers. (1) The Character Embedding Layer applies a CNN to every character vector to produce a character-level embedding – embeddings for each word are max-pooled to yield a fixed-size vector for each word. (2) Words are then passed through a pre-trained word embedding model and mapped to a vector space. Part of Speech embeddings are also formed for each word, and word-, character-, and POS- embeddings are concatenated. (3) The contextual embedding layer uses a bi-directional LSTM to model temporal interactions between words. (4) The attention layer takes as inputs the contextual vectors for query and answer, and outputs query-aware feature vectors for each word using a similarity matrix between context and query word, along with the inputted contextual embedding vector. Attentions are computed bidirectionally (query-to-answer, and answer-to-query). (5) The Modeling layer employs another bidirectional LSTM to output context of each word with respect to its query and answer context. (6) Output layer applies softmax to produce the probability distribution for start index and end index respectively across the entire input paragraph.

3.3 Simple Cloze Question Generation

A cloze question is a "fill-in-the-blank" style sentence containing a span of missing words. The answer to the question is the correct word/phrase that fills in the blank.

Using an unlabeled document, we create pairs of introduction sentence (q_i) and passage sentence (p_i) such that (q_i, p_i) is a question-context pair. For a given article question, question sentences (q1… qj) constituted the first 20% of the sentences in the article [4]; successive paragraphs were accordingly labelled as context passages (p1,... pn). The reasoning behind this split is the presumption that Wikipedia articles (which serve as the base document in this context) contain introduction sentences that are likely to make reference to elaborate content later in the article.

In order to create appropriate pairings, the algorithm finds the best match sequences between q_i and p_i, by identifying an exact-match sequence of words between introduction and passage text. If a matching sequence satisfies certain grammatical constraints (i.e., the sequence is a noun phrase, named entity, or verb phrase (consisting of the part of speech regex structure of "<VERB>*<ADV>*<PART>*<VERB>+<PART>*<NOUN>*" to account for adverb-verb-object phrases)), it constitutes the answer A. The final set of cloze tuples are of the form (P, Q, A), where P is a passage, Q is a question, and A is the answer consisting of the text of the final answer, and the index of the passage where the answer begins.

Below is an example of a tuple consisting of an answer, query, and context passage (A, Q, P) generated from a Wikipedia dataset using our simple cloze-generation algorithm:

A: Stunt casting

Q: ___ is a term in casting that refers to the use of a gimmick or publicity stunt to fill a role in a television series or film.

P: Stunt casting is used to generate media attention. It may also be employed in order to garner studio support or financing for a project; for example, according to DVD featurette commentary, the 1978 version of Superman (1978 film) received studio support only after the producers were able to enlist A-list actors Marlon Brando and Gene Hackman to appear.

4 Experiments

4.1 Data

We use the SQUAD 2.0 dataset as our primary dataset, splitting into training, validation, and testing data. In order to simulate low-resource domains, we randomly selected 25% and 10% of the training data on which to evaluate our models.

We use a random sub-sample of the Wikipedia library consisting of 5500 articles to generate 748 cloze pairs as our pretraining dataset for the semi-supervised model [2, 1]. Cloze pairs were generated by parsing and stripping a sampling of the Wikipedia dumps dataset, and using the methods described above to record all possible question-answer pairs.

4.2 Evaluation

We report F1 and EM scores after 1) training on the full training set 2) training on 10% and 25% of the training set 3) pre-training on cloze pairs and then training on 10% and 25% of the training set. Our goal was to surpass the baseline model's performance on a limited labelled corpus through the addition of both complex embeddings to the model itself, as well as pretraining on a cloze dataset.

4.2.1 Experimental Methods

Training over the full SQuAD 2.0 dataset occurred over approximately 2.45M steps, with a constant learning rate of 0.5. All models ran for full 30 epochs. Word embeddings are of size 300, character embeddings are of size 64, and part-of-speech embeddings are of size 20. A dropout probability of 0.2 is applied to all dropout layers.

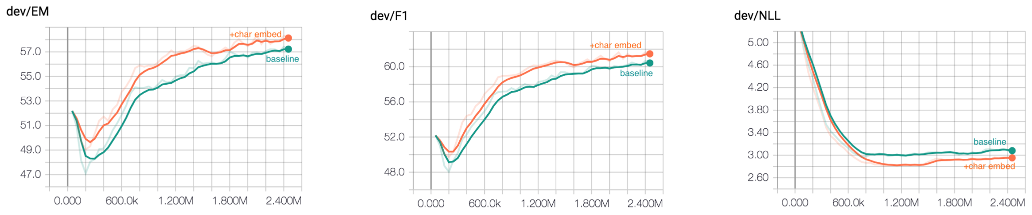
Training on the baseline model takes about 5 hours with 100% data, three-four hours with 25% data, and two-three hours with 10% data. Char embeddings runs for about 9 hours on 100% data, but takes similar time as baseline for 25% and 10% data. Adding POS embeddings and pre-training on the cloze sample make negligible changes to training times.

5 Results

5.1 Model Improvements: Embeddings

	F1	EM
Baseline (100%)	60.44	57.22
Baseline + Char Embeddings (100%)	61.48	58.13

Character-level embeddings improved performance by approximately one percentage point in both F1 and EM for the full train split, and approximately two percentage points for the 25% train split. Figure 1 illustrates improvements on full training dataset.



(a) Dev EM score

(b) Dev F1 score

(c) Dev Loss

Figure 1: Improvement upon the baseline for SQuAD, full train split

The improvement of performance with character embeddings was also evident on a smaller training split of 25%. Part-of-speech embeddings further improved performance in this setting (Figure 2). The best model performance on the 25% split was attained with a combination of character and part-of-speech embeddings, achieving a test F1 score of **52.039** and test EM score of **48.335**.

	F1	EM
Baseline (25%)	53.07	49.67
Baseline + Char Embeddings (25%)	53.80	50.20
Baseline + Char + POS Embeddings (25%)	54.09	50.97

5.2 Pretraining on Cloze Questions

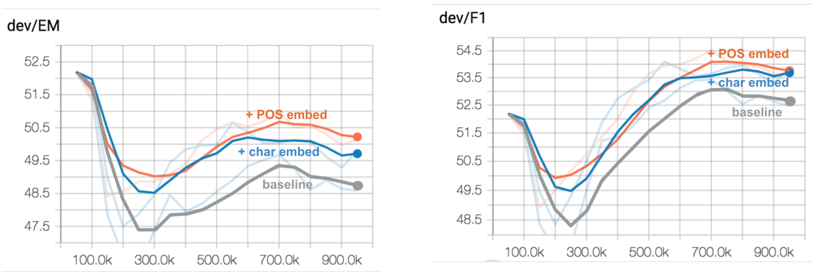
Pre-training on generated data before fine-tuning on a smaller train split provided a slight boost to dev performance, even on a small sample of pretraining data of 748 samples (Figure 3).

	F1	EM
Char Embed BiDAF (No pretraining) (25%)	54.10	50.65
Char Embed BiDAF + Cloze pretraining(25%)	54.55	50.80

However, the pretrained model performed significantly worse with only 10% of SQuAD data (Figure 4), likely because the limited scope of cloze-style questions could not adequately capture question-answer structures in the absence of a more robust set of SQuAD data.

	F1 (final value)	EM (final value)
Char Embed BiDAF (No pretraining) (10%)	47.92	45.69
Char Embed BiDAF + Cloze pretraining(10%)	46.87	43.63

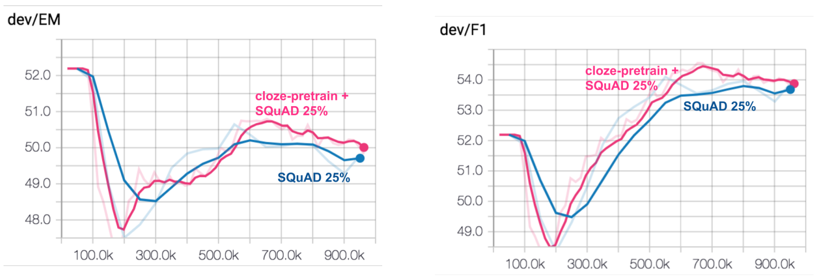
The best model for pre-training in a low-resource setting was pretraining on 748 Cloze questions followed by finetuning on the 25% split of SQuAD data, using character embeddings. This model yielded a test EM score of **48.352** and a test F1 score of **52.657**.



(a) Dev EM score

(b) Dev F1 score

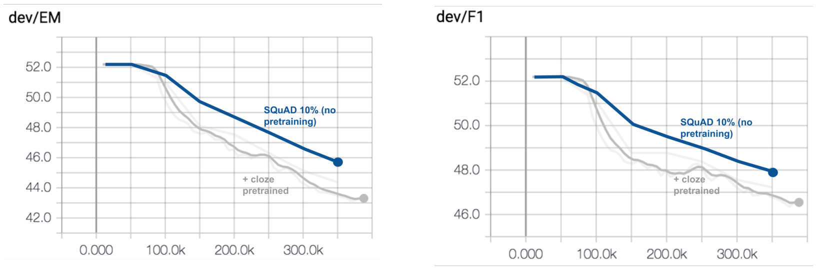
Figure 2: SQuAD 25% train split with embeddings



(a) Dev EM score

(b) Dev F1 score

Figure 3: Performance on 25% train split with Cloze pretraining + Character Embeddings



(a) Dev EM score

(b) Dev F1 score

Figure 4: Performance on 10% train split with Cloze pretraining + Character Embeddings

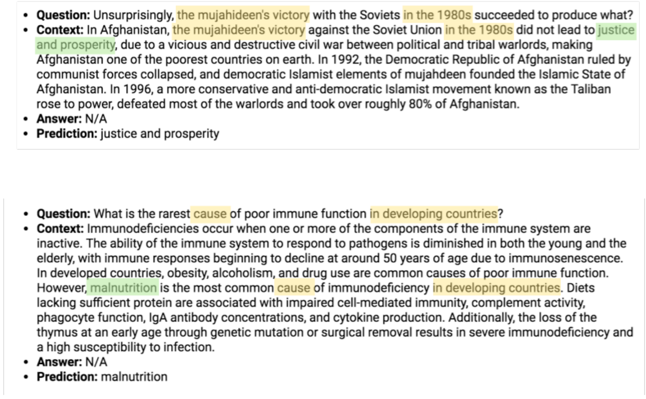
6 Analysis

6.1 Embedding Layer Trade-offs

In a low-resource semi-supervised setting, we find that character and part-of-speech embeddings do improve performance. While they require about double the training time on 100% of training data, the training time is similar to baseline on 25% and 10% of training data. Thus, more complex embeddings may be a worthwhile addition to QA models in low-resource settings.

6.2 Effectiveness of Cloze Pretraining

One particular downside to using cloze questions is the syntactic structure of the questions. Notably, since these questions simply omit the answer phrase from an existing sentence, they do not contain question words, and are not semantically coherent. While there are plausible concerns with using cloze datasets, we found that pre-training on even an extremely small sample of such questions (less than 800) did improve model performance when further trained on a 25% split of SQuAD data. However, the pretrained model's poor performance on an even smaller 10% split suggests that higher quality cloze generation that captures a wider range of question types may be required to improve performance in very low-resource settings. Cloze models may perform better on "What" questions than they do on more complex causal questions, since the process of cloze generation primarily selects noun phrases and named entities as its answers. Furthermore, our model for cloze generation relies on identifying exact phrase matches between question and passage – so the model may learn to favor these exact matches in favor of deeper semantic relations between question sentence and candidate passage. For instance, consider the following erroneous predictions by the model pretrained on Cloze questions and finetuned on only 10% of SQuAD data (Figure 5)



- **Question:** Unsurprisingly, the mujahideen's victory with the Soviets in the 1980s succeeded to produce what?
- **Context:** In Afghanistan, the mujahideen's victory against the Soviet Union in the 1980s did not lead to justice and prosperity, due to a vicious and destructive civil war between political and tribal warlords, making Afghanistan one of the poorest countries on earth. In 1992, the Democratic Republic of Afghanistan ruled by communist forces collapsed, and democratic Islamist elements of mujahdeen founded the Islamic State of Afghanistan. In 1996, a more conservative and anti-democratic Islamist movement known as the Taliban rose to power, defeated most of the warlords and took over roughly 80% of Afghanistan.
- **Answer:** N/A
- **Prediction:** justice and prosperity

- **Question:** What is the rarest cause of poor immune function in developing countries?
- **Context:** Immunodeficiencies occur when one or more of the components of the immune system are inactive. The ability of the immune system to respond to pathogens is diminished in both the young and the elderly, with immune responses beginning to decline at around 50 years of age due to immunosenescence. In developed countries, obesity, alcoholism, and drug use are common causes of poor immune function. However, malnutrition is the most common cause of immunodeficiency in developing countries. Diets lacking sufficient protein are associated with impaired cell-mediated immunity, complement activity, phagocyte function, IgA antibody concentrations, and cytokine production. Additionally, the loss of the thymus at an early age through genetic mutation or surgical removal results in severe immunodeficiency and a high susceptibility to infection.
- **Answer:** N/A
- **Prediction:** malnutrition

Figure 5: Example predictions on 10% train split with Cloze pretraining

Note that in both examples, the model may be relying on the matching phrases to upweight the likelihood of there being an answer, despite semantic differences (such as "did not lead") that would suggest otherwise. Also note that in both cases, the model failed to predict a No Answer; since cloze questions do not incorporate pairs with "No Answer", in a low-resource setting the model may perform worse at correctly predictions for questions without answers. Future automated question generation techniques should consider incorporating answer/no answer variation into the generated pretraining set to account for this gap.

7 Conclusion

Our primary findings are the following: 1) Character-level and part-of-speech embeddings improve question-answering performance on our BiDAF model, for both large and small training sets, though they increase training time as well. (F1 = 52.04, EM = 48.34 for 25% split of SQuAD) 2) Pre-training our model on cloze question-answer pairs before fine-tuning improves performance on 25% of SQuAD training data (F1 = 52.66, EM = 48.35), but worsens performance on 10% of SQuAD training data.

More complex embeddings are certainly a worthwhile investment in low-resource domains, particularly since they do not have a significant impact on training time for small sets of SQuAD training data. Further experimentation and improvement of the cloze question-generation model will be required to determine the optimal amount of cloze question-answer pairs for a given amount of training data.

Future work in the area of semi-supervised question answering should consider other sizes of automated cloze generation (besides 800 samples) to find the smallest pretraining dataset that can produce the largest boost in performance. Additionally, the promise of POS embeddings and other more complex embeddings may be combined with this low-resource setting to produce higher performance. Although our approach to cloze question generation performs reasonably well on 25% SQuAD data, future work may focus on implementing a more robust question generation model to go beyond the model proposed by Dhingra et al., by incorporating neural network techniques [5]. Through a feature-rich input embedding, LSTM encoder, and decoder with attention, Du and Cardie obtain more natural, human-style question answer pairs from input passages (as opposed to cloze fill-in-the-blank style questions). Such techniques may also address some larger gaps in cloze question structure, such as its poorer performance on questions related to causality and other more complex concepts. Furthermore, our study makes the assumption that the first K% of input text is the introduction in order to generate cloze questions – while this model is conducive to the structure of Wikipedia articles, a different paradigm for question/passage distinction may be required for other types of base documents with less predictable document structure.

Special thanks to our project mentor, Sahil Chopra.

References

[1] Wiki extractor. `https://github.com/attardi/wikiextractor`.

[2] Wikimedia downloads. `https://dumps.wikimedia.org/`.

[3] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia to answer open-domain questions. *CoRR*, abs/1704.00051, 2017.

[4] Bhuwan Dhingra, Danish Pruthi, and Dheeraj Rajagopal. Simple and effective semi-supervised question answering. *CoRR*, abs/1804.00720, 2018.

[5] Xinya Du and Claire Cardie. Harvesting paragraph-level question-answer pairs from wikipedia. *CoRR*, abs/1805.05942, 2018.

[6] Michael Heilman. *Automatic Factual Question Generation from Text*. PhD thesis, Pittsburgh, PA, USA, 2011. AAI3528179.

[7] Edison Marrese-Taylor, Ai Nakajima, Yutaka Matsuo, and Yuichi Ono. Learning to automatically generate fill-in-the-blank quizzes. *CoRR*, abs/1806.04524, 2018.

[8] Annamaneni Narendra, Manish Agarwal, and Rakshit shah. Automatic cloze-questions generation. In *RANLP*, 2013.

[9] Jong-Hoon Oh, Kentaro Torisawa, Chikara Hashimoto, Ryu Iida, Masahiro Tanaka, and Julien Kloetzer. A semi-supervised learning approach to why-question answering, 2016.

[10] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100, 000+ questions for machine comprehension of text. *CoRR*, abs/1606.05250, 2016.

[11] Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. *CoRR*, abs/1611.01603, 2016.

[12] Zhilin Yang, Junjie Hu, Ruslan Salakhutdinov, and William W. Cohen. Semi-supervised QA with generative domain-adaptive nets. *CoRR*, abs/1702.02206, 2017.