Question Answering with Recurrent Span Representations

CodaLab Submitter: jwlouie

Timothy Lee
Stanford University
Department of Biomedical Informatics
timothyl@stanford.edu

Kevin Wu
Stanford University
Department of Computer Science
kwu93@stanford.edu

John Louie
Stanford University
Department of Computer Science
jwlouie@stanford.edu

Abstract

Using the Stanford Question and Answering Dataset (SQuAD) released by Rajpurkar et al. [1], we leverage a modified Recurrent Span Representation (RaSoR) model by Lee et al. [2] but with linear space complexity. This contrasts the original model by Lee et. al which requires a demanding quadratic space requirement to enumerate all answer spans. In addition, we introduce a novel architecture, Sequence Question-Induced Distances (SQuID) as an extension to our modified RaSoR implementation, which allows us to impose a prior on the answer span based on a question representation. Using our modified RaSoR implementation, we were still capable of obtaining good model accuracy, getting F_1 and Exact Match (EM) scores of around 60% and 45% respectively on both the CodaLab dev and test set, with less memory and shorter time requirements. However, the results from our extension SQuID model are inconclusive; we believe that while imposing a question-induced prior on what the answer span may allow us to encode better answer distances, a generic question representation may not in itself be a very reliable indicator for the length of an answer span.

1 Introduction

Reading comprehension has long been a hallmark of natural language processing (NLP) research. For computers to demonstrate aptitude at reading comprehension, a question and answering framework is often used. Requiring a machine to answer questions, allows the machine to demonstrate its understanding of potentially complex, multi-sentence logic. As put by Burges, "A machine comprehends a passage of text if, for any question regarding that text that can be answered correctly by a majority of native speakers, that machine can provide a string which those speakers would agree both answers that question, and does not contain information irrelevant to that question" [3].

Thus, we seek to evaluate a novel neural model, SQuID, as well as modified implementations of the RaSoR model, on the SQuAD question and answering framework in order to test our models' reading comprehension.

1.1 Dataset: Stanford Question Answering Dataset (SQuAD)

The SQuAD dataset consists of over 100,000 curated question and context paragraph, **q** and **p**, respectively, tuples where the answer to the question **q** must be drawn from the context paragraph **p** [1]. The answer must be in the form

of a *span*, which is essentially a tuple consisting of two indices, the start of the answer span, a_s, and the end of the answer span a_e. Formally, we have the following SQuAD task:

1.1.1 The SQuAD Reading Comprehension Task (Extractive Answering)

Given a question **q** of length n and a context paragraph **p** of length m, we wish to predict an answer span tuple, $a = \{a_s, a_e\}$, where a_s and a_e denote the beginning and end, respectively, of the answer to question **q** in **p**. The sets $\{p'_1, p'_2, ..., p'_m\}$ and $\{q'_1, q'_2, ..., q'_n\}$ correspond to the GloVE word embeddings for the m words in the context paragraph and n words in the question.

2 Background/Related Work

A significant amount of research has been dedicated to achieving machine comprehension and as mentioned earlier, question and answering frameworks are frequently used to evaluate comprehension. Many previous question and answering datasets are categorized as answer selection datasets, where the machine has a set number of answer possibilities (much like multiple choice questions) to choose from. Using an answer selection approach drastically limits the answer search space and reduces task complexity. However, the question and answering task presented by the SQuAD dataset is one of answer extraction, which is a significantly harder computational task compared to an answer selection task [4].

Research in the realm of the answer selection task includes work by Hill et al. in which the authors proposed a new machine comprehension test known as the Children's Book Test (CBT) [5]. The CBT challenges a machine to utilize contextual information to understand the difference between syntactic function words and words with semantic content. Hill et al. evaluated various models including recurrent neural networks on their CBT [5]. Additional work in answer selection includes work such as that done by Richardson et al. in which they developed a machine comprehension test called MCTest and evaluated a sliding window baseline model on their test [6].

For answer extraction, we drew most of our model inspiration from the RaSor model described by Lee et al. [2]. The RaSor model itself was an extension to a model by Wang & Liang dubbed Match-LSTM. In a later research endeavor, both Want & Liang used their Match-LSTM model along with the Pointer Net neural network model proposed by Vinyals et al. to perform extractive question answering [4]. Both research groups utilized the SQuAD dataset for their extractive answering task.

3 Approach

We implemented a baseline sequence attention mix model (not discussed for brevity), a recurrent span model partially adopted from Lee et. al [2], and developed a novel architecture that extends on our recurrent span model. This model, Sequence Question-Induced Distance (SQuID), layers on top of our recurrent span question answering model by imposing a prior on the length of the answer span. By using the question itself to encode an estimate of the answer span length, we hope to achieve similar model performance as enumerating quadratic span pairs while maintaining a linear runtime.

3.1 Recurrent Span Model

We adopt a similar approach to the RaSor Model from Lee et al. [2] that augments the representation of a passage word with the representation of the question. That is, we define an augmented passage word embedding, called the **question-focused passage word embedding** as $p_i^* = \left[p_i, q_i^{align} q^{indep}\right]$, which is formed from the concatenation of p_i, the original passage word embedding, q^{align}, the passage-aligned question representation, and q^{indep}, the passage-independent question representation. Suppose our question length is n and our passage length is m. Then our representations are described as follows:

1. The **passage-word embedding**, p_i, is simply the original GloVe word embedding for the passage word
2. The **passage-independent question representation**, q^{indep}, provides a coarse-grained summary of the question in a way that captures word ordering and structure. As its name suggests, its representation is independent of the passage and is shared across all passage words. We compute it as follows:

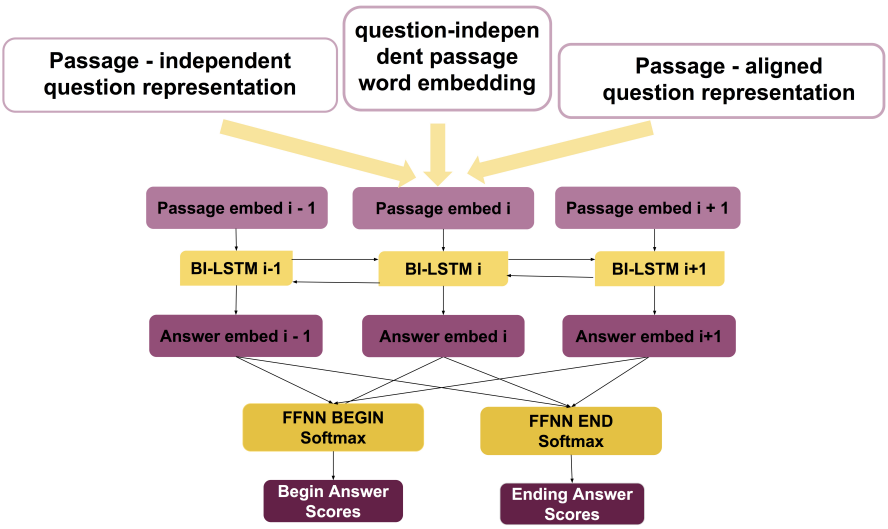

Figure 1: This figure illustrates the modified RaSoR architecture used to predict answer spans from a given SQuAD passage and question (adapted from Lee et al. [2]) For each passage word, three representations were concatenated: passage-independent question, question-independent passage word embedding, and passage-aligned question representations. These embeddings were fed through a BiLSTM, creating hidden vectors that represented the answer embedding for the particular passage word. Each answer embedding was fed through two separate feed-forward neural networks which outputted a softmax-normalized probability whether the word was an beginning or ending answer token.

$$
\begin{aligned}
\{q'_1, q'_2, ..., q'_n\} &= \text{BILSTM}(\mathbf{q}) \\
s_j &= w_q \cdot \text{FFNN}(q'_j) \qquad \forall j \in \{1, ...n\} \\
a_j &= \frac{\exp(s_j)}{\sum_{k=1}^{n} \exp(s_k)} \qquad \forall j \in \{1, ...n\} \\
q^{indep} &= \sum_{j=1}^{n} a_j q'_j
\end{aligned}
$$

3. The **passage-aligned question representation**, $q^{aligned}$, allows us to exploit lexical overlap or similarities that may be contained between the question representation and the passage representation. This is essential as the overlap may often occur near the correct answer span. We use soft-alignment computed via neural attention in a simplified dot product form. Passage-aligned question representation:

$$
\begin{aligned}
s_{ij} &= \text{FFNN}(p_i) \cdot \text{FFNN}(q_j) \qquad \forall j \in \{1, ..., n\}, \forall i \in \{1, ..., m\} \\
a_{ij} &= \frac{\exp(s_{ij})}{\sum_{k=1}^{n} \exp(s_i k)} \qquad \forall j \in \{1, ..., n\} \\
q_i^{align} &= \sum_{j=1}^{n} a_{ij} q_j
\end{aligned}
$$

Once we finish encoding our question-focused passage representation, $\mathbf{p}^* = \left[p_i, q_i^{align} q^{indep}\right]$, we feed this through another BiLSTM to get our answer span embedding h_i^*, which is the concatenation of the forward and backward

outputs of the BiLSTM. However, instead of enumerating the quadratic number of answer spans $(a_s, a_e) \in \mathbf{A}(\mathbf{p})$ that are possible, the idea originally proposed in RaSoR, we pass our answer embeddings through two feed-forward Neural networks to compute a span start score s^{start} for each passage word and similarly an answer end score s^{end}. We then apply a `softmax` to obtain the highest probability answer start index, and further impose a conditional probability on the where the answer end index can reside. That is, given the encoder output of the question-focused passage word embeddings, we first compute a score for the start span s_i^{start}:

$$\begin{aligned} h_i^* &= \text{BiLSTM}(p_i^*) \qquad \forall i \in \{1, ..., m\} \\ s_i^{start} &= \text{FFNN}(h_i^*) \qquad \forall i \in \{1, ..., m\} \\ P(a_i^{start} \mid \mathbf{q}, \mathbf{p}) &= \frac{\exp\left(s_i^{start}\right)}{\sum_{k=1}^{m} \exp\left(s_k^{start}\right)} \\ a_s &= \text{argmax}_i P(a_i^{start} \mid \mathbf{q}, \mathbf{p}) \\ a_e &= \text{argmax}_i P(a_i^{end} \mid a_s, \mathbf{q}, \mathbf{p}) \end{aligned}$$

where the probability distribution for the answer end indices $\mathbf{a^{end}}$ is computed similarly to the start indices using a feed-forward Neural Network, but we derive a conditional probability distribution based on our predicted answer start index a_s.

3.2 SQuID (Sequence Question-Induced Distance Model)

On top of the RaSoR recurrent span model, we introduce SQuID (Sequence Question-Induced Distances) to impose a prior on the length of an answer span given the question only, independently of the passage. The original RaSoR model proposed by Lee et al. considers the scores of $O(n^2)$ different combinations of (a_s, a_e) spans, concatenating the two answer embeddings (one for start, and one for end) for a span embedding. These $O(n^2)$ spans each go through a singular feed-forward neural network and a softmax layer to obtain a final score. On the other hand, our model decouples the scores for the start and end answer tokens, allowing us to decrease the $O(n^2)$ computational complexity to simply $O(n)$. However, our model loses valuable information about the relationship between the two answer span tokens. Furthermore, the model is prone to making inaccurate predictions about the length of the answer by losing this information, leading to overtly long answer spans and lower precision.

We propose the SQuID model to consider the $O(n^2)$ spans, coupling together the scores of beginning answer token and ending answer token with a prediction of the length of the answer span. The goal of the SQuID model is to reason about the complexity of a question and thus the supposed answer length. A question such as "What year was George Washington born?" would require a much shorter answer than a question such as "Why did George Washington not run for a third term in office?". This would help the model impose a restriction on the length of an answer, coupling together the begin and end token answering mechanisms to recognize overly long answer spans as previously discussed. Additionally, this continues to let us bypass the computationally expensive $O(n^2)$ forward propagation step in the original RaSoR model.

First the question word embeddings are fed through a bidirectional LSTM layer, with shared weights across the directions. The final hidden states are concatenated together and are passed through a feed-forward neural network that results in a scalar number ℓ that represents the predicted answer length. The model is trained against a squared loss objective against the true length of the question.

$$\begin{aligned} \{q_1', q_n'\} &= \text{BILSTM}(\mathbf{q}) \\ q_{concat}' &= [q_1'; q_n'] \\ Squid_l &= \text{FFNN}(q_{concat}') \end{aligned}$$

During the evaluation of a question and passage pair, we consider each of the $O(n^2)$ answer spans and find the corresponding answer indices that maximize the sum of the beginning answer and end answer scores (derived from the RaSoR model) and the score loss associated with the difference of the proposed answer span length and the predicted answer length (derived from SQuID). Mathematically, the objective is:

$$a_s^*, a_e^* = \underset{a_s, a_e}{\text{argmax}} \quad s_s^{start} + s_e^{end} - \gamma \cdot \min\left(\left((a_e - a_s) - \ell\right)^2, \Delta\right)$$

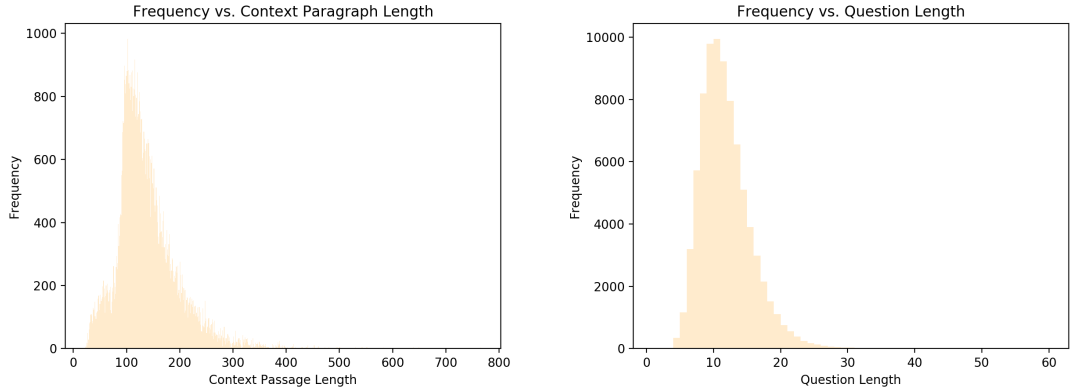

Figure 2: Distribution of passage lengths and question lengths for training examples. Most of the probability mass for our passage lengths distribution resides at lengths $m \leq 300$. Similarly, almost all questions have length $n \leq 30$. As a result, for convenience in the computational graph, a limit of 300 words was set for the analysis of a context paragraph and the question.

where $0 \leq a_s \leq a_e < n$, with a_s, a_e as proposed beginning and end answer indices, a_s^*, a_e^* as the final predicted answer indices. s_i^{start}, s_i^{end} denote the RaSoR decoupled beginning and end answer probabilities, respectively. ℓ represents the predicted answer length from the SQuID model, γ, is a hyperparameter that controls the amount of influence SQuID has on the overall score; it has the effect of controlling the strength of the SQuID. Δ, or SQuID loss threshold, is another hyperparameter that limits the decisive activity of the SQuID model, determining when it should consider all answer spans above or below a certain length to have the same cost for deviation from the predicted answer length.

4 Experimental Setup

All models were implemented in TensorFlow and trained on the SQuAD dataset. All word embeddings are represented using either 100- or 300-dimensional trimmed GloVe word embeddings. These GloVe embeddings were pretrained on the Wikipedia 2014 and Gigaword 5 datasets. We initially experimented with no dropout and then subsequently used dropout to varying degrees (10%, 30%, 50%) on all feed-forward networks and using a single dropout mask across all LSTM time steps for recurrent networks [7]. In feed-forward neural networks, we use rectified linear units (ReLU) for our non-linear activation function. For LSTMs, we allow bidirectional flow and experiment with single layer and multi-layer LSTM cells. Hidden sizes for LSTMs and feed-forward Network varied between 50 and 200 units. For our optimizer, we used the Adam optimizer and experimented with (10, 32, 40, 64) minibatch size [8].

In the context of our question answering system, we truncated certain passages to a fixed passage length, m, which allowed us to improve convergence times. To determine an optimal m, we examined a histogram of passage lengths within our training set and tried a few different values (namely 250, 300, 750). Models trained with either $m = 250, 300$ ran roughly twice as quickly. Because masks were used, we ignored setting a maximum question length n and instead opted to pad our question sequences. However, an optimization for improved convergence for any additional future work would be to truncate question lengths to $n = 30$ as well, leading to less memory usage and extraneous computations on padding tokens with no loss of accuracy.

Similar to Match-LSTM and RaSoR, we impose a fixed span limit of 15 on our recurrent span model to ensure that answer spans do not exceed 15 words. We vary this hyperparameter (12, 15, 20, 30), though find that it only marginally changed the F_1 and/or EM score, provided it is a reasonable number.

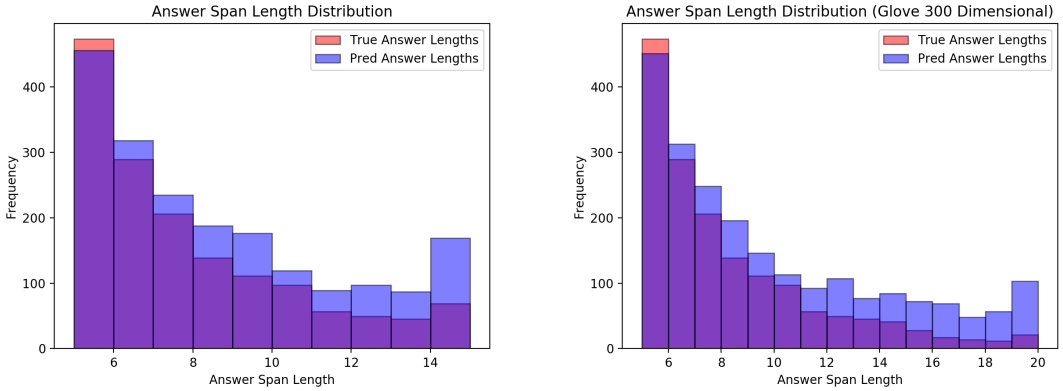

Figure 3: These graphs show histograms of predicted and true answer span lengths superimposed upon each other. The purple area represents the number of samples for the where both the true and predicted samples agreed on the same span length, blue area represents an overflow amount of predicted answer spans over the true answer spans, and vice versa for red area. This demonstrates that the adapted RaSoR model tends to have longer answer spans than necessary, decreasing precision. This thus necessitates a model such as SQuID to impose a span length restriction. When using a higher dimensional word embedding representation (in this case a 300 dimensional GloVe representation) and greater leniency in possible span lengths, we observe that the skew tends towards even longer span predictions, thus, further lowering precision

5 Results and Analysis

Before detailing our results, we begin by describing the evaluation metrics we used in order to compare our various models' performance.

5.1 Evaluation Metrics: F1 score and EM score

Within the original SQuAD paper by Rajpurkar et al., the authors utilized two metrics, F_1 and Exact Match (EM), both of which we utilized during model evaluation [1]. Using these two metrics allows us to compare our models' performance to state-of-the-art models on the SQuAD dataset (a leaderboard of top performing models is provided at https://rajpurkar.github.io/SQuAD-explorer/).

F_1 is defined as the harmonic mean between precision and recall, $F_1 = 2 \times \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$. Precision refers to the number of tokens classified correctly as part of an answer over the total number of tokens classified as part of an answer, whereas recall refers to the number of tokens classified correctly as part of an answer over the total number of answer tokens. We utilize F_1 in order to obtain "partial credit" for what answer spans we predicted with our model. The F_1 score is computed using a "bag of words" approach, meaning the words are effectively unordered, non-unique sets and all we care about is the intersection between the words of our predicted answer and the ground truth. Relative to EM, the F_1 is a more lenient metric of correctness.

Unlike F1 score which allows for partial credit of a predicted answer, EM (Exact Match) is the percentage of predictions for question-paragraph pairs which matched, token for token, with the corresponding ground truth answers for the question-paragraph pairs.

5.2 Results

At the time of submission, our strongest performing model was the modified recurrent span model with a span limit of 15 and dropout of 0.1 (entry bolded in Table 1 below). For Table 1, where not specified, we trained our model with the 100 dimensional GloVe word embeddings, and output size of 300. In addition, where not specified, we used a span limit of 15 when evaluating on the dev set.

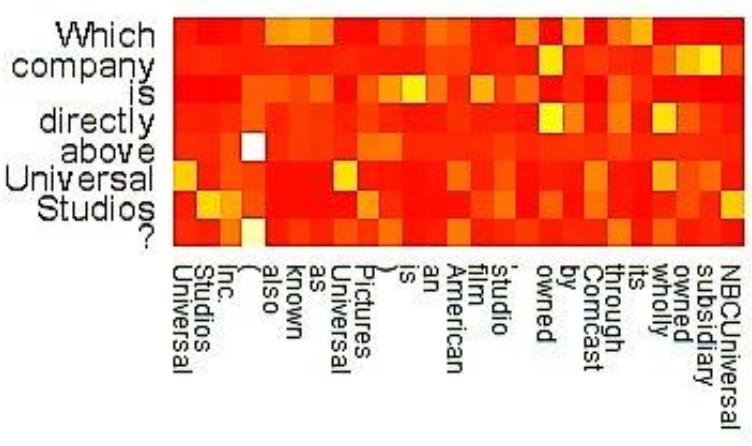

Figure 4: Heatmap visualization of soft attention alignment scores between our question and the passage. Red indicates less alignment and yellow indicates greater alignment. As we can see, the question words align up significantly with context paragraph words that are of semantic similarity (Universal - Universal, Studios - Studios, company - subsidiary).

Table 1: Various models' `dev` set performance, measured with F_1 and EM. Our best model is the recurrent span model with dropout probability 0.1. Varying the span limit incrementally does not change the F_1 and EM score by much. Too high of dropout appears to cause performance to deteriorate. Additionally, higher-dimensional word embeddings appear to do 1-2% worse. We suspect this may be due to early overfitting which plays poorly with the momentum of the Adam optimizer. Finally, our SQuID model does not achieve near the performance of our basic recurrent span model.

MODEL	F_1 **SCORE**	**EM SCORE**
Recurrent Span Model (No dropout)	58.682	45.08
Recurrent Span Model (Dropout = 0.1)	**59.009**	**45.307**
Recurrent Span Model (Dropout = 0.1, Span Limit = 10)	58.667	45.449
Recurrent Span Model (Dropout = 0.1, Span Limit = 20)	59.098	45.184
Recurrent Span Model (Dropout = 0.1, Span Limit = 30)	59.035	44.967
Recurrent Span Model (Dropout = 0.3)	53.863	40.511
Multilayer Recurrent Span Model	54.620	41.325
Recurrent Span Model with 300D GloVe	57.685	43.822
Recurrent Span Model with 300D GloVe (Span Limit = 20)	57.694	43.614
Recurrent Span Model w/ SQuID	32.263	20.916

As for the SQuID extension model, with the given architecture and time constraints, we were not able to successfully see incremental improvement during its training. This could be due to general variance in answer span lengths for any question without the passage. For example, the question "What is a butterfly?" could have a set of possible answer spans from the most general "insect" to highly specific "the month-long result of a chrysalis of the order Lepidoptera".

6 Conclusion

Overall, through developing our recurrent span model, we were able to produce competitive results while training for fairly short periods of time (≤ 5 epochs). The tradeoff between model complexity and accuracy was highly apparent as we chose to simplify some of the architecture presented in RaSoR to optimize for runtime; while we sacrificed performance, we were able to explore a wider array of interesting ideas relating to our model development. Random initialization of hyperparameters allowed us to better understand the cost landscape for our neural architecture; for example, the balance between too much regularization, causing us to distort the learning process, and too little, causing us to overfit too quickly in the first epoch. These are key concepts that we took away from this project.

There are many potential extensions that can be implemented for our current model. Due to time constraints and limited computing resources, we did not get to explore ensemble methods. With respect to creating an ensemble of learned models, one possible approach would be to average answer start and end scores produced by all our models

during evaluation on our development set. Past literature has generally reported a guaranteed 3-5% improvement in F1/EM with ensemble techniques. Another extension would be to perform more extensive hyperparameter tuning, allowing us to better adjust for overfitting that occurs within the training set. With respect to SQuID, a higher-level, heuristic approach to using and predicting span distances may be desired. Also, perhaps our representation was not complex enough to correctly predict the length answer spans. Given more time, we would love to perform some more in-depth studies of how different question representations and perhaps key words within questions lend themselves to shorter or longer answers. Finally, allowing SQuID to be less rigid and more open to answers of different lengths would be desired; this would ideally allow SQuID to aid, rather than enforce, a specific length restriction.

7 References

Acknowledgements

We would like to thank the entire CS 224N teaching staff for all their help in making this course run as smoothly as possible. We would also like to thank Microsoft for generously allowing us to pursue our research on their Azure GPUs.

References

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

[2] Kenton Lee, Tom Kwiatkowski, Ankur Parikh, and Dipanjan Das. Learning recurrent span representations for extractive question answering. *arXiv preprint arXiv:1611.01436*, 2016.

[3] Christopher J.C. Burges. 2013. Towards the machine comprehension of text: An essay. Technical report, Microsoft Research Technical Report MSR-TR-2013- 125.

[4] Shuohang Wang and Jing Jiang. Machine comprehension using match-lstm and answer pointer. *arXiv preprint arXiv:1608.07905*, 2016.

[5] Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. The Goldilocks principle: Read- ing childrens books with explicit memory representations. In *Proceedings of the International Conference on Learning Representations*, 2016.

[6] Matthew Richardson, Christopher J.C. Burges, and Erin Renshaw. MCTest: A challenge dataset for the open-domain machine comprehension of text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2013.

[7] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In Proceedings of ICML, 2010.

[8] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. Proceedings of ICLR, 2015.