Ranking Quality of Answers Drawn from Independent Evident Passages in Neural Machine Comprehension

Avi Bleiweiss
BShalem Research, Sunnyvale, U.S.A.

Keywords: Machine Comprehension, Gated Recurrent Neural Networks, Coattention Mechanism, Passage Ranking.

Abstract: Machine comprehension has gained increased interest with the recent release of real-world and large-scale datasets. In this work, we developed a neural model built of multiple coattention encoders to address datasets that draw answers to a query from orthogonal context passages. The novelty of our model is in producing passage ranking based entirely on the answer quality obtained from coattention processing. We show that using instead the search-engine presentation order of indexed web pages, from which evidence articles have been extracted, may affect performance adversely. To evaluate our model, we chose the MSMARCO dataset that allows queries to have anywhere from no answer to multiple answers assembled of words both in and out of context. We report extensive quantitative results to show performance impact of various dataset components.

1 INTRODUCTION

Machine comprehension (MC), an extractive form of question answering over a provided context paragraph, is an active research area in natural language processing (NLP). MC models that answer questions over unstructured text are expected to have profound impact on people who inquire about medical, legal, or technical documents. In practice, trained MC systems can be applied to a variety of real-world problem domains, including personal assistance, recommendations, customer support, and dialog management.

The rise of large-scale human-curated datasets with over 100,000 realistic question-answer pairs, such as SQuAD (Rajpurkar et al., 2016), NewsQA (Trischler et al., 2016), MSMARCO (Nguyen et al., 2016), and TriviaQA (Joshi et al., 2017), inspired a large number of successful deep learning models for MC (Lee et al., 2016; Wang and Jiang, 2016; Seo et al., 2017; Xiong et al., 2017; Dhingra et al., 2017; Raiman and Miller, 2017; Chen et al., 2017). In Table 1, we show a comparison of key dataset properties that sets MSMARCO as the more realistic and thus posing many unique challenges to an MC engine, on several levels. Questions in MSMARCO are real user queries tagged with categorical information, as free-formed answers are generated by humans, each potentially derived from several authentic web documents (see Appendix for a dataset example composition). Moreover, a subset of the queries has multiple answers and possibly no answer. Unlike a single answer of contiguous text snippet that is confined to either one document as in SQuAD and NewsQA, or to a single article member of a set of articles, in TriviaQA.

To date, MSMARCO remains fairly understudied and most of the work published (Shen et al., 2016; Wang and Jiang, 2016; Weissenborn et al., 2017; Wang et al., 2017; Tan et al., 2017) was primarily designed to run natively on SQuAD and NewsQA, and followed by coercing MSMARCO to behave like the former datasets. For instance, all text passages associated with a query were concatenated to a single context paragraph (Weissenborn et al., 2017; Wang et al., 2017) and similarly, only the first of multiple answers was evaluated (Tan et al., 2017). In this paper, we propose a neural MC design that is tightly linked to the MSMARCO interface abstraction. We draw answers to a query from each of the text passages separately, and to sustain performance our passage scoring is derived independently of the search-engine page ranking. A ranking we assigned an instructive role rather than binding as in the work by Tan et al. (2017).

Recently, recurrent neural networks (RNN) became a widespread foundation for language modeling tasks (Sutskever et al., 2011; Sutskever et al., 2014; Hoang et al., 2016; Tran et al., 2016). This had spurred a growing interest in developing machine learning models that use an attention-based RNN to extend other NLP tools (Luong et al., 2015; Xu et al., 2015) to MC, and achieve compelling results (Wang

Bleiweiss, A.
Ranking Quality of Answers Drawn from Independent Evident Passages in Neural Machine Comprehension.
DOI: 10.5220/0006923300610071
In *Proceedings of the 10th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2018) - Volume 1: KDIR*, pages 61-71
ISBN: 978-989-758-330-8
Copyright © 2018 by SCITEPRESS – Science and Technology Publications, Lda. All rights reserved

Table 1: Comparing key properties of existed large-scale datasets with over 100,000 realistic question-answer pairs. MS-MARCO is the only dataset to draw answers by crossing evidence page boundaries.

Dataset	Segment	Query Source	Answer	Queries	Documents
SQuAD	N	Crowdsourced	Span of Words	107,785	536
NewsQA	N	Crowdsourced	Span of Words	119,633	12,744
MSMARCO	Y	User Logs	Human Generated	102,023	837,729
TriviaQA	Y	Crowdsourced	Span of Words	110,495	662,659

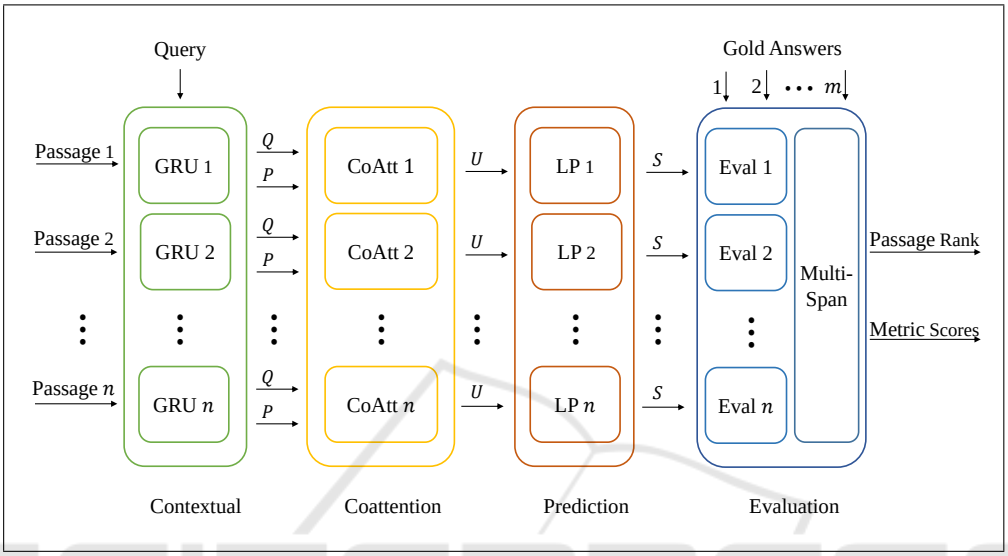


Figure 1: Architecture overview of our deep learning model. Word embeddings of a query, n context passages, and m answers form a machine comprehension primitive to enter our computational pipeline. Pairs of the query and each of the passages are first fed into independent gated recurrent neural networks (GRU). Contextualized query Q and passage P from each GRU are then passed onto coattention encoders, each emitting the coattention context U. Simple linear decoders take U and predict the endpoints of an answer span S in a context passage. Lastly, the evaluation layer combines multiple spans to address non-contiguous answers, and produces both passage ranking and answer quality scoring in ROUGE and BLEU metrics.

and Jiang, 2016) (Cui et al., 2016) (Xiong et al., 2017) (Seo et al., 2017). Perceived almost unanimously by the research community, the latest pace of advancing MC in the text domain is largely attributed to the attention mechanism that requires cross-encoding and exchanging information between a context paragraph and a given query. In creating our neural MC model, we expanded upon the Bidirectional Attention Flow (BiDAF) network (Seo et al., 2017) that is one of the best performing models on the SQuAD dataset with an Exact Match (EM) and F1 scores of 73.3 and 81.1, respectively. BiDAF computes complementary context-to-query and query-to-context attention, and predicts the endpoints of an answer span. Given the generative nature of the MSMARCO dataset, we produce instead a sequence of words for a predicted answer length, and use ROUGE (Lin, 2004) and BLEU (Papineni et al., 2002) as our evaluation metrics.

The main contribution of this paper is an effective architecture proposed for MC datasets that draw answers from each of multiple context passages. We hypothesize that using the PageRank algorithm to quantify answer quality is inferior to our implicit passage ranking that we derive directly from the coattention pipelines, and provide compelling proof in our experimental results. A number of our major design decisions were based on inspecting the MSMARCO data, as we trained our model on its train set, and analyzed performance on the development set. The rest of this paper is structured as follows. In section 2, we define MC primitives and overview computational flow of our model. Section 3 renders pre-processed statistical data of the MSMARCO dataset, and in Section 4 we present our methodology for evaluating performance, and report extensive quantitative results over a range of ablation studies. Summary and identified avenues for prospective work are provided in Section 5.

2 METHOD

In Figure 1, we provide an overview of our proposed multiple-pipeline MC model that progresses through

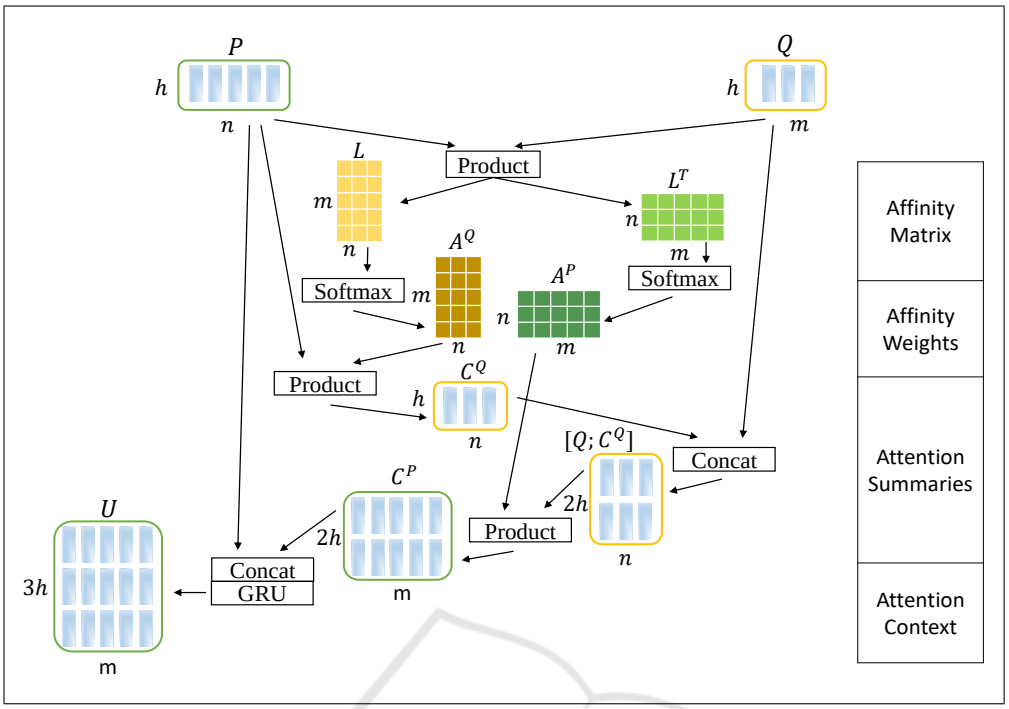


Figure 2: Architecture diagram of the coattention encoder that simultaneously induces attention contexts for both the query Q and passage P, and fuses them together to output the U matrix. The encoder execution order is shown on the right.

a series of contextual, attention, prediction, and evaluation layers we describe next.

2.1 Word Embeddings

We define a machine comprehension primitive in text word space as the triplet $\lambda_w = (q_w, P_w, A_w)$, where q_w the query, $P_w = \{p_w^{(1)}, \ldots, p_w^{(n)}\}$ a set of n context passages, and $A_w = \{a_w^{(1)}, \ldots, a_w^{(m)}\}$ a collection of m reference, or gold answers. Drawn from a pre-fixed vocabulary V, each of q_w, $p_w^{(i)}$, and $a_w^{(j)}$ is a sequence of tokens that are first mapped to a sparse one-hot vector representation $\in \mathbb{R}^{|V|\times 1}$, and are then transformed to dense d-dimensional vectors using an embedding matrix $E \in \mathbb{R}^{d\times|V|}$ (Mikolov et al., 2013a; Mikolov et al., 2013b; Pennington et al., 2014). We use the colon notation $v_{1:k}$ to denote the sequence $(v_1, \ldots, v_k)$ of k vectors, and a text of length T words is therefore represented as a sequence of d-dimensional word vectors $x_{1:T} \in \mathbb{R}^d$. In the embedding space, components of the triplet $\lambda_x = (q_x, P_x, A_x)$ unfold to the sequences $q_x = x_{1:|q|}$, $p_x^{(i)} = x_{1:|p^{(i)}|}$, and $a_x^{(j)} = x_{1:|a^{(j)}|}$ that render a query, context passage, and answer, respectively, where cardinalities $|q|$, $|p^{(i)}|$, and $|a^{(j)}|$ are denoted invariably to any of w or x projection spaces. At any time, our system responds to querying both the forward and backward transforms $\lambda_w \rightleftarrows \lambda_x$.

2.2 Contextual RNN

Our first pipeline stage contextualizes word vector representations of the query q_x paired with each of the context passages $p_x^{(i)}$, by feeding them to a set of n independent recurrent neural networks. We chose to use the Gated Recurrent Unit (GRU) (Cho et al., 2014; Chung et al., 2014) variant of RNN that captures long term dependencies of the input as it encodes each word x_t into a sequence $h_t = \text{RNN}(h_{t-1}, x_t)$. GRU performs similarly but is computationally more efficient than LSTM (Hochreiter and Schmidhuber, 1997). We denote $B = \text{GRU}(A, h_0)$ to indicate that the matrix B is comprised of the output states of a unidirectional GRU-based RNN, where each column of A is an input word vector and h_0 is the starting hidden state. Word vectors of each of the query-passage pairs are fed through the same GRU block and computationally follow the set of formulas

$$Q' = \text{GRU}(q_x, 0) \in \mathbb{R}^{h\times|q|}$$
$$P = \text{GRU}(p_x, Q'[:|q|]) \in \mathbb{R}^{h\times|p|}$$
$$Q = \tanh(W^{(Q)}Q') \in \mathbb{R}^{h\times|q|},$$

where Q' is an intermediate query representation of the GRU output layer, and $W^{(Q)}$ is a weight matrix to learn. In the remainder of this discussion, we follow a concise matrix notation, as we review computation

of a single pipeline of the architecture, and drop passage enumeration (i) altogether for clarity. Q' follows a non-linear projection step to produce Q and ensures the query and passage reside in state spaces of no linear relation. To simulate the comprehension test practice of 'read the query before the passage', we route the final hidden state of query encoding to the initial hidden state for encoding the context passage.

2.3 Coattention Encoder

In this section, we provide a brief overview of the attention mechanism (Xiong et al., 2017; Seo et al., 2017; Andress and Zanoci, 2017). From the contextual stage, a pair of Q and P matrices enter each a coattention pipeline (Figure 2) to produce final passage encoding. First, we compute all-word-pairs similarities between a context passage and a query to produce the affinity matrix L

$$L = P^T W^{(L)} Q \in \mathbb{R}^{|p|\times|q|},$$

where $W^{(L)} \in \mathbb{R}^{h\times h}$ is a trainable weight matrix. An affinity element l_{ij} holds the similarity between word i in the context passage and word j in the query. Next, L is normalized over rows and columns to produce attention weights in the shape of matrices A^Q and A^P, respectively

$$A^Q = \text{softmax}(L) \in \mathbb{R}^{|p|\times|q|}$$
$$A^P = \text{softmax}(L^T) \in \mathbb{R}^{|q|\times|p|},$$

where softmax$(\cdot)$ is a row-wise operator. Using the attention weights, we compute attention summaries for the query

$$C^Q = PA^Q \in \mathbb{R}^{h\times|q|},$$

where for each word k in the query, we compute a weighted average over the entire word vectors of the passage, using the weights in the k-th row of A^Q. To calculate the summaries of each word in the context passage, we first concatenate the original state encoding of the query Q with the query attention summaries C^Q and compute the attention summaries for the context

$$C^P = [Q; C^Q]A^P \in \mathbb{R}^{2h\times|p|},$$

where $[\cdot;\cdot]$ is a row-bound matrix concatenation. Likewise we fuse the native passage encoding P and the summarized passage attention C^P in the concatenated matrix $[P; C^P] \in \mathbb{R}^{3h\times|p|}$ that is identified as the final knowledge representation of the coattention encoder.

The last operational step of the encoder temporally merges different parts of the coattention context using a dedicated GRU

$$U = \text{GRU}([P; C^P], 0) \in \mathbb{R}^{3h\times|p|},$$

where matrix U is the output of the coattention encoder that is passed to the following prediction layer of the MC pipeline.

2.4 Decoder

In the MSMARCO dataset, answer complexity ranges widely from simply binary or an entity name, through a contiguous phrase confined to a single context passage, to possibly non-contiguous text that might cross passage boundaries. Nonetheless our initial data analysis revealed that 82.6 percent of the dataset answers are composed of a contiguous sequence of tokens contained in its entirety in one passage. Our design predicts not only one (Rajpurkar et al., 2016) but n answer spans, each delimited by a begin and end word positions in a passage, and provides span concatenation to address non-contiguity. We use a linear projection to convert coattention word vectors to scores and apply softmax to obtain the probability distribution of a span begin index s_b over the entire passage context

$$p(s_b) = \text{softmax}(W^{(s)} U) \in \mathbb{R}^{|p|},$$

where $W^{(s)}$ is a weight vector $\in \mathbb{R}^{1\times 3h}$. Using another GRU node, we project U onto a different hyperspace U' before obtaining posterior probabilities of the span end point s_e

$$U' = \text{GRU}(U, 0) \in \mathbb{R}^{3h\times|p|}$$
$$p(s_e) = \text{softmax}(W^{(e)} U') \in \mathbb{R}^{|p|},$$

where $W^{(e)}$ has the same dimensionality as $W^{(s)}$. Our prediction for the answer span S follows

$$S = \underset{1\le s_b\le s_e\le|p|}{\text{argmax}} \; p(s_b)p(s_e),$$

that we overload as $S = \{s_b, s_e\}$. Each of the individual passage decoders emits S that is conveyed to the next layer for comprehension evaluation.

2.5 Deferred Passage Ranking

In both the MSMARCO and TriviaQA datasets, contextual passages to draw answers from are extracted from open web pages. Passages provided to either human judges or MC systems have the attractive property of ranked order presentation from the perspective of the search engine. Perceived more as a suggestion rather than an imperative requisite to follow, human crowdsourced answers may not abide by a page ranking protocol to best match an answer to a query.

In their model, Tan et al. (2017) feed the webpage origin order directly to a passage ranking task that they chose to integrate into the coattention encoder. This had led to an improvement of answer span

Table 2: MC component statistics of train and development sets, including per-query average of passages and answers.

	Queries	Passages	Answers	Mean Passages	Mean Answers
train	82,326	676,193	90,306	8.21	1.09
dev	10,047	82,360	10,973	8.19	1.09

Table 3: Distribution of per-query passage count $[1,12]$ extracted from the train and development sets.

	1	2	3	4	5	6	7	8	9	10	11	12
train	3	43	316	1,249	3,224	5,854	10,239	19,682	27,378	14,335	1	2
dev	0	5	39	145	399	713	1,291	2,444	3,319	1,692	0	0

Algorithm 1: Deferred Passage Ranking.

input: MC primitive λ_w, predicted answers A_p
output: ranked passage scores $R \leftarrow \emptyset$
$A_r \leftarrow$ answers(λ_w)

for $i = 1$ **to** num_passages(λ_w) **do**
  $s \leftarrow 0$
  for $j = 1$ **to** num_answers(λ_w) **do**
    if null$(A_r[j])$ **then**
      $s \leftarrow s + 0$
    else
      $s \leftarrow s$+ROUGE_or_BLEU$(A_p[i], A_r[j])$
    end if
  end for
  $R \leftarrow R \cup (s/\text{num\_answers}(\lambda_w)$
end for
return sort(R)

prediction by about 2.5 percentage points, and also allows for more efficient training. However, by using attention pooling their method predicts from an inherently biased context. Instead, we propose an MC system that throughout the entire machine comprehension task remains impartial to subdivided context that enters our pipelines. We evaluate all the query-bound passages and rate context spans either independently or concatenated, by scoring answers individually first and then as a collection. This process is further illustrated in Algorithm 1. Moreover, implicitly producing a rank of passage scoring at the last step of the evaluation stage (Figure 1), merits the analysis of comparative performance between our unbiased passage order and MSMARCO web-page ranking, to facilitate the testing of our hypothesis.

3 MSMARCO DATASET

Version 1.1 of the MSMARCO dataset consists of 102,023 queries with their corresponding answers, of which 82,326 are used for training, 10,047 for development, and 9,650 for test. Intended primarily for official leaderboard submissions, the test set is released to the public either without or with incorrect answers. In this work, we chose to report main evaluation results on the development set.

Table 4: Distribution of per-query answer count $[0,4]$ drawn from the train and development sets.

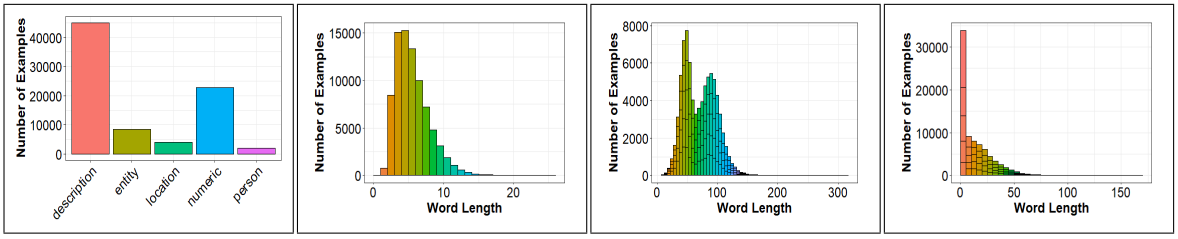
	0	1	2	3	4
train	2,183	70,616	8,912	594	21
dev	293	8,608	1,076	67	3

Table 5: Distribution of passage-bound contiguous and passage-straddled non-contiguous answer spans drawn from the train and development sets.

	Contiguous Spans	Non-Contiguous Spans
train	635,047	110,596
dev	77,042	13,405

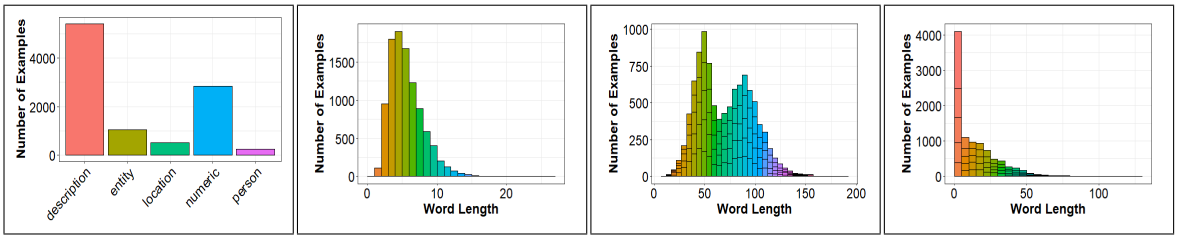
In Table 2, we list aggregates of MC components for the train and development collections, and highlight per-query average of context passages and answers at about 8.2 and 1.1, respectively. Table 3 further expands on passage distribution and shows over 87 percent of the queries have between seven to ten context passages to draw the answers from. Similarly, Table 4 shows answer distribution with at most four answers per-query, as the majority of nearly 86 percent of queries have a single answer. A small subset of queries, at about 2.6 percentage points, have no answers at all.

Answers presented in the MSMARCO dataset may not be syntactically fluent and potentially require reasoning across multiple context passages. Still, our initial data analysis, as evidenced by Table 5, shows that more than 83 percent of answers comprise contiguous spans of text confined to a single passage, where the remaining 17 percent are formed of non-contiguous text fragments bound to either one or multiple passages. This data firmly supports our proposed MC model that reasons over both a single and multiple answer spans. Table 6 follows to show distribution of context-unaware answers that include about 7.5 percent of binary answers to Yes/No question types,



(a) Query Types (b) Queries (c) Passages (d) Answers

Figure 3: Train set histograms of (a) query types, and word length of (b) queries, (c) passages, and (d) answers.



(a) Query Types (b) Queries (c) Passages (d) Answers

Figure 4: Development set histograms of (a) query types, and word length of (b) queries, (c) passages, and (d) answers.

Table 6: Distribution of binary answers to Yes/No type questions and special-case misconstrued answers drawn from the train and development sets.

	Binary	Misconstrued
train	6,728	136
dev	755	18

and inconsequential cases of misconstrued answers. The latter are formed of non-standard English tokens that often transpire in online forum discussions. For instance, words like ‘yez’, ‘maxico’, and ‘crockery’.

The distributions of query segment types and word length for each the queries, context passages, and answers in the train and development sets are shown in Figure 3 and Figure 4, respectively. Histograms are consistent on both sets, barring the passage maximal length of 321 vs. 192 words. Over half of the queries belong to the description category, the numeric class is second with 28.3 percent, as entity, location, and person trail behind. Notably a little over fifty percent of MC examples have short answers of ten words or less, with an average answer length of fourteen words.

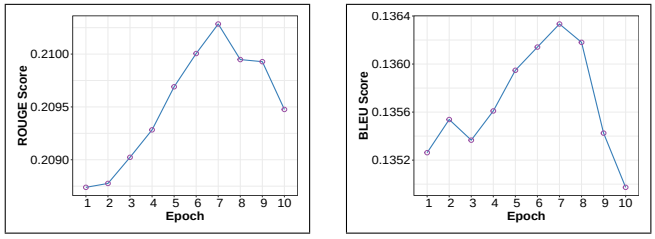
4 EMPIRICAL EVALUATION

In this section, we provide an extensive quantitative analysis of our neural MC architecture. To benchmark answer quality for generated text, we expand our predicted spans and abide by MSMARCO scoring of the longest common subsequence ROUGE-L (Lin, 2004) and the unigram BLEU-1 (Papineni et al., 2002) for performance metrics. We denote these measurements as ROUGE and BLEU for brevity.

4.1 Experimental Setup

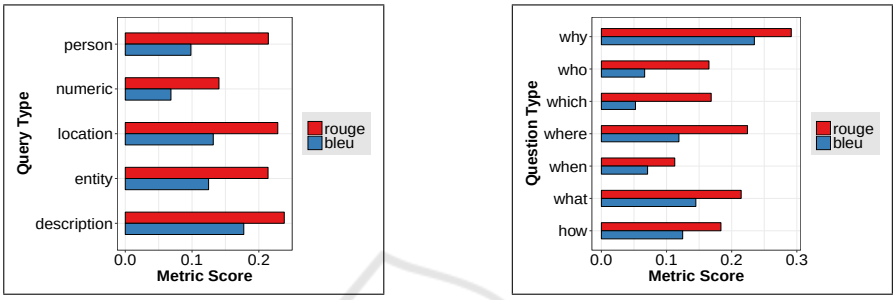
To evaluate our system in practice, we have implemented our neural MC model (Figure 1) natively in R (R Core Team, 2013). Our linguistic process commences by tokenizing and lowercasing queries, context passages and answers into vocabularies of 371,807 and 110,310 words for the train and development sets, respectively. We chose to train our own word vectors using Word2Vec (Mikolov et al., 2013a) with dimensionality $d = 100$. The GRU hidden state dimension h was set to 100, as all weight matrices and vectors were bootstrapped using Xavier initialization (Glorot and Bengio, 2010). Components of MC primitives in word space, λ_w, that enter our pipeline are each of a variable length, as our system avoids any of truncation or padding to uniformly shape each of the query and context passages, a practice often used to reduce training time (Andress and Zanoci, 2017).

We trained our model on MSMARCO train set with the Adam stochastic gradient optimizer (Kingma and Ba, 2014) for ten epochs, using its provided default settings. Five percent of the train split was allocated for validation and used for hyperparameter tuning, with final choices based on both loss and performance behavior. To mitigate overfitting the train data in our model, we used both dropout (Srivastava et al., 2014) at a rate of 0.2 to regularize our network, and early termination of the training process by monitoring the loss after each epoch. We use the basic cross-



(a) ROUGE (b) BLEU

Figure 5: Development performance as a function of the stochastic optimization epoch for (a) ROUGE and (b) BLEU score metrics. Our best model is produced around the seventh epoch.



(a) Query Type (b) Question Type

Figure 6: Breakdown of development model performance based on different types of (a) queries and (b) questions. Shown for each the ROUGE and BLEU score metrics.

entropy loss with the objective function to minimize

$$\mathcal{L} = -\frac{1}{N}\sum_{i} a_b^{(i)} \log p^{(i)}(s_b) + a_e^{(i)} \log p^{(i)}(s_e),$$

where N is the number of aggregate context passages in the train set (Table 2), and the pair denoted $\{a_b^{(i)}, a_e^{(i)}\}$ are the ground-truth endpoint labels for a gold answer converted to a span that has the longest common subsequence (LCS) in the i-th context passage. Our best model is produced right about crossing seven epochs for both the ROUGE and BLEU performance metrics, as shown in Figure 5.

In a single span mode, we evaluate each query-bound context passage individually for any of none, single, or multiple answer scenarios, as is outlined in Algorithm 1. All of the queries that have no reference answer, and for that matter gold answers that are not well formed, are assigned uniformly a score of zero. Otherwise, each answer is rated independently first, and the mean of all the answer scores makes up the final rating of the currently evaluated text passage. The answer quality we report for the query culminates in the score of its highest ranked passage. Our produced passage ranking is contrasted against the dataset order and moreover, it feeds into the following multi-span stage and aids to prioritize span selection. Algorithm 1 is slightly modified in rendering multi-span mode, as the outer loop iterates over concatenated spans.

4.2 Experimental Results

We report our results on MSMARCO development set in ROUGE and BLEU performance metrics, and unless noted otherwise, we uniformly use normalized score figures in the $[0, 1]$ range.

The first set of ablation experiments we conducted let us better understand how various components in our training methodology impact our overall system performance. In Figure 6(a), we show our model behavior over query intent segments that the dataset creators solicited through classification based on crowdsource labeled data. As expected, we performed best on the description segment that identifies over fifty percent of the dataset queries (Figure 4). The location category came closely second and followed by a similar performance for the entity and person type classes. However, despite a relative large share of examples, about a third, the numeric grouping is our least performing, due primarily to the difficulty to contextualize a single number. From an alternate perspective, performance as a function of the query question type is illustrated in Figure 6(b). We note that only 68.8 percent of the entire MSMARCO query set explicitly contain ‘what’, ‘how’, ‘where’ question type keywords (Nguyen et al., 2016), even though the remaining queries might have an identical expressive goal. Surprisingly, we achieved our top performance, 0.28 ROUGE, on ‘why’ headed questions that are gener-

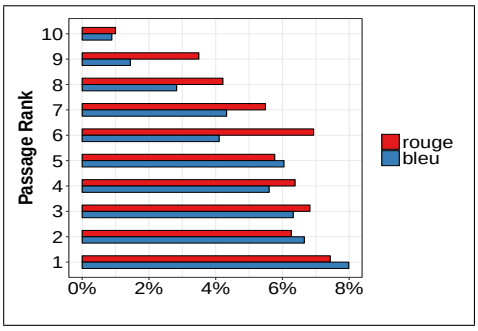
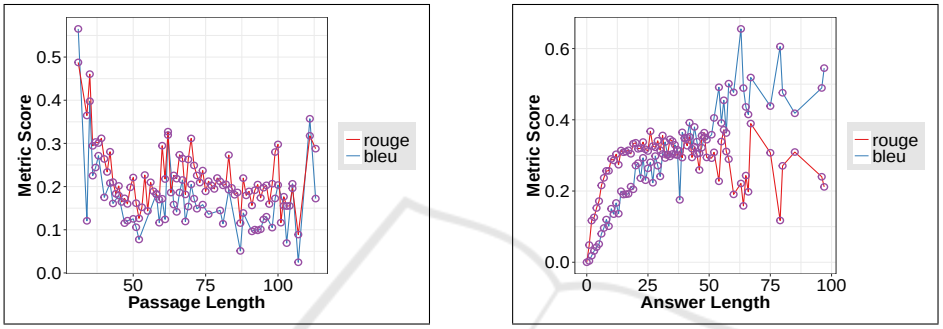


Figure 7: Distribution of development set examples based on top-scoring passage ranking. Shown for each the ROUGE and BLEU score metrics.



(a) Passage Length (b) Answer Length

Figure 8: Development model performance as a function of (a) passage length and (b) answer length. Shown for each the ROUGE and BLEU score metrics.

ally perceived more challenging in question answering systems (Yu et al., 2016), although they represent a fairly small sample of 1.8 percentage points of the dataset queries. On the more simpler 'what' questions that are most prominent with 42.2 percent of queries, we are inline with average system performance.

Next, we tested our hypothesis and contrasted performance impact of ranked web pages for search relevancy with ranked passages for answer prediction. The former was obtained by the Bing search engine and the latter is based entirely on our quality metrics that were produced by multiple coattention engines. We note that for each query, the passage enumeration outlined in the dataset matches the ranked order of the originating web pages. In Figure 7, we present the distribution of our top passage ranking in percentage points of examples, across the range of ten passage enumerations (Table 3). The first enumerated passage also ranks as our top scorer for the highest share of examples, however it only claims a little over eight percent of the queries. Stated differently, the quality of answers drawn from the first passage is likely to decline for about 92% of the examples. Similarly, the succeeding top-ranked page enumerations appear to drop moderately in their example proportion, and thus we expect for them a slightly larger performance fall off. This markedly supports our hypothesis that MC performance might be impacted unfavorably for explicit out-of-domain context order. We note that distribution percentages do not add up to one hundred, since passages with a zero score assume no ranking.

Figure 8 shows our model performance as a function of (a) context passage length and (b) answer length. Apart from spiky performance behavior apparent for passages shorter than 25 words, ROUGE and BLEU scores follow a trend of relative constant performance for passages of up to 100 words, and jagged again for larger context. This observation concurs with the histogram of Figure 4(c) that highlights only a few samples of passage length under 25 words or larger than 100 words. Our consistent performance over increased passage lengths suggests an appreciable signal-to-noise ratio for determining an answer span. On the other hand, our model performs poorly on extremely short-text answers of up to a handful of words, mostly attributed to a compromised context, and then the rate climbs precipitously till about 25-word length, after which it levels off for ROUGE and increases moderately for BLEU. Table 7 supplements the graphic plots of Figure 8 with a statistical summary and shows the average ROUGE results over either passage length or answer length are more con-

Table 7: Summary statistics of development model performance as a function of passage length and answer length. Shown for each the ROUGE and BLEU scoring metrics.

	Passage Length			Answer Length		
	Min	Max	Mean	Min	Max	Mean
ROUGE	0.09	0.49	0.24	0.05	0.39	0.28
BLEU	0.02	0.56	0.17	0.01	0.66	0.29

sistent than the BLEU measures. However, the maximal score favors the BLEU metric over both passage and answer length parameters, with an uncharacteristic performance disparity of 0.27 over answer length.

MC systems that commonly use a single predicted span of words for quality rating, often fall short in rendering non-contiguous answers. Notably for the MSMARCO dataset, non-contiguous answers take up a sizable proportion of about seventeen percent of the examples (Table 5). To address this shortcoming, our model retains predicted spans, $S^{(i)}$, from all the query-bound passages, and that facilitates a multi-span answer mode in our evaluation layer (Figure 1). A user settable hyperparameter controls the number of spans to expand each to words and further concatenate all words from multiple spans to a single text sequence. The sequence is then paired with the reference answers, provided by the dataset, to commence our LCS evaluation. In Table 8, we compare the performance of our GRU-based model to both a previously published LSTM-based architecture and a recent native BiDAF implementation by the MSMARCO development team, all evaluated on the development set. Our performance for non-contiguous answers improves system quality by about nine percentage points as the number of predicted spans increases from one to two, and then tapers off with a diminishing performance return for three and four spans. We expected this behavior as top ranked passages in single span mode are selected first for the multi-span method, and hence the lesser contribution from lower ranked passages.

At the time of this paper submission, our top scores on the dataset development set, ROUGE of 23.54 and BLEU of 15.25, would rank seventh on MSMARCO V1.1 leaderboard [1] for test-set evaluated models that are identified with published literature links. Our model performs slightly over a native BiDAF baseline with an average ROUGE of 22.56.

Table 8: ROUGE and BLEU percentage scores for human judges and neural models evaluated on MSMARCO development set. Our performance is shown for a varying number of predicted spans to address non-contiguous answers.

Model	Spans	ROUGE	BLEU
Human interpreted	NA	47.00	46.00
GRU-based (our work)	1	21.02	13.63
	2	22.91	14.84
	3	23.33	15.12
	4	**23.54**	**15.25**
Native BiDAF baseline (MSMARCO team, 2018)	1	22.56	10.96
LSTM-based (Higgins and Nho, 2017)	1	12.08	9.30

5 CONCLUSIONS

In this work, we introduced an effective MC neural-model to regulate quality ranking of answers drawn from multiple context passages. Primarily motivated by the more realistic MSMARCO dataset, our model combines multiple coattention spans to seamlessly address answers that straddle passage bounds. A task proven both challenging and almost impractical to supervise in state-of-the-art MC systems that only support a single span. One of the key insights into our study is the considerable performance gains achieved when using our coattention-based passage ranking, compared to applying directly the query relevance order of the originated web-document pages, as presented in the dataset. Our model performance appears to lag behind the rating of average human comprehension on MSMARCO, but despite its simplicity is ahead of a leaderboard top-performing model that was trained on the SQuAD dataset.

Aiming in this work for a decent but not top performance, one topic for further study would be to explore bidirectional GRU networks in the contextual layer to improve our system performance. A more direct progression of our work is to evolve into a task that handles efficiently answers of extremely short-text sequences, which are potentially unsearchable in the corresponding context. One of unanswered questions in our work that we plan to investigate is a generalized approach that satisfies non-contiguous answers confined to a single passage. We look forward to adapt our model to the recently introduced TriviaQA dataset that incorporates multiple evidence documents per-query as well, however each comprises a much increased search space of several thousand words. The above intuitions are generally purposed

[1] http://www.msmarco.org/leaders.aspx

to benefit an improved understanding of how to mitigate computational complexity in an MC engine. Finally, we plan to experiment with MSMARCO V2.1 that was just released and comprises over one million user queries. This version allows to explore additional evaluation metrics to overcome semantic-equivalence limitations of ROUGE and BLEU.

ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their insightful suggestions and feedback.

REFERENCES

Andress, J. and Zanoci, C. (2017). Coattention-based neural network for question answering. Technical report, Stanford University. http://stanford.edu/class/cs224n/reports/2762015.pdf.

Chen, Z., Yang, R., Cao, B., Zhao, Z., Cai, D., and He, X. (2017). Smarnet: Teaching machines to read and comprehend like human. *CoRR*, abs/1710.02772. http://arxiv.org/abs/1710.02772.

Cho, K., van Merrienboer, B., Bahdanau, D., and Bengio, Y. (2014). On the properties of neural machine translation: Encoder-decoder approaches. *CoRR*, abs/1409.1259. http://arxiv.org/abs/1409.1259.

Chung, J., Gülçehre, c., Cho, K., and Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555. http://arxiv.org/abs/1412.3555.

Cui, Y., Chen, Z., Wei, S., Wang, S., Liu, T., and Hu, G. (2016). Attention-over-attention neural networks for reading comprehension. *CoRR*, abs/1607.04423. http://arxiv.org/abs/1607.04423.

Dhingra, B., Liu, H., Cohen, W. W., and Salakhutdinov, R. (2017). Gated-attention readers for text comprehension. *CoRR*, abs/1606.01549. http://arxiv.org/abs/1606.01549.

Glorot, X. and Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *Artificial Intelligence and Statistics*, pages 249–256, Sardinia, Italy.

Higgins, B. and Nho, E. (2017). LSTM encoder-decoder architecture with attention mechanism for machine comprehension. Technical report, Stanford University. http://stanford.edu/class/cs224n/reports/2784756.pdf.

Hoang, C. D. V., Cohn, T., and Haffari, G. (2016). Incorporating side information into recurrent neural network language models. In *Human Language Technologies: North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 1250–1255, San Diego, California.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Joshi, M., Choi, E., Weld, D., and Zettlemoyer, L. (2017). TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1601–1611, Vancouver, Canada.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980. http://arxiv.org/abs/1412.6980.

Lee, K., Kwiatkowski, T., Parikh, A. P., and Das, D. (2016). Learning recurrent span representations for extractive question answering. *CoRR*, abs/1611.01436. http://arxiv.org/abs/1611.01436.

Lin, C. (2004). ROUGE: a package for automatic evaluation of summaries. In *Workshop on Text Summarization Branches Out of the Association for Computational Linguistics (ACL)*.

Luong, M., Pham, H., and Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. *CoRR*, abs/1508.04025. http://arxiv.org/abs/1508.04025.

Mikolov, T., Chen, K., Corrado, G. S., and Dean, J. (2013a). Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781. http://arxiv.org/abs/1301.3781.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3111–3119. Curran Associates, Inc., Red Hook, NY.

Nguyen, T., Rosenberg, M., Song, X., Gao, J., Tiwary, S., Majumder, R., and Deng, L. (2016). MS MARCO: A human generated MAchine Reading COmprehension dataset. *CoRR*, abs/1611.09268. http://arxiv.org/abs/1611.09268.

Papineni, K., Roukos, S., Ward, T., and jing Zhu, W. (2002). BLEU: a method for automatic evaluation of machine translation. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Pennington, J., Socher, R., and Manning, C. D. (2014). GloVe: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar.

R Core Team (2013). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. http://www.R-project.org/.

Raiman, J. and Miller, J. (2017). Globally normalized reader. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1059–1069, Copenhagen, Denmark.

Rajpurkar, P., Zhang, J., Lopyrev, K., and Liang, P. (2016). SQuAD: 100,000+ questions for machine comprehension of text. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 2383–2392, Austin, Texas.

Seo, M. J., Kembhavi, A., Farhadi, A., and Hajishirzi, H. (2017). Bidirectional attention flow for machine comprehension. *CoRR*, abs/1611.01603. http://arxiv.org/abs/1611.01603.

Shen, Y., Huang, P., Gao, J., and Chen, W. (2016). ReasoNet: Learning to stop reading in machine comprehension. *CoRR*, abs/1609.05284. http://arxiv.org/abs/1609.05284,.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Machine Learning Research*, 15(1):1929–1958.

Sutskever, I., Martens, J., and Hinton, G. (2011). Generating text with recurrent neural networks. In *International Conference on Machine Learning (ICML)*, pages 1017–1024, Bellevue, Washington.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3104–3112. Curran Associates, Inc., Red Hook, NY.

Tan, C., Wei, F., Yang, N., Lv, W., and Zhou, M. (2017). S-net: From answer extraction to answer generation for machine reading comprehension. *CoRR*, abs/1706.04815. http://arxiv.org/abs/1706.04815,.

Tran, K. M., Bisazza, A., and Monz, C. (2016). Recurrent memory networks for language modeling. In *Human Language Technologies: North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 321–331, San Diego, California.

Trischler, A., Wang, T., Yuan, X., Harris, J., Sordoni, A., Bachman, P., and Suleman, K. (2016). Newsqa: A machine comprehension dataset. *CoRR*, abs/1611.09830. http://arxiv.org/abs/1611.09830.

Wang, S. and Jiang, J. (2016). Machine comprehension using Match-LSTM and answer pointer. *CoRR*, abs/1608.07905. http://arxiv.org/abs/1608.07905.

Wang, W., Yang, N., Wei, F., Chang, B., and Zhou, M. (2017). Gated self-matching networks for reading comprehension and question answering. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 189–198, Vancouver, Canada.

Weissenborn, D., Wiese, G., and Seiffe, L. (2017). Fastqa: A simple and efficient neural architecture for question answering. *CoRR*, abs/1703.04816. http://arxiv.org/abs/1703.04816.

Xiong, C., Zhong, V., and Socher, R. (2017). Dynamic coattention networks for question answering. *CoRR*, abs/1611.01604. http://arxiv.org/abs/1611.01604.

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., and Bengio, Y. (2015). Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning (ICML)*, pages 2048–2057, Lille, France.

Yu, Y., Zhang, W., Hasan, K. S., Yu, M., Xiang, B., and Zhou, B. (2016). End-to-end reading comprehension with dynamic answer chunk ranking. *CoRR*, abs/1610.09996. http://arxiv.org/abs/1610.09996.

APPENDIX

Table 9 shows a sample of an MC primitive in text word space drawn from the MSMARCO dataset and consists of a query, query type, six context passages, and a single reference answer. The bold text span in the first context passage matches the gold answer.

Table 9: An entry sample of the MSMARCO dataset consisting of a query, query type, six context passages, and a single reference answer.

Query	walgreens store sales average
Query Type	numeric
Passages	1. the average walgreens salary ranges from **approximately 15000 per year** for customer service associate cashier to 179900 per year for district manager average walgreens hourly pay ranges from approximately 735 per hour for laboratory technician to 6890 per hour for pharmacy manager salary information comes from 7810 data points collected directly from employees users and jobs on indeed 2. the average revenue in 2011 of a starbuck store was 1078000 up from 1011000 in 2010 the average ticket total purchase at domestic starbuck stores in no vember 2007 was reported at 636 in 2008 the average ticket was flat 00 change 3. in fiscal 2014 walgreens opened a total of 184 new locations and acquired 84 locations for a net decrease of 273 after relocations and closings how big are your stores the average size for a typical walgreens is about 14500 square feet and the sales floor averages about 11000 square feet how do we select locations for new stores there are several factors that walgreens takes into account such as major intersections traffic patterns demographics and locations near hospitals 4. th store in 1984 reaching 4 billion in sales in 1987 and 5 billion two years later walgreens ended the 1980s with 1484 stores 53 billion in revenues and 154 million in profits however profit margins remained just below 3 percent of sales and returns on assets of less than 10 percent 5. the number of walgreen stores has risen from 5000 in 2005 to more than 8000 at present the average square footage per store stood at approximately 10200 and we forecast the figure to remain constant over our review period walgreen earned 303 as average front - end revenue per store square foot in 2012 6. your walgreens store select a store from the search results to make it your walgreens store and save time getting what you need your walgreens store will be the default location for picking up prescriptions photos in store orders and finding deals in the weekly ad
Gold Answer	approximately 15000 per year