A Question-Focused Multi-Factor Attention Network for Question Answering

Souvik Kundu and **Hwee Tou Ng**
Department of Computer Science
National University of Singapore
{souvik, nght}@comp.nus.edu.sg

Abstract

Neural network models recently proposed for question answering (QA) primarily focus on capturing the passage-question relation. However, they have minimal capability to link relevant facts distributed across multiple sentences which is crucial in achieving deeper understanding, such as performing multi-sentence reasoning, co-reference resolution, etc. They also do not explicitly focus on the question and answer type which often plays a critical role in QA. In this paper, we propose a novel end-to-end question-focused multi-factor attention network for answer extraction. Multi-factor attentive encoding using tensor-based transformation aggregates meaningful facts even when they are located in multiple sentences. To implicitly infer the answer type, we also propose a max-attentional question aggregation mechanism to encode a question vector based on the important words in a question. During prediction, we incorporate sequence-level encoding of the first wh-word and its immediately following word as an additional source of question type information. Our proposed model achieves significant improvements over the best prior state-of-the-art results on three large-scale challenging QA datasets, namely NewsQA, TriviaQA, and SearchQA.

Introduction

In machine comprehension-based (MC) question answering (QA), a machine is expected to provide an answer for a given question by understanding texts. In recent years, several MC datasets have been released. **?** (**?**) released a multiple-choice question answering dataset. **?** (**?**) created a large cloze-style dataset using CNN and Daily Mail news articles. Several models (**?**; **?**; **?**; **?**; **?**; **?**) based on neural attentional and pointer networks (**?**) have been proposed since then. **?** (**?**) released the SQuAD dataset where the answers are free-form unlike in the previous MC datasets.

Most of the previously released datasets are closed-world, i.e., the questions and answers are formulated given the text passages. As such, the answer spans can often be extracted by simple word and context matching. **?** (**?**) attempted to alleviate this issue by proposing the NewsQA dataset where the questions are formed only using the CNN article summaries without accessing the full text. As a result, a significant proportion of questions require reasoning beyond simple word matching. Two even more challenging open-world QA datasets, TriviaQA (**?**) and SearchQA (**?**), have recently been released. TriviaQA consists of question-answer pairs authored by trivia enthusiasts and independently gathered evidence documents from Wikipedia as well as Bing Web search. In SearchQA, the question-answer pairs are crawled from the Jeopardy archive and are augmented with text snippets retrieved from Google search.

Recently, many neural models have been proposed (**?**; **?**; **?**; **?**; **?**; **?**; **?**), which mostly focus on passage-question interaction to capture the context similarity for extracting a text span as the answer. However, most of the models do not focus on synthesizing evidence from multiple sentences and fail to perform well on challenging open-world QA tasks such as NewsQA and TriviaQA. Moreover, none of the models explicitly focus on question/answer type information for predicting the answer. In practice, fine-grained understanding of question/answer type plays an important role in QA.

In this work, we propose <u>a</u>n end-to-end question-focused <u>m</u>ulti-factor <u>a</u>ttention <u>n</u>etwork for <u>d</u>ocument-based question <u>a</u>nswering (AMANDA), which learns to aggregate evidence distributed across multiple sentences and identifies the important question words to help extract the answer. Intuitively, AMANDA extracts the answer not only by synthesizing relevant facts from the passage but also by implicitly determining the suitable answer type during prediction. The key contributions of this paper are:

- We propose a multi-factor attentive encoding approach based on tensor transformation to synthesize meaningful evidence across multiple sentences. It is particularly effective when answering a question requires deeper understanding such as multi-sentence reasoning, co-reference resolution, etc.
- To subsume fine-grained answer type information, we propose a max-attentional question aggregation mechanism which learns to identify the meaningful portions of a question. We also incorporate sequence-level representations of the first wh-word and its immediately following word in a question as an additional source of question type information.

Problem Definition

Given a pair of passage and question, an MC system needs to extract a text span from the passage as the answer. We for-

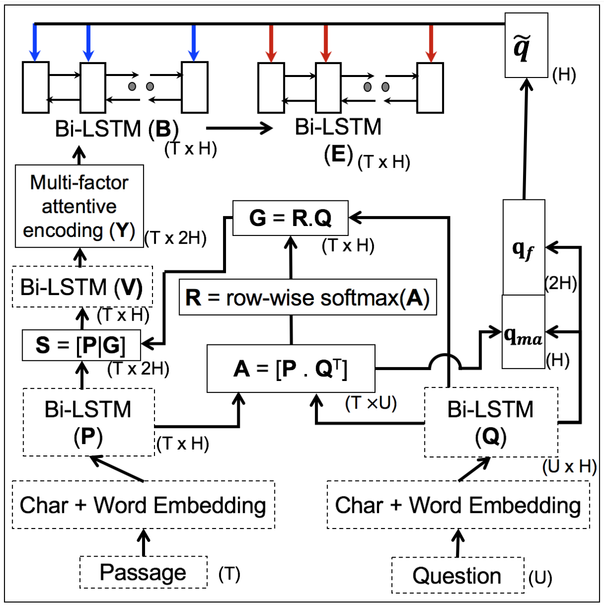


Figure 1: Architecture of the proposed model. Hidden unit representations of Bi-LSTMs, **B** and **E**, are shown to illustrate the answer pointers. Blue and red arrows represent the start and end answer pointers respectively.

mulate the answer as two pointers in the passage, which represent the beginning and ending tokens of the answer. Let $\mathcal{P}$ be a passage with tokens $(\mathcal{P}_1, \mathcal{P}_2, \dots, \mathcal{P}_T)$ and $\mathcal{Q}$ be a question with tokens $(\mathcal{Q}_1, \mathcal{Q}_2, \dots, \mathcal{Q}_U)$, where T and U are the length of the passage and question respectively. To answer the question, a system needs to determine two pointers in the passage, b and e, such that $1 \le b \le e \le T$. The resulting answer tokens will be $(\mathcal{P}_b, \mathcal{P}_{b+1}, \dots, \mathcal{P}_e)$.

Network Architecture

The architecture of the proposed question-focused multi-factor attention network[1] is given in Figure 1.

Word-level Embedding

Word-level embeddings are formed by two components: pre-trained word embedding vectors from GloVe (**?**) and convolutional neural network-based (CNN) character embeddings (**?**). Character embeddings have proven to be very useful for out-of-vocabulary (OOV) words. We use a character-level CNN followed by max-pooling over an entire word to get the embedding vector for each word. Prior to that, a character-based lookup table is used to generate the embedding for every character and the lookup table weights are learned during training. We concatenate these two embedding vectors for every word to generate word-level embeddings.

Sequence-level Encoding

We apply sequence-level encoding to incorporate contextual information. Let $\mathbf{e}_t^p$ and $\mathbf{e}_t^q$ be the tth embedding vectors of the passage and the question respectively. The embedding vectors are fed to a bi-directional LSTM (BiLSTM) (**?**). Considering that the outputs of the BiLSTMs are unfolded across time, we represent the outputs as $\mathbf{P} \in \mathbb{R}^{T \times H}$ and $\mathbf{Q} \in \mathbb{R}^{U \times H}$ for passage and question respectively. H is the number of hidden units for the BiLSTMs. At every time step, the hidden unit representation of the BiLSTMs is obtained by concatenating the hidden unit representations of the corresponding forward and backward LSTMs. For the passage, at time step t, the forward and backward LSTM hidden unit representations can be written as:

$$\begin{aligned} \overrightarrow{\mathbf{h}}_t^p &= \overrightarrow{\text{LSTM}}(\overrightarrow{\mathbf{h}}_{t-1}^p, \mathbf{e}_t^p) \\ \overleftarrow{\mathbf{h}}_t^p &= \overleftarrow{\text{LSTM}}(\overleftarrow{\mathbf{h}}_{t+1}^p, \mathbf{e}_t^p) \end{aligned} \tag{1}$$

The tth row of $\mathbf{P}$ is represented as $\mathbf{p}_t = \overrightarrow{\mathbf{h}}_t^p \,||\, \overleftarrow{\mathbf{h}}_t^p$, where $||$ represents the concatenation of two vectors. Similarly, the sequence level encoding for a question is $\mathbf{q}_t = \overrightarrow{\mathbf{h}}_t^q \,||\, \overleftarrow{\mathbf{h}}_t^q$, where $\mathbf{q}_t$ is the tth row of $\mathbf{Q}$.

Cartesian Similarity-based Attention Layer

The attention matrix is calculated by taking dot products between all possible combinations of sequence-level encoding vectors for a passage and a question. Note that for calculating the attention matrix, we do not introduce any additional learnable parameters. The attention matrix $\mathbf{A} \in \mathbb{R}^{T \times U}$ can be expressed as:

$$\mathbf{A} = \mathbf{P}\,\mathbf{Q}^\top \tag{2}$$

Intuitively, $A_{i,j}$ is a measure of the similarity between the sequence-level encoding vectors of the ith passage word and the jth question word.

Question-dependent Passage Encoding

In this step, we jointly encode the passage and question. We apply a row-wise softmax function on the attention matrix:

$$\mathbf{R} = \text{row-wise softmax}(\mathbf{A}) \tag{3}$$

If $\mathbf{r}_t \in \mathbb{R}^U$ is the tth row of $\mathbf{R} \in \mathbb{R}^{T \times U}$, then $\sum_{j=1}^{U} r_{t,j} = 1$. Each row of $\mathbf{R}$ measures how relevant every question word is with respect to a given passage word. Next, an aggregated question vector is computed corresponding to each sequence-level passage word encoding vector. The aggregated question vector $\mathbf{g}_t \in \mathbb{R}^H$ corresponding to the tth passage word is computed as $\mathbf{g}_t = \mathbf{r}_t\mathbf{Q}$. The aggregated question vectors corresponding to all the passage words can be computed as $\mathbf{G} = \mathbf{R}\ \mathbf{Q}$, where $\mathbf{g}_t$ is the tth row of $\mathbf{G} \in \mathbb{R}^{T \times H}$.

The aggregated question vectors corresponding to the passage words are then concatenated with the sequence-level passage word encoding vectors. If the question-dependent passage encoding is denoted as $\mathbf{S} \in \mathbb{R}^{T \times 2H}$ and $\mathbf{s}_t$ is the tth row of $\mathbf{S}$, then $\mathbf{s}_t = \mathbf{c}_t \,||\, \mathbf{g}_t$, where $\mathbf{c}_t$ is the sequence-level encoding vector of the tth passage word (tth row of $\mathbf{P}$). Then a BiLSTM is applied on $\mathbf{S}$ to obtain $\mathbf{V} \in \mathbb{R}^{T \times H}$.

[1]Our code is available at `https://github.com/nusnlp/amanda`

Multi-factor Attentive Encoding

Tensor-based neural network approaches have been used in a variety of natural language processing tasks (?; ?). We propose a multi-factor attentive encoding approach using tensor-based transformation. In practice, recurrent neural networks fail to remember information when the context is long. Our proposed multi-factor attentive encoding approach helps to aggregate meaningful information from a long context with fine-grained inference due to the use of multiple factors while calculating attention.

Let $\mathbf{v}_i \in \mathbb{R}^H$ and $\mathbf{v}_j \in \mathbb{R}^H$ represent the question-dependent passage vectors of the ith and jth word, i.e., the ith and jth row of $\mathbf{V}$. Tensor-based transformation for multi-factor attention is formulated as follows:

$$\mathbf{f}^m_{i,j} = \mathbf{v}_i \, \mathbf{W}^{[1:m]}_f \, \mathbf{v}_j^\top \ , \tag{4}$$

where $\mathbf{W}^{[1:m]}_f \in \mathbb{R}^{H \times m \times H}$ is a 3-way tensor and m is the number of factors. The output of the tensor product $\mathbf{f}^m_{i,j} \in \mathbb{R}^m$ is a vector where each element $f^m_{i,j,k}$ is a result of the bilinear form defined by each tensor slice $\mathbf{W}^{[k]}_f \in \mathbb{R}^{H \times H}$:

$$f^m_{i,j,k} = \mathbf{v}_i \, \mathbf{W}^{[k]}_f \, \mathbf{v}_j^\top = \sum_{a,b} v_{i,a} W^{[k]}_{f_{a,b}} v_{j,b} \tag{5}$$

$\forall i, j \in [1, T]$, the multi-factor attention tensor can be given as $\mathbf{F}^{[1:m]} \in \mathbb{R}^{m \times T \times T}$. For every vector $\mathbf{f}^m_{i,j}$ of $\mathbf{F}^{[1:m]}$, we perform a max pooling operation over all the elements to obtain the resulting attention value:

$$F_{i,j} = \max(\mathbf{f}^m_{i,j}) \ , \tag{6}$$

where $F_{i,j}$ represents the element in the ith row and jth column of $\mathbf{F} \in \mathbb{R}^{T \times T}$. Each row of $\mathbf{F}$ measures how relevant every passage word is with respect to a given question-dependent passage encoding of a word. We apply a row-wise softmax function on $\mathbf{F}$ to normalize the attention weights, obtaining $\tilde{\mathbf{F}} \in \mathbb{R}^{T \times T}$. Next, an aggregated multi-factor attentive encoding vector is computed corresponding to each question-dependent passage word encoding vector. The aggregated vectors corresponding to all the passage words, $\mathbf{M} \in \mathbb{R}^{T \times H}$, can be given as $\mathbf{M} = \tilde{\mathbf{F}} \ \mathbf{V}$. The aggregated multi-factor attentive encoding vectors are concatenated with the question-dependent passage word encoding vectors to obtain $\tilde{\mathbf{M}} \in \mathbb{R}^{T \times 2H}$. To control the impact of $\tilde{\mathbf{M}}$, we apply a feed-forward neural network-based gating method to obtain $\mathbf{Y} \in \mathbb{R}^{T \times 2H}$. If the tth row of $\tilde{\mathbf{M}}$ is $\tilde{\mathbf{m}}_t$, then the tth row of $\mathbf{Y}$ is:

$$\mathbf{y}_t = \tilde{\mathbf{m}}_t \odot \text{sigmoid}(\tilde{\mathbf{m}}_t \mathbf{W}^g + \mathbf{b}^g) \ , \tag{7}$$

where $\odot$ represents element-wise multiplication. $\mathbf{W}^g \in \mathbb{R}^{2H \times 2H}$ and $\mathbf{b}^g \in \mathbb{R}^{2H}$ are the transformation matrix and bias vector respectively.

We use another pair of stacked BiLSTMs on top of $\mathbf{Y}$ to determine the beginning and ending pointers. Let the hidden unit representations of these two BiLSTMs be $\mathbf{B} \in \mathbb{R}^{T \times H}$ and $\mathbf{E} \in \mathbb{R}^{T \times H}$. To incorporate the dependency of the ending pointer on the beginning pointer, the hidden unit representation of $\mathbf{B}$ is used as input to $\mathbf{E}$.

Question-focused Attentional Pointing

Unlike previous approaches, our proposed model does not predict the answer pointers directly from contextual passage encoding or use another decoder for generating the pointers. We formulate a question representation based on two parts:

- max-attentional question aggregation ($\mathbf{q}_{ma}$)
- question type representation ($\mathbf{q}_f$)

$\mathbf{q}_{ma}$ is formulated by using the attention matrix $\mathbf{A}$ and the sequence-level question encoding $\mathbf{Q}$. We apply a maxcol operation on $\mathbf{A}$ which forms a row vector whose elements are the maximum of the corresponding columns of $\mathbf{A}$. We define $\mathbf{k} \in \mathbb{R}^U$ as the normalized max-attentional weights:

$$\mathbf{k} = \text{softmax}(\text{maxcol}(\mathbf{A})) \tag{8}$$

where softmax is used for normalization. The max-attentional question representation $\mathbf{q}_{ma} \in \mathbb{R}^H$ is:

$$\mathbf{q}_{ma} = \mathbf{k} \, \mathbf{Q} \tag{9}$$

Intuitively, $\mathbf{q}_{ma}$ aggregates the most relevant parts of the question with respect to all the words in the passage.

$\mathbf{q}_f$ is the vector concatenation of the representations of the first wh-word and its following word from the sequence-level question encoding $\mathbf{Q}$. The set of wh-words we used is $\{$*what, who, how, when, which, where, why*$\}$. If $\mathbf{q}_{t_{wh}}$ and $\mathbf{q}_{t_{wh}+1}$ represent the first wh-word and its following word (i.e., the t_{wh}th and $(t_{wh}+1)$th rows of $\mathbf{Q}$), $\mathbf{q}_f \in \mathbb{R}^{2H}$ is expressed as:

$$\mathbf{q}_f \ = \ \mathbf{q}_{t_{wh}} \,||\, \mathbf{q}_{t_{wh}+1} \tag{10}$$

The final question representation $\tilde{\mathbf{q}} \in \mathbb{R}^H$ is expressed as:

$$\tilde{\mathbf{q}} = \tanh((\mathbf{q}_{ma} \,||\, \mathbf{q}_f)\mathbf{W}_q \ + \ \mathbf{b}_q) \tag{11}$$

where $\mathbf{W}_q \in \mathbb{R}^{3H \times H}$ and $\mathbf{b}_q \in \mathbb{R}^H$ are the weight matrix and bias vector respectively. If no wh-word is present in a question, we use the first two sequence-level question word representations for calculating $\tilde{\mathbf{q}}$.

We measure the similarity between $\tilde{\mathbf{q}}$ and the contextual encoding vectors in $\mathbf{B}$ and $\mathbf{E}$ to determine the beginning and ending answer pointers. Corresponding similarity vectors $\mathbf{s}_b \in \mathbb{R}^T$ and $\mathbf{s}_e \in \mathbb{R}^T$ are computed as:

$$\mathbf{s}_b = \tilde{\mathbf{q}} \, \mathbf{B}^\top \ , \ \mathbf{s}_e = \tilde{\mathbf{q}} \, \mathbf{E}^\top \tag{12}$$

The probability distributions for the beginning pointer b and the ending pointer e for a given passage $\mathcal{P}$ and a question $\mathcal{Q}$ can be given as:

$$\begin{aligned} \Pr(b \mid \mathcal{P}, \mathcal{Q}) &= \text{softmax}(\mathbf{s}_b) \\ \Pr(e \mid \mathcal{P}, \mathcal{Q}, b) &= \text{softmax}(\mathbf{s}_e) \end{aligned} \tag{13}$$

The joint probability distribution for obtaining the answer a is given as:

$$\Pr(a \mid \mathcal{P}, \mathcal{Q}) = \Pr(b \mid \mathcal{P}, \mathcal{Q}) \Pr(e \mid \mathcal{P}, \mathcal{Q}, b) \tag{14}$$

To train our model, we minimize the cross entropy loss:

$$\text{loss} = -\sum \log \ \Pr(a \mid \mathcal{P}, \mathcal{Q}) \tag{15}$$

summing over all training instances. During prediction, we select the locations in the passage for which the product of $\Pr(b)$ and $\Pr(e)$ is maximum keeping the constraint $1 \leq b \leq e \leq T$.

Passage: ... The family of a Korean-American missionary believed held in North Korea said Tuesday they are working with U.S. officials to get him returned home. Robert Park told relatives before Christmas that he was trying to sneak into the isolated communist state to bring a message of "Christ's love and forgiveness" to North Korean leader Kim ...
Question: What is the name of the Korean-American missionary?
Reference Answer: Robert Park

Table 1: Example of a (passage, question, answer)

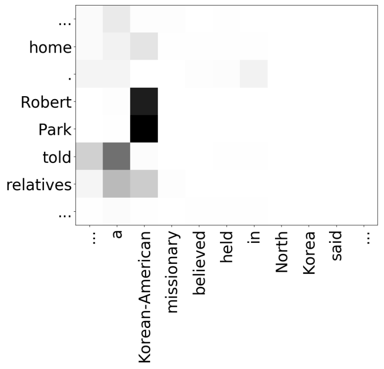


Figure 2: Multi-factor attention weights (darker regions signify higher weights).

Visualization

To understand how the proposed model works, for the example given in Table 1, we visualize the normalized multi-factor attention weights $\tilde{\mathbf{F}}$ and the attention weights $\mathbf{k}$ which are used for max-attentional question aggregation.

In Figure 2, a small portion of $\tilde{\mathbf{F}}$ has been shown, in which the answer words *Robert* and *Park* are both assigned higher weights when paired with the context word *Korean-American*. Due to the use of multi-factor attention, the answer segment pays more attention to the important keyword although it is quite far in the context passage and thus effectively infers the correct answer by deeper understanding. In Figure 3, it is clear that the important question word *name* is getting a higher weight than the other question words. This helps to infer the fine-grained answer type during prediction, i.e., a person's name in this example.

Experiments

We evaluated AMANDA on three challenging QA datasets: NewsQA, TriviaQA, and SearchQA. Using the NewsQA development set as a benchmark, we perform rigorous analysis for better understanding of how our proposed model works.

Datasets

The NewsQA dataset (?) consists of around 100K answerable questions in total. Similar to (?; ?), we do not consider the unanswerable questions in our experiments. NewsQA is more challenging compared to the previously released

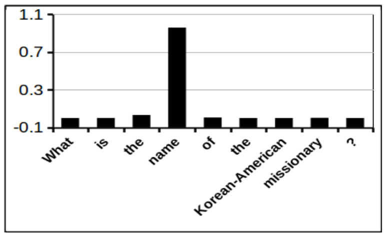


Figure 3: Max-attentional weights for question (the origin is set to -0.1 for clarity).

Model	**Dev**		**Test**	
	EM	**F1**	**EM**	**F1**
(?)				
Match-LSTM	34.4	49.6	34.9	50.0
BARB	36.1	49.6	34.1	48.2
(?) BiDAF on NewsQA	-	-	37.1	52.3
†Neural BoW Baseline	25.8	37.6	24.1	36.6
†FastQA	43.7	56.4	41.9	55.7
†FastQAExt	43.7	56.1	42.8	56.1
(?) R2-BiLSTM	-	-	43.7	56.7
AMANDA	**48.4**	**63.3**	**48.4**	**63.7**

Table 2: Results on the NewsQA dataset. † denotes the models of (?).

datasets as a significant proportion of questions requires reasoning beyond simple word- and context-matching. This is due to the fact that the questions in NewsQA were formulated only based on summaries without accessing the main text of the articles. Moreover, NewsQA passages are significantly longer (average length of 616 words) and cover a wider range of topics.

TriviaQA (?) consists of question-answer pairs authored by trivia enthusiasts and independently gathered evidence documents from Wikipedia and Bing Web search. This makes the task more similar to real-life IR-style QA. In total, the dataset consists of over 650K question-answer-evidence triples. Due to the high redundancy in Web search results (around 6 documents per question), each question-answer-evidence triple is treated as a separate sample and evaluation is performed at document level. However, in Wikipedia, questions are not repeated (each question has 1.8 evidence documents) and evaluation is performed over questions. In addition to distant supervision, TriviaQA also has a verified human-annotated question-evidence collection. Compared to previous datasets, TriviaQA has more complex compositional questions which require greater multi-sentence reasoning.

SearchQA (?) is also constructed to more closely reflect IR-style QA. They first collected existing question-answer pairs from a Jeopardy archive and augmented them with text snippets retrieved by Google. One difference with TriviaQA is that the evidence passages in SearchQA are Google snippets instead of Wikipedia or Web search documents. This

Model	Domain	Distant Supervision				Verified			
		Dev		Test		Dev		Test	
		EM	F1	EM	F1	EM	F1	EM	F1
‡Random	Wiki	12.72	22.91	12.74	22.35	14.81	23.31	15.41	25.44
‡Classifier		23.42	27.68	22.45	26.52	24.91	29.43	27.23	31.37
‡BiDAF		40.26	45.74	40.32	45.91	47.47	53.70	44.86	50.71
⋆MEMEN		43.16	46.90	-	-	49.28	55.83	-	-
AMANDA		**46.95**	**52.51**	**46.67**	**52.22**	**52.86**	**58.74**	**50.51**	**55.93**
‡Classifier	Web	24.64	29.08	24.00	28.38	27.38	31.91	30.17	34.67
‡BiDAF		41.08	47.40	40.74	47.05	51.38	55.47	49.54	55.80
⋆MEMEN		44.25	48.34	-	-	53.27	57.64	-	-
AMANDA		**46.68**	**53.27**	**46.58**	**53.13**	**60.31**	**64.90**	**55.14**	**62.88**

Table 3: Results on the TriviaQA dataset. ‡(**?**), ⋆(**?**)

Model	Set	Unigram Accuracy	N-gram F1
(**?**)			
TF-IDF Max	Dev	13.0	-
	Test	12.7	-
ASR	Dev	43.9	24.2
	Test	41.3	22.8
AMANDA	Dev	**48.6**	**57.7**
	Test	**46.8**	**56.6**

Table 4: Results on the SearchQA dataset.

Model	EM	F1
minus multi factor attn.	46.4	61.2
minus $\mathbf{q}_{ma}$ and $\mathbf{q}_f$	46.2	60.5
minus $\mathbf{q}_{ma}$	46.6	61.3
minus $\mathbf{q}_f$	46.8	61.8
AMANDA	**48.4**	**63.3**

Table 5: Ablation of proposed components on the NewsQA development set.

makes reasoning more challenging as the snippets are often very noisy. SearchQA consists of 140,461 question-answer pairs, where each pair has 49.6 snippets on average and each snippet has 37.3 tokens on average.

Experimental Settings

We tokenize the corpora with NLTK[2]. We use the 300-dimension pre-trained word vectors from GloVe (**?**) and we do not update them during training. The out-of-vocabulary words are initialized with zero vectors. We use 50-dimension character-level embedding vectors. The number of hidden units in all the LSTMs is 150. We use dropout (**?**) with probability 0.3 for every learnable layer. For multi-factor attentive encoding, we choose 4 factors (m) based on our experimental findings (refer to Table 7). During training, the mini-batch size is fixed at 60. We use the Adam optimizer (**?**) with learning rate of 0.001 and clipnorm of 5. During testing, we enforce the constraint that the ending pointer will always be equal to or greater than the beginning pointer. We use exact match (EM) and F1 scores as the evaluation metrics.

Results

Table **??** shows that AMANDA outperforms all the state-of-the-art models by a significant margin on the NewsQA dataset. Table **??** shows the results on the TriviaQA dataset. In Table **??**, the model named Classifier based on feature engineering was proposed by **?** (**?**). They also reported the performance of BiDAF (**?**). A memory network-based approach, MEMEN, was recently proposed by (**?**). Note that in the Wikipedia domain, we choose the answer which provides the highest maximum joint probability (according to Eq. (14)) for any document. Table **??** shows that AMANDA achieves state-of-the-art results in both Wikipedia and Web domain on distantly supervised and verified data.

Results on the SearchQA dataset are shown in Table 4. In addition to a TF-IDF approach, **?** (**?**) modified and reported the performance of attention sum reader (ASR) which was originally proposed by **?** (**?**). We consider a maximum of 150 words surrounding the answer from the concatenated ranked list of snippets as a passage to more quickly train the model and to reduce the amount of noisy information. During prediction, we choose the first 200 words (about 5 snippets) from the concatenated ranked list of snippets as an evidence passage. These are chosen based on performance on the development set. Based on question patterns, question types are always represented by the first two sequence-level representations of question words. To make the results comparable, we also report accuracy for single-word-answer (unigram) questions and F1 score for multi-word-answer (n-gram) questions. AMANDA outperforms both systems, especially for multi-word-answer questions by a huge margin. This indicates that AMANDA can learn to make inference reasonably well even if the evidence passages are noisy.

Effectiveness of the Model Components

Table 5 shows that AMANDA performs better than any of the ablated models which include the ablation of multi-factor attentive encoding, max-attentional question aggrega-

[2]http://www.nltk.org/

Model	EM	F1
minus char embedding	47.5	61.4
minus question-dependent passage enc.	32.1	45.0
minus 2nd LSTM during prediction	46.5	61.6
AMANDA	**48.4**	**63.3**

Table 6: Ablation of other components on the NewsQA development set

tion ($\mathbf{q}_{ma}$), and question type representation ($\mathbf{q}_f$). We also perform statistical significance test using paired t-test and bootstrap resampling. Performance of AMANDA (both in terms of EM and F1) is significantly better ($p < 0.01$) than the ablated models.

One of the key contributions of this paper is multi-factor attentive encoding which aggregates information from the relevant passage words by using a tensor-based attention mechanism. The use of multiple factors helps to fine-tune answer inference by synthesizing information distributed across multiple sentences. The number of factors is the granularity to which the model is allowed to refine the evidence. The effect of multi-factor attentive encoding is illustrated by the following example taken from the NewsQA development set:

What will allow storage on remote servers?

*...The **iCloud service** will now be integrated into the iOS 5 operating system. It will work with apps and allow content to be stored on remote servers instead of the users' iPod, iPhone or other device...*

When multi-factor attentive encoding is ablated, the model could not figure out the cross-sentence co-reference and wrongly predicted the answer as *apps*. On the contrary, with multi-factor attentive encoding, AMANDA could correctly infer the answer as *iCloud service*.

Another contribution of this work is to include the question focus during prediction. It is performed by adding two components: $\mathbf{q}_{ma}$ (max-attentional question aggregation) and $\mathbf{q}_f$ (question type representation). $\mathbf{q}_{ma}$ and $\mathbf{q}_f$ implicitly infer the answer type during prediction by focusing on the important question words. Impact of the question focus components is illustrated by the following example taken from the NewsQA development set:

who speaks on Holocaust remembrance day?

*... Israel's vice prime minister **Silvan Shalom** said Tuesday "Israel can never ... people just 65 years ago" ... He was speaking as Israel observes its Holocaust memorial day, remembering the roughly...*

Without the $\mathbf{q}_{ma}$ and $\mathbf{q}_f$ components, the answer was wrongly predicted as *Israel*, whereas with $\mathbf{q}_{ma}$ and $\mathbf{q}_f$, AMANDA could correctly infer the answer type (i.e., a person's name) and predict *Silvan Shalom* as the answer.

Ablation studies of other components such as character embedding, question-dependent passage encoding, and the second LSTM during prediction are given in Table 6. When the second LSTM ($\mathbf{E}$) is ablated, a feed-forward layer is used instead. Table 6 shows that question-dependent passage encoding has the highest impact on performance.

Value of m	1	2	3	4	5
EM	45.8	47.4	**48.7**	48.4	48.0
F1	61.2	61.9	62.9	**63.3**	62.5

Table 7: Variation of m on the NewsQA development set.

Aggregation	EM	F1
Mean	46.6	61.3
Sum	47.9	62.2
Max (AMANDA)	**48.4**	**63.3**

Table 8: Variation of question aggregation formulation on the NewsQA development set.

Variation on the number of factors (m) and $\mathbf{q}_{ma}$

Table 7 shows the performance of AMANDA for different values of m. We use 4 factors for all the experiments as it gives the highest F1 score. Note that $m = 1$ is equivalent to standard bilinear attention.

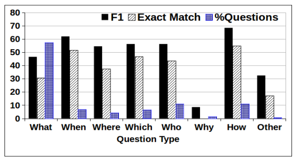
Table 8 shows the variation of question aggregation formulation. For mean aggregation, the attentional weight vector $\mathbf{k}$ is formulated by applying column-wise averaging on the attention matrix $\mathbf{A}$. Intuitively, it is giving equal priority to all the passage words to determine a particular question word attention. Similarly, in the case of sum aggregation, we apply a column-wise sum operation. Table 8 shows that the best performance is obtained when $\mathbf{q}_{ma}$ is obtained with a column-wise maximum operation on $\mathbf{A}$. Effectively, it is helping to give higher weights to the more important question words based on the most relevant passage words.

Quantitative Error Analysis

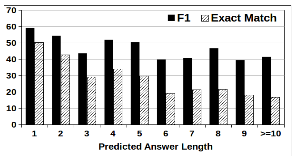
We analyzed the performance of AMANDA across different question types and different predicted answer lengths. Figure 4(a) shows that it performs poorly on *why* and *other* questions whose answers are usually longer. Figure 4(b) supports this fact as well. When the predicted answer length increases, both F1 and EM start to degrade. The gap between F1 and EM also increases for longer answers. This is because for longer answers, the model is not able to decide the exact boundaries (low EM score) but manages to predict some correct words which partially overlap with the reference answer (relatively higher F1 score).

Qualitative Error Analysis

On the NewsQA development set, AMANDA predicted completely wrong answers on 25.1% of the questions. We randomly picked 50 such questions for analysis. The observed types of errors are given in Table 9 with examples. 42% of the errors are due to answer ambiguities, i.e., no unique answer is present. 22% of the errors are due to mismatch between question and context words. 10% of the errors are due to the need for highly complex inference. 6% of the errors occur due to paraphrasing, i.e., the question is posed with different words which do not appear in the passage context. The remaining 20% of the errors are due to insufficient evidence, incorrect tokenization, wrong coreference resolution, etc.



(a)



(b)

Figure 4: (a) Results for different question types. (b) Results for different predicted answer lengths.

Answer ambiguity (42%) Q: What happens to the power supply? ... customers possible.” The outages were due mostly to power **lines downed** by Saturday’s hurricane-force winds, which knocked over trees and utility poles. At ...
Context mismatch (22%) Q: Who was Kandi Burrus’s fiance? Kandi Burruss, the newest cast member of the reality show “The Real Housewives of Atlanta” ... fiance, who died ... fiance, 34-year-old **Ashley “A.J.” Jewell**, also...
Complex inference (10%) Q: When did the Delta Queen first serve? ... the Delta Queen steamboat, a floating National ... scheduled voyage this week ... The Delta Queen will go ... Supporters of the boat, which has roamed the nation’s waterways since **1927** and helped the Navy ...
Paraphrasing issues (6%) Q: What was Ralph Lauren’s first job? Ralph Lauren has ... Japan. For four ... than the former **tie salesman** from the Bronx. “Those ties ... Lauren originally named his company Polo because ...

Table 9: Examples of different error types and their percentages. Ground truth answers are bold-faced and predicted answers are underlined.

Related Work

Recently, several neural network-based models have been proposed for QA. Models based on the idea of chunking and ranking include **?** (**?**) and **?** (**?**). **?** (**?**) used a fine-grained gating mechanism to capture the correlation between a passage and a question. **?** (**?**) used a Match-LSTM to encode the question and passage together and a boundary model determined the beginning and ending boundary of an answer. **?** (**?**) reimplemented Match-LSTM for the NewsQA dataset and proposed a faster version of it. **?** (**?**) used a co-attentive encoder followed by a dynamic decoder for iteratively estimating the boundary pointers. **?** (**?**) proposed a bi-directional attention flow approach to capture the interactions between passages and questions. **?** (**?**) proposed a simple context matching-based neural encoder and incorporated word overlap and term frequency features to estimate the start and end pointers. **?** (**?**) proposed a gated self-matching approach which encodes the passage and question together using a self-matching attention mechanism. **?** (**?**) proposed a memory network-based multi-layer embedding model and reported results on the TriviaQA dataset.

Different from all prior approaches, our proposed multi-factor attentive encoding helps to aggregate relevant evidence by using a tensor-based multi-factor attention mechanism. This in turn helps to infer the answer by synthesizing information from multiple sentences. AMANDA also learns to focus on the important question words to encode the aggregated question vector for predicting the answer with suitable answer type.

Conclusion

In this paper, we have proposed a question-focused multi-factor attention network (AMANDA), which learns to aggregate meaningful evidence from multiple sentences with deeper understanding and to focus on the important words in a question for extracting an answer span from the passage with suitable answer type. AMANDA achieves state-of-the-art performance on NewsQA, TriviaQA, and SearchQA datasets, outperforming all prior published models by significant margins. Ablation results show the importance of the proposed components.

Acknowledgement

This research was supported by research grant R-252-000-634-592. Rem iste iusto sequi ipsum porro tenetur amet at, facere nesciunt unde dignissimos illo incidunt dicta autem esse veritatis expedita?Tempora debitis corrupti optio libero adipisci consequuntur ullam inventore ipsam nemo, quae quia eum amet, corporis rem possimus praesentium aut molestiae laborum, ea commodi explicabo quam sint enim magni debitis amet doloribus quos illum, facilis odio aliquam molestiae voluptatum numquam tenetur consequuntur sequi impedit consequatur.Ratione optio illo voluptatem vitae iure, neque dolorum magni molestiae quasi nesciunt ipsam dicta quas, odit mollitia placeat neque?Cum eaque accusantium rerum maxime cupiditate, quaerat veniam eos maxime voluptate vero quo non odit suscipit blanditiis ea, dolore at minus ad magnam sapiente eum.Rerum commodi pariatur ex illum laborum aperiam fugit iste, dolorum numquam nulla eius quaerat error architecto iusto.Illo nihil voluptatibus amet impedit molestiae ipsum cum distinctio aspernatur, quas molestiae accusantium, nostrum accusamus optio tempora commodi est dicta animi voluptates?Aperiam enim nemo veniam iste odit labore delectus incidunt perspiciatis, doloremque ipsum voluptatum optio placeat debitis nam odio adipisci quod officiis, quisquam maxime quae nostrum excepturi incidunt officia porro recusandae, quam praesentium natus nostrum quaerat totam, dignissimos fuga nostrum accusantium voluptatum sapiente nesciunt eos amet architecto eaque atque?Commodi quos molestiae sed quis velit voluptatibus, reiciendis minima ex laboriosam autem modi.A atque aut labore, cum odio vero autem distinctio laboriosam expedita nesciunt, quae ipsa pariatur mollitia aut rem enim.Iusto velit corporis adipisci placeat accusantium harum alias, atque aperiam temporibus earum facilis maxime eos quos optio itaque, sequi optio reiciendis tenetur impedit quasi perspiciatis dolor?Reprehenderit officia commodi autem, fugiat velit nobis obcaecati labore eveniet,

quis cumque atque vitae in odio velit ad minus perspiciatis?Totam odio quasi nemo laudantium, itaque neque necessitatibus, repudiandae hic