Supplementary Materials: Open-Domain Why-Question Answering with Adversarial Learning to Encode Answer Texts

Jong-Hoon Oh[§] **Kazuma Kadowaki**[§‡] **Julien Kloetzer**[§] **Ryu Iida**[§¶] **Kentaro Torisawa**[§¶]

Data-driven Intelligent System Research Center (DIRECT),
National Institute of Information and Communications Technology (NICT)[§]

Advanced Technology Laboratory, The Japan Research Institute, Limited (JRI)[‡]

Graduate School of Science and Technology, NAIST[¶]

{rovellia, kadowaki, julien, ryu.iida, torisawa}@nict.go.jp

A Details of Our Why-QA Model

The overview of our why-question answering (why-QA) model is illustrated in Fig. 1. The model takes as input question q and passage p, and then calculates as output the probability that the passage includes an answer to the question. In this model, *fake-representation generator* F, which was detailed in Section 3.2, generates compact-answer representation $\mathbf{r}_c$ from answer passage p. A *question encoder* and a *passage encoder* respectively generate vector representation $\mathbf{r}_q$ of question q and that $\mathbf{r}_p$ of passage p. The three representations, $\mathbf{r}_q$, $\mathbf{r}_p$ and $\mathbf{r}_c$, are given to an *answer selector*, which is a neural classifier to judge whether passage p contains an answer to question q.

In the following, we will explain the details of the question and passage encoders, and the answer selector.

A.1 Question encoder

The question encoder has the same architecture as the generators in our *Adversarial networks for Generating compact-answer Representation* (AGR), which was presented in Section 3.3. It takes a question as a primary input to be encoded as well as a passage as an additional input used for calculating attention feature vectors (See Section B.2). Following the notations of the encoder in Section 3.2, the question encoder is formalized as:

$$\mathbf{r}_q = \text{Encoder}(q; \theta_Q, p),$$

where θ_Q is a set of parameters to be learned.

A.2 Passage encoder

The passage encoder computes passage representation $\mathbf{r}_p$ of given answer passage $p = s_1, \ldots, s_{|p|}$, where s_i is the i-th sentence in p and $|p|$ is the number of sentences in p. In this encoding, we exploit compact-answer representation $\mathbf{r}_c$ computed

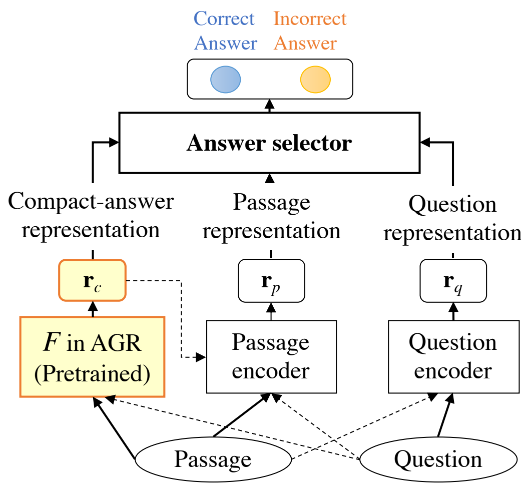

Figure 1: Why-QA model

by fake-representation generator F in AGR (See Section 3.2).

This encoder first computes d-dimensional vector representation[1] $\mathbf{s}_i$ of the i-th sentence in p. This computation is performed using the same architecture as the generators in our AGR (i.e., also the same architecture as the question encoder). That is,

$$\mathbf{s}_i = \text{Encoder}(s_i; \theta_S, q),$$

where θ_S is a set of parameters to be learned.

Then, the encoder computes *attention-weighted passage representation* $\mathbf{s}_{att} \in \mathbb{R}^{d\times|p|}$ by combining $\mathbf{r}_c$ and $\mathbf{s}_p = [\mathbf{s}_1, ..., \mathbf{s}_{|p|}] \in \mathbb{R}^{d\times|p|}$ as:

$$\mathbf{s}_{att} = \text{ReLU}(\mathbf{W}_s(\mathbf{s}_p + \boldsymbol{\beta}\mathbf{s}_p)),$$
$$\boldsymbol{\beta} = \text{softmax}(\mathbf{s}_p^\top \mathbf{W}_p \mathbf{r}_c),$$

where $\mathbf{s}_p = [\mathbf{s}_1, ..., \mathbf{s}_{|p|}] \in \mathbb{R}^{d\times|p|}$. $\mathbf{W}_s, \mathbf{W}_p \in \mathbb{R}^{d\times d}$ are trainable matrices.

Finally, the CNN layer in the passage encoder computes passage representation $\mathbf{r}_p \in \mathbb{R}^d$ using

[1] In our experiments, d was set to 300 (See Section 3.3).

$\mathbf{s}_{att}$ as:

$$\mathbf{r}_p = \text{CNN}(\theta_P, \mathbf{s}_{att}),$$

θ_P is a set of trainable parameters in this CNN. In this CNN layer, convolutions are performed using multiple filters as well as multiple filter windows (sliding over 1, 2, or 3 sentence windows at a time and 100 filters for each window) and then an average pooling operation is applied to the filter outputs.

A.3 Answer selector

Our answer selector consists of a logistic regression layer with dropout and softmax output. It produces the class probabilities of labels $y \in \{\textit{correct}, \textit{incorrect}\}$ (*correct* if answer passage p includes an answer of question q; *incorrect* otherwise). More precisely, the answer prediction is performed by

$$\hat{\mathbf{y}} = \text{softmax}(\mathbf{W}_o\mathbf{r} + \mathbf{b}_o),$$

where $\mathbf{r} = [\mathbf{r}_q, \mathbf{r}_p, \mathbf{r}_c, \mathbf{r}_q^{\mathsf{T}}\mathbf{r}_p, \mathbf{r}_c^{\mathsf{T}}\mathbf{r}_p]^{\mathsf{T}} \in \mathbb{R}^{3d+2}$, $\hat{\mathbf{y}}$ is the predicted label distribution and $\mathbf{W}_o \in \mathbb{R}^{2\times(3d+2)}$ is a trainable matrix. Finally, given answer passages for the same question are ranked by their probability of $y = \textit{correct}$.

B Details of Word Embeddings and Attention in Encoder

In the following, we explain the details of word embeddings and two types of attentions that were briefly described in Section 3.3.

B.1 Word embeddings

Let $q = q_1, \ldots, q_{|q|}$ be a question composed of $|q|$ words, and let $t = t_1, \ldots, t_{|t|}$ be an answer passage or compact answer that are composed of $|t|$ words. We convert the word sequence of q and p into the sequence of word embedding vectors as $\mathbf{q} = [\mathbf{q}_1, \ldots, \mathbf{q}_{|q|}] \in \mathbb{R}^{2d'\times|q|}$ and $\mathbf{t} = [\mathbf{t}_1, \ldots, \mathbf{t}_{|t|}] \in \mathbb{R}^{2d'\times|t|}$, respectively, where $\mathbf{q}_i \in \mathbb{R}^{2d'}$ and $\mathbf{t}_j \in \mathbb{R}^{2d'}$ represent a $2d'$-dimensional word embedding vector[2] for question word q_i and answer passage word (or compact-answer word) t_j. $\mathbf{q}_i$ and $\mathbf{t}_j$ are concatenations of two types of pre-trained word embedding vectors, *general word embedding* (Mikolov et al., 2013) and *causal word embedding vectors* (Sharp et al., 2016), each of which has d'-dimension: $\mathbf{q}_i = [\mathbf{q}_i^g; \mathbf{q}_i^c]$ and $\mathbf{t}_j = [\mathbf{t}_j^g; \mathbf{t}_j^c]$, where $\mathbf{q}_i^g \in \mathbb{R}^{d'}$ and $\mathbf{t}_j^g \in \mathbb{R}^{d'}$ are general embedding vectors of q_i and t_j, and $\mathbf{q}_i^c \in \mathbb{R}^{d'}$ and $\mathbf{t}_j^c \in \mathbb{R}^{d'}$ are causal embedding vectors of q_i and t_j.

B.1.1 General word embeddings

We pre-trained general word embedding vectors, such as $\mathbf{q}_i^g$ and $\mathbf{t}_j^g$, for about 1.65 million words using the skip-gram model with negative-sampling in *word2vec* (Mikolov et al., 2013) on about 35 million sentences from the database dump of Japanese Wikipedia (January 2015 version).

B.1.2 Causal word embeddings

The causal word embedding vectors, such as $\mathbf{q}_i^c$ and $\mathbf{t}_j^c$, are used to represent the *causal associations* between words and are created from *causality expressions*. Each causality expression consists of *cause* and *effect* parts as exemplified below.

CE: $\underline{\text{Volcanoes erupt}}_{\text{[effect]}}$ *because* $\underline{\text{magma pushes through vents and fissures.}}_{\text{[cause]}}$.

In causality expression CE, *magma* and *volcanoes* appear in the cause and effect parts, respectively. Given CE, we can know that if word *volcanoes* appears in a why-question, like “Why do volcanoes erupt?” then *magma* will be more likely to appear in the answer, like “Because magma pushes through vents and fissures” due to the causal association between the two words. To exploit such an association in our why-QA method, which was detailed in Section A, we use causal word embeddings that represent such causal associations.

We extracted causality expressions from 4-billion web pages by using an existing causality recognizer (Oh et al., 2013). The recognizer firstly extracts one or two consecutive sentences that contain pre-defined cue words/phrases[3] and then identifies the boundaries of the cause and effect parts linked to the cue words/phrases using sequential labeling with conditional random fields (CRFs) (Lafferty et al., 2001), which were trained to assign an IOB label to each word using manually annotated training data[4]. The precision, recall, and F-scores reported in Oh et al. (2013) were

[2]In our experiments, d' was set to 300 (See Section 3.3).

[3]We used as cue words/phrases the Japanese translations of the following words/phrases: *for*, *as a result*, *from the fact that*, *because*, *due to*, *this causes*, and *the reason is*.

[4]We used the same data as the one in (Oh et al., 2013) and it was composed of 16,051 causality expression candidates, where one or two consecutive sentences contained at least one cue word/phrase.

83.8%, 71.1%, and 77.0%, respectively (for more details, see Oh et al. (2013)).

As a result, we obtained about 100 million causality expressions. Then, we created the causal word embeddings from them by a method for computing the generalized skip-gram embeddings[5] (Levy and Goldberg, 2014). The basic idea of Levy's method is to use various types of a word's contexts instead of linear bag-of-words contexts used in *word2vec*. In our method, given a causality expression, we regard all the words in the cause part as the contexts of any word in the effect part, and *vice versa*. For example, in the case of CE above, the context of each of *volcanoes* and *erupt* in the effect part is {*magma*, *pushes*, *through*, *vents*, *and*, *fissures*} and the context of each of *magma*, *pushes*, *through*, *vents*, *and*, and *fissures* in the cause part is {*volcanoes*, *erupt*}. As a result of training using 100 million causality expressions, we obtained 300-dimensional causal word embedding vectors for about 1.85 million words. Note that we mapped both the cause and effect words into the same embedding space, while in the original work by Sharp et al. (2016), two types of causal and effect word embedding vectors were mapped into two different embedding spaces.

B.2 Attention

In fake-representation generator F and real-representation generator R, which were introduced in Section 3.1, we use two types of attention mechanisms: *similarity-attention* (dos Santos et al., 2016; Tan et al., 2016) and *causality-attention* (Oh et al., 2017). These attentions help the generators to focus on words in an answer passage that are similar to or causally associated with those in a why-question.

For example, "Honey can last a long time" in the second sentence of answer passage P in Table 1 is related to question Q (i.e., "Why does honey last a long time?"). The similarity-attention is used to attend words in an answer passage that are similar to ones in the question by using the similarities between the word embeddings of the answer passage and the question. On the other hand, some words in compact answer C in Table 1, such as "water" and "microbes," should be regarded as causally associated with words in the question, such as "last" and "long." Such causal associations can be extracted from a large set of causality expressions,

[5]https://bitbucket.org/yoavgo/word2vecf

Q	Why does honey last a long time?
A	While excavating Egypt's pyramids, archaeologists have found pots of honey in an ancient tomb: thousands of years old and still preserved. Honey can last a long time **due to three special properties.** Its average pH is 3.9, which is quite acidic. **Such high level of acidity** is certainly hostile and **hinders the growth of many microbes**. Though honey contains around 17–18% water, **its water activity is too low to support the growth of microbes**. Moreover **honey contains hydrogen peroxide**, which is thought to help **prevent the growth of microbes in honey**. Despite these properties, honey can be contaminated under certain circumstances.
C	Because its acidity, low water activity, and hydrogen peroxide together hinder the growth of microbes.

Table 1: Answer passage P to why-question Q and its compact answer C

such as "Dried fish *lasts long* because it does not contain much *water* and *microbes* cannot grow in it." The causality attention is computed from such causality expressions and is used to attend the passage words causally associated with the words in the question so that generator F generates a more appropriate compact-answer representation.

In the following, we explain the details of similarity-attention and causality-attention, respectively.

B.2.1 Similarity-attention

As presented in Section 3.3, two generators, F and R, have the same architecture Encoder$(t;\theta,q)$, where θ is a set of parameters, q is a why-question, and t is either an answer passage or a compact answer. Each generator first computes the word embedding vectors $\mathbf{t} = [\mathbf{t}_1,\ldots,\mathbf{t}_{|t|}] \in \mathbb{R}^{2d'\times|t|}$ of t and $\mathbf{q} = [\mathbf{q}_1,\ldots,\mathbf{q}_{|q|}] \in \mathbb{R}^{2d'\times|q|}$ of q as in Section B.1.

Similarity-attention is computed by the following two steps. First, we calculate cosine similarity scores for any pair of the words in q and the words in t by using their word embedding vectors $\mathbf{q}_i$ and $\mathbf{t}_j$. Second, we use the similarity scores to produce *similarity-attention feature vector* $\mathbf{a}^s = [a^s_1,\ldots,a^s_{|t|}] \in \mathbb{R}^{|t|}$ as:

$$a^s_j = \max(\cos(\mathbf{q}_1,\mathbf{t}_j),\ldots,\cos(\mathbf{q}_{|q|},\mathbf{t}_j)) \in \mathbb{R},$$

where $\cos(\mathbf{q}_i,\mathbf{t}_j)$ represents the cosine similarity score between question word embedding vector $\mathbf{q}_i$ and passage (or a compact answer) word embedding vector $\mathbf{t}_j$. Note that, as the j-th element of attention feature vector, a^s_j, we use the most similar question word for each passage word t_j by taking the maximum among the cosine similarity scores, i.e., $\max_i\{\cos(\mathbf{q}_i,\mathbf{t}_j)\}$.

B.2.2 Causality-attention

To compute causality-attention, we applied Oh et al. (2017)'s method to 100 million causality expressions that was presented in Section B.1.2. Oh et al. (2017) used a normalized point-wise mutual information (NPMI) score for measuring the strength of causal association to focus on words in an answer passage that are causally associated with the question. Their NPMI score between cause word x and effect word y, $\mathrm{npmi}(x;y)$ ($0 \leq \mathrm{npmi}(x;y) \leq 1$), is defined as:

$$\mathrm{npmi}(x;y) = \max\left(\frac{\mathrm{pmi}(x;y)}{-\log p(x,y)}, 0\right),$$
$$\mathrm{pmi}(x;y) = \log \frac{p(a,b)}{p(x,*)p(*,y)},$$

where $p(x,y)$ is the probability that words x and y respectively appear in the cause and effect parts of the same causality expression. $p(x,*)$ is the probability that word x appears in the cause part, and $p(*,y)$ is the probability that word y appears in the effect part. It is assumed that the causal association between x and y is strong when $\mathrm{npmi}(x;y)$ is high.

After calculating NPMI scores, Oh et al. (2017) used them to produce *causality-attention feature vector* $\mathbf{a}^c = [a_1^c, \ldots, a_{|t|}^c] \in \mathbb{R}^{|t|}$ as:

$$a_j^c = \max(\mathrm{npmi}(t_j; q_1), \ldots, \mathrm{npmi}(t_j; q_{|q|})) \in \mathbb{R}.$$

In our method, given word t_j in a passage (or a compact answer), we regard that t_j as cause word x and a word in the question as effect word y, and compute a causality-attention feature a_j^c of t_j.

B.2.3 Combining two types of attentions

We then create a vector representation $\mathbf{a} \in \mathbb{R}^{2\times|t|}$ by concatenating similarity-attention feature vector $\mathbf{a}^s$ in Section B.2.1 and causality-attention feature vector $\mathbf{a}^c$ in Section B.2.2. Finally, we compute *attention-weighted word embedding* $\mathbf{t}_{att} \in \mathbb{R}^{2d'\times|t|}$ by combining $\mathbf{t} \in \mathbb{R}^{2d'\times|t|}$ and $\mathbf{a}$ as:

$$\mathbf{t}_{att} = \mathrm{ReLU}(\mathbf{W}_t\mathbf{t} + \mathbf{W}_a\mathbf{a})$$

where $\mathbf{W}_t \in \mathbb{R}^{2d'\times 2d'}$ and $\mathbf{W}_a \in \mathbb{R}^{2d'\times 2}$ are trainable parameters, and ReLU represents the rectified linear units. The attention-weighted word embedding $\mathbf{t}_{att}$ is used as an input to the CNN layer that was presented in Section 3.3.

References

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289.

Omer Levy and Yoav Goldberg. 2014. Neural word embedding as implicit matrix factorization. In *Advances in Neural Information Processing Systems 27*, pages 2177–2185. Curran Associates, Inc.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

Jong-Hoon Oh, Kentaro Torisawa, Chikara Hashimoto, Motoki Sano, Stijn De Saeger, and Kiyonori Ohtake. 2013. Why-question answering using intra- and inter-sentential causal relations. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1733–1743.

Jong-Hoon Oh, Kentaro Torisawa, Canasai Kruengkrai, Ryu Iida, and Julien Kloetzer. 2017. Multi-column convolutional neural networks with causality-attention for why-question answering. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, WSDM '17, pages 415–424.

Cícero Nogueira dos Santos, Ming Tan, Bing Xiang, and Bowen Zhou. 2016. Attentive pooling networks. *CoRR*, abs/1602.03609.

Rebecca Sharp, Mihai Surdeanu, Peter Jansen, Peter Clark, and Michael Hammond. 2016. Creating causal embeddings for question answering with minimal supervision. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 138–148.

Ming Tan, Cicero dos Santos, Bing Xiang, and Bowen Zhou. 2016. Improved representation learning for question answer matching. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 464–473, Berlin, Germany. Association for Computational Linguistics.