Learning to Answer Questions from Wikipedia Infoboxes

Alvaro Morales
CSAIL MIT
alvarom@mit.edu

Varot Premtoon
CSAIL MIT
varot@mit.edu

Cordelia Avery
CSAIL MIT
cavery@mit.edu

Sue Felshin
CSAIL MIT
sfelshin@mit.edu

Boris Katz
CSAIL MIT
boris@mit.edu

Abstract

A natural language interface to answers on the Web can help us access information more efficiently. We start with an interesting source of information—infoboxes in Wikipedia that summarize factoid knowledge—and develop a comprehensive approach to answering questions with high precision. We first build a system to access data in infoboxes in a structured manner. We use our system to construct a crowdsourced dataset of over 15,000 high-quality, diverse questions. With these questions, we train a convolutional neural network model that outperforms models that achieve top results in similar answer selection tasks.

1 Introduction

The goal of open-domain question answering is to provide high-precision access to information. With many sources of knowledge on the Web, selecting the right answer to a user's question remains challenging. Wikipedia contains over five million articles in its English version. Providing a natural language interface to answers in Wikipedia is an important step towards more effective information access.

Many Wikipedia articles have an *infobox*, a table that summarizes key information in the article in the form of attribute–value pairs like "Narrated by: Fred Astaire". This data source is appealing for question answering because it covers a broad range of facts that are inherently relevant: a human editor manually highlighted this information in the infobox.

Although many infoboxes appear to be similar, they are only semi-structured—few attributes have consistent value types across articles, infobox templates do not mandate which attributes must be included, and editors are allowed to add article-specific attributes. Infobox-like tables are very common on the Web. Since it is infeasible to incorporate every such source into structured knowledge bases like Freebase (Bollacker et al., 2008), we need techniques that do not rely on ontology or value type information.

Q	Who took over after Nelson Mandela?
A	Succeeded by: Thabo Mbeki
Q	Who designed Central Park?
A	Architect: Frederick Law Olmsted
Q	Where did Oscar Wilde earn his degree?
A	Alma mater: Trinity College, Dublin
Q	What does Intel do?
A	Industry: Semiconductors

Table 1: Example questions and answers with little lexical overlap from the INFOBOXQA dataset.

We focus on the answer selection problem, where the goal is to select the best answer out of a given candidate set of attribute–value pairs from infoboxes corresponding to a named entity in the question. Table 1 illustrates how questions from users may have little lexical overlap with the correct attribute–value pair. Answer selection is an important subtask in building an end-to-end question answering system.

Our work has two main contributions: (1) We compiled the INFOBOXQA dataset, a crowdsourced corpus of over 15,000 questions with answers from infoboxes in 150 articles in Wikipedia. Unlike existing answer selection datasets with answers from knowledge bases or long-form text, INFOBOXQA targets tabular data that is not augmented with value types or linked to an ontology. (2) We built a multi-channel convolutional neural network (CNN) model that achieves the best results on INFOBOXQA compared to other neural network models in the answer selection task.

2 The INFOBOXQA dataset

Infoboxes are designed to be human-readable, not machine-interpretable. This allowed us to devise a crowdsourced assignment where we ask participants to generate questions from infoboxes. With little to no training, humans can form coherent questions out of terse, potentially ambiguous attribute–value pairs.

Wikipedia does not provide a way to access specific information segments; its API returns the entire article. We first worked on the data access problem and developed a system called WikipediaBase to robustly extract attribute–value pairs from infoboxes. Inspired by Omnibase (Katz et al., 2002), we organize infoboxes in an *object–attribute–value* data model, where an object (Lake Titicaca) has an attribute ("Surface area") with a value (8,372 km^2). Attributes are grouped by infobox class (for instance, the `film` class contains attributes like "Directed by" and "Cinematography"). The data model allowed us to extend WikipediaBase to information outside of infoboxes. We implemented methods for accessing images, categories, and article sections.

We then created a question-writing assignment where participants see infoboxes constructed using data from WikipediaBase. These infoboxes visually resembled the original ones in Wikipedia but were designed to control for variables. To prevent participants from only generating questions for attributes at the top of the table, the order of attributes was randomly shuffled. To ensure that the task could be completed in a reasonable amount of time, infoboxes were partitioned into assignments with up to ten attributes. A major goal of this data collection was to gather question paraphrases. For each attribute, we asked participants to write two questions. It is likely that at least one of the questions will use words from the attribute, but requiring an additional question encouraged them to think of alternative phrasings.

Number of questions	15266
Number of attributes	762
Average number of questions per attribute	20.0
Average number of answers per question	17.8

Table 2: Statistics of the INFOBOXQA dataset.

Every infobox in the experiment included a picture to help disambiguate the article. For instance, the cover image for "Grand Theft Auto III" (in concert with the values in the infobox) makes it reasonably clear that the assignment is about a video game and not a type of crime. We asked participants to include an explicit referent to the article title in each question (e.g., "Where was Albert Einstein born?" instead of "Where was he born?").

We analyzed the occurrences of infobox attributes in Wikipedia and found that they fit a rapidly-decaying exponential distribution with a long tail of attributes that occur in few articles. This distribution means that with a carefully chosen subset of articles we can achieve a large coverage of frequently appearing attributes. We developed a greedy approximation algorithm that selects a subset of infobox classes, picks a random sample of articles in the class, and chooses three representative articles that contain the largest quantity of attributes. 150 articles from 50 classes were selected, covering roughly half of common attributes found in Wikipedia.

The dataset contains example questions q_i, with an attribute–value pair (a_i, v_i) that answers the question. To generate negative examples for the answer selection task, we picked every other tuple $(a_j, v_j); \forall j \neq i$ from the infobox that contains the correct answer. If we know that a question asks about a specific entity, we must consider every attribute in the entity's infobox as a possible answer. In INFOBOXQA, candidate answers are just attribute–value pairs with no type information. Because of this, every attribute in the infobox is indistinguishable a priori, and is thus in the candidate set. Not having type information makes the task harder but also more realistic. Table 2 shows statistics of INFOBOXQA. The dataset is available online.[1]

[1] http://groups.csail.mit.edu/infolab/infoboxqa/

3 Model description

Deep learning models for answer selection assume that there is a high similarity between question and answer representations (Yu et al., 2014). Instead of comparing them directly, the main intuition in our model is to use the *attribute* as an explicit bridge to facilitate the match between question and answer. Consider the question "Who replaced Dwight D. Eisenhower?", with answer "Succeeded by: John F. Kennedy". Clearly, the attribute "Succeeded by" plays a crucial role in indicating the match between the question and the answer. If the question and attribute have certain semantic similarities, and those similarities match the similarities of the answer and the attribute, then the answer must be a good match for the question.

We propose an architecture with three weight-sharing CNNs, each one processing either the question, the attribute, or the answer. We then use an element-wise product merge layer to compute similarities between the question and attribute, and between the attribute and answer. We refer to this model as Tri-CNN. Tri-CNN has five types of layers: an input layer, a convolution layer, a max-pooling layer, a merge layer, and a final multi-layer perceptron (MLP) scoring layer that solves the answer selection task. We now describe each layer.

Input layer. Let s_q be a matrix $\in \mathbb{R}^{|s_q| \times d}$, where row i is a d-dimensional word embedding of the i-th word in the question. Similarly, let s_{attr} and s_{ans} be word embedding matrices for the attribute and answer, respectively. s_q, s_{attr}, and s_{ans} are zero-padded to have the same length. We use pre-trained GloVe[2] embeddings with $d = 300$ (Pennington et al., 2014), which we keep adaptable during training.

Convolution layer. We use the multi-channel CNN architecture of (Kim, 2014) with three weight-sharing CNNs, one each for s_q, s_{attr}, and s_{ans}. Different lengths of token substrings (e.g., unigrams or bigrams) are used as channels. The CNNs share weights among the three inputs in a Siamese architecture (Bromley et al., 1993). Weight-sharing allows the model to compute the representation of one input influenced by the other inputs; i.e., the representation of the question is influenced by the representations of the attribute and answer.

[2]http://nlp.stanford.edu/projects/glove/

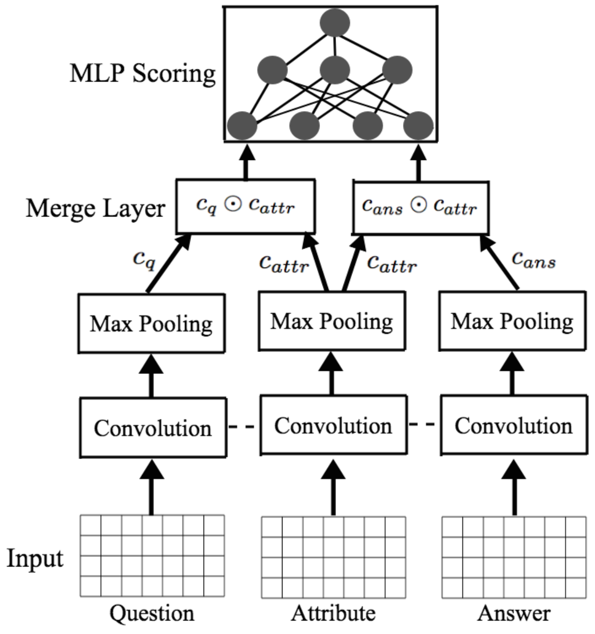

Figure 1: A schematic of the Tri-CNN model.

We describe the convolution layer with respect to the input s, which can stand for s_q, s_{attr}, or s_{ans}. For each channel $h \in [1...M]$, a filter $w \in \mathbb{R}^{h \times d}$ is applied to a sliding window of h rows of s to produce a feature map C. Formally, C is a matrix where:

$$C[i,:] = \tanh(w \cdot s[i...i+h-1,:] + b) \quad (1)$$

and $b \in \mathbb{R}^d$ is the bias. We use *wide convolution* to ensure that terminal and non-terminal words are considered equally when applying the filter w (Blunsom et al., 2014).

Max-pooling layer. Pooling is used to extract meaningful features from the output of convolution (Yin et al., 2015). We apply a max-pooling layer to the output of each channel h. The result is a vector $c_h \in \mathbb{R}^d$ where

$$c_h[i] = \max\{C[:,i]\} \quad (2)$$

Max-pooling is applied to all M channels. The resulting vectors c_h for $h \in [1...M]$ are concatenated into a single vector c.

Merge layer. Our goal is to model the semantic similarities between the question and the attribute, and between the answer and the attribute. We compute the element-wise product of the feature vectors generated by convolution and max-pooling as follows:

$$d_{q,attr} = c_q \odot c_{attr} \tag{3}$$

$$d_{ans,attr} = c_{ans} \odot c_{attr} \tag{4}$$

where $\odot$ is the element-wise product operator, such that d_{ij} is a vector. Each element in d_{ij} encodes a similarity in a single semantic aspect between two feature representations.

MLP scoring layer. We wish to compute a real-valued similarity between the distance vectors from the merge layer. Instead of directly computing this using, e.g., cosine similarity, we follow (Baudiš and Šedivý, 2016) and first project the two distance vectors into a shared embedding space. We compute element-wise sums and products of the embeddings, which are then input to a two-layer perceptron.

4 Experiments

We implemented Tri-CNN in the `dataset-sts`[3] framework for semantic text similarity, built on top of the Keras deep learning library (Chollet, 2015). The framework aims to unify various sentence matching tasks, including answer selection, and provides implementations for variants of sentence-matching models that achieve state-of-the-art results on the TREC answer selection dataset (Wang et al., 2007). We evaluated the performance of various models in `dataset-sts` against INFOBOXQA for the task of answer selection. We report the average and the standard deviation for mean average precision (MAP) and mean reciprocal rank (MRR) from five-fold cross validation. We used 10% of the training set for validation.

In answer selection, a model learns a function to score candidate answers; the set of candidate answers is already given. Entity linking is needed to generate candidate answers and is often treated as a separate module. For INFOBOXQA, we asked humans to generate questions from pre-specified infoboxes. Given this setup, we already know which entity the question refers to; we also know that the question is answerable by the infobox. Entity linking was therefore out of scope in our experiments. By effectively asking humans to identify the named entity, our evaluation results are not affected by noise caused by a faulty entity linking strategy.

[3]https://github.com/brmson/dataset-sts

Model	**MAP**		**MRR**	
	Avg	*SD*	*Avg*	*SD*
TF-IDF	0.503	0.004	0.501	0.065
BM-25	0.531	0.007	0.532	0.056
AVG	0.593	0.021	0.609	0.042
RNN	0.685	0.024	0.674	0.028
ATTN1511	0.772	0.016	0.771	0.014
CNN	0.757	0.015	0.754	0.024
Tri-CNN	**0.806**	**0.014**	**0.781**	**0.025**

Table 3: Results of five-fold cross validation. Our Tri-CNN model achieves the best results in MAP and MRR.

4.1 Benchmarks

We compare against TF-IDF and BM25 (Robertson et al., 1995), two models from the information retrieval literature that calculate weighted measures of word co-occurrence between the question and answer. We also experiment with various neural network sentence matching models. AVG is a baseline model that computes averages of unigram word embeddings. CNN is the model most similar to Tri-CNN, with two CNNs in a Siamese architecture, one for the question and one for the answer. Max-pooling is computed on the output of convolution, and then fed to the output layer directly. RNN computes summary embeddings of the question and answer using bidirectional GRUs (Cho et al., 2014). ATTN1511 feeds the outputs of the bi-GRU into the convolution layer. It implements an asymmetric attention mechanism as in (Tan et al., 2015), where the output of convolution and max-pooling of the question is used to re-weight the input to convolution of the answer. The convolution weights are not shared. For these neural architectures, we use the same MLP scoring layer used in Tri-CNN as the output layer and train using bipartite RankNet loss (Burges et al., 2005).

4.2 Results

Table 3 summarizes the results of experiments on INFOBOXQA. The performance of the baselines indicates that unigram bag-of-words models are not sufficiently expressive for matching; Tri-CNN makes use of larger semantic units through its multiple channels. The attention mechanism and the combination of an RNN and CNN in ATTN1511 achieves better results than RNN, but still performs slightly worse than the CNN model with weight-sharing. The Siamese architecture allows an input's representation to be influenced by the other inputs. The convolution feature maps are thus encoded in a comparable scheme that is more amenable to a matching task. Our Tri-CNN model built on top of this weight-sharing architecture achieves the best performance. Tri-CNN computes the match by comparing the similarities between question–attribute and answer–attribute, which leads to improved results over models that compare the question and answer directly.

5 Related work

Deep learning approaches to answer selection have been successful on the standard TREC dataset and the more recent WIKIQA corpus (Yang et al., 2015). Models like (Feng et al., 2015) and (Wang and Nyberg, 2015) generate feature representations of questions and answers using neural networks, computing the similarity of these representations to select an answer. Recently, attention mechanisms to influence the calculation of the representation (Tan et al., 2015) or to re-weight feature maps before matching (Santos et al., 2016) have achieved good results. Our work differs from past approaches in that we use the attribute as an additional input to the matching task. Other approaches to question answering over structured knowledge bases focus on mapping questions into executable database queries (Berant et al., 2013) or traversing embedded sub-graphs in vector space (Bordes et al., 2014).

6 Conclusion

We presented an approach to answering questions from infoboxes in Wikipedia. We first compiled the INFOBOXQA dataset, a large and varied corpus of interesting questions from infoboxes. We then trained a convolutional neural network model on this dataset that uses the infobox attribute as a bridge in matching the question to the answer. Our Tri-CNN model achieved the best results when compared to recent CNN and RNN-based architectures. We plan to test our model's ability to generalize to other types of infobox-like tables on the Web. We expect our methods to achieve good results for sources such as product descriptions on shopping websites.

Acknowledgments

We thank Andrei Barbu and Yevgeni Berzak for helpful discussions and insightful comments on this paper. We also thank Ayesha Bose, Michael Silver, and the anonymous reviewers for their valuable feedback. Federico Mora, Kevin Ellis, Chris Perivolaropoulos, Cheuk Hang Lee, Michael Silver, and Mengjiao Yang also made contributions to the early iterations and current version of Wikipedia-Base. The work described in this paper has been supported in part by AFRL contract No. FA8750-15-C-0010 and in part through funding provided by the Center for Brains, Minds, and Machines (CBMM), funded by NSF STC award CCF-1231216.

References

Petr Baudiš and Jan Šedivỳ. 2016. Sentence pair scoring: Towards unified framework for text comprehension. *arXiv preprint arXiv:1603.06127*.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *EMNLP*.

Phil Blunsom, Edward Grefenstette, and Nal Kalchbrenner. 2014. A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*. Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, pages 1247–1250. ACM.

Antoine Bordes, Sumit Chopra, and Jason Weston. 2014. Question answering with subgraph embeddings. *arXiv preprint arXiv:1406.3676*.

Jane Bromley, James W Bentz, Léon Bottou, Isabelle Guyon, Yann LeCun, Cliff Moore, Eduard Säckinger, and Roopak Shah. 1993. Signature verification using a "Siamese" time delay neural network. *International Journal of Pattern Recognition and Artificial Intelligence*, 7(04):669–688.

Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to rank using gradient descent. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 89–96. ACM.

Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. In *Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation (SSST-8)*.

François Chollet. 2015. Keras. `https://github.com/fchollet/keras`.

Minwei Feng, Bing Xiang, Michael R Glass, Lidan Wang, and Bowen Zhou. 2015. Applying deep learning to answer selection: A study and an open task. *arXiv preprint arXiv:1508.01585*.

Boris Katz, Sue Felshin, Deniz Yuret, Ali Ibrahim, Jimmy Lin, Gregory Marton, Alton Jerome McFarland, and Baris Temelkuran. 2002. Omnibase: Uniform access to heterogeneous data for question answering. In *Natural Language Processing and Information Systems*, pages 230–234. Springer.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *EMNLP 2014*, 13.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, and Mike Gatford. 1995. Okapi at TREC-3. *NIST Special Publication SP*, 109:109.

Cicero dos Santos, Ming Tan, Bing Xiang, and Bowen Zhou. 2016. Attentive pooling networks. *arXiv preprint arXiv:1602.03609*.

Ming Tan, Bing Xiang, and Bowen Zhou. 2015. LSTM-based deep learning models for non-factoid answer selection. *arXiv preprint arXiv:1511.04108*.

Di Wang and Eric Nyberg. 2015. A long short-term memory model for answer sentence selection in question answering. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*.

Mengqiu Wang, Noah A Smith, and Teruko Mitamura. 2007. What is the Jeopardy model? A quasi-synchronous grammar for QA. In *EMNLP-CoNLL*, volume 7, pages 22–32.

Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. WikiQA: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2013–2018. Citeseer.

Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. 2015. ABCNN: Attention-based convolutional neural network for modeling sentence pairs. *arXiv preprint arXiv:1512.05193*.

Lei Yu, Karl Moritz Hermann, Phil Blunsom, and Stephen Pulman. 2014. Deep Learning for Answer Sentence Selection. In *NIPS Deep Learning Workshop*, December.