IUCM at SemEval-2018 Task 11: Similar-Topic Texts as a Comprehension Knowledge Source

Sofia Reznikova
Innopolis University
Innopolis, Russia
s.reznikova@innopolis.ru

Leon Derczynski
IT University of Copenhagen
Denmark
leod@itu.dk

Abstract

This paper describes the IUCM entry at SemEval-2018 Task 11, on machine comprehension using commonsense knowledge. First, clustering and topic modeling are used to divide given texts into topics. Then, during the answering phase, other texts of the same topic are retrieved and used as commonsense knowledge. Finally, the answer is selected. While clustering itself shows good results, finding an answer proves to be more challenging. This paper reports the results of system evaluation and suggests potential improvements.

1 Introduction

The goal of SemEval-2018 Task 11 is to find a way to incorporate commonsense knowledge into a question-answering task (Ostermann et al., 2018b). In this case, questions are either directly or indirectly related to given English texts; some questions may be answered using the text while others require background (commonsense) knowledge. The challenge is to use this knowledge in such a way as to enhance the quality of chosen answers.

There are many approaches to question answering including using structured knowledge (Yao and Durme, 2014), knowledge databases (Yih et al., 2015), deep learning methods (Minaee and Liu, 2017) and hybrid methods (Xu et al., 2016; Das et al., 2017). The present task accepts any method or any source of background knowledge.

The training data consists of 1469 texts covering more than 100 topics. The number of questions per text varies from 1 to 14 and there are two answer options. The development data has 219 texts and the test data has 430. (Ostermann et al., 2018a)

The main idea behind the method proposed in this paper is to use the given texts as potential sources of knowledge. Texts from training and development data can be divided into topics using existing clustering algorithms (e.g. k-Means, Hierarchical, Grid-based or Density-based). The hypothesis is that texts which come from the same topic as the current question's text may contain the correct answer. A matching function with a scoring scheme is used to identify the correct combination of words for the answer.

Another potential source of knowledge is scripts (Wanzare et al., 2016) that have been used to search for an answer. The DeScript dataset includes descriptions of everyday activities such as baking, getting a haircut, going grocery shopping, and others, corresponding to topics present in the given data.

Section 2 describes the methods as well as specifics of implementation. Section 3 provide interpretation and analysis of the results. Section 4 concludes the present paper.

2 Methodology

All texts, questions and answers were tokenized, punctuation and extra symbols were removed. WordNet (Fellbaum, 1998) was used for lemmatization using its *morphy()* function. Transforming all words into their initial form resulted in an approximately 1% increase of accuracy.

The overall process of answering a question could be broadly divided into two phases: clustering texts (or searching for the most similar DeScript's topic), and finding the correct answer. The former is discussed in the following subsections, and the latter is described below.

There are minor modifications to the base *choose-answer* method across all solutions but overall the structure is as follows. First, we search for a full-length match. If one is found, then we consider it to be a correct one. If not, we remove all common words (such as articles, prepositions

Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018), pages 1068–1072
New Orleans, Louisiana, June 5–6, 2018.

and auxiliary verbs) from the answer, and count how many words can be found in text. Finally, we compare all answers and choose the one with the highest match count.

A modification was introduced to account for yes-no questions. If words from the question were present in the text, the "yes" answer was selected; otherwise, the "no" answer was selected. This, however, actually decreased the accuracy as it did not consider negations that occurred in the text and were tokenized separately. Therefore, the prior version of the method was used in all submissions.

The baseline solution used the method described above to find the answer in the given text only. This resulted in accuracy of around 60% for all data sets (see Section 3).

2.1 Comparison of BigARTM and KMeans

The next step was to cluster texts from training and development data into topics, in order to use them later as sources of background knowledge. BigARTM, a tool to infer topics based on additive regularization of topic models proposed in Vorontsov and Potapenko (2014), was used to model the texts as topics.

Since the language of the given texts consisted of many everyday words, it was necessary to first make sure that the data used for clustering was clear of common, uninformative words. BigARTM provides tools to make the resulting matrix of document-topic mapping sparser, however, it did not provide as good a result as simple removal of the English stopwords contained in the NLTK library (Bird and Loper, 2004).

The texts then were transformed into batches in vectorized form, the number of topics varied between 100 and 110, and the model initialized with scores for sparsity (PhiScore and ThetaScore) and perplexity. The model was then trained with 15 passes through the collection of texts (bigger numbers didn't result in better accuracy).

Topic	Probability
topic 33	0.024480
topic 90	0.021954
topic 26	0.020483
topic 37	0.017545
topic 82	0.016841

Table 1: Probability of the topic being the top choice for the text

Table 1 shows five most frequent topics and the probability of the topic being the top one for the text. Below are the top 15 tokens for three most frequent topics:

- 'pan', 'eggs', 'milk', 'heat', 'stove', 'egg', 'turn', 'make', 'pour', 'cook', 'add', 'hot', 'get', 'omelette', 'bowl'
- 'wall', 'paint', 'room', 'new', 'floor', 'look', 'decide', 'house', 'want', 'put', 'buy', 'color', 'get', 'painting', 'work'
- 'water', 'shower', 'get', 'soap', 'rinse', 'towel', 'turn', 'body', 'hot', 'take', 'bucket', 'dry', 'clean', 'start', 'warm'

Another approach to clustering was to use k-Means. First, texts were embedded in the vector space using the gensim (Řehůřek and Sojka, 2010) implementation of Doc2Vec (Le and Mikolov, 2014), and then NLTK's k-Means was applied over the result.

The Doc2Vec model was trained on the texts from all data sets and for several runs DeScript's gold standard was also added. At each step the learning rate was decreased by 0.002. Number of epochs varied from 20 to 30 and window size was also experimented with.

The top 15 tokens for three most frequent topics are as follows:

- 'wall', 'paint', 'painting', 'room', 'look', 'get', 'would', 'want', 'go', 'hang', 'decide', 'put', 'nail', 'wallpaper', 'color'
- 'bed', 'sheet', 'pillow', 'put', 'make', 'take', 'top', 'get', 'corner', 'tuck', 'fit', 'sure', 'clean', 'mattress'
- 'dish', 'put', 'dishwasher', 'dry', 'sink', 'plate', 'water', 'clean', 'rack', 'wash', 'silverware', 'one', 'start', 'top', 'take'

The top clusters for the two methods are slightly different in terms of their topics but there are obvious differences in words: k-Means clusters include more general language (e.g. verbs like 'go', 'take', 'make') and less specific language related to the topic. At the same time k-Means' distribution of classes has a larger number of texts per cluster on average. This can be seen in Figures 1 and 2. The horizontal axis is number of texts and the vertical one is the cluster ID number.

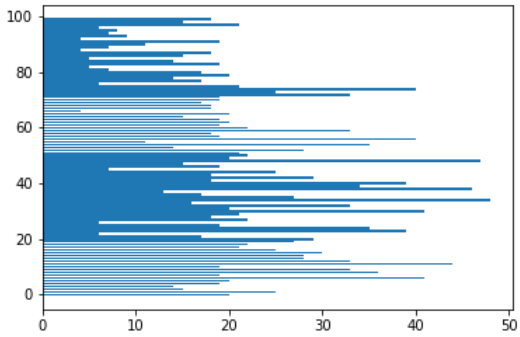

Figure 1: Distribution of texts for KMeans (*y-axis*: topic numbers, *x-axis*: number of texts belonging to the topic

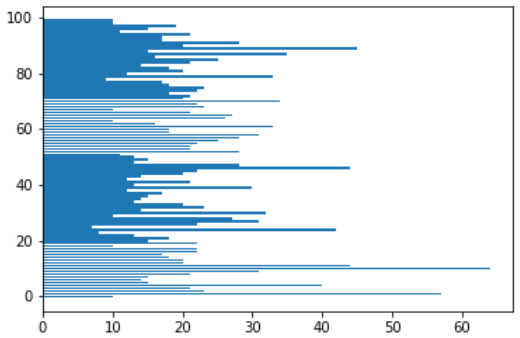

Figure 2: Distribution of texts for BigARTM (*y-axis*: topic numbers, *x-axis*: number of texts belonging to the topic

For both clustering methods, choosing the correct answer was done in two steps. The first step was the same as in the baseline, finding full answers or scoring individual words. If the answer was not found, the second step was carried out, which involved looking up other texts from the same cluster as the given text and searching for an answer in them.

2.2 Using DeScript

DeScript (Wanzare et al., 2016) sequences are divided into events which, in turn, include many different paraphrases of the same event (*check timetable, locate a train schedule, check train schedules* and other similar ones).

To simplify comparison between training data texts and DeScript sequences, all events corresponding to the same topic were combined. Then the vector for each topic was built using the 15 most frequent words from the topic as keys and their TF-IDFs as values. The same was done for each text.

The method for choosing the answer was as described above. The difference was that instead of using texts from the same cluster as a source of commonsense knowledge, DeScript paraphrases were used. Cosine similarity between the given text and each DeScript topic was calculated based on most frequent word vectors, in order to find the most suitable events. This approach resulted in better performance than with clustering methods (Section 3).

3 Evaluation

The results for the test data are summarized in Table 2. These are the configurations that resulted in the best performance.

Model	Accuracy
Baseline	60.70
Yes/No Modification	59.52
BigARTM topics	61.38
DeScript paraphrases	61.67
Doc2Vec/KMeans	61.95

Table 2: Results

BigARTM's advantage over the baseline solution is not much, but there is an interesting trend that explains why the score is higher. Questions with no word-for-word answer in the texts were answered correctly when individual words were found within the same-topic clusters. This showed that given texts could be a useful knowledge source.

There are also cases when both answers get zero scores and in that case the first one is chosen.

Another observation is that correct answers were more often selected if they contained full sentences rather than a couple of words.

The DeScript and BigARTM methods answered 6% of questions differently. These were, for the most part, for answers that were not explicitly phrased in the text but obvious to a human (such as *evening* when the text talked about dinner, or *bedroom* when a bed was mentioned). This requires an additional logical step, so this kind of questions can be in a category of their own – neither text nor commonsense.

For k-Means the number of clusters was 100. Table 3 describes the results for Doc2Vec/k-Means method with various configurations (number of

epochs, window size, whether DeScript texts were included into the training or not).

Epochs	Window	DeScript	Accuracy
20	10	No	61.70
20	12	No	61.60
20	10	Yes	61.56
30	10	No	61.74
30	12	No	61.88
30	10	Yes	61.24
30	12	Yes	61.95

Table 3: Doc2Vec/KMeans configurations and accuracy

The model with a larger window size performs better as it takes into account more words at the same time, sometimes spanning multiple sentences at once. Adding DeScript dat does not have significant impact on the results. However, as the scripts are succinct and topic-related they give a slight boost to the overall accuracy.

The k-Means-based system generally does better in questions related to timing (e.g. how much some activity takes) and in questions about text's meta-information (answers that include *author* or *narrator*). This observation could be explained by the fact that there are some activities that happen at a specific time of the day (e.g. breakfast, going out and others) and Doc2Vec could do a better embedding for numbers.

Overall, while the clustering step provided commonsense knowledge for the system and successfully mapped texts to topics, the bottleneck was the method of choosing an answer. It is based on the assumption that finding the exact answer or individual words from it leads to the correct solution. Different scoring and prioritizing methods for searching did not improve accuracy in any significant way. Therefore, a function that incorporates different approaches (e.g. comparing vector representations of questions and answers, POS-tagging for the question, deep similarity) along with simple matching might lead to better results.

4 Conclusion

This paper described the methodology behind the IUCM at SemEval-2018 Task 11 on machine comprehension using commonsense knowledge. The proposed solution is based on different techniques of unsupervised learning. The method shows above-the-baseline performance and results in clear topic division and mapping. The code for the system is available here: https://github.com/sonyareznikova/semeval2018task11.

References

Steven Bird and Edward Loper. 2004. NLTK: the natural language toolkit. In *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*, page 31. Association for Computational Linguistics.

Rajarshi Das, Manzil Zaheer, Siva Reddy, and Andrew McCallum. 2017. Question answering on knowledge bases and text using universal schema and memory networks. *In ACL*, arXiv:1704.08384.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA.

Quoc V. Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. arXiv:1405.4053.

Shervin Minaee and Zhu Liu. 2017. *Automatic Question-Answering Using A Deep Similarity Neural Network.*, arXiv:1708.01713.

Simon Ostermann, Ashutosh Modi, Michael Roth, Stefan Thater, and Manfred Pinkal. 2018a. MCScript: A Novel Dataset for Assessing Machine Comprehension Using Script Knowledge. In *Proceedings of the 11th International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan.

Simon Ostermann, Michael Roth, Ashutosh Modi, Stefan Thater, and Manfred Pinkal. 2018b. SemEval-2018 Task 11: Machine Comprehension using Commonsense Knowledge. In *Proceedings of International Workshop on Semantic Evaluation (SemEval-2018)*, New Orleans, LA, USA.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta. ELRA. http://is.muni.cz/publication/884893/en.

Konstantin Vorontsov and Anna Potapenko. 2014. Tutorial on probabilistic topic modeling: Additive regularization for stochastic matrix factorization. *International Conference on Analysis of Images, Social Networks and Texts*.

Lilian D. A. Wanzare, Alessandra Zarcone, Stefan Thater, and Manfred Pinkal. 2016. DeScript: A Crowdsourced Corpus for the Acquisition of High-Quality Script Knowledge.

Kun Xu, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2016. Hybrid question answering over knowledge base and free text. In *COLING*.

Xuchen Yao and Benjamin Van Durme. 2014. Information extraction over structured data: Question answering with Freebase. In *ACL*.

Scott Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. Semantic parsing via staged query graph generation: Question answering with knowledge base. ACL Association for Computational Linguistics.