Machine Comprehension using Commonsense Knowledge - a Dynamic Memory Network Approach

Daniel Shaprin **Denis Kyashif** **Kuzman Belev**
{dshaprin, denis.kyashif, kuzman.belev}@gmail.com

Sofia University - Faculty of Mathematics and Informatics

Abstract

In this paper we describe the system we built for *SemEval-2018 Task 11: Machine Comprehension using Commonsense Knowledge*. We solve the task with a **Dynamic Memory Network** and incorporate commonsense knowledge from corpora of event sequence descriptions like DeScript, OMCS Stories and RKP. As a result our system achieves 68.5% accuracy rate on the provided test data. The source code is made publicly available.

1 Introduction

The **SemEval-2018 Task 11: Machine Comprehension using Commonsense Knowledge** challenges the participants to implement a system, that given narrative texts about performing common tasks and activities, answers correctly questions related not only these texts but also some questions, answers to which are regarded as "commonsense". The system should be able to pick the correct answer from two options. Use of external knowledge for training the system is encouraged and there are no restrictions to the training data size. We present a solution based on a **Dynamic Memory Network** architecture.

2 Training Data

The data provided by the authors of the task represents a set of entries, where each entry consists of a *text*, *questions about the text* and *two answers for each question* where one is marked as correct.

We also used external corpora with event sequence descriptions such as DeScript (Wanzare et al., 2016), OMCS Stories (Singh et al., 2002) and RKP (Regneri et al., 2010) to introduce commonsense knowledge. DeScript, OMCS and RKP are collections of *scenarios/scripts* describing everyday activities. E.g. we have separate scenario for *baking a cake* and *borowing a book from the library*. Each scenario has numerous event sequence descriptions (ESDs) (each written by a different contributor) that in turn have numerous **event descriptions** (EDs) - simple actions, e.g. *"Place pan into oven", "Set a timer"*.

3 System Description

3.1 Preprocessing

Word Embeddings: We use pre-trained word vectors from **fastText** (Joulin et al., 2017) and **GloVe** (Pennington et al., 2014) to augment the commonsense knowledge picker and the neural network respectively.

Co-reference Resolution: The training data consists of mostly simple sentences which contain many pronoun references to people or objects. We used **py-corenlp**[1] to access the **Stanford CoreNLP** (Manning et al., 2014) API from Python code and implemented a procedure that replaces these pronouns inside a text with the name of the object they refer to.

Choosing the Relevant Commonsense Knowledge: It is clear that every text needs to be augmented with some a piece of commonsense knowledge and our approach in choosing the relevant one evolved both in terms of abstract idea and tools used.

Since DeScript, OMCS and RKP follow similar structure, we regard them as one big resource. We wanted to check whether given text corresponds to one of the scenarios in the corpora. To do that, we tried using TF-IDF (by **sklearn**) and finding the cosine similarity between the text and the scenarios. After we found that a fair portion of the texts look similar in terms of TF-IDF to some of the scenarios, we decided to use word embedding vector representations (by **fastText**) instead . Unfortunately, we discovered that these vector representa-

[1] https://github.com/smilli/py-corenlp

tions do not work well with large texts (some of the scenarios have hundreds of ESDs with dozens of EDs each). The result was high cosine similarity (over 0.75) for every text and scenario. Therefore that approach was abandoned.

To choose the correct piece of knowledge from our corpora, we implemented a function that given an input text returns the closest ESD available. The corpora are split into ESDs, that are in turn split into EDs, that are finally split into words. A 300-dimentional *word vector* is then assigned to each word. We create *weighted word vectors* by multiplying the initial vectors by the word IDF, calculated over the comlete corpora after stemming with **NLTK**. To create an *event description vector* we take the mean of the weighted word vectors in the event description. Similarly, to finally create an *ESD vector*, for each ESD we take the mean of all ED vectors in the ESD.

On the other hand, we concatenate every question with the associated pair of answers and in a similar way produce a *question-answers vector* also weighting the words in the concatenated text. The question-answers vector is compared with all the ESD vectors using cosine similarity and the closest ESD is then selected. High cosine similarity value indicates that an entry is closely related to the ESD thus is good candidate for containing relevant information regarding the question.

3.2 Dynamic Memory Network

Dynamic Memory Networks (DMN) (Kumar et al., 2016) represent a system of neural networks which are widely used for developing question answering systems for a given text. They consist of four modules: input, question, episodic memory and an answer module.

Input Module: The input module represents a Recurrent Neural Network of type GRU (Gated Recurrent Unit), which takes as an input the text upon which we ask the questions. The words are replaced with the embeddings from GloVe and a designated symbol is added to mark the end of a sentence. The value of the hidden layer is taken on each step and provided to the episodic memory module. The outputs of this network are called *facts* and encode information about the sequence of the words.

Question Module: Like the input module, the question module encodes the question of the task into a distributed vector representation. The representation is fed into the episodic memory module, and forms the initial state, upon which it iterates.

Episodic Memory Module: This module uses the attention mechanism to determine which parts of the inputs from the given collection of input representations to focus on. It considers the question and the previous memory in order to produce a memory vector representation. Each iteration provides the module with new relevant information about the input.

Answer Module: The answer module takes the **last memory** of the episodic memory module together with the question and the potential answers and returns a probability for each of the answers.

Training: Training falls down to a supervised classification problem, that is to minimize cross-entropy error of the answer sequence. The entire DMN model can be trained via backpropagation and gradient descent.

4 Experiments and Results

Our model achieved 68.5% accuracy in the provided test data. Our team takes 16'th place in the final ranking, placed 6 points behind the 7'th placed team in the heavily congested mid-range of the table. The accuracy over the validation data was 71.7% with co-reference resolution and 69.8% without.

We incorporated also a rule-based question classifier (Tayyar Madabushi and Lee, 2016) in order to train separate models for each question class. The classification is a two level system containing a coarse and a fine level of classification for each question as described in (Li and Roth, 2002). We used only the coarse level classes which are as follows: ABBR (abbreviation), DESC(description), ENTY (entity), HUM (person), LOC (location), NUM (number). The models trained only over single question type with co-reference resolution had average accuracy of 68.6% and all of them had less than 71.7% which showed that this approach was not more successful than training on the whole dataset.

5 Conclusion

We integrated a DNM with common sense knowledge obtained from ESDs for solving SemEval-18, Task11. The method to select a suitable ESD for each text and the co-reference resolution were essential to raise our score and achieve 68.5% accuracy on the test data.

References

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain. Association for Computational Linguistics.

Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. 2016. Ask me anything: Dynamic memory networks for natural language processing. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1378–1387, New York, New York, USA. PMLR.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *Proceedings of the 19th International Conference on Computational Linguistics - Volume 1*, COLING '02, pages 1–7, Stroudsburg, PA, USA. Association for Computational Linguistics.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

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.

Michaela Regneri, Alexander Koller, and Manfred Pinkal. 2010. Learning script knowledge with web experiments. In *Proceedings of ACL 2010*, Uppsala, Sweden. Association for Computational Linguistics.

Push Singh, Thomas Lin, Erik T. Mueller, Grace Lim, Travell Perkins, and Wan Li Zhu. 2002. Open mind common sense: Knowledge acquisition from the general public. In *On the Move to Meaningful Internet Systems, 2002 - DOA/CoopIS/ODBASE 2002 Confederated International Conferences DOA, CoopIS and ODBASE 2002*, pages 1223–1237, London, UK, UK. Springer-Verlag.

Harish Tayyar Madabushi and Mark Lee. 2016. High accuracy rule-based question classification using question syntax and semantics. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1220–1230, Osaka, Japan. The COLING 2016 Organizing Committee.

Lilian D. A. Wanzare, Alessandra Zarcone, Stefan Thater, and Manfred Pinkal. 2016. Descript: A crowdsourced corpus for the acquisition of high-quality script knowledge.