Instructions for ACL-2018 Proceedings

|  |  |
| --- | --- |
| 000  001  002  003  004  005  006  007  008  009  010  011  012  013  014  015  016  017  018  019  020  021  022  023  024  025  026  027  028  029  030  031  032  033  034  035  036  037  038  039  040  041  042  043  044  045  046  047  048  049 | 050  051  052  053  054  055  056  057  058  059  060  061  062  063  064  065  066  067  068  069  070  071  072  073  074  075  076  077  078  079  080  081  082  083  084  085  086  087  088  089  090  091  092  093  094  095  096  097  098  099 |

|  |
| --- |
|  |
| Anonymous ACL submission |
|  |
|  |
|  |
|  |
|  |

Abstract

The goal of the project is to build a question answering system which can answer users’ question about a document. Question answering system is very useful since it can extract small pieces of key information from large volume of text. Therefore, it is widely used in modern browsers and auto-answering services.

Credits

This document has been adapted from the instructions for earlier ACL and NAACL proceedings, including those for ACL 2017 by Dan Gildea and Min-Yen Kan, NAACL-2016 by Margaret Mitchell, ACL-2012 by Maggie Li and Michael White, those from ACL-2010 by Jing-Shing Chang and Philipp Koehn, those for ACL-2008 by Johanna D. Moore, Simone Teufel, James Allan, and Sadaoki Furui, those for ACL-2005 by Hwee Tou Ng and Kemal Oflazer, those for ACL-2002 by Eugene Charniak and Dekang Lin, and earlier ACL and EACL formats. Those versions were written by several people, including John Chen, Henry S. Thompson and Donald Walker. Additional elements were taken from the formatting instructions of the *International Joint Conference on Artificial Intelligence* and the *Conference on Computer Vision and Pattern Recognition*.

Introduction

The following instructions are directed to authors of papers submitted to ACL-2018 or accepted for publication in its proceedings. All authors are required to adhere to these specifications. Authors are required to provide a Portable Document Format (PDF) version of their papers. **The proceedings are designed for printing on A4 paper**.

All MSWord formatting for ACL-2018 is made available in the MSWord Styles in this template. In newer versions of MSWord, click Home, then expand the Styles tile by clicking the diagonal arrow on the lower left corner. This should open all styles in the template for you to apply to your document as needed. Otherwise, you may expose the Styles following the instructions provided at:

<http://blogs.technet.com/b/hub/achive/2010/11/22/view-and-edit-styles-quickly-in-word-2010.aspx>.

Suppose we already got the tag of target question the target sentence which contains the expected answer we want to output, CogComp-NLP helps us tagging the target sentence, by matching each word’s tag with given questions' tag, we can finally catch all the possible words that have high probability that be included in correct answer. Next, I will introduce how to get the answer of target question and the accuracy of the model step by step.

Finding the sentences that contains an answer

In our project, two possible scenarios are considered – with context and without context and in both supervised and unsupervised way. Our goal is to find the sentence that contains the answer. And later this sentence is extracted to get the final answer. The basic approach here is as following. First, we represent both the question and the sentences in the context as a high dimensional vector. This could be done using sentence encoders and the details will be discussed later. After that, we can use this feature along with other feature for the later unsupervised learning as well as supervised learning.

As we can see sentence embedding plays a key role in both scenarios as it enables us to compare the similarity of two sentences in terms of the distances of the corresponding vector generated by sentence embedding. Despite there are many sentence encoders learned in an unsupervised manner, it has been proved that models learned on NLI can perform better than models trained in unsupervised conditions or on other supervised tasks. One of the relative work is InferSent, which is developed by Facebook AI Research.

In later sections, we will first discuss the result in the case that we have a context that contains answer provided. Then we will discuss the case without context at the end.

|  |  |
| --- | --- |
| Sentences | Vector Representation |
| 0 | [ 0.05519996, 0.05013141, 0.04787038, ..., 0.00821209, -0.03642813, 0.044685 ] |
| 1 | [ 0.07475325, 0.11794458, 0.06240867, ..., 0.01915886, -0.02436746, 0.10806957] |
| 2 | [0.11262652, 0.11146841, 0.14750297, ..., 0.00293285, 0.03322018, 0.06657628] |
| Question | [ 0.1095634 , 0.1142294 , 0.04428943, ..., 0.02811733, -0.01866924, 0.12806854] |

Table 2: Sentence Embedding Result

Data Preprocessing

In the data preprocessing step, what we do is first import data from Squad dataset. Then we use TextBlob to separate the context into sentences. Table 1 below is an example of the separation result.

|  |  |
| --- | --- |
| Context | 'Architecturally, the school has a Catholic character. Atop the Main Building\'s gold dome is a golden statue of the Virgin Mary. Immediately in front of the Main Building and facing it, is a copper statue of Christ with arms upraised with the legend "Venite Ad Me Omnes". |
| Sentences | 'Architecturally, the school has a Cat-holic character.' |
| ‘Atop the Main Building's gold dome is a golden statue of the Virgin Mary.’ |
| 'Immediately in front of the Main Building and facing it, is a copper statue of Christ with arms upraised with the legend"Venite Ad Me Omnes".' |
| Question | 'To whom did the Virgin Mary allegedly appear in 1858 in Lourdes France?' |

Table 1: Separation result using TextBlob

After the separation, we get separate sentences suitable for sentence embedding. Besides, the questions as a sentence are also encoded with InferSent. Infersent provides a range of models trained on different vectors. According to our observation, Infersent 1 with glove.840B.300d dataset works best. Table 2 shows the vector representation of the three sentences in Table 1. The following work will be based on this dataset but we will also discuss the impact of other datasets at the end of the section.

Unsupervised Learning

Our unsupervised learning approach is simply evaluate each candidate sentence with the question by the distance between the two vectors and select the sentence with the lowest distance.

|  |  |  |  |
| --- | --- | --- | --- |
| Sentence index | 0 | 1 | 2 |
| Cosine similarity | 0.43262481689453125 | 0.36864835023880005 | 0.3676990866661072 |
| Euclidean distance | 3.7561161518096924 | 3.8744754791259766 | 4.247783660888672 |
| Manhattan distance | 175.40135 | 182.2885 | 207.47821 |
| Chebyshev  distance | 0.32886288 | 0.3149855 | 0.28589985 |

Table 3: Distance of each sentence with question

There are many metrics to evaluate the distance between two vectors such as cosine similarity, Euclidean distance and others. Table 3 shows the result of various distance computed on the sentences and question in Table 2.

For this case, the sentence that contains answer is 'It is a replica of the grotto at Lourdes, France where the Virgin Mary reputedly appeared to Saint Bernadette Soubirous in 1858.' with a cosine similarity of 0.193090915679931. Table 4 shows the overall result of unsupervised learning using different distances.

We could see that cosine similarity gives the best accuracy, followed by Manhattan distance, Euclidean distance and Chebyshev distance. This is kind of counter-intuitive that despite cosine similarity, distance that consider less dimension (Manhattan distance) performs better than distance that consider multiple and even infinite dimension (Euclidean distance and Chebyshev distance).

|  |  |
| --- | --- |
| Question 1 | How many Nobel laureates has Oxford educated ? |
| Answer 1 | The university is consistently cited as among the world's best.Oxford has educated many notable alumni, including 29 Nobel laureates, 27 prime ministers of the United Kingdom and many heads of state and government around the world. |
| Question 2 | When is Barack Obama born ? |
| Answer 2 | Obama was succeeded by Republican Donald Trump, who won the 2016 presidential election. |

Table 6: Examples of context-free approach using Squad dataset

|  |  |
| --- | --- |
| Question 1 | The Basilica of the Sacred heart at Notre Dame is beside to which structure? |
| Answer 1 | The Basilica of the Sacred Heart in Notre Dame, Indiana, USA, is a Roman Catholic church on the campus of the University of Notre Dame, also serving as the mother church of the Congregation of Holy Cross (C.S.C.) |
| Question 2 | What is Congregation of Holy Cross in Latin? |
| Answer 2 | The Congregation of Holy Cross or Congregatio a Sancta Cruce (C.S.C.) |

Table 5: Examples of context-free approach using Squad dataset

|  |  |
| --- | --- |
| Distance Metric | Accuracy |
| Cosine similarity | 0.6149162861491628 |
| Euclidean distance | 0.3995433789954338 |
| Manhattan distance | 0.43262481689453125 |
| Chebyshev  distance | 0.2815829528158295 |

Table 4: Accuracy of unsupervised learning base on different distance

Context-free unsupervised learning

In the below example, we developed an approach to with given context. Then we notice that the given context is not necessary as we could get the related context from web-based search engine. The context-free approach here uses a Python library for Wikipedia search engine to perform passage retrieval. And to avoid the ambiguity of the search term, I combine various search result into one context. At first, I try to use the questions in the Squad dataset, but the result is far from unsatisfying as shown in Table 5. This is mainly because of that many answers to the questions do not exist in Wikipedia page and some answer could also take up more than one sentence and thus our approach failed.

Then I tried with other question and some of the examples is shown in Table 6. We could reasonably conclude that the performance is largely dependent on the structure of the question. It the question is complicated, like the first question, the embedding of the question would contain more information and thus it is easier to find similar sentence. On the contrary, if the question is simple like question 2, the embedding may not clearly reflect the true meaning of it and thus this approach would easily fail.

Model selection for Infersent

As we have mentioned before, one key parameter of this project to select suitable dataset and model for sentence embedding. Infersent v1 use GloVe dataset and Infersent v2 use fastText dateset. Table 7 shows the accuracy on the same dataset using crawl-300d-2M dataset. We could find that the performance is limited and interestingly now Manhattan distance has the best performance. The difference may result from the difference between GloVe and fastText. GloVe treats each word as an atomic entity and generated the corresponding vector. However, fastText treats each word as composed of character n grams. Therefore, the vector for a word in fastText is made of the sum of the character n grams. The advantage of fastText such as better performance of rare and out of dictionary is of little use in this project. On the contrary, it does not treat word as atomic unit, which makes it difficult to measure the similarity between sentence embeddings.

|  |  |
| --- | --- |
| Distance Metric | Accuracy |
| Cosine similarity | 0.24885844748858446 |
| Euclidean distance | 0.2929984779299848 |
| Manhattan distance | 0.3036529680365297 |
| Chebyshev  distance | 0.24733637747336376 |

Table 7: Accuracy of unsupervised learning base on different distance

Question Classification

The second part of our project is question classification, which is the task of classifying a question based on expected answer. Question classification has shown to be significantly improve the performance of QA systems. As an example, the question “Who is teaching this lecture?” can be assigned a class of “PERSON”. Here, “PERSON” is a named entity or tag.

In this project, we extract answers based on 18 named entities defined in the OntoNotes corpus. To match the entities that will be extracted from the sentence containing answers, we classified questions into the same 18 classes and used hand-written rules listed as follows:

* Queries starting with “Who” or “Whom” are taken to be of type “PERSON”;
* Queries starting with “Where”, “Whence”, or “Whither” are taken to be of type “LOCATION”;
* Queries starting with “How few”, “How great”, “How little”, “How many” or “How much” are taken to be of type “QUANTITY”;
* Queries string with “Which” or “What”, look up head noun in lexicon to determine answer type.

As for the last rule, to find the head noun, we used StanfordPOSTagger to tag each word with part-of-speech tag. Then we find the head noun by extracting the first “NN” after “What/Which”. In this step, we also eliminate the first “NN” that is a part of preposition phrase.

To determine the type of question with head noun, this project used hypernyms from WordNet corpus. The idea is to take the hypernyms as the parent class entity of this head noun, and check whether head noun itself or its parent or its grandparent is in the 18 name entities by using regex expression. Theoretically, we can trace back to the root of hypernym, but actually this will consume large amount of time, so we decided to stop tracing at the grandparent layer.

In addition to these rules, we also add a pre-defined entity-to-word map. There are a series of words that can be easily classified into an entity. Therefore, we can directly check the map with the head noun as checking a dictionary to classify questions.

There are totally 1314 questions in our dataset. Based on the rules described above, we can determine the exact type for 75.8% questions. For the left part, we define their types as “OTHER”.

Supplementary Material

ACL 2018 also encourages the submission of supplementary material to report preprocessing decisions, model parameters, and other details necessary for the replication of the experiments reported in the paper. Seemingly small preprocessing decisions can sometimes make a large difference in performance, so it is crucial to record such decisions to precisely characterize state-of-the-art methods.

Nonetheless, supplementary material should be supplementary (rather than central) to the paper. **Submissions that misuse the supplementary material may be rejected without review.** Essentially, supplementary material may include explanations or details of proofs or derivations that do not fit into the paper, lists of features or feature templates, sample inputs and outputs for a system, pseudo-code or source code, and data.

The paper should not rely on the supplementary material: while the paper may refer to and cite the supplementary material and the supplementary material will be available to the reviewers, they will not be asked to review the supplementary material.

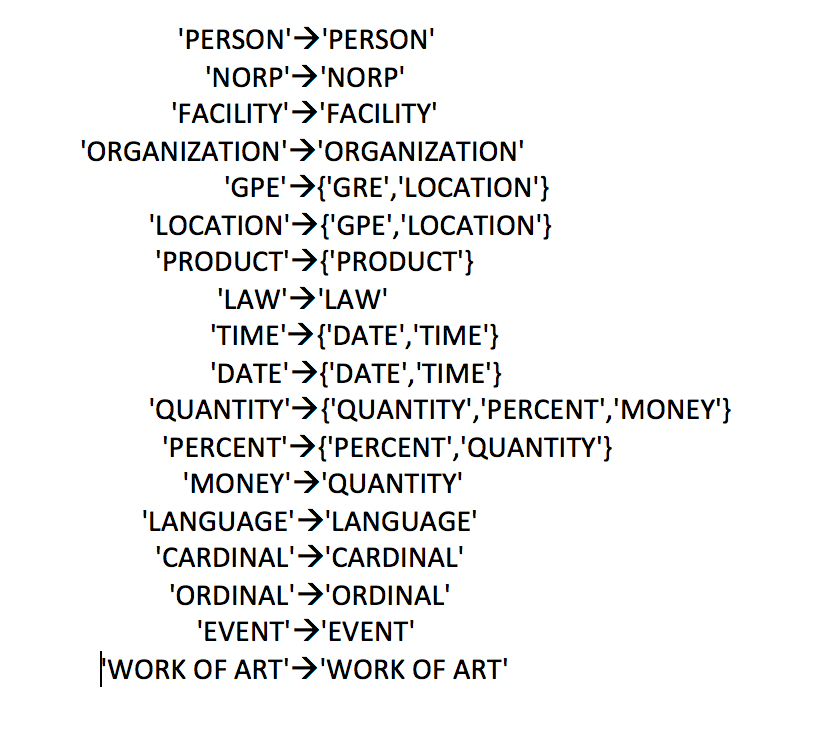
The supplementary material does not form part of the paper, does not count towards the page limit, and should not be included in an “Appendix” section following the references in this template. The “container” for supplementary materials is a separate document, and such materials should be submitted separately from the paper using the appropriate fields on the review form and the camera-ready upload form.

# Tag-matching and Building Answer

  By the previous steps, we can get the two .csv files: one is train\_detect\_sent.csv, which mainly includes context, the corresponding question and the target sentence (got from unsupervised learning) that contains the expected answer; the other one is question\_classification.csv, which contains the tag for question corresponding to each context in train\_detect\_sent.csv, respectively.

**6.1  Mapping Question Tag To All The Related Tags**

Since the step of tagging every question is not as accurate as we expect, for example, if we tagged a question as “DATE”, the expected answer might be a time, so we map the question tag to all its possible related tags, below is the coding part to do this mapping:



**8.2  Tagging Words in Target Sentence**

The next step is we tagged each word in target sentence by using CogComp-NLP, which provides a suite of state-of-the-art Natural Language Processing (NLP) tools that allows you to annotate plain text inputs. By matching each word’s tag with questions’ tag, if they are same, adding the word into our final answer list.

**8.3  Accuracy**

  Finally, let’s talk about how to calculate the accuracy of the model. From the SQuAD dataset, we can get the correct or expected answer for the corresponding question. The first method we use to calculate the accuracy is doing full-match, which means all the words in expected answer shown in the answer list we get. We got accuracy 0.295, which is pretty low.

Then we adjust the method by using partial-match, which means we classify the answer is “true” even if only one word in answer list shown in correct answer. After adjusting, the accuracy of our method increases to 0.38163, which is much better than before.

Acknowledgments

The acknowledgments should go immediately before the references. Do not number the acknowledgments section. Do not include this section when submitting your paper for review.

References

Alfred. V. Aho and Jeffrey D. Ullman. 1972. *The Theory of Parsing, Translation and Compiling, volume 1*. Prentice-Hall, Englewood Cliffs, NJ.

American Psychological Association. 1983. *Publications Manual.* American Psychological Association, Washington, DC.

Ashok K. Chandra, Dexter C. Kozen, and Larry J. Stockmeyer. 1981. [Alternation](http://dl.acm.org/citation.cfm?doid=322234.322243). *Journal of the Association for Computing Machinery*, 28(1):114-133. https://doi.org/10.1145/322234.32224.

Association for Computing Machinery. 1983. *Computing Reviews*, 24(11):503-512.

James Goodman, Andreas Vlachos, and Jason Naradowsky. 2016. [Noise reduction and targeted exploration in imitation learning for abstract meaning representation parsing](http://aclweb.org/anthology/P16-1001). In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, pages 1–11. <https://doi.org/10.18653/v1/P16-1001>.

Dan Gusfield. 1997. *Algorithms on Strings, Trees and Sequences*. Cambridge University Press, Cambridge, UK.

Mary Harper. 2014. [Learning from 26 languages: Pro- gram management and science in the babel program](http://aclweb.org/anthology/C14-1001). In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*. Dublin City University and Association for Computational Linguistics, page 1. <http://aclweb.org/anthology/C14-1001>.

Xin Li and Dan Roth. 2006. Learning question classifiers: The role of semantic information. Nat. Lang. Eng.,12(3):229–249, September.

Sundblad, H., 2007. Question classification in question answering systems. Linköpings Universitet.

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, December. The COLING 2016 Organizing Committee.

|  |  |
| --- | --- |
| 600  601  602  603  604  605  606  607  608  609  610  611  612  613  614  615  616  617  618  619  620  621  622  623  624  625  626  627  628  629  630  631  632  633  634  635  636  637  638  639  640  641  642  643  644  645  646  647  648  649 | 650  651  652  653  654  655  656  657  658  659  660  661  662  663  664  665  666  667  668  669  670  671  672  673  674  675  676  677  678  679  680  681  682  683  684  685  686  687  688  689  690  691  692  693  694  695  696  697  698  699 |