UNDER REVIEW

RESCIENCE C

Replication / ML Reproducibility Challenge 2022

Reproducibility report for the paper "Two is Better than Many? Binary Classification as an Effective Approach to Multi-Choice Question Answering (EMNLP 2022)"

Vasileios Ntaoulas[1] and Emmanouil Kalyvas[1]

[1]Aristotle University of Thessaloniki

Edited by
Deepanway G., Navonil M., Rada M., Soujanya P.

Reviewed by
(-)
(-)

Received
03 February 2023

Published
22 Oct 2022

DOI
10.48550/ARXIV.2210.16495

Reproducibility Summary

*This is a reproducibility report for the paper '**Two is Better than Many? Binary Classification as an Effective Approach to Multi-Choice Question Answering**' by Deepanway Ghosal, Navonil Majumder, Rada Mihalcea and Soujanya Poria. The authors of this paper suggests an improved method, they call TEAM method (two is better than more), to try and solve the multi class question answering (MCQA) problem.*

Scope of Reproducibility — The original paper claims that the suggested TEAM method, surpasses the classical MCQA score-based method in terms of accuracy. The objective of the reproduction is to try and support the claim by executing the scripts provided by the authors.

Methodology — In order to reproduce this paper, we used the authors' code, but we made our own selections over the values of the training parameters. We chose wisely different types of datasets in order to show how the authors' method works for different question-answer instances.

For our first experiments we used the free version of Google Colab, which gives us access to some shared GPU so we don't know exactly home much VRAM was available to us.Then, we used Google Colab Pro to be able to execute some more experiments a little faster. With this approach, we had available 40GB of VRAM.

Results — We executed six experiments in order to check if the TEAM-based method is better than the classical SCORE-based one. Only half of our experiments (three out of six) agreed with the original paper in terms of which method performs better using specific dataset and pre-trained language model.
All in all, based on the original paper, five out of six experiments should have supported the idea that the TEAM-based method is better while in reality only two out of six actually did.

What was easy — It was quite easy to understand the whole idea of this paper and the reason they decided to do such a work. Also, their implementation is described clearly and

Copyright © 2023 Anonymous, released under a Creative Commons Attribution 4.0 International license.
Correspondence should be addressed to ()
The authors have declared that no competing interests exists.
Code is available at https://github.com/MnAppsNet/TEAM$_{T}EST_{R}UNS$.

was not difficult at all to uderstand the differences between the classic MCQA method. Finally, the datasets they chose to evaluate their implementation was described also clearly.

What was difficult — Firstly, it was difficult to understand the whole repository, because there was no explanation or comments inside their python files and the code was not clearly written. Also, the parameters they used for training their model are not mentioned. But the final and most difficult part was the reproduction of the results. The datasets that the authors used were too big and we didn't have the resources to run such large processes.

Communication with original authors — We contacted with one of the authors, in order to explain to us the difference between the two accuracy metrics that their code generate. The author answered back immediately and he was really helpful.

1 Introduction

One of the major challenges encountered in question answering is the evaluation, which often requires human input to evaluate the textual answers thoroughly. Because of this, the alternative that has been proposed is that of multi-choice question answering, where the correct answer is provided together with other incorrect answers. The MCQA task is generally performed by scoring each (question, answer) pair normalized over all the pairs, and then selecting the answer from the pair that yield the highest score.

In this paper [1], the task of multi-choice question answering is reformulated as a binary classification task and show that this re-framing leads to significant performance improvements on several datasets. This is done using the method TEAM (**T**wo is b**E**tter th**A**n **M**any). Our work was to try and reproduce a piece of that work, check if we came into the same conclusions and suggest or implement further improvements on the authors' method.

2 Scope of reproducibility

The paper in review is dealing with the multi class question answering problem (MCQA). It is suggesting an alternative method to tackle the problem which claims that is producing better results than the classical approach followed. The authors refer to this classical approach as the score-based method. The newly suggested model is named TEAM (**t**wo is b**e**tter th**a**n **m**ore) by the authors.
The authors make the claims below about their TEAM model :

- It exceeds the performance of the score-based method traditionally used in the past
- It can be used in the more natural open-ended answer generation setups, thus providing a "bridge" between the MCQA and answer generation frameworks for question answering

3 Methodology

In order to reproduce this paper, we used the authors' code, but we made our own selections over the values of the training parameters. We chose wisely different types of datasets in order to show how the authors' method works for different question-answer instances.

Except from the authors' work, we also used this GitHub repository in order to understand the classic MCQA method.

The model training part and the evaluation part was done using Google Colab Pro platform.

3.1 Model descriptions

Our experiments are based on the two models that the authors present. The first one is the classic Multi-Choice Question Answering (MCQA) method. This approach is denoted as SCORE. The second one is a refactoring of the first method as a series of binary classifications, which the authors called TEAM.
Next, we describe those two methods:

SCORE-BASED METHOD

Given question q, optional context c, and the answer choices $A = [a_1, a_2, ..., a_n]$, n different input sequences are constructed each containing the concatenation of the question q, context c, and one possible answer choice a_i. The sequences are independently encoded through a pre-trained transformer language model such as RoBERTa (Liu et al., 2019) or DeBERTa (He et al., 2021). A score s_i is predicted for each input sequence which is then normalized with a softmax layer across the n outputs to obtain score q_i. The cross-entropy loss is used to train the encoder model. Assuming the answer a_k is correct, the loss can be obtained as follows:

$$L = \sum_{i=1}^{n} p_i log(q_i) = -log(q_k) \tag{1}$$

where p_i are considered as the class labels. The class p_k corresponding to the gold answer a_k is valued as 1, and all other classes are valued as 0. The choice providing the highest score is the predicted answer during inference.

TEAM-BASED METHOD

Firstly, the pre-trained language model is extended by adding a classification head with two nodes. The values of these two nodes will denote the unnormalized scores for the negative and positive classes in the classification setup. n different input sequences are constructed by concatenating the question q, the optional context c, and each possible answer choice ai. Then, the unnormalized negative and positive scores si- and si+ are obtained for each sequence by independently encoding them through the modified language model. Each pair of scores is normalized through a softmax layer to obtain probabilities of negative and positive classes: qi- and qi+. The sequence corresponding to the gold answer ak is considered as positive, and all the other sequences as negative. The loss function takes the following form:

$$L = -\sum_{i=1}^{n} (p_i^+ log(q_i^+) + p_i^- log(q_i^-)) = -log(q_k^+) - \sum_{i=1, i \neq k}^{n} log(q_i^-) \tag{2}$$

where p_i^+ and p_i^- are considered as the class labels. As a_k is the gold answer, we have p_k^+ = 1, p_k^- = 0 and p_i^+ = 0, p_i^- = 1, when i $\neq k$.

3.2 Datasets

We use the RoBERTa Large (Liu et al., 2019) and DeBERTa Large (He et al., 2021) models to benchmark the Score and TEAM method across the experimental datasets. The paper uses 12 different datasets to evaluate the implemented methods, but because of lack of resources, we used only 3 of them, which are not familiar with each other.

CommonsenseQA (Talmor et al., 2019) or CQA is a dataset for commonsense QA based on knowledge encoded in ConceptNet (Speer et al., 2017). Given a question, there are five possible choices , among which only one is correct. Any additional knowledge or context for this task is not used.
CommonsenseQA 2.0 (Talmor et al., 2021) or CQA2 is a recent challenging QA dataset collected with a model-in-the-loop approach. The dataset contains commonsense questions from various reasoning categories with either yes or no answer.
QASC (Khot et al., 2020) or **Question Answering via Sentence Composition** task requires fact retrieval from a large corpus and composing them to answer a multi-choice science question. Each question q has eight choices, among which one is correct. We use the question and choices without any retrieved facts for this task.

The size of train, test, validation sets for SCORE and TEAM methods are shown in the below table:

Dataset	Train	Test	Validation
CQA	9741	1140	1221
CQA2	9264	2473	2541
QASC	8134	920	926

3.3 Experimental setup and code

For our experiments, we used a simple setup with a shared Google Colab notebook in which we executed the scripts provided by the authors. In the notebook, we clone the repository mentioned in the paper, install the python dependencies and execute the scripts. We went with this approach because we need a lot of resources to fine-tune a pre-trained language model like RoBERTa or DeBERTa which we didn't have. The free version of Google Colab is providing enough resources just to execute the scripts on small datasets and with a small batch size . Of course this required a lot of time to train the models. For our experiments we used Google Colab Pro which uses more resources for a limited time and gave us a little more flexibility to execute more experiments.
The Google Colab notebook can be found at the following link:

github.com\MnAppsNet\TEAM_TEST_RUNS.

3.4 Computational requirements

The experiments we executed are very resource intensive tasks and require a GPU with a lot of video ram (VRAM). For our first experiments we used the free version of Google Colab which gives us access to some shared GPU so we don't know exactly home much VRAM was available to us. The time of each script execution using this method was about 1 to 2 hours. We should also note that with this method we were unable to execute the score-based method using the DeBERTa model as it was always requesting more VRAM than Google allowed for its free version of Colab.
Then we used Google Colab Pro to be able to fine-tune also the DeBERTa using the score-based method and execute some more experiments a little faster. With this approach we had available 40GB of VRAM which also gave us the the possibility to increase the batch size and execute the scripts in about 0.5 to 1 hour.

4 Results

Due to luck of resources, we were unable to execute all the needed experiments in order to support the authors claims. That being said, the results we managed to produces, does not seem to always agree with the authors.

4.1 Results reproducing original paper

In this section we will list all the experiments we successfully executed in order to investigate the authors' claim that the TEAM-based method surpasses the score-based method in terms of accuracy. All our experiments involve executing the provided scripts using both methods with the same parameters so we can check which one performs better.

Experiment 1 – We used the **CQA dataset**, the **RoBERTa-large** pre-trained model, a **learning rate of** 10^{-6} and **5 epochs**. We conclude that TEAM has a better accuracy than the SCORE-based method for this case.

Method	Our results	Paper results
TEAM-BASED	**74.28%**	**75.32%**
SCORE-BASED	65.36%	73.63%

Experiment 2 – We used the **CQA2 dataset**, the **RoBERTa-large** pre-trained model, a **learning rate of** $3*10^{-6}$ and **5 epochs**. We conclude that the SCORE-based method has a better accuracy than TEAM for this case.

Method	Our results	Paper results
TEAM-BASED	51.71%	**55.83%**
SCORE-BASED	**54.62%**	54.76%

Experiment 3 – We used the **QASC dataset**, the **RoBERTa-large** pre-trained model, a **learning rate of** $3*10^{-6}$ and **5 epochs**. We conclude that the SCORE-based method has a better accuracy than TEAM for this case.

Method	Our results	Paper results
TEAM-BASED	47.52%	**57.24%**
SCORE-BASED	**49.14%**	53.46%

Experiment 4 – We used the **CQA dataset**, the **DeBERTa-large** pre-trained model, a **learning rate of** 10^{-6} and **5 epochs**. We conclude that the SCORE-based method has a better accuracy than TEAM for this case.

Method	Our results	Paper results
TEAM-BASED	76.66%	83.34%
SCORE-BASED	**82.06%**	**83.75%**

Experiment 5 – We used the **CQA2 dataset**, the **DeBERTa-large** pre-trained model, a **learning rate of** $3 * 10^{-6}$ and **5 epochs**. We conclude that TEAM has a better accuracy than the SCORE-based method for this case.

Method	Our results	Paper results
TEAM-BASED	**66.90%**	**68.38%**
SCORE-BASED	63.36%	67.37%

Experiment 6 – We used the **QASC dataset**, the **DeBERTa-large** pre-trained model, a **learning rate of** $3*10^{-6}$ and **5 epochs**. We conclude that the SCORE-based method has a better accuracy than TEAM for this case.

Method	Our results	Paper results
TEAM-BASED	55.94%	**74.35%**
SCORE-BASED	**73.33%**	71.74%

4.2 Results beyond original paper

Based on the experiments we executed, we come to a conclusion that is not mentioned by the authors. Our observation was that the TEAM model seems to over-fit on the negative class when there are a lot of answers for a question. This was observed in the QASQ dataset which has 8 answers for each question with only one being positive. The authors' scripts are also calculating the classifier accuracy of the TEAM method apart from the

actual accuracy of the QA pairs. The classifier accuracy is not reported on the original paper. Our observation was based on this classifier accuracy.

Bellow are the actual and classifier accuracy of the TEAM method for each of the experiments described on the section above. It is also reported the number of answers each question has for the used dataset.

Test Run	Actual Accuracy	Classifier Accuracy	Number of answers
CSQA - RoBERTa	74.28%	83.14%	5
CSQA - DeBERTa	76.66%	85.37%	5
CSQA2 - RoBERTa	51.71%	50.75%	2
CSQA2 - DeBERTa	66.9%	67.10%	2
QASC - RoBERTa	**47.52%**	**83.71%**	**8**
QASC - DeBERTa	**55.94%**	**85.06%**	**8**

5 Discussion

Unfortunately, our experimental results do not support exactly the claims of the paper. The reason for that could be the lack of resources that we had and also the lack of information about the parameters that the authors used for training their model. Despite that, we managed to make some experiments with the smallest datasets and came to some conclusions of our own.

5.1 What was easy

It was quite easy to understand the whole idea of this paper and the reason they decided to do such a work. Also, their implementation is described clearly and was not difficult at all to understand the differences between the classic MCQA method. Finally, the datasets they chose to evaluate their implementation was described also clearly.

5.2 What was difficult

At the beginning, it was difficult to understand the whole repository, because there was no explanation or comments inside their python files and the code was not clearly written. Specifically, they did not mention which metric they used from their method in order to compare their results against the classic MCQA method results. Also, the parameters they used for training their model are not mentioned.

Next, the final and most difficult part was the reproduction of the results. The datasets that the authors used were too big and we didn't have the resources to run such large processes.

5.3 Communication with original authors

We contacted with one of the authors, in order to explain to us the difference between the 2 accuracy metrics that their code generate. The author answered back immediately and he was really helpful. We also asked him to upload the scripts from the classic MCQA method, in order to see how they produced the results they present into their paper. He responded to our requested by uploading the script in their git repository.

References

1. D. Ghosal, N. Majumder, R. Mihalcea, and S. Poria. **Two is Better than Many? Binary Classification as an Effective Approach to Multi-Choice Question Answering**. 2022.