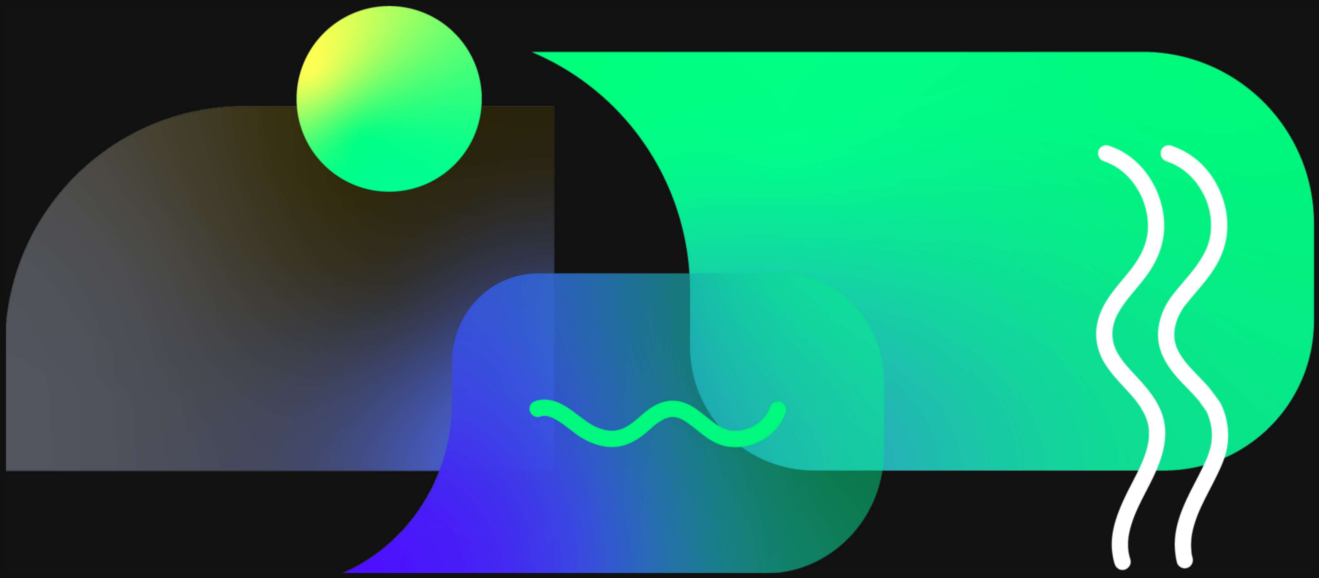
Inter IIT Tech Meet 11.0
IIT Kanpur



DevRev ● High Prep

Expert Answers in a Flash: Improving Domain-Specific QA

“Facing an issue? Chat with us.” We have seen this almost on all support platforms. This is followed by an automated bot message to seek initial information which further checks if this should be routed to a human support agent. Checking “If this should be routed to a human support agent” and providing useful suggestions to the user is an interesting and a tough challenge to solve and worth tons of dollars if done right.

DevRev High Prep

INTRODUCTION

DevRev in pursuit of bringing Developers closer to their customers (or Revs) is trying to make intelligent systems which can take care of mundane tasks present in a product lifecycle and minimizes manual intervention. One such use case is communication of support agents with users. There is an industry wide attempt to minimize the number of communication which reaches the support agent and let the system handle it initially while providing an automated response with some suggestions. In our case, we want to have a system which provides a Knowledge Base article (KB article) given a user query. For example, if the user comes up with a query about how payment API works, our system should be able to pull up a related API documentation which would be able to answer the query. And also highlight a part of the document which is the potential answer. Getting this done right for different distribution of documents makes it even more complex. In our example, it may happen that different organizations have different domains of KB articles and hence require fine tuning for each organization.

DevRev ● High Prep

PROBLEM STATEMENT

Task 1 Given a question and a set of paragraphs, predict if the question can be answered with the given paragraphs. If yes, return the paragraph that answers the question. Each question and paragraph is associated with a specific theme. This could be "Sports", "English" or "Mathematics" etc. A question of a theme can be answered by one of the paragraphs in that theme.

Task 2 For the given questions, also predict the exact answer from the predicted paragraph. Predict the start_index and the answer_text field for the given question. Note: Both the tasks will be marked individually. However, to perform better in Task 2, your model needs to perform better in Task 1.

Training dataset

The dataset contains the following fields:

1. Question: Question for which answer is to be found
2. Theme: Name of the domain this question & paragraph belongs to. For e.g. a paragraph could be on the theme "cricket" / "mathematics" / "biology" etc.
3. Paragraph: Paragraph from the mentioned theme which may contain the answer
4. Answer_possible: If the answer is possible from the given paragraph
5. Answer_text: Answers from the given paragraph
6. Answer_start: Index position from where the answer starts

Training dataset is available here:

https://drive.google.com/file/d/1Z-yb752A3o7b9dqrGt24XU0sl53FVqya/view?usp=share_link

DevRev ● High Prep

TESTING

**Please note that the mentioned testing format may change as per requirement.*

Round 1 Goal for the first round is to test how well your model performs for questions and paragraphs for any new theme. For example: your training dataset contained questions and paragraphs for the theme "Cricket". The model will be tested on a new theme, say "Football".

Test Input

1. CSV containing list of paragraphs.
2. CSV containing list of questions that should be answered from the given list of paragraphs.

Note: The test input will be shared with the participant and will be given enough time to fine tune for new themed paragraphs and run inference over new sets of questions.

Deliverables 1. **Training notebook** - Colab notebook that was used to train and export the models (Model can be exported to any choice of format).

2. **Inference notebook** - Collab notebook that uses the exported models and the test files to perform predictions and returns the predictions in the expected format.

3. The expected format is a **csv file** that contains the following fields:

a. Question

b. Paragraph (Empty if the question can't be answered from any given paragraph)

c. Answer_start: Start index of the answer that has been correctly predicted

d. Answer_text: Answer from the given chosen paragraph

Note: The result for this test set 1 will be released to the candidates and they will be provided enough time to fine tune their models.

TESTING

**Please note that the mentioned testing format may change as per requirement.*

Round 2 The goal for the second round is to see how well and efficiently you can fine tune the models for each theme after receiving sample question-answer pairs for that theme. Here, the test data contains questions and paragraphs from the theme that were shared with you as a training dataset and test dataset of Round 1. For e.g., you already have some question answer pairs for a given theme say "Kubernetes". How does the fine tuned model work for any new question related to the theme "Kubernetes"?

Test Input

1. CSV containing list of paragraphs.
2. CSV containing list of questions that should be answered from the given list of paragraphs.

Note: The test input will be shared with the participant and they will be given 2 hours to run inference and submit their deliverables. Teams are allowed to finetune their models on the data that was shared with them in Round 1.

Deliverables 1. **Training notebook** - Colab notebook that was used to train and finetune the models using previous datasets. The notebook should export the models. (Model can be exported to any choice of format).

2. **Inference notebook** - Colab notebook that uses the exported models and the test input to perform predictions and returns the predictions in the expected CSV format.

3. The expected format is a **csv file** that contains the following fields:

a. Question

b. Paragraph (Empty if the question can't be answered from any given paragraph)

c. Answer_start: Start index of the answer that has been predicted

KEY POINTS TO NOTE

● Training notebook when run over the given dataset must be able to reproduce the exported models used during inference for both Round 1 and Round 2.

● The results csv file should be named "<TeamName_predictions_1>" for Round 1 and <TeamName_predictions_2>" for Round 2.

● Any mismatch between the submitted csv file and the file generated from the inference notebook will lead to disqualification.

METRICS

● F1 score for paragraph search task.

● F1 score for QA task.

● There will be different weightage to different themes such that even if the model is overfit over publicly available QA dataset, it doesn't help much with the final score.

● Average inference time for each question must be less than 200ms. If average inference time is above that, it would be penalized accordingly.

● Your inference collab notebook must run within 4GB memory.

REPORT

Apart from the above mentioned deliverables, you would also be required to submit the mid-term and end-term report which should necessarily include the following:

● Literature review

● Different techniques evaluated and their metric score

● Final technique being used, latency and accuracy metric corresponding to it

● Future work

● References

The *tentative* date for **mid-term evaluation** is **10th January 2023**. Final dates will

DevRev ● High Prep

SCORING

Final score would comprise of following components:

- 25% - Midterm report
- 20% - F1 score for paragraph prediction
- 20% - F1 score for QA task
- Rest of the score would include
    - End term report
    - Code work
    - Presentation, QA round

REFERENCES

- Know What You Don't Know: Unanswerable Questions for SQuAD

https://arxiv.org/pdf/1806.03822.pdf

- F1 scores for QA task

https://www.tensorflow.org/hub/tutorials/tf2_semantic_approximate_nearest_neighbors

https://www.tensorflow.org/hub/tutorials/retrieval_with_tf_hub_universal_encoder_qa

- Fine tuning

https://deeplizard.com/learn/video/5T-iXNNiwls

- Faiss - Approximate nearest neighbours search

https://github.com/facebookresearch/faiss