This text describes a base-line solution for “Data-science exercise: Generate a FAQ bot”

Input

- Scraped pages of 4k-ultra-hd-tvs forum (data-science-faq-bot-task-sources)
- Exercise description: Mavenoid data-science task.pdf

Solution files

exctrat-data.ipynb - extracts forum discussions from scraped pages in two files:

- *questions.csv* and *comments.csv.*

answers.ipynb - tries to answer exercise questions and creates:

- *best_answers.csv* - that contains a row for each thread started in 2019 with the thread URL, the first post, and which of the five answer strings the bot should respond with.

Questions and answers

This section comments on solutions in *answers.ipynb*.

Pages not related to questions were filtered out during data extraction in *exctrat-data.ipynb*.

Also at the data extraction step the following additional attributes were added to forum question and comments objects:

Question object:

- *‘wifi-tv’* flag to select questions about WiFi-TV connection problems. Questions that have both WiFi and TV tokens in different variations of case and spelling are tagged as related to the connection problem.

Comment object:

- *‘rating’* - calculated as difference between ‘helpful’ and ‘not helpful’ reactions to the comment (see below).

Data exploration and other answers to exercise questions can be found in *answers.ipynb.*

Finding forum comments with the best answers to forum questions is a main goal of the exercise.

This goal is similar to the one used in Chat Bot systems which can be grouped in two categories:
- **Retrieval-based models:** use lookup tables or a knowledge base to select an answer from a predefined set of answers.
- **Generative models:** probabilistic models or models based on machine learning that generate responses on the fly instead of using a lookup-based approach.

Solution described here is based on the combination of these two model types. We do not have a predefined set of answers and scraped forum pages have very little information about usefulness of answers for user questions. Some, more or less useful, answers may be found in forum 'comments' though many comments have unrelated content, emotional or some other.

To find best answers we use **Multiple Choice Question Answering** model [1] based on BERT sentence embeddings. Model is implemented with PyTorch Sentence Transformers: Sentence Embeddings framework that "fine-tunes BERT / RoBERTa / DistilBERT / ALBERT / XLNet with a siamese or triplet network structure to produce semantically meaningful sentence embeddings that can be used in unsupervised scenarios: Semantic textual similarity via cosine-similarity, clustering, semantic search." [2]

Every forum question and comment is mapped only once to a fixed sized vector. Question (query) that we are looking the answer for is also mapped to a vector. In this setup, we only need to run BERT for one text fragment of a question (at inference), independent of how large our corpus is.

Then we calculate cosine-similiarity of the question vector to the answer vectors in the corpus. We define the best answer as a comment with embeddings that are closest (most similar) to our question embeddings.

This approach requires a good corpus of potential answers to select the best one that will match a query (question). As it was already said, in our case we have very little indication of what forum comments may be used as answers. There are some, very few comments marked by users as 'helpful' and some as 'not helpful' which does not necessarily indicates their usefulness as answers.

To estimate quality of our model, several sets of experiments were performed. All experiments can be grouped in the following categories of models used :

- Model trained on answer corpus built from all 2019 comments related only to WiFI-TV connection problem

- Model trained on answer corpus built from all *'helpful'* 2019 comments related only to WiFI-TV connection problem

- Model trained on answer corpus built from all 2019 comments

- Model trained on answer corpus built from all *'helpful'* 2019 comments

Each model was tested on a random set of questions from the same category (WiFI-TV or all) from year 2019 that have comments (answers) tagged as 'not helpful'. As expected models trained on a bigger corpuses found answers better related to questions then models trained on smaller corpuses.

Also model trained on 'helpful' comments gives better results then model trained on a bigger set including all comments from 2019. This is the main result of this work which shows that comments tagged as 'helpful' still may be of use to build a FAQ Bot.

Ideas for future work

The main goal is to create a model that will be able to find/select comments that in turn could be used in automatic generation of good answers. From this follows two tasks:

- Finding useful comments

- Generating FAQ answers from useful comments

FINDING USEFUL COMMENTS

Finding useful comments can be done manually or using at least some automation.

- The first thing to try is to use information about users when selecting comments with good answers.

This includes finding comments from special groups of forum users: 'leaders' and 'power users'. There is a need to find mechanism that tags their comments as accepted. Estimate answers in 'helpful' and 'accepted' comments provided by 'leaders' and 'power users'.

For each user, not only for 'leaders' and 'power users', calculate user rating based on the number of her 'helpful' and 'not helpful' comments, such as for example: $R = (H - N) / T$, where: R - rating, H - number of helpful comments, N - number of NOT helpful comments and T - number of all comments from the user. Start with a model trained on comments with rating R close to 1, then find best user rating threshold resulting in best answers.

- Other more complicated ways to analyse semantics of user comments, such as NER, relation and event extraction.

GENERATING FAQ ANSWERS FROM USEFUL COMMENTS

- Build Encoder-Decoder LSTM model and train it on set of pairs (question, good answer), Find answers for new questions using the trained model. Compare results with the ones from Multiple Choice Question Answering from this work.

- Use Multiple Choice Question Answering model [1] based on BERT sentence embeddings as was done in this work. Yet in this case trained on corpus of comments with better answers, found at previously proposed step. Use additional pre-trained corpuses such as based on SWAG[3]

- Build a Question Answering system using transfer learning and based on pre-trained Transformer models from BERT.

References

[1] Multiple Choice Question Answering
https://en.wikipedia.org/wiki/Multiple_choice

[2]Sentence Transformers: Sentence Embeddings using BERT / RoBERTa / DistilBERT / ALBERT / XLNet with PyTorch
https://github.com/UKPLab/sentence-transformers

[3] SWAG (Situations With Adversarial Generations)
https://www.kaggle.com/jeromeblanchet/swag-nlp-dataset