Natural Language

First Project

A retrieval-based chatbot

Técnico, Alameda

2019

1) Group identification

Group number: 15

Group members names:

- André João Beja Guerra (nr. 86382)
- Mark Baltič (nr. 94859)
- António Manuel Peres Celorico Borba da Silva (nr. 97096)

2) Introduction

Given a knowledge base (KB) based on a validated XML file containing questions and answers, we were requested to build a retrieval-based conversational agent type of "chatbot", that, in response to a user request, searches the most "similar question" from the unchanged KB and returns the ID of its answer. The "chatbot" is implemented in "Python 3" and the program invocation is compliant to the one described in the assignment, with an option parameter to indicate a different output file:

```
% python3 chatbot.py <KB.xml file> <test.txt file> [<results.txt file>]
```

3) Proposal

In order to achieve the project goals, we have considered the following approach and steps to take our chatbot to its final form.

Approach:

Besides the script referred in the previous point which directly answers the project goal, we have created a second script (chat_bot_accuracy.py) which evaluates the accuracy for any given configuration. The configuration is specified in a configuration file (configuration.py) which is common to both scripts, and while with the later, it is used to display the chatbot accuracy, with the former it used to find the ID of the most "similar question" – if any.

The configuration file considers the choice of a distance evaluator – currently EDIT, JACCARD and DICE are supported, the selection of a sequence of pre-processors to apply – those referred ahead in the text, the list of stop words to consider – should the respective pre-processor be applied, and a threshold level – only relevant for the "chatbot.py", which specifies the maximum distance up to where the most "similar question" shall be considered meaningful.

For the accuracy evaluation, the DS splits the original dataset into two parts, randomly reserving one question from each FAQ for the test set and the remaining ones for the training set. The accuracy script then supplies the DS with the pre-processors indicated in the configuration file, request DS the training set and the test set, and calculates the accuracy based on the true and false positives occurred. By default, it loops 10 times, supplying each individual accuracy value and the mean of the set. The accuracy results for the different configurations tested are detailed ahead.

Steps:

Transforming data

The first step was to parse the supplied XML file and feed it into an entity that we named a “data source” (DS), whose single responsibility is to handle the KB data. This means handle null entries, handle duplicate information, and whatever other data manipulation is found necessary to be able to provide consistent streams of data, and to supply KB views as required by the rest of the chatbot and other related scripts.

The DS also accepts destructive commands – destructive in the sense that the original data set is changed – for the application of pre-processors to improve the accuracy of the chatbot. In this way, a clear interface is defined that enables future development, test and application of additional pre-processors, which maintains an open door for evolution in a controlled way.

One key challenge on the development was to keep the whole system’s flexibility with a reasonable level of resources consumption – ex.: without multiplying copies of answers per each question. We have achieved this by carefully maintaining a linkage between a question and its FAQ of origin. This is of importance when the list of questions is requested to the DS, and later it is necessary to retrieve the FAQ of origin of that question (the ID is contained in the FAQ). The referred linkage is maintained by having a tag list, that maps one-to-one with the list of all questions, so that from the index in the questions list the origin FAQ can be retrieved.

Preprocessing

We considered multiple approaches for the preprocessing method where we wanted to put questions in a suitable form for the model. Unifying the alphabet case seems a good approach for this specific application, so we lowercased text under evaluation and then considered three approaches.

- Punctuation and letter marks removal to try not to penalize the QA distance in case of certain types of spelling errors or text marking.
- Tokenization and stemming to try to improve distance calculation for different forms of reference to the same term – even in case of some nonexistent forms in the KB.
- Removal of “stop words” to try to improve the signal-to-noise ratio by removing words that don’t add to the meaning of the question. The list of stop words used in our model is listed below, although the set of questions that we used is far too short and too related with the original questions for it to be accurate, especially with unrelated questions:

  StopWords = ['o', 'teu', 'seu', 'se', 'caso', 'quando', 'qual', 'como', 'a', 'após', 'até', 'de', 'em', 'para', 'que', 'com', 'é']

Distance calculation

So, the next aspect to consider in the quest to identify the best answer to a given question, is which distance evaluation algorithm performs better. We tried three different measurements: edit distance, Jaccard distance and Dice distance. The Edit and Jaccard distance algorithms implementations used are the ones from NLTK. For the Dice distance calculation – not available in NLTK, the Jaccard coefficient is derived from the Jaccard distance, the Dice coefficient is derived from the Jaccard coefficient, as indicated in the following formula, and the Dice distance is derived from the Dice coefficient. The idea is clearly explained in https://en.wikipedia.org/wiki/S%C3%B8rensen%E2%80%93Dice_coefficient

$J = 1 - D_J \qquad S = \frac{2J}{1+J} \qquad D_S = 1 - S$

Where D_J is the Jaccard distance, J is the Jaccard coefficient, S is the Dice coefficient and D_S is the Dice distance.

Threshold for meaningful distance

One of the requirements of the assignment is to return the ID=0 in case the closest question is not related with the user request. The approach take was to use trial-and-error to define a distance threshold from where the relation between the question and the answer stop making sense. When the model determines the KB question with the shortest distance to the user request and finds it larger than the threshold defined in the configuration file, instead of reporting the matching ID, it reports ID=0, and an adequate "answer not found" message.

4) Corpora

For HW2 we created a part of the final knowledge base. For each FAQ there should be four questions with the same answer in the XML file. The data we were given for the project is an XML file resulting from merging the contributions of all groups.

To detect and deal with abnormal/frontier situations, and to prepare the DS accordingly, we prepared multiple "unit test" to identify, characterize and resolve these situations in the dataset. The tests revealed the existence on empty questions entries – those entries were discarded, data duplication – the first of each was considered, the others were discarded.

5) Experimental results

We tested our model the following way: we divided the dataset into a train and test set. Test set was built by randomly taking one question for each FAQ (thus removing that question from train set). Then we used the train set as our knowledge base and tried to predict the correct answers for the questions from the test set. We calculated the accuracy of the classification (number of correctly assigned answers divided by the number of questions in the test set). We repeated this step 10 times and averaged the results. We changed the model to obtain the best combinations of distance and preprocessing. The results are in the following tables:

Table 1: Accuracy of models. (mr - letter marks removal, ts - tokenization and stemming, sw - removing stopwords)

Number	Pre-Processing Techniques	Edit	Jaccard	Dice
1	___ , ___ , ___	0.62	0.72	0.73
2	___, ___, sw	0.68	0.79	0.80
3	___, ts, ___	0.66	0.78	0.79
4	___, ts, sw	0.7	0.87	0.87
5	mr, ___, ___	0.65	0.77	0.77
6	mr, ___, sw	0.67	0.8	0.8
7	mr, ts, ___	0.64	0.8	0.79
8	mr, ts, sw	0.69	0.84	0.83

According to the results we discard the models which use the edit distance. We pick 4 models: the preprocessing techniques 4 and 8 with Jaccard and Dice distance and start considering the threshold for unknown answers. We construct a set of 30 questions (we searched for common Portuguese questions on Google; we then made some more questions to enlarge the set; we also provided some questions in English, which should have nothing in common with our knowledge base, and use it as an additional set for testing the threshold. So, we want to have a high accuracy at the test set (which is constructed same as before) but at the same time a high number of "0 labels" answers in the additional set (of common questions).

Table 2: Performance of the models according to different tresholds t. The values are presented as A\Z, where A is the accuracy on the test set and Z is the fraction of zero id's from the set of 30 questions.

Number	Model	t=0.79	t=0.68	t=0.62	t=0.55
1	___, ts, sw + Jaccard	0.87\0.2	0.86\0.7	**0.86\0.9**	0.82\0.97
2	mr, ts, sw + Jaccard	0.83\0.67	0.83\0.87	0.8\0.9	0.74\0.93
3	___, ts, sw + Dice	0.87\0.0	0.87\0.13	0.86\0.27	0.86\0.63
4	mr, ts, sw + Dice	0.84\0.23	0.84\0.63	0.82\0.7	0.82\0.87

6) Conclusions and future work

It seems reasonable that the edit distance preformed the worst as it tries to transform one question to another in as few steps possible. But the meaning of a question is not hidden in the form of the sentence but in the words, which are crucial for the question. Therefore, Jaccard and Dice produce better results, as they treat sentences as sets of words and just compare the number of similar words. From Table 1 we can see, that preprocessing is important for the model. The removal of punctuations becomes less important than the other two. But as we combine all three of the preprocessing methods the results are good as well.

For choosing the best threshold we took 4 best models from previous step and calculate 2 values for each model according to changing threshold. If we look the first method in Table 2, we can see that moving from t=0.79 to t=0.68 the accuracy of the model does not change a lot, but the fraction of correctly labeled questions in the set of irrelevant questions enlarges for 0.5. That means that the questions (from the train test set) that were labeled wrong in the first example are at a large distance, so the model now marks them as unknown and they stay wrong – thus not changing the accuracy. We can observe this phenomenon in most of the cases in the Table 2. The exception are the first 2 models, when we change threshold from 0.62 to 0.55 – here the accuracy drops which means, that the filter for unknown answers is "too strong".

One more expected property of Table 2 is that the accuracy numbers decrease as we lower the threshold, as more questions from test set are set to 0. In Table 2 we see a big difference between Jaccard and Dice distance. As they preform similarly when used only on test set, the Jaccard is better at distinguishing the unknown answers from known ones.

As we want to maximize accuracy and the proportion of correctly labeled "unknown" questions, we choose the best model to be the model number 1 with the threshold t=0.62.

If we had more time and a large base of questions, we would try more different sets of "stop words" and improve the set of the unrelated questions (which were used for testing the best threshold value). We would also consider some other distance measures or came up with some variations of the known ones (for example: linear combinations of Jaccard and Dice distances). One more possibility would we, that we do not consider the tag of the closest question, but to find the closest two questions with the same tag and use their tag to return the corresponding id.

7) Bibliography

- Lectures notes
- Slides
- https://gist.github.com/alopes/5358189 (Stop words)
- https://en.wikipedia.org/wiki/S%C3%B8rensen%E2%80%93Dice_coefficient (Dice coefficient)