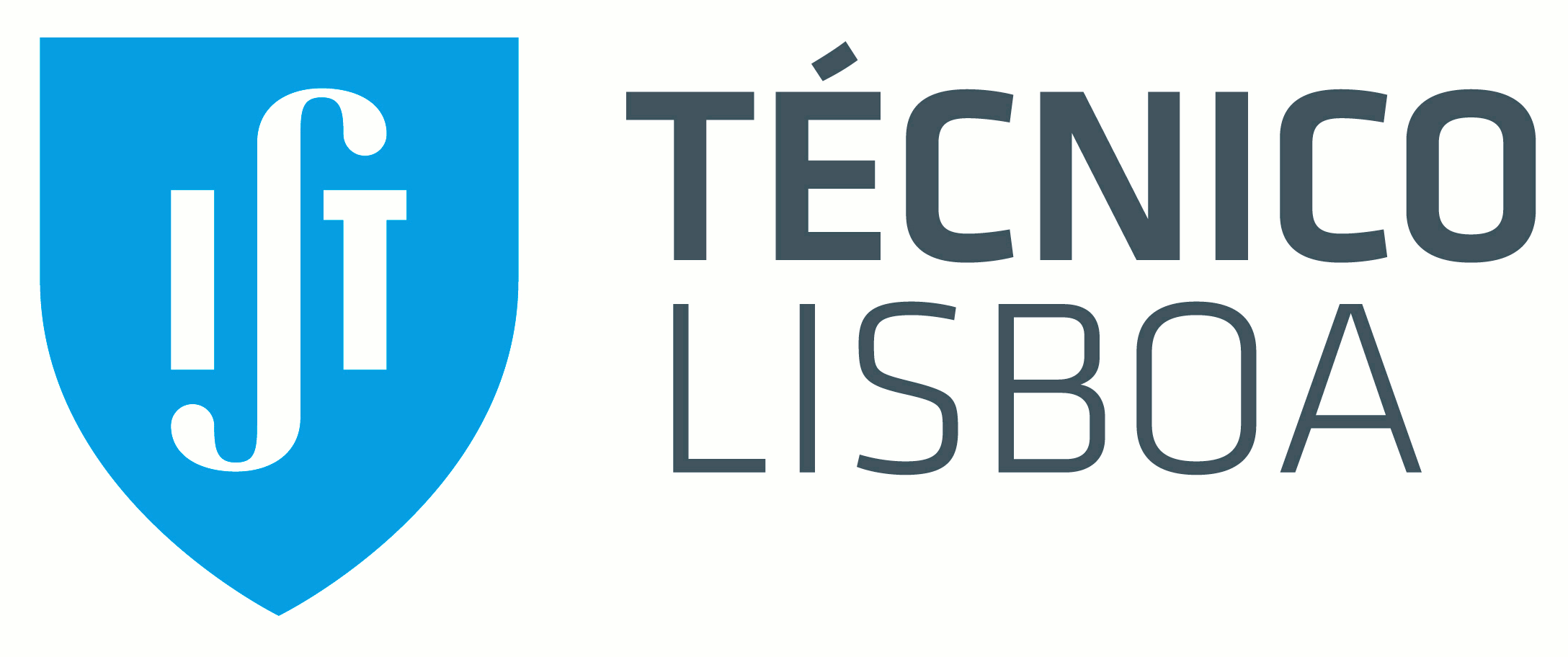
**Natural Language**

****

Scientific Report

First Project – A Retrieval-based chatbot

Group 7

Ricardo Ventura – 79031; Maxwell Junior – 79457; Isabel Raposo - 81524

**Introduction**

On this project we were proposed to build a retrieval-based chatbot with FAQs from “Balcão do Empreendedor”. These FAQs constitute the chatbot’s knowledge base (KB). By being given a user request, the chatbot should find the most similar question from the KB and return an ID corresponding to its answer. Therefore, this is a retrieval-based conversational agent as it doesn’t really reply an answer, nor does it perform any kind of additional interaction.

**Proposal**

After getting all data, properly parsed from the KB (getting all FAQs questions and respective IDs), we divided it in 2 data sets, the **Development** set and the **Test** set. For the **Development** set we chose 80% of the questions of each FAQ (the result was rounded using a floor function) and the remaining 20% went to the **Test**.   
The reason of this choice is because we noticed that the majority of FAQs would have 4 questions, therefore we get at least 1 question to the **Test** set. The only exception was for the FAQs with only 1 question, which was directly sent to the development set.  
  
We considered 4 preprocessing techniques (all use lowercasing, removal of punctuation and tokenization):

**P1** - standard procedure with no additional preprocessing

**P2** – with stemming

**P3** – with stop words removal

**P4** – P2 and P3 combined

and combined them with 3 metrics used, the Cosine similarity, the Jaccard distance and the Dice distance.

The **Cosine Similarity** was applied after vectorizing the sentences using the **TF-IDF** technique. This metric is very effective for evaluating sparse vectors, which happens in this case by having many terms in our corpora. With TF-IDF, the more frequent terms are considered less relevant, therefore when comparing sentences, the key terms will have more weight in the similarity measure.   
The **Jaccard distance** computes the percentage of words that don’t belong to both sentences, so we could get the Jaccard measure by doing 1-(Jaccard distance) and evaluate the similarity between sentences. This is a good technique when considering that the order of the words is not relevant, and all words have the same weight.  
The **Dice distance** is a variation of the Jaccard distance but considers the repetition of words from both sentences.

In order to choose the best metric and preprocessing technique, we tested all possible combinations for 10 different test sets and computed the average accuracy. The results obtained are shown below on table 1, on Experimental Results section.

**Used Corpora**

The corpora we used was based on a FAQ, on which we added more questions by paraphrasing the originals, and thus having a wider KB.

We found out that here are indistinguishable questions with the same answers, but these answers have different IDs. When searching the best answer Id for a question, we are probably getting some Ids correctly but the result is incorrect, which might lead to a decrease on accuracy (we’ll get into more detail on Experimental Results).  
Also, some FAQs only have 1 question what might have decrease accuracy

Due to the dimension of the KB we couldn’t be aware of some errors that might appear and could influence the results.

**Experimental Results**

The average accuracies from combinations of metric with preprocessing techniques can be checked on this table.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Metrics / Pre-processing** | P1 | P2 | P3 | P4 |
| Cosine + Tf-Idf | 67.94% | 67.34% | 70.03% | **70.58%** |
| Jaccard | 61.42% | 66.80% | 63.17% | 69.30% |
| Dice | 59.39% | 64.24% | 61.73% | 68.17% |

Table 1. Values on the table correspond to accuracy, resulting from an average of 10 runs of each combination of metrics and preprocessing techniques.

**P1** - standard procedure with no additional preprocessing   
**P2** – with stemming  
**P3** – with stop words removal  
**P4** – P2 and P3 combined

After analyzing the table, we could conclude some interesting results:

* P3 > P1
* P2 > P1 except on cosine
* P4 is the king
* Cosine > Jaccard > Dice
* Cosine + P4 is the best one therefore it’s the one implemented on chatbot.py

On general, removing stopwords lead to better results. However, there are some stopwords that could be useful to determine the similarity between sentences as they sometimes are the ones who could disambiguate the meaning of the question. For example, after analyzing the list of Portuguese stopwords we found “como” and “quando”. As there’s very similar questions that only differ on those words, we can conclude that, in this case, a stopword would help to find the right question. Furthermore, the list also includes the conjugation of “Ser” and “Estar” verbs which are often used on the questions of the KB.

Finally, we can conclude that the best combination is the **Cosine similarity + P4**. Even though it isn’t by much, there’s some evidence that using all the preprocessing techniques is the best choice and the Cosine similarity using TF-IDF is the best metric against Jaccard and Dice distance. For this reason, it was our choice to use this implementation on the chatbot.py file in order to compute the solution.

**Conclusion and future work**

**Bibliography**

|  |  |  |  |
| --- | --- | --- | --- |
| 0.6833 | 0.7077 | 0.6782 | 0.7067 |
| 0.6142 | 0.6317 | 0.6666 | 0.6930 |
| 0.5958 | 0.6215 | 0.6347 | 0.6793 |