The wizard: Naïve bayes approach and normalized cosine similarity for question-answering chatbot

Jonathan J. Allarassem
Department of Computer Science
Grove City College
Grove City, PA
AllarassemJJ20@gcc.edu

ABSTRACT
Given a database about a specific population (students, employees etc...), it is not obvious how we can we build an interface, such that given a question W containing the id w* of the row that has the info and some more information w_j we provide the proper answer C={c_i} i>0. This paper tackles this problem assuming the population is the student body of a college, and the id is the first and last name of a student. The approach of this paper is to use a Naive Bayes classifier and word-embeddings to determine the right answer to provide. Despite its simplicity, this approach does very well, especially when the number of columns is small, and the column of the database contains information about things that are different enough.

KEYWORDS
Question-answering, chatbot, database, word-embedding, Naïve Bayes Classifier

1 Introduction

The method described below is not trying to tackle general question answering problems but rather a very specific task. We have a database of n columns containing student's data. For the program to work, the user needs to know the first name and last name of the student and to ask a question whose answer might be in the database. These two assumptions still leave a lot of flexibility to the user. The approach will be to rank the different columns of the database based on how similar they are the question received by the interface. Columns are ranked based on how individual words are like the column's name. This assumes that the names for the different columns of the database reflect well enough their content (which is not always the case). This method also assumes that the question will contain the id to locate the right row to work with. In our example, the id will allow us to find the right student when the other words in the sentence will allow us to locate the right column.

2.1 Column ranking

Here we try to classify the columns based on their relevance to the question. Let c_i be a vector representing a column in a database, with $c_i \in C$. C being the set of all columns in the database. Let $W = \{w_0, w_1, w_2, \dots, w_n\}$ a set of vectors representing the embedding of each word in the sentence (a question more specifically) with w_i an embedding. We want to find $P(c_i|W)$ the probability that the user is looking for column c_i considering that he or she used the set of words W. This probability will be a score to rank the different columns of the database by relevance to the question asked.

$$P(c_i|W) = P(c_i|w_0, w_1, w_2, w_3, \dots, w_n)$$

Because of the bag of words assumption, we can suppose conditional probability between the words in the sentence. Hence:

$$P(c_i|W) = P(c_i|w_0)P(c_i|w_1)P(c_i|w_2) \dots P(c_i|w_n)$$

Which can be rewritten the following way:

$$P(c_i|W) = \prod_j P(c_i|w_j)$$

To avoid dealing with small numbers we can use the loglikelihood instead:

$$\log P(c_i|W) = \log \prod_j P(c_i|w_j) = \sum_j \log P(c_i|w_j)$$

Recalling the fact that each word in the sentence is mapped to an embedding w_j we can compute $P(c_i|w_j)$ as followed:

$$P(c_i|w_j) = \frac{P(w_j|c_i)\,P(c_i)}{P(w_j)}$$

Or if we assume that all words in W have the same probability of occurring as well as the columns then:

$$P(c_i|w_j) \propto P(w_j|c_i)\,P(c_i) = P(w_j, c_i)$$

We know that for two embeddings c_i and w_j the dot product offers a measure of how related they are. Hence using a SoftMax for example, we can get a normalized score for every column.

$$P(w_j, c_i) = \frac{e^{\cos(w_j.c_i)}}{\sum_k e^{\cos(w_k.c_i)}}$$

Consequently,

$$P(c_i|W) \propto \prod_j P(w_j, c_i) = \prod_j \frac{e^{\cos(w_j.c_i)}}{\sum_k e^{\cos(w_k.c_i)}}$$

To avoid dealing with very small numbers when doing the calculations, we can use the log likelihood instead of the actual probability metric:

$$\log P(c_i|W) \propto \log P(c_i, W) = \log \prod_j \frac{e^{\cos(w_j.c_i)}}{\sum_k e^{\cos(w_k.c_i)}}$$

$$\log P(c_i, W) = \sum_j \log \frac{e^{\cos(w_j.c_i)}}{\sum_k e^{\cos(w_k.c_i)}}$$

Consequently, we get the following:

$$\log P(c_i|W) \propto \sum_j \log \frac{e^{\cos(w_j.c_i)}}{\sum_k e^{\cos(w_k.c_i)}}$$

We can easily simplify the expression:

$$\sum_j \log \frac{e^{\cos(w_j.c_i)}}{\sum_k e^{\cos(w_k.c_i)}} = \sum_j (\log e^{\cos(w_j.c_i)} - \log \sum_k e^{\cos(w_k.c_i)})$$

$$\log P(c_i|W) \propto \sum_j (\cos(w_j.c_i) - \log \sum_k e^{\cos(w_k.c_i)})$$

From this we can pick the column:

$$\hat{c} = argmax_i \left[\sum_j (\cos(w_j.c_i) - \log \sum_k e^{\cos(w_k.c_i)}) \right]$$

This will give us the column. We get the vector representation using spacy and computing this expression for all columns. The column with the highest score is using for query. For computational reasons, we could even remove the second part of the expression $\log \sum_k e^{\cos(w_k.c_i)}$ since it will be the same for every value it will not impact on the final score. The final expression is then:

$$\hat{c} = argmax_i \sum_j \cos(w_j.c_i)$$

2.2 Gender classification

It is well known that there is a correlation between the ending characters of a subject and his or her sex. Using that we try to label the gender of a given set of names. Let S be the set of 3 last characters of a given set of names with $S = \{s_0, s_1, s_2, s_3, \dots, s_n\}$ and s_i a set of three characters. Let G be a binary variable taking the values 0 or 1 depending on whether the patient is female or male. We want to find $P(G|s_i)$ for every element of S which is the probability that a name is male or female. Using bayes rule, It is easy to derive:

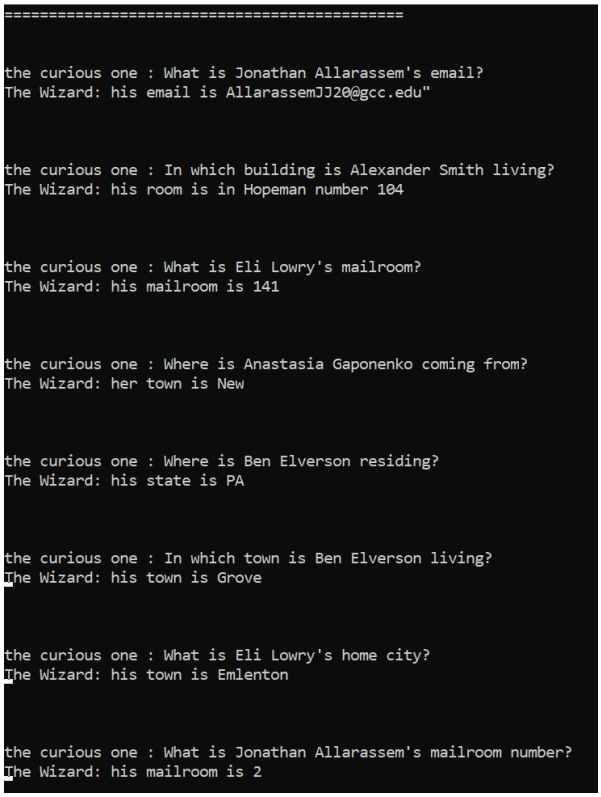
$$P(G|s_i) = \frac{P(s_i|G)\,P(G)}{P(s_i)}$$

If the probability of getting a man versus a female in the population is the same (.5) and the probability of getting different sequences of characters is the same, we can reduce this expression:

$$P(G|s_i) \propto P(s_i|G)$$

We can learn $P(s_i|G)$ easily using an already labeled dataset and then test the model on a new name. One advantage here is that it is very rare to get a name whose last characters' probability have not been learned yet. The gender classification is then achieved by looking up the learned probabilities for different sequences of characters. Using this method the chatbot will use the proper pronouns/

3. Samples Question and answers



4. Conclusion and future work

The system is not perfect unfortunately (see Q5 and Q6). Some issues are due to corrupted information in the database, but other inaccuracies occur from the fact that the two or multiple columns in the database had very high similarity to the question instead of just one column. For eg a question like “where does X lives?” could trigger the column about campus location or home location. So, the next step with this work will be to put some more rigorous bound on how much difference between the columns is needed for accurate results.