Abstract—For many years, the communication competences of a robot have been oversimplified due to its complexity. Over the last few years, there were a couple of breakthroughs in the Machine Learning and Natural Language Processing areas that brought a lot of new possibilities to human-robot interactions. However, much of the research and data made in these fields are in English, with other languages being mostly overlooked. In order to apply these techniques to the Portuguese language, a solution suitable to the available databases is investigated and the differences between both languages, such as the plural and gender endings and the word stemming, are taken into consideration. Here, the robot communicates with humans in Portuguese and the discourse should work in a reactive and rapid manner, considering and inspecting previous dialogues to build a conversational model as natural as possible. Bearing this in mind, the Latent Semantic Analysis, a Natural Language Processing technique, is used intertwined with a Naïve Bayes classifier to predict what the robot should respond based on the human utterance. The usage of a stop words list and a keyword extractor, like in the English research, are carefully inspected along with the system parameters to understand their influence on the final performance. A new approach to gather new Portuguese dialogues based on a form is also proposed since the quantity of data available is almost non-existent.

Index Terms—Natural Language Processing, Latent Semantic Analysis, Machine Learning, Multinomial Naïve Bayes Classifier, Social Robotics, Human-Robot Interaction, Keywords

I. INTRODUCTION

Humans are social beings who mingle in the community as the basis of the functionality of the society [1], encouraging social relationships and avoiding loneliness. However, groups, such as the elderly, cannot escape this feeling. Due to the aging problem present in almost all countries, this group is growing in the society and the isolation they endure urges them to seek emotional assistance [2] [4]. Robotics enters along these lines, as interactions with artificial machines can compensate the scarcity of human communication.

The development in Robotics during these last decades made it possible to look at conversations between humans and robots in a new light, with more complex and sophisticated procedures. In order to produce more refined communications, many different areas, being NLP (Natural Language Processing) and ML (Machine Learning) the most important, came together to work and created a new field of study - HRI (Human-Robot Interaction) [5].

Nevertheless, programmers often overlook the robotic communication due to the complexity of NLP tasks. The approach taken is usually simplistic, through template-based, rule-based or even hard-coded systems. However, recent progress in artificial intelligence steered away from these types of systems to increasingly use data-driven techniques, taking advantage of a better understanding of the language functionality and more efficient ML algorithms. These breakthroughs have greatly improved these areas, promoting its application to more complex problems even if the result isn’t still perfect.

The main obstacles in using these newer methods to create decent textual classifiers are acquiring the suitable quantity of data and numerically representing the textual information. In order to build a system that can work in every context and interaction, a huge dataset is needed so that the robot is sufficiently trained for every situation. Although there aren’t enough data to perfectly accomplish this system, it is already possible to obtain a good performance with a reasonable dataset and the right language processing and utterance generation methods [6]. However the majority of the research made in these areas is in the English Language, completely disregarding other languages, e.g. Portuguese.

Therefore, the main goal of this article is to develop a software for a robot that interacts with the elderly in a Portuguese speech-based dialogue. The dialogue should work in a reactive and rapid manner, considering and inspecting previous dialogues between elders and other people to build a conversational model as natural as possible. In order to achieve the principal goal, different techniques are to be researched and analysed to choose the best one and, due to the scarcity of Portuguese dialogues, it is also necessary to investigate a feasible approach to obtain new data so that the Software developed can be properly tested.

II. THEORETICAL BACKGROUND

After executing a deep research according to the quantity of data available, solutions including NLP and ML techniques are the most appropriate. The NLP method transforms the textual information uttered by a human into a numerical vector that correctly quantifies the sentence. Each sentence uttered by a human is followed by a robot utterance, so each NLP human vector and its corresponding robot discourse coincide with
each element of the input used in the ML classifier’s learning phase, so that the system is able to predict new human phrases.

In that sense, the solution chosen is NBC (Naive Bayes Classifier) [7] [8] with feature selection and extraction through LSA (Latent Semantic Analysis). The later algorithm is a NLP technique that comprises of preprocessing tasks to clean the textual information and then, applies SVD (Singular Value Decomposition) to the sentences by terms matrix, and reduces its dimensionality so that the most important dimensions are selected and a better data insight is achieved [9]. The NBC is used as a support for predicting which utterance the robot should respond, when triggered by human discourse [10]. The preference of NBC over other ML methods is mainly due to its rapidity and implementation facility. Nevertheless the enhancements in ML throughout the years made NBC undesirable when compared to other algorithms [12]. However, the LSA data preprocessing transformed the classifier into a more viable method, maintaining the NBC’s easy employment while increasing its accuracy [7]. Also, NBC is a technique that thrives under small datasets in opposition to more complex solutions, like neural networks or SVM (Support Vector Machine) [11]. The solution adopted will be subsequently explained.

A. Feature extraction and selection through Latent Semantic Analysis

LSA comprises of the following preprocessing steps:

1) Tokenization: Given a string, it splits them into tokens through a process of delimitation based on the blank space. Tokens represent words with a particular meaning or a punctuation sign. The latest are eliminated from the set.

2) Stop Words: Represented by a group of terms that are common in oral and written forms and, consequently, do not give important content to the sentence. These words should be eliminated from the previous set of tokens, however it is crucial to remark that there is n’t an objective list of stop words and, as such, the best choice is unclear and ambiguous. Nonetheless, it should include the terms that fail to provide extra value to the sentence.

3) Porter Stemming: It is the action of replacing each flexed or derived word in the set with its root, i.e. deleting the verb conjugations, the singular and plural aspect and also the gender factor if the language verifies gender-specific words. In this work the Porter Stemmer rules are to be used. This process is important as many words would be considered different by LSA despite their equal context without stemming.

4) Building of Text Frequency - Inverse Document Frequency matrix: A TF-IDF matrix is built from the term frequency (TF) multiplied by the inverse document frequency (IDF) [24]. In the end, a sentences×terms matrix is obtained and a term’s relevance rises with the increase of its weight in the phrase.

5) Singular Value Decomposition and Dimensionality Reduction: The next phase is to decompound the previous matrix, labeled A, into a product of three matrices according to SVD which is defined as follows:

\[ A = U \cdot \Sigma \cdot V^T, \]  

where A is a \( m \times n \) matrix, U is \( m \times m \), \( \Sigma \) is \( m \times n \) and V is \( n \times n \). \( \Sigma \) is a diagonal matrix, called singular value matrix as its non-zero cells represent the eigenvalues, each one linked to one particular dimension.

The need to reduce the dimensions of the feature space arises from the fact that if the irrelevant dimensions are withdrawn and the matrix is rebuilt into the initial dimensions, the obtained matrix is a least-square best fit. The resulted output has each cell changed due to indirect relations between all of the phrases, increasing their correlation if two terms appear in the same context and decreasing it when the opposite happens. It is through this approach that LSA performs induction [9].

Now, a criteria to evaluate which dimensions are selected is crucial and, as the magnitude of each eigenvalue is related to its importance on the feature space, the higher values should be chosen. The criteria picked is the percentage of cumulative singular values [13] as it is advantageous compared to others. For example, the number of dimensions can incorporate dimensions with low magnitude (if the number is too big) or exclude relevant dimensions (if too small) while the threshold value criteria can leave important dimensions out if the magnitudes of the eigenvalues are not really large enough or, if the results are high, too many dimensions are picked. In this way, the criteria selected leads to choosing always the same percentage of the most important, independently of the number of dimensions or its value, leaving out the ones that give little information to the task at hand.

B. Utterance generation through Naive Bayes Classification

The NBC is a ML algorithm that is based on the Bayes’ Theorem that follows:

\[ P(y|X) = \frac{P(X|y) \cdot P(y)}{P(X)}, \]  

where y is defined as the class label, having \( m \) possible outcomes, and X is a vector, composed of \( n \) features. Taking the equation 2 into account and assuming the independence between the features of the evidence X (\( x_1, ..., x_n \)), the final equation is obtained:

\[ P(y|X) = \frac{P(y) \cdot \prod_{i=1}^{n} P(x_i|y)}{\prod_{i=1}^{n} P(x_i)} \]  

After estimating the parameters of the classifier in the learning phase, the classifier is ready to predict a new specific X. For this purpose, the equation 3 is computed for all \( m \) class labels and, in the end, the class predicted is the one with higher probability. Since the denominator in equation 3 is always the same for every label it can be suppressed.

1) Multinomial Naive Bayes Classifier: Each of the attributes of X is related to a term of the vocabulary extracted by LSA and, since each feature has a multinomial distribution, MNBC (Multinomial Na"{i}ve Bayes Classifier) is the most desirable. The likelihood of observing a data \( X \) is given by the following equation:
$P(X|y) \propto \prod_{i} \hat{\theta}_{yi}^{x_{i}}, \tag{4}$

where $\hat{\theta}_{yi}$ represents the smoothed version of maximum likelihood (i.e. relative frequency counting) and it is computed through:

$$\hat{\theta}_{yi} = \frac{N_{yi} + \alpha}{N_{y} + \alpha n}, \tag{5}$$

where $N_{yi} = \sum_{x \in S_*} x_{i}$ is the sum of the frequencies of feature $i$ in all of the sentences in $S_*$ (set of human utterances belonging to class $y$) and $N_{y} = \sum_{i=1}^{n} N_{yi}$ is the sum of the frequencies of all the vocabulary terms in $S_*$. The $\alpha$ represents the Smoothing parameter and it is used in order to avoid the final value of $P(y|X)$ to become zero when one of the conditional probabilities of the features is zero.

In order to transform the MNBC classifier into a linear classifier and to avoid underflow, the equations 3, 4 and 5 are expressed together in the log-space [7]:

$$\log(\hat{\theta}) \propto \text{argmax}_{y} \left[ \log P(y) + \sum_{i} x_{i} \log \hat{\theta}_{yi} \right] \tag{6}$$

It is crucial to note that because the possible LSA values for each feature can be negative and frequencies cannot, each array must be normalized to values between 0 and 1.

### III. System Evaluation

In this section the software adopted in this work is described and, since Python has already plenty of suitable packages that can easily perform the desirable tasks, it is adopted as the computational language. All of the software done for the LSA and NBC belongs to the scikit-learn [21] [22]. The full developed software code is described completely and can be visualized in the following GitHub repository [23].

#### A. Speech Recognition Software

Firstly, a software capable of recognizing the human discourse and transforming it into textual information is necessary. Google Speech API [18] is a reasonable approach as it supports the Portuguese Language and is also powered by ML which improves the still imperfect SR (Speech Recognition) [19]. For Python, the package is speechrecognition [19].

#### B. Keyword Extractor

A Keyword Extractor is a program that withdraws the most relevant words of a text according to the algorithm of the extractor. Three possible options were investigated: Azure [15], LinguaKit [16] and Yake [17].

For an independent program evaluation, the keywords need to be withdrawn from a text by one or more people which contributes to a certain ambiguity. Therefore, a dataset composed by texts and its respective keywords are gathered [20]. In furthermore of diminishing the keywords uncertainty, two metrics are created and used to evaluate each extractor.

In the first metric, the output of each keyword extractor and the keywords of the dataset extracted by people are compared and the percentage of incorrect keywords is computed. The second metric is characterized by comparing each set of keywords extracted by each program to the other two and calculating the percentage of nonequivalent keywords (a keyword that only appears in one software, being absent in the remaining).

<table>
<thead>
<tr>
<th></th>
<th>Azure</th>
<th>LinguaKit</th>
<th>Yake</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st Metric</td>
<td>34.50%</td>
<td>57.89%</td>
<td>44.65%</td>
</tr>
<tr>
<td>2nd Metric</td>
<td>5.93%</td>
<td>28.78%</td>
<td>17.81%</td>
</tr>
</tbody>
</table>

**TABLE 1**

**Mean error for each software with both methods.**

It is easily visualized that Azure is the best one. Although the percentage of mistakes in the first method is quite high, it is significantly lower than the other two. In the second method only a minor error is observed in Azure, contrary to the others.

#### C. Latent Semantic Analysis

1) **Tokenization:** It is executed by the method word_tokenize from the nltk.tokenize package.

2) **Stop Words:** It is obtained from stopwords of package nltk.corpus. The word é was added to the set as it was creating non-existing relationships between sentences and terms.

3) **Porter Stemmer:** It is provided by class RSLPStemmer of nltk.stem. RSLP stands for Removedor de Sufixos da Língua Portuguesa (Portuguese Language Suffix Removal) [20].

4) **Building of TF-IDF matrix:** It is easily executed by the class TfidfVectorizer from the package sklearn.feature_extraction.text. It needs a couple of arguments for it to work properly:

- The tokenizing method, tokenizer.
- The list of stop words, stop_words.
- The minimum document frequency, min_df, the minimum frequency value of each term in all sentences to be admitted to the set of terms.
- The N-Gram’s minimum and maximum values, ngram range. A N-Gram represents a contiguous set of N words of a sentence. Note that if the minimum N-Gram value is 1, the total set of vocabulary includes all of the N-sets from 1 to the maximum value.

5) **Singular Value Decomposition and Dimensionality Reduction:** The SVD is done through the method linalg.svd from package numpy. After normalizing the eigenvalues, the number of dimensions needed according to the percentage of cumulative eigenvalues is obtained and the final matrix is reconstructed with the needed dimensions.

6) **Parameters’ values computation:** There are particular parameters that need a numeric value for their classes to correctly operate. Briefly, the arguments to take into consideration are: ngram_range, min_df and p_eigen. Since it is important to consider each sequence constituted by one word only, the minimum value of N-Gram selected is one.
Regarding the choices of the other parameters, a collection of experiments is conducted to decide which values deliver a better algorithm performance. In order to test the diverse parameters, a training and testing set are needed and, hence, a small database is created with 9 different labels and a couple of distinct human utterances associated to each label, either for training or testing. The experiments are divided into three and in each group, only one of the three parameters are modified to control each parameters’ variance. For every different experiment the percentage of correct predictions is computed and after the trials are completed, the best value for the parameter is chosen.

- **Minimum document frequency, \( min_{df} \)**. The possible values include all natural numbers until 6 as from this frequency onwards the system’s performance deteriorates too much. It is crucial to note that, as the minimum frequency increases, the extent of the set of terms decreases. As such, a smaller vocabulary is preferred since it means that the words eliminated did not contribute to the result. The best values for the \( min_{df} \), with an accuracy of 90.5%, are either 1 or 2. Therefore, the best value is two.

- **Maximum value of N-Gram, \( n_{gram\_max} \)**
  
The values to be analysed range from the minimum N-Gram \( (1) \) to 6, including only integers. For higher \( N \), not only the number of N-Gram elements increases but also, the bigger elements contain more words. Therefore, if two \( N \) values deliver the same performance, the minimal set should be chosen. The best outcome belongs to maximum N-Gram greater or equal to 3 (95.2%). Hence, it is determined that the best value is three.

- **Percentage of cumulative singular values, \( p_{eigen} \)**
  
The possible percentages include every number divisible by 10 from 10% to 90% and every divisible by 5 from 50 to 70 as the results around these values are more interesting. If there is more than one value that attains the same performance, the lowest percentage should be selected as it contains less dimensions. The system manages to function accordingly (100%) when the values of the percentage of cumulative eigenvalues are either 50% or 60% and, thus, the value chosen is 50% (0.5). Also, it is confirmed that is better to execute the Dimensionality Reduction.

D. Multinomial Naive Bayes Classifier

The class MultinomialNB of package sklearn.naive_bayes is picked since it contains the type of MNBC desired. The class has one important variable, the Laplace Smoothing parameter \( (\alpha) \) represented in equation 5 which ranges from 0 (no smoothing) to 1. A set of tests are carried out to choose the best value. The Laplace parameters to be analysed are the following: \{0, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1\} and the performance obtained is the same for every \( \alpha \) (100%). It is known in ML that the employment of \( \alpha \) usually leads to a better output and, thus, a number bigger than zero and not excessively high is picked. Hence, 0.01 is assumed for \( \alpha \).

Before the final results are computed, these trials are to be rerun to confirm the best values and this assumption.

E. Discourse Build-up

For all intents and purposes, the human utterances used to train the system must be done by people unaware of the advantages or disadvantages of the system functionality. However, it didn’t happen in the previous subsections C6 and D, leading the system to better performances than it would otherwise obtain with unknown real data.

The system constructed throughout this work has to possibil- ite a conversation between an elder and a robot. Because each of the distinct utterances uttered by the robot represents a different class label, only the discourse associated to the elderly is mutable. Thus, the only new needed sentences are the human utterances and they need to be associated to a phrase of the robot discourse. For this purpose it is created a form that follows a base dialogue that keeps the robot speech and conceals all the human utterances of this discourse so that, a person that fills the form rewrites them and creates human phrases similar to the ones written in the base dialogue, making a coherent discourse when aligned with the robot speech. The main goal of the form is to capture the variance of how people express the same concepts in distinct words.

Briefly, the discourse starts with a basic greeting and if the elder agrees, they initiate a conversation about their preferences. Because the majority of the questions posed by the robot have \( n \) possible outcomes, always only 2 of them will be picked in order to demonstrate that the system is capable of understanding the conversation path taken by the elder. The possible themes vary from spending time with the family and taking a walk to the topic food preferences. The form is composed through Google Forms and is filled in by 35 people to check if it is a valid approach to create a dataset representative of a conversation between the elderly.

The final set contains 455 human utterances and 22 robot utterances, resulting in 22 class labels used in the MNBC. Ultimately, each human utterance’s length ranges between 1 word and 27 words.

F. Training and Testing Set

It is very important to divide the acquired data into the training and testing set. The first set is used in the learning phase while the second one is to analyse the system’s performance with regard to unfamiliar data. Since every different division of the data will produce distinct training and testing sets, some performances attained are better than the others. As such, to dismiss the influence of the data, a model validation technique based on re-sampling the data into different training and testing samples is used: CV (Cross Validation).

There are many different CV types but, for this purpose, the KCV (K-Fold Cross Validation) is chosen as it divides the data into \( K \) equal-sized portions with one of them being used for testing and the rest for training. Each portion of the set is used once as the testing set so that all of the phrases get the chance to participate in both subsets, creating
K different set divisions. The value chosen for K is 5 in order to obtain an 80%/20% division in training and testing subsets. However, the number of elements in each class label is not constant and to avoid achieving an unbalanced data division a stratified split is needed. Hence, the SKCV (Stratified K-Fold Cross Validation) of class StratifiedKFold from package sklearn.model_selection is selected. For it to operate accordingly, the minimum number of elements per class must be equal to K (5) and, thus, any class whose size is below the threshold is eliminated.

G. System Tuning

Looking for validation of the system parameters (\textit{min\_df}, \textit{ngram\_max}, \textit{p\_eigen} and \textit{\alpha}), the tests executed are repeated.

1) Minimum Document Frequency and Maximum N-Gram Value: First of all, the parameters tested are the maximum value of the N-Gram range intertwined with the minimum document frequency to understand their influence on each other. So, for each N-Gram, all of the possible frequencies are tested. The potential values for each variable are the same as before and the results are represented in figures 1 to 6, with the bar depicting the deviation towards the average performance.

It is observed that, comparing each N-Gram, the worst output is for the smallest value (1) as the average performance for all frequencies corresponds to 70.5% while the ones from other ngram\_max range between 73% and 74%, obtaining its peak for 2 (74.0%). This means that the introduction of the N-Gram improves the performance of LSA.

Regarding the minimum document frequency, it is noted that the best result corresponds to either 2 (figures 1, 4 and 6) or 3 (figures 2, 3 and 5), confirming that introducing the minimum frequency threshold leads to a better LSA performance. Figures 1 and 6 are eliminated as their best value is less than 76%. Observing each maximum average performance obtained for each of the four \textit{N}'s, table II is retrieved.

<table>
<thead>
<tr>
<th>N/min_df</th>
<th>Average</th>
<th>Upper Bound</th>
<th>Lower Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>2/ 3</td>
<td>77.84%</td>
<td>+9.92%</td>
<td>-9.43%</td>
</tr>
<tr>
<td>3/ 3</td>
<td>77.60%</td>
<td>+7.11%</td>
<td>-6.04%</td>
</tr>
<tr>
<td>4/ 2</td>
<td>76.71%</td>
<td>+6.22%</td>
<td>-5.11%</td>
</tr>
<tr>
<td>5/ 3</td>
<td>76.71%</td>
<td>+4.58%</td>
<td>-4.32%</td>
</tr>
</tbody>
</table>

TABLE II

<table>
<thead>
<tr>
<th>N</th>
<th>Average</th>
<th>Upper Bound</th>
<th>Lower Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>70.5%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>74.0%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>74.3%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>74.0%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>74.0%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

As explained before, a bigger minimum frequency for a similar output is preferable and, thus, 3 is chosen for the parameter \textit{min\_df}. It is also apprehended that the best output is the one for \textit{N} = 2 and the worst for \textit{N} = \{4, 5\} whereas the smallest classification standard deviation scores the best value for \textit{N} = 5 and the worst for \textit{N} = 2. The difference between both average values is very small when compared to the classification deviation and, hence, for \textit{N} = 2 the deviation is way too high for a classification system where for \textit{N} = 5 even if the system performance cannot attain such high results as \textit{N} = 2, at least the output is more reliable and precise. Therefore, the optimum value for N-Gram is 5.
Fig. 5. Influence of min_df when the maximum NGram is 5.

Fig. 6. Influence of min_df when the maximum NGram is 6.

2) Percentage of Cumulative Eigenvalues: The experiments are repeated with the same values but adding percentages 45%, 75%, 85%, 95% and 100% and the results are in image 7.

Fig. 7. Influence of p_eigen on the final data set. The maximum and minimum performance values are represented by the bar while the average performance by each point.

Analysing figure 7 the best performances correspond to values 75% and 95%. Even if it isn’t very different from the highest output, the percentage 100% does not correspond to the best score. Therefore, it can be concluded that the LSA technique improves the outcome, specially the dimensionality reduction. Even though the value chosen is way higher than the initial 50% previously selected, it is seen that from 55% to 95% the output obtained is quite similar apart from the standard deviation. Regarding the choice between the two values, a deeper outlook is carried out in table III.

<table>
<thead>
<tr>
<th>p_eigen</th>
<th>Average</th>
<th>Upper Bound</th>
<th>Lower Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>75%</td>
<td>81.98%</td>
<td>89.29%</td>
<td>74.21%</td>
</tr>
<tr>
<td>95%</td>
<td>82.27%</td>
<td>85.94%</td>
<td>78.27%</td>
</tr>
</tbody>
</table>

In this case, the best average performance and the one who has less standard deviation corresponds to the same percentage, 95%. Consequently the number selected is 95%.

3) Laplace Smoothing Parameter: The values for the Laplace Smoothing are the same numbers as before. The trials are repeated and presented in image 8.

Fig. 8. Influence of α on the final data set. The maximum and minimum performance values are represented by the bar while the average performance by each point.

The parameter α is the one that less influences the data, merely fluctuating a meager 2.5% between the best and the worst result. Notwithstanding, it is clear the assumption made in III-D was not valid and, in fact, the best one is not introducing any smoothing since the best output is for α = 0 for both the best performance (81.13%) and also the narrower standard deviation. The values chosen for the parameters in this section are quite different from the first ones determined, and the reason for that is based on the bigger dataset presented in this section. Therefore, it is concluded that for big changes of the set it is necessary to do a revaluation of the parameters’ values to maintain the best outcome possible.

4) Stop Words: There isn’t a stop words set that is guaranteed to describe the most common words of a language that do not provide information to the sentence. Therefore, it is verified if the set chosen in II-A provides the best performance to the system. Four different sets are proposed: A, the one chosen in II-A, B which contains all of the words of set A except the word "é", the set C that is the same as B but
excludes verbs and personal pronouns from the list while the last set bypasses the utilization of a list. These results are illustrated in the following table IV.

<table>
<thead>
<tr>
<th>LSA type</th>
<th>Average</th>
<th>Upper Bound</th>
<th>Lower Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stop Words set A</td>
<td>82.88%</td>
<td>+10.53%</td>
<td>-9.25%</td>
</tr>
<tr>
<td>Stop Words set B</td>
<td>82.76%</td>
<td>+9.55%</td>
<td>-9.13%</td>
</tr>
<tr>
<td>Stop Words set C</td>
<td>84.02%</td>
<td>+8.29%</td>
<td>-9.29%</td>
</tr>
<tr>
<td>No Stop Words</td>
<td>81.83%</td>
<td>+10.48%</td>
<td>-11.50%</td>
</tr>
</tbody>
</table>

**TABLE IV**
Best average value for each system configuration followed by its upper and lower bounds

As according to theory, the worst result is the trial which doesn’t use any stop words list. On the other hand, even though the set previously chosen scored second, the trial that got the best result (set C) also uses a stop word set and it is much simpler. It is also noticed that the introduction of the word “é” in the stop words set enhances the performance of the whole system when compared to using the original stop words set.

It is crucial to acknowledge that since the Inverse Document Frequency in the TF-IDF is adopted in the computation of the word frequency, the weight of the more common words, not present in the stop words list, is reduced quite heavily, securing the integrity of the whole system.

5) **Keywords**: The influence on the performance due to the introduction of the keywords extracted by Azure [15] in the system is to be verified to know if its addition is the right pursuit. For this purpose, 2 trials are executed where in the first experiment, the LSA’s vocabulary purely consists of the terms withdrawn by the TfidfVectorizer while the second also uses the words gathered by the software Azure. The results can be observed in the table V.

<table>
<thead>
<tr>
<th>LSA type</th>
<th>Average</th>
<th>Upper Bound</th>
<th>Lower Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Keywords</td>
<td>84.15%</td>
<td>+9.26%</td>
<td>-8.33%</td>
</tr>
<tr>
<td>Keywords</td>
<td>81.99%</td>
<td>+9.55%</td>
<td>-10.56%</td>
</tr>
</tbody>
</table>

**TABLE V**
Best average value for each system configuration followed by its upper and lower bounds

Inspecting table V, the standard deviation is larger for the second experiment but the highest average performance belongs to the first trial. It is important to mention that the second system has 792 terms of which 576 are extracted through the TfidfVectorizer, way more than the extractor Azure which extorts almost 300 words but only 216 aren’t represented in the rest of the vocabulary. As the introduction of the keywords from Azure doesn’t enhance the overall performance but actually disturbs it enough to slightly lower its accuracy and precision, they are rather useless. In conclusion, the final system will not use the keywords extracted by Azure.

**H. Result and Output Analysing**

1) **Class Labels Robustness**: A deeper look is taken into the behaviour of the system when including different numbers of classes, analysing the loss of the performance. For this purpose, it is created an experiment that picks \( n \) random numbers belonging to an interval between the first label no. 1 and the last one, no. 22. The \( n \) is every natural number between 1, the minimum number of classes possible, and 22 which is the maximum. Afterwards, the classes that correspond to the numbers picked and the associated phrases are selected. Each \( n \) is executed 10 times, obtaining 10 different sets of classes in order to obtain the general influence of numerous distinct classes on the performance. The results of the trial are represented in image 9.

As expected, it is visualized in figure 9 that the highest value (100%) corresponds to using one class, while the lowest value (82.2%) corresponds to the trial with the highest number of labels. In general, the performance of the system declines throughout the addition of more classes. Analysing it with more detail, the decrease is higher for \( n \) smaller than 7 while for the rest of the values the accuracy loss is rather low and slowly decreases. Moreover, even if the classification standard deviation greatly increases from 1 to 7, it diminishes for the rest of the dominion. This demonstrates that introducing many classes in the set, raises the precision of the system (even if a little of accuracy is lost). **Nonetheless, the system is capable of maintaining an adequate result with at least 22 classes.**

2) **Performance on number of phrases per class**: Moving onto the number of phrases per class, the following experiment selects for each trial, \( k \) phrases associated to each one of the 22 class labels. The value of \( k \) ranges from 5, the minimum number of phrases for SKCV, to 34, the total number of sentences of the biggest class. Because the classes have a different total number of phrases associated, when the total number of sentences of a class is below \( k \), that class is eliminated from the set used in that trial. Each trial is also run 10 different times, so that, for lower \( k \), different phrases of a bigger class are used in the same trial. The table VI depicts the number of classes for each \( k \) and the output of the experiment is displayed in figure 10.

The last two \( k \) are ignored since they only have 1 class label in the set and, consequently, have a 100% average
performance. By following the rule of thumb of amount of data samples, the 7 classes with fewer phrases will have a worse performance than the rest of them since they don’t have as many training utterances. On the other side of the spectrum the biggest 5 classes would enhance the system’s performance. To confirm this hypothesis, a trial with the whole dataset is done to see the errors associated to each class and the results of each different set division. Carefully inspecting the results and taking into account the classes that, at least once, have a test set that wrongly predicts 10% or more of the phrases, a similar conclusion is reached. From the previous description, 9 classes are selected where only 3 of them correspond to bigger classes while the rest to smaller ones. Although the majority of the absolute errors belongs to the biggest classes, the inverse happens to the relative error due to their greater size, validating the hypothesis.

Table VI

<table>
<thead>
<tr>
<th>k</th>
<th>5-7</th>
<th>8-21</th>
<th>22-25</th>
<th>24-27</th>
<th>28</th>
<th>29-32</th>
<th>33-34</th>
</tr>
</thead>
<tbody>
<tr>
<td>#classes</td>
<td>21-22</td>
<td>14-15</td>
<td>12-13</td>
<td>11</td>
<td>9</td>
<td>4-5</td>
<td>1</td>
</tr>
</tbody>
</table>

3) Example of a correct dialogue: Here, one of the alternatives of a correct conversation, similar to the base dialogue previously produced, is presented to prove the system works when applied to the problem described in chapter I.

Human(H): Bom dia.
Robot(R): Bom dia. Está tudo bem?
H: Sim e contigo?
R: Eu estou bem-disposto. Tem vontade de conversar agora?
H: Sim, tenho!
R: Muito bem, falaremos dos nossos gostos. O que é que mais gosta de fazer?
H: Então, gosto muito de sair de casa e dar longos passeios.
R: Onde é que gosta mais de ir?
H: Gosto de ir até ao largo do Rato.
R: Também prefiro estar no exterior com bom tempo. O que faz quando está chova ou frio?
H: Fico em casa com amigos a ver um filme.
R: Portanto estou a ver que dispensa o frio.
H: Sim, sou uma pessoa friorenta.
R: Já somos dois que não gostam do frio. Falando agora noutras preferências, qual é a sua comida de eleição?
H: Não sei bem, gosto de muitas coisas.
R: Certo, então vamos por partes. Qual é a sua carne favorita?
H: Gosto de carne de vaca, tem mais sabor.
R: E em relação ao peixe?
H: Como bom português, bacalhau sem dúvida!
R: Realmente é dos melhores peixes. E qual é a melhor sobremesa para si?
H: Olhe, já comia um bom tiramisú. É uma boa sobremesa italiana.
R: Itália é um dos países com melhor comida. Especialmente, com a melhor pizza.
H: Também gosto muito de massas.
R: Nem mais, é mesmo impossível resistir a esses pratos italianos.
H: Estou a gostar da conversa mais preciso de ir trabalhar. Foi um prazer.
R: Percebo completamente. Também foi um prazer falar consigo.
H: Obrigada, igualmente.
R: Até logo.

4) Wrongful predicted phrases: In order to inspect a couple of wrong results made by the system, 5 incorrect predictions of the classes with at least 10% of relative error are selected. The main objective is to demonstrate some of the system’s faults, so that further iterations have already a guideline. For each human phrase that resulted in a wrong outcome, the class, the analogous correct and predicted phrases are presented.

Fig. 10. The impact of the number of phrases for each class label on the final result.

The first clear impressions after observing image 10 are that the worst results are for the lower $k$, i.e. when only a handful of class phrases are used, and the performance slowly increases as the number of phrases rises. This means that as more utterances are available for LSA and more training results are accessible for MNBC, the better is the system’s accuracy. As forecast before and confirmed now with these results, a smaller set of phrases leads to a weaker performance.

Even though all of the $k$ do not have the same amount of class labels, the ones that do have, e.g. 5 – 6, 8 – 11 or 12 – 21, clearly show that the performance enhances when introducing more phrases into the system. It is also evident that the performance improvement is much higher when $k$ is lower, obtaining around 90% of performance when $k$ is bigger than 23 phrases. Because last $k$s have a lower total number of class labels, it can be said that the system slowly converges its performance to 90% not only when introducing more phrases but also when including more classes. Additionally, the highest $k$ with the majority of the class labels is 23 which corresponds to a very favorable classification accuracy (almost 87%). To conclude, a bigger dataset is needed in further iterations to improve the output.
the human phrase, the words in the vocabulary selected by the system are in bold followed by their weight, computed through the TF-IDF function.

- **Class 5:**
  
  **Human phrase:** Bem (0.53) perto (0.84).
  **Correct phrase:** Ainda bem, fico contente. Agora gostaria de saber a sua comida preferida.
  
  **Predicted phrase:** Certo, então vamos por partes. Qual é a sua carne favorita?
  
  The word perto (near) is more important and more connected to the correct phrase while bem (well), with a more reduced weight, is more related to the predicted one. As a result, the human phrase should be associated to the real class label according to the likelihood probabilities. The problem here is the class probability, much higher for the predicted class than the correct one due to the quantity of phrases of the predicted label being much greater. Thus, even if the conditional probability favours the right phrase, the class probability doesn’t, resulting in a higher total for the predicted phrase and in the correct label coming 14th out of 22 classes.

- **Class 6:**
  
  **Human phrase:** Nem por isso, os mais perto (0.61) vivem (0.54) a uma hora (0.57).
  **Correct phrase:** É uma pena, às vezes a vida podia ser mais como nós queremos. Mas para além disso, não há nada que aprecie fazer?
  
  **Predicted phrase:** Ainda bem, fico contente. Agora gostaria de saber a sua comida preferida.
  
  The system fails between deciding if the human’s relatives do or do not live nearby. The main reason is because the term perto (near) is responsible for having the biggest weight in the phrase but also it is 5 times more probable to be associated with the predicted label than the correct one. Furthermore, the words vivem (live), hora (hour) and the a priori probability do not contribute much to the prediction as both probabilities of each class are very similar. Not having any negative keywords to counterbalance the term perto, the system selects the wrong output and puts the correct class in 19th out of 22 classes, quite far from being selected.

- **Class 7:**
  
  **Human phrase:** Caminhar e ir (0.40) às compras (0.58).
  **Correct phrase:** Onde é que gosta mais de ir?
  
  **Predicted phrase:** Tem por hábito comprar roupas?
  
  Scanning the words in bold, it is observed why the system erroneously predicted a different class label. The keywords ir (go), comprar (buy) and the combination ir comprar (weight of 0.70) have a much bigger probability of being associated to the predicted label. Even though the probability of the correct class is higher, it is not enough to counterbalance the conditional ones and prevent the system of correctly predicting the phrase. The final probability of the phrase belonging to the correct label is almost equal to the the correct a priori probability, demonstrating how low the correct likelihood probabilities are. Nonetheless, the right class comes 3rd.

- **Class 13:**
  
  **Human phrase:** Ocasionalmente.
  **Correct phrase:** Pois às vezes sabe bem oferecer uma peça bonita, mesmo que seja a nós próprios. O que mais faz?
  
  **Predicted phrase:** Percebo completamente. Também foi um prazer falar consigo.
  
  Because the only word that composes the human phrase is not considered a keyword by the system, the LSA vector is filled purely with zeros. Therefore, the system bases its prediction purely in the a priori probability and, so the predicted class is chosen as it has the highest a priori probability (0.074). On the other hand, the correct label only comes 17th as its class probability is only 0.016. It is verified that single word phrases should be avoided at all cost (apart from greetings) and that the classes should be as balanced as possible.

- **Class 15:**
  
  **Human phrase:** Só se tiver (0.39) fome (0.45), não (0.24) sou (0.40) grande (0.46) fã.
  **Correct phrase:** É a primeira pessoa que eu conheço que não gosta. O que é que lhe agrada comer?
  
  **Predicted phrase:** Certo, então vamos por partes. Qual é a sua carne favorita?
  
  First of all, the words tiver (be or have), fome (hungry) and grande (big) are the most important terms of the phrase, nonetheless, their probability of belonging to either label is equal, making them useless in the prediction. Another important term, the word sou (am) is twice more probable of belonging to the predicted phrase than the corrected one. On the other hand, the probability of the word não belonging to the correct class is twice as great as the predicted one and both are bigger than the probabilities of the term sou, however, the weight of não is much lower. When this term joins the word sou, their weight increases to 0.45 and the probability tends more to the predicted phrase. The much higher class probability of the predicted label leads the system, along with the likelihood probabilities to wrongly forecast and pushes the right class to score only 15th.

**IV. Conclusion**

The thesis created a solution to the problem of producing a system that generates a robot utterance for every human discourse uttered, so that a coherent conversation between a robot and an elder is obtained. After a meticulous research,
it is found that LSA is the most promising technique for feature selection and extraction. It not only incorporates many NLP techniques in a single algorithm but also, when it comes to text categorization, it is quite efficient. Regarding the ML algorithm, the NBC is picked as it maintains an easy implementation with the NLP technique used and the dataset built.

Regardless of the results obtained with the developed system, it is not faultless. Inspecting some of the phrases wrongly predicted by the system, it is observed that some times the correct and predicted phrases belonged to the same topic but the predicted phrase should either come before or after the correct phrase. Because the robot is reactionary, it only responds to the last human phrase uttered and completely misses the course of the conversation. In future work, this could be solved with the introduction of the last human and robot utterances, along with the new human phrase to be predicted in the algorithm and using neural networks to predict the robot utterance [14]. Additionally, single-word phrases where the word doesn’t belong to the set of terms, also create ambiguity in the system. On the next iterations a solution for this issue could either be totally eliminating that phrase from the dataset or including it directly in the vocabulary, avoiding the possibility of scoring null LSA vectors that mislead the MNBC into picking the class based purely on the a priori probabilities. It must also be noted that the approach of building up a conversation and using a form to obtain variations of the human utterances is an acceptable technique to obtain diverse data, however, it is not efficient enough since more data is required. Therefore, further work on the theme should focus on the creation of a much more complete dataset that can capture the essence of a dialogue between humans. For this purpose, more data can be acquired through the form approach proposed in this work or through recording and transcribing conversations between elders [10].

After the research made, many parts of the whole system were tested in order to see if they, indeed, improve the overall performance. When the keywords extracted by Azure were employed in the system, the performance decreased an overall of 3%, which demonstrated that through the most prevailing words, the system is already capable of rightly defining the textual information. Another piece tested was the list of stop words, i.e the most common words in a language. Although the lack of this list damages the accuracy, the withdrawal of some words from this list, such as, verbs and personal pronouns actually brings a better performance to the system (around 2%). Furthermore, it was verified that the NLP technique adopted, LSA, could be used with the Portuguese language and obtain a favorable result. It was also confirmed that components such as N-Gram and Minimum document frequency indeed improve the final result and, even though the best percentage of cumulative eigenvalues was quite high it is visualized that reducing the dimensionality improves the system, already attaining a very satisfactory output from a percentage of 50% onwards. In conclusion, the LSA still provides a positive result in current days.

REFERENCES