Computer challenges guillotine: how an artificial player can solve a complex language TV game with web data analysis

Luca Squadrone
University TorVergata
Rome, Italy
luca.squadrone@yahoo.it

Abstract

English. This paper describes my attempt to build an artificial player for a very popular language game, called "The Guillotine", within the Evalita Challenge (Basile et al., 2018). I have built this artificial player to investigate how far we can go by using resources available on the web and a simple matching algorithm. The resources used are Morph-it (Zanchetta and Baroni, 2005) and other online resources. The resolution algorithm is based on two steps: in the first step, it interrogates the knowledge base Morph-it with the five data clues, download the results and perform various intersection operations between the five data sets; in the second step, it refines the results through the other sources such as the Italian proverbs database and the IMDb. My artificial player identified the solution among the first 100 solutions proposed in 25% of cases. This is still far from systems like OTTHO (Semeraro et al., 2012) that obtained the solution in 68% of the cases. However, their result was obtained larger resources and not only with a simple web analysis.

Italiano. *Il contributo descrive il tentativo di costruire un giocatore artificiale per un gioco linguistico molto popolare, chiamato "La Ghigliottina", nell'ambito dell'Evalita Challenge (Basile et al., 2018). Ho costruito questo giocatore artificiale per verificare il limite raggiungibile utilizzando unicamente le risorse disponibili sul web e un semplice algoritmo di matching. Le risorse utilizzate sono Morph-it (Zanchetta and Baroni, 2005) e altre risorse online. L'algoritmo di risoluzione si basa su due fasi: nella prima fase, interroga la base di conoscenza Morph-it con i cinque indizi, scarica i risultati ed esegue varie operazioni di intersezione tra i cinque set di dati; nella seconda fase, affina i risultati attraverso altre fonti come il database dei proverbi italiani e l'IMDb. Il mio giocatore artificiale ha identificato la soluzione tra le prime 100 soluzioni proposte nel 25% dei casi. Il risultato ottenuto è ancora lontano da sistemi come OTTHO (Semeraro et al., 2012) che ha ottenuto la soluzione nel 68% dei casi. Tuttavia, il loro risultato è stato ottenuto con risorse più ampie e non solo con una semplice analisi web.*

1 System description

I have used Morph-it (Zanchetta and Baroni, 2005) as a basis for knowledge, instead of building one, as it is free, easy to interrogate and above all suitable for our purpose. Furthermore, it should not be underestimated that building a knowledge base involves an enormous amount of work in terms of time.

After querying the knowledge base with the five data clues, the results are downloaded and various intersection operations are performed between the five data sets. This procedure allows us to find all possible solutions and is the basis for choosing the solution.

Then to find the final solution is verified the existence of proverbs, Aphorisms, movies or books (etc.) that contain both the clue and the possible solution.

2 The memory of the system: Morph-it!

Morph-it! is a free morphological resource for the Italian language, a lexicon of inflected forms with their lemma and morphological features. It was designed by Marco Baroni and Eros Zanchetta. The lexicon currently contains 505,074 entries and 35,056 lemmas. Morph-it! can be used as a data source for a lemmatizer/morphological analyzer/morphological generator.

The main source of linguistic data was the "Repubblica" corpus (approximately 380 million tokens), from which was extracted lemmas and inferred morphological information not present in the original corpus (i.e. gender) using distributional as well as morphological cues. With that information then was generated inflected forms for all extracted lemmas.

Morph-it includes several corpora. A corpus (plural corpora) or text corpus is a large and structured set of texts. In order to make the corpora more useful for doing linguistic research, they are often subjected to a process known as annotation. An example of annotating a corpus is part-of-speech tagging, or POS-tagging, in which information about each word's part of speech (verb, noun, adjective, etc.)

Since only some of these corpora are publicly available, I have chosen:

- La Repubblica, a corpus of Italian newspaper texts published between 1985 and 2000 (approximately 380M tokens);
- ItWac Complete, a corpus of web pages in Italian crawled from Italian university websites

These two corpora form our knowledge base and are united to provide the widest and most comprehensive knowledge base possible.

Then, these resources will be used to match the results found.

- Proverbs and Aphorisms on Italian version of wikiquotes and on the database of Italian proverbs by "Accademia della Crusca"
- Books database crawled from ibs web site
- Film titles, crawled from the Internet Movie Database
- Word definitions, from Dictionary of the Italian language HOEPLI

3 The algorithm

The basic idea of the algorithm[1] is to derive a set of words from MORPH-IT!, using only the 5 clues given in input. This will be the set of possible solutions.

Then the probability of each of these words being the actual solution will be evaluated. This is done through a second phase consisting of a verification of the existence of proverbs, Aphorisms, movies or books (etc.) that contain both the clue and the possible solution. The words with the most associations found are the solutions.

With the term "possible solutions" I will indicate a selection of words where the solution is contained, while with the term "final solutions" I will indicate the 100 words chosen among all the possible solutions.

I would also like to point out that the speed of execution of the algorithm depends on the speed of the internet connection, since I use an online knowledge base and different data must be downloaded each time.

Here is the pseudo code of the first part of the algorithm that finds the possible solutions. It is illustrated in Figure 1:

1. The algorithm takes the 5 clues as input.
2. For each clue it executes two queries, one respectively in each corpus, Repubblica and itWac Complete.
3. After downloading, the results from queries are concatenated as text.

   At the end of this procedure, for each clue, the algorithm will generate a single text. In this text there will be all the concepts with a relation to the clue.
4. Each of the five texts is transformed into a set of single words. So I get 5 sets of words, one for each clue.
5. The intersection between these sets (which I will call "Final Set") is made. The solution, semantically linked to all five clues, in most cases will be contained in the final set. This hypothesis will be verified later, in the next section.

[1]Source code https://gitlab.com/osiast/computer-challenges-guillotine

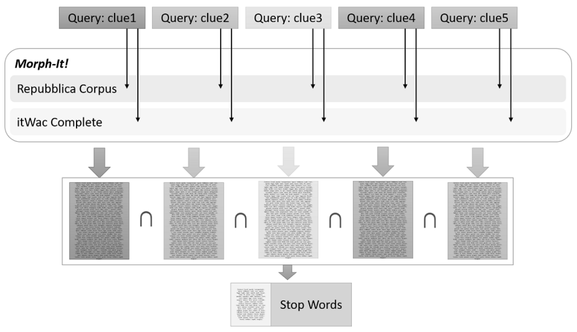

Figure 1: Process for finding possible solutions for a run of the game

6. Among the possible solutions there are many insignificant words such as articles, conjunctions and prepositions. To delete them, I subtracted the set of Stop Words from the final set.

   Stop words are words which are filtered out before or after processing of natural language data. Though ”stop words” usually refers to the most common words in a language, there is no single universal list of stop words used by all natural language processing tools, and indeed not all tools even use such a list. So, I built a special list of Stop Words specifically for this purpose, based on two online resources:

   - A collection of stopwords on github [2]
   - A collection of stopwords from the Snowball site [3]

   After the stopwords have been removed from the final set, I finally have the set of possible solutions.

The next step is the verification of the existence of proverbs, Aphorisms, movies or books (etc.) that contain both one clue and the possible solution. As already mentioned, the words with the most associations found are the final solution.

So, along with each possible solution, I search for clues within the following repositories:

- Proverbs and Aphorisms from two different resources: Italian version of wikiquotes and on the database of Italian proverbs by ”Accademia della Crusca”

[2]github.com/stopwords-iso/stopwords-it/blob/master/stopwords-it.txt

[3]snowball.tartarus.org/algorithms/italian/stop.txt

- Books database crawled from ibs web site
- Film titles, crawled from the Internet Movie Database
- Word definitions, from Dictionary of the Italian language HOEPLI

Whenever the clue and the solution are found together, for example in the same proverb or title of a film, an additional weight of 0.2 is assigned to that solution. The weight can vary from 0 to 1. It indicates the probability that this is the solution of the game.

Here is the pseudo code of this second part of the algorithm.

1. For each of the 5 clues download proverbs, film and book titles, vocabulary definitions and aphorisms containing that clue and put them together in one text.

   At this point we have 5 texts.

2. To all the possible solutions I assign the value 0 as weight.

3. Whenever one of the possible solutions is found in one of these texts, its weight increases by 0.2.

4. Finally the first 100 are taken which have the largest weight in descending order.

4 Test and Results

Testing the algorithm is a key step in the validation process of the proposed solution.

In the first test below, I will run the algorithm on 315 instances of the game in order to evaluate its efficiency and study the results obtained. As already mentioned, each game is composed of five clues and a solution.

The second test will be on the ”knowledge base”. I’m going to measure how many clues contains on average. This will be useful to indicate an upper-bound of efficiency that the algorithm can not overstep.

4.1 Test the algorithm

To evaluate the efficiency of the algorithm, I tried it for 315 different games.[4]

[4]The games used can be downloaded from this link https://goo.gl/6FpK3p

Results	Number	Percentage
Successful	242	76,83%
Fail	73	23,17%
Total games	315	

Table 1: Results on 315 games

	MRR	Returned game	Test game	Solved
This	0,0134	400	105	27
Other	0,6428	405	105	86

Table 2: comparison systems

The query is limited to 8,000 lines per request, as statistically I have noticed that it is a good compromise between the resolution speed and the efficiency of the algorithm.

I downloaded the set of games and coded them into a list. Then I ran the algorithm for each game in the list and I memorized the results.

The data has been processed to create table 1.

In over 76% of cases the exact solution is found in the "Possible Solution". This shows that "The solution, semantically linked to all five clues, in most cases will be contained in the final set". [5]

Regarding the final solutions, in 24,76% of cases the solution is among the first 100. I got a similar result with the test set, where I reached 25,7%.

Table 2 shows the distributions of the scores for the correct and missed solutions of our system on the full set of games in the test set in comparision with other system.[6]

4.2 Test the knowledge base: does it always contain the solution?

To verify that the knowledge base is suitable for our purpose and that it always contains the solution, I have performed tests on 1575 clues, that is, all the ones I had.

In particular I wanted to know if the knowledge base contained a relationship between the solution and each of them. The basic idea was to look for the clue inside the corpora and then filter the results. For this task I used the NoSketch Engine, an open-source tool to perform corpus searches.

As already mentioned, I did tests with 1575 clues of the game looking for them (one at a time) inside the corpora "Repubblica" and "ItWac Complete". The results were very positive. In fact, out of 1575 clues, 1519 of them were always connected with one or more correspondences to the solution.

We can see that out of 315 games:

- 18% (56 of them) can not be resolved due to the absence of 1 or more clues in the chosen knowledge base;

- only 5% (17 of them) can not be resolved due to the limit set on the algorithm query;

This means that without limiting the queries, I will find the solution at most 82% of the time.

So this result shows that the limit of 8000 lines per query penalized the efficiency of the algorithm by only 5% percent.

This confirms that limiting the query to 8000 rows is a good compromise.

5 Analysis of the results and future work

The algorithm's execution, both with the training set and with the test set, produced similar results. From the data obtained we can notice that the percentage of the solutions found in the first phase of the algorithm decreases in the second phase.

Furthermore the value of the MRR is very low despite the solution being found 27 times out of 105.

The reason for these results is that the number of resources in the matching phase are limited. So the solution, despite being found, is often not among first in the output of the 100 proposed solutions.

As future developments, we could improve the algorithm to find the solution from the possible solutions by increasing resources to provide more accurate results.

References

Pierpaolo Basile, Marco de Gemmis, Lucia Siciliani, and Giovanni Semeraro. 2018. Overview of the evalita 2018 solving language games (nlp4fun) task. In Tommaso Caselli, Nicole Novielli, Viviana Patti, and Paolo Rosso, editors, *Proceedings of the 6th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA'18)*, Turin, Italy. CEUR.org.

Giovanni Semeraro, Pasquale Lops, Marco de Gemmis, and Pierpaolo Basile. 2012. OTTHO: an artificial player for a complex language game. In *Popularize Artificial Intelligence, Proceedings of the*

[5]The full results can be viewed at this link https://goo.gl/BdCee9.

[6]MRR (Mean Reciprocal Rank)

*AI*IA Workshop and Prize for Celebrating 100th Anniversary of Alan Turing's Birth, Rome, Italy, June 15, 2012*, pages 47–53.

Eros Zanchetta and Marco Baroni. 2005. Morph-it! a free corpus-based morphological resource for the italian language. *Corpus Linguistics 2005*, 1(1).