

Topic categorization in Portuguese news articles

André F. Santos

CRACS & INESC-Porto LA

Faculty of Sciences, University of Porto

Porto, Portugal

afs@inesctec.pt

Abstract—This document is a model and instructions for L^AT_EX. This and the IEEEtran.cls file define the components of your paper [title, text, heads, etc.]. *CRITICAL: Do Not Use Symbols, Special Characters, Footnotes, or Math in Paper Title or Abstract.

Index Terms—topic categorization, machine learning, text mining

I. INTRODUCTION

Online news articles first appeared as reprints from traditional newspapers; nowadays, however, they represent now the primary source of news for some segments of the population, both in developed and developing countries (whether consumed directly in the newspaper website, or indirectly e.g. through a social media application or a feed catcher).

Unofficially known as *the fourth branch of government*, the press plays a vital role within our society, keeping us informed about the current state of affairs (at a local and global scale) and acting as a watchdog for the other three branches (legislative, executive and judicial). The (lack of) freedom of press and access to the news in a given country is even often considered an indicator of a lack of democracy[1].

As such, improving the ways citizens can access the information (view it, query it and search it) contained in news articles has the potential to contribute for a more informed and, ultimately, a better society.

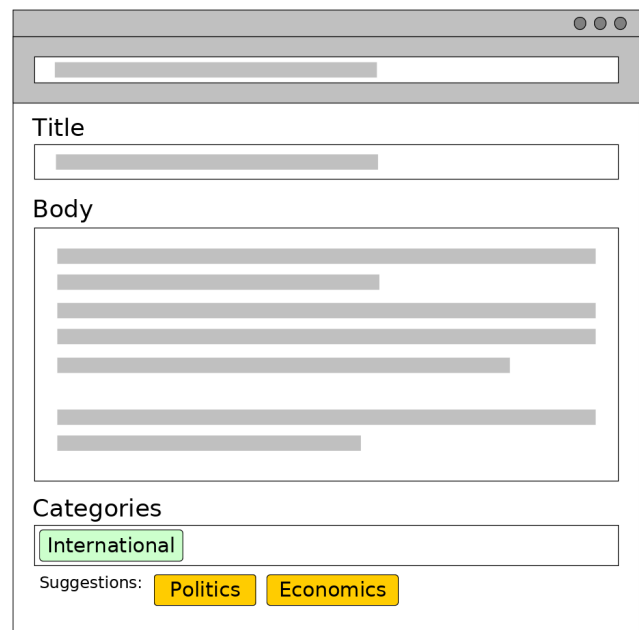
On the other hand, the last decades have witnessed a fast increase on the rate of publication of digital text documents. Traditional document types, such as news articles, scientific papers or books are now published online together with new formats, such as blog posts or tweets, each having thousands or millions of new documents published each day.

And it was not only the publication step which has moved to the digital world; in fact, most often nowadays the whole document lifecycle happens digitally, with virtual tools available for researching, writing, styling, publishing and sharing.

Having the entire workflow happening within the digital world presents some opportunities when compared to the traditional process. In particular, due to the current processing power commonly available, tasks related to the manipulation of the information contained within these documents (searching, compiling, annotating, sharing, ...) can now be performed automatically and targetting a large amount of articles.

In addition to the document content (for example, in a news article, the *title*, *lead* and *body*), its metadata is also important: author(s), date of publication, source, topic, mentioned entities and their relations, etc. Some of this metadata might be filled in and stored along with the document (e.g. *author* and *date of publication*); other is usually extracted from the document content (e.g. mentioned entities).

An example of a feature which improves information access is the categorization of news articles by the topic (or topics) of its content. The presence of such a categorization may influence the way the information is stored, organized, displayed and queried.



The image shows a client-side form for categorizing news articles. It has a 'Title' input field, a 'Body' input area with several horizontal bars representing text lines, and a 'Categories' section. In the 'Categories' section, 'International' is highlighted in a green box. Below it, under 'Suggestions:', there are two yellow boxes labeled 'Politics' and 'Economics'.

Fig. 1. Category classification and suggestion on the client side

The simplest way of achieving this categorization is to have the author of the article to manually introduce it (e.g. the journalist typing it on the news article authoring framework); however, this solution presents some challenges:

- it increases the amount of work the author has to do
- the author might not be sure which categories are available
- the author might not be sure which category is the best (e.g. *Economics* vs *Finance*)

- it does not scale – e.g. if the goal is to categorize an existing (large) corpus

Thus, an automated way of categorizing news articles could solve some of these problems and decrease the burden of this task. Additionally, a lightweight version of such a classifier could be implemented on the client side code of a web application, for example, allowing the categorization to happen in real time (i.e. as the author types in the article text). Figure 1 presents a suggestion of how this feature could look like if implemented on a web application.

The challenges of document classification have been well studied within the machine learning research field of study. Given a corpus of already classified documents, several algorithms might be applied to train a classifier capable of determining the category of additional articles.

In this article, we describe the preliminary results obtained in developing a classifier to categorize news articles using a previously manually categorized corpus. Additionally, we evaluate the possibility of implementing such a classifier as lightweight as possible to allow it to run on the client side of a web application.

II. METHODS

we first needed to choose and obtain a suitable dataset, and then cleaning and preparing it to be used to train the classifiers.

A. The dataset

We gathered a dataset of news articles published in *Observador*¹, one of the main Portuguese newspapers, which stands out from the others for being fairly young (it was created in May 2014) and for existing exclusively online. The initial dataset comprised 42.475 entries, the most recent ones dated from November 2016, from which we used only a subset, for reasons later described.

We gathered all the categories used by Observador, and ordered them from the most common to the least common. We selected the ones which had more than 1000 articles in our dataset, and reduced our original dataset to include articles from these categories only. Figure 2 presents an overview of the selected categories and the number of articles available for each one.

We then randomly selected, from each category, 700 articles to be used to train the classifiers, and 200 to be used to evaluate their performance.

For each article, we had available its contents (title, lead, body) and several metadata fields (publication date, category, tags, etc). A truncated JSON representation of an article can be found in Listing 1.

B. Preprocessing the articles

Originally, the dataset was obtained as large MongoDB collection (more than 2.5 million entries), containing articles from several Portuguese and international newspapers. The process needed to transform this collection into data our classifiers

Listing 1. Example of JSON representation of an article

```
{
  Type: "sapo.obj.creativework.article",
  Source: {
    Name : "Observador"
  },
  Pretitle: "Benfica",
  Title: "Ruben Amorim com rotura total do ligamento
    cruzado",
  Author: {
    Name: "Observador"
  },
  Tags: [
    "benfica",
    "desporto",
    "futebol",
    "ruben amorim"
  ],
  PublishDate: ISODate("2014-08-25T18:33:00Z"),
  Lead: "Depois de Fejsa, mais uma baixa. O
    internacional português [...]",
  Body: "<p>O pior cenário confirmou-se. O Benfica
    informou esta segunda-feira [...]",
  URL: "http://observador.pt/2014/08/25/ruben-amorim
    -com-rotura-total-ligamento-cruzado/",
  CategoryPaths: ["Desporto"],
  Domain: "observador.pt",
  Language: "pt_PT",
  ...
}
```

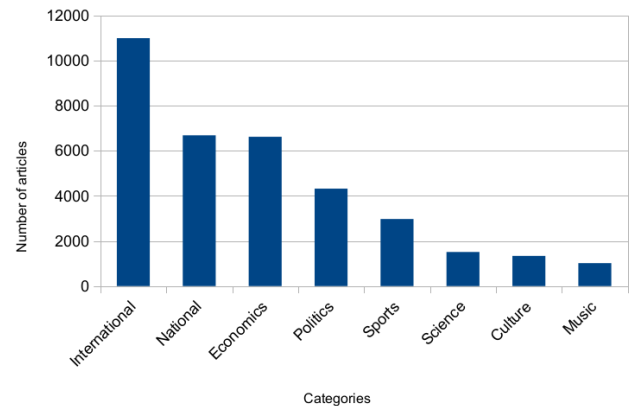


Fig. 2. Total number of articles retrieved for each category

could process required querying the database, exporting the news articles and splitting them into a train and an evaluation datasets.

The database query selected articles from Observador where the body had a length greater than 100 characters (to discard some malformed articles which had an empty body or a body composed of only a few words), and the categories included at least one of the most common categories.

For each article returned by the query, the pretitle, title, subtitle and lead fields, if present, were simply copied to a plain text file, separated by blank lines. The body field, however, was stored in the database in HTML format. As such, the HTML tags had to be stripped, and then it was also added to the plain text file.

¹<http://observador.pt>

The files were then stored in folders, separated by the category. Furthermore, for each category, 700 articles were allocated to the train set, and 200 to the evaluation set. These preprocessing tasks were accomplished using Bash and Node.js scripts.

Both datasets were then loaded into R using the `tm`² package. Each was then passed to a function responsible for preprocessing the text of the articles:

- the text was converted to lowercase characters
- the Portuguese stopwords were removed
- diacritics were converted to their normalized form (e.g. `â` → `a`)
- punctuation signs were removed
- numbers were removed
- word were stemmed (e.g. `conseguiram` → `consequi`)
- whitespace was removed and the text was tokenized

A document-term matrix was then calculated using the `DocumentTermMatrix` method from the `tm` package. For each document in the train set the matrix included all the terms which met the following requirements:

- the term length was between 3 and 30 characters (to discard things like URLs or badly tokenized sentences)
- the term appeared in the document at least twice
- the term appeared at least in 10 documents

The algorithm used to weigh the terms in the matrix was the *tf-idf*, and resulted in 3234 terms.

The final step was to remove the sparse terms from the matrix. This allows to discard terms which might be too specific of the train set and which might negatively influence the performance of the classifiers by overfitting them to the train set. Additionally, a smaller list of terms reduces the execution time of both training and applying the classifier, and the memory footprint of the classifier. However, while discarding sparse terms we might end up removing relevant terms and increase our classifiers error rate.

We used the `removeSparseTerms` function from the `tm` package, and produced three distinct lists of terms:

TS_{99} contained the terms with a sparsity level below 99% (912 terms)

TS_{95} contained terms whose sparsity was under 95% (139 terms)

TS_{90} contained terms with 90% or less sparsity (45 terms)

C. Classification

To create the classifiers, we used 5 well knows algorithms: decision tree (DT), k-nearest neighbors (KNN), naive Bayes (NB), neural network (NN) and support vector machine – with radial kernel (SVM_{RK}) and linear kernel (SVM_{LK}).

We trained each of the classifiers three different times, one for each list of terms (TS_{99} , TS_{95} and TS_{90}). Then we evaluated each trained classifier using the test dataset.

The test dataset was preprocessed in a way similar to the train set (described in the previous section, II-B), but the final

list of terms in each test document was restricted to the terms considered for the train set.

III. RESULTS

TABLE I
ERROR RATES OBTAINED FOR EACH ALGORITHM AND EACH LIST OF TERMS

	DT	KNN	NB	NN	SVM_{RK}	SVM_{LK}
TS_{90}	0.60	0.71	0.85	0.61	0.88	0.88
TS_{95}	0.70	0.56	0.83	0.58	0.87	0.87
TS_{99}	0.70	0.35	0.87	0.76	0.87	0.87

TABLE II
CATEGORIES CONFUSION MATRIX (K-NEAREST NEIGHBORS)

	<i>ciencia</i>	<i>cultura</i>	<i>desporto</i>	<i>economia</i>	<i>mun</i>	<i>musica</i>	<i>pais</i>	<i>politica</i>
<i>ciencia</i>	64	18	25	29	21	37	4	2
<i>cultura</i>	4	37	16	9	19	107	3	5
<i>desporto</i>	1	4	130	8	19	35	2	1
<i>economia</i>	0	6	9	127	12	32	1	13
<i>mun</i>	9	8	25	31	82	32	1	12
<i>musica</i>	0	6	1	1	0	180	0	1
<i>pais</i>	8	15	24	42	17	64	14	16
<i>politica</i>	1	8	7	36	19	24	3	102

A. Evaluation

B. Limited lexicon

IV. DISCUSSION

ACKNOWLEDGMENT

André Santos has a scholarship from Fundação para a Computação Científica Nacional.

REFERENCES

- [1] Goode, L.: Social news, citizen journalism and democracy. *New media & society* 11(8), 1287–1305 (2009)

²<https://cran.r-project.org/web/packages/tm/>