UNIVERSITÉ DE LORRAINE

L'INSTITUT DES SCIENCES DU DIGITAL MANAGEMENT COGNITION
SPEECH PROCESSING

Term identification system for a specific domain - Final Report

Authors:
Albert MILLERT,
Wenjun SUN

A final report for the Terminology class

Academic year 2021-2022

December 5, 2021

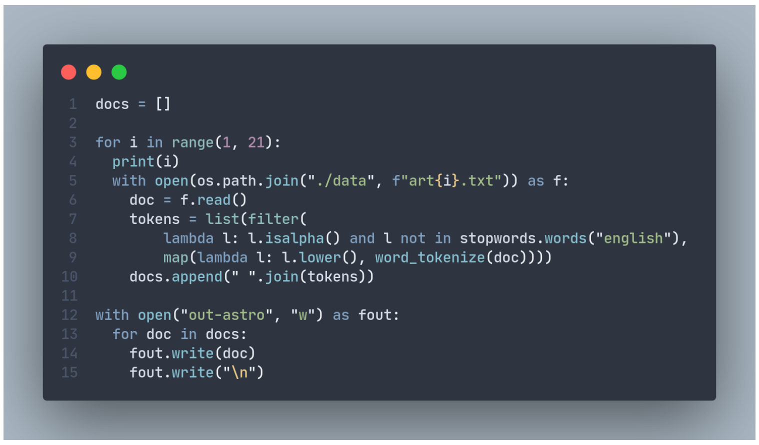
Contents

1 Introduction

The goal of the project was to develop an end-to-end solution for term extraction given raw documents in some domain. The domain of our choice was *astrophysics*. More on steps taken in preprocessing and corpus building can be found in section 2. We then discuss term extraction in 3 and annotation techniques in section 4. The last part of the report consists of description of the utilized model for sequence tagging section 5 as well as evaluation of two methods in section section 6. We finish of by summarizing our work and discussions in section 7.

2 Corpus

The corpus was built from scratch. We started by extracting textual contents from 20 most recent articles from arxiv[1] under *astrophysics* category. The formatted *pdf* files tend to break sentences into separate lines or even split words between lines. Such a format is undesirable; therefore, we mitigated the problem by manually fixing the documents - joining lines together; removing sections' names, references, authors' names and easily separable formulas. *Vim's* functionality came in handy!



```python
docs = []

for i in range(1, 21):
  print(i)
  with open(os.path.join("./data", f"art{i}.txt")) as f:
    doc = f.read()
    tokens = list(filter(
        lambda l: l.isalpha() and l not in stopwords.words("english"),
        map(lambda l: l.lower(), word_tokenize(doc))))
    docs.append(" ".join(tokens))

with open("out-astro", "w") as fout:
  for doc in docs:
    fout.write(doc)
    fout.write("\n")
```

FIGURE 1: Preprocessing code snippet

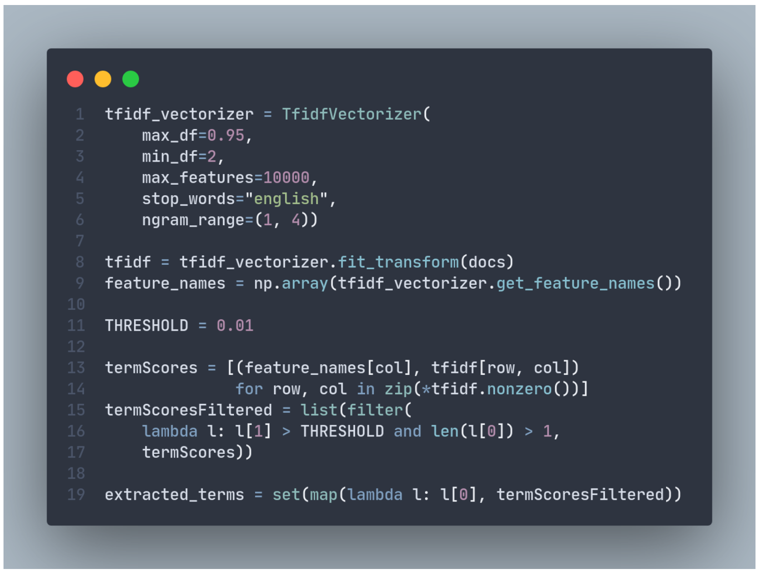
For further text processing we've utilized *python* and *nltk*; details visible in figure 1. For each document we performed the following procedure: 1) Load contents, 2) Tokenize text on a word-level, 3) Lowercase all tokens, 4) Remove English stopwords, 5) Filter out non-alphabetic tokens, 6) Join result as a space-separated string, 7) Save preprocessed docs as a separate line in the result file.

Corpus obtained in this way, served as a basis for all further steps in the project.

[1] arxiv.org

3 Terms candidates extraction

As a foundation of the term extraction method, we've utilized *TF-IDF*. The method is based on assumption that domain-specific terms occur in only a few documents but when they do - their frequencies are rather high. We considered unigrams, bigrams, trigrams, and quadrigrams as term candidates and we set a *TF-IDF* score's threshold to 0.01. The details can be found in snippet 2. The value was chosen by trial-and-error approach.



```python
tfidf_vectorizer = TfidfVectorizer(
    max_df=0.95,
    min_df=2,
    max_features=10000,
    stop_words="english",
    ngram_range=(1, 4))

tfidf = tfidf_vectorizer.fit_transform(docs)
feature_names = np.array(tfidf_vectorizer.get_feature_names())

THRESHOLD = 0.01

termScores = [(feature_names[col], tfidf[row, col])
              for row, col in zip(*tfidf.nonzero())]
termScoresFiltered = list(filter(
    lambda l: l[1] > THRESHOLD and len(l[0]) > 1,
    termScores))

extracted_terms = set(map(lambda l: l[0], termScoresFiltered))
```

FIGURE 2: TF-IDF-based term candidates selection code snippet

Of course, obtained list of terms' candidates is far from being perfect but it's enough as the first step to achieve goal. In the list we managed to observe terms such as: *coordinate singularity*, *galaxy spin*, *entropic*, *black hole mass*, *collapse*, *magnitudes*, *cosmological simulation*, *gravitational lensing*, etc. We can observe that the list is not ideal but captures some *astrophysics*-related terms.

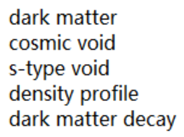
4 Annotation

We first utilized an online labeling tool *termcoord* [2], and then adjusted the annotation manually.

4.1 Term Annotation

We annotated unigram, bigram and trigram terms. For every source document we created one annotation file, each line in the annotation file is one term. Figure **??** below shows a sample from a term annotation file:

[2]termcoord.eu

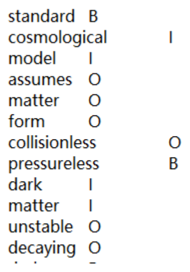


```
dark matter
cosmic void
s-type void
density profile
dark matter decay
```

FIGURE 3: Example of term annotation file

4.2 IOB Annotation

IOB is an annotation format used to achieve sequence tagging. *B* stands for *beginning* and signifies beginning of a term; I - for *inside* and means that the word is inside a term; *O* - for *outside* and denotes that the word is just a regular token, not specific to a considered domain. In the *IOB* annotation, words are arranged in the order of the original text, and each line in the file corresponds to a single word-*IOB* tag couple. A sample of such annotation file is shown in figure 6 below:



```
standard	B
cosmological		I
model	I
assumes	O
matter	O
form	O
collisionless		O
pressureless		B
dark	I
matter	I
unstable	O
decaying	O
```

FIGURE 4: Example of *IOB* annotation file

5 Model

5.1 Rule-based model

Rule-based model refers to setting a specific model to match text sequences to obtain candidate terms. First, we annotated the original data with POS tags, then - we obtained candidates by pattern matching POS-tagged text, finally - we facilitated the filter criteria to get the final term from all the candidates. The whole process of the model is shown in figure 5.

For patterns, we set 3 different kinds of them: unigram, bigram, and trigram. Table 1 shows the pattern, evaluation method and threshold of each setting.

TABLE 1: Pattern, Evaluation Method and Threshold - 1, 2, 3 word(s)

Length	Pattern	Evaluation Method	Threshold
1-word	[NOUN]	tf-idf	0.05
2-word	([NOUN] | [ADJ] | [VERB])([NOUN])	PMI	3
3-word	([NOUN] | [ADJ] | [VERB])([ALL POS])([NOUN])	PMI	5

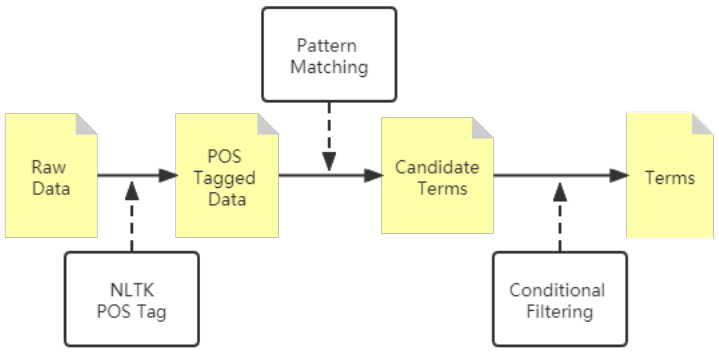


FIGURE 5: Visualization of rule-based model process

5.2 BIO Tag Model

The entire BIO labeling process is: first use the BERT model to obtain the 768-dimensional word vector of each word, and then send the word vector and the corresponding data label to the LSTM network for training, and finally obtain the BIO label. The whole process of BIO Tag Model is shown below.

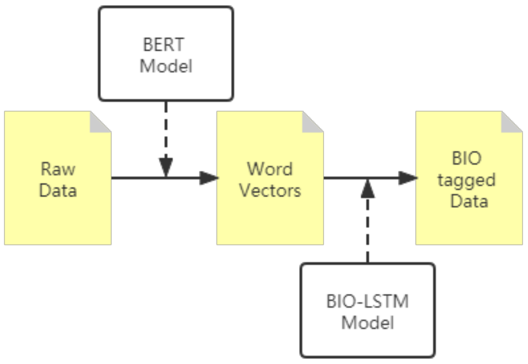


FIGURE 6: Process of BIO-LSTM Model

We selected an *LSTM* neural network model to achieve *IOB* tagging task. Generally, *LSTM* has advantages in sequence modeling and has long short-term memory (hence the name) functions. It's also relatively simple to implement. These reasons convinced us to use the model to test in our work its performance. The high-level overview of our *IOB-LSTM* model is visualized in the figure 7 below:

6 Comparison

All the results from our experiments were juxtaposed in the table 2. We can notice that, among all the results, unigram terms yield the best result. With the increasing length of terms, the recognition effect gradually deteriorates. It's understandable, as the complexity of terms grows with the number of words taken into consideration.

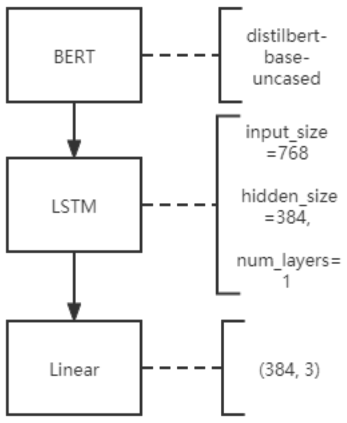


FIGURE 7: Structure of LSTM network

TABLE 2: Results of unigrams, bigrams, trigrams

Length	P	R
1-word	16.55	27.90
2-word	14.28	28.08
3-word	4.84	15.44
All	10.01	23.81

Unfortunately, the model's performance is very unsatisfactory. Although it can mark sentences in an *IOB* format, there are way too many *O*'s making the dataset too unbalanced, s.t. it could hardly be used for term extraction. We tried many methods, such as changing the way of word embedding, improving the network structure, etc., but in the end we did not obtain more satisfactory results.

7 Conclusions

Working on this project we got exposed to a slightly new research domain. We've developed an end-to-end solution starting from corpus building, through text processing, both manual and semi-automatic *IOB* / *POS* annotation with pattern matching for the rule-based term candidates extraction, and finished off by designing a model for automatic term extraction / recognition. We're not entirely satisfied with the model's performance but the overal pipeline consists of all the points required for the task.

We believe that in terms of text processing we could improve our methods in the following dimensions: 1) More careful *POS* tagging of the original document, 2) Streamlining the extraction mode to reduce the generation of wrong terms.

For the *LSTM* model, we concluded that the reasons for our failure were: 1) The original documents lacked high-quality annotations, 2) In the selected documents, the proportion of terms is low, making it difficult for the model to obtain the ability to recognize terms, 3) Our current *LSTM* model's architecture may benefit from several improvements but it would take way more time to investigate.