Near Duplicate Competition Names Detection: Cosine Similarity

Vasilii Salata
University of Tübingen / Wilhelmstr. 19-23
72074 Tübingen, Germany
vasilii.salata@student.uni-tuebingen.de

Abstract

This paper describes a participation in near-duplicate detection task, where the aim it to create the program that could match competition names from one sport-betting online resource to competition names from another one as precisely as possible. Here a binary classification approach with usage of cosine similarity between n-gram vectors of names is presented. The model showed rather good precision. However, the evaluating system was not as accurate as it could be.

1 Introduction

The data set for this task includes names of the competitions and the corresponding coefficients that have been parsed from two separate sport-betting online resources(https://www.fonbet.ru/live/, https://1xstavka.ru/en/live/) for a month several times a day.

Difficulties here arise when competition names that define the same competition differ and they differ in most of the cases. For example:

```
'Natalija Kostic - Nefisa Berberovic'

'Kostic N - Berberovic N'
```

or:

```
'Seoul Cityhall Amazones (Women) -
 Gyeongju (Women)'
'Seoul Amazones(w)- Gyeongiu KHNP(w)'
```

We see that not transliteration only causes the different spelling but also reductions, abbreviations, etc. So our model should find the appropriate option from many others and make a match, that afterwards is assessed using the corresponding coefficients.

We present a system to tackle this matching problem. We split strings into n-grams and look for the closest n-gram vector using cosine similarity technique.

2 Related Work

In his work, Saputra (2011) detected plagiarism calculating cosine similarity between documents. First, he extracted terms from documents, then he calculated term frequency and afterwards compared the resulting vectors using cosine similarity. In other work, Kraisak Kesorn (2014) modified cosine similarity to define ranks of documents in different languages using keywords and concept words.

Liwei Ren and Qiuer Xu (Qiuer Xu) are the authors who worked in the near-duplicate detection area. they proposed an algorithm to define similarity between documents based on the common sub-strings ($0 \leq$ SIM(d1,d2) ≤ 1, where SIM(d1,d2)=1 if d1 and d2 are the same documents; SIM(d1,d2)=0 if d1 and d2 have no common sub-strings) and length of the documents.

3 Model Description

3.1 Cosine Similarity

The main method used in the paper is Cosine similarity. **Cosine similarity** measures the similarity between two vectors of an inner product space. It is measured by the cosine of the angle between two vectors and determines whether two vectors are pointing in roughly the same direction. It is often used to measure document similarity in text analysis.

A word can be represented by hundreds of attributes, each recording the frequency of a particular n-gram. Thus, each competition name is an object represented by a n-gram-frequency vector. Let‘s take, for example, 3 competition names from our data:

```
'Al-Hussein - Al-Sareeh'
'Al Hussein - AL Sareeh'
    'Molde - Brann'
```

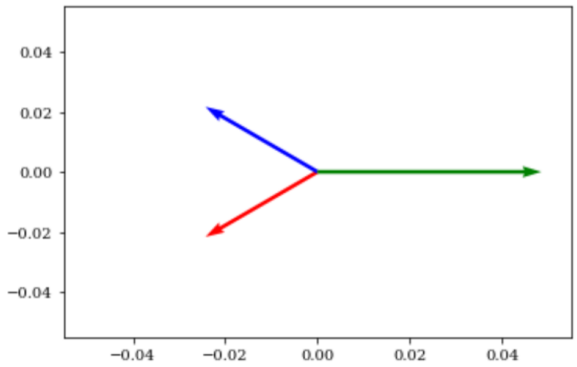

Figure 1: Vectors of competition names where red color represents 'Al-Hussein - Al-Sareeh', blue color - 'Al Hussein - AL Sareeh' and green color - 'Molde - Brann' with dimensions reduced by PCA

in Table 1 we see that name one contains two instances of the bigram **al**, while **ss** occurs only once. The bigram **al** is absent from the entire third name, as indicated by a count value of 0. Such data can be highly asymmetric. Such vectors are typically long and sparse. As we see in the Figure 1, vectors pointing at the similar direction are likely to represent the same competition.

> Jiawei Han and Pei write that **Cosine similarity** is a measure of similarity that can be used to compare documents or, say, give a ranking of documents with respect to a given vector of query words. Let **x** and **y** be two vectors for comparison. Using the cosine measure as a similarity function, we have
>
> $$sim(x, y) = \frac{x \cdot y}{||x||||y||}$$
>
> where $||x||$ is the Euclidean norm of vector $x = (x_1, x_2, x_3, ..., x_n)$, defined as $\sqrt{x_1^2 + x_2^2 + ... + x_n^2}$.

Conceptually, it is a dot product of normalized vectors, i.e. length of the vector. This measure computes the the cosine of the angle between two vectors, where value of 0 means that vectors are orthogonal and have no match, the closer the cosine similarity value to 1, the greater the match between vectors.

3.2 Data and Encoding

A web-parser downloading competition names and corresponding coefficients of live or coming competitions to csv-files from two resources was made especially to collect data for this task. Overall,

names	**re**	**sa**	**se**	**ss**	**...**	**-h**	**-s**	**al**
name1	1	1	1	1	...	1	1	2
name2	1	1	1	1	...	0	0	2
name3	0	0	0	0	...	0	0	0

Table 1: Bigram-name matrix

94 csv-files was created (47 for both of resources) and more than **3 thousand** useful (having appropriate coefficients) competition names that can be matched and then evaluated.

Then every competition name was encoded into n-gram vector. This procedure was described in 3.1 subsection. In order to improve the results we add a special beginning-of-string and end-of-string signs to our competition names. We fit the list of given competition names adding a competition name from another list. Then we transform it into **n-gram-name matrix**.

3.3 Matching Function

After all the necessary data is encoded, we calculate the cosine similarity between vectors of competition names from one resource and vectors of competition names from another resource using everyone-to-everyone method as shown in Figure 2. Then the best cosine similarity score for each name with the corresponding competition name from another resource is returned provided that the score has passed the threshold. We then use coefficients from competition name from **n** resource and matched competition name from **m** resource to evaluate the match. If coefficients overlap, match is considered to be right. There is also a way that a competition name doesn't have its' match in the other resource. In that case function returns the best cosine similarity score and it doesn't pass the threshold. If this happens and coefficients don't overlap, this competition name is considered to have no match. The model is shown in Figure 3.

4 Model Demonstration

4.1 Parameters choosing

To obtain good results we should define 2 parameters: the **threshold for the "match"** and **n-gram range**. To set these parameters we use ROC (receiver operating characteristic) and AUC (area under ROC-curve) - the technique to assess the quality of binary classifier based on confusion matrices given different thresholds. Thus, we compare three encoding system: using **2-gram** only, using **1-to-3-**

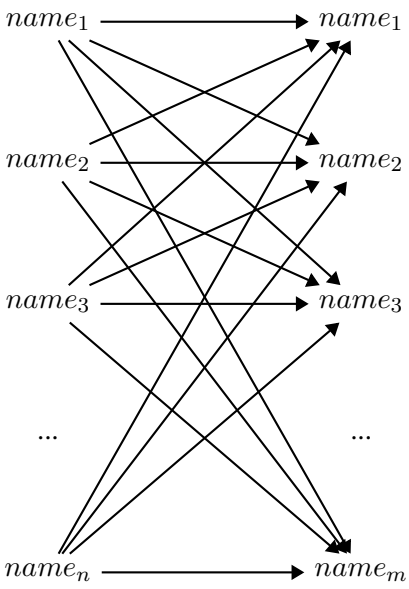

Figure 2: representation of how cosine similarity is calculated between competition names of different resources at one parsing. n and m are numbers of competition names from different resources at one parsing time. Arrows represent similarity functions

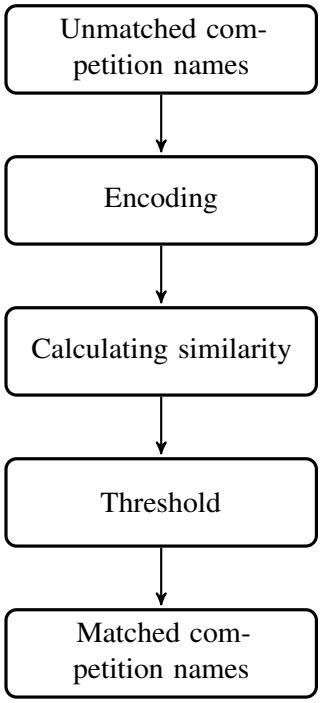

Figure 3: Presented model

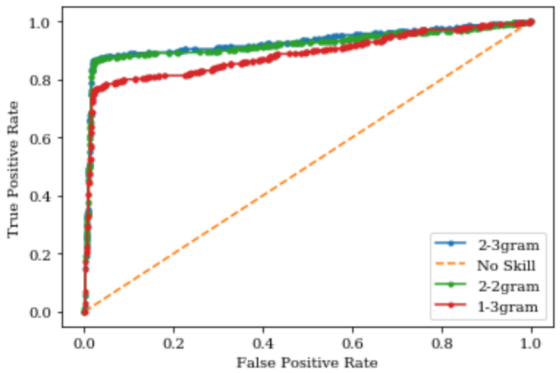

Figure 4: ROC-curves

n-gram range	AUC score
baseline	0.5
1-3-gram	0.885
2-2-gram	0.921
2-3-gram	0.928

Table 2: AUC scores

gram range and using **2-to-3-gram** range. Figure 4 represents these ROC-curves. To determine the best n-gram range we should compare the area under the curve. "No Skill" line shows the baseline. If the curve goes under the baseline curve, it represents a classifier worse than the dummy one. Table 2 shows AUC scores of models with different n-gram ranges. We see that 2-2-gram and 2-3-gram range models have bigger area under the curves than 1-3-gram range, but 2-3-gram model works slightly better.

To choose the optimal threshold we should minimize **False Positives** and maximize **True Positives** matches. Looking at the Figure 4 it is dots where **2-3-gram** range model curve starts to plateau. It appeared to be the value between **0.485** and **0.541** depending on how many False Positives we're willing to accept. In our case we aim to get a **"true match"**, so **False Negatives** are not as important as small number of **False Positives**, so we set the threshold to 0.541.

4.2 Results

The results we obtained using the parameters defined in the section 4.1 are presented in the Table 3. However, after manual exploring of obtained resulting matches some flaws of the evaluation were discovered. It appeared that the biggest part of matches considered as False Negatives are actually True Negatives but considered as False by a mis-

	Actually positive	**Actually Negative**
Predicted positive	TP=520	FP=11
Predicted Negative	TN=962	FN=104

Table 3: Confusion matrix

take. In some cases the most similar competition name, which cosine similarity score did not pass the threshold, had the same corresponding coefficients as competition name we compared with, but was not actually signified the same competition.

Thus, we should conclude that a high cosine similarity score and coefficients overlapping can be considered as "match", but coefficients overlapping only doesn't mean "match". Not relying on False Negatives we assume that the best metric for our model is **Precision**. The precision of the model is **97,9%**

5 Conclusion and Discussion

For this task we considered the approach using cosine similarity of competition names vectors to match the closest ones. The success of our approach suggests a 'simplicity first' strategy is sensible in this case. However, the evaluating system demonstrated some weaknesses, so it is hard to evaluate the accuracy of our model correctly. Nevertheless, our model demonstrates **97,9% of right matches**.

One of the suggestion to improve the evaluating system is to rely on **manual data markup**. To remain evaluating automatized we need more coefficients. Using competition names with more than two coefficients would have led to a significant shortage of useful data in our case.

References

[1] Jiawei Han, M. K. and Pei, J. (2012). Data Mining: Concepts and Techniques. Elsevier Inc.

[2] Kesorn, K. (2014). A Modified Cosine Similarity for Cross Language Information Retrieval. Advanced Materials Research *931-932*, 1348–1352.

[Qiuer Xu] aQiuer Xu, L. R. Near Duplicate Document Detection: Mathematical Modeling and Algorithms.

[4] Saputra, W. S. J. (2011). Calculate Similarity of Document Using Word Count and Cosine Similarity. Bali international Seminar on Science and Technology *978-979*, BII.9–1 – BII.9–3.