N-gram Based Text Classification According To Authorship

Anđelka Zečević
Faculty of Mathematics, University of Belgrade, Serbia
andjelkaz@matf.bg.ac.rs

Abstract

Authorship attribution studies consider author's identification of an anonymous text. This is a long history problem with a great number of various approaches. Those ones based on n-grams single out by their performances and good results. A n-gram approach is language independent but the selection of a number *n* is actually not. The focus of this paper is determination of a set of optimal values for number *n* for specific task of classification of newspaper articles written in Serbian according to authorship. We combine two different algorithms: the first one is based on counting common n-grams and the another one is based on relative frequency of n-grams. Experimental results are obtained for pairs of n-gram and profile sizes and it can be concluded that for all profile sizes the best results are obtained for $3 \le n \le 7$.

1 Introduction

Language is just one of many possible ways for expressing individuality. For researchers in the field of authorship attribution the focus of interests is how this uniqueness enacts on writing and how it can be measured. During the period of nontraditional approach to this problem the variety of features for quantifying the writing style are considered – from lexical to application-specific (Stamatatos, 2009). N-grams are treated as character features and they are widely used in statistical natural language processing.

From a machine learning point of view, the authorship attribution problem can be viewed as text classification task: automatically sorting a set of texts into classes from a predefined set (Sebastiani, 2001). Here, each class represents a concrete author.

A goal of this paper is to identify authors of anonymous articles from the local daily newspapers using n-gram based algorithm. The articles discuss similar topics, all are written in Serbian and published in the same period of time.

The scheme of our algorithm is depicted in Figure 1 and represents a classical profile-based algorithm:

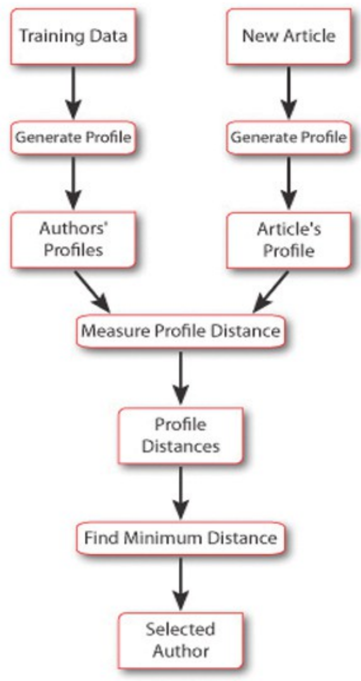

Figure 1: The algorithm schema

the training data set is used for generating authors profiles, authors profiles are laid aside until a new article arrives. Then, the article profile is generated and compared to all authors' profiles. The system selects the author whose profile has the smallest distance to the article's profile.

The remainder of the paper is organized as follows. In Section 2 we define a byte level n-gram, in Section 3 we discuss n-gram generation and propose a data structure for storing all relevant information. A text profile is defined in Section 4 while Section 5 introduces a distance measure between two profiles. Section 6 discusses measures for estimation of classification effectiveness. Section 7 summarizes obtained results and finally, Section 8 presents some conclusions and future directions.

2 N-grams

A n-gram is a continuous sequence of *n* bytes or *n* characters or *n* words of a longer portion of a text. Therefore, we distinguish *byte level*, *character level* and *word level* n-grams. For example, for portion of a text *green tee* all character level 5-grams are: *green*, *reen_*, *een_t*, *en_te* and *n_tee* where the underscore character (_) represents a blank and all word level 1-grams are: *green* and *tee*.

In this paper we will focus on byte level n-grams. Character byte representation depends on character encoding. We will consider only UTF-8 encoding. For example, for the English word *day* the appropriate sequence of bytes is 01100100 01100001 01111001 and for the Serbian word *šta* (English *what*) the sequence is 11000101 10100001 01110100 01100001. These two representations differ in size because there are a few Serbian letters (š, ž, ć, č, đ) which are two bytes long. That means we can split them in two meaningless parts if we use byte level n-grams. Despite the fact that a great number of n-grams will contain both bytes we can benefit from this approach in aspect of more efficient memory usage.

In general, n-grams afford language independent processing, tracking of lexical and contextual information, more robust behaviour in the presence of different kinds of textual errors because errors affect only a limited number of n-grams (Cavnar and Trenkle, 1994) and automatic detection of words that share the same root form (Stamatatos, 2009).

3 N-gram Generation

The computational requirements of byte level n-grams extraction are minimal – a single pass through the text is sufficient and requires no special tools. Some text preprocessing such as character selection or conversion of letters to uppercase or lowercase can be done, but it is not the case in our approach.

Adjacent n-grams overlap and contain redundant information so the memory requirements are more intensive in comparison to methods that store only words. If the portion of a text is *K* byte length the number of n-grams is *K+1-N* so the total size of the memory is *(K+1-N)*N* bytes.

We decided to store all n-grams in a data structure called a trie or prefix tree (Aho et al., 1983; Sedgewick, 2002). A node of a trie (Figure 2) contains a single byte value and from the node position we discover the value of the corresponding n-gram - picking up one by one byte on the path from the root of the trie to that node. Besides the field *byte* containing the byte value, each node contains the field *count* with the number of occurrences. The field *isEnd* determines if the *byte* is the end byte of the n-gram or not and it is obilagatory field because we work with n-grams of fixed size. Two additional link fields are included - one to its children named *children* and one to the next node in the trie named *nextTrieNode*.

```
struct TrieNode{
    char byte;
    unsigned int count;
    int isEnd;
    struct TrieNode* children[SIZE];
    struct TrieNode* nextTrieNode;
}
```

Figure 2: The node definition

The main advantage of using tries over using other data structures such as trees or hash tables is effective retrieval (looking up a n-gram takes worst case O(n) time) and the time for the operations insert, find and delete a node is almost indentical.

4 Text Profile

A text profile is a set of M most frequent n-grams. Precisely, it is a set of pairs

{(x_1, f_1), … , (x_{M-1}, f_{M-1})} (Kešelj et al., 2003)

where x_i denotes a n-gram value and f_i its relative frequency. The number *M* is called profile size.

For the purpose of classification we need text profile generation in the following steps: when we define authors' profiles and when a new article arrives for classification.

Single author's profile is generated from its training data set. For each text in the training data set all n-grams are extracted and added to the author's trie. Every time a n-gram is added to the author's trie the number of its occurrences is increased by one. When done, all n-grams are put in the array and sorted into descending order by the number of occurrences. Only the first *M* n-grams are kept and their numbers of occurrences are divided by the total number of n-grams in training data set to obtain relative frequencies.

The process of generating a new article's profile is the same as above except there is no training data set but only the given article.

5 Distance Measure

A distance measure used in this paper is

$$d(A,a) = \sum_{x \in A} \left(\frac{2 \cdot (f_A(x) - f_a(x))}{f_A(x) + f_a(x)} \right)^2$$

where *A* is an author profile, *a* is an article profile, *x* is byte level n-gram and $f_A(x)$ and $f_a(x)$ are the relative frequencies of that n-gram in author and article profile. The same distance values are obtained if condition *x in A* is replaced with *x in a* cause only common n-grams are taken into account.

This measure combines two measures originally proposed by Keselj et al. (2003) and Frantzeskou et al. (2006). The first one is *dissimilarity measure* and takes into account all n-grams of author's profile and article's profile. In case where at least one author's profile is shorter then profile size *M*, this function favors that author. The second measure is called *simplified profile intersection* and takes into account only the common n-grams of author's and article's profile. In case where one of the author's profile is longer then others, this function favors that author. In our case the first problem is avoided by using profiles intersection, while the second problem is avoided by calculating relative frequencies of n-grams.

When a new article arrives for classification, the distance among article's profile and each of the author's profile is calculated. The system applies 1-nearest neighbour algorithm (Mitchell, 1997), picks the minimal distance and assigns the article to the winning author.

6 Classification Effectiveness

For estimating the effectiveness of a single class classification we use F_1 measure:

$$F_1 = \frac{P \cdot R}{P + R}$$

which attributes equal importance to precision P and recall R:

$$P = \frac{TP}{TP+FP}, \quad R = \frac{TP}{TP+FN}$$

Another measure we use is accuracy:

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

Values TP, TN, FP and FN are values from the contingency table: number of yes-yes, no-no, yes-no and no-yes labeled articles.

For overall estimation of effectiveness we used macroaverage of individual values (Sebastiani, 2001):

$$F_1 = \frac{\sum_{i=1}^{c} F_{1i}}{c}, \quad accuracy = \frac{\sum_{i=1}^{c} accuracy_i}{c}$$

where *c* is the total number of authors.

7 Results

We tried to classify anonymous articles of three authors[1]. A training data set for each author is approximately 100KB in size and contains respectively 20, 17 and 27 articles available on the Internet archive[2]. The program is tested on two test data sets – the first one sizes approximately 220KB and consists of 45 articles (15 for each author) and the second one sizes approximately 330KB and consist of 60 articles (20 for each author). Table 1 and Table 2 represent obtained values in respect to accuracy and F1 measure for the second test set.

The system achieves accuracy over 80% for all n-gram sizes greater then 2 and the profile sizes greater then 500 n-grams. Accuracy increases with the size of the profile and the best results are obtained for $3 \leq n \leq 7$.

The similar conclusion can be drawn from the Table 2. F_1 measure values are over 80% for all n-gram sizes greater then 4 and the profile sizes greater then 1000.

1 Authors names are: S. Biševac, Z. Panović and A. Roknić

2 http://www.danas.rs

8 Conclusions and Future Directions

The presented method is not novel but it gives some insights into results referring to Serbian. The algorithm has demonstrated good performance, but it should be applied to other languages to see how it works. Also a threshold existence should be examined in order to achieve precise and more effective classification according to authorship.

Profile size	N-gram size									
	1	2	3	4	5	6	7	8	9	10
20	0.7	0.68	0.62	0.64	0.71	0.7	0.62	0.63	0.54	0.58
50	0.75	0.82	0.68	0.68	0.67	0.71	0.67	0.62	0.58	0.6
100	0.56	0.66	0.71	0.72	0.71	0.75	0.7	0.7	0.64	0.67
500	0.56	0.84	0.87	0.86	0.85	0.91	0.9	0.88	0.84	0.84
1000	0.56	0.55	0.87	0.86	0.86	0.92	0.88	0.88	0.82	0.82
1500	0.56	0.55	0.91	0.93	0.93	0.95	0.91	0.91	0.88	0.91
2000	0.56	0.55	0.92	0.92	0.9	0.92	0.9	0.9	0.88	0.91
3000	0.56	0.55	0.91	0.93	0.94	0.93	0.96	0.91	0.91	0.9
4000	0.56	0.55	0.81	0.92	0.93	0.92	0.95	0.88	0.88	0.91
5000	0.56	0.55	0.82	0.92	0.94	0.92	0.88	0.92	0.92	0.84

Table 1: Accuracy

Profile size	N-gram size									
	1	2	3	4	5	6	7	8	9	10
20	0.53	0.54	0.44	0.46	0.57	0.53	0.38	0.42	0.28	0.35
50	0.61	0.72	0.53	0.53	0.5	0.53	0.5	0.39	0.35	0.37
100	0	0.5	0.56	0.59	0.56	0.61	0.51	0.53	0.41	0.48
500	0	0.75	0.81	0.8	0.78	0.86	0.84	0.79	0.78	0.76
1000	0	0	0.81	0.8	0.8	0.88	0.82	0.82	0.7	0.71
1500	0	0	0.86	0.9	0.9	0.93	0.86	0.86	0.83	0.86
2000	0	0	0.88	0.88	0.85	0.88	0.85	0.84	0.83	0.86
3000	0	0	0.86	0.9	0.91	0.89	0.94	0.86	0.86	0.85
4000	0	0	0.71	0.88	0.89	0.88	0.93	0.83	0.83	0.86
5000	0	0	0.73	0.88	0.91	0.88	0.83	0.88	0.88	0.76

Table2: F_1 measure
in our calculation when TP=FP=FN=0, F_1 measure is defined as 0

9 References

A. Aho, J. Hopcroft and D. Ulman. 1983. *Data Structures and Algorithms.* Pearson. Addison-Wesley. Paperback

A. Rahmoun and Z. Elberrichi. 2007. *Experimenting n-grams in Text Categorization.* Issue of International Arab Journal of Information Technology, Vol 4, N°.4 : 377-385.

E. Stamatatos. 2009. *A Survey of Modern Authorship Attribution Methods.* Journal of the American Society for Information Science and Technology, 60(3): 538-556. Wiley.

F. Sebastiani. 2001. *Machine Learning in Automated Text Categorization.* ACM Computing Surveys, 34(1):147.

G. Frantzeskou, E. Stamatatos, S. Gritzalis and S. Katsikas. 2006. *Effective identification of source code authors using byte-level information.* In Proceedings of the 28th International Conference on Software Engineering, 893-896. ACM Press.

R. Sedgewick. 2002. *Algorithms in C.* Adison-Wesley.

T. Mitchell. 1997. *Machine Learning*. McGraw Hill.

V. Kešelj, F. Peng, N. Cercone and C. Thomas. 2003. *N-gram based author profiles for authorship attribution.* Pacific Association for Computational Linguistics, 255–264, Halifax, Canada.

W. Cavnar and J. Trenkle. 1994. *N-gram-Based Text Categorization.* Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval.