Generating Term Weighting Schemes through Genetic Programming

Ahmad Mazyad
Université du Littoral Côte d'Opale
Calais, France
ahmad.mazyad@univ-littoral.fr

Fabien Teytaud
Université du Littoral Côte d'Opale
Calais, France
teytaud@univ-littoral.fr

Cyril Fonlupt
Université du Littoral Côte d'Opale
Calais, France
fonlupt@univ-littoral.fr

ABSTRACT

Term-Weighting Scheme (TWS) is an important step in text classification. It determines how documents are represented in Vector Space Model (VSM). Even though state-of-the-art TWSs exhibit good behaviors, a large number of new works propose new approaches and new TWSs that improve performances. Furthermore, it is still difficult to tell which TWS is well suited for a specific problem. In this paper, we are interested in automatically generating new TWSs with the help of evolutionary algorithms and especially genetic programming (GP). GP evolves and combines different statistical information and generates a new TWS based on the performance of the learning method. We experience the generated TWSs on three well-known benchmarks. Our study shows that even early generated formulas are quite competitive with the state-of-the-art TWSs and even in some cases outperform them.

KEYWORDS

Genetic Programming, Machine Learning, Term-weighting Learning, Support Vector Machine

ACM Reference Format:
Ahmad Mazyad, Fabien Teytaud, and Cyril Fonlupt. 2018. Generating Term Weighting Schemes through Genetic Programming. In *GECCO '18 Companion: Genetic and Evolutionary Computation Conference Companion, July 15–19, 2018, Kyoto, Japan.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3205651.3205799

1 INTRODUCTION

Text Classification (TC) aims to automatically assign a set of predefined categories to a text document based on their content. TC is an important machine learning problem that has been applied to numerous applications such as spam filtering [4], language identification [5], authorship recognition [3], sentiment analysis [2], and so on. Generally, the TC approach is to learn an inductive classifier from a set of predefined categories. This approach requires that documents are represented in a suitable format such as the Vector Space Model (VSM)representation (Salton and Buckley, 1988).

In a VSM, a document d_j is represented by a term vector $d_j = (w_{1,j}, w_{2,j}, ..., w_{t,j})$ where each term is associated with a weight $w_{k,j}$.

GECCO '18 Companion, July 15–19, 2018, Kyoto, Japan

ACM ISBN 978-1-4503-5764-7/18/07.
https://doi.org/10.1145/3205651.3205799

The weight represents how much a term contributes to the semantics of a document. The method which assigns a weight to a term is called Term Weighting Scheme (TWS).

Numerous TWSs exist such as Term Frequency-Inverse Document Frequency (TF-IDF). They are generated according to human a priori and mathematical rules. TWSs are usually simple mathematical expressions. Unfortunately, depending on the application, it is not easy to know a priori which TWS will be effective.

As expression discovery may naturally be addressed by genetic programming [1], we are interested in this paper to study the effectiveness of Genetic Programming (GP) generated formulas and their aspects. We are also interested to know if a stochastic evolutionary process with no information about the complexity, the shape and the size of the expression can find at least competitive discriminative TWS.

2 EVOLVING TERM WEIGHTING SCHEME USING GENETIC PROGRAMMING

A TWS is a combination of statistical information. It is intended to measure the discriminative power of a term, i.e. it tells how much a term is related to a certain category. These statistics combined by means of mathematical operators and functions.

We are interested in automatically evolving a TWS (an individual) using GP.

In our context of automatically evolving term weighting methods, an individual is a combination of the function set that is built with simple arithmetical operators (+,-,*,/,log,...) and the terminal set (constant values and inputs to our problem).

As in most conventional GP approach, programs (generated TWS) are depicted as trees. In this problem, the terminal nodes consist of statistical information extracted from training data, while the inner nodes are a set of defined operators that combines the statistical information to form a new TWS.

In this study, we try to generate new TWS by evolving the Collection Frequency factor and then combines it with the Term Frequency factor. The CF factor is a combination of constants, statistical information (N, N_t,...), and mathematical operators. Hence we define the terminals as the statistical information shown in Table 1. Regarding the mathematical operators, they are defined as one of the following ($+$, $-$, $/$, $*$, $\sqrt{x}$, $log1(x) = \log(1 + x)$ and $log2(x) = \log(2 + x)$).

We should note that the statistical information has different types (single value, vector, and matrix). For instance, the number of documents in the training data N is a constant (single value), the number of documents that contains a term t is a vector containing the number of documents for each term and finally, the number

Table 1: Statistical information (Terminals) used to evolve a TWS.

Label	Description
N	Total number of documents
C	Number of categories
C_t	Number of categories that contain the term t
N_t	Number of documents that contain t
$\overline{Nt}$	Number of documents that do not contain t
N_{cat}	Number of documents in the positive category cat
$\overline{N_{cat}}$	Number of documents that do not belong to cat

Table 2: Parameters used in our genetic program.

Parameter	Value	
Population Size	100	
Max Individual Size	20	
Number of generations	100	
Function set	$+, -, /, *, \sqrt{x}$, $log(x) = \log(x)$	
Terminal set	$a, b, c, d, N, N_t, \overline{N_t}$, $N_{cat}, \overline{N_{cat}}, C, C_t$	
Mutation	Type	OnePointMutation
	Probability	1/indvidual size
CrossOver	Type	SubtreeCrossover
	Probability	0.85

of documents that belongs to a category *cat* and contains a term t is a matrix. Operations on these different types of statistical information are taken care of by Eigen library using element-wise transformations.

In GP, a set of individuals is initialized and then evolved according to a set of genetic operators. At first, we randomly generate a random size individuals with a max size of twenty genes (the max size could be overpassed during the cross-over operation). As for genetic operators, we use the elite selection and re-insertion, a subtree crossover with a probability of 0.85 and one point mutation with a probability of 1/size of the individual.

In order to assess the performance of generated TWSs, classification models are evaluated using the f_1 measure.

Table 1 shows the statistical information used as terminal set for generating formulas (the function set) which represent TWSs. Table 2 shows the parameters used in the genetic programming algorithm. As it can be seen, the function set is made of very simple arithmetical functions while the terminal set includes to the best of our knowledge all the statistical information used to build a TWS.

3 RESULTS

In our experiments, we have used three widely well-known benchmarks in TC: Reuters-21578 Benchmark Corpus[1], Oshumed Benchmark Corpus[2] and the 4 Universities data set also called Webkb[2].

[1]http://disi.unitn.it/moschitti/corpora.htm
[2]http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/

Table 3: Average classification performance of random TWS learned for a single-label task on its corresponding data set and the best baseline. The selected TWS is randomly chosen between the best generated TWSs for each category.

	GP-Based		Baseline	
Data Set	TWS	f_1	f_1	Best
Reuters	$(\frac{a}{cN} * \log(2+c))C^2$	86.88	85.92	tf.rf
Oshumed	$\frac{a}{d*(N_t+log(2+C_t))}$	60.30	57.10	tf.chi
Webkb	$\log(1 + log(2 + a))$	88.43	87.53	tf.rf

Regarding Reuters-21578, the generated TWSs and the baseline schemes have similar performances. However, on Oshumed and Webkb data sets, the GP-Based TWSs outperforms the best baseline schemes.

From Table 3, we can see that the average performance (macro-f_1) of the generated TWSs outperforms the best baseline on the three corpora which means that the three learned TWS have good generalization performance.

4 CONCLUSION

In this paper, we have studied the benefits of using genetic programming for generating term-weighting schemes for text categorization. Unlike previous studies, we generate formula by combining statistical information at a microscopic level. This kind of generation is new, and we can conclude that :

- Different data sets require a different formula. This means that having a good generic formula is really hard to find.
- Within a corpus, it is even better to use a different formula for each category. The hard task is to find the best for each one.
- Genetic programming is able to find very good formulas which outperform standard formulas given by experts in the literature.
- Eventually, even if the generated formula is specific to a given category, results show that the best formula for one category is generic enough to be good (but not best) for other categories.
- Further learning can be done in order to find more generic and robust formula.

REFERENCES

[1] Tristan Cazenave. 2010. Nested Monte-Carlo Expression Discovery.. In *ECAI*. 1057–1058.
[2] Akshi Kumar and Teeja Mary Sebastian. 2012. Sentiment analysis on twitter. *IJCSI International Journal of Computer Science Issues* 9, 3 (2012), 372–378.
[3] Efstathios Stamatatos. 2009. A survey of modern authorship attribution methods. *Journal of the Association for Information Science and Technology* 60, 3 (2009), 538–556.
[4] Konstantin Tretyakov. 2004. Machine learning techniques in spam filtering. In *Data Mining Problem-oriented Seminar, MTAT*, Vol. 3. 60–79.
[5] Marc A Zissman. 1996. Comparison of four approaches to automatic language identification of telephone speech. *IEEE Transactions on speech and audio processing* 4, 1 (1996), 31.