Towards the Automated Generation of Term-Weighting Schemes for Text Categorization

Mauricio Garcia, Hugo J. Escalante, Manuel Montes, Alicia Morales, Eduardo Morales
INAOE, Luis Enrique Erro 1, Puebla, 72840, Mexico
{mauricio.garcia, hugojair, mmontesg, a.morales, emorales}@inaoep.mx

ABSTRACT

This paper describes ongoing research on the use of genetic programming to learn term-weighting schemes to be used for text classification. A term-weighting scheme (TWS) determines the way in which documents are represented before applying a text classification model. We propose a genetic program that aims at learning an effective TWS that can improve the performance in text classification. We report preliminary experimental results that give evidence of the validity of the proposal.

Categories and Subject Descriptors

I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*Text analysis*

Keywords

GP; Text classification; Representation learning.

1. INTRODUCTION

Text classification (TC) is the task of associating documents with predefined categories that are related to their content. TC is an important and active research field because of the large number of digital documents available and the consequent need to organize them. TC is a typical problem that has been approached with pattern classification methods, where documents are represented as numerical vectors and standard classification methods (e.g., SVM, KNN) are applied. A crucial component of TC systems is the TWS, which specifies how a document is represented in a way that classifiers can be applied.

Given a TWS, a document d_i is represented by a numerical vector $\mathbf{w}_i$ of length equal to the size of the vocabulary V (the set all the different terms/words that appear in a corpus). Each element $w_{i,j}$, $j = 1, \ldots, |V|$, of $\mathbf{w}_i$ indicates how relevant word j is for describing the content of document i, the value of $w_{i,j}$ is deterimed by the TWS. Many TWS have been proposed so far, including unsupervised (e.g., Boolean and TFIDF) and supervised schemes (e.g., TF-IG and TF-CHI). Unsupervised TWS are the most used ones, for example, under the Boolean TWS $w_{i,j} = 1$ iff word j appears in document i and 0 otherwise. Under the term-frequency (TF) TWS, $w_{i,j} = \#(i,j)$, where $\#(i,j)$ accounts for the times word j appears in document i. On the other hand, supervised term-weighting schemes aim at incorporating discriminative information into the weighting scheme. For instance in the TF-IG scheme, $w_{i,j} = \#(i,j) \times I(j)$, is the product of the TF weight for word j and document i with the information gain of word j ($I(j)$). In this way, the discrimination power of each word is taken into account for the document representation.

Although acceptable performance has been reported with existing TWS, it is still an art determining the adequate TWS for a particular data set. Besides, one should note that most term weighting schemes are combinations of other word-document (e.g., $\#(i,j)$) and/or word (e.g., $I(j)$) weighting factors. Hence, it is worth asking ourselves whether better weighting schemes than the ones proposed so far can be obtained by merging the already known TWS. In this direction, this paper presents preliminary results on the use of genetic programming for learning TWSs to be used in TC tasks. A genetic program is proposed in which a set of basic TWSs (terminals) are combined through arithmetic operators in order to generate alternative schemes that can improve the classification performance. We report experimental results in a number of data sets that show the proposed formulation is a promising solution to term-weighting learning.

To the best of our knowledge, this is the first effort trying to learn TWSs from existing ones. The most related work is that from [1], where genetic programming is used to evolve weighting schemes for information retrieval. The main differences rely in that: we focus on TC instead on information retrieval; we learn TWS for a data set using the data set itself (not requiring additional collections as in [1]); we maximize the classification performance using labeled training samples, instead of requiring queries and the corresponding relevance-judgments, which are rather difficult to obtain.

2. LEARNING TWS

For learning TWS we propose a genetic program with standard tree-representation using as terminals a set of basic TWSs widely used in the TC literature [2, 3], see Table 1. The genetic program will learn how to combine basic TWSs to generate better schemes for TC. We consider term-document TWSs (e.g., $\mathbf{W}_5$, $\mathbf{W}_6$, $\mathbf{W}_{21}$), term weights (e.g.,

GECCO'14, July 12–16, 2014, Vancouver, BC, Canada.
ACM 978-1-4503-2881-4/14/07.
http://dx.doi.org/10.1145/2598394.2602286.

$\mathbf{W}_3$, $\mathbf{W}_4$) and constants (e.g., $\mathbf{W}_1$). Each (non-constant) terminal is a matrix of size $m \times n$, with m the number of training documents and n the number of terms. Term-document TWSs are matrices already, for term weights we generate a matrix where each row is the corresponding term-weight vector (constants are treated as scalar constants). The function set for combining these terminals is the set of arithmetic operators: $\{+, -, *, /, \log_2 x, \sqrt{x}, x^2\}$. The population is initialized with Ramped Half-and-Half and the selection method was half-elitism. Standard mutation and crossover operators were considered.

Table 1: Terminal set

Variable	Meaning
$\mathbf{W}_1$	$\|D\|$, Constant array that stores the total number of documents.
$\mathbf{W}_2$	$\|W\|$, Constant array that stores the total terms.
$\mathbf{W}_3$	C, Matrix containing in each row vector Chi^2.
$\mathbf{W}_4$	I, Matrix containing in each row vector IG.
$\mathbf{W}_5$	$TF-IDF$, An array containing the TF-IDF term weighting.
$\mathbf{W}_6$	TF, An array containing the TF term-weighting.
$\mathbf{W}_7$	FGT, Array containing the global frequency of terms in the document.
$\mathbf{W}_8$	TP, Matrix containing in each row vector of True Positives.
$\mathbf{W}_9$	FP, Matrix containing in each row vector of False Positives.
$\mathbf{W}_{10}$	TN, Matrix containing in each row vector of True Negatives.
$\mathbf{W}_{11}$	FN, Matrix containing in each row vector of False Negatives.
$\mathbf{W}_{12}$	$Accuracy$, Matrix containing the Accuracy per FP (term, class).
$\mathbf{W}_{13}$	$Accuracy\_Balance$, Matrix containing the AC_Balance each (term, class).
$\mathbf{W}_{14}$	BNS, An array that contains the value for each BNS per (term, class).
$\mathbf{W}_{15}$	$DFreq$, DocumentFrecuency matrix containing the value for each (term, class).
$\mathbf{W}_{16}$	$FMeasure$, FMeasure matrix containing the value for each (term, class).
$\mathbf{W}_{17}$	$OddsRatio$, An array containing the OddsRatio term-weighting.
$\mathbf{W}_{18}$	$Power$, Matrix containing the Power value for each (term, class).
$\mathbf{W}_{19}$	$ProbabilityRatio$, Matrix containing the ProbabilityRatio each (term, class).
$\mathbf{W}_{20}$	Max_Term, Matrix containing the vector with the highest repetition for each term.
$\mathbf{W}_{21}$	RF, Matrix containing the RF vector.
$\mathbf{W}_{22}$	$RF * TF$, Array containing $RF * TF$ (element by element)

As fitness function we use the TC performance of a linear SVM when using the documents represented by the TWS obtained from each individual. The fitness function is estimated using $k-$fold cross validation ($k = 3$) over a subset of the original training documents this in order to reduce the computational cost and avoid overfitting (to some extent). The TWS achieving the highest classification performance is returned by the genetic program, this TWS is then evaluated in a test data set, that was not used during the optimization process.

3. EXPERIMENTAL RESULTS

We performed experiments in a suite of benchmark data sets. On the one hand, we consider 2 thematic categorization (TC) and 6 authorship attribution (AA) data sets, see Table 2. In AA the categories are authors, hence the problem is to associate documents with authors. We applied the proposed GP for 50 generations in each of these data sets, using a population size of 50 individuals.

Table 2: Data sets considered for experimentation.

Type	Data set	Authors	Terms	Train	Test
TC	20 Newsgroup	20	61188	11269	7505
TC	Reuters	8	5310	5459	2179
AA	Business	6	10550	85	90
AA	Cricket	4	10044	98	60
AA	Football	3	8620	52	45
AA	Poetry	6	8016	145	55
AA	Travel	4	11581	112	60
AA	CCA	10	15587	500	500

Tables 3 and 4 show the average classification performance (macro f_1 measure) of an SVM in the test partitions when using the TWSs learned with the genetic program, when using the top 1000 most frequent terms and when using all of the terms, respectively. We report the average (across TC, AA and overall, data sets) performance obtained by the different weighting schemes. We report the performance of the best TWS learned by GP for both TC and AA data sets. It can be seen that in average the TWSs learned with the genetic program obtain better average performance than standard schemes (rows 1-5). This result holds for experiments using the top 1000 most frequent terms or when all of the terms were considered, although the classification performance is higher when all features are used. Different TWS were selected for each data set, although we cannot show the learned TWSs it is worth mentioning that most learned schemes contained a few terminals.

Table 3: Results 1000 features

	Baseline	Average AA		Average TC		Overall average	
		Fscore	Std	Fscore	Std	Fscore	Std
1	TF	0.7606	0.1219	0.6592	0.2596	0.7353	0.1498
2	Boolean	0.7639	0.1484	0.6602	0.2313	0.738	0.1602
3	TF-IDF	0.7576	0.059	0.6789	0.2017	0.7379	0.0981
4	RF*TF	0.7606	0.1219	0.6592	0.2596	0.7353	0.1498
5	CHI*TF	0.7339	0.1067	0.6767	0.2166	0.7196	0.1246
6	IG*TF	0.735	0.1143	0.6513	0.1952	0.7141	0.1276
7	GP - Best TC	**0.8108**	**0.1532**	0.6746	0.2431	**0.7768**	**0.1708**
8	GP - Best AA	0.7996	0.1136	**0.7055**	**0.2281**	0.7761	0.1362

Table 4: Results all features

	Baseline	Average AA		Average TC		Overall average	
		Fscore	Std	Fscore	Std	Fscore	Std
1	TF	0.785	0.1136	0.7193	0.1601	0.7686	0.1175
2	BOW	0.7783	0.199	0.7464	0.1394	0.7703	0.1769
3	TF-IDF	0.7305	0.0576	0.7602	0.0894	0.7379	0.0608
4	TF*RF	0.785	0.1136	0.7194	0.16	0.7686	0.1175
5	CHI*TF	0.7583	0.1243	0.7256	0.158	0.7501	0.1218
6	IG*TF	0.7601	0.1235	0.688	0.1458	0.7421	0.1226
7	GP - Best TC	**0.8116**	**0.1667**	0.749	0.1416	**0.7959**	**0.1534**
8	GP - Best AA	0.7895	0.1709	**0.7832**	**0.1203**	0.7879	0.1514

4. CONCLUSIONS

We described a novel method to learn TWSs via Genetic Programming. This method generated TWSs that showed better classification performance than standard schemes, outperforming state-of-art TWSs by a substantial margin. The improvements hold for both: TC and AA data sets, hence we can argue that the proposed approach has appealing robustness and generalization features.

Acknowledgements. This work was partially supported by the LACCIR programm under project ID R1212LAC006. Mauricio García-Limón was supported by CONACyT with scholarship No. 345683.

5. REFERENCES

[1] R. Cummins and C. O'Riordan. Evolving local and global weighting schemes in information retrieval. *Information Retrieval*, 9:311–330, 2006.

[2] G. Forman. An extensive empirical study of feature selection metrics for text classification. *J. of Mach. Learn. Res.*, 3:1289–1305, 2003.

[3] M. Lan, C. L. Tan, J. Su, and Y. Lu. Supervised and traditional term weighting methods for automatic text categorization. *Trans. PAMI*, 31(4):721–735, 2009.