Evolutionary Learning of Meta-Rules for Text Classification

Juan Carlos Gomez
Department of Electronics
Universidad de Guanajuato
jc.gomez@ugto.mx

Stijn Hoskens
Deparment of Computer Science
KU Leuven
stijn_hoskens2@hotmail.com

Marie-Francine Moens
Deparment of Computer Science
KU Leuven
sien.moens@cs.kuleuven.be

ABSTRACT

This paper presents an evolutionary method for learning lists of meta-rules for generalizing the selection of the best classifier for a given text dataset. The method builds rules based on features of a set of training text datasets, and evolves them using special crossover and mutation operators. Once the rules are learned, they are tested in a different set of datasets to demonstrate their accuracy and generality. Our experiments show encouraging results.

CCS CONCEPTS

•Information systems → Association rules; Clustering and classification; •Computing methodologies → Heuristic function construction;

KEYWORDS

Automatic Machine Learning, Text Classification, Genetic Algorithms, Hyper-heuristics

ACM Reference format:
Juan Carlos Gomez, Stijn Hoskens, and Marie-Francine Moens. 2017. Evolutionary Learning of Meta-Rules for Text Classification. In *Proceedings of GECCO '17 Companion, Berlin, Germany, July 15-19, 2017,* 2 pages.
DOI: http://dx.doi.org/10.1145/3067695.3075601

1 INTRODUCTION

Text classification is a popular topic in machine learning and data mining. The task has many applications and there are several methods to solve it. Nevertheless, the methods perform differently depending on the application, and some of them include hyperparameters to tune. Recently, Automatic Machine Learning [1, 2] (AML) has emerged as an approach to optimally find the best method to solve a given machine learning task. However, methods in AML optimize the process for a single task for a single dataset.

In this paper we present the novel Evolutionary Learning of Meta-Rules (ELMR) method for text classification. ELMR uses an hyper-heuristic approach [3] by going through a training evolutionary process to learn a set of meta-rules to determine the most appropriate models for classifying text datasets. The rules are built using statistical features of a set of training text datasets and evolved using adapted crossover and mutation operators. After learning the rules, ELMR tests them for accuracy and generality with a different set of datasets. Our experiments show that ELMR is able to find a set of short rules that produce a near optimum performance.

2 EVOLUTIONARY LEARNING OF META-RULES

ELMR first splits a set of text datasets in two parts, genetic training and genetic test, and it splits each genetic part in a training set and a test set. For each training set, it calculates a set of statistical features: Principal Component Analysis Coefficient (pcac); number of documents; number of categories; average, standard deviation, ratio of average and standard deviation and entropy for the number of documents per category and words per document. The pcac is the fraction of the variance captured in the first 10 principal components: $pcac = \frac{\sum_{i=1}^{10} \sigma_i^2}{\sum_{i=1}^{m} \sigma_i^2}$, with σ_i as the i-th singular value of the correlation matrix $\mathbf{X}^T\mathbf{X}$, with $\mathbf{X}$ as the term-document matrix. Afterwards, ELMR uses a genetic algorithm to evolve a population of lists of rules as individuals. A list of $n + 1$ rules has the form:

```
IF CONDITION1 THEN ACTION1
...
IF CONDITIONn THEN ACTIONn
ELSE ACTION(n+1)
```

A condition is a conjunction of several exclusive range checks. A range check tests if a given feature of a dataset is between a lower and an upper bound (one of the bounds can be optional). A condition is satisfied if all of its range checks succeed. We limit the maximum number of rules in a list to 11 (i.e. the number of dataset features), including the if statement, the minimum to 2, and the number of range checks in a condition to 3. An example of a condition is: `feature2 <0.6 && 52.6 <feature5 <Infinity`

The actions in the rules correspond to classifiers. ELMR uses the following: Multinomial Naive Bayes (NB), K-Nearest Neighbors (KNN) with K=1, 2, 3, 4, 5, 10 and cosine similarity, Support Vector Machines (SVM) and Logistic Regression (LR) with the regularization parameter C=0.01, 0.1, 1, 10, 100. Additionally, LR and SVM include the L1 or L2 regularization (primal and dual forms), and SVM the method by Crammer & Singer (C&S). In total there is a pool of 42 possible classifiers/actions.

Initially, ELMR creates a population of lists at random, and creates subsequent children populations by crossover. Crossover selects two parent individuals at random and does one of two actions: 1) randomly chooses one rule from each parent and switches their actions, 2) switches portions of the lists between parents. ELMR mutates with a given probability each individual in the new population. There are 5 mutation operators: 1) select a random rule and

GECCO '17 Companion, Berlin, Germany

DOI: http://dx.doi.org/10.1145/3067695.3075601

change its condition, 2) or its action, 3) swap two rules in the list, 4) insert a new rule in the list, 5) delete a rule from the list.

In each generation ELMR evaluates every list of rules using all the genetic training datasets. The list takes the training part of each genetic training dataset, determines what rule applies for such dataset, trains the classifier attached to that rule using the training part and classifies the test part of the genetic training dataset to obtain an accuracy. The fitness function for a list is the average accuracy after performing classification in all the genetic training datasets. In each generation, parent and children populations are merged and the lists with the highest accuracy are selected.

After the evolutionary process, ELMR evaluates the list with the highest accuracy with the genetic test datasets as before: testing what rule satisfies the training part of a genetic test dataset, training the attached classifier using that training part and classifying the test part of the genetic test dataset. ELMR obtains a final average accuracy after classifying all the genetic test datasets.

We implemented ELMR in Java, using Weka[1] for the Naive Bayes classifier, Liblinear[2] for all the versions of SVM and LR, and a proprietary implementation for KNN.

3 RESULTS

In our experiments we used nine text datasets with different types of documents: news (20ng, R52, RCV1), web pages (DMOZ, WebKB), journal abstracts (Classic4, CoRA), movie reviews (Movies) and patents (WIPO). We subsampled datasets RCV1, WIPO and DMOZ to 34,000 documents to avoid memory issues.

We formed the genetic training and test parts using for each 50% of each dataset, and split the genetics part in 70% for training and 30% for testing. For each dataset ELMR extracted word features and then we chose to filter-out stopwords, words appearing in less than 3 documents, words with alphanumeric characters and words with only one character (feature selection eliminated some documents). ELMR weighted the word features in each dataset using tf-idf to form a term-document matrix, and normalized each document vector to 1, and then it calculated the statistical features for the training sets of each genetic part. Table 1 shows the statistical features for the training sets of the genetic training datasets.

Table 1: Feature values for the genetic train datasets

	20ng	clssc	cora	dmoz	mov	r52	rcv1	webkb	wipo
#Docs	5710	2436	4153	11900	694	3256	11590	1406	11900
#Tops	20	4	70	11	2	51	78	4	114
dptAvg	285.5	609	59.33	1081	347	63.84	148.6	351.5	104.4
dptStd	29.58	326.3	38.33	1372	7.07	225.3	942.5	166.6	145.8
std/avg	0.10	0.54	0.65	1.27	0.02	3.53	6.34	0.47	1.40
dptEntr	4.12	2	5.58	3.46	1	4.74	5.22	2	6.29
wpdAvg	119.1	53.34	74.05	375.1	273.3	61.68	170.2	140.2	776.9
wpdStd	238.8	41.32	45.87	1308	117.5	63.31	128.4	559.0	608.0
std/avg	2.01	0.78	0.62	3.49	0.43	1.03	0.76	3.99	0.78
wpdEntr	7.88	6.80	7.18	9.14	8.16	7.11	8.44	7.94	10.05
pcac	0.67	0.40	0.96	0.80	0.48	0.88	0.95	0.87	0.64

For the genetic algorithm we used a mutation probability of 0.3, a population of 200 and 300 generations. In order to speed up the optimization process, every time ELMR trains and tests a classifier with a specific dataset, it stores the estimated accuracy in a cache.

Table 2: Optimal classifiers for the genetic test datasets

Dataset	Classifier	Accuracy
20ng	NB	81.09
clssc	NB	94.96
cora	L2-regularized SVM (dual) C=0.01	54.16
dmoz	L2-regularized LR (dual) C=0.01	69.29
mov	L1-regularized LR C=10	78.37
r52	L1-regularized SVM C=1	91.97
rcv1	L2-regularized SVM (primal) C=0.01	90.45
webkb	L2-regularized LR (dual) C=0.1	89.07
wipo	L2-regularized LR (primal) C=0.01	54.24

Table 2 shows the optimal classifiers for every genetic test dataset. These were obtained after training and testing all the possible classifiers with all the genetic test datasets. This gave an average optimal accuracy of 78.18%.

For obtaining more robust results, we set ELMR to perform 500 independent runs of the complete evolutionary process and then to merge all the produced intermediate optimal lists of rules in a single final list. ELMR merges the lists as follows. 1) It finds all the tuples <feature,range_check,classifier>, if a rule contains more than one range check, it is split in the corresponding tuples. 2) It merges the range checks by averaging its bounds. 3) It selects the more frequent tuples per classifier. 4) Finally it selects the 5 tuples with the highest average performance, and consider the fifth as the `ELSE` statement. The final list obtained in our experiments is:

```
IF wpdStd>399.7 THEN L2-Regularized LR (primal) C=0.01
IF pca>0.76 THEN SVM (C&S) C=1
IF nbOfTopics>35.2 THEN SVM (C&S) C=1
IF nbOfDocs<4853.6 THEN L1-Regularized LR C=10
ELSE Naive Bayes
```

Evaluation of this list with the genetic test datasets produces an average accuracy of 77.74%, which is very close to the optimal performance of 78.18%. We observe that models based on LR and SVM appear at the top of the list as the most recommended models, and Naive Bayes as a final option. KNN does not appear, showing its results were dominated by the other classifiers.

4 CONCLUSIONS

In this paper we have presented the Evolutionary Learning of Meta-Rules (ELMR) method. ELMR works by using a training evolutionary process to learn a set of meta-rules to determine what are the most appropriate models for classifying text datasets. Our results have shown that the learned meta-rules are able to generalize the model selection process and are able to reach a near optimum performance when evaluated on a set of diverse test datasets.

REFERENCES

[1] E. R. Sparks et al. 2015. Automating model search for large scale machine learning. In *Proceedings of the Sixth ACM Symposium on Cloud Computing*. ACM, 368–380.

[2] I. Guyon et al. 2016. A brief review of the ChaLearn AutoML challenge: Any-time any-dataset learning without human intervention. In *Proceedings of the 2016 Workshop on Automatic Machine Learning*. 21–30.

[3] J. C. Gomez and H. Terashima-Marín. 2012. Building general hyper-heuristics for multi-objective cutting stock problems. *Computación y Sistemas* 16, 3 (2012), 321–334.

[1] www.cs.waikato.ac.nz/ml/weka/

[2] www.csie.ntu.edu.tw/~cjlin/liblinear/