Towards an Automated Classification of Spreadsheets

Jorge Mendes[1], Kha N. Do[2], and João Saraiva[1]

[1] HASLab, INESC TEC & Universidade do Minho, Portugal
{jorgemendes,jas}@di.uminho.pt,
[2] University of Science, Vietnam National University - Ho Chi Minh
dnkha@fit.hcmus.edu.vn

Abstract. Many spreadsheets in the wild do not have documentation nor categorization associated with them. This makes difficult to apply spreadsheet research that targets specific spreadsheet domains such as financial or database.
We introduce with this paper a methodology to automatically classify spreadsheets into different domains. We exploit existing data mining classification algorithms using spreadsheet-specific features. The algorithms were trained and validated with cross-validation using the EUSES corpus, with an up to 89% accuracy. The best algorithm was applied to the larger Enron corpus in order to get some insight from it and to demonstrate the usefulness of this work.

Keywords: Spreadsheets, Data Mining, Classification

1 Introduction

Spreadsheets are widely used at all levels of organizations. In fact, they are used both from professional programmers at large worldwide organizations, to non-professional programmers in small family-run businesses. As recent research studies [8, 1] and frequent reports of horror stories[3] show, spreadsheets are prone to errors. Recently advanced techniques have been proposed (some of them already incorporated in regular programming languages), in order to improve both the efficiency and productivity of spreadsheet users. To support such ongoing research activity, several spreadsheet corpora have been proposed in the literature [4, 3, 7] so that researchers can experiment their techniques in a corpus that represent real-world spreadsheet applications. For example, the EUSES corpus [4] divides its 5607 spreadsheets in 11 distinct categories, including *finances*, *databases*, etc. As a consequence, researchers can apply their techniques to one specific application domain of spreadsheets.

The classification of software artifacts, in particularly source code files, are usually performed by the administrator of a repository [15]. The EUSES corpus

[3] Please see the spreadsheet horror stories available at http://www.eusprig.org/horror-stories.htm.

is no exception, and its creators gathered spreadsheets from different sources and put them together in a single repository for easy access by researchers. When dealing with large corpora, this process can be tedious and time consuming. Thus, it is not surprising that the large Enron spreadsheet repository [7] is not classified yet.

This paper presents the use of automated software classification algorithms in determining the appropriate application domain for a particular spreadsheet. We configure well-known classification algorithms with spreadsheet-specific properties. The EUSES corpus is used as the basis for training and testing the classification algorithms. In this training study we considered five different classification algorithms, which are provided by the widely used Java-based Weka machine learning suite [14, 6]. Our first experimental results show that the best spreadsheet classification algorithms are based on decision trees, which correctly classifies 89% of the spreadsheets during cross-validation.

In order to perform the feature extraction and data preprocessing, we developed a Java-based tool to interact directly with both spreadsheets and Weka. This helps to automate the whole process: spreadsheets have their features automatically extracted and then packed in a file format compatible with the Weka machine learning suite.

Having defined the best classification algorithm for spreadsheets, we then automatically applied the classification process to the Enron repository. We were able to: evaluate the performance of the process, get some information about biases from the EUSES training, get some insight on the Enron corpus from the point of view of the EUSES corpus. These results are available to the spreadsheet research community and show that further work on this subject is required.

The remaining of this paper is structured as follows: Section 2 describes the spreadsheet classification process: the EUSES spreadsheet corpus, the spreadsheet specific features used in the classification algorithms, and the five used classification algorithms. Section 3 briefly describes our spreadsheet classification framework. Section 4 contains the experiments we performed and the results obtained with the five classification algorithms and the spreadsheet specific features. Section 5 presents the results of classifying the Enron corpus. Section 6 discusses our results and, finally, we conclude with Section 7.

2 Classification Environment

The classification of software artifacts [14] is usually performed in the following steps:

- select data to train classification algorithms
- preprocess the data
- train the algorithms
- evaluate the derived models
- classify new artifacts

In the classification of spreadsheets we also followed these steps. First, we sampled the EUSES corpus to obtain a training set (Section 2.1). Then, we preprocessed the spreadsheet data to extract features from the training set spreadsheets (Section 2.2). Next, we considered and applied several classification algorithms to the training set to obtain different classification models (Section 2.3). After, we evaluated all models using a five fold cross validation with this sampled dataset (Section 4). Then, the best classifier will be used to classify new spreadsheet instances, namely the Enron dataset (see Section 5).

2.1 The EUSES Spreadsheet Corpus

The data used to classify the algorithms is extracted from the EUSES spreadsheet corpus [4]. Most of the spreadsheets in this corpus were obtained from the Internet through searches using the Google search engine [5], but some of them result from other researchers or individuals. It has a total of 5607 spreadsheet files, organized in 11 distinct categories:

- cs101
- database
- filby
- financial
- forms3
- grades
- homework
- inventory
- jackson
- modeling
- personal

Some processing was already applied to this corpus. Each of these categories has up to three directories: *bad*, *duplicates*, and *processed*. The *bad* directories contain files that the authors of the EUSES corpus were unable to use for some reason[4]. The *duplicates* directories, as the name suggests, contain duplicate files. The *processed* directories contain the remaining files.

From the available categories, only six were kept for classification due to the reduced number of spreadsheets in the other categories (see Table 1). The six categories kept are: *database*, *financial*, *grades*, *homework*, *inventory*, and *modeling*. All the spreadsheets in these categories are from the Internet searches. Moreover, only the files in the *processed* directories were taken into account for the classification, resulting in a total of 4402 spreadsheet files, with an average of 734 files per category.

2.2 Feature Extraction

Sets of spreadsheet files are not directly usable to train a classifier. Thus, a preliminary step that extracts features from the spreadsheets is required.

Spreadsheets have many attributes that can be extracted as features, hence a selection must be made. Starting from common knowledge, having in mind the selection of attributes that could distinguish spreadsheet categories, only the words present in cell contents were extracted. Each word is considered an

[4] This information is not clearly specified by the authors, but range from password protected files to spreadsheets with disruptive macros [4].

Table 1. Spreadsheet file count in the EUSES corpus.

	Total	*bad*	*duplicates*	*processed*
cs101	9	1	0	8
database	904	59	125	**720**
filby	45	0	0	45
financial	902	31	91	**780**
forms3	26	0	0	26
grades	895	17	148	**731**
homework	951	29	239	**683**
inventory	891	49	86	**756**
jackson	13	0	0	13
modeling	966	51	183	**732**
personal	5	0	0	5
Total	5607	236	872	4499

attribute, and its value for each spreadsheet is the number of occurrences of that word in it. This makes the *words* feature.

The extraction process is as follows. If a cell contains a sentence, this sentence is split into the several words that compose it. The resulting set of words passes then through a cleaning process, where words that are present in all the categories are removed from the set. Moreover, words that appear in less than 10% of the spreadsheets in a given category are removed from the set of words in that category.

2.3 Algorithm Selection

Several algorithms are available to classify software artifacts based on the extracted features. In order to select the one that best suits spreadsheet classification based on the mentioned features, several experiments were performed with Weka. The following algorithms from the Weka suite were used in these experiments:

- DecisionTable – Implementation of the IDTM algorithm [11]
- J48 – Java implementation of the C4.5 algorithm to generate decision trees [13]
- REPTree – A decision tree learner
- NaiveBayes – Implementation of a Naive Bayes classifier [9]
- NaiveBayesMultinomial – Implementation of a multinomial Naive Bayes classifier [12]

3 SSClassifier: A Java/Weka-based Spreadsheet Classifier

In order to automatically classify large data sets of spreadsheets, like the EUSES and Enron corpora, we developed a Java-based tool to process in batch textual spreadsheets. This was accomplished using the Apache POI [2] library

to read the spreadsheet files and access their contents. The extraction of spreadsheet features, as described in Section 2.2, was directly implemented in the Apache POI spreadsheet representation. Then, we implemented a bridge between the Apache POI and the Weka data representations, so that we could experiment with different classification algorithms and spreadsheet features. The architecture of the developed framework, named SSClassifier, is presented in Fig. 1.

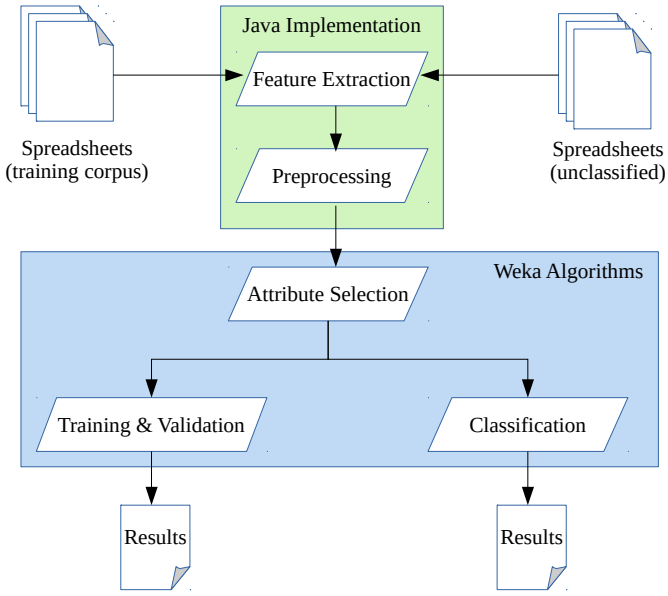

Fig. 1. Architecture of SSClassifier.

The SSClassifier is publicly available from:
`https://bitbucket.org/SSaaPP/spreadsheet-classification/`

4 Experiments

Several experiments were performed to select the best set of attributes and algorithms from the ones defined in the previous section. A common flow was defined for the several algorithms (depicted in Fig. 2), where we then experimented with different inputs using five-fold cross-validation.

The attributes in the *words* feature consist in counts of words. The words are the ones present in the spreadsheet contents, and some filtering is required in order to obtain better results, much like with Natural Language Processing (NLP). Some filtering was already applied, as described in Section 2.2, namely removing words present in all categories and that do not provide any new information

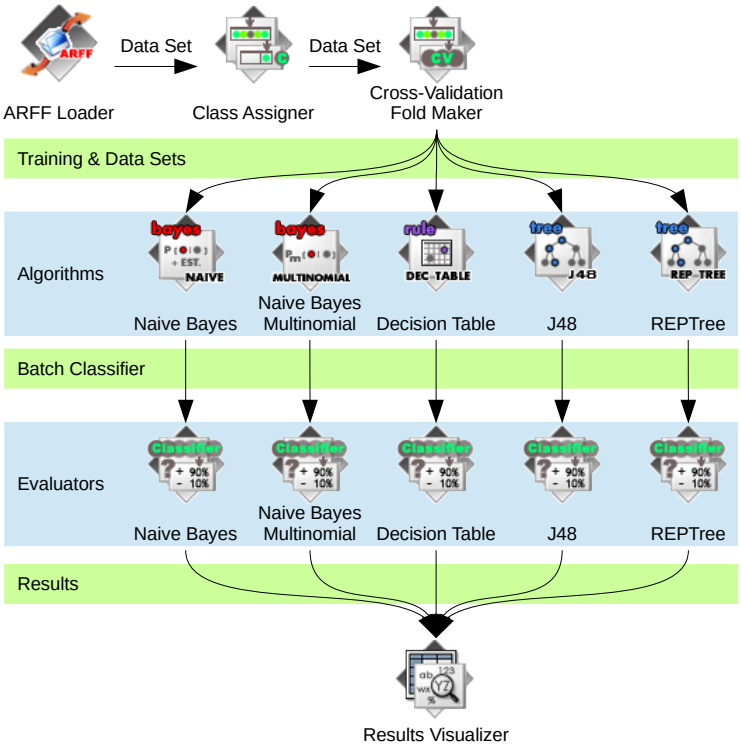

Fig. 2. Experiment flow layout.

(analogously to stop words in NLP), and discarding words that appear only in a small subset of spreadsheets from a category.

The attributes used and their order is important in order to generate better models. Thus, a selection of the attributes and a reordering was performed to select the best option. The different sets of data based on the set of all the word counts present in spreadsheets used are:

A full set of data;
B selection using the *CfsSubsetEval* attribute evaluator and *BestFirst* search method;
C selection using the *CorrelationAttributeEval* attribute evaluator and *Ranker* search method;
D selection using the *GainRatioAttributeEval* attribute evaluator and *Ranker* search method;
E selection using the *InfoGainAttributeEval* attribute evaluator and *Ranker* search method;
F selection using the *ReliefFAttributeEval* attribute evaluator and *Ranker* search method.

Only the data set that went through *CfsSubsetEval* has less attibutes. Only the ones relative to the words, *database*, *financial*, *grades*, *homework*, *inventory*, *modeling*, *size*, and *west* are kept. The other data sets (from C to F) have only the order of the attributes changed.

After putting each of these data sets through the experiment flow, it is possible to see that some options provide better results than others. The results are presented in Table 2. The best overall algorithm, the best overall data set, and the best overall result are displayed in bold font face.

Table 2. Five-fold cross-validation results using the *words* feature.

	A	**B**	C	D	E	F
NaiveBayesMultinomial	57.8846	82.3473	57.8846	57.8846	57.8846	57.8846
NaiveBayes	41.0046	50.6158	41.1736	41.1736	41.1736	41.1253
J48	87.8773	88.0705	87.4909	87.8532	87.829	87.7324
REPTree	88.9882	87.8049	**89.1331**	89.0365	88.9882	89.0848
DecisionTable	85.0278	85.4866	84.9070	84.9070	84.9070	84.9070

From the results, we can notice that the algorithm with the best overall accuracy is *REPTree*, providing the best result with the C data set. The data set B, with only 8 word attributes, improves considerably the results for the *NaiveBayesMultinomial* algorithm, with the *NaiveBayes* algorithm also encountering some improvements. However, the other algorithms suffer in terms of accuracy, but only slightly. Nevertheless, this data set provides a lot of improvements in terms of time for model training. All the resulting data from this work is provided with the source code of the tool.[5]

5 Classifying the Enron Corpus

The Enron corpus [10] is an email data set that was released to the public. This data set was processed in order to remove private and confidential data, but many emails and respective attachments still remain.

Hermans and Murphy-Hill [7] analyzed the Enron email data set and found spreadsheets as attachments in those emails. They extracted those spreadsheets and provided it as its own corpus[6].

The Enron spreadsheets have been submitted through a similar process than the one applied to the EUSES corpus in order to classify them. First, all spreadsheets were preprocessed in order to extract the *words* feature. Some of the spreadsheets were not analyzed due to size limits or due to not being supported by our toolset; 210 spreadsheets were left out. Then, this data was classified using the *REPTree* algorithm that was trained with the data from the EUSES

[5] `https://bitbucket.org/SSaaPP/spreadsheet-classification/src/2259e60/paper/data/`

[6] The Enron spreadsheet corpus is available through here: www.felienne.com/enron

corpus after the *CorrelationAttributeEval* attribute selection process. The results are presented in Table 3.

Table 3. Results of the prediction on the Enron corpus.

Prediction	Count
database	2039
financial	3176
grades	448
homework	8915
inventory	1057
modeling	83
Total Result	15718

The results of the classification of the Enron corpus are very preliminary and need a proper validation that due to time limitations we were unable to include in this paper. As we can notice, most of the spreadsheets are classified as *homework*. In fact, the *homework* class in the EUSES original classification includes a large set of different domains. This results in a large vocabulary for that category in the training of the classification algorithms. Of course, the original data set (the EUSES corpus) and its classification, that we use to train the algorithms, does influence the results. A proper validation of our preliminary results is needed, indeed.

6 Discussion

The Apache POI [2] library was used to read the spreadsheet files and access their contents. However, it several limitations. Its Excel file support is considerably limited, with support only for recent file formats:

- Excel '97(-2007)
- 2007 OOXML

From the *processed* spreadsheets in the EUSES corpus, 261 spreadsheets were discarded (around 6%) due to the lack of support for them from Apache POI. Even for the supported file formats, many features are not available or are very limited, e.g., charts and pivot tables. Thus, we were highly constrained in the features to extract for classification. The issues of using Apache POI can be solved by switching to LibreOffice Calc or Microsoft Excel extensions[7], which have better support for these file formats. Nevertheless, Apache POI is a relatively simple point of entry for spreadsheet analysis, thus its use in this work.

The EUSES corpus was used as a basis for this work. This corpus already has some kind of categorization and contains many spreadsheets. However, the

[7] Only tools that can be used locally were considered to avoid issues related to transfering much data across networks.

categories for the spreadsheets gathered from Internet searches (i.e., the ones that were used in this work) are not based on spreadsheet characteristics, but on keywords that the EUSES creators thought being commonly associated with spreadsheets. This does not make the categories invalid, but might not reflect what people think about the spreadsheets in those categories. Moreover, some categories may contain some overlap. For example, one expects the *homework* category to contain spreadsheets from different domains since homework can be on diverse subjects.

Hence, two possible issues might arise:

- the categories do not match with what one can find from a random set of spreadsheets;
- the categorization of the spreadsheets was dependent on an Internet search, thus the contents/categorizations might be questionable.

In order to overcome these issues, a large set of spreadsheets can be gathered and then clustering algorithms be ran on them. This would allow to organize the spreadsheets based on their characteristics. Another option is to find spreadsheets with a clear categorization (e.g., from the intended purpose by their creators) and then perform a new classification based on these new spreadsheets and categories. Both of these possible solutions can provide a better training corpus. However, the second solution might not yield a large enough data set to apply data mining techniques.

Nevertheless, the work herein presented allows to augment the EUSES corpus in an automated way with spreadsheets which do not have a category associated with them, but are close to the ones already present in the corpus.

The classification model obtained from the EUSES corpus was applied to the Enron spreadsheet repository in order to try obtaining insight on both the classification process and the spreadsheets in the Enron repository. The high number of *homework* spreadsheets found suggests that the EUSES classification model is inappropriate to classify generically any spreadsheet.

The algorithms used make only a small subset of the available algorithms for classification. This work can be easily expanded to include other algorithms. Moreover, more than a selection and ranking of algorithms for spreadsheet classification, this work provides a methodology and work flow to extract features from spreadsheets, filter them, train and then test classification algorithms.

Another close point is the selection of attributes. Much work can still be done in order to find the best set of attributes for a classification algorithm. Improvements are left for future work.

7 Conclusion

This paper presents an automatic classification technique for spreadsheets. We considered five well-known data mining classification algorithms available in the widely used software classification framework Weka. We considered spreadsheet-specific features when we trained and validated such algorithms with the EUSES

spreadsheet corpus. The decision tree learner algorithm *REPTree* correctly classified 89% of the EUSES corpus using the *words* spreadsheet feature during cross-validation. In order to train and validate the classification algorithm, we developed a Java tool to extract spreadsheet features in order to use them with Weka to process and classify spreadsheet corpora.

References

1. Abreu, R., Cunha, J., Fernandes, J.P., Martins, P., Perez, A., Saraiva, J.: Smelling faults in spreadsheets. In: Software Maintenance and Evolution (ICSME), 2014 IEEE International Conference on. pp. 111–120 (Sept 2014)
2. Apache Software Foundation: Apache POI, `http://poi.apache.org`
3. Aurigemma, S., Panko, R.R.: The detection of human spreadsheet errors by humans versus inspection (auditing) software. In: Proc. of EuSpRIG Conf. (2010)
4. Fisher, M., Rothermel, G.: The euses spreadsheet corpus: A shared resource for supporting experimentation with spreadsheet dependability mechanisms. In: Proceedings of the First Workshop on End-user Software Engineering. pp. 1–5. WEUSE I, ACM (2005)
5. Google: Google, `https://www.google.com`
6. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: An update. SIGKDD Explor. Newsl. 11(1), 10–18 (Nov 2009)
7. Hermans, F., Murphy-Hill, E.: Enron's spreadsheets and related emails: A dataset and analysis. In: Proceedings of the 37th International Conference on Software Engineering - Volume 2. pp. 7–16. ICSE '15, IEEE Press, Piscataway, NJ, USA (2015)
8. Jannach, D., Schmitz, T., Hofer, B., Wotawa, F.: Avoiding, finding and fixing spreadsheet errors – a survey of automated approaches for spreadsheet {QA}. Journal of Systems and Software 94, 129 – 150 (2014)
9. John, G.H., Langley, P.: Estimating continuous distributions in bayesian classifiers. In: Eleventh Conference on Uncertainty in Artificial Intelligence. pp. 338–345. Morgan Kaufmann, San Mateo (1995)
10. Klimt, B., Yang, Y.: Introducing the Enron corpus. In: 1st Conference on Email and Anti-Spam (CEAS) (2004)
11. Kohavi, R.: The power of decision tables. In: 8th European Conference on Machine Learning. pp. 174–189. Springer (1995)
12. Mccallum, A., Nigam, K.: A comparison of event models for naive bayes text classification. In: AAAI-98 Workshop on 'Learning for Text Categorization' (1998)
13. Quinlan, R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers (1993)
14. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2005)
15. Yusof, Y., Rana, O.F.: Classification of software artifacts based on structural information. In: Proceedings of the 14th International Conference on Knowledge-based and Intelligent Information and Engineering Systems: Part IV. pp. 546–555. KES'10, Springer-Verlag, Berlin, Heidelberg (2010)