Classifying Pastebin content through the generation of PasteCC labeled dataset

Adrián Riesco[a], Eduardo Fidalgo[bc], Mhd Wesam Al-Nabki[bc], Francisco Jáñez-Martino[c] and Enrique Alegre[bc]

[a]Summer Internship at the Universidad de León with the VARP research group
[b]Department of Electrical, Systems and Automation, Universidad de León, Spain
[c]Researcher at INCIBE (Spanish National Cybersecurity Institute), León, Spain

Abstract. Online notepad services allow users to upload and share free text anonymously. Reviewing Pastebin, one of the most popular online notepad services websites, it is possible to find textual content that could be related to illegal activities, such as leaks of personal information or hyperlinks to multimedia files containing child sexual abuse images or videos. An automatic approach to monitor and to detect these activities in such an active and a dynamic environment could be useful for Law Enforcement Agencies to fight against cybercrime. In this work, we present Pastes Content Classification 17K (PasteCC_17K), a dataset of 17640 textual samples crawled from Pastebin, which are classified in 15 categories, being 6 of them suspicious to be related to illegal ones. We used PasteCC_17K to evaluated two well-known text representation techniques, ensembled with three different supervised approaches to classify the pastes of the Pastebin website. We found that the best performance is achieved ensembling TF-IDF encoding with Logistic Regression obtaining an accuracy of 98.63%.The proposed model could assist the authorities in the detection of suspicious content shared in Pastebin.

Keywords: Natural Language Processing, Machine Learning, Pastebin, Text Classification, Bag of Words, Term Frequency-Inverse Document Frequency

1 Introduction

In the digital age where all information is shared the online notepad services are an essential meeting point. These web sites allow users to upload and to share anonymously text with a limit of extension far superior to other online platforms. The most common use of such services is to share code snippets or personal experiences. However, although the majority of the shared text is legitimate, these services have been abused by sharing suspicious activities including, but not limited to, child pornography multimedia hyperlinks, drug sales and leaked information. The three most known web applications that offer online notepad services are PiratePad[1], codepad[2], and Pastebin [3].The majority of these services

[1] `http://piratepad.net`
[2] `http://codepad.org/`
[3] `http://pastebin.com`

offer a free membership for users which allows them to publish pastes and to access pastes uploaded by other users. The *paste* is known in Pastebin community as any piece of a text published by users. Pastebin was the first one to provide online notepad services since the year 2002, and in the year 2016, Pastebin registered 18 million visitors, more than 95 million of pastes and a monthly mean of 14.430.000 pastes published. According to Herath[9], pastes from Pastebin could contain suspicious activities such as leaked personal data, credit cards, login credentials or confidential financial documents. Fig. 1 shows an example of an anonymous paste leaking Netflix accounts to the public.

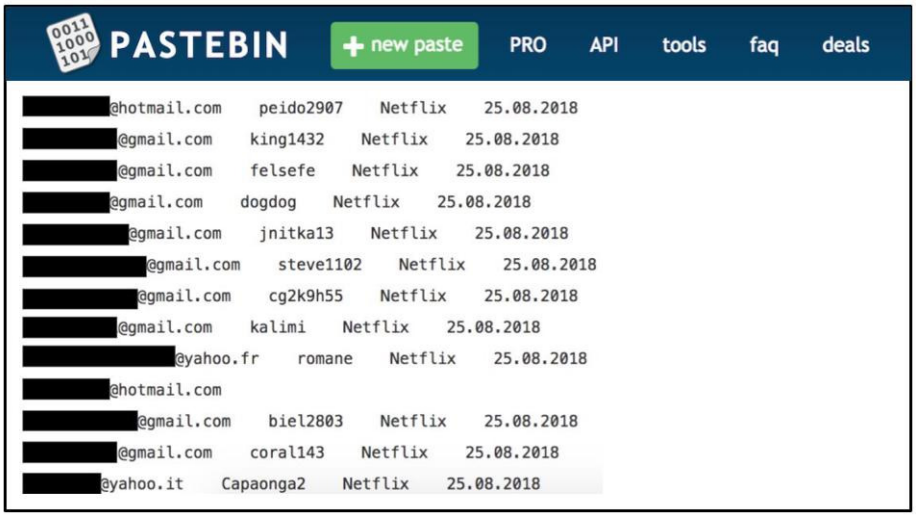

Fig. 1. Paste from Pastebin that leaks Netflix accounts, passwords and verification dates

Motivated by the huge amount of public text data which Pastebin handles, and the possibility of finding suspicious text inside motived us to propose a fully-automatic approach to classify and detect this possible malicious content. To support this task, we first built **Paste**s **C**ontent **C**lassification 17K (PasteCC_17K), a new publicly available dataset with 17640 pastes extracted from Pastebin. Then, we used PasteCC_17K data to train a classifier using six different classification pipelines. We ensembled two encoding methods, Term Frequency–Inverse Document Frequency (TF-IDF) and Bag of Words (BoW), with three different classifiers, Logistic Regression (LR), Support Vector Machine (SVM), and Näive Bayes(NB). Fig 2 illustrates the described process.

The rest of the paper is organized as follows. In Section 2, we review the state of the art on text classification techniques. Next, in Section 3, we expose the followed methodology. After that, Section 4 introduces the PasteCC_17K dataset.

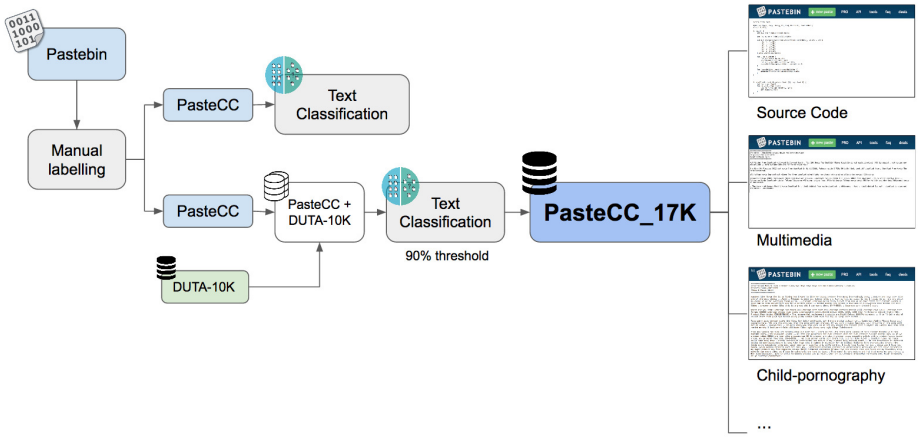

Fig. 2. PasteCC_17K creation process

Section 5 describes the configuration of the experiments and our results along with a discussion. Finally, in Section 6, we report our conclusions by pointing out to possible future works.

2 State of the Art

2.1 Pastebin

Pastebin was created in 2002 by Paul Dixon as a web application for handling pastes. Pastebin does not need authentication, the interface allows users to post pastes without much effort and create a unique URL for each paste, without any limitation on the number of characters, as happens in Twitter, Matic et al.[14].

In recent years, it is increasingly common to find illicit content within Pastebin; hackers tend to publish everything and often use Pastebin for sharing their information. A group of hackers published on Pastebin thousands of names, usernames, passwords, addresses, and phone numbers of students, faculty, and staff members from 53 universities, including Harvard, Stanford, Cornell, Princeton, Johns Hopkins, the University of Zurich and other universities around the world, Perlroth[18].

To the best of our knowledge, the conventional classification and monitoring methods used with Pastebin content depend on a predefined list of keywords, which creation needs a significant amount of time, its maintenance is very costly and become outdated in the short term. For example, *PasteHunter* tool[4] and *AIL-framework* [5] depend heavily on a sequence of rules and regular expressions

[4] `https://github.com/kevthehermit/PasteHunter`
[5] `https://github.com/CIRCL/AIL-framework`

to parse particular tokens such as email addresses, IBAN account numbers, passwords, or even phone numbers. Furthermore, Herath[9] proposed a framework called *LeakHawk*[6] that employs a number of filters and classifiers whereas the first two components screen out the pastes according to a predefined list of keywords, which makes these approaches susceptible to be hacked.

2.2 Machine Learning and Text Classification

Text classification is an important task of Natural Language Processing (NLP), which aim is to separate text documents in different classes or categories based on their content, in a similar way as a human been could do that. Sebastiani [19], presented a survey where discusses the main approaches to text classification using machine learning, discussing in detail the namely document representation, classifier construction, and classifier evaluation. Joachims[11] analyzed the particular properties of SVM and text classification, and recommended using SVM for the classification of text because most of the problems of text classification are linearly separable.

Text classification has been studied extensively during the last few years. Mironczuk and Protasiewicz[16] reviewed every element of text classification and its state-of-art; they also presented a quantitative analysis of studies including the research topics more used, articles per year or articles per journal. Diab and Hindi[7] used three different meta-heuristic methods to find better estimations for the probability terms of NB, these methods were differential evolution, genetic algorithms, and simulated annealing. Silva et al.[20] proposed a novel multinomial text classification technique based on the MDL principle due to the limitations of the traditional classifiers such as SVM or NB with high dimensionality corpus and computational costs for training. The statistical analysis of their results indicated that the proposed technique called MDLText outperformed all of the methods that they evaluated. Zhu and Wong[25] evaluated the five commonly used text classifiers, which are SVM, NB, KNN, AdaBoost and Neural Network. They used an automatically generated text document collection which is labeled by a group of experts. The results verify that different algorithms performed differently depend on document datasets and the need for feature selection method to reduce the high dimensionality of features.

Text Classification is widely used in detection, categorization, and translation domains. Lochter et al.[13] presented an ensemble of text expansion method and the well-known and established classification methods available in literature such as LR, NB, SVM, C4.5 and KNN to detect opinions inside social network. The overall results showed that their ensemble enhanced the performance of the classifiers used individually. Bui et al.[4] presented a different approach to recognize the structure of a PDF to extract raw text in order to be able to classify it later. Panchenko et al.[17] took English text from Common Crawl and constructed a large web-scale corpus using text classification.

[6] `https://github.com/isuru-c/LeakHawk`

2.3 Text Classification using Deep Learning techniques

During the last years appeared some works using Deep Learning models for the text classification process. Since word representation is an essential step for success, Meng et al.[15] proposed an RNN-based generative model for predicting key-phrases in scientific text. Also, Zhang et al.[23] presented a Deep Learning method based on recurrent neural networks to perform automatic key-phrase extraction ensembling keywords and context information to perform the key-phrase extraction task, especially useful in Twitter domains. In terms of text classification, Zhang et al.[24] empirically evaluated the advantages of using ConvNets for the detection of characteristics at character-level. They built several large-scale datasets to demonstrate that the ConvNets model offers better results.

Additionally, Facebook AI Research created two methods based on Deep Learning oriented to text classification. Joulin et al. [12] presented and compared the text classifier called FastText with the recent text classification models based on Deep Learning or Wu et al.[22], who presented StarSpace, a general-purpose neural embedding model that can solve a wide variety of problems. Stein et al.[21] evaluated and compared FastText with different algorithms, including SVM, in the hierarchical text classification. FastText achieved very satisfactory results that showed that the use of word inlays is a very promising approach in hierarchical text classification. These works show that it is necessary to use large datasets to use these Deep Learning methods to obtain good results. Despite the superiority of the broad learning approach in solving various NLP-related problems, there is a need for an enormous number of labeled samples for training, which is not available in the context of our problem.

3 Methodology

We proposed the following methodology to (i) build PasteCC_17K dataset and (ii) to assess the performance of the six pipelines evaluated. First, we applied a preprocessing stage to the text crawled from the Pastebin website. Then, we encoded the previous text, obtaining the features that represent each paste. Finally, we applied supervised learning to train a model with Pastebin samples and classify the online notepad content. We used two well-known features extractors ensembling with three different supervised classifiers, i.e. six different text classification pipelines. To build the classifier, we only worked with one pipeline, i.e. one encoding and classifier, filtering the classified pastes if they obtained an accuracy higher than a predefined threshold. To define what is the best approach to classifying the Pastebin content, we evaluated the six pipelines.

3.1 Text Preprocessing

Text preprocessing was used to delete no important characters before starting the text classification process. The text preprocessing *cleans* the text of words known as stop-words. We built a text preprocessing that removed all the format files, URL links, currency units, a single letter, long words, repeated text, numbers, words with numbers and special characters.

3.2 Feature Extraction

In this Section, we have reviewed two text coding techniques, Bag of Words (BOW) and Term Frequency-Inverse Document Frequency (TF-IDF). **Bag of Words** was proposed by Harris[8], is a form of simplified text representation that creates a list of vocabulary that is found within the corpus or body of the message. Once created, the frequency of each word in the text to be processed is measured and a vector with said data is obtained, using each word of the text as a characteristic to train the classifiers. Aizawa [1] presented **Term Frequency-Inverse Document Frequency** (TF-IDF), a method to assign weights for the dataset vocabulary statistically. It assigns a high weight value for words which occur frequently in a given text and with small frequency in all the other documents.

3.3 Classifiers

Our work uses three well-known supervised classifiers: Logistic Regression (LR), Support Vector Machines (SVM) and Naive Bayes (NB). **Logistic Regression**. LR is a supervised classification technique, which function was created by Cox[6]. It tries to obtain an accurate prediction for one or a set of variables by obtaining a logistic function. This function is the one that finds the probability that a certain variable presents the predicted characteristic. **Support Vector Machines**. SVM is a form of supervised classification firstly developed by Cortes and Vapnik[5], and it is based on maximizing the distance between the decision boundary and the data. SVM machines have a kernel function that defines how data will be split. **Naïve Bayes**. NB is a classifier based on the Bayes theorem, which was developed in the 18th century, and it applies the principle of independence assumption. It can be explained as a calculation of the probability of a document of belonging to a certain category, and then it assigns to this document the class with the best probability.

4 Pastes Content Classification

4.1 PasteCC_17K Building Procedure

To best of our knowledge, there is no labeled dataset related the content on the Pastebin web service. Hence, we created the first public Pastebin dataset. To build the dataset we followed the procedure used to generate a dataset with Darknet content by Wesam et al.[3]. We crawled pastes from Pastebin site via the website official API along with premium (PRO) license. In particular, we created a customized scraper to download pastes along with their meta-data. Our crawler downloaded $500K$ pastes from Pastebin, but we only labeled manually 804 samples into 14 categories as shown in Table 1. We refer to this dataset by Pastes Content Classification (PasteCC). PasteCC is publicly available on our web page[7].

[7] `http://gvis.unileon.es/dataset/paste-bin`

Among PasteCC samples, we found that the major category is **Source Code** which is expected, given that Pastebin is basically oriented for developers to share code snippets and logs. The category **Multimedia** contains pastes which share links to access videos, images or documents, and television hosting. **Encoded Text** category groups pastes whose content is related to sets of letters and numbers without semantic meaning. **Forum** category includes pastes in relation to any topic coming from a community as videogames or recommendations. **Personal** category has conversations or personal experiences. The pastes outside the previous categories belong to the category **Others**.

Interestingly, we noticed that suspicious activities are the minority. However, despite there being a minority, they might be dangerous and holding critical information that the authorities are interested in monitoring [9]. This fact makes the problem of building a supervised approach to detect and to classify suspicious activities even more challenging. Because of the lack of training examples hiders the capability of training a classifier to recognize them.

Table 1. Classification of the manually labelled pastes. Initial version of PasteCC dataset

Category	# Samples
Source Code	298
Multimedia	230
Forum	90
Logs	60
Others	33
Leaked-Data	16
Cryptocurrency	12
Encoded Text	12
Drugs	11
Child-pornography	11
Counterfeit Credit-Cards	10
Hacking	10
General-pornography	6
Personal	5
Total	**804**

To overcome this drawback, we incorporated an external dataset that contains activities similar to the ones we are looking for in Pastebin because they are samples of activities suspicious to be illegal. We made use of the Darknet Usage Text Addresses (DUTA-10K) dataset[8], which is the state-of-the-art dataset in terms of suspicious activities in the Tor Darknet and holds 26 classes with regular and suspicious categories found in 10367 unique onion domains in the Tor

[8] `http://gvis.unileon.es/dataset/duta-darknet-usage-text-addresses/`

Darknet. We hypothesize that if we joined DUTA-10K dataset with PasteCC dataset, we would have more samples per category and more classes as well. The resulting dataset is a union of *PasteCC* and *DUTA-10K*, and in total it holds 11171 samples distributed in a total of 28 categories.
After joining both dataset DUTA-10K and PasteCC, we trained models based on an ensemble of two encoding techniques, TF-IDF and BoW, together with three classifiers, namely SVM, LR, and Naive Bayes. We chose TF-IDF with LR since their overall results overcame the rest of classifiers. Therefore, we applied trained TF-IDF and LR classifier with PasteCC + DUTA-10K samples on 100000 unlabelled pastes extracted by our search engine feature from Pastebin. The classifier predicted the classes of pastes above three probability-of-success thresholds, which are 85%, 90% and, 95%. We checked 100 pastes per threshold in order to determine the achievement of the classifier. The pastes predicted by the threshold based on the 85% probability of success had many classification mistakes and the pastes obtained by the thresholds with success probability of 95% despite their correct prediction, the number of contributed pastes was scant. We discarded the pastes classified by both thresholds. We selected the pastes predicted with the 90% probability of success due to the correct pastes prediction was similar to 95% threshold and contributed with many more number of pastes. Finally, we removed DUTA dataset examples in order to keep only the text content provided by Pastebin, the resulting dataset was called PasteCC_17K. Table 2 shows the number of samples per category.

4.2 Statistics

PasteCC_17K comprises 15 classes, including 17640 samples taken from Pastebin, the most of the samples belongs to "Source Code" category, 70.27%, the next most influential class is "Multimedia", 26.75%. "Counterfeit Personal-Identification" is a class imported from DUTA and represents the categories with fewer examples. About suspicious content, there are 217 suspect pastes i.e. 1.23% of pastes. To the best of our knowledge and the extensions of our current evaluation, we can summarize that 98.7% of the Pastebin content could be considered safe. However, although unlawful content represents a small percentage, since 481,000 of pastes are uploaded per day, it results in a significant amount of suspicious content that should be considered, since it is publicly available and visible for the whole Web. A text classifier capable of finding these suspicious content would represent a useful tool for monitoring Pastebin.

5 Experimentation

5.1 Experimental Settings

We used Python 3 with PyCharm IDE. To build the text vectorizers and the classifiers, we used Scikit-learn library[9], Numpy and Pandas. For classification,

[9] Machine Learning library for Python. (Source: http://scikit-learn.org/stable)

Fig. 3. Example of illicit categories examples provided to PasteCC_17K, Child-pornography category

Table 2. Influence of the categories in PasteCC_17K. The categories of Counterfeit Credit-Cards and Counterfeit Personal-Identification are named respectively as Counterfeit C-C and Counterfeit P-I.

Category type	Category	# Samples	Percent(%)
Suspicious 1.23%; 217 Samples			
	Hacking	86	0,49
	Counterfeit C-C	67	0,38
	Child-pornography	29	0,16
	Leaked-Data	19	0,11
	Drugs	14	0,08
	Counterfeit P-I	2	0,01
Normal 98.70%; 17423 Samples			
	Source Code	12396	70,27
	Multimedia	4719	26,75
	Forum	90	0,51
	Others	81	0,46
	Logs	65	0,37
	Encoded Text	25	0,14
	Crytocurrency	20	0,11
	General-pornography	17	0,10
	Personal	10	0,06
Total		**17640**	

we used 70% of the samples as a training set and the remaining 30% as a set of testing, following the work of Al Nabki et al.[3][2].

5.2 Results

Table 3 shows the results of the different models. Given the low number of samples and the unbalanced classes, the learning curve of the models created presented overfitting.

Table 3. Results of training classifier with PasteCC_17K (%)

		TF-IDF LR	TF-IDF SVM	TF-IDF NB	BOW LR	BOW SVM	BOW NB
Accuracy		**98.63**	97.80	97.52	98.37	97.62	98.05
Precision	Macro	71.60	72.50	21.10	58.80	62.90	41.70
	Micro	98.60	97.80	97.50	98.40	97.60	98.10
	Weighted	98.70	98.80	95.40	98.70	98.70	97.50
Recall	Macro	72.10	63.20	21.00	63.20	59.90	38.10
	Micro	98.60	97.80	97.50	98.40	97.60	98.10
	Weighted	98.60	97.80	97.50	98.40	97.60	98.10
F1 Score	Macro	**72.10**	62.20	21.10	58.70	58.30	35.64
	Micro	**98.70**	97.80	97.50	98.40	97.60	98.10
	Weighted	**98.70**	98.10	97.50	98.50	98.00	97.70

5.3 Discussion

Due to we are working with an unbalanced multiclass dataset where two categories, Source Code and Multimedia, represent 97.02% of the total, we present the precision, recall, and F1 score for each ensemble. The model with the best performance is the LR together with TF-IDF as feature extractor, yielding a cross-validation accuracy of 98.63% and micro F1 Score of 98.60%. The ensemble of TF-IDF with LR overcomes the rest of ensembles in overall results. The results show that NB obtains higher accuracy with BOW than TF-IDF, which only occurs with NB classifier. The SVM classifier was less influenced by feature extractors due to the linear kernel function of SVM we used that mapped the data in order to be able to separate linearly. All classifier achieved low performance when the evaluation does not account imbalanced dataset as *Macro* measures show with both feature extraction, NB classifier presents the lowest performance.

6 Conclusions and future works

In this work, we proposed a solution to solve the problem of classifying Pastebin the content, with the aim of identifying suspicious content. We first crawled and created labeled dataset in a semi-automatic way, Pastes Content Classification 17K (PasteCC_17K), which comprises 17640 pastes grouped in 15 categories, being six of them suspicious to contain illicit content that represents the 1.23% of the total classified pastes. The category that shows the habitual use of Pastebin, sharing source code, constitutes the $70,27\%$ of our dataset. The small percentage of suspicious content over the total can be misleading since applied over the daily Pastebin pastes would imply a significant amount of illegal and publicly available content that ought to be considered. We have found pastes with leaked data, hacking offers, and even child pornography links which should be monitored. We evaluated the ensemble of two encoding techniques, TF-IDF and BoW, together with three well-known classifiers, namely SVM, LR and Naive Bayes. We found that the ensemble of TF-IDF together with LR achieves the best performance in PasteCC_17K, with an accuracy of 98.90%.

The obtained results encourage us to extend the PasteCC_17K dataset using Active Learning techniques considering the work of Hu et al.[10], to find new patterns in Pastebin. Moreover, given the variety of the categories in Pastebin, we are planning to use different text encoding techniques for each category. In future works, we will explore the use of Active Learning techniques to monitor Pastebin, trying to decrease the labelling cost of unknown samples while the classifier accuracy increases.

7 Acknowledgements

This research is supported by the INCIBE grant "INCIBEI-2015-27359", corresponding to the "Ayudas para la Excelencia de los Equipos de Investigación avanzada en ciberseguridad" and also by the framework agreement between the University of León and INCIBE (Spanish National Cybersecurity Institute) under Addendum 22 and 01.

References

1. Aizawa, A.: An information-theoretic perspective of tf–idf measures. Information Processing & Management 39(1), 45–65 (2003)
2. Al-Nabki, M.W., Fidalgo, E., Alegre, E., Fernández-Robles, L.: Torank: Identifying the most influential suspicious domains in the tor network. Expert Systems with Applications 123, 212 – 226 (2019)
3. Al Nabki, M.W., Fidalgo, E., Alegre, E., de Paz Centeno, I.: Classifying illegal activities on tor network based on web textual contents. In: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics. Association for Computational Linguistics, Association for Computational Linguistics, Valencia, Spain (04/2017 2017)

4. Bui, D.D.A., Fiol, G.D., Jonnalagadda, S.: Pdf text classification to leverage information extraction from publication reports. Journal of Biomedical Informatics 61, 141 – 148 (2016)
5. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning 20(3), 273–297 (Sep 1995)
6. Cox, D.R.: The regression analysis of binary sequences. J Roy Stat Soc B 20, 215–242 (1958)
7. Diab, D.M., Hindi, K.: Using differential evolution for fine tuning naïve bayesian classifiers and its application for text classification. Applied Soft Computing 54 (12 2016)
8. Harris, Z.S.: Distributional structure. Word 10(2-3), 146–162 (1954)
9. Herath, H.: Web information extraction system to sense information leakage. Master's thesis, University of Moratuwa, Sri Lanka (2003)
10. Hu, R., Delany, S.J., Mac Namee, B.: Egal: Exploration guided active learning for tcbr. In: International Conference on Case-Based Reasoning. pp. 156–170. Springer (2010)
11. Joachims, T.: Text categorization with support vector machines: Learning with many relevant features. In: Machine Learning: ECML-98. pp. 137–142. Springer Berlin Heidelberg (1998)
12. Joulin, A., Grave, E., Bojanowski, P., Mikolov, T.: Bag of tricks for efficient text classification. CoRR abs/1607.01759 (2016)
13. Lochter, J.V., Zanetti, R.F., Reller, D., Almeida, T.A.: Short text opinion detection using ensemble of classifiers and semantic indexing. Expert Systems with Applications 62, 243 – 249 (2016)
14. Matic, S., Fattori, A., Bruschi, D., Cavallaro, L.: Peering into the muddy waters of pastebin. ERCIM News 2012(90) (2012)
15. Meng, R., Zhao, S., Han, S., He, D., Brusilovsky, P., Chi, Y.: Deep keyphrase generation. CoRR abs/1704.06879 (2017)
16. Mironczuk, M., Protasiewicz, J.: A recent overview of the state-of-the-art elements of text classification. Expert Systems with Applications 106 (03 2018)
17. Panchenko, A., Ruppert, E., Faralli, S., Ponzetto, S.P., Biemann, C.: Building a web-scale dependency-parsed corpus from commoncrawl. CoRR abs/1710.01779 (2017)
18. Perlroth, N.: Hackers breach 53 universities and dump thousands of personal records online. New York Times, New York (2012)
19. Sebastiani, F.: Machine learning in automated text categorization. ACM Comput. Surv. 34(1), 1–47 (Mar 2002)
20. Silva, R.M., Almeida, T.A., Yamakami, A.: Mdltext: An efficient and lightweight text classifier. Knowledge-Based Systems 118, 152 – 164 (2017)
21. Stein, R.A., Jaques, P.A., Valiati, J.F.: An analysis of hierarchical text classification using word embeddings. CoRR abs/1809.01771 (2018)
22. Wu, L., Fisch, A., Chopra, S., Adams, K., Bordes, A., Weston, J.: Starspace: Embed all the things! CoRR abs/1709.03856 (2017)
23. Zhang, Q., Wang, Y., Gong, Y., Huang, X.: Keyphrase extraction using deep recurrent neural networks on twitter. In: EMNLP (2016)
24. Zhang, X., Zhao, J., LeCun, Y.: Character-level convolutional networks for text classification. In: Advances in Neural Information Processing Systems. vol. 2015-January, pp. 649–657. Neural information processing systems foundation (2015)
25. Zhu, D., Wong, K.W.: An evaluation study on text categorization using automatically generated labeled dataset. Neurocomputing 249, 321 – 336 (2017)