File Name Classification Approach to Identify Child Sexual Abuse

Mhd Wesam Al-Nabki [1,2][a], Eduardo Fidalgo[1,2][b], Enrique Alegre[1,2][c] and Rocío Aláiz-Rodríguez[1,2][d]

[1]*Department of Electrical, Systems and Automation, Universidad de León, Spain*

[2]*Researcher at INCIBE (Spanish National Cybersecurity Institute), León, Spain*

{*mnab, efidf, ealeg, ralar*}*@unileon.es*

Keywords: Short Text Classification, File Name Classification, Active Learning, Character-level Convolutional Networks, Child Sexual Abuse.

Abstract: When Law Enforcement Agencies seize a computer machine from a potential producer or consumer of Child Sexual Exploitation Material (CSEM), they need accurate and time-efficient tools to analyze its files. However, classifying and detecting CSEM by manual inspection is a high time-consuming task, and most of the time, it is unfeasible in the amount of time available for Spanish police using a search warrant. An option for identifying CSEM is to analyze the names of the files stored in the hard disk of the suspect person, looking in the text for patterns related to CSEM. However, due to the particularity of this file names, mainly its length and the use of obfuscated words, current file name classification methods suffer from a low recall rate, which is essential in the context of this problem. This paper presents our ongoing research to identify CSEM through their file names. We evaluate two approaches of short text classification: a proposal based on machine learning classifiers exploring the use of Logistic Regression and Support Vector Machine and an approach using deep learning by adapting two popular Convolutional Neural Network (CNN) models that work on character-level. The presented CNN achieved an average class recall of 0.86 and a recall rate of 0.78 for the CSEM class. The CNN based classifier could be integrated into forensic tools and services that might support Law Enforcement Agencies to identify CSEM without the need to access systematically to the visual content of every file.

1 INTRODUCTION

Child Sexual Exploitation Material (CSEM) is defined as sexual abuse of a person under 18 years old, together with producing images or videos of the abuse and sharing the content online (Europol, 2019a). One of the features of some Darknet networks, such as The Onion Router (Tor)[1] or FreeNet[2] (Al-Nabki et al., 2017; Al-Nabki et al., 2019) and also Peer to Peer (P2P) networks, like eDonkey, (Panchenko et al., 2012; Peersman et al., 2016) is their capability of preserving a high level of privacy and anonymity of their users. This characteristic allows pedophiles to easily share CSEM far away from Law Enforcement Agencies (LEAs) monitoring. Therefore, in 2017, the Council of the European Union (EU) has prioritized cybercrimes related to Child Sexual Abuse (CSA) to be the most serious crime between the years 2018 and 2021 (Europol, 2019b).

CSEM producers and consumers might save this content on their local computer machines, at least temporary. When an LEA enters a home to inspect a computer of a suspect, a police agent reviews the files contained in the investigated hard drive, trying to determine whether or not the suspected of pedophilia has stored CSEM in the computer (Gangwar et al., 2017). This process needs to be accomplished in a limited time and as accurately as possible (Chaves et al., 2019).

In this paper, we present our ongoing research based on Natural Language Processing to identify CSEM. More specifically, we are designing a text classifier to decide whether a given file is CSEM or not, according to its name. The same process, based on the classification of the content of the files, whether images or videos, is out of the scope of this paper and research.

Building a supervised text classifier for CSEM

[a] https://orcid.org/0000-0002-3975-3478
[b] https://orcid.org/0000-0003-1202-5232
[c] https://orcid.org/0000-0003-2081-774X
[d] https://orcid.org/00000-0003-4164-5887

[1] https://www.torproject.org/

[2] https://freenetproject.org/

identification is a challenging task due to several reasons. First, a binary supervised algorithm requires training samples of Non-CSEM and CSEM files. However, there are no publicly available datasets of the latter class, and crawling samples from a P2P network or the Darknet is illegal (García-Olalla et al., 2018). Therefore, only CSEM file names obtained legally, i.e., provided by LEAs, could be used.

Second, a file name typically is a text of small length, which leads to a sparse representation of the samples because we have a massive number of features, while an instance is only represented with a few of them. Finally, CSEM producers or consumers tend to invent a personalized file name style to create their own vocabulary, abbreviations, and acronyms to circumvent detection tools, using a personalized obfuscated style of writing. For example, in a sample named "*!!!!yoB0yXX*", the exclamation marks refer to the age of a boy, and the letter *O* is replaced by the number zero. Hence, most likely, this file could be related to the abuse of a four years old boy.

In this paper, we present an approach to address the aforementioned challenges by designing a supervised File Name Classification (FNC) system. We explore two different text classification approaches: 1) conventional machine learning (ML) pipeline and 2) deep learning (DL) approach using a Convolutional Neural Network (CNN). We evaluate the classifiers on a custom dataset of $65,351$ samples.

The rest of the paper is organized as follows: Section 2 presents the related work. After that, Section 3 describes the designed classification pipelines. Next, Section 4 explains how the dataset is created and what are its main features. Then, in Section 5, we describe the experimentation performed and discuss the results obtained. Finally, Section 6 presents the conclusions by pointing to our ongoing and future research.

2 RELATED WORK

The problem of short text classification is widely studied under different research lines (Sriram et al., 2010; Sun, 2012; Shang et al., 2013; Rana et al., 2014; Lee and Dernoncourt, 2016; Alsmadi and Hoon, 2018). In the following, we explore two text classification problems that are close to File Name Classification (FNC).

First, news headlines classification task attempts to group news articles based on their titles, in which the title is made up of a few words. (Rana et al., 2014) proposed a pipeline for news headlines classification that consists of three stages: data pre-processing, text representation, and classification. In the *data pre-processing* step, the text was tokenized into words, and spaces replaced special characters, stop words were removed, and the text was stemmed. For the *text representation*, the authors used Term Frequency-Inverse Document Frequency (TF-IDF), Information Gain (IG) (Shang et al., 2013), and Boolean Weight (BW) (Chouchoulas and Shen, 1999). Finally, in the *classification* stage, Rana et al. explored Naive Bayes (NB) (Kim et al., 2006), Support Vector Machine (SVM) (Joachims, 1998), K-Nearest Neighbor (KNN) (Hotho et al., 2005), and Decision Trees (DT) (Safavian and Landgrebe, 1991). However, the core difference between our problem and news headlines classification is that the latter has high-quality input text, where the punctuation marks are maintained correctly, and there are no misspelled words.

The second research line is classifying tweets from Twitter. According to (Perez, 2019), the most common length of a tweet is 33 characters, while the maximum number of characters is 280. Despite the length of the text, Perez (Perez, 2019) work also deals with the low-quality input text as in our work. Thus, both might use abbreviations to save space or misspell some words. (Imran et al., 2016) pre-processed the tweets by removing hyperlinks, mentions and stop words, and then they used the N-grams and IG techniques for feature extraction and Random Forest classifier (Breiman, 2001). (Alsmadi and Hoon, 2018) addressed short noisy text classification on Twitter. They proposed a supervised word weighting schema to highlight essential terms in short noisy text along with an SVM classifier.

(Chen et al., 2018) proposed a framework to identify cyberbullying on Twitter. For text representation, they compared pre-trained language models, like Word2Vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014), with traditional text encoding techniques, such as TF-IDF, and they realized a decline in the performance when embedding-based were used. For classification, they compared traditional ML classifiers like Logistic Regression (LR) and SVM with DL classifiers, like Long Short-Term Memory (LSTM) (Lee and Dernoncourt, 2016) and Convolutional Neural Network (CNN) (Zhang et al., 2015).

Few researchers employed file names classification approach to recognize child sexual activities. (Panchenko et al., 2012) attempted to normalize file names using short message service (SMS) normalization techniques proposed by (Beaufort et al., 2010). On the top of the normalized text, they trained an SVM classifier and obtained an accuracy of 96.97% on their dataset. The study of (Peersman et al., 2014) proposed a framework called iCOP to detect CSEM

activities on P2P networks is proposed. The first stage of their classification pipeline was a dictionary-based filter that was constructed manually and held CSEM keywords. Also, they used character n-gram of size two, three, and four, to capture more features about the file name. These features were used to train a binary SVM classifier. Afterward, in their recent work (Peersman et al., 2016), they used the same representation but benchmarked more classifiers, like SVM and NB. They evaluated their proposal on a custom dataset, and they observed that the SVM classifier could identify CSEM file names with a recall rate of 0.43.

3 METHODOLOGY

In this section, we present the methodology used to build the proposed file name classifier (FNC). In the following, we explore two distinct approaches based on machine learning and deep learning.

The pre-processing step is common to both approaches, where special characters and numbers are replaced by *#* and *$*, respectively, reducing the sparsity of the features. For example, the input text "*!!!!yoB0yXX*" will be encoded into "*####yoB$yXX*". Another benefit of this representation is the removal of the duplicated instances that differ only by their names. For example, a folder in a seized computer could have more than 100 images named IMG01.png, IMG02.png, ... IMG100.png, whereas all these names are repeating the same information of IMG##.png.

3.1 Machine Learning approach

The typical design of a supervised text classifier has two main components: text representation to encode the input samples into feature vectors and a classifier that separates these feature vectors into CSEM and Non-CSEM examples.

A crucial step in a text classification pipeline is finding an adequate representation of the text. Tokenizing a file name on word-level may not be the most efficient solution because the text of a file name might be joined as a single word or using a special character. Therefore, we tokenize the text at the character level, following the work of other researchers (Peersman et al., 2014; Peersman et al., 2016).

The N-gram technique extracts all the patterns of two to five consecutive characters of a given file name, which form a set of tokens. In this work, we combine the widely used Term Frequency-Inverse Document Frequency (TF-IDF) technique (Aizawa, 2003) with N-gram. TF-IDF is a statistical model that assigns weights to file name tokens. It accentuates those whose frequency is higher in a given file name and, at the same time, de-emphasizes tokens that frequently occur in many files. This way, it overcomes the issue of misspelled words or personalized naming style in file names. Table 1 shows an example of two to five grams of a file name "*####yoB$yXX*". Furthermore, to discard noisy tokens, we set thresholds for the minimum and the maximum term frequency.

Table 1: Example of preprocessing and tokenizing a file name with two, three, four, and five grams.

Original File Name	!!!!yoB0yXX
Preprocessing	####yoB$yXX
2-grams	##, ##, ##, #y, yo, oB, B$, $y, yX, XX
3-grams	###, ###, ##y, #yo, yoB, oB$, B$y, $yX, yXX
4-grams	####, ###y, ##yo, #yoB, yoB$, oB$y, B$yX, $yXX
5-grams	####y, ###yo, ##yoB, #yoB$, yoB$y, oByX, ByXX

For this work, we select two commonly used supervised classifiers that have shown good performance on text classification tasks (Peersman et al., 2016; Al-Nabki et al., 2017). They are Support Vector Machine (SVM) (Schohn and Cohn, 2000) and Logistic Regression (LR) (Genkin et al., 2007).

3.2 Deep Learning approach based on Convolutional Neural Networks

Convolutional Neural Networks (CNN) have been widely and successfully used for image classification (Fidalgo et al., 2018; Fidalgo Fernández et al., 2019). Besides the machine learning approaches mentioned above, we employed CNN to classify file names. We adapted the models of (Zhang et al., 2015) and (Kim et al., 2016), as they showed promising performance on Natural Language Processing (NLP) tasks.

The model of Kim et al. was intended for the language modeling task and uses only character-level inputs but the prediction is performed at the word-level. However, we adapt it to a text classification problem by replacing the subsequent recurrent layers with a dense layer to perform softmax over the classes. Similarly, the model of Zhang et. al applied only on characters and does not require acquiring knowledge about the syntactic or semantic structure of the addressed language. Unlike Kim's model, this one is dedicated to text classification tasks.

4 DATASET CONSTRUCTION

Building a supervised file name classifier requires collecting samples of both classes, CSEM and Non-

CSEM. For the Non-CSEM class, we refer to a dataset published by the National Software Reference Library (NSRL)[3]. We were able to access more than 32 million file names, but this will make our problem very imbalanced and skewed to the Non-CSE class. Therefore, we selected an initial subset of $64,000$ Non-CSEM examples, resulting in $56,021$ after applying the pre-processing step.

Regarding the CSEM class, we were able to collect these examples thanks to the collaboration between the Spanish National Cybersecurity Institute (INCIBE)[4] and Spanish LEAs. This latter provided us a list with dumps of hard disks that had been seized from criminals' computers. The list had $64,857$ CSEM samples. However, after pre-processing them, the number decreased to $9,330$ unique instances.

Table 2: Description of the used dataset to train the file names classifier

File Name Class	CSEM	Non-CSEM	Total
Before preprocessing	64,857	64,000	128,857
After preprocessing	9,330	56,021	65,351

5 EMPIRICAL EVALUATION

5.1 Experimental Setting

The experiments were conducted on a PC with an Intel(R) Core(TM) i7 processor with 32 GB of RAM under Windows-10. We used Python3 with Keras framework[5] for the CNN implementation and Scikit-Learn[6] for the machine learning classifiers.

Regarding the configuration of the ML approach, we used character n-grams, extracting patterns from two to five grams. Also, we set thresholds for the minimum and the maximum gram proportion to 0.995 and 0.001, respectively. For the LR, we set the parameter C to 10, empirically, which refers to the inverse of regularization strength. Concerning the SVM classifier, we used a linear kernel, and we left all the other parameters to their default values, as set by the Scikit-Learn library.

Concerning the neural network model of (Zhang et al., 2015), we used the same model hyperparameters as described in their paper, except for the input feature-length. We set it to 128 instead of 1014 because the longest file name we had was 125 characters. Likewise, for the CNN model of (Kim et al., 2016), we left the same hyperparameters as described in their paper. The only change was made to the output layer when we replaced the recurrent layers with a dense layer with softmax function.

In order to estimate the model performance, we used 70% of the samples to train the model, and 30% to test the models. The validation set forms 10% of the training set. Table 3 describes the dataset used to train and test the model.

Table 3: A description of the dataset used to train, validated, and test the file name classifier

File Name Class	Train	Validation	Test
Number of Samples	36,548	9,137	19,666

To control the number of iterations while training the neural network, we set an early stopping criterion, which is triggered if there is no further improvement on the validation set during 10 epochs of training.

5.2 Evaluation Metric

The principal objective of this work is to assist LEAs in the detection of CSEM through their file names, avoiding the exposure of an agent to CSEM. In this context, it is desirable to have a low number of false negatives - a file named with CSEM content identified as a Non-CSEM - than a low number of false positives, i.e., Non-CSEM file name wrongly categorized as a CSEM. Therefore, we need to pay more attention to the recall of the CSEM class rather than the Non-CSEM class.

Recall metric for a class is calculated as the total number of samples correctly classified for that class (the True Positives *TP*), over the total number of samples of that class (the True Positives *TP* and the False Negatives *FN*). Equation (1) shows how Recall is estimated for a given class.

$$Recall = \frac{TP}{TP+FN}. \quad (1)$$

Nevertheless, the *precision* of a classifier is also a crucial factor in measuring its performance, as it shows the proportion of samples that were correctly identified. Class precision is calculated as a ratio of correctly classified file names of that class (the True Positives *TP*) to the total number of predicted positive samples of that class (the True Positives *TP* and the False Positives *FP*), and it is given in Equation (2).

$$Precision = \frac{TP}{TP+FP}. \quad (2)$$

[3] https://www.nist.gov/software-quality-group/about-nsrl/nsrl-introduction

[4] In Spanish, it stands for the Instituto Nacional de Ciberseguridad de España

[5] https://github.com/fchollet/keras

[6] https://scikit-learn.org/stable/

Finally, the F1 score of a class summarizes the two before mentioned metrics as it refers to the harmonic mean of the precision and recall and it is calculated following to Equation (3).

$$F1 = \frac{2 * (Recall * Precision)}{(Recall + Precision)}. \quad (3)$$

Additionally, it has been proved that the accuracy metric is not reliable when the dataset is imbalanced (Chen et al., 2017), as in our case, where the majority of the samples are Non-CSEM file names. An alternative metric is to use *average class accuracy*, which refers to the average recall of the CSEM and the Non-CSEM classes, rather than using overall dataset level accuracy.

5.3 Empirical Results

In this section, we compared the performance of the four pipelines described in Section 3 in the dataset of CSEM. Table 4 shows that both CNN models outweigh the machine learning classifier. Furthermore, by comparing the average class recall of both CNN models, we observe that the model of Zhang et al. has obtained an average class recall of 0.86, while the model of Kim et al. has scored 0.84. Also, by comparing the performance on class level, we can see that the model of Zhang et al. has the best recall rate for the CSEM class of 0.78, while the model of n-grams with SVM classifier has the best recall rate for the class Non-CSEM of 0.98. It is noteworthy that the SVM classifier obtained the lowest performance in terms of the F1 score and the average class recall metrics.

Table 4: A comparison between four algorithms to classify file names. The values in bold refer to the best result obtained

Method Name	Category	Precision	Recall	F1
N-gram LR	Non-CSEM	0.95	0.97	0.96
	CSEM	0.78	0.65	0.71
	Average	0.86	0.81	0.84
N-gram SVM	Non-CSEM	0.94	0.98	0.96
	CSEM	0.81	0.61	0.70
	Average	0.88	0.79	0.83
Zhang et al. model	Non-CSEM	0.96	0.95	0.96
	CSEM	0.71	0.78	0.74
	Average	**0.84**	**0.86**	**0.85**
Kim et al. model	Non-CSEM	0.95	0.97	0.96
	CSEM	0.78	0.70	0.74
	Average	0.86	0.84	0.85

In addition to the analyzed measures, the required time to predict the legality of a file name plays a vital role when police forces investigate a hard disk of a suspect. Therefore, we compare the aforementioned classification methods in terms of the time needed to predict the category of one million file names. We carried out this experiment on a consumer-level computer with 16GB RAM, an Intel Core i7 processor and a GPU Nvidia GeForce GTX 1060. Regarding the CNN-based methods, we set the batch size to $10,000$ examples and we tried to predict the file names twice, once using the GPU and another using the CPU. While for the machine learning methods, we examined them on the CPU only. Table 5 shows that the logistic regression classifier along with char n-gram for text representation achieves the highest prediction speed using the CPU. Meanwhile, the CNN-based models were enormously time-consuming. However, when the GPU is used, the model of (Zhang et al., 2015) surpasses the other proposals in terms of the prediction time of 59 seconds.

Table 5: A comparison between four algorithms in terms of the required time (in seconds) to predict one million file names using a GPU or a CPU processor. The values in bold refer to the best prediction time

Method Name	**CPU**	**GPU**
N-gram LR	**106**	-
N-gram SVM	852	-
Zhang et al. model	1022	**59**
Kim et al. model	1172	62

6 CONCLUSIONS AND FUTURE WORK

In this paper, we explored two machine learning classifiers (Support Vector Machine and Logistic Regression), versus two Convolutional Neural Network (CNN) models to classify file names related to Child Sexual Exploitation Material (CSEM). We compared these models on a dataset of $65,351$ samples distributed over two classes ($64,857$ Non-CSEM and $9,330$ CSEM file names) concerning the average class recall metric.

Our results strengthen the superiority of CNN over regular machine learning classifiers to categorize file names. Incorporating the CNN model by Zhang et al., we were able to identify CSEM file names with a recall rate of 0.78, and an average class recall of 0.86. Also, we demonstrated how the adapted CNN model outperforms the model by Kim et al. (Kim et al., 2016) and two supervised machine learning classifiers (Support Vector Machine and Logistic Regression).

The presented results were conducted on a dataset extracted from the file names only. However, there is still further work to be investigated in the future during our ongoing research. In particular, we are

planning to include orthographic features extracted from file names, as it showed promising performance on other NLP tasks, such as Named Entity Recognition (NER) (Aguilar et al., 2017). Furthermore, the presented work did not investigate paths of the files, which could be pivotal evidence when the file name is meaningless, such as a file with a name made up of numbers or random characters (e.g., *kf3kfk3985.png*). Also, the metadata of the file, such as its header, size, extension, could provide further clues to predict its class correctly.

The assessment of transformer-based models, such as BERT (Luo et al., 2018), RoBERTa (Liu et al., 2019), and XLNet (Yang et al., 2019) for text classification is part of our immediate future research, as they have shown promising results on various NLP tasks.

ACKNOWLEDGEMENTS

This research has been funded with support from the European Commission under the 4NSEEK project with Grant Agreement 821966. This publication reflects the views only of the author, and the European Commission cannot be held responsible for any use which may be made of the information contained therein.

REFERENCES

Aguilar, G., Maharjan, S., Monroy, A. P. L., and Solorio, T. (2017). A multi-task approach for named entity recognition in social media data. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 148–153.

Aizawa, A. (2003). An information-theoretic perspective of tf–idf measures. *Information Processing & Management*, 39(1):45–65.

Al-Nabki, M. W., Fidalgo, E., Alegre, E., and de Paz, I. (2017). 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*, volume 1, pages 35–43.

Al-Nabki, M. W., Fidalgo, E., Alegre, E., and Fernández-Robles, L. (2019). Torank: Identifying the most influential suspicious domains in the tor network. *Expert Systems with Applications*, 123:212–226.

Alsmadi, I. and Hoon, G. K. (2018). Term weighting scheme for short-text classification: Twitter corpuses. *Neural Computing and Applications*, pages 1–13.

Beaufort, R., Roekhaut, S., Cougnon, L.-A., and Fairon, C. (2010). A hybrid rule/model-based finite-state framework for normalizing sms messages. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 770–779. Association for Computational Linguistics.

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

Chaves, D., Fidalgo, E., Alegre, E., and Blanco, P. (2019). Improving speed-accuracy trade-off in face detectors for forensic tools by image resizing. In *V Jornadas Nacionales de Investigación en Ciberseguridad (JNIC)*, pages 1–2.

Chen, H., Mckeever, S., and Delany, S. J. (2017). Harnessing the power of text mining for the detection of abusive content in social media. In *Advances in Computational Intelligence Systems*, pages 187–205. Springer.

Chen, J., Yan, S., and Wong, K.-C. (2018). Verbal aggression detection on twitter comments: convolutional neural network for short-text sentiment analysis. *Neural Computing and Applications*.

Chouchoulas, A. and Shen, Q. (1999). A rough set-based approach to text classification. In *International Workshop on Rough Sets, Fuzzy Sets, Data Mining, and Granular-Soft Computing*, pages 118–127. Springer.

Europol (2019a). Child sexual exploitation. `https://www.europol.europa.eu/crime-areas-and-trends/crime-areas/child-sexual-exploitation`. Accessed: 2019-11-08.

Europol (2019b). Eu policy cycle - empact. `https://www.europol.europa.eu/crime-areas-and-trends/eu-policy-cycle-empact`. Accessed: 2019-11-08.

Fidalgo, E., Alegre, E., González-Castro, V., and Fernández-Robles, L. (2018). Boosting image classification through semantic attention filtering strategies. *Pattern Recognition Letters*, 112:176–183.

Fidalgo Fernández, E., Alegre Gutiérrez, E., Fernández Robles, L., and González Castro, V. (2019). Fusión temprana de descriptores extraídos de mapas de prominencia multi-nivel para clasificar imágenes. *Revista Iberoamericana de Automática e Informática industrial*, 0(0).

Gangwar, A., Fidalgo, E., Alegre, E., and González-Castro, V. (2017). Pornography and child sexual abuse detection in image and video: A comparative evaluation. In *8th International Conference on Imaging for Crime Detection and Prevention (ICDP)*, pages 37–42.

García-Olalla, O., Alegre, E., Fernández-Robles, L., Fidalgo, E., and Saikia, S. (2018). Textile retrieval based on image content from CDC and webcam cameras in indoor environments. *Sensors (Switzerland)*, 18(5).

Genkin, A., Lewis, D. D., and Madigan, D. (2007). Large-scale bayesian logistic regression for text categorization. *Technometrics*, 49(3):291–304.

Hotho, A., Nürnberger, A., and Paaß, G. (2005). A brief survey of text mining. In *Ldv Forum*, pages 19–62. Citeseer.

Imran, M., Mitra, P., and Srivastava, J. (2016). Cross-language domain adaptation for classifying crisis-related short messages. In *ISCRAM 2016 Conference Proceedings - 13th International Conference on Information Systems for Crisis Response and Management*. Information Systems for Crisis Response and Management, ISCRAM.

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. In Nédellec, C. and Rouveirol, C., editors, *Machine Learning: ECML-98*, pages 137–142, Berlin, Heidelberg. Springer Berlin Heidelberg.

Kim, S.-B., Han, K.-S., Rim, H.-C., and Myaeng, S. H. (2006). Some effective techniques for naive bayes text classification. *IEEE transactions on knowledge and data engineering*, 18(11):1457–1466.

Kim, Y., Jernite, Y., Sontag, D., and Rush, A. M. (2016). Character-aware neural language models. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Lee, J. Y. and Dernoncourt, F. (2016). Sequential short-text classification with recurrent and convolutional neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 515–520, San Diego, California. Association for Computational Linguistics.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Luo, J., Zhou, W., and Du, Y. (2018). An active learning based on uncertainty and density method for positive and unlabeled data. In Vaidya, J. and Li, J., editors, *Algorithms and Architectures for Parallel Processing*, pages 229–241, Cham. Springer International Publishing.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Panchenko, A., Beaufort, R., and Fairon, C. (2012). Detection of child sexual abuse media on p2p networks: Normalization and classification of associated filenames. In *Proceedings of the LREC Workshop on Language Resources for Public Security Applications*, pages 27–31.

Peersman, C., Schulze, C., Rashid, A., Brennan, M., and Fischer, C. (2014). icop: Automatically identifying new child abuse media in p2p networks. In *2014 IEEE Security and Privacy Workshops*, pages 124–131. IEEE.

Peersman, C., Schulze, C., Rashid, A., Brennan, M., and Fischer, C. (2016). icop: Live forensics to reveal previously unknown criminal media on p2p networks. *Digital Investigation*, 18:50–64.

Pennington, J., Socher, R., and Manning, C. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Perez, S. (2019). Twitter's doubling of character count from 140 to 280 had little impact on length of tweets – techcrunch.

Rana, M. I., Khalid, S., and Akbar, M. U. (2014). News classification based on their headlines: A review. In *17th IEEE International Multi Topic Conference 2014*, pages 211–216. IEEE.

Safavian, S. R. and Landgrebe, D. (1991). A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics*, 21(3):660–674.

Schohn, G. and Cohn, D. (2000). Less is more: Active learning with support vector machines. In *ICML*, page 6. Citeseer.

Shang, C., Li, M., Feng, S., Jiang, Q., and Fan, J. (2013). Feature selection via maximizing global information gain for text classification. *Knowledge-Based Systems*, 54:298 – 309.

Sriram, B., Fuhry, D., Demir, E., Ferhatosmanoglu, H., and Demirbas, M. (2010). Short text classification in twitter to improve information filtering. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 841–842. ACM.

Sun, A. (2012). Short text classification using very few words. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 1145–1146. ACM.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J. G., Salakhutdinov, R., and Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237.

Zhang, X., Zhao, J., and LeCun, Y. (2015). Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657.