# Tuning Pre-Trained Neural Text Classifiers to Identify Ransomware Notes

Samuel Miller, University of Maryland – College Park

## Abstract

The ability to identify files on a system that indicate malicious activity is an important technique to maintain system security. Research in this area has focused on the code in the malicious files themselves, but the human language text files that come with certain types of malware (i.e. ransomware) are often overlooked. In this paper, we present a novel method for ransomware note classification, applying natural language processing techniques to the study of malicious activity detection. We present our results which show strong performance by tuning a pre-trained sequence classifier on ransomware notes and other types of benign files, demonstrating accurate identification of ransomware note text.

## 1. Introduction

Ransomware threats are on the rise and constitute a serious and damaging threat to global balance; Infosecurity[1] reports that by 2023, ransomware damages are expected to exceed $30 billion, with devastating impacts on cities (e.g. Antwerp in Dec 2022), organizations (e.g. the Metropolitan Opera where over $200K was lost in Dec 2022) and individuals. Given the severe fiscal implications of such attacks, this paper contributes to a new direction of risk mitigation, namely classifying the content and form of ransomware notes used to inform users of the attack and provide instruction for ransom payment.

Neural classifiers have been used to great success for a variety of text classification tasks (Devlin et. al, 2019; Sun et. al, 2019). Expanding these technologies to ransomware note classification may add another useful tool for malware prevention: if these files are detected when scanning a download or existing folder, the data being accessed can be isolated and prevented from initially taking root. It is potentially also a very useful tool in malware detection: the presence of a ransomware note file on a system is a strong indicator that the system is compromised. Therefore, detection of ransomware note files is a highly relevant task for cybersecurity interests. However, this task is made difficult due to the low amounts of data available and lack of large public ransomware note datasets.

This paper presents the initial viability of using neural text classifiers to automatically distinguish ransomware notes from other files. This represents an important step forward in identification of digital human language content that indicates malicious activity.[2]

## 2. Background

There is some work on training neural networks to identify malicious internet traffic

[1] https://www.infosecurity-magazine.com/news/ransomware-exceed-30bn-dollars-2023/

[2] This project was completed as fulfillment of the experiential learning requirement for the minor within the Advanced Cybersecurity Experience for Students (ACES) program at the University of Maryland, College Park. The project was supervised by Dr. Judith Klavans.

(Shenfield et. al 2018, Gao et. al 2020) and malicious URLs (Yuan et. al 2020, Chen et. al 2021). There is also quite a bit of research on a similar task that uses an approach closer to what I am proposing: spam message detection (Jain et. al 2020, Sheikhi et. al 2020). However, there does not seem to have been much progress directly on the task of malicious text classification. A related paper on identifying malicious text in emails and public comments does not use ransomware notes (Baccouche et. al 2020). Lemmou et. al, 2021 published research that investigates the content of ransomware note files. In this paper, the names and contents of such files were analyzed, and they ultimately used the data they accrued to train various machine learning models to classify files as malicious and non malicious only by their filename. In this paper, we present a classification method based on content (not filename), and using pre-trained neural sequence classifiers (not one of the model types used in the previous research), achieving comparable results with some improvements. Our results provided some unexpected insight that, to our knowledge, is novel in the published literature.

## 3. Methodology

### 3.1. Gathering Data

Acquiring and aggregating data was necessary to assess if neural text classifiers can be trained to identify ransomware notes and other files that indicate malicious activity. Ensuring that the data used is clean, relevant, and robust was likely the most crucial step in this project.

#### 3.1.1. Ransomware Notes

The ransomware note data collected came from two main sources. The first was from the research published in Lemmou et. al. 2021. As of the current usage, their archive[3] contains 182 ransom note files (html-based or txt-based) from 71 distinct ransomware families. The other source of data is from RansomWhere[4], an open, crowdsourced ransomware payment tracker, where anyone can submit the ransomware they were infected with and how much bitcoin they were being asked to pay (Cable 2022). The site collects proof that a ransomware event actually occurred when someone supplies data to them, and one of the methods of proof is attaching the ransomware note. The authors of this repository have generously shared the notes collected for the use of this paper. From their data, we were able to filter out 20 useful, unique files.

After carefully selecting the files that were useful, the collection consisted of 180 files.

#### 3.1.2. Non-Malicious Files

Non-malicious files were also collected to run the classification task. Data was sampled from 4 datasets to create a somewhat comparable dataset in order to test the ability of our methods to discriminate ransomware notes from other files. These data include translated German technical support emails[5], bitcoin news articles[6], Instructables DIY articles[7], and

[3] https://github.com/lemmou/RansomNoteFiles
[4] https://ransomwhe.re
[5] https://www.kaggle.com/datasets/jordanrich/german-emails-in-xml
[6] https://www.kaggle.com/datasets/balabaskar/bitcoin-news-articles-text-corpora
[7] https://www.kaggle.com/datasets/kingburrito666/instructables-diy-all-projects

some randomly selected prose[8]. The first three were chosen for their content: technical support emails and bitcoin news articles may use vocabulary and language which is similar to that of ransomware notes, while Instructables are structured as instructions, a common feature in ransomware notes where attackers provide specific steps for those attacked to recover files. The prose was intended to be non-malicious data which is very different from ransomware notes in as many aspects as possible, to act as a sanity check to make sure the model could differentiate between text which is wildly different in nature.

The combination of all these data is meant to help the model train in a manner that is robust to various data types. Further, each setting of text has its own generally accepted grammar conventions, adding to the robustness. The emails, which are translated from German to English, allow for the peculiarities of translated text to be learned, as many ransomware notes are written as if translated from another language. (Often the ransomware note composer is not a native speaker of English). The instructables often contained typographical errors, as they are generally crowdsourced with no editing.

### 3.2. Processing Data

For the focus of this paper, predicting malicious intent from human language alone, each of these files, malicious and non-malicious, needed detailed sanitizing.

All non-alphanumeric characters except for select punctuation (period, comma, and exclamation point) and whitespaces were removed. This ensures that certain unique features are kept (e.g. '!!!' was a common punctuation sequence that could be considered a malicious feature of written human language, although this particular punctuation is common to SMS writing), while others (malformed text, undisplayed binary characters, etc) are stripped from the files. Whitespaces were standardized to a single space between words, and newline characters to no more than one consecutive instance. All html-based files were run through an html parser (html2text[9] for downloaded files, trafilatura[10] for files accessed directly from the web), so only the human language in the body of files was used, and not html tags nor undisplayed notes. Additionally, all links were replaced with a token `REPLACED_LINK`, since their existence in the text is required for grammaticality, but we did not want the model to try to learn them as distinct words.

#### 3.2.1. Ransomware Note-Specific Processing

Some ransomware note files required extra cleaning. Surprisingly, some ransomwares actually attempted to obscure their notes from detection by randomly inserting s and s of apparently randomly-generated gibberish. They appear to rely on a parser to remove all tags with certain classes. This was manually reimplemented to properly sanitize these files. Here is an example:

```
RemembetyT7ggBn03r! The worst
sicdPSxiEKStuation already
happM4cv4ened and now the future of
your files deprpends on your
determFbH9DSsfsination and speed of
your actions
```

[8] https://www.kaggle.com/proselotis

[9] https://pypi.org/project/html2text/

[10] https://pypi.org/project/trafilatura/

Some txt files also obfuscated content by assigning certain escape codes to letters (and randomly inserting them as well, which is not shown here):

```
If y\'eeu still w\'e0nt t\'ee try t\'ee
d\'e5crypt th\'e5m by y\'eeurs\'e5lf
pl\'e5\'e0s\'e5 m\'e0k\'e5 \'e0
b\'e0ckup \'e0t first b\'e5c\'e0us\'e5
th\'e5 d\'e5\'f1rypti\'een will
b\'e5c\'eem\'e5 imp\'eessibl\'e5 in
c\'e0s\'e5 \'eef \'e0ny ch\'e0ng\'e5s
insid\'e5 th\'e5 fil\'e5s.
```

Overall, the data sanitation methods were carefully tailored to the files used, but entirely reproducible.

### 3.3. Creating the Dataset

The data was all aggregated, then chunked into sequences of 128 words. Each chunk therefore yielded sequences of approximately 300-400 tokens after tokenization, ensuring all sequences fit into the BERT-based sequence model used, which has a 512 maximum token limit per sequence. This prevents truncation of any sequence: preventing any data loss might be critical in a low-resource context. Since the sequences were not evenly divisible by 128, any sequence with less than 15 words was discarded. This decision was made after examining the data and deciding that the results would not be impacted.

The data was randomly split into training, validation and test sets, as shown in Table 1. All chunks yielded from any particular file went into only one of the sets to minimize data overlap among the datasets. Examples of a chunk of each data type (ransomware note, instructable, etc.) are shown in Appendix A.

**Table 1. Data Split**

| | Number of Files | | Num. of Chunks | |
|---|---|---|---|---|
| **Train** | 306 | 70.0% | 805 | 70.9% |
| **Valid** | 66 | 15.1% | 167 | 14.7% |
| **Test** | 65 | 14.9% | 163 | 14.4% |
| **Total** | 437 | 100% | 1135 | 100% |

### 3.4. Training the Model

The distilbert-base-cased[11] model was used. This model was chosen due to its wide use in text classification tasks, and because it is a relatively small model: only 4GB of GPU was available for use in this paper (Sanh et. al, 2020). Letter case was considered an important feature for ransomware note files, as it is often employed to grab attention. See Appendix A for examples.

The model was trained using the following hyperparameters: learning_rate=1e-5, batch_size=16, epochs=4, weight_decay=0.02. The batches were split into 4 gradient accumulation steps to reduce memory usage. As the results show below, the model performed very strongly. This is why a small learning rate and large weight decay were used: to ensure smooth training and to reduce the potential for overfitting. Training was ended after epoch 4, because after that, the model began to increasingly overfit, regardless of hyperparameter setting.

## 4. Results

The results, as shown in Table 2, indicate that pre-trained neural classifiers are very easily able to predict if a 128 word excerpt is from a ransomware note or from

[11] https://huggingface.co/distilbert-base-cased

prose/instructables/bitcoin news articles. Note that in Table 2 each of the runs result in over 94% precision, recall and F1, demonstrating the high effectiveness of the classifier on the data. These unusually good results led us to further analyze our findings.

Although these results may seem surprisingly good, in actuality, this is only a marginal improvement of previous methods when classifying on just a filename: Random Forests, Support Vector Machine, Decision Tree, Naive Bayes, & Logistic Regression all achieve F1 scores of 0.890-0.920 and accuracy 0.976-0.983 (Lemmou et. al, 2021). However, what is different in this study is that our results are from classifying content, which is far more difficult to fully obfuscate than filename, as the victim ultimately needs to be able to read the ransomware note and then take appropriate action based on content

**Table 2. Accuracy, F1 Score, Precision, & Recall for 3 Runs of the Model**

| | | Acc. | F1 | Prec. | Rec. |
|---|---|---|---|---|---|
| Run 1 | Valid | 0.982 | 0.969 | 0.959 | 0.979 |
| | Test | 0.975 | 0.958 | 1.000 | 0.920 |
| Run 2 | Valid | 0.988 | 0.980 | 0.960 | 1.000 |
| | Test | 0.969 | 0.948 | 0.979 | 0.920 |
| Run 3 | Valid | 0.982 | 0.970 | 0.941 | 1.000 |
| | Test | 0.975 | 0.961 | 0.942 | 0.980 |
| **Avg** | Valid | **0.988** | **0.973** | **0.953** | **0.993** |
| | Test | **0.973** | **0.956** | **0.974** | **0.940** |

Upon investigation, two main features stood out among the sequences that the model was classifying incorrectly. The first is data that could not be perfectly cleaned: chunks which happened to contain particularly noisy data were most often misclassified. We believe that this is due to the fact that too much noise confuses the classifier. We intend to examine this further in future research. The second feature is length: though any sequence less than 15 words was discarded, nevertheless the model had difficulty with a few (but not most) sequences shorter than 30 words. This may be caused by there not being enough features in those sequences to properly classify them. Notably, all incorrect guesses were on chunks from different files, so the model does appear to have extremely good performance on predicting whether a file is a ransomware note or not. If we classify each file by averaging the scores for each of their chunks, then rounding to 0 or 1, 100% of all files will be classified correctly.

## 5. Conclusions and Contributions

In this paper, we present a novel method for ransomware note classification. We achieve strong performance by tuning a pre-trained sequence classifier. From this, a strong, but narrow conclusion emerges: bert-base-cased is able to almost perfectly distinguish ransomware notes from instructables, technical support emails, bitcoin news articles, and prose. It is our hope that future work can extend these results and demonstrate that pre-trained sequence classifiers are capable of distinguishing ransomware notes and other digital forms of human language that indicate malicious activity from all other files on a system.

This paper also makes available a cleaned dataset for future research. In doing so, we hope to support future research in this area

for the development of additional tools to defend against cyberattacks.

All code used in this paper and the dataset created can be found here: https://github.com/samm00/ransomware-note-classification

## References


A. Baccouche, S. Ahmed, D. Sierra-Sosa, and A. Elmaghraby, “Malicious text identification: Deep learning from public comments and emails,” *Information*, vol. 11, no. 6, p. 312, 2020.

A. Shenfield, D. Day, and A. Ayesh, “Intelligent intrusion detection systems using artificial neural networks,” *ICT Express*, vol. 4, no. 2, pp. 95–99, 2018.

C. Sun, X. Qiu, Y. Xu, and X. Huang, “How to fine-tune bert for text classification?,” *Lecture Notes in Computer Science*, pp. 194–206, 2019.

J. Cable, “Ransomwhere: A Crowdsourced Ransomware Payment Dataset”. *Zenodo*, May 02, 2022. doi: 10.5281/zenodo.6512123.

G. Jain, M. Sharma, and B. Agarwal, “SPAM detection on social media using semantic convolutional neural network,” *Deep Learning and Neural Networks*, pp. 704–719, 2020.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional Transformers for language understanding,” *arXiv.org*, 24-May-2019. Available: https://arxiv.org/abs/1810.04805.

J. Yuan, G. Chen, S. Tian, and X. Pei, “Malicious URL detection based on a parallel neural joint model,” *IEEE Access*, vol. 9, pp. 9464–9472, 2021.

M. Gao, L. Ma, H. Liu, Z. Zhang, Z. Ning, and J. Xu, “Malicious network traffic detection based on Deep Neural Networks and association analysis,” *Sensors*, vol. 20, no. 5, p. 1452, 2020.

S. Sheikhi, A. Bazzazi, and M. T. Kheirabadi, “An effective model for SMS SPAM detection using content-based features and averaged neural network,” *International Journal of Engineering*, vol. 33, no. 2, 2020.

V. Sanh, L. Debut, J. Chaumond, and T. Wolf, “Distilbert, a distilled version of Bert: Smaller, faster, cheaper and lighter,” *arXiv.org*, 01-Mar-2020. [Online]. Available: https://arxiv.org/abs/1910.01108.

Y. Lemmou, J.-L. Lanet, and E. M. Souidi, “In-depth analysis of ransom note files,” *Computers*, vol. 10, no. 11, p. 145, 2021.

Z. Chen, Y. Liu, C. Chen, M. Lu, and X. Zhang, “Malicious URL detection based on improved multilayer recurrent convolutional neural network model,” *Security and Communication Networks*, vol. 2021, pp. 1–13, 2021.

## Appendices

### Appendix A. Examples

| | |
|---|---|
| **Ransom-ware Note** | "bar of your browser and press ENTER; 3. wait for the site loading; 4. on the site you will be offered to download Tor Browser; download and run it, follow the installation instructions, wait until the installation is completed; 5. run Tor Browser; 6. connect with the button Connect if you use the English version; 7. a normal Internet browser window will be opened after the initialization; 8. type or copy the address REPLACEDLINK 456789ABCDEF0123 in this browser address bar; 9. press ENTER; 10. the site should be loaded; if for some reason the site is not loading wait for a moment and try again. If you have any problems during installation or use of Tor Browser, please, visit REPLACEDLINK and type request in the search bar Install Tor Browser" |
| **Instruct-able** | "automation of mechanical printing with the reliability and low cost of using sharpies. I have been pondering using my 3d printer as a plotter for quite a while now, so this was a great excuse to go ahead and figure it out.<br>Step 2: Material<br>For this ible, you will need A 3D printer. I am using a printrbot simple metal A plotter accessory for your 3D printer some 1 32th drill bit. They are fragile, so either get a multipack, or a quality carbide one To use those, you need an incredibly stable hand, a drill press or drill stand. The drill stands start at about 1 3rd the price of a drill press and go up to the price of a starter drill press. Meanwhile, for not too" |
| **Prose** | "HOWELL, M's Butler and confidential Servant. CHARLES PAGE, Footboy. BULKELEY, Lady Kicklebury's Servant. MR. BONNINGTON. Coachman, Cabman; a Bluecoat Boy, another Boy Mrs. Prior's Sons. LADY KICKLEBURY, Motherinlaw to Milliken. MRS. BONNINGTON, Milliken's Mother married again. MRS. PRIOR. MISS PRIOR, her Daughter, Governess to Milliken's Children. ARABELLA MILLIKEN, a Child. MARY BARLOW, Schoolroom Maid. A grownup Girl and Child of Mrs. Prior's, Lady REPLACEDLINK Maid, Cook. THE WOLVES AND THE LAMB. ACT I. Scene.MILLIKEN'S villa at Richmond; two drawingrooms opening into one another. The late MRS. MILLIKEN'S portrait over the mantelpiece; bookcases, writingtables, piano, newspapers, a handsomely furnished saloon. The backroom opens, with very large windows, on the lawn and pleasureground; gate, and wallover which the heads of a cab and a carriage are seen, as persons arrive. Fruit, and a ladder on" |
| **BTC News Article** | "credibility under threat, as a corruption scandal damaged lawmakers careers and fingers were pointed at Qatari officials accusedRead Article Latest Stock Market News 5 hours agoNew FTX CEO says lax oversight, bad decisions caused failure5 hours agoFlorida Senate passes property insurance overhaul6 hours agoWall Street closes higher after inflation cooled in November6 hours agoHow major US stock indexes fared Tuesday 12 13 20227 hours agoFTX's BankmanFried charged by US for scheme to defraud8 hours agoWhat is Market Structure in Trading?12 hours agoInstitutional Selling Is No Headwind For Nike 13 hours agoStocks open sharply higher as inflation cools in November13 hours agoMullen Automotive Shifts Into Higher Gear13 hours agoInvesting Inheritance: Creating a Cycle of Wealth14 hours agoMillennial Money: Prepping to buy a home or invest in 2023?15 hours agoSEC" |
| **Support Email** | "I tried to burn my product preview with 330 files. There was an error message issued: wmem reports disc project build exception 86661f3 encrypting the process failed. After this I have implemented the recommended assistance without burning the sequence and chapter display, I was informed again that the encryption failed. It can maybe this be the product preview version 10 first class on the same machine installed, could be the cause of these problems? I don't know anyone now Advice more !!" |