Basem Saleh (bsaleh2@jhu.edu)

Keyboard Keystroke Audio Prediction Analysis

Abstract

This project seeks to experiment with the paper titled *A Practical Deep Learning-Based Acoustic Side Channel Attack on Keyboards.* The goal of the paper was to show how keystrokes of a keyboard can be identified through a neural network using the audio information grabbed. The focus was placed on recreating this same experiment with a similar data collection process. In addition to training on different mechanical switches, the CoAtNet model used was modified to become a fusion model, so it can accept additional keyboard information that is non-auditory to help further training. Results shown indicate reliable performance for single key prediction, but poor across multiclass predictions and with a limited dataset collected. The overall dataset used had a dashboard created on Tableau Public to illustrate properties of the collected data.

Introduction

While traditional cyber attacks can be as minimal as phishing forms, good practices like having password vaults and general avoidance of these forms can prove very useful for general users[1]. When clicking malicious links, users may encounter key-loggers. This type of program will consistently record all keystrokes performed by a user once the file reaches the computer[2]. Recent developments have occurred that now been discovered that allows keystroke prediction without the need getting into the computer itself but rather utilize just an external microphone.

This project will focus on predicting keystrokes using recorded data from a MacBook Scissor keyboard, a linear mechanical keyboard, and tactile mechanical keyboard to attempt to

prove the alternate hypothesis to accurately predict correct key presses using a machine learning model trained on different key switch keystroke data with an 80% accuracy or greater. Otherwise, reject the null hypothesis of training a model that is no better than random guess of keyboard strokes. This will check to see if the model requires further understanding of the keystroke audio, such as keyboard material and keyboard sizes to better predict keys.

Background

The main background for this paper falls upon the paper titled *A Practical Deep Learning on Acoustic Side Channels* (also known as DeepKeyAttack). The paper[3] this project focuses on accomplished the ability of predicting keyboard strokes using a trained neural network. The paper focused on data that was collected from a single MacBook Pro keyboard while recording keystrokes from a phone microphone only a few inches away to the side. In an effort to better preserve the controlling nature of the project, the phone was kept on a thick cloth and each desired key was pressed 25 times at varying pressures. The model also predicted keystrokes over a Zoom call after recording data from keys in the same Zoom environment. Accuracy was very high, around 92% over zoom and 95% without zoom.

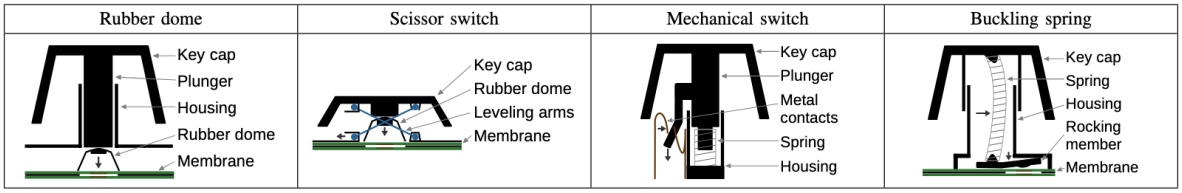

Fig. 1. Common Key switches[4]

One part of this paper that stuck was that this was only done on a MacBook Pro keyboard. This keyboard has a specific scissor design that is different than traditional membrane keyboards found on laptops or desktops, and also mechanical keyboards. While this paper proved

useful in predicting keystrokes on a scissor keyboard, the chance to see if mechanical keyboards can have keystrokes predicted in a similar fashion would prove a bigger challenge since there is a much bigger variety in switch types such as clicky, tactile, and linear type switches[6]. On top of that, keyboard material, the size of the keyboard, and other variables can also influence sound.

Two additional papers, *SoK: Keylogging Side Channels*[4] and *Robust Keystroke Transcription from the Acoustic Side-Channel*[5] also perform similar model trainings and demonstrate results that show how easily predictions can be made. The three papers train various models to help with keystroke prediction. There is emphasis on tackling work across different types of keyboards, microphones, and environments. So, a focus of this project is developing a recording setup that can be easily distributed and used to record data from different users who are enthusiasts of different keyboard types. Since recording different keyboards from different users includes different environments, this also tackles the issue of different microphones being utilized within different rooms as well.

Approach

Keyboard Recorder Dataset

To start, the keyboard_recorder[7] repository was created to start the actual recordings of the data. This repository houses the data collection scripts and a link to the recorded dataset within a JHU OneDrive folder. This repository aimed to repeat the same recording experiment from the DeepKeyAttack dataset by prompting a user to record specific keys and press them a certain number of times to be later preprocessed to extract those pulses.

Fig. 2. Recording setup of Tactile portion of Keyboard Recorder dataset.

The recording scripts instruct the user to place the microphone next to the keyboard, and then use two scripts. The first one to record general information, such as the type of key switch used, keyboard size, and the brand name of the keyboard, which was used to determine what the material of the keyboard was. This will allow the model to learn additional, tabular features that are separate from the audio and could prove useful for training. The second script allows user to select the key they want to record, and then record said data. After completing a key, users are prompted to move to the next key to continue recording. In total, there are three users worth of recordings that are comprised of a linear keyboard, a tactile keyboard, and scissor switches from a Macbook Pro keyboard(grabbed from the Deep Key Attack dataset). There are 48 total classes, with keys ranging 0-9, a-z, brackets, semicolon, tilde, quote, forward slash, backslash, period, comma, plus, minus, and the space bar. Importing the Deep Key Attack dataset only provided 0-9 and a-z which becomes 36. The linear and tactile portions have all 48 classes. In addition, tabular data is recorded for each set that includes the keyboard size, material, and switch types for further model inputs.

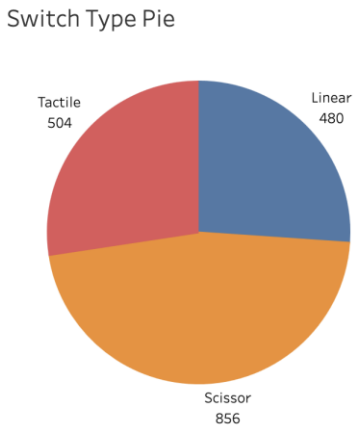

Fig. 3. Pie chart detailing split between the different recorded keyboards

Data Processing and Experimentation

In addition to the recording, proper audio file preprocessing was required to take place. Each audio file was a recording of twenty-five keystrokes. To better isolate the strokes to automatically, each file had its energy calculated, and then normalized across all of the data using a sliding window of size 50 samples. Then, two thresholds were placed on to the average sample: a lower threshold and an upper threshold. These were tested across various keys to get a threshold that can best get all of the keys from an audio file without accidentally grabbing unknown noise artifacts. And values within a continuous range of the signal where it exceeds the noise and upper threshold is considered a pulse.

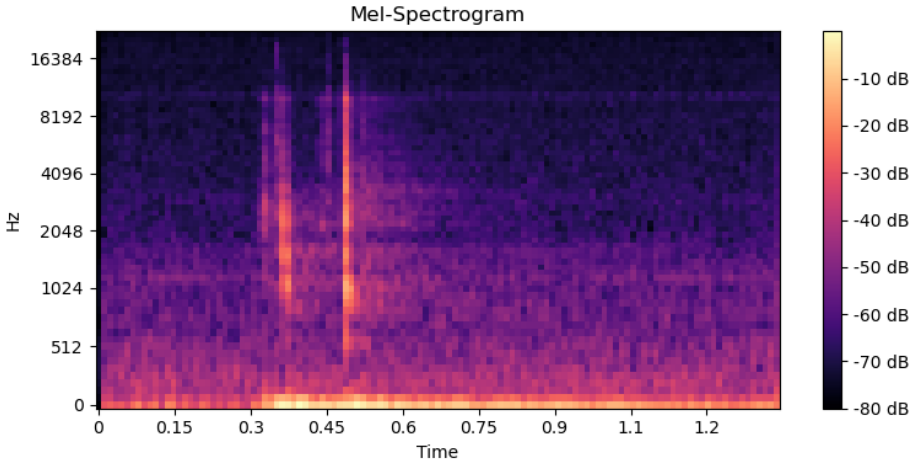

Fig. 4. Mel-Spectrogram of the Z key from the Tactile keyboard recording.

Audio was then passed through the Mel-spectrogram[12]. A Mel-spectrogram is a version of a general spectrogram[13]. The Mel-spectrogram will convert each unit of frequency to an adjusted Mel-scale: a logarithmic scale more representative of how humans hear sound. After pulse extractions from the recorded keystrokes are complete, they are placed through a function to produce Mel-spectrograms with the number of Mel coefficients at 64, a window size of 1024, and a hop length of 500. This was done to mimic the results of DeepKeyAttack paper preprocessing.

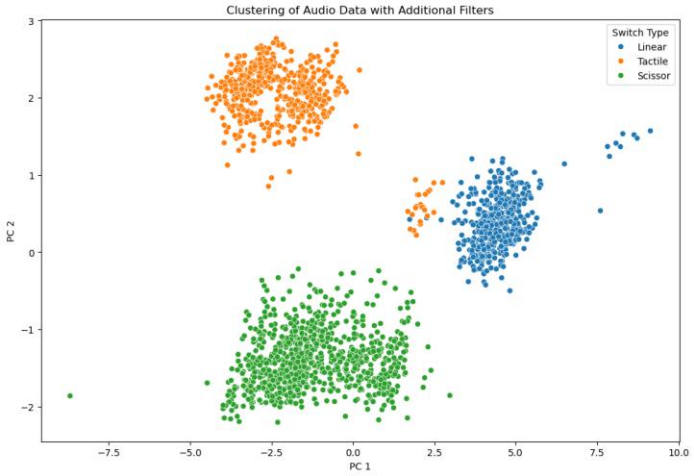

Fig. 5. Clustering of audio features, reduced to two PCA components for plotting. Hue is the type of switch from the Keyboard Recorder dataset.

To get a better understanding of the recorded data, each extracted pulse was placed through processing to extract thirteen Mel-frequency cepstral coefficients. This was chosen to extract features of the pulses while reducing dimensionality[8]. After returning the coefficients, they were put through a PCA algorithm to extract two components[9], which were then grouped into 3 clusters through KMeans clustering[10]. The figure above shows clear distinction between the different audio pulses.

However while the pulses look separated from each other due to different switch types, this does not account for the different microphones used for the various recordings of each dataset[11]. So, instead of the pulse itself, there is potential that this is just the microphones

being separated into different clusters and the clusters also just happen to have the different switches for each recording. A more substantial recording dataset could help further get around this.

The model used in this experiment is a fusion-modified Convolution and Self Attention Neural Network(CoAtNet)[14]. CoAtNet is a neural network architecture that combines convolutional layers and transformer encoders, aiming to leverage the strengths of both CNNs and transformers[15,16]. The CoAtNet is the same model used from DeepKeyAttack. The fusion[17] was my personal addition that adds a Multi-layer Perceptron[18] to ingest the additional encoded features, like keyboard size. This will help learn both audio and features of keyboards to better understand key distinctions.

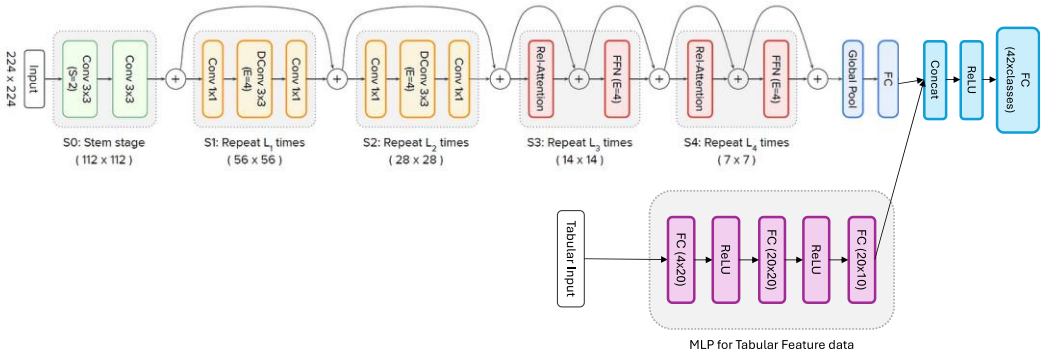

Fig. 6. Architecture of the CoATNet Fusion Modal. Top section is the CoAtNet before the fusion portion. Bottom is the Multi-layer Perceptron that processes feature data before concatenation. After concatenation, a final linear layer combines both outputs.

Accuracy was chosen as the metric since getting exact keys is needed to verify a working algorithm. So, accuracy was chosen as the metric. Previously mentioned papers have used the same metric. Five experiments were run that remove portions of the dataset, perform binary classification, multi-class classification, or a mix of these.

Training Method

All data was preprocessed in one jupyter notebook by creating one dataframe that featured encoded audio file data, alongwith a path to that dataset. During training, a custom

PyTorch dataset performs the mel-spectrogram transformation when being loaded into the model. Since there weren't as many keyboards as desired, the only features, outside of the mel-spectrogram of audio, was the switch type used(tactile, linear, scissor) and the size of the keyboards. Data was split 80/20 for training and testing using a total of 1840 pulses.

Results

TABLE I. KEYBOARD RECORDER EXPERIMENT ACCURACY METRICS

Experiments	Classes	Keyboard Recorder Dataset	Accuracy
DeepKeyAttack	0-9, a-z	MBPWavs	95.00%
Exp. #1	0-9, a-z, punctuation	Full	23.64%
Exp. #2	0-9, a-z, punctuation	Tactile	1.04%
Exp. #3	'h'	Full	98.64%
Exp. #4	'h'	Tactile	96.87%
Exp. #5	'space'	Tactile	94.79%

The table here highlights the accuracy of each experiment performed, the classes used during the experiment, and which portions of the Keyboard Recorder dataset used. The experiment from *A Practical Deep Learning-Based Acoustic Side Channel Attack on Keyboards* is included here for comparison purposes.

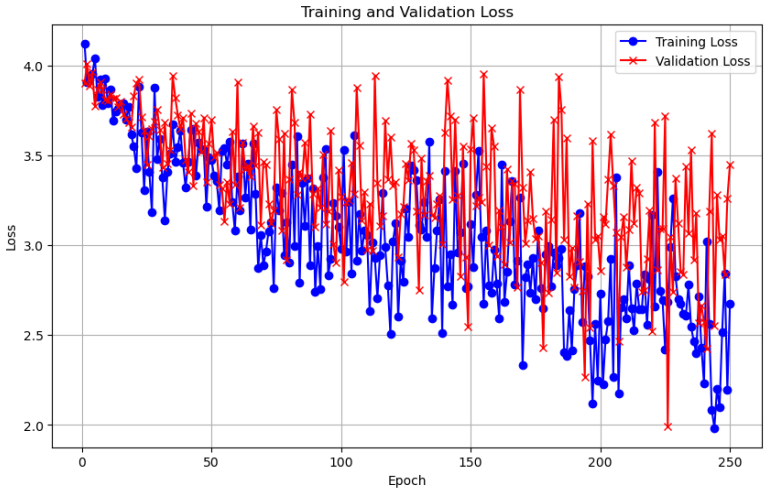

Fig. 7. Experiment #1 training and validation curves on the CoAtNet Fusion model. Using all of the Keyboard Recorder dataset

While convergence was continuing on the training and validation curves for Experiment #1, after 250 epochs, the overfitting was then present as the validation curve began to increase in error while training decreased. Table I indicates the accuracy after training to be close to 24%. While more training epochs show it fitting better on the data, total accuracy never exceeded beyond what's shown on Table I. Similar training curves were present for the rest of the experiments and are not shown here.

Conclusion

The experiments aimed to demonstrate that keystrokes could be differentiated based on keyboard switch types, but the results showed that multiclass classification was difficult for this model. Binary classification was good, but does not fall under the bigger hypothesis to predict between all classes at once.

The poor performance in the experiments could be due to several factors: First, an insufficient amount of data was the most paramount. Had the dataset been more diverse with more sample recordings, then accuracy may have improved for a few experiments. Second, microphone variability may have heavily influenced these results. The tabular features may have encoded the background environment for which the three keyboards were recorded so instead of learning filters between the keyboards, the model may have only learned to differentiate the microphones used. Future experiments should consider recording all keyboards under the same. And lastly, there were more scissor keys present in the data than the other two switch types, causing a slight imbalance that may have affected performance of certain experiments. Overall, suggestions of better data collection and potential data augmentation could improve keystroke classification efforts.

References

[1] J. Wayburn, “Two-Step Phishing Campaign Exploits Microsoft Office Forms,” *Perception Point*, Jul. 25, 2024. https://perception-point.io/blog/two-step-phishing-campaign-exploits-microsoft-office-forms/ (accessed Aug. 11, 2024).

[2] “What is a Keylogger? | How to Detect Keyloggers,” *Malwarebytes*. https://www.malwarebytes.com/keylogger#:~:text=Keyloggers%20are%20a%20particularly%20insidious

[3] J. Harrison, E. Toreini, and M. Mehrnezhad, “A Practical Deep Learning-Based Acoustic Side Channel Attack on Keyboards,” 2023. Available: https://arxiv.org/pdf/2308.01074

[4] John V. Monaco (2018). SoK: Keylogging Side Channels. *2018 IEEE Symposium on Security and Privacy (SP)*, 211-228.

[5] Slater, D., Novotney, S., Moore, J., Morgan, S., & Tenaglia, S. (2019). Robust keystroke transcription from the acoustic side-channel. In *Proceedings of the 35th Annual Computer Security Applications Conference* (pp. 776–787). Association for Computing Machinery.

[6] “Switch Types - Mechanical Keyboard,” *Mechanical-Keyboard.org*, Nov. 26, 2015. https://www.mechanical-keyboard.org/switch-types/

[7] B. Saleh, “Keyboard Recorder,” Github, Aug. 11, 2024. https://github.com/bsaleh524/keyboard_recorder (accessed Aug. 11, 2024).

[8] U. Kiran, “MFCC technique for speech recognition,” Analytics Vidhya, Jun. 13, 2021. https://www.analyticsvidhya.com/blog/2021/06/mfcc-technique-for-speech-recognition/

[9] Z. Jaadi, “A Step by Step Explanation of Principal Component Analysis,” Built In, Feb. 23, 2024. https://builtin.com/data-science/step-step-explanation-principal-component-analysis

[10] P. Sharma, “The Most Comprehensive Guide to K-Means Clustering You’ll Ever Need,” Analytics Vidhya, Aug. 19, 2019. https://www.analyticsvidhya.com/blog/2019/08/comprehensive-guide-k-means-clustering/

[11] P. Sharma, “The Most Comprehensive Guide to K-Means Clustering You’ll Ever Need,” Analytics Vidhya, Aug. 19, 2019. https://www.analyticsvidhya.com/blog/2019/08/comprehensive-guide-k-means-clustering/

[12] “Mel spectrogram - MATLAB melSpectrogram,” www.mathworks.com. https://www.mathworks.com/help/audio/ref/melspectrogram.html

[13] “What is a Spectrogram?,” Pacific Northwest Seismic Network. https://www.pnsn.org/spectrograms/what-is-a-spectrogram

[14] Z. Dai, H. Liu, Q. Le, and M. Tan, “CoAtNet: Marrying Convolution and Attention for All Data Sizes.” Accessed: Aug. 11, 2024. [Online]. Available: https://arxiv.org/pdf/2106.04803

[15] IBM, “What are Convolutional Neural Networks? | IBM,” www.ibm.com. https://www.ibm.com/topics/convolutional-neural-networks

[16] A. Vaswani et al., “Attention Is All You Need,” Jun. 2017. Available: https://arxiv.org/pdf/1706.03762

[17] W. Li, Y. Peng, M. Zhang, L. Ding, H. Hu, and L. Shen, “Deep Model Fusion: A Survey.” Accessed: Aug. 11, 2024. [Online]. Available: https://arxiv.org/pdf/2309.15698

[18] S. Abirami and P. Chitra, “Multilayer Perceptron - an overview | ScienceDirect Topics,” www.sciencedirect.com, 2020. https://www.sciencedirect.com/topics/computer-science/multilayer-perceptron