

Interpreting Deep-learning Models: An Exploratory Study with TypeNet for Keystroke Dynamics

1st Mia Onodera
Electrical and Computer Engineering
University of Washington
Seattle, Washington
onoderam@uw.edu

2nd Charles Devlen
Computing and Information Sciences
Rochester Institute of Technology
Rochester, USA
cmd3616@rit.edu

3rd Daqing Hou
Software Engineering
Rochester Institute of Technology
Rochester, USA
dqvse@rit.edu

Abstract—In recent years, researchers have been exploring the usage of Siamese Neural Network (SNN) architectures for use in keystroke dynamics. One such model, TypeNet [1], achieves state-of-the-art performance for keystroke dynamics based authentication. If how TypeNet accomplishes classification can not be explained, user trust will be low, and subsequently, usage of the model will be low. In order to address these concerns, we have created a new methodology for model explainability, and evaluated it on the TypeNet model architecture using the Clarkson II dataset [4]. Our method is two part; firstly, it focuses on finding the most impactful features for the model, then focuses on finding which indices of the output embedding are most impacted by each feature per input digraph. These findings contribute to explaining embedding generation using the TypeNet model, and this methodology could be adapted to many different architectures due to its flexible nature.

Index Terms—Explainability, Deep learning, keystroke dynamics, TypeNet

I. INTRODUCTION

Siamese Neural Networks (SNN) show great potential for deep learning-based authentication. One such model, TypeNet [1], utilizes the SNN architecture for keystroke dynamics to identify users based on their unique typing patterns. TypeNet has demonstrated state-of-the-art performance, achieving an Equal Error Rate (EER) of 2.2% [1] on the Aalto Desktop public dataset [2]. This model enhances community security by providing a non-invasive, user-friendly authentication method, particularly benefiting individuals who struggle with traditional passwords or biometric systems. For potential commercialization, TypeNet’s effectiveness and ease of use make it a strong candidate for integration into various applications, from consumer electronics to enterprise security systems. Despite its effectiveness, TypeNet and similar deep learning models often face challenges with user trust due to their black-box nature, which obscures how or why decisions are made. To build user confidence in TypeNet’s behavior it’s crucial to develop methods that enhance transparency, making the model’s processes more understandable and explainable.

To address these transparency concerns and enhance user trust, several strategies can be implemented. One approach is using techniques similar to SHAP (SHapley Additive exPlanations), which is commonly used in machine learning

models to provide interpretable insights [3]. Another method is employing masking techniques, which can help identify which parts of the input data (such as specific keystrokes or typing patterns) are most influential in the model’s decision-making process [6]. These techniques can enhance the interpretability of deep learning models, making them more transparent and trustworthy to users. We firstly examine a masking-based approach to determine relative feature contribution to model performance in terms of EER, then use an approach that involves both masking and taking the difference between model embeddings to get the importance of features at a per-index basis in the output embedding. The simplicity of our method allows use cases on any embedding generating deep learning model, and makes it possible to map specific keystroke digraph inputs to specific model indices in the output embedding.

This paper is organized as follows: Section II examines works related to our own in the area of explainability. Section III details our methodology for our model explainability method, and the rationale behind it. Section IV describes the results we found when implementing our approach on the TypeNet model with the Clarkson II dataset. Finally, Section V closes our paper.

II. RELATED WORKS

Other methods for explainability exist and have been successfully used to visualize how models are transforming input features to output embeddings, as well as the contribution of specific features for a specific input. These methods seek to make the models align further with human intuition, as well as be visually explainable.

Ribeiro et al. created LIME (Local Interpretable Model-agnostic Explanations) [5] to interpret model predictions by learning a model locally around the model prediction. The resulting model highlights which features both contributed to/against a certain prediction. This leads to the outputs of models to be much more interpretable, as LIME ties the input features contributions to the output. This approach is described as “model agnostic” as it can be applied to any model, as all that’s needed is a trained model to make predictions.

Another example is Lundberg and Lee’s SHAP (SHapley Additive exPlanations) [3], which unified six other model ex-

plainability approaches, including LIME, in a way that results in an “explanation model”. The premise of the explanation model is to view any explanation of a model prediction as a model itself. This allows visualization of models in a way that is more consistent with human intuition. SHAP also produces “SHAP values”, which are a per-feature measure of how the features contribute to the model prediction. Through experimentation, Lundberg and Lee were able to demonstrate that their unified approach produced SHAP values that were desirable, as they were more in line with the human reasoning/explanation behind a decision.

Our method differs from the above, as it gives per-index measures of feature contributions to embeddings for specific input digraphs. This gives an idea of how the model transforms the input to the 128 length output embedding. When this is done across multiple input samples, the resulting values can be visualized using a heat map, allowing for trends in how embeddings are generated to be observed visually.

III. METHODOLOGY

To investigate the impact of specific features on the performance of the TypeNet model for keystroke dynamics-based authentication, we employed a feature masking approach. This method, commonly used to visualize the importance of features in convolutional networks, involves setting certain features to zero and observing changes in performance metrics, specifically the EER. By applying this technique, we looked to differentiate which features are most influential in the model’s decision-making process.

To simulate real life scenarios, we decided to use the Clarkson II dataset [4], which consists of keystroke data of 103 subjects typing on a desktop keyboard over 2.5 years, with the nature of data collected being fully free text. We processed three types of sentences: anchor, genuine, and imposter, each containing digraph (key pair) features, which are defined as follows:

- 1) *Dwell Time (M)* - represents the time spent lingering on a single key, from key press to release.
- 2) *Flight Time Up-Down (UD)* - represents the time between a key being released (up) until the next key being pressed (down).
- 3) *Flight Time Down-Down (DD)* - represents the time between a key being released (down) until the next key being pressed (down).
- 4) *Flight Time Up-Up (UU)* - represents the time between a key being released (up) until the next key being pressed (up).
- 5) *Digraph ID (ID)* - denotes the two keys being pressed consecutively, represented by the product of their respective ASCII values, each divided by 255.

These features were extracted through preprocessing data from the raw Clarkson II dataset. The users were then split into train and test datasets, consisting of 65 and 5 users respectively, with a further 5 users set aside as the test set. The topology of the TypeNet model used for triplet loss training is as follows:

- 1) *Masking Layer* - This layer is used to skip or ignore certain timesteps in the input data that are marked with 0. This is useful for variable-length sequences where shorter sequences are padded with zeros to match the length of the longest sequence in a batch.
- 2) *Batch Normalization Layer* - This layer normalizes the output of the previous layer by adjusting and scaling the activations to have a mean of zero and a standard deviation of one. This helps in speeding up training and achieving more stable gradients.
- 3) *128 length LSTM Layer* - The Long Short-Term Memory (LSTM) layer is a type of recurrent neural network (RNN) layer that is capable of learning long-term dependencies. It is particularly effective for sequential data as it maintains a memory of previous inputs, allowing it to understand context over time.
- 4) *Dropout Layer* - This layer is used to prevent overfitting by randomly setting a fraction of input units to zero during training.
- 5) *Batch Normalization Layer* - Another instance of batch normalization applied after subsequent layers to normalize activations and improve training stability and speed.
- 6) *128 length LSTM Layer* - Another instance of an LSTM layer, which further processes the sequential data, often allowing the network to capture more complex patterns and dependencies by adding depth to the model. This layer generates the final 128 length output embedding.

The TypeNet model performs triplet loss in order to minimize the Euclidean distance between the embeddings of anchor and genuine sentences, and maximize the distance between the embeddings of anchor and imposter sentences. Equation 1 defines the formula for triplet loss, where α , commonly known as margin, represents the desired degree of separation between the anchor and negative samples.

$$L_t = \max\{0, \|\mathbf{f}(\mathbf{x}_A^i) - \mathbf{f}(\mathbf{x}_P^i)\|^2 - \|\mathbf{f}(\mathbf{x}_A^i) - \mathbf{f}(\mathbf{x}_N^j)\|^2 + \alpha\} \quad (1)$$

The EER for the model is determined using the embeddings produced by the model for the test dataset. After generating these embeddings, we calculate the Euclidean distances between them and compare these distances to the threshold calculated on the test set. The EER is then computed based on whether these distances fall below or above this threshold, providing an indication of the model’s performance in distinguishing between genuine and imposter samples.

We established a baseline EER for models trained on sentences of varying lengths of 70, 150, and 200 digraphs in order to assess how sentence length impacts model performance. For each sentence, we performed feature ablation by masking one feature across all digraphs in a sequence and observed the resulting EER value. This process allowed us to understand how the removal of specific features influences the model’s accuracy and its ability to distinguish between genuine and imposter sentences across different sentence lengths.

In addition to the feature masking approach, we employed a second method to further analyze the TypeNet model’s output embedding. This method involved masking each feature in every sample and then averaging the resulting embeddings across the selected digraph. We then applied the same methodology without masking any features to establish a baseline. By comparing these baseline embeddings to those obtained from the masked feature approach, we aimed to understand the structure of the embedding to see if specific indices correlated with specific features.

For this analysis, we focused on key features such as key ID, dwell time, and flight time. After masking a particular feature, we computed the model’s embeddings for all samples and then averaged these embeddings to assess the cumulative impact of the masked feature on the model’s representation space. This approach allowed us to identify which features, when masked, caused the most significant shifts in the embeddings, thereby indicating their relative importance.

IV. RESULTS

Our analysis through feature masking highlighted the varying impacts of different features on model performance. The control EER values were 17.1, 10.47, and 12.77 for sentence lengths of 70, 150, and 200 digraphs, respectively, with the 150-token model showing the lowest EER of 10.47.

Masking specific features led to notable changes in EER. For instance, masking the digraph ID (ID) increased the EER to 34.8, and masking the dwell time (M) raised it to 37.5, significantly higher than the control value of 10.47. The model also found interest in Flight Time Down-Down (DD) with an EER increase to 30.8. In contrast, masking other features, such as Flight Time Up-Down (UD) and Up-Up (UU), resulted in smaller increases in EER, with values reaching 17.0 and 19.4, respectively.

Combining the two least influential features (Flight Time Up-Down and Up-Up) did not lead to a significant increase in EER. When both features were masked simultaneously, the EER rose to 25.7, which is a modest increase compared to the individual effects of the more influential features. This indicates that while key press intervals and the masking layer do affect model performance, their impact is less significant compared to features like digraph ID, dwell time, and Flight Time Down Down.

Further examination through embedding analysis indicated that certain letter digraphs played a more critical role in the model’s output, highlighting their importance in the authentication process. By isolating the TypeNet model and comparing output vectors from both original and masked input samples for specific digraphs, we gained insights into how different features contribute to the model’s embedding space and overall performance.

We observed that specific indices within the 128-dimensional output embeddings were consistently activated across different digraphs for the ID feature, specifically at indices 75, 111, and 112 as seen in Figure 2a. This suggests that the ID feature is transformed into these indices when its

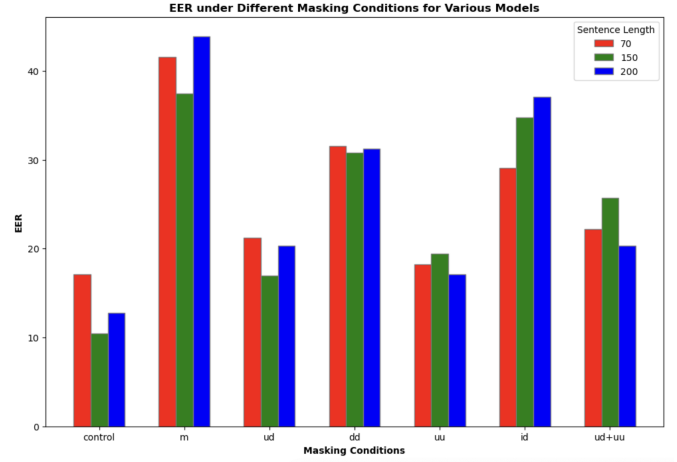
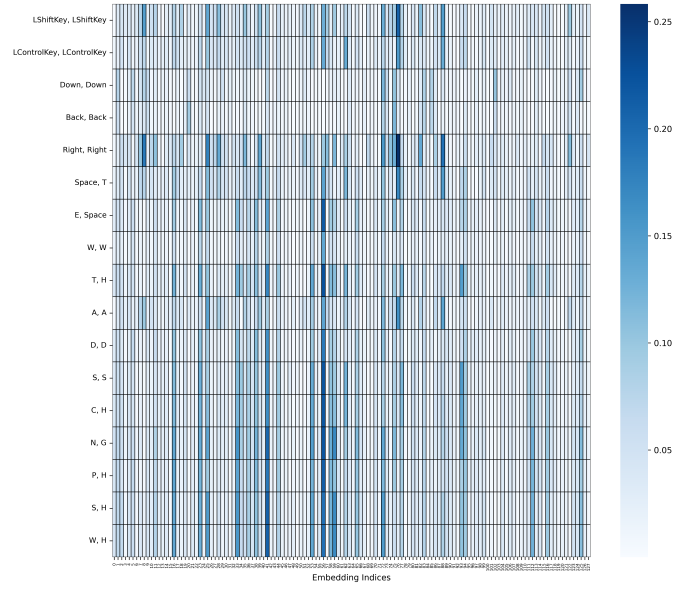
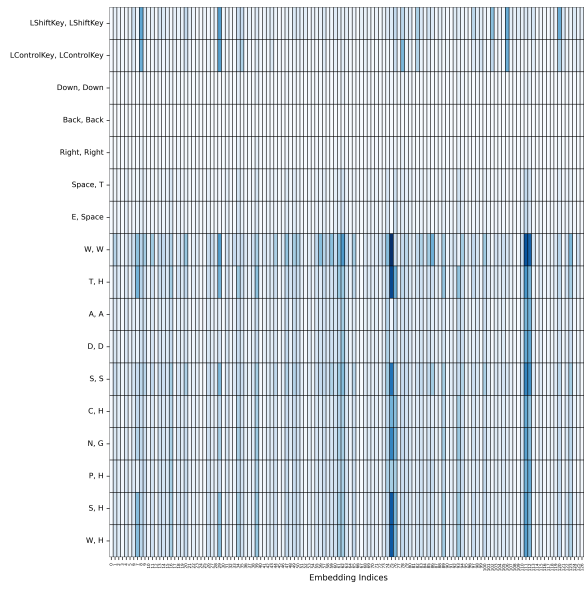


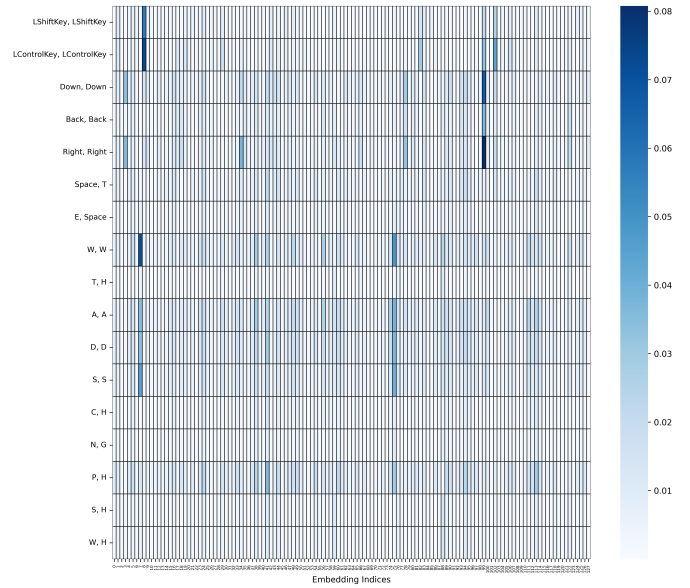
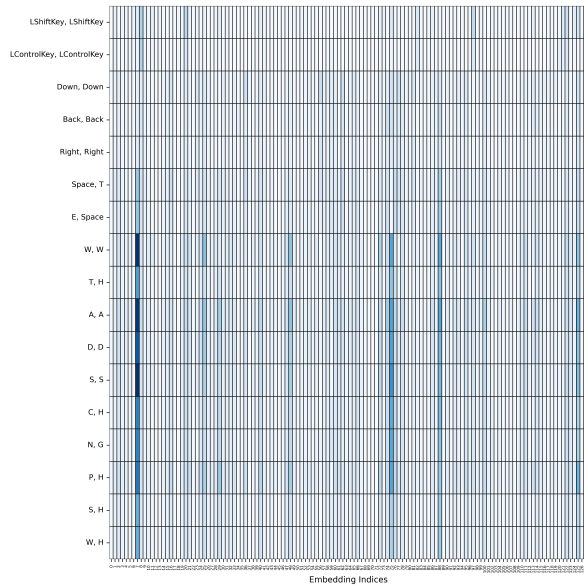
Fig. 1: EER Under Different Masking Conditions for Various Models

passed through the model. In addition, we found that letter digraphs are more important for the model’s performance compared to control characters, which is shown through the darker hue within the heat map indices.

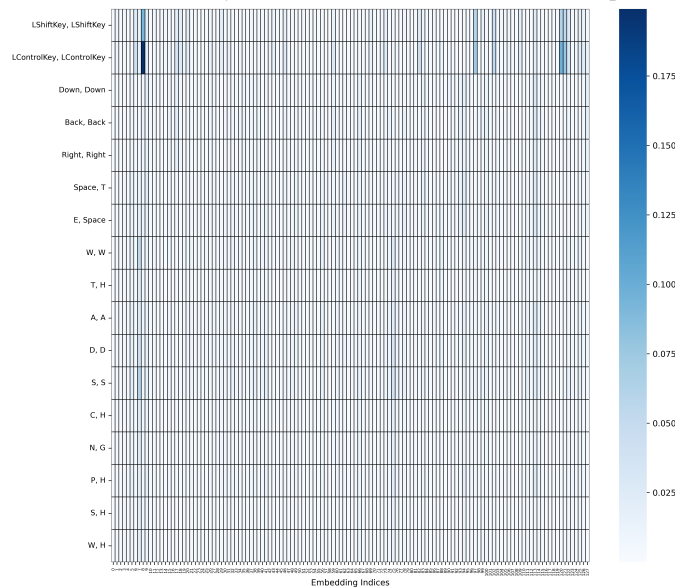
Using this same procedure, we found that Flight Time Down-Down consistently activated index 7 among letter keys, as seen in Figure 2c, suggesting that letter digraphs might have more consistent index value for this feature. We observed that dwell time was most predominantly activated throughout the heatmap, with consistent activation at indices 41 and 56, as illustrated in Figure 2b. However, Flight Time Up-Down, and Flight Time Up-Up did not show consistent results in specific indices, as seen in Figures 2d, and 2e. Their activations were either dispersed across many indices or only highlighted for specific digraphs, indicating that the feature had limited impact to the embedding, making it more difficult to pinpoint specific indices. This is likely due to these features being less important as found in our ablation experiment with EER.



(a) Heat Map for Digraph ID (75, 111, and 112 Consistently Activated) (b) Heat Map for Dwell Time (41 and 56 Consistently Activated)



(c) Heat Map for Flight Time Down-Down (7 Consistently Activated) (d) Heat Map for Flight Time Up-Down



(e) Heat Map for Flight Time Up-Up

Fig. 2: Heat Map Visualization of How Indices of Output Embedding Are Activated by Selected Features

V. CONCLUSION

In this study, we explored two primary methods to assess model explainability under masking conditions and identified that dwell time, digraph ID, and flight time down-down are significant features, with dwell time being the most critical. Our analysis of the model embeddings revealed that specific indices were activated by particular features, suggesting a potential for quantifying these indices to gain deeper insights into the model's behavior. Notably, letter digraphs were found to be more influential than control digraphs, especially for digraph ID and flight time down-down, as they consistently triggered more indices among letter digraphs. Lastly, features such as flight time up-down and up-up proved challenging to correlate with specific indices.

Moving forward, the next steps include quantifying the heatmaps and establishing thresholds to validate our hypothesis that indices are related to features. This will help determine if our observations hold true and further enhance our understanding of the model's interpretability. By refining these methods, we aim to improve model performance and provide more transparent and robust solutions for real-world applications. Testing our method on different embedding models

and modalities, such as mouse dynamics, could also provide a deeper insight into different modalities and how applicable this method is to other types of data.

ACKNOWLEDGMENT

This work was partially supported by NSF award OAC-2244049.

REFERENCES

- [1] A. Acien, A. Morales, J. V. Monaco, R. Vera-Rodriguez, and J. Fierrez. TypeNet: Deep learning keystroke biometrics. *IEEE Transactions on Biometrics, Behavior, and Identity Science*, 2021.
- [2] V. Dhakal, A. M. Feit, P. O. Kristensson, and A. Oulasvirta. Observations on typing from 136 million keystrokes. In *Proceedings of the 2018 CHI conference on human factors in computing systems*, pages 1–12, 2018.
- [3] S. M. Lundberg and S.-I. Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.
- [4] C. Murphy, J. Huang, D. Hou, and S. Schuckers. Shared dataset on natural human-computer interaction to support continuous authentication research. In *2017 IEEE international joint conference on biometrics (IJCB)*, pages 525–530. IEEE, 2017.
- [5] M. T. Ribeiro, S. Singh, and C. Guestrin. "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 1135–1144, 2016.
- [6] A. Shrikumar, P. Greenside, and A. Kundaje. Learning important features through propagating activation differences. *CoRR*, abs/1704.02685, 2017.