# Group Leakage Overestimates Performance: A Case Study in Keystroke Dynamics

Blaine Ayotte, Mahesh K. Banavar, Daqing Hou, and Stephanie Schuckers
Department of Electrical and Computer Engineering, Clarkson University *
8 Clarkson Ave, Potsdam, NY 13699
{ayottebj, mbanavar, dhou, sschucke} @clarkson.edu

## Abstract

*Keystroke dynamics is a powerful behavioral biometric capable of user authentication based on typing patterns. As larger keystroke datasets become available, machine learning and deep learning algorithms are becoming popular. Knowledge of every possible impostor is not known during training which means that keystroke dynamics is an open set recognition problem. Treating open set recognition problems as closed set (assuming samples from all impostors are present) can cause models to incur data leakage, which can provide unrealistic overestimates of performance. It is a common problem in machine learning and can cause models to report higher accuracies than would be expected in the real world. In this paper, we outline open set recognition and discuss how, if not handled properly, it can lead to data leakage. The performance of common machine learning methods, such as SVM and MLP are investigated with and without leakage to clearly demonstrate the differences in performance. A synthetic dataset and a publicly available keystroke dynamics fixed-text dataset are used for research transparency and reproducibility.*

## 1. Introduction

Keystroke dynamics is a behavioral biometric that can determine identity based on typing patterns [1, 2, 6, 24]. By requiring the typing patterns to match, in addition to existing measures, keystroke dynamics can be used to provide an additional layer of security to traditional user authentication systems. This additional layer does not require any additional hardware as most computers have a physical or virtual keyboard. Keystroke dynamics can have other applications such as narrowing down suspects in an online chat (identification, 1:n) or continuous user authentication (1:1, verification).

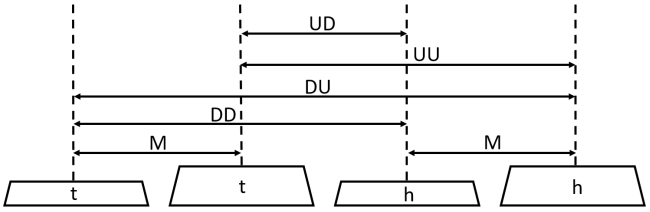



Figure 1. Graphical representation of how features can be extracted from two consecutive keystrokes. The digraphs (DD, DU, UD, UU) are also referred to as press-press, press-release, release-press, and release-release (see [3–5, 11]).

There are two main types of keystroke dynamics: fixed-text and free-text. Fixed-text requires the keystrokes of the test sample to exactly match with the keystrokes of the profile, making deploying machine learning models easier than for free-text. It is common practice to extract features such as durations of monographs and digraphs (hold time and flight time of key-presses associated with specific letter combinations as shown in Figure 1) from the keystrokes.

Now that larger datasets exist, and with recent computational advances, machine learning and deep learning techniques are becoming increasingly popular for keystroke dynamics. When using binary or multi-label classifiers (or any algorithm requiring impostor data) to learn the performance of keystroke dynamics, the research community has largely treated keystroke dynamics as a closed set recognition (CSR) problem rather than an open set recognition (OSR) problem. However, there has been work in keystroke dynamics where OSR has been handled perfectly and helped to benchmark existing algorithms [19]. Furthermore, some existing work in biometrics quantifies the accuracy of open set identification performance through the use of false positive identification rate (FPIR) and false negative identification rate (FNIR) [17, 23].

Closed set recognition, in this context, assumes full

*This work is supported in part by the NSF CPS award 1646542, Clarkson Niklas Ignite Fellowship, and material is based upon work supported by the Center for Identification Technology Research (CITeR) and the NSF under Grants 1650503 and 1314792.

knowledge of all impostors. Real-world keystroke dynamics systems are trying to authenticate a genuine user against impostors, and the majority of times there is no available knowledge of impostors before they attack. Many performance evaluations appear to have been performed under the assumption that the only impostors are contained in the dataset. Of course, for keystroke dynamic authentication systems we are actually more interested in the performance on unknown impostors rather than known impostors. If we knew which impostors would attack the genuine user, it would be much easier to develop a model to detect them.

Furthermore, by treating the open set recognition problem as a closed set recognition problem leakage can occur, which can cause researchers to falsely conclude performance is better than it actually is. The performance is measured against impostors that have been trained on, which will cause the performance to appear better than (or at least equal to) the performance on impostors not seen during training. Information about which impostor is attacking has been leaked into the model. Data leakage is defined as "The introduction of information about the target of a data mining problem that should not be legitimately available to mine from" [14]. In the case of keystroke dynamics, knowledge of all impostors is not available during training and by introducing all impostors to the model during training causes leakage to occur.

Deemed to be one of the top 10 data mining mistakes [20], data leakage can cause models to report strong performance on testing sets, but when deployed in the real world, the performance can be lackluster. Data Leakage is undesirable as a suboptimal model that overestimates the models performance may be learned [14]. As a result, it is important for researchers to be aware of data leakage and take appropriate action to not only avoid leakage but to also disclose how models are trained. For example, Kaufman *et al.*, point out that the majority of documented examples of data leakage occur in data mining competitions such as the Data Mining and Knowledge Discovery competition (KDD-Cup) [14]. While documented much less often in practical applications of data mining, leakage is no less likely to appear in the real world than in the competitions.

There are multiple ways of leaking information from the testing set into the training set for a machine learning model. These types of leakage include normalization leakage, group leakage, and augmentation leakage [14]. In this paper, we focus on group leakage, which not only has the largest overall effect on performance, but is commonly overlooked. The publicly available CMU fixed-text dataset is used as a well-known benchmark in addition to a synthetic dataset to demonstrate group leakage in machine learning models. The percent difference between the equal error rates (EER) for the leakage and leakage-free models was found to be 31.3% and 32.2% for the SVM and MLP algorithms, respectively.

The rest of this paper is organized as follows: The open set recognition framework is outlined in Section 2. In Section 3 the keystroke dynamics fixed-text dataset used is described and the two machine learning algorithms, support vector machine (SVM) and multilayer perceptron (MLP) algorithms are presented. Section 4 quantifies the effects of group leakage in keystroke dynamics, Section 5 discusses potential ways of avoiding leakage, and concluding remarks are made in Section 6.

## 2. Open Set Recognition

Open set recognition is defined as having incomplete knowledge of the world at training time [13, 21, 22]. In contrast, closed set recognition assumes that all classes are known at training time. The majority of machine learning algorithms are designed for the closed-set case, and far fewer work has been done to develop algorithms that are specifically designed for the open-set case.

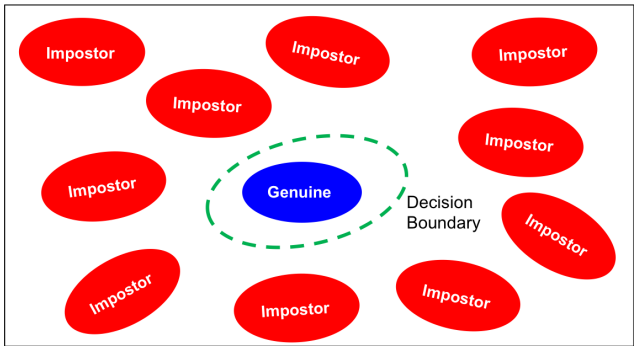
Most research involving open set recognition uses computer vision as an example, but open and closed set recognition can be generalized to any application. One example of open set recognition in computer vision is object detection. For object detection, the goal is to recognize a specified object of interest. Every other possible image that does not contain the object is a negative example. Of course, it is impossible to collect every possible negative example (one can see there would be an infinite number of them).

For keystroke dynamics, knowledge of one class, the genuine user, is known, but only a subset of all possible impostor classes are known at the time of training. Figure 2, in 2a, 2b, and 2c, shows the discrepancy between CSR and OSR with and without leakage. It is possible that there could be some keystroke dynamics application where knowledge of all impostors may be known beforehand, but we argue those applications would not be common. Therefore, keystroke dynamics is an OSR problem and needs to treated as such to properly measure expected performance.

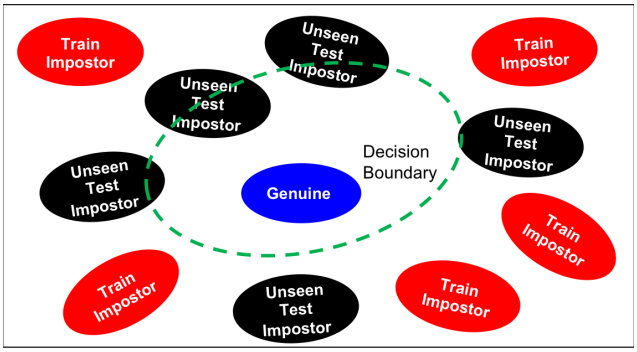
Open set recognition can be challenging especially when there are large numbers of classes missing or unavailable during training. Intuitively, it is expected that the more unavailable classes the more complex the problem is to solve. Researchers have tried to define the "openness" of a problem, defined in equation 1, in order to quantify how complex the problem will be [9, 21].

$$O = 1 - \sqrt{\frac{2 \times |C_{TR}|}{|C_{TR}| + |C_{TE}|}} \tag{1}$$

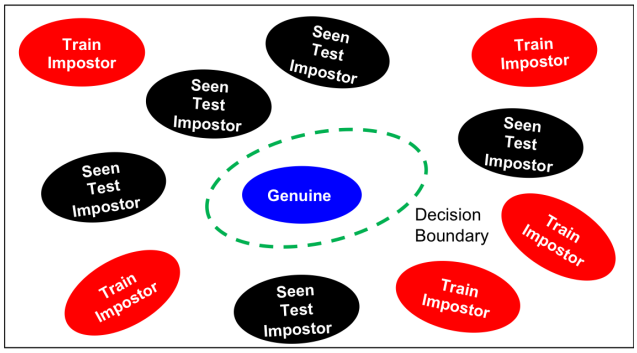
where $C_{TR}$ and $C_{TE}$ are the set of classes used in training and the set of classes used during testing. If the number of training and testing classes are equal then $O = 0$, which means the problem is 0% open (closed).




(a) CSR: Closed set recognition has access to all impostors and a decision boundary can be easily drawn around the genuine user.




(b) OSR without leakage: Open set recognition does not have access to all impostors at the time of training. It is important to evaluate performance in this scenario by holding out impostors during training to avoid incurring leakage. The model can be evaluated with unseen impostors to properly determine expected system performance.




(c) OSR with leakage: Open set recognition does not have access to all impostors at the time of training. Unlike the system in Figure 2b, this model is trained and tested with all of the impostors in the dataset. When this model is deployed in the real world, it will come across classes not seen before. As a result the system reports a stronger testing accuracy that is not representative of what will occur in the real world.

Figure 2. Graphical depiction of (a) a closed set recognition; (b) an open set recognition without leakage; and (c) an open set recognition with leakage. Leakage can lead to a significant loss in performance when systems trained and tested under leakage conditions are deployed in the real world. In this example, the openness, as defined in Equation (1), is $O = 0.16$ because there are 6 training classes and 11 testing classes.

In general, the less open a problem, the less complex it is and it will likely be easier to solve. As a problem becomes more open, less information is available and it will be harder to effectively generalize to the unseen classes. For keystroke dynamics, the theoretical total number of impostors classes, $|C_{TE}|$, is the total number of people you expect could launch an attack, which is often an enormous number (although a sub-population may be reduced to a similar cluster, thus effectively reducing the total number of classes that we have to consider). Training data from multitudes of different impostors is hard to collect so due to a lack of training data and classes the openness of a problem can rapidly approach 100%.

While the keystroke dynamics problem is very open, it is not necessarily essential to collect data from every possible impostor. Many impostors may have similar distributions and a reasonably sized subset may be adequate to train models effectively. However, in practice, knowing when you have adequate impostors can be near impossible and more data is always preferred. This phenomenon is discussed further in Section 4.1.

Mistreating the OSR problem using a closed set approach will result in group leakage. Group leakage causes performance results to be higher during testing than can be expected in deployment. For some applications misreported error rates could be the difference between life and death. Additionally, this can be detrimental to researchers by slowing the advancement of the field. Not only is the reported performance misleading, but, in many situations, the leakage model is overfitting to particular impostors rather than learning the differences between the genuine and impostor typing patterns. Group leakage and related problems are discussed in more detail in Section 4.

## 3. Dataset and Algorithms

To ensure reproducibility of our results, the CMU fixed-text dataset is used to demonstrate group leakage [15]. This dataset was collected to study password hardening and consists of 51 users, each with 400 total password entries across 8 different sessions. All users were required to type ".tie5Roanl" without any errors. The dataset consists of 31 features including monographs, DD digraphs, and UD digraphs. Monographs are defined as the hold time of a key, DD digraphs are the elapsed time between the key-down of a key to the key-down of the following key, and UD digraphs are the time of a key released to the press of the following key (see Figure 1) [3–5, 11]. The CMU dataset is one of the largest publicly available fixed-text datasets and is used by many different researchers. Therefore, it is perfect to demonstrate the hazards of data leakage.

Our goal is not to advance the state-of-the-art in keystroke dynamics algorithms, but instead to reveal the impacts of group leakage on performance. Therefore, we use

two common machine learning algorithms, support vector machine (SVM) and multilayer perceptron (MLP), to evaluate the effects of data leakage on fixed-text keystroke dynamics. SVM is a supervised machine learning technique that constructs a hyperplane between classes in a high dimensional space [7, 8]. We use an rbf kernel with parameters $C = 10$ and $\gamma = 0.01$. Neural networks are another common supervised machine learning technique inspired by the human brain [7, 10]. Our neural network structure is based off existing works for keystroke dynamics [18, 25] and can be seen in Figure 3.

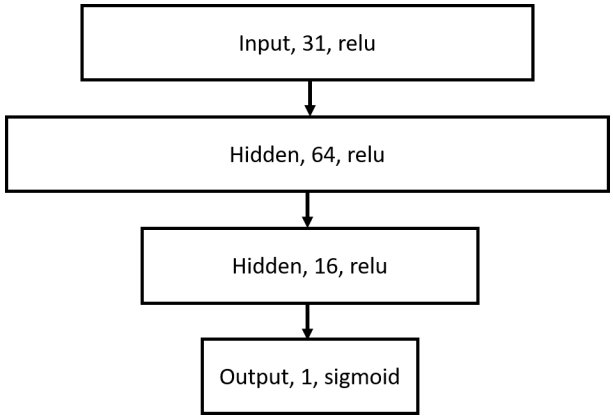



Figure 3. Network architecture for the multilayer perceptron. There are two hidden layers with 64 and 16 neurons followed by dropout layers of 0.25 and 0.5, respectively. Both have a relu activation function and L2 regularization parameters of 0.99 and 0.05 respectively.

## 4. Quantifying Group Leakage

In the context of keystroke dynamics, group leakage will occur when an open set recognition problem is treated as a closed set recognition problem. Under OSR, it assumed there is incomplete knowledge of every impostor. The model should be trained with a subset of impostor data and evaluated with unseen impostors to obtain an accurate measure of how well the model can be expected to perform in deployment. Group leakage occurs when an impostor in the testing set is also present in the training set. The exact typing samples are not necessarily shared between training and testing sets, but often samples from the same impostor are highly correlated. Therefore, if an impostor's data is in the training set, all other data from that user should be excluded from the testing set. This process ensures that no group leakage will occur.

A majority of keystroke dynamics systems are OSR problems and assume no knowledge of impostors. Therefore, attacker keystrokes are unknown. In this case, training the model with impostors that also appear in the testing set gives the model information about those impostors (leakage) that it should not have. In this case, the model may be biased by the keystroke patterns of the impostors in a way that will not generalize well to other unseen impostors. On the other hand, we can also envision keystroke dynamics systems that authenticate a user while knowing all the possible impostors (CSR). While this scenario may be rare, no leakage has occurred and it is perfectly valid to train the model with all impostors. The case of known attackers is almost never present in the real world but is still an important distinction to make.

Group leakage is dangerous because it will almost always result in higher testing accuracy compared to the real-world performance. However, group leakage may have very little impact if the data is sufficiently large or representative of the entire population. Unfortunately, in practice it is almost impossible to know when there is enough data. Therefore, it is still preferred to avoid group leakage entirely.

To better illustrate the performance difference group leakage can cause, a simple synthetic dataset is introduced and fit with an SVM classifier. The effects of group leakage are then demonstrated on a well known fixed-text dataset [15] using two commonly used machine learning algorithms: support vector machine (SVM) and multilayer perceptron (MLP).

### 4.1. Synthetic Example

The artificial dataset exists in two dimensions with the genuine user's samples centered around $(0, 0)$, generated with unit variance. Data for each impostor are generated centered at a random (but fixed) angle on a circle of radius four; the distribution of the features has unit variance. The dataset can be seen in Figure 4, where the genuine user can be seen plotted against 1, 3, 5, and 50 impostors respectively. With enough impostors, the synthetic data will take on the form of a donut with the genuine user in the center.

In this simulated experiment the dataset consists of 151 impostors and one genuine user. In order to obtain deployment performance, 100 impostors are set aside. The leakage and leakage-free models will be trained using the genuine user and 51 impostors. Therefore, the openness, as defined in Equation (1), is $O = 0.29$ because there are 51 training classes and 151 testing classes. From Figure 4, it is easy to tell that if only one impostor is used to train the model, it will not be able to learn the underlying nature of the data (a donut shape). However, if a sufficient number of impostors are used to train the model, the model can easily learn the donut-type nature of the data.

We now illustrate the effects of leakage using SVMs on this data. Two cases are considered when training the models, with leakage and leakage-free. The leakage case randomly splits the data from all available impostors (CSR), while the leakage-free case uses data from two thirds of the impostors for training and data from one third of the im-

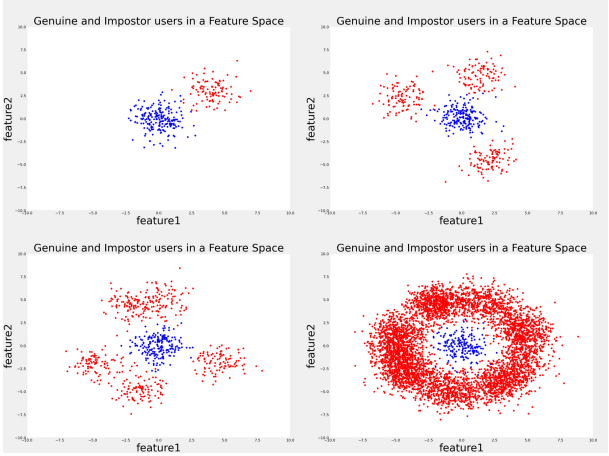



Figure 4. Illustration of the synthetic experiment. For simplicity, there are only two features. The genuine user is centered at $(0, 0)$ with unit variance. The impostors fall on a circle with radius four, each at a different angle, and have unit variance. The genuine user is plotted against 1, 3, 5, and 50 impostors, respectively. The openness, as defined in Equation (1), is $O = 0.29$ because there are 51 training classes and 151 testing classes.

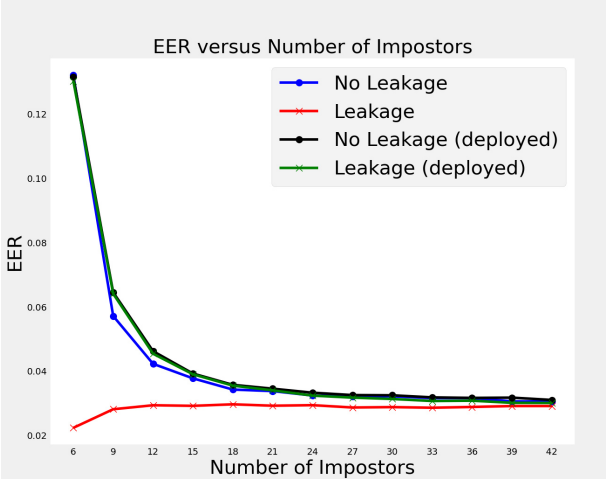



Figure 5. EER versus number of impostors for the leakage and leakage-free cases on the synthetic dataset. As more impostors are added to the system, the leakage EER increases, while the leakage-free EER decreases. The deployment performance of each model, determined with additional unseen impostors, is also shown.

postors for testing (OSR). For the leakage-free case, if data from a user is in the training set, that user will not be used for testing, and vice-versa.

The deployment performance of each model is plotted to show the performance of the model in the real world. The deployment performance is determined using the leakage and leakage-free models trained with a certain number of impostors, but tested with the 100 impostors set aside before training the models. These impostors are unseen by both models during training and testing and can be considered real world impostors attacking the genuine user. The models are trained 1,000 times with different genuine and impostor data and the average accuracies are reported. A plot of EER versus number of impostors is shown in Figure 5. Notice that in a synthetic example it is easy to show the true in-the-wild performance using these 100 unseen impostors, but with real-world data this is usually impossible without a follow-up study or additional data.

From Figure 5, looking at the leakage and no leakage curves, it can be seen that as more impostors are added to the system, the EER for the model with leakage increases. However, as more impostors are added, the model is actually fitting the donut shape of the data better. This is not surprising because for a closed set recognition problem adding additional classes often reduces the overall accuracy. Group leakage can be especially dangerous when working with small amounts of data as it can be tempting to conclude a model has excellent performance, when in fact, it is overfit to the specific impostors used during training. As a result of this overfitting, when deployed, this model will likely perform worse than expected.

Initially, the model trained without leakage (OSR) performs worse than the model with leakage (CSR). This is representative of the fact that more data is needed to learn the optimal donut shape boundary. The leakage-free EER decreases as the number of of impostors are added and researchers can correctly conclude that more data will improve this model. When deployed, this model should behave as expected. As we add impostors, the leakage and leakage-free cases tend to converge to the same performance. Once there is plenty of data, leakage is no longer an issue. Knowing when you are at this point is difficult to determine with complex real-world data, but can be easy to spot in this synthetic example.

Figure 5, in the black and green lines, also shows the deployment performance of both the leakage and leakage-free models. The deployment performance illustrates the in-the-wild performance. The leakage and leakage-free models in deployment are very similar to each other, and to the leakage-free model's testing performance. The leakage model appears to perform very well during testing, but in deployment the performance is significantly worse. On the other hand, the leakage-free model in testing performs almost identically to the leakage-free model in deployment. This further highlights the importance of properly handling OSR problems and avoiding group leakage.

### 4.2. Real-world Example

For the real-world CMU dataset, the effects of group leakage are demonstrated through differences in the EER to be more comparable with previous keystroke dynamics works. For the leakage case, data is treated as a CSR problem and randomly partitioned into training and testing with-

out considering which impostor the data was drawn from. Leakage is present because OSR guidelines were not followed and impostors that are in the training set can also be in the testing set. The leakage-free case follows OSR guidelines and ensures if data from a user is in the training set, that user will not be used for testing, and vice-versa.

For training in the leakage and leakage-free cases, 300 password attempts from the genuine user and 300 impostor attempts are used. Testing is done with the remaining 100 genuine samples and 100 impostor samples. Each experiment is repeated 50 times using different random subsets of the data for training and testing to ensure representative results. For the results in Table 1, the leakage-free case holds out 20% of the 50 impostors so that 10 unseen impostors are in the testing set and 40 impostors are seen during training. The percentage of impostors for training is increased from 67% (Figure 6) to 80% (Table 1) to match the generally recommended 80/20 split (Pareto principle [16]). Training and testing with 40 impostors and 10 impostors gives 41 training classes and 51 testing classes resulting in an openness value of $O = 0.06$. The leakage case contains training and testing data from all impostors.

Table 1. EERs with standard deviations for the SVM and MLP algorithms with and without group leakage. For the leakage-free case, 20% of impostors are held so that 10 unseen impostors are in the testing set and 40 impostors are seen during training. When training the model with leakage, impostors in the training set can also appear in the testing set. The openness, as defined in Equation (1), for this scenario is $O = 0.06$

| **Algorithm** | **Sampling Method** | | |
|---|---|---|---|
| | **Leakage** | **No Leakage** | **Difference** |
| **SVM** | $3.983 \pm 0.159$ | $5.461 \pm 0.340$ | 31.3% |
| **MLP** | $3.544 \pm 0.157$ | $4.903 \pm 0.387$ | 32.2% |

The differences in EER between the leakage and leakage-free case range from 31% to 33% demonstrating group leakage can have a significant impact on performance. An independent two-sample t-test [12] is used to determine if the differences in performance are significant. For the SVM algorithm and MLP algorithms the t-scores were 27.8 and 23.0. The probability of getting those results if the leakage and leakage-free EERs were equal, for both algorithms, was less than 0.0001. Therefore, we can conclude the difference in performance is significant and that leakage has an impact on performance measures.

To illustrate this effect further, Figure 6 shows the SVM algorithm's average EER versus the number of impostors. The SVM algorithm is chosen over the MLP as it is simpler and was slightly less affected by group leakage. The leakage case trains and tests with data from all impostors. For the leakage-free case, two thirds of the impostors are used for training and the remaining one third of unseen impostors are used for testing. The 67/33 split is chosen instead of 80/20 to allow for a data point every three impostors.

As more impostors are added to the system, the leakage EER increases, while the leakage-free EER decreases. This is consistent with the trends in the synthetic example. This means that the leakage model is overfitting to particular impostors rather than learning the differences between the genuine and impostor typing patterns.

For the synthetic data, after about 30 impostors the leakage and leakage-free EER converged (see Figure 5). When sufficient numbers of impostors are used for the real-world data, the leakage and leakage-free performances will converge as well. However, this real-world data is more complex than the synthetic example and it is not possible to tell exactly how much data is adequate for leakage to become negligible. Looking at Figure 6, at least 100 impostors, if not 250 or more, would be needed for the leakage and leakage-free cases to converge. Note that this convergence point might imply that no more impostors are needed, but it is possible that all impostors came from a university background and that their typing patterns may still not be completely representative of the entire world. This further illustrates the challenge of dealing with OSR problems.

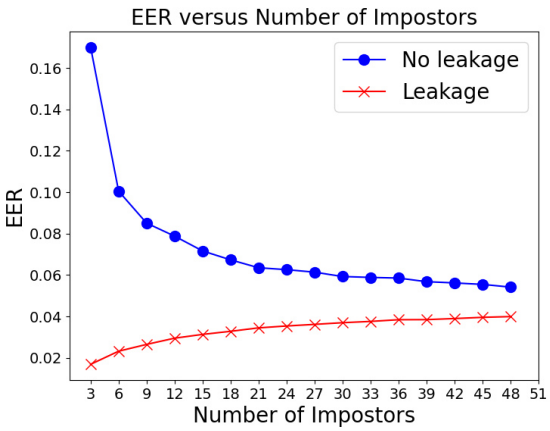



Figure 6. EER versus number of impostors for the SVM algorithm and the CMU dataset. As the number of impostors increases, the leakage ERR increases while the leakage-free EER decreases.

## 5. Avoiding Leakage

As we have shown throughout the paper, group leakage can have a serious impact on performance. There can be a substantial difference in performance between a model with and without leakage. For certain applications, a slight difference between expected performance and actual performance when deployed, could have serious ramifications. Additionally, the overly optimistic results may lead to non-optimal models being deployed, when a model trained without leakage would give actual better performance.

In this section, we present guidelines that can be used

to prevent types of leakage that are common to keystroke dynamics. This is not a comprehensive list, but is intended to be useful for the keystroke dynamics and the broader behavioral biometrics community.

First of all, it is important to understand if the problem at hand is CSR or OSR. When working CSR problems, group leakage will not be relevant. However, when working with OSR problems, care should be taken to ensure the same impostors do not appear in both the training and testing sets. This ensures that the performance of a model during training and testing will be consistent (as much as possible) with the deployment performance. Eliminating group leakage also helps to prevent models from memorizing individual impostor typing patterns, promotes learning the differences in how people type, and can deal with new data from unknown users in a more effective manner.

After accounting for OSR, the best way to avoid other types of data leakage is to separate the data into training and testing sets before doing anything else. This can help to eliminate both normalization and augmentation leakage. These types of leakage occur because information from unseen data leaks into the model before it is evaluated with that data. Therefore, to avoid these types of leakage, the testing set should be untouched until testing time. Scaling features after training and testing separation can help avoid accidentally introducing information from the testing set to the model during training. Similarly, generating synthetic data using only the training data ensures information about the distribution of the testing set data remains unseen.

## 6. Conclusion and Future Work

We have shown that keystroke dynamics is an open set recognition problem. Open set recognition problems have incomplete knowledge of the world at training time and we have demonstrated that, when not handled properly, can incur group leakage. Furthermore, through synthetic and real-world keystroke dynamics data, we have shown that group leakage can cause a significant difference in performance between testing and deployment. This performance difference, depending on the application, can have catastrophic impacts and even lead to the selection of sub-optimal models. Lastly, methods of avoidance are presented to help researchers reduce leakage in their machine learning models.

Future work includes studying the impact of different types of leakage as well as on different types of data. We have demonstrated the effects of group leakage for keystroke dynamics, but this work can be expanded to not only other forms of leakage, but also other research areas well. Additional experiments using different datasets and algorithms are needed to better explain and characterize data leakage. Raising awareness about OSR and data leakage is important so that researchers can provide reasonable performance estimates and keep improving performance within their respective fields.

## References


[1] Alejandro Acien, Aythami Morales, John V Monaco, Ruben Vera-Rodriguez, and Julian Fierrez. TypeNet: Deep learning keystroke biometrics. *arXiv preprint arXiv:2101.05570*, 2021. 1

[2] Arwa Alsultan and Kevin Warwick. Keystroke dynamics authentication: a survey of free-text methods. *International Journal of Computer Science Issues (IJCSI)*, 10(4), 2013. 1

[3] Blaine Ayotte, Mahesh Banavar, Daqing Hou, and Stephanie Schuckers. Fast free-text authentication via instance-based keystroke dynamics. *IEEE Transactions on Biometrics, Behavior, and Identity Science*, 2(4):377–387, 2020. 1, 3

[4] Blaine Ayotte, Mahesh K Banavar, Daqing Hou, and Stephanie Schuckers. Fast and accurate continuous user authentication by fusion of instance-based, free-text keystroke dynamics. In *International Conference of the Biometrics Special Interest Group (BIOSIG). IEEE*, 2019. 1, 3

[5] Blaine Ayotte, Jiaju Huang, Mahesh K Banavar, Daqing Hou, and Stephanie Schuckers. Fast continuous user authentication using distance metric fusion of free-text keystroke data. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019. (Accepted). 1, 3

[6] Salil P Banerjee and Damon L Woodard. Biometric authentication and identification using keystroke dynamics: A survey. *Journal of Pattern Recognition Research*, 7(1):116–139, 2012. 1

[7] Steven L Brunton and J Nathan Kutz. *Data-driven science and engineering: Machine learning, dynamical systems, and control*. Cambridge University Press, 2019. 4

[8] Christopher JC Burges. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2):121–167, 1998. 4

[9] Chuanxing Geng, Sheng-jun Huang, and Songcan Chen. Recent advances in open set recognition: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2020. 2

[10] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016. 4

[11] Athul Harilal, Flavio Toffalini, John Castellanos, Juan Guarnizo, Ivan Homoliak, and Martín Ochoa. TWOS: A dataset of malicious insider threat behavior based on a gamified competition. In *Proceedings of the International Workshop on Managing Insider Security Threats*, pages 45–56. ACM, 2017. 1, 3

[12] Anthony J Hayter. *Probability and statistics for engineers and scientists*. Nelson Education, 2012. 6

[13] Lalit P Jain, Walter J Scheirer, and Terrance E Boult. Multi-class open set recognition using probability of inclusion. In *European Conference on Computer Vision*, pages 393–409. Springer, 2014. 2

[14] Shachar Kaufman, Saharon Rosset, Claudia Perlich, and Ori Stitelman. Leakage in data mining: Formulation, detection, and avoidance. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 6(4):1–21, 2012. 2

[15] Kevin S Killourhy and Roy A Maxion. Comparing anomaly-detection algorithms for keystroke dynamics. In *2009 IEEE/IFIP International Conference on Dependable Systems & Networks*, pages 125–134. IEEE, 2009. 3, 4

[16] Richard Koch. *The 80/20 Principle: The Secret of Achieving More with Less: Updated 20th anniversary edition of the productivity and business classic*. Hachette UK, 2011. 6

[17] Ajay Kumar and Chenye Wu. Automated human identification using ear imaging. *Pattern Recognition*, 45(3):956–968, 2012. 1

[18] Saket Maheshwary, Soumyajit Ganguly, and Vikram Pudi. Deep secure: A fast and simple neural network based approach for user authentication and identification via keystroke dynamics. In *IWAISe: First International Workshop on Artificial Intelligence in Security*, page 59, 2017. 4

[19] Aythami Morales, Julian Fierrez, Ruben Tolosana, Javier Ortega-Garcia, Javier Galbally, Marta Gomez-Barrero, André Anjos, and Sebastien Marcel. Keystroke biometrics ongoing competition. *IEEE Access*, 4:7736–7746, 2016. 1

[20] Robert Nisbet, John Elder, and Gary Miner. *Handbook of statistical analysis and data mining applications*. Academic Press, 2009. 2

[21] Walter J Scheirer, Anderson de Rezende Rocha, Archana Sapkota, and Terrance E Boult. Toward open set recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(7):1757–1772, 2012. 2

[22] Walter J Scheirer, Lalit P Jain, and Terrance E Boult. Probability models for open set recognition. *IEEE transactions on pattern analysis and machine intelligence*, 36(11):2317–2324, 2014. 2

[23] Elham Tabassi, Craig Watson, Gregory Fiumara, Wayne Salamon, Patricia Flanagan, and Su Lan Cheng. Performance evaluation of fingerprint open-set identification algorithms. In *IEEE International Joint Conference on Biometrics*, pages 1–8. IEEE, 2014. 1

[24] Pin Shen Teh, Andrew Beng Jin Teoh, and Shigang Yue. A survey of keystroke dynamics biometrics. *The Scientific World Journal*, 2013, 2013. 1

[25] Yasin Uzun and Kemal Bicakci. A second look at the performance of neural networks for keystroke dynamics using a publicly available dataset. *Computers & Security*, 31(5):717–726, 2012. 4