Exploring Anomaly Detection in Credit Card Transactions with Autoencoders

Harry Denell

December 9, 2024

1 Introduction

Detecting credit card fraud poses significant challenges due to the rarity of fraudulent transactions and the resulting imbalance in datasets. Traditional supervised approaches often struggle in these scenarios, relying heavily on labeled data that is costly and difficult to obtain. This study explores the use of autoencoders for anomaly detection. The goal is to have the autoencoder capture patterns in legitimate transactions and flag deviations as potential fraud. Additionally, the autoencoder's performance is compared against a Support Vector Machine baseline, providing insights into the strengths and limitations of each approach.

2 Background

2.1 Anomaly Detection

An anomaly, or outlier, is an observation that deviates significantly from normal, often described as unusual, unexpected, or rare [1]. Anomaly detection, also known as outlier- or novelty detection, is the process of identifying anomalies, meaning data points that deviate significantly from the majority of the dataset [2, 3]. Anomaly detection can be of great value, as anomalies in data frequently indicate important or actionable information [3]. But anomaly detection can poses significant challenges. Identifying a clear and reliable distinction between normal and anomalous events is challenging [1]. Variability within normal data can lead to two common errors, misclassifying normal samples as anomalous (type I error) or failing to detect anomalous events (type II error) [1]. Furthermore, anomalous events are often rare compared to normal occurrences, leading to highly imbalanced datasets [2]. The challenge is further compounded in many cases by the lack of labeled data, making it unclear which instances are anomalies or why they are classified as such [1]. Acquiring labeled data is often extremely costly and resource intensive [3]. As a result, anomaly detection typically becomes an unsupervised learning problem [1].

2.2 Credit Card Fraud

Credit card fraud can be described as the unauthorized use of another person's credit card information for purchases or withdrawals, and is a problem that can lead to substantial financial losses for both individuals and organizations [4, 5]. Overall, credit card fraud poses a significant problem to financial institutions and consumers worldwide. Global card fraud losses have been on an upward trajectory, reaching approximately $32 billion in 2021 [6]. This trend is expected to continue, with projections indicating that losses could grow by more than $10 billion between 2022 and 2028 [6].

Defining a precise boundary between normal and anomalous behavior is challenging due to ambiguity in what actually constitute a normal activity [3]. Moreover, normal behavior might be dynamic, meaning that what is considered normal at one point may not be in the future [3]. Furthermore, fraud detection should be cost effective, which means balancing detection expenses against potential losses [7]. Finally, the unavailability of accessible datasets poses another challenge, as credit card companies generally cannot share data publicly due to confidentiality, privacy, and legal concerns [7].

2.3 Deep learning techniques for Anomaly Detection

Anomaly detection has been studied in many different fields, with methods broadly categorized into supervised, unsupervised, and semi-supervised learning [1]. Shallow models like Support Vector Machines (SVMs) can perform well on low-dimensional data but can struggle in high-dimensional settings [1]. Additionally, supervised approaches face limitations in anomaly detection due to the scarcity of labeled anomalies [1]. In contrast, self-supervised learning has emerged as a promising alternative, leveraging unlabeled data to identify deviations from normal patterns [1]. Deep learning techniques are increasingly favored for their ability to handle complex datasets [1]. Anomaly detection often involves identifying deviations from patterns learned from normal data. Reconstruction-based methods train models to encode and decode data, minimizing reconstruction error [1]. With primarily normal data for training, typical instances yield low errors, while anomalies show higher errors due to deviations from learned patterns [1].

An autoencoder (AE) is an example of such a neural networks, designed for unsupervised learning of compact representations of data by mapping inputs to a lower-dimensional latent space (encoding) and reconstructing them back to their original form (decoding) [2]. The network is trained to minimize reconstruction loss, with the goal of having the output closely resemble the input [8, 2]. In anomaly detection, AEs aims to learn a normal data distribution by focusing on dominant patterns and discarding less relevant details using a bottleneck architecture [2]. This results in anomalies, which should deviate from normal patterns, being reconstructed poorly and therefore exhibiting high reconstruction errors [2, 1]. Their ability to extract meaningful features and highlight deviations makes AEs widely applicable for identifying anomalies in complex datasets [2].

2.4 Explainability in Anomaly Detection

Explainability plays an important role in anomaly detection. This also applies to credit card fraud detection, where its imperative for organizations to foster trust among stakeholders and ensuring ethical decision-making [9]. Regulatory frameworks like the EU's GDPR further underscore the importance of explainability [9]. Explainability tools, such

as Local Interpretable Model-agnostic Explanations (LIME), enable complex models to be interpreted more clearly, helping to explain why a transaction was flagged as fraudulent. LIME works by perturbing the input data, observing the model's responses, and using these observations to train an interpretable model that mimics the black-box model's behavior near the input of interest [10]. Importantly, LIME does not aim to explain the global behavior of the black-box model. Instead, it focuses on providing a local explanation for individual predictions [10].

3 Methodology

3.1 Data Acquisition

The Kaggle credit card transaction dataset[1] contains 284,807 transactions from European cardholders over two days in September 2013, with 492 labeled as fraudulent (0.172%). It includes 28 anonymized PCA-derived variables ($V1$ to $V28$), and two untransformed features: 'Time' (seconds since the first transaction) and 'Amount' (transaction value). The 'Class' variable indicates if a transaction is fraudulent or legitimate.

3.2 Data Exploration

To get a sense for the dataset its shape was visualized in tabular form. Table 1 shows the first few rows in the dataset.

Time	V1	V2	...	V28	Amount	Class
0	-1.359807	-0.072781	...	-0.021053	149.62	0
0	1.191857	0.266151	...	0.014724	2.69	0
1	-1.358354	-1.340163	...	-0.059752	378.66	0

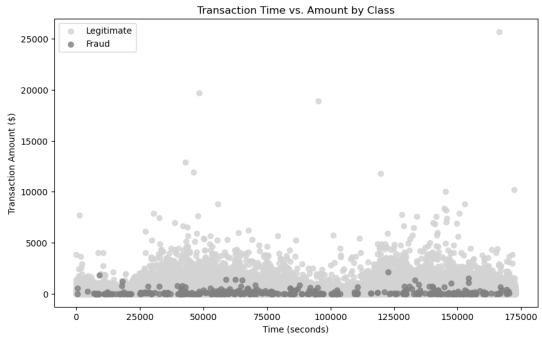
Table 1: Credit Card Fraud Dataset values

To better understand the dataset, an initial step in data exploration focused on investigating the time feature. A simple plot was generated to visualize the relationship between transaction time and amount for the two classes. Figure 1a illustrates this relationship, where we observe a lack of clear separation between the two classes when plotted with time on the x-axis.

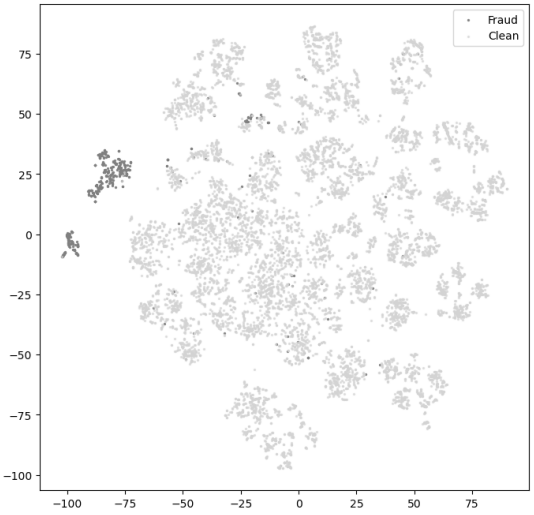
Furthermore, an analysis to determine wheter the classes could potentially be distinguished based on their features was done. t-SNE [11], a technique for visualizing high-dimensional data, was used. This method tries to preserve the local structure of data while revealing global patterns, such as clusters [11].

Before applying t-SNE it is often beneficial to perform dimensionality reduction using PCA, especially for dense datasets [12]. In our analysis, as PCA had already been applied to the dataset, t-SNE was directly applied. From the results of the t-SNE analysis, it appeared feasible to distinguish between fraudulent and normal transactions, as illustrated in Figure 1b. Fraudulent transactions were represented as distinct points, forming a cluster that is visually separable from the cluster of normal transactions.

[1] `https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud`



(a) Scatter plot of transaction time vs. amount for both classes.



(b) t-SNE visualization of the dataset showing fraudulent and normal transactions.

Figure 1: Transaction time vs. amount and t-SNE visualization of the dataset.

3.3 Data Preprocessing

From the data exploration, it was concluded that time did not seem to appear to be a significant feature for detecting anomalies in this dataset, so it was dropped. To reduce the skewness in the amount feature, a log transformation was applied to it. The dataset was divided into two subsets: fraudulent and clean transactions. The clean transactions were shuffled and split into training (80%) and validation (20%) sets. All fraudulent transactions were included in the test set to ensure that every fraud case was accounted for during evaluation.

Normalization is a common preprocessing step in neural network training, scaling inputs to zero mean and unit variance to improve learning efficiency and model stability [13]. A preprocessing pipeline was implemented. This pipeline was fitted on the training set and also applied to the validation and test sets. In figure 2, we observe features $V1$ and $V2$ before preprocessing.

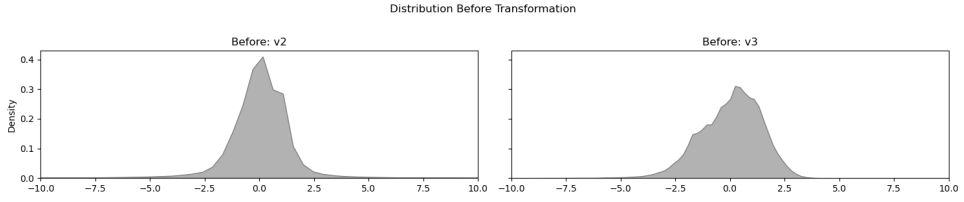


Figure 2: Features $V1$ and $V2$ before applying standardization and normalization.

After applying the preprocessing pipeline, the transformed features, shown in figure 3, were normalized and scaled. Notably, the variance observed in $V1$ and $V2$ were reduced, and the features were more uniformly distributed.

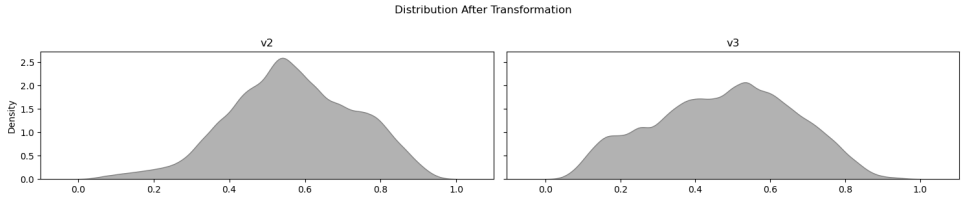


Figure 3: Features $V1$ and $V2$ after applying standardization and normalization.

3.4 Autoencoder

3.4.1 Model Architecture

A symmetric network architecture was chosen to ensure the encoder and decoder are well-balanced, facilitating effective learning. After experimenting with the number of layers and activation functions, it was decided to use five fully connected layers in the encoder and an equivalent number in the decoder. ReLU activations were used for all layers except the output layer, as detailed in Table 2.

Table 2: Autoencoder architecture

Layer	Number of Neurons	Activation Function
Input	29	-
Hidden Layer 1	16	ReLU
Hidden Layer 2	8	ReLU
Hidden Layer 3	4	ReLU
Bottleneck Layer	2	ReLU
Hidden Layer 4	4	ReLU
Hidden Layer 5	8	ReLU
Hidden Layer 6	16	ReLU
Output	29	Linear

3.4.2 Model training

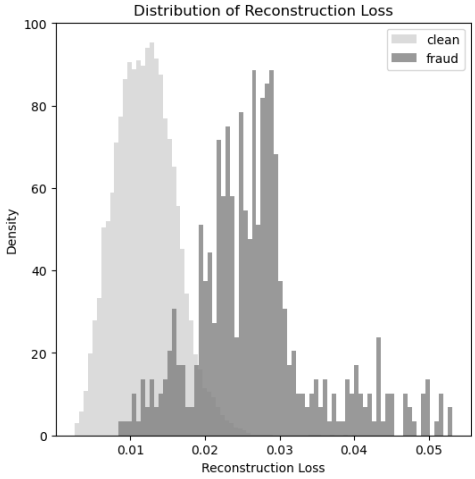
The model was trained to minimize reconstruction error, measured by mean squared error, using the Adam optimizer. The model was trained exclusively on normal transactions in hope that it would capture the underlying patterns of legitimate transactions. Early stopping was implemented based on the validation loss, with training stopping after 10 consecutive epochs without improvement to prevent overfitting. A batch size of 256 was used, and training was conducted for up to 1000 epochs. The best performing model weights, as determined by the minimum validation loss, were saved during training using checkpointing.

3.4.3 Model evaluation

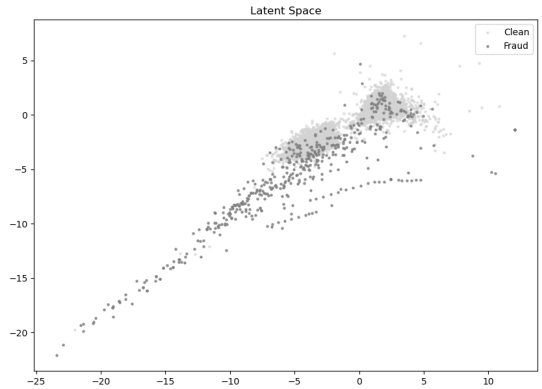
Post-training, the autoencoder's performance was evaluated on the validation set by calculating reconstruction errors as the mean squared difference between the original and reconstructed features. To analyze the model's ability to distinguish between legitimate and fraudulent transactions, reconstruction losses were separated into two distributions: one for legitimate transactions (clean) and one for fraudulent transactions. Figure 4a

shows the density distributions of these losses. Overall higher reconstruction losses are observed for fraudulent transactions compared to legitimate ones. This indicates the autoencoder's potential effectiveness in identifying anomalies, as it appears to struggle with reconstructing patterns not encountered during training.

To gain insight into the model's ability to distinguish between classes, the latent space of the autoencoder was visualized. The encoder's bottleneck layer was used to project the data into a latent space for analysis. As shown in figure 4b the majority of the legitimate samples form a cohesive cluster, while most fraud samples diverge from this, possibly indicating that the autoencoder captures distinct patterns associated with anomalous behavior.



(a) Density distributions of reconstruction loss for legitimate (clean) and fraudulent transactions



(b) Latent space visualization for clean and fraudulent data

Figure 4: Reconstruction loss distributions and latent space visualization for legitimate and fraudulent transactions.

To separate normal transactions from potential fraud, a threshold value was needed to classify reconstruction errors as either inliers or outliers. A threshold value was determined using the Median Absolute Deviation (MAD) method. MAD calculates the median of absolute deviations from the data's median, providing a reliable measure of scale for anomaly detection [14]. A threshold of 2.5 was applied to the MAD-based scores to identify outliers, representing potential fraudulent transactions.

Performance metrics, including precision and recall, were calculated to assess the model's accuracy in detecting fraudulent transactions. The classification performance is detailed in the confusion matrix, see table 3.

	Predicted Clean	Predicted Fraud
Actual Clean	56364	499
Actual Fraud	135	357

Table 3: Confusion matrix for classification results of the autoencoder on validation set

We conclude that while the model demonstrates strong recall (72.56%), successfully detecting the majority of fraudulent transactions, its precision is relatively low (41.71%).

This indicates that a large proportion of transactions flagged as fraudulent are false positives. Moreover to evaluate the model's ability to distinguish between legitimate and fraudulent transactions, a Precision-Recall Curve (PRC) was generated, and the Area Under the Precision-Recall Curve (AUPRC) was calculated. As shown in Figure 5b, the autoencoder achieved an AUPRC value of 0.6326, indicating a moderate ability to balance precision and recall.

3.5 Support Vector Machine

To establish a clear benchmark against which to compare our autoencoder-based anomaly detection model, we implemented a Support Vector Machine (SVM) classifier. The same preprocessing pipeline to the data as was used for the autoencoder, ensuring that both models were given same feature space.

3.5.1 Model architecture

For the baseline SVM, Radial Basis Function (RBF) was selected as kernel, with the intentt that the RBF kernel should be able to capture non-linear decision boundaries. Additionally, the SVM's probability parameter was enabled, allowing generation of probability estimates of the class estimate. This facilitates evaluation using metrics such as the Precision-Recall curve, aligning the baseline's evaluation with that used for the autoencoder. To keep the baseline simple, we retained default hyperparameters. By avoiding hyperparameter tuning, the SVM hopefully served as a rather neutral benchmark.

3.5.2 Model evaluation

Similar evaluation as for the autoencoder were performed to evaluate the SVM classifier's performance. A confusion matrix and key metrics such as precision and recall were computed. The confusion matrix is seen in table 4. We conclude that the SVM classifier achieved a high recall (82%), effectively identifying most fraudulent transactions. However, its precision is low (32%), indicating that a substantial number of transactions flagged as fraudulent are actually legitimate, leading to a high false positive rate.

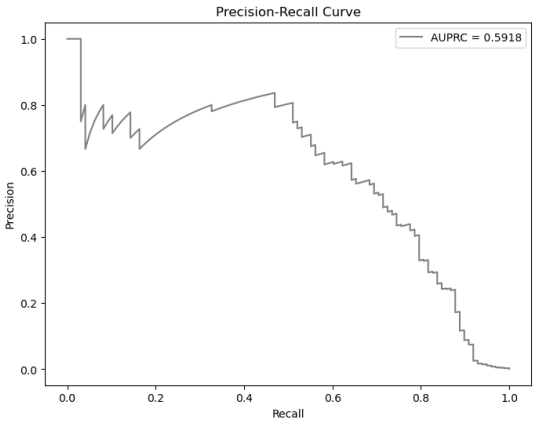
	Predicted Normal	Predicted Fraud
Actual Normal	56695	169
Actual Fraud	18	80

Table 4: Confusion matrix for the SVM classifier.

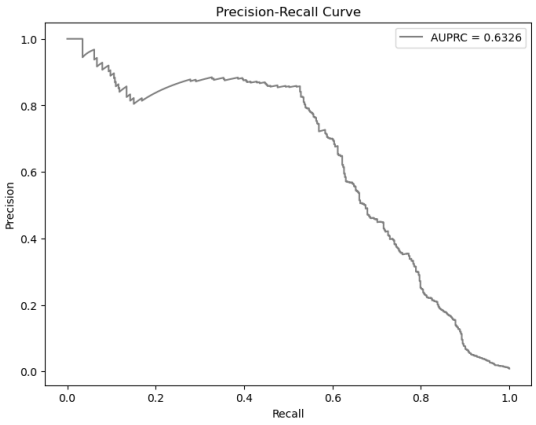
After this, AUPRC was plotted. This can be seen in figure 5a. We observe that the SVM achieved an AUPRC value of 0.5918.

3.6 Explainability integration

To gain insights into the autoencoder and the features contributing to the reconstruction loss of anomalous datapoints, LIME was used. It was applied on the two datapoints with the highest reconstruction errors. The analysis with LIME identified some features influencing the autoencoder's reconstruction errors, as shown in figure 6. For both the most and second most anomalous datapoints, $v4 > 0.60$ had the highest positive contribution,



(a) Precision-Recall Curve for the SVM classifier



(b) Precision-Recall Curve for the autoencoder

Figure 5: Comparison of Precision-Recall Curves for the autoencoder and SVM

followed by $v10 \leq 0.33$ which also contributed positively in both cases. Negative contributions were consistent across both datapoints for $v1 \leq 0.40$ and $v17 \leq 0.43$. $v11$ exhibited instance-specific behavior, contributing positively for the most anomalous datapoint but having minimal impact on the second.

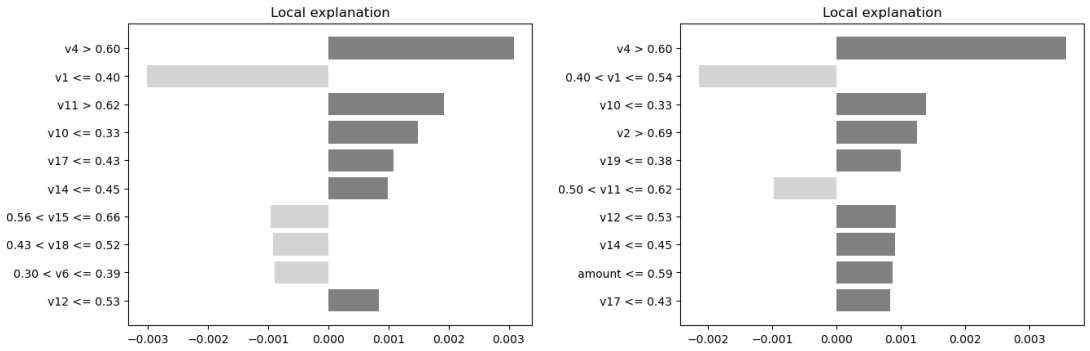


Figure 6: LIME explanations for the most anomalous (left) and second-most anomalous (right) datapoints

4 Discussion

4.1 Performance Comparison

The autoencoder leveraged reconstruction errors to distinguish between legitimate and fraudulent transactions, showing promise for unsupervised anomaly detection where labeled data is scarce. However, the overlap in reconstruction losses between some normal and fraudulent transactions indicates potential issues with robustness and generalization to unseen data. Refining the model architecture and feature engineering could potentially improve its performance and reliability.

The SVM classifier achieved higher recall (82%) than the autoencoder (72.56%), indicating better fraud detection, though it relies on labeled data and cannot be extended to an unsupervised setting. Both models showed mediocre precision (SVM: 32%, Autoencoder:

41.71%), reflecting a high false positive rate. This highlights the recall-precision trade-off, where high recall detects more fraud but lowers precision. What is most desirable in the this context would likely require domain knowledge.

4.2 Interpretability with LIME

LIME provided insights into features driving anomalies, but the anonymized dataset limits actionable conclusions. Without contextual meaning for features, understanding their influence or is challenging. Future work with interpretable datasets could enhance LIMEs impact.

4.3 Future Work and Alternative Techniques

Future work could explore advanced techniques like Variational Autoencoders (VAEs) and Generative Adversarial Networks (GANs), which offer potential for fraud detection despite their complexity. Another improvement could involve secondary thresholds, such as ignoring low-value flagged transactions, to better balance recall, precision, and cost-effectiveness.

5 Conclusion

This project examined the use of autoencoders for anomaly detection in credit card fraud and compared their performance with an SVM. The autoencoder demonstrated some ability to identify fraudulent transactions, essentially without requiring labeled data, only relying on reconstruction errors to detect anomalies. While the SVM achieved higher recall, the autoencoder achieved higher precision. This study highlighted the potential of deep unsupervised methods in addressing fraud detection challenges, particularly in scenarios with imbalanced and unlabeled datasets.

References

[1] L. Ruff, J. R. Kauffmann, R. A. Vandermeulen, G. Montavon, W. Samek, M. Kloft, T. G. Dietterich, and K.-R. Müller, "A unifying review of deep and shallow anomaly detection," *Proceedings of the IEEE*, vol. 109, no. 5, pp. 756–795, 2021.

[2] G. Pang, C. Shen, L. Cao, and A. V. D. Hengel, "Deep learning for anomaly detection: A review," *ACM computing surveys (CSUR)*, vol. 54, no. 2, pp. 1–38, 2021.

[3] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, pp. 1–58, 2009.

[4] Legal Information Institute, Cornell Law School, "Credit Card Fraud," [Online]. Available: https://www.law.cornell.edu/wex/credit_card_fraud/ (accessed on: 2024-12-04).

[5] Financial Consumer Agency of Canada, "Credit card fraud.," [Online]. Available: https://www.canada.ca/en/financial-consumer-agency/services/credit-fraud.html (accessed on: 2024-12-05).

[6] Statista, "Total value of losses due to card fraud worldwide - split between the United States and rest of the world - from 2014 to 2022, with forecasts on the total size of fraud for 2024, 2026, and 2028.," [Online]. Available: https://www.statista.com/statistics/1264329/value-fraudulent-card-transactions-worldwide/ (accessed on: 2024-12-04).

[7] G. K. Kulatilleke, "Challenges and complexities in machine learning based credit card fraud detection," *arXiv preprint arXiv:2208.10943*, 2022.

[8] Lilian Weng, "From Autoencoder to Beta-VAE," [Online]. Available: https://lilianweng.github.io/posts/2018-08-12-vae/ (accessed on: 2024-12-04).

[9] W. Samek and K.-R. Müller, "Towards explainable artificial intelligence," *Explainable AI: interpreting, explaining and visualizing deep learning*, pp. 5–22, 2019.

[10] M. T. Ribeiro, S. Singh, and C. Guestrin, "Model-agnostic interpretability of machine learning," *arXiv preprint arXiv:1606.05386*, 2016.

[11] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.

[12] Scikit-learn, "TSNE," [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html (accessed on: 2024-12-05).

[13] Jeremy Jordan, "Normalizing your data (specifically, input and batch normalization).," [Online]. Available: https://www.jeremyjordan.me/batch-normalization/ (accessed on: 2024-12-04).

[14] C. Leys, C. Ley, O. Klein, P. Bernard, and L. Licata, "Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median," *Journal of experimental social psychology*, vol. 49, no. 4, pp. 764–766, 2013.