C^{2}BNVAE: Dual-Conditional Deep Generation of Network Traffic Data for Network Intrusion Detection System Balancing

Yifan Zeng [1]

Abstract

Network Intrusion Detection Systems (NIDS) face challenges due to class imbalance, affecting their ability to detect novel and rare attacks. This paper proposes a Dual-Conditional Batch Normalization Variational Autoencoder (C^{2}BNVAE) for generating balanced and labeled network traffic data. C^{2}BNVAE improves the model's adaptability to different data categories and generates realistic category-specific data by incorporating Conditional Batch Normalization (CBN) into the Conditional Variational Autoencoder (CVAE). Experiments on the NSL-KDD dataset show the potential of C^{2}BNVAE in addressing imbalance and improving NIDS performance with lower computational overhead compared to some baselines.

1. Introduction

Network intrusions are growing exponentially, diversifying, and becoming more severe. They threaten personal privacy, data security, and can lead to significant economic and social impacts. Deep Learning-based Network Intrusion Detection Systems (DLNIDS) have shown powerful intrusion detection capabilities. However, DLNIDS face a notable challenge in practical applications: the severe class imbalance in network traffic data. In real-world network environments, normal traffic typically constitutes the vast majority, while various types of attack traffic are relatively rare. This imbalance makes it difficult for DLNIDS to effectively learn the features of minority classes, thereby reducing the system's ability to detect novel and rare attacks.

To address data imbalance, researchers have proposed various methods, including resampling techniques (e.g., SMOTE (Chawla et al., 2002)) and generative approaches (Liu et al., 2022; Zeng, 2025). Among generative methods, Variational Autoencoders (VAEs) (Kingma et al., 2013) have shown the ability to learn the latent distribution of data and generate new, synthetic samples. Conditional Variational Autoencoders (CVAEs) (Sohn et al., 2015) further enhance this capability by allowing the generation of data for specific categories using conditional information.

In this paper, we propose C^{2}BNVAE (Dual-Conditional Batch Normalization Variational Autoencoder), a generative model designed to effectively produce balanced and labeled network traffic data for DLNIDS. Our primary contribution is the integration of Conditional Batch Normalization (CBN) (Yin et al., 2019) into the CVAE architecture for network traffic data generation. CBN, by learning separate affine transformation parameters (γ, β) for each data category, allows the model to better adapt its normalization process to the specific characteristics of different traffic classes. This "dual-conditional" approach (conditioning in CVAE and conditioning in CBN) aims to enhance the model's adaptability and the realism of generated category-specific data. Through experiments on the NSL-KDD dataset, we validate the potential of C^{2}BNVAE in generating minority class samples and subsequently improving NIDS detection performance when using a Decision Tree classifier.

2. Proposed Method: C^{2}BNVAE

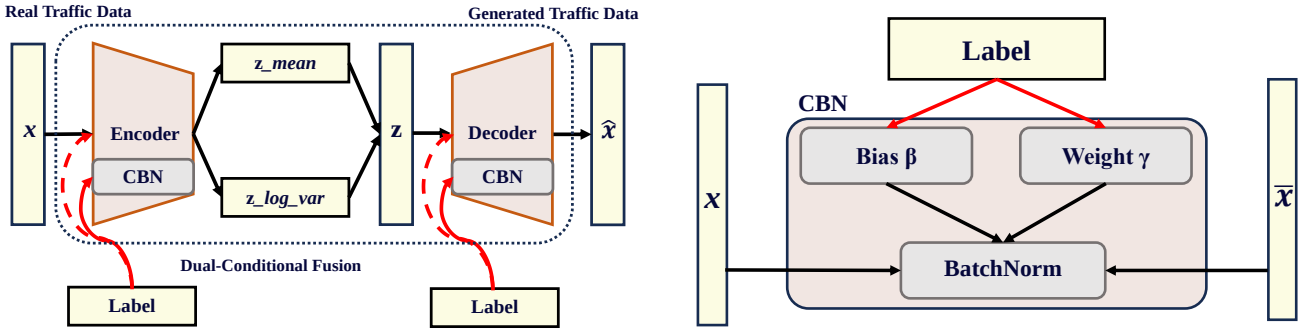
The C^{2}BNVAE model builds upon the Conditional Variational Autoencoder (CVAE) by incorporating Conditional Batch Normalization (CBN) to enhance category-specific data generation.

2.1. Conditional Variational Autoencoder (CVAE) Backbone

A standard VAE learns a probabilistic mapping from the input data x to a latent space z and back. It is trained by maximizing an evidence lower bound (ELBO), which typically consists of a reconstruction term and a regularization term (KL divergence between the learned latent distribution and a prior, usually a standard normal distribution). However, a VAE cannot generate data for specific categories by default.

CVAE extends VAE by conditioning the generation process on additional information, typically class labels y. In our

[1]Sun Yat-sen University, Guangzhou, China. Correspondence to: Yifan Zeng <yifanzeng0615@foxmail.com>.



(a) Overall C^{2}BNVAE structure.

(b) CBN mechanism.

Figure 1. (a) The overall structure of C^{2}BNVAE. Real data is input, and generated data is output. Traffic labels are integrated into the Encoder, Decoder, and CBN layers. (b) The structure of CBN. The class label selects the learned scaling factors γ_i and β_i for normalizing data belonging to that specific class.

C^{2}BNVAE, the one-hot encoded class label y of the traffic data is concatenated with the input x for the encoder and with the latent variable z for the decoder. This allows the model to learn class-specific latent representations and generate samples belonging to a target minority class by providing its label y. The loss function for C^{2}BNVAE:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{regu}} \tag{1}$$

where $\mathcal{L}_{\text{recon}}$ is the reconstruction loss and $\mathcal{L}_{\text{regu}}$ is the regularization loss. Let N be the number of training samples, x_i be the i-th training sample, and $\hat{x}_i$ be the i-th sample generated by the model conditioned on y_i. $\mathcal{L}_{\text{recon}}$ is measured by the mean squared error:

$$\mathcal{L}_{\text{recon}} = \frac{1}{N}\sum_{i=1}^{N}(x_i - \hat{x}_i)^2 \tag{2}$$

$\mathcal{L}_{\text{regu}}$ is the KL divergence between the learned posterior distribution of latent variables $q(z|x,y)$ (approximated by $N(\mu(x,y), \sigma^2(x,y))$) and the prior $p(z|y)$ (often simplified to $N(0,I)$):

$$\mathcal{L}_{\text{regu}} = \mathcal{D}_{\text{KL}}(q(z|x,y) \,||\, p(z|y)) \tag{3}$$

2.2. Conditional Batch Normalization (CBN)

In standard CVAEs, Batch Normalization (BN) (Bjorck et al., 2018) is often used to stabilize training and improve generalization. However, BN applies a single set of learned affine parameters (γ, β) across all samples in a batch, regardless of their class. This can be suboptimal when generating data for diverse categories, potentially leading to more uniform or less distinct category-specific features.

CBN addresses this by making the affine transformation parameters conditional on the class label y. For each class i, CBN learns separate scaling (γ_i) and shifting (β_i) parameters. Given a mini-batch input x, its sample mean $\hat{\boldsymbol{\mu}}$ and standard deviation $\hat{\boldsymbol{\sigma}}$, the CBN transformation for samples belonging to class i is:

$$\overline{x} = \text{CBN}(x|y=i) = \gamma_i \frac{x - \hat{\boldsymbol{\mu}}}{\sqrt{\hat{\boldsymbol{\sigma}}^2 + \epsilon}} + \beta_i \tag{4}$$

where ϵ is a small constant for numerical stability. By incorporating CBN into the decoder (and potentially encoder) layers of the CVAE, C^{2}BNVAE aims to provide greater flexibility in modeling the distinct statistical properties of each traffic category. This "dual conditioning" – first through the CVAE structure and second through CBN – is hypothesized to improve the model's adaptability and the diversity and authenticity of the generated samples, especially for minority classes.

The overall structure of C^{2}BNVAE is depicted in Figure 1.

C^{2}BNVAE aims to:

- Generate high-quality, realistic traffic data for specific (minority) categories.
- Balance imbalanced datasets by synthesizing minority class samples, thereby enhancing the detection performance of downstream NIDS classifiers.
- Create more diverse and authentic synthetic data that accurately reflects the features and categorical distinctions within network traffic.

3. Experimental Setup

3.1. Dataset

We conducted experiments on the widely used NIDS benchmark dataset, NSL-KDD (Ravipati & Abualkibash, 2019). NSL-KDD is an improved version of the KDD Cup '99 dataset and is frequently used for evaluating intrusion detection systems. It contains various types of attacks and normal

traffic records, characterized by 41 features. A key characteristic of NSL-KDD is its significant class imbalance, making it a suitable benchmark for assessing algorithms designed to handle imbalanced data. For our experiments, we used the KDDTrain+ dataset for training the generative models and the KDDTest+ for evaluating the NIDS classifier. To balance the dataset for training the classifier, with the Normal class in KDDTrain+ having the most samples (67343), we generated a matching number of samples for each minority attack category using the respective data augmentation algorithms.

3.2. Evaluation Metrics

To evaluate the performance of the NIDS classifier trained on data augmented by different methods, we employed comprehensive metrics suitable for imbalanced datasets:

- **Accuracy (Acc):** The proportion of correctly classified instances among the total instances:

$$\text{Acc} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{5}$$

- **Weighted Precision (Pre$_w$):** The weighted average of precision for each class, where Pre is TP / (TP + FP):

$$\text{Pre}_w = \sum_{c=1}^{C} \left(\frac{N_c}{N_{\text{total}}} \cdot \frac{\text{TP}_c}{\text{TP}_c + \text{FP}_c} \right) \tag{6}$$

- **Weighted Recall (Recall$_w$):** The weighted average of recall for each class, where Recall is TP / (TP + FN):

$$\text{Recall}_w = \sum_{c=1}^{C} \left(\frac{N_c}{N_{\text{total}}} \cdot \frac{\text{TP}_c}{\text{TP}_c + \text{FN}_c} \right) \tag{7}$$

- **Weighted F1-Score (F1$_w$):** The weighted average of the F1-score for each class. The F1-score is the harmonic mean of precision and recall ($2 \times (Pre \times Recall)/(Pre + Recall)$). The weighted F1-score is calculated as:

$$\text{F1}_w = \sum_{c=1}^{C} \left(\frac{N_c}{N_{\text{total}}} \cdot 2 \cdot \frac{\text{Pre}_c \cdot \text{Recall}_c}{\text{Pre}_c + \text{Recall}_c} \right) \tag{8}$$

  where C is the number of classes, N_c is the number of instances in class c, and N_{total} is the total number of instances.

- **FLOPs:** Floating-point operations, measuring computational complexity during inference. Lower values indicate higher efficiency.

- **Params:** Number of trainable parameters, reflecting model size and memory requirements.

Table 1. Model Hyperparameters of C^{2}BNVAE

Hyperparameters	**Encoder**	**Decoder**
Layers	[128,60×4,32]	[37,60×4,123]
Activation	LeakyReLU	LeakyReLU
Initialization	He	He
Batch size	128	128
Learning rate	1e-4	1e-4
Epoch	120	120
Optimizer	Adam	Adam
Loss function	MSELoss	MSELoss

These weighted metrics give more importance to classes with more samples but still provide a holistic view of performance across all classes.

3.3. Baselines and Classifier

We compared C^{2}BNVAE with several baseline data balancing techniques: Original imbalanced data (No balancing), Random Oversampling, SMOTE (Synthetic Minority Over-sampling Technique) (Chawla et al., 2002), Borderline SMOTE (Dey & Pratap, 2023), KMeans SMOTE (Maulana et al., 2024), SVM SMOTE (Wang et al., 2017), Standard CVAE (without CBN) (Sohn et al., 2015).

The downstream NIDS classifier employed in all experiments is a Decision Tree (DT). We chose DT for its simplicity, interpretability, and common use as a baseline classifier. The performance of the DT classifier trained on data generated/balanced by these algorithms serves as a proxy for the quality and utility of the augmented data.

3.4. Implementation Details

The experiments were conducted on a computing environment with an Intel (R) Xeon (R) Gold 6240 CPU @ 2.60GHz and a Tesla V100S-PCIE-32GB GPU. The operating system was Ubuntu 18.04.3 LTS. All code was implemented in Python 3.7.6 using PyTorch 1.13.1+cu117. Specific model hyperparameters for C^{2}BNVAE are detailed in Table 1.

4. Results and Discussion

4.1. Computational Overhead

C^{2}BNVAE has a total of 43,627 parameters and incurs 43,200 FLOPs per sample. The encoder component contributes 22,744 parameters and 22,560 FLOPs, while the decoder has 20,883 parameters and 20,640 FLOPs. For comparison, a Conditional Generative Adversarial Network (CGAN) baseline (Mirza & Osindero, 2014; Zeng, 2025), implemented with a generator having layers [128, 100×5,

Table 2. Intrusion detection performance experimental results (%) on KDDTest+ using a Decision Tree classifier trained on augmented KDDTrain+ data.

Algorithms	**Acc**	**Pre**$_w$	**Recall**$_w$	**F1**$_w$
Original imbalanced Data	75.88	79.32	75.88	72.74
Random oversampling	77.02	79.14	77.02	73.84
SMOTE	76.11	78.04	76.11	73.33
Borderline SMOTE	75.89	78.73	75.89	73.59
KMeans SMOTE	76.13	79.03	76.13	72.81
SVM SMOTE	78.11	79.09	78.11	76.02
CVAE	78.45	79.10	78.45	77.18
C^{2}BNVAE	**79.40**	**80.69**	**79.40**	**78.19**

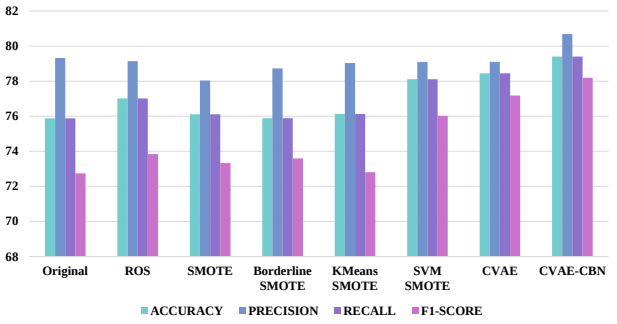


Figure 2. Bar chart of Decision Tree detection performance when trained on balanced data processed by different algorithms. Higher bars indicate better performance.

123] and a discriminator with layers [128, 100, 50, 50, 1], has significantly more parameters (87,820) and higher computational cost (109,892 FLOPs). This comparison highlights the relatively lower computational overhead of the proposed C^{2}BNVAE model, making it potentially more efficient for deployment in resource-constrained NIDS environments.

4.2. Intrusion Detection Performance

The primary evaluation of C^{2}BNVAE lies in its ability to generate synthetic minority class data that, when used to balance the training set, improves the performance of a downstream NIDS classifier. Table 2 presents the intrusion detection performance of the Decision Tree classifier on the KDDTest+ dataset when trained on data augmented by various methods.

As shown in Table 2, training the Decision Tree classifier on data augmented by C^{2}BNVAE yielded the best performance across all evaluated metrics: Accuracy (79.40%), Weighted Precision (80.69%), Weighted Recall (79.40%), and Weighted F1-Score (78.19%). This represents a notable improvement over training on the original imbalanced data (F1-Score: 72.74%) and also surpasses other common oversampling techniques like Random Oversampling (F1-Score: 73.84%) and various SMOTE variants (F1-Scores ranging from 72.81% to 76.02%).

Importantly, C^{2}BNVAE also outperforms the standard CVAE (F1-Score: 77.18%). This suggests that the introduction of Conditional Batch Normalization (CBN) provides a tangible benefit in generating more effective synthetic samples for balancing the dataset. The CBN allows the model to learn class-specific normalization parameters, potentially leading to the generation of minority class samples that are more distinct and better capture the unique characteristics of each attack type. This, in turn, helps the downstream classifier to learn more robust decision boundaries, particularly for underrepresented attack classes.

The bar chart in Figure 2 further visualizes these performance differences, clearly positioning C^{2}BNVAE as the top-performing data augmentation method in this experimental setup.

4.3. Discussion

The superior performance of C^{2}BNVAE can be attributed to its dual-conditional mechanism. The CVAE component ensures that generated samples belong to the specified target class, while the CBN component fine-tunes the feature generation process by applying class-specific normalization. This likely results in synthetic data that not only increases the representation of minority classes but also maintains or even enhances the separability between different classes. Traditional BN, used in standard CVAE, might inadvertently smooth out some class-specific features by applying uniform normalization parameters. CBN mitigates this by allowing the normalization to adapt to each class's statistical profile, leading to more realistic and useful synthetic samples.

The lower computational overhead of C^{2}BNVAE compared to GAN (Goodfellow et al., 2020) or CGAN is also a practical advantage, particularly for NIDS applications where frequent retraining or deployment on edge devices might be necessary.

While these results are promising, it is important to acknowledge that they are based on the NSL-KDD dataset and a Decision Tree classifier. The effectiveness of C^{2}BNVAE might vary with other datasets possessing different characteristics or when paired with more complex deep learning classifiers for NIDS. Future work should explore its generalizability across a wider range of network environments, attack types, and NIDS architectures. Additionally, qualitative analysis of the generated samples (e.g., using t-SNE or UMAP for visualization) could provide further insights into how well C^{2}BNVAE captures the underlying data distributions compared to other methods.

5. Conclusion

In this paper, we proposed C^{2}BNVAE, a dual-conditional generative model for addressing class imbalance in network traffic data for NIDS. By integrating CBN into a CVAE, C^{2}BNVAE enhances the model's adaptability to different data categories and its ability to generate realistic, category-specific synthetic samples. Our experimental results on the NSL-KDD dataset demonstrate that C^{2}BNVAE outperforms several baseline oversampling techniques and standard CVAE in improving the detection performance of a Decision Tree based NIDS, while also exhibiting lower computational overhead compared to a CGAN baseline. This study highlights the potential of incorporating class-conditional normalization techniques like CBN into generative models for effectively tackling data imbalance in security applications. While further research is needed to assess its generalizability, C^{2}BNVAE offers a promising and computationally efficient approach for enhancing the robustness of NIDS against novel and rare attacks by creating more balanced and representative training datasets.

References

Bjorck, N., Gomes, C. P., Selman, B., and Weinberger, K. Q. Understanding batch normalization. *Advances in neural information processing systems*, 31, 2018.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

Dey, I. and Pratap, V. A comparative study of smote, borderline-smote, and adasyn oversampling techniques using different classifiers. In *2023 3rd international conference on smart data intelligence (ICSMDI)*, pp. 294–302. IEEE, 2023.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

Kingma, D. P., Welling, M., et al. Auto-encoding variational bayes, 2013.

Liu, C., Antypenko, R., Sushko, I., and Zakharchenko, O. Intrusion detection system after data augmentation schemes based on the vae and cvae. *IEEE Transactions on Reliability*, 71(2):1000–1010, 2022.

Maulana, D. J., Saadah, S., Yunanto, P. E., et al. Kmeans-smote integration for handling imbalance data in classifying financial distress companies using svm and naïve bayes. *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, 8(1):54–61, 2024.

Mirza, M. and Osindero, S. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

Ravipati, R. D. and Abualkibash, M. Intrusion detection system classification using different machine learning algorithms on kdd-99 and nsl-kdd datasets-a review paper. *International Journal of Computer Science & Information Technology (IJCSIT) Vol*, 11, 2019.

Sohn, K., Lee, H., and Yan, X. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28, 2015.

Wang, Q., Luo, Z., Huang, J., Feng, Y., and Liu, Z. A novel ensemble method for imbalanced data learning: Bagging of extrapolation-smote svm. *Computational intelligence and neuroscience*, 2017(1):1827016, 2017.

Yin, G., Liu, B., Sheng, L., Yu, N., Wang, X., and Shao, J. Semantics disentangling for text-to-image generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2327–2336, 2019.

Zeng, Y. CSAGC-IDS: A dual-module deep learning network intrusion detection model for complex and imbalanced data. *arXiv preprint arXiv:2505.14027*, 2025.