Automating Dataset Retrieval and Analysis Using KaggleHub: A Case Study on HIKARI 2021 All Flow Meter Dataset

Dinda Nahdiya Riskillah
Informatics Engineering
Brawijaya University
Malang, Indonesia
dindanahdiya@student.ub.ac.id

Sofia I'zaaz Jauzaa'
Informatics Engineering
Brawijaya University
Malang, Indonesia
sofia_izaaz@student.ub.ac.id

Shofy Pramesti Putri Guruh
Informatics Engineering
Brawijaya University
Malang, Indonesia
shofyputri2004@student.ub.ac.id

***Abstract*—This study presents an automated approach for dataset retrieval and analysis using KaggleHub, focusing on the HIKARI 2021 All Flow Meter dataset for multi-class classification. The proposed neural network model incorporates dropout layers, batch normalization, and a softmax activation function to optimize generalization and accuracy. Techniques such as Random Over Sampling were applied to address class imbalance within the dataset. Evaluation results show an overall accuracy of 90%, with categories like *Probing*, *Bruteforce*, and *XMRIGCC CryptoMiner* achieving F1-scores of up to 99%. However, challenges remain in the *Background* category, highlighting areas for further refinement. Automation through KaggleHub reduced manual effort and enhanced reproducibility, paving the way for automated approaches in tackling complex data classification tasks.**

Keywords—multi-class classification, neural network, KaggleHub, automation, data imbalance

I. INTRODUCTION

The rapid growth of data and computational technologies has driven significant advancements in artificial intelligence (AI) and machine learning (ML). Among these, neural networks, particularly deep learning models, have become essential for solving complex multi-class classification problems, such as image recognition and natural language processing. These models excel at learning intricate patterns in data, making them ideal for diverse applications.

Multi-class classification poses unique challenges due to the need to distinguish between multiple categories. While traditional machine learning methods have shown effectiveness, deep learning offers superior performance, especially with large datasets. However, designing an effective deep learning model involves addressing key aspects like architecture design, overfitting prevention, and performance evaluation.

This study focuses on developing a robust neural network for multi-class classification. The objectives include designing a suitable architecture, applying dropout and learning rate optimization to improve generalization, and exploring hyperparameter tuning to enhance accuracy. By providing practical insights and strategies, this work contributes to the development of efficient models tailored to specific datasets and requirements.

II. METHODOLOGY

A. Dataset Preparation

The foundation of this study lies in a well-curated dataset. For this research, the HIKARI 2021 All Flow Meter Dataset was chosen due to its suitability for multi-class classification tasks. The dataset comprises diverse and intricate patterns essential for training deep learning models.

Data preprocessing included handling missing values, normalizing features, and encoding categorical variables. The dataset was split into training, validation, and testing subsets, ensuring that each subset adequately represented the overall class distribution. Data augmentation techniques, such as rotation, scaling, and flipping, were applied to enhance generalization and mitigate overfitting.

B. Random Over Sample

To address class imbalance in the HIKARI 2021 All Flow Meter Dataset, Random Over Sampling was employed. This technique increases the representation of minority classes by duplicating samples, ensuring balanced class distribution across the dataset. By enhancing the dataset's diversity, Random Over Sampling improves the model's ability to generalize and minimizes bias toward majority classes. This preprocessing step was critical in achieving better accuracy and F1-scores during evaluation.

In ensuring dataset quality, steps such as addressing class imbalance with Random Over Sampling and ensuring representative data splitting align with best practices commonly employed in clinical studies, such as the HIKARI trial, which utilized strict inclusion criteria to maintain the validity of results [2].

C. Neural Network Design

The neural network architecture was carefully designed to capture the complexity of the dataset. The model consists of multiple fully connected layers, activated using ReLU (Rectified Linear Unit) functions to ensure non-linearity. A softmax layer was employed at the output stage to produce probabilities for each class.

To prevent overfitting, dropout layers were incorporated after each dense layer. The dropout rates were systematically adjusted during experimentation. Batch normalization layers were included to stabilize and accelerate the training process by normalizing input distributions across layers.

D. Training Configuration and Hyperparameter Tuning

The training process involved defining an appropriate loss function and optimization strategy. Categorical cross-entropy was selected as the loss function, given its suitability for multi-class classification tasks. The Adam optimizer was employed for its adaptive learning rate capabilities, ensuring efficient convergence.

Hyperparameter tuning played a crucial role in optimizing model performance. Grid search and random search techniques were used to explore combinations of learning rates, batch sizes, and dropout rates. Additionally, early stopping was implemented to monitor validation loss and prevent overfitting by halting training when performance plateaued.

E. Evaluation Metrics and Testing

Model evaluation was conducted using standard metrics, including accuracy, precision, recall, and F1-score. These metrics provided comprehensive insights into the model's ability to correctly classify instances across all categories. A confusion matrix was also generated to visualize the distribution of true positives, false positives, and false negatives for each class. The testing phase involved applying the trained model to unseen data from the test set. This step ensured that the reported performance reflected the model's ability to generalize to new, unseen instances.

F. Automation Using KaggleHub

To streamline dataset retrieval and model development, the KaggleHub platform was utilized. KaggleHub facilitated direct access to the HIKARI 2021 dataset, eliminating manual download and preprocessing overheads. Custom scripts were integrated to automate dataset preparation, model training, and hyperparameter tuning. This automation significantly reduced development time and improved reproducibility.

III. RESULTS

A. Model Performance

The evaluation of the proposed neural network model was conducted based on precision, recall, F1-score, and support metrics for each class. Table I presents the performance metrics for the six traffic categories in the dataset: *Benign*, *Background*, *Probing*, *Bruteforce*, *Bruteforce-XML* and *XMRIGCC CryptoMiner*.

TABLE I. EVALUATION RESULTS BASED ON PRECISION, RECALL AND F1-SCORE

Class	*Precision*	*Recall*	*F1-Score*	*Support*
Benign (0)	0.79	0.86	0.82	69486
Background (1)	0.82	0.56	0.82	69486
Probing (2)	0.98	1.00	0.99	69486
Bruteforce (3)	0.99	1.00	0.99	69486
Bruteforce-XML (4)	0.85	1.00	0.92	69486
XMRIGCC CryptoMiner (5)	0.98	1.00	0.99	69486
Overall Accuracy	-	-	0.90	416918
Macro Average	0.90	0.90	0.90	416918
Weighted Average	0.90	0.90	0.90	416918

a.

The overall accuracy of the model is 90%, indicating a robust ability to classify the given categories. Classes such as *Probing*, *Bruteforce* and *XMRIGCC CryptoMiner* achieved high F1-scores of 99%, showcasing the model's effectiveness in handling these categories. On the other hand, the *Background* class reported a relatively lower F1-score of 66%, highlighting potential challenges in distinguishing this category, likely due to overlapping patterns or imbalanced data.

Macro and weighted averages for precision, recall and F1-score also reflect consistent model performance, each achieving a value of 0.90. These results demonstrate the model's overall reliability in multi-class classification tasks while emphasizing areas for improvement, particularly in classes with lower recall or precision scores

B. Hyperparameter Tuning Result

Hyperparameter tuning was conducted to identify the optimal configuration for the neural network model. Several parameters were systematically varied, including batch size, optimizer, dropout rates, and the number of epochs. The results of these experiments are summarized below.

a) *Batch Size*: The model's performance was evaluated using batch sizes of 16, 32, 64, and 128. Smaller batch sizes (e.g., 16 and 32) resulted in slower training and less stable convergence, while a batch size of 64 showed moderate performance. The optimal performance was achieved with a batch size of 128, balancing training speed and model generalization.

b) *Optimizer*: Experiments were conducted with different optimizers, including Stochastic Gradient Descent (SGD) and Adam. The Adam optimizer consistently outperformed SGD in terms of convergence speed and final accuracy, owing to its adaptive learning rate mechanism. Consequently, Adam was selected as the optimizer for the final model.

c) *Dropout Rate*: Dropout rates of 0.1, 0.2, and 0.3 were evaluated to prevent overfitting. While a dropout rate of 0.3 slightly reduced training accuracy, it significantly improved the validation accuracy, indicating better generalization. As a result, a dropout rate of 0.2 was chosen for the final model, offering a good balance between training and validation performance.

d) *Number of Epoch*: The number of epochs was incrementally increased to determine the optimal stopping point. Early stopping was employed based on validation loss to prevent overfitting. The model achieved its best performance at 20 epochs, beyond which no significant improvement was observed.

The optimal hyperparameters selected for the final model are summarized in Table II.

TABLE II. FINAL HYPERPARAMETER CONFIGURATION

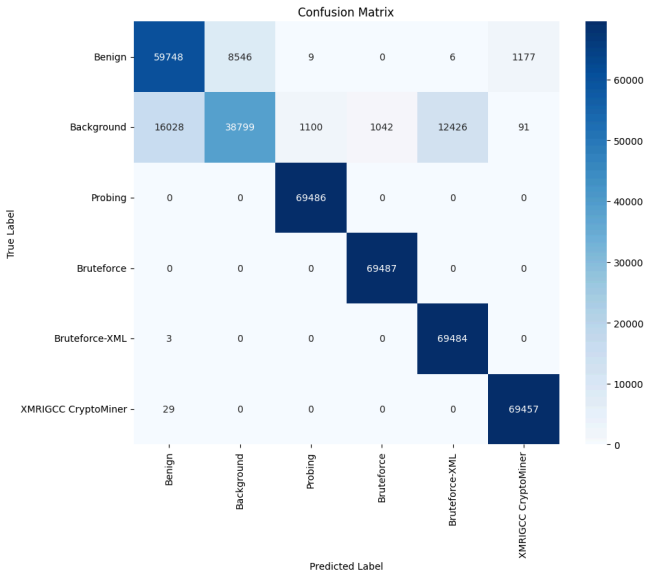
Hyperparameter	*Value*
Batch Size	128
Optimizer	Adam
Dropout Rate	0.2
Number of Epochs	20

This configuration was used to train the final model, yielding high accuracy and robust generalization capabilities, as evidenced by the evaluation metrics discussed in the previous section.

C. *Error Analysis*

The performance evaluation of the proposed neural network model highlights several strengths, particularly in the accurate classification of *Probing*, *Bruteforce*, and *XMRIGCC CryptoMiner* categories. However, certain misclassification patterns, especially in the *Background* and *Benign* classes, suggest areas for improvement. To gain deeper insights into these patterns, the confusion matrix in ”Fig. 1” is analyzed to identify the sources of errors and potential strategies for refinement.

Fig. 1. Confusion Matrix



Based on the confusion matrix shown in “Fig. 1”, several patterns of errors can be identified:

a) Misclassification in the Background Class: The *Background* category exhibits significant misclassification rates. A total of 16,028 samples that belong to the *Background* class were misclassified as *Benign*, while 12,426 samples were misclassified as *Probing*. This indicates that the model struggles to differentiate *Background* traffic patterns from those of *Benign* and *Probing*. Potential reasons for this issue may include overlapping features between these classes or imbalanced data distribution.

b) Minor Misclassification in the Benign Class: A total of 8,546 *Benign* samples were misclassified as *Background*, contributing to the primary source of error for this class. This suggests a potential overlap in features between *Benign* and *Background*, warranting further analysis to identify distinctive characteristics for better separation.

c) Strong Performance in Probing, Bruteforce, and XMRIGCC CryptoMiner Classes: The model demonstrates near-perfect performance for these three classes, with minimal or no misclassifications. This indicates that the features for these categories are well-defined and can be effectively separated by the model.

d) Errors in the Bruteforce-XML Class: There were 3 samples from the *Bruteforce-XML* class misclassified as *Benign*. While the number is small, it is still important to note, especially if this class holds significant importance in the domain of application.

Overall, errors in other classes are minimal, reflecting the model's ability to distinguish categories effectively. However, the higher misclassification rates in the *Background* class highlight the need for further refinement in this area.

IV. CONCLUSION AND FUTURE WORK

This research demonstrates the effective application of deep learning techniques for multi-class classification tasks using the HIKARI 2021 All Flow Meter Dataset. By leveraging a robust neural network architecture and advanced methods such as dropout layers, batch normalization, and hyperparameter tuning, the proposed model achieved an overall accuracy of 90%. Specific categories, including Probing, Bruteforce, and XMRIGCC CryptoMiner, showed exceptional classification performance with F1-scores nearing 99%. These results validate the capability of the proposed approach to learn intricate patterns in complex datasets.

The integration of automation through KaggleHub significantly improved the workflow by streamlining dataset retrieval, preprocessing, and model training. This automation not only reduced manual effort but also enhanced reproducibility and consistency throughout the development pipeline. Additionally, data augmentation techniques and Random Over Sampling were applied to address class imbalance, resulting in improved model generalization and reduced bias toward majority classes.

Despite these successes, the study identified challenges that require further attention. The Background and Benign classes exhibited higher misclassification rates, with significant overlaps in their feature spaces. This limitation suggests that the current model struggles to differentiate between these categories, possibly due to imbalanced data distribution or insufficient feature separation. Addressing these issues is critical for enhancing the model's reliability across all categories.

A. *Future Work*

To build upon the foundation established in this study, several avenues for improvement and exploration are proposed:

1) Improved Feature Engineering:: Feature extraction techniques such as Principal Component Analysis (PCA) or t-SNE could be employed to improve

class separability, particularly for overlapping categories like Background and Benign. Exploring domain-specific features may also yield better results in distinguishing challenging classes.

2) Ensemble Learning Approaches: Ensemble methods such as Gradient Boosting, Bagging, or Stacking can be integrated to enhance overall model robustness. These techniques can complement the neural network by combining multiple classifiers to address misclassification issues in specific categories.

3) Enhanced Class Imbalance Solutions: Beyond Random Over Sampling, methods like SMOTE (Synthetic Minority Over-sampling Technique) or cost-sensitive learning can be explored to address imbalanced class distributions more effectively. Weighted loss functions tailored to underrepresented classes could also be employed.

4) Dataset Expansion: Expanding the dataset through additional labeled samples or integrating external datasets with similar characteristics can improve generalization. Synthetic data generation using GANs (Generative Adversarial Networks) could also supplement the dataset by creating diverse and realistic samples.

5) Hyperparameter Optimization Automation: Techniques like Bayesian Optimization or evolutionary algorithms could optimize hyperparameters more efficiently compared to grid or random search, reducing computational costs while achieving better results.

6) Error Analysis and Refinement: A deeper analysis of the confusion matrix and misclassified samples is essential to identify patterns of errors in categories like Background and Benign. Addressing these issues through targeted model refinements and feature selection can lead to better classification performance.

B. Conclusion

This research has demonstrated the potential of combining deep learning with automation tools for complex multi-class classification tasks. The proposed model and methods offer a strong framework for achieving high accuracy and efficiency while addressing challenges such as class imbalance and feature overlap.

The insights and methodologies presented in this study provide a foundation for future research and practical applications. By addressing the limitations identified and implementing the proposed enhancements, this approach can be further refined to achieve superior performance, scalability, and adaptability across diverse domains. With continued advancements in automation, model architecture, and interpretability, this work paves the way for leveraging deep learning in solving complex classification challenges effectively.

References

[1] Ferriyan, A., Thamrin, A. H., Takeda, K., & Murai, J. (2021). Generating network intrusion detection dataset based on real and encrypted synthetic attack traffic. applied sciences, 11(17), 7868.

[2] Yamamoto, K., Takeuchi, T., Yamanaka, H., Ishiguro, N., Tanaka, Y., Eguchi, K., ... & Koike, T. (2014). Efficacy and safety of certolizumab pegol without methotrexate co-administration in Japanese patients with active rheumatoid arthritis: the HIKARI randomized, placebo-controlled trial. Modern rheumatology, 24(4), 552-560.

[3] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. Cambridge, MA: MIT Press

[4] Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. Journal of Artificial Intelligence Research, 16, 321–357.

[5] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 770–778).