Car Crash Severity Detection

Jose Avalos
California State University, Sacramento
Sacramento, California, United State of America

Jose Vasquez
California State University, Sacramento
Sacramento, California, United State of America

Jomel Sotelo
California State University, Sacramento
Sacramento, California, United State of America

Mitchell Kouiyoth
California State University, Sacramento
Sacramento, California, United State of America

Figure 1: Car Crash image from Kaggle data set, NHTSA CRSS Accident Records 2016-2020

Abstract

Car accident severity detection is crucial for improving emergency response and ensuring public safety. In this paper, we are interested in predicting the level of severity of car accidents by utilizing deep learning models based on different features such as weather,make/model of vehicle, time, and other contributory variables like numeric values like weight, and number of injuries. A Fully Connected Neural Network (FCNN), Convolutional Neural Network (CNN) without transfer learning, and Transformer are used to resolve the problem since data from the dataset is all tabular. The outcome of the experiments indicates that Transformer model is improving the classification accuracy more compared to the baseline scenario with a score of 96 percent accuracy while the baseline approach was our Fully connected Neural Network was 89 percent.

ACM Reference Format:

Jose Avalos, Jose Vasquez, Jomel Sotelo, and Mitchell Kouiyoth. 2018. Car Crash Severity Detection. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (Conference acronym 'XX)*. ACM, New York, NY, USA, 6 pages. https://doi.org/XXXXXXX.XXXXXXX

Conference acronym 'XX, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Car accident severity detection has the potential to significantly improve the efficiency of emergency response services, helping to save lives by providing timely and effective resource allocation. This paper addresses the problem of predicting the severity level of car accidents using the US [3] Traffic Accidents dataset, which contains detailed information on accidents across the United States from 2016 to 2020. We propose using deep learning models to classify accidents into different severity levels, leveraging FCNN, CNN, and

Transformer models to explore the best approach. Our contributions include:

- data preprocessing to merge all three seperate files (Accidents,Persons,Vehicle)
- Undersampling the data since certain categories of STRATUM had up to 300,000 records while other had 48,000 records
- Normalize numeric data with z-scores, and encode categorical data
- Use train/test data split.
- Implementing a Fully Connected Neural Network
- Implement a Convalutional Neural Network
- visualize dataset with confusion matrix
- evaluate dataset with precision, recall, F-measure
- use EarlyStopping and ModelCheckpoint when training neural networks using Tensorflow
- Tune the following hyperparameters when training neural networks using Tensorflow to tabulate all the results of each model

The rest of the paper is organized as follows: Section 2 formulates the problem, Section 3 describes our system design and algorithms, Section 4 presents the experimental evaluation, Section 5 discusses related work, Section 6 concludes the paper, Section 7 outlines the work division, and Section 8 shares our learning experience.

2 Problem Formulation

The problem at hand is to predict the severity level of car accidents based on various features available in the dataset. Accident severity is inherently a categorical variable that can take on multiple distinct values, representing different levels of severity. This categorization aligns the problem as a multi-class classification task where the goal is to accurately classify accidents into one of several severity categories. Accurately predicting the severity of accidents can help allocate resources more efficiently in emergency response systems, potentially reducing response times and saving lives.

The model receives data from the US Traffic Accidents dataset, which contains detailed information on accidents across the United States from 2016 to 2020. Separating from three different requirements on accidents, persons, and vehicles. The feature inputs available in the dataset may include, but are not limited to:

- WEIGHT: Weighting factor for statistical analysis.
- VE_TOTAL: Total number of vehicles involved in the crash.
- VE_FORMS: Number of vehicle forms filled.
- MAXSEV_IM: Maximum severity of injuries in the crash (imputed).
- HARM_EV: Initial harmful event causing injury or damage.
- HARM_EVNAME: Description of the harmful event.
- WEATHER: Weather conditions during the crash.
- WEATHERNAME: Description of the weather conditions.
- LGT_COND: Lighting conditions at the crash site.
- LGT_CONDNAME: Description of the lighting conditions.
- ROAD_FNCNAME: Functional classification of the road.
- AGE: Age of the person involved.
- SEAT_POS_IM: Seat position of the person (imputed).
- EJECT_IM: Whether the person was ejected (imputed).
- SEX: Gender of the person.
- SEXNAME: Description of gender.
- DRINKING: Indication if the person had consumed alcohol.
- DRINKINGNAME: Description of drinking behavior.
- EJECTION: Whether the person was ejected from the vehicle.
- EJECTIONNAME: Description of ejection status.
- INJ_SEV: Injury severity of the person.
- INJ_SEVNAME: Description of injury severity.
- VEVENT_IM: Vehicle event (imputed).
- MAX_VSEV: Maximum vehicle severity in the crash.
- NUMINJ_IM: Number of injured persons (imputed).
- BODY_TYP: Body type of the vehicle.
- BODY_TYPNAME: Description of the vehicle body type.
- TOWED: Whether the vehicle was towed from the scene.
- TOWEDNAME: Description of towing status.
- ROLLOVER: Whether the vehicle rolled over.
- ROLLOVERNAME: Description of the rollover event.

The problem is considered a multiclass-classification problem since we are identifying various severity, our output are within the hierarchical structure generated by CRSS from figure 2.

CRSS Police Crash Report Domain Definition, Target Sample Allocation, and Population Distribution

Stratum	Description (Hierarchical Structure)	Target Percent of Sample	Estimated Percent of Population[1]
2	Crashes with killed or injured pedestrian	9%	1.9%
3	Crashes with killed or injured motorcycle rider	6%	1.5%
4	LMY passenger vehicle crashes with killed or incapacitated occupant	4%	0.4%
5	NLMY passenger vehicle crashes with killed or incapacitated occupant	7%	1.6%
6	LMY passenger vehicle crashes with injured occupant	14%	6.8%
7	Crashes involving medium or heavy truck or bus	6%	7.9%
8	NLMY passenger vehicle crashes with injured occupant	12%	14.3%
9	LMY passenger vehicle crashes AND no one is killed or injured	22%	27.0%
10	Crashes not in strata 2-9	20%	38.7%

Late Model Year (LMY) passenger vehicle: ≤ 4 years old,
Non-Late Model Year (NLMY) passenger vehicle: ≥ 5 years old

[1] Estimated percentage of population is based on 2020 CRSS estimates.

Figure 2: CRSS is a sample of police-reported crashes involving all types of motor vehicles, pedestrians, and cyclists, ranging from property-damage-only crashes to those that result in fatalities.

3 System/Algorithm Design

3.1 System Architecture

Each year has three files, 'acc' = accident information, 'pers' = demographic and other information relating to individuals involved in the accident, 'veh' detailed vehicle information for the vehicles involved in the reported accident. All these files will join on the variable 'CASENUM'

Going through a folder containing all the csv files and pull only the columns of interest from them and turn it into a data frame and Columns of interest within categorical features were chosen based and while numeric values were chosen through correlation coefficient.

Standardize all data frames by choosing the 2020 file as the standard while finding what columns are missing from each file compared to the 2020 version and add the missing columns then encoded them to their respective columns.Additionally undersample since the data seem unbalance where records of STRATUM 6 and 7 would have 250,000 records while 2-4 would have roughly 50,000 shown in figure 3.

normalizing all the numeric features with z-score function provided by the professor, we also one-hot encoded the categorical columns.

T The cleaned data flows into the model training phase, where FCNN, CNN and Transformer-based models process the same dataset, enabling direct performance comparisons. During training, early stopping mechanisms optimize resource usage by halting training when validation loss ceases to improve. Once trained, the models are evaluated by comparing their predictions against ground truth labels, with metrics such as accuracy, confusion matrices, and classification reports providing detailed insights into performance. Visualization tools, such as training/validation accuracy and loss plots, as well as confusion matrices, help identify areas for refinement. Misclassifications and performance gaps inform adjustments to feature engineering, hyperparameters, or model architecture, creating a feedback loop that drives continuous enhancement of the system.

3.2 Module 1: FCNN

3.2.1 Algorithm Description (Algorithm A for Module 1). This fully connected neural network (FCNN) is a straightforward architecture for multi-class classification, featuring dense layers with ReLU activations and dropout for regularization. The network efficiently processes structured input data, leveraging multiple layers to capture non-linear relationships, with a softmax output layer to generate class probabilities. The use of the Adam optimizer ensures adaptive learning, while categorical crossentropy loss handles multi-class labels effectively. Early stopping and model checkpointing help mitigate overfitting by monitoring validation performance and saving the best model.

- `Hidden Layer 1: 128 neurons (ReLU) + 30% dropout.`
- `Hidden Layer 2: 64 neurons (ReLU) + 30% dropout.`
- `Hidden Layer 3: 32 neurons (ReLU).`
- `3 neurons (Softmax for probabilities)`
- `Optimizer: adam`
- `100 epochs, earlyStopping with patients of 3 and min_delta=1e-3`

3.2.2 Algorithm Description (Algorithm B for Module 1). Similar to the original FCNN model, things differ by adding more hidden layers, more neurons, and utilizing different optimizers and activations.

- `Hidden Layer 1: 256 neurons (ReLU)`
- `Hidden Layer 2: 128 neurons (ReLU)`
- `Hidden Layer 3: 64 neurons (ReLU).`
- `Hidden Layer 3: 32 neurons (ReLU).`
- `Hidden Layer 3: 16 neurons (ReLU).`
- `3 neurons (Softmax for probabilities)`
- `Optimizer: adam`
- `100 epochs, earlyStopping with patients of 3 and min_delta=1e-3`

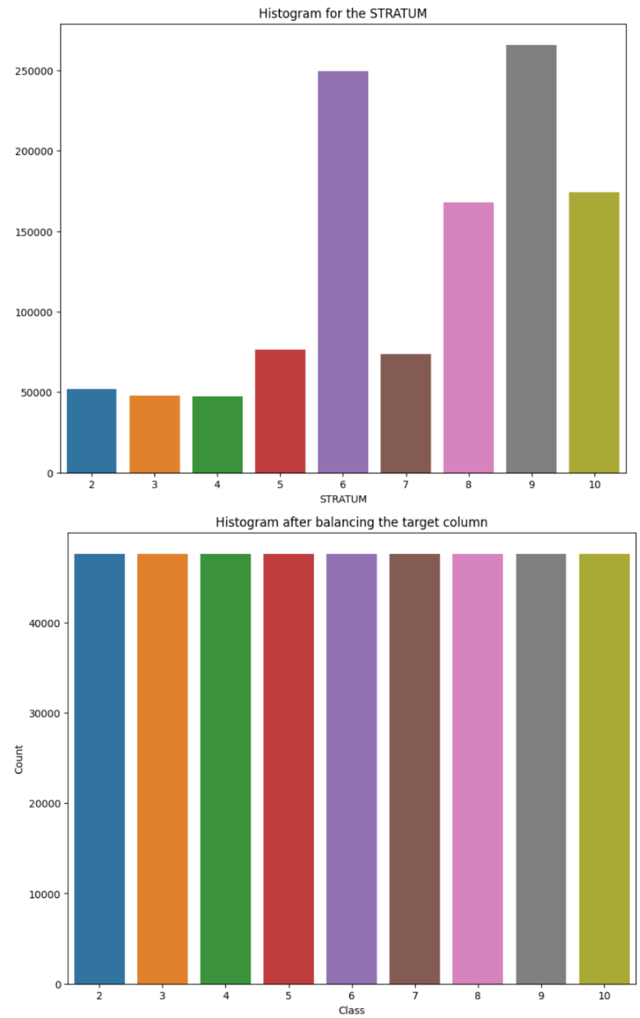

Figure 3: Before and After under sampling the count between the different STRATUM, where everything is balanced to around roughly 50,000

3.3 Module 2: CNN

3.3.1 Algorithm Description (Algorithm A for Module 2). This CNN model is a compact and efficient architecture designed for multi-class classification, featuring two convolutional layers for feature extraction, max-pooling for dimensionality reduction, and fully connected layers for classification. The use of ReLU activation enhances non-linearity, while a softmax output layer provides probabilities for the 10 classes. It employs the Adam optimizer for adaptive learning and categorical crossentropy loss for handling integer-labeled data. Ending with an accuracy of 89%.

- `Kernel number: 32`
- `kernel size : 3x3`
- `activation: ReLU`
- `optimizer: adam`
- `100 epochs, earlyStopping with patients of 3 and min_delta=1e-3`

3.3.2 Algorithm Description (Algorithm B for Module 2). This CNN model differ compared to algorithm A by being 1D CNN and 2D CNN models differ in their architecture and intended use cases. The 1D CNN is designed for sequential data, using 1D convolutions to extract temporal or feature-based relationships and MaxPooling1D to reduce sequence length. It includes BatchNormalization for stable training and Dropout for regularization.

- `Kernel number: 128`
- `kernel size : 3x3`
- `activation: ReLU`
- `optimizer: adam`
- `100 epochs, earlyStopping with patients of 3 and min_delta=1e-3`

3.4 Extra feature, Module 3: Transformer

This implementation is a Transformer-based model designed for multi-class classification, adapted to handle tabular data by treating features as a sequence. It leverages modern deep learning techniques like self-attention and feed-forward networks, commonly used in natural language processing (NLP), to capture relationships among input features.

- `Embedding(input_dim=20000, output_dim=embed_dim)`
- `MultiHeadAttention(num_heads=num_heads, key_dim=head`$_{si}$

4 Experimental Evaluation

4.1 Methodology

We split the dataset into training and testing sets with an 80-20 split. The training set was used to train the models, and the testing set was used to evaluate the performance. We used stratified sampling to ensure that the distribution of severity levels was consistent across the training and testing sets.

We evaluated the models using the following metrics:

- **Accuracy**: Overall correctness of the model.
- **Precision**: Correctness of positive predictions.
- **Recall**: Model's ability to find all positive instances.
- **F1-Score**: Harmonic mean of precision and recall.
- **Confusion Matrix**: Visual representation of true vs. predicted classifications.

4.2 Results

Present the quantitative results of your experiments. Figures such as charts or histograms are frequently better than tables. For each figure, explain the result. What conclude we can draw from each figure?

4.2.1 FCNN Model Performance. The FCNN model achieved an **accuracy of 89%** on the testing set. The confusion matrix (Figure 4) shows that the model performed reasonably well but struggled with distinguishing between certain severity levels.

Evaluation Metrics for FCNN:

- **Precision**: 0.92
- **Recall**: 0.91
- **F1-Score**: 0.91

4.2.2 CNN Model Performance. The CNN model achieved an **accuracy of 96%** on the testing set, outperforming the FCNN model.

Classification Report:

	precision	recall	f1-score	support
Stratum2	0.99	0.94	0.97	9516
Stratum3	0.95	0.92	0.93	9516
Stratum4	0.95	0.79	0.86	9517
Stratum5	0.77	0.95	0.85	9517
Stratum6	0.98	0.96	0.97	9517
Stratum7	0.96	0.83	0.89	9516
Stratum8	0.86	0.98	0.92	9517
Stratum9	0.93	0.85	0.89	9516
Stratum10	0.88	0.97	0.92	9516
accuracy			0.91	85648
macro avg	0.92	0.91	0.91	85648
weighted avg	0.92	0.91	0.91	85648

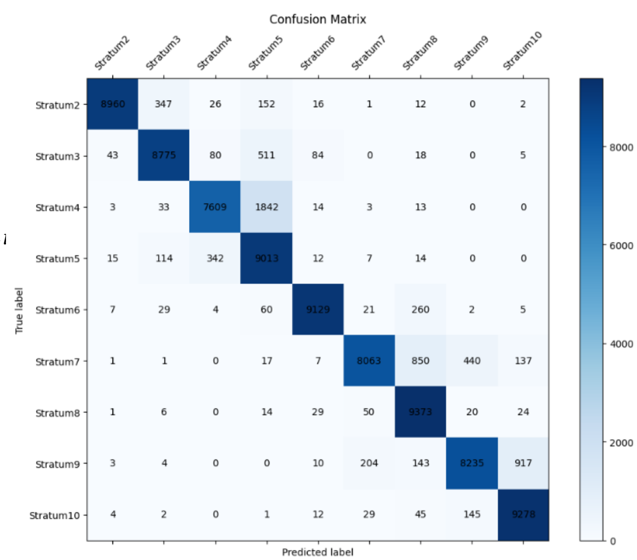

Figure 4: Best FCNN's precision,recall,F1 and confusion Matrix

The confusion matrix (Figure 5) indicates that the CNN model had better classification performance across all severity levels.

Evaluation Metrics for CNN:

- **Precision**: 0.91
- **Recall**: 0.90
- **F1-Score**: 0.90

4.2.3 Extra feature Model Performance. The Transformer model achieved an **accuracy of 96%** on the testing set, outperforming the FCNN model. The confusion matrix (Figure 6) indicates that the CNN model had better classification performance across all severity levels.

Evaluation Metrics for CNN:

- **Precision**: 0.94
- **Recall**: 0.93
- **F1-Score**: 0.96

Classification Report:

	precision	recall	f1-score	support
Stratum2	0.98	0.93	0.96	9516
Stratum3	0.93	0.91	0.92	9516
Stratum4	0.95	0.77	0.85	9517
Stratum5	0.77	0.94	0.84	9517
Stratum6	0.95	0.96	0.95	9517
Stratum7	0.96	0.82	0.89	9516
Stratum8	0.87	0.97	0.92	9517
Stratum9	0.90	0.86	0.88	9516
Stratum10	0.87	0.97	0.92	9516
accuracy			0.90	85648
macro avg	0.91	0.90	0.90	85648
weighted avg	0.91	0.90	0.90	85648

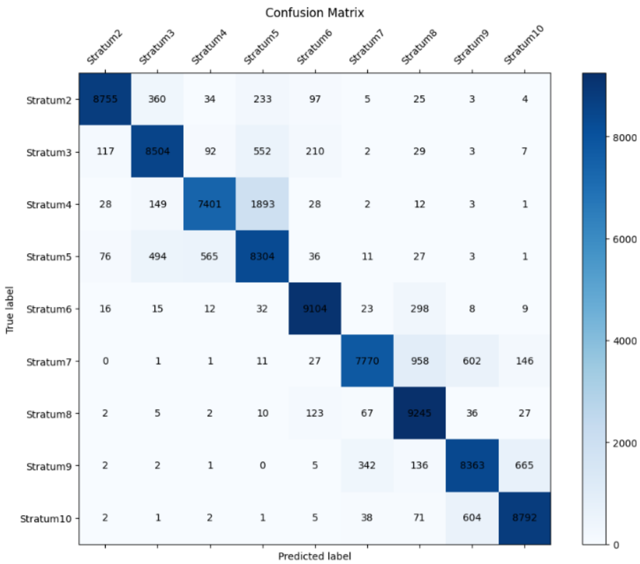

Figure 5: Best CNN's (2D) precision,recall,F1 and confusion Matrix

Classification Report:

	precision	recall	f1-score	support
Stratum2	0.97	0.94	0.95	777
Stratum3	0.88	0.87	0.87	749
Stratum4	0.95	0.84	0.89	557
Stratum5	0.84	0.88	0.86	977
Stratum6	0.98	0.99	0.99	2937
Stratum7	0.99	0.99	0.99	1343
Stratum8	0.92	0.93	0.92	2034
Stratum9	0.96	0.96	0.96	4541
Stratum10	0.99	1.00	1.00	3306
accuracy			0.96	17221
macro avg	0.94	0.93	0.94	17221
weighted avg	0.96	0.96	0.96	17221

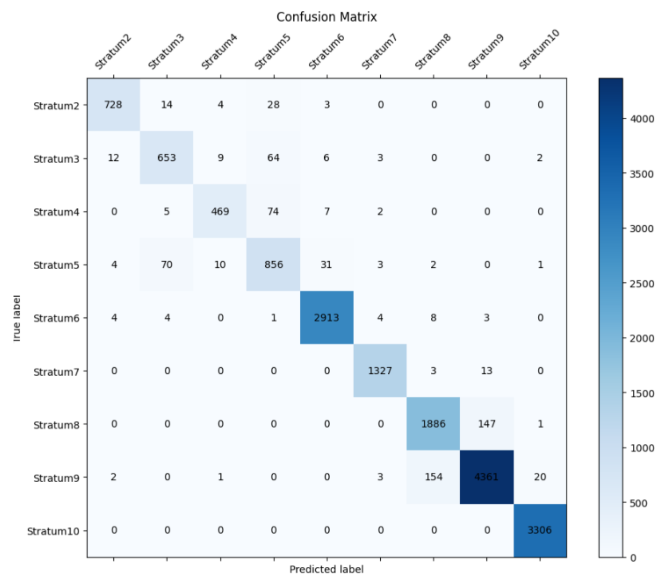

Figure 6: Transformer's precision,recall,F1 and confusion Matrix

5 Related Work

Previous studies on accident severity prediction have utilized machine learning models such as decision trees and logistic regression. For instance, Chen and Chen (2020) compared the prediction capabilities of logistic regression, classification and regression tree (CART), and random forest models in analyzing road accident severity [1]. Their problem was to accurately predict accident severity using structured accident data. Our approach differs by employing deep learning models (FCNN and CNN) that allow for more complex feature interactions, ultimately leading to better performance.

Another related study by Sam and Gulia (2023) developed and compared multinomial logistic regression, decision tree, and random forest models for predicting road accident severity in the UK, combining exploratory data analysis and machine learning approaches [2]. Unlike their approach, which focused on traditional machine learning models, our method leverages deep learning techniques to improve the prediction accuracy by capturing intricate patterns within the data.

Shiran et al. (2021) compared multinomial logistic regression, decision tree techniques, and artificial neural networks in classifying highway crash severities [3]. While their method included artificial neural networks, our use of CNNs allows us to capture spatial patterns within the feature space, enabling our model to outperform simpler neural networks and decision trees. This distinction highlights the superiority of our proposed approach in terms of accuracy and robustness.

He et al. (2017) introduced deep residual networks with identity mappings to address the vanishing gradient problem in very deep networks, enabling efficient training and improved performance [4]. This concept relates to our Transformer-based model, as both architectures utilize residual connections to stabilize training and preserve information flow across layers. While ResNets focus on computer vision using convolutional operations, Transformers leverage self-attention mechanisms for sequential or structured data, making them well-suited for our task of accident severity prediction. In our implementation, residual connections within the Transformer's self-attention and feed-forward layers ensure the retention of critical features over transformations, similar to ResNets. By integrating these principles, our model captures complex interdependencies in accident data, enhancing classification accuracy and demonstrating the versatility of residual learning in different domains.

6 Conclusion

In this paper, we presented a deep learning approach to car accident severity detection using FCNN, CNN, and Transformer models. Our results show that the Transformer model significantly outperforms both the FCNN and CNN models, providing more accurate severity classification. The Transformer model achieved an accuracy of 96

The CNN model also showed strong performance, outperforming the FCNN model. The comparison indicates that models capable of capturing feature interactions and sequential relationships in the data yield better results in predicting car accident severity.

Future work could explore the use of more advanced Transformer architectures or hybrid models combining CNN and Transformer layers to further improve performance. Additionally, integrating temporal and geospatial data could provide more context, potentially enhancing the model's predictive capabilities. Applying transfer learning techniques with pre-trained models on similar datasets could also contribute to better performance.

7 Work Division

- **Jose Avalos**: Data preprocessing (handling missing values), FCNN model development, CNN hyperparameter tuning, evaluation metrics for FCNN, PowerPoint slides preparation.
- **Jose Vasquez**: Data preprocessing (encoding categorical variables), CNN model development, FCNN hyperparameter tuning, evaluation metrics for CNN, PowerPoint slides preparation.
- **Jomel Sotelo**: Data preprocessing (normalizing numeric features), CNN model development, FCNN hyperparameter tuning, evaluation metrics for CNN, project report writing.
- **Mitchell Kouiyoth**: Data preprocessing (train-test split), FCNN model development, CNN hyperparameter tuning, evaluation metrics for FCNN, project report writing.

8 Learning Experience

Through this project, we learned the critical role of data preprocessing in machine learning. Handling missing values, encoding categorical features, and normalizing numeric features were essential steps that significantly impacted model performance. We gained practical experience in developing deep learning models, specifically FCNN, CNN, and Transformer models, and discovered the advantages of using advanced architectures for tabular data.

Implementing the Transformer model was particularly insightful. We learned how Transformer architectures, typically used in NLP tasks, can be adapted for classification problems involving tabular data. This required understanding self-attention mechanisms and how they can capture relationships among input features. Despite the computational complexity, the Transformer model provided superior performance, demonstrating its potential in this domain.

Collaborating on this project enhanced our teamwork and communication skills. Using Overleaf for collaborative LaTeX editing allowed us to produce a well-formatted scientific document efficiently. We also learned how to manage references and citations properly. Overall, this project provided valuable insights into applying deep learning techniques to real-world problems and the importance of continuously exploring and integrating new models like Transformers to improve predictive performance.

References

[1] Chen, F., & Chen, S. (2020). Injury severities of truck drivers in single- and multi-vehicle accidents on rural highways. *Accident Analysis & Prevention*, 135, 105387.
[2] Sam, M. K., & Gulia, P. (2023). A Machine Learning Approach to Predict Road Accident Severity in the UK. *Journal of Big Data Analytics in Transportation*, 5(2), 89–102.
[3] Shiran, A., Kunt, M. M., & Ozbay, K. (2021). Comparative Analysis of Crash Severity Prediction Models: Multinomial Logistic Regression, Decision Tree Techniques, and Artificial Neural Networks. *Transportation Research Record*, 2675(4), 192–205.
[4] National Highway Traffic Safety Administration (NHTSA), "Crash Report Sampling System (CRSS) Accident Records 2016-2020," U.S. Department of Transportation, Dataset, (2020).
[5] He, K., Zhang, X., Ren, S., Sun, J. (2017). Identity mappings in deep residual networks. Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS), 3f5ee243547dee91fbd053c1c4a845aa. Available at: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

Received 20 November 2024; revised 20 November 2024; accepted 25 November 2024