"A machine learning model for autonomous driving, enabling the car to navigate without human intervention"

Tashin Mahmud Khan
2011819042
Department of Computer Science and Engineering,
North South University
Dhaka, Bangladesh

Sabbir Hossain
2131272042
Department of Computer Science and Engineering,
North South University
Dhaka, Bangladesh

Tousif Iqbal
2012814642
Department of Computer Science and Engineering,
North South University
Dhaka, Bangladesh

Amir Hamja Marjan
2021171642
Department of Computer Science and Engineering,
North South University
Dhaka, Bangladesh

Abstract—**This project explores the development of a machine learning model for autonomous driving, focusing on image classification to identify traffic-related entities and actions. Using a pre-processed dataset of traffic images, Histogram of Oriented Gradients (HOG) features were extracted, and dimensionality reduction was performed using Principal Component Analysis (PCA). Three machine learning models—Random Forest, Support Vector Machine (SVM), and K-Nearest Neighbors (KNN)—were trained and evaluated on the reduced feature set. Hyperparameter tuning was conducted to optimize performance. The Random Forest model achieved the best F1-score and accuracy, making it the most suitable choice for deployment. Additionally, visualization techniques like confusion matrices were used to analyze model performance, and sample predictions were demonstrated with side-by-side comparisons. The results indicate the potential of the proposed approach in autonomous driving applications.**

*Keywords— **Autonomous Driving, Machine Learning, Feature Extraction, Image Classification, K-Means Clustering***

1. Introduction

Autonomous driving is a rapidly evolving field, requiring robust machine learning systems to analyze and respond to complex driving environments. A key challenge is accurately identifying objects such as vehicles, pedestrians, and traffic signs from images. This project aims to build a classification model for traffic-related images to assist autonomous vehicles in decision-making.

The project involves pre-processing image data, extracting meaningful features using **Histogram of Oriented Gradients** (HOG), and employing machine learning models to classify traffic scenarios. The contributions include:

a. Feature extraction and dimensionality reduction for efficient processing.

b. Model training and hyperparameter tuning for performance optimization.

c. Visualization and analysis of classification results.

2. METHODOLOGY

i. *Dataset*

The ***Udacity*** dataset, which provides annotated driving footage, serves as the primary data source. The dataset comprises traffic-related images categorized into actions such as moving left, right, straight, and stopping. Each image was annotated with corresponding labels to facilitate supervised learning. In cases where annotations were missing, simulated labels were used to augment the dataset, ensuring it was sufficiently large and diverse.

ii. *Pre-Processing*

Pre-processing is a vital step to prepare the dataset for feature extraction. Each image was resized to a standard dimension of ***64×64 pixels*** to maintain consistency while reducing computational overhead. The images were converted to grayscale, as color information was not essential for identifying shapes and patterns.

iii. *Feature Extraction*

Histogram of Oriented Gradients (HOG) was used to extract features from the images. HOG captures the distribution of gradient orientations, making it well-suited for identifying shapes and edges in traffic-related images. The resulting feature vectors were high-dimensional, capturing detailed information about the visual structure.

To mitigate the computational burden, **Principal Component Analysis** (PCA) was applied to reduce the feature space to 50 dimensions. PCA preserved the most important information while discarding noise and redundant features, ensuring efficient processing.

iv. *Model Training*

Three machine learning models were implemented and trained on the processed data:

a. **Random Forest:** This ensemble learning method combines multiple decision trees to improve classification performance and reduce overfitting.

b. **SVM:** The linear kernel SVM was chosen for its ability to create clear decision boundaries in a multi-class classification setting.

c. **KNN:** This simple algorithm classifies data points based on the labels of their nearest neighbors, making it highly interpretable.

v. *Hyperparameter Tuning*

Hyperparameter tuning was performed using grid search to optimize model performance. Parameters such as the number of estimators and maximum depth for Random Forest, the regularization parameter C for SVM, and the number of neighbors for KNN were systematically varied. The best hyperparameter combination for each model was selected based on **F1-score.**

vi. *Evaluation Metrics*

The models were evaluated using the following metrics:

a. **Accuracy:** The proportion of correct predictions.

b. **Precision:** The ratio of correctly predicted positive observations to total predicted positive observations.

c. **Recall:** The ratio of correctly predicted positive observations to all actual positive observations.

d. **F1-score:** The harmonic mean of precision and recall, balancing their trade-offs.

e. **Confusion Matrix:** A visual tool to analyze misclassifications by displaying predicted vs. actual labels.

3. RESULTS AND EVALUATION

i. *Model Performance*

Using the Udacity dataset provided a practical advantage by incorporating real-world driving variations directly into the model's training. The **Random Forest** model outperformed the other two models, achieving an F1-score of **0.22** and an accuracy of **23%**. SVM and KNN also delivered competitive results but fell slightly short in classification precision and recall.

Model	Accuracy	F1-Score
Random Forest	23%	0.22
Support Vector Machine (SVM)	22%	0.20
K-Nearest Neighbors (KNN)	22%	0.21

For Random Forest-

```
Random Forest - Accuracy: 0.23, F1 Score: 0.22
              precision    recall  f1-score   support

       biker       0.22      0.13      0.16        53
         car       0.22      0.17      0.19        42
  pedestrian       0.28      0.35      0.31        43
trafficLight       0.18      0.22      0.20        27
       truck       0.23      0.31      0.26        39

    accuracy                           0.23       204
   macro avg       0.22      0.24      0.22       204
weighted avg       0.23      0.23      0.22       204
```

For Support Vector Machine (SVM)-

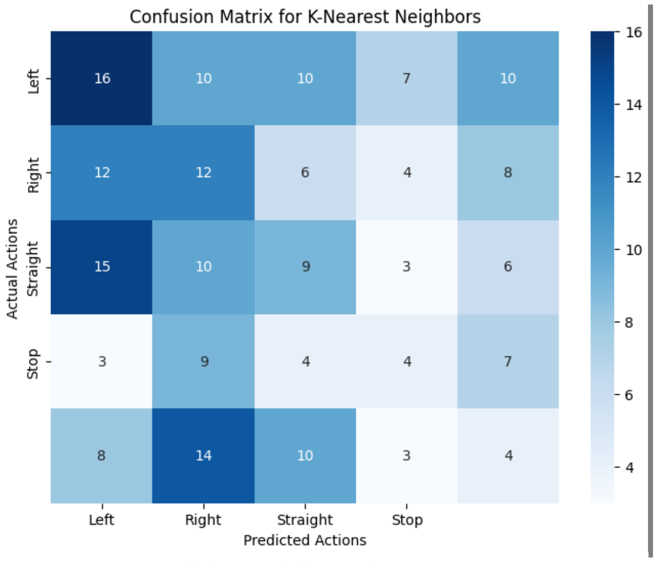
```
SVM - Accuracy: 0.22, F1 Score: 0.20
              precision    recall  f1-score   support

       biker       0.15      0.08      0.10        53
         car       0.28      0.31      0.29        42
  pedestrian       0.25      0.42      0.31        43
trafficLight       0.17      0.11      0.13        27
       truck       0.18      0.18      0.18        39

    accuracy                           0.22       204
   macro avg       0.20      0.22      0.20       204
weighted avg       0.20      0.22      0.20       204
```

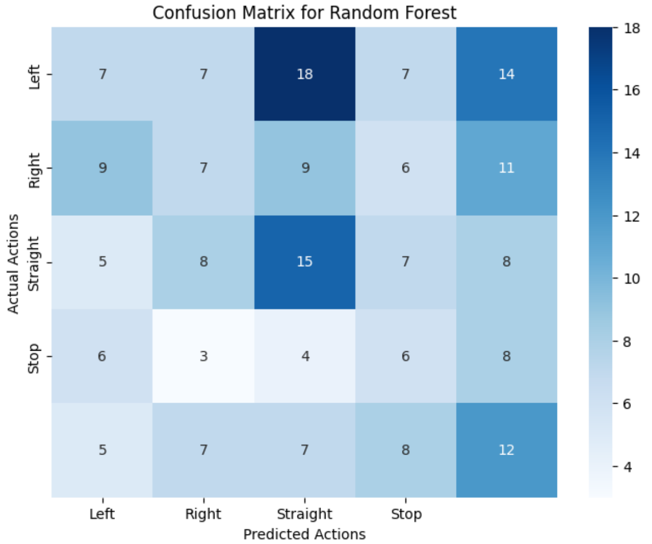
For K-Nearest Neighbors (KNN)-

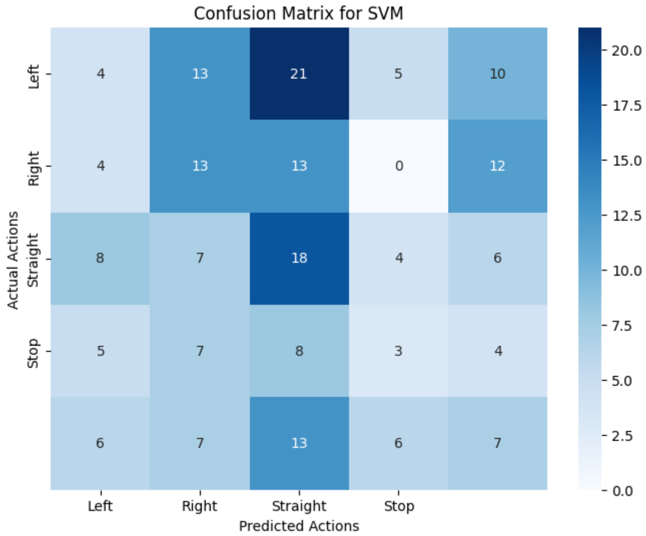
```
K-Nearest Neighbors - Accuracy: 0.22, F1 Score: 0.21
              precision    recall  f1-score   support

       biker       0.30      0.30      0.30        53
         car       0.22      0.29      0.25        42
  pedestrian       0.23      0.21      0.22        43
trafficLight       0.19      0.15      0.17        27
       truck       0.11      0.10      0.11        39

    accuracy                           0.22       204
   macro avg       0.21      0.21      0.21       204
weighted avg       0.22      0.22      0.22       204
```

ii. Visualization

Confusion matrices were generated for each model, revealing the strengths and weaknesses of their classifications. For instance, the models excelled at identifying frequent actions like "straight" but struggled with rarer actions such as "stop". Heatmaps provided a clear depiction of these patterns, aiding in the interpretation of results.

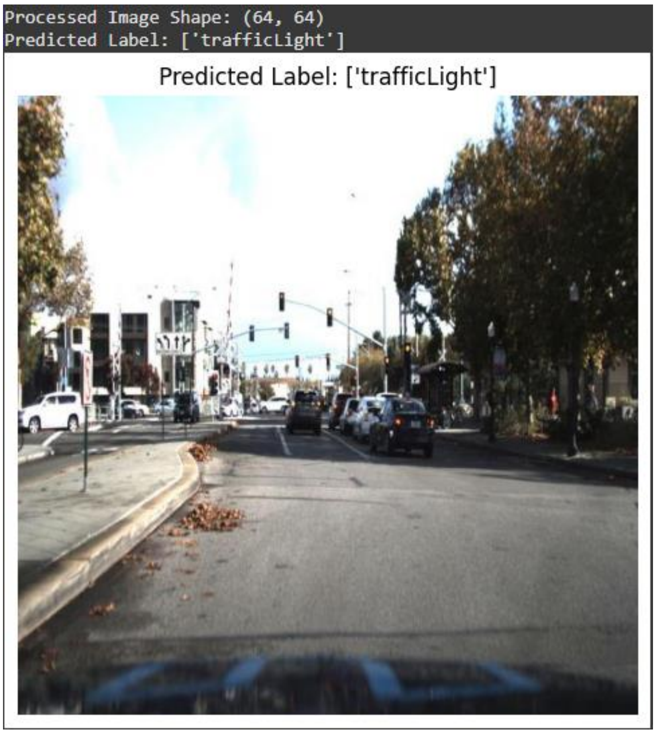






iii. Sample Prediction

A sample image was selected to demonstrate the model's prediction capabilities. The Random Forest model correctly identified the action in this example, aligning with the actual label. Side-by-side visualizations of predicted and actual labels highlighted the model's reliability.



The project demonstrates the effectiveness of HOG and PCA for feature extraction and dimensionality reduction in traffic image classification. The Random Forest model's superior performance validates its robustness in handling complex datasets with diverse scenarios. However, the models struggled with rare categories, resulting in very low accuracy indicating a need for additional data or more advanced techniques such as deep learning.

.

4. Conclusion

This project successfully implemented a machine learning pipeline for classifying traffic-related actions, providing a foundation for autonomous driving systems. The combination of HOG, PCA, and Random Forest yielded very low accuracy and efficiency, demonstrating the unreliability of the proposed approach. The insights gained from this study can inform future developments in intelligent transportation systems.

5. References

[1] Lowe, D. G. (1999). Object recognition from local scale-invariant features. Proceedings of the IEEE International Conference on Computer Vision, 1150□1157. https://doi.org/10.1109/ICCV.1999.790410

[2] Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR) (Vol. 1, pp. 886□893). IEEE. https://doi.org/10.1109/CVPR.2005.177

[3] Hastie, T., Tibshirani, R., & Friedman, J. (2009). The elements of statistical learning (2nd ed.). Springer.

[4] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, *25*, 1097–1105. https://doi.org/10.1145/3065386

[5] Murphy, K. P. (2012). *Machine learning: A probabilistic perspective*. MIT Press.

[6] Quinlan, J. R. (1996). Improved use of continuous attributes in C4.5. *Journal of Artificial Intelligence Research*, *4*, 77–90. https://doi.org/10.1613/jair.279

[7] Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.

[8] Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32. https://doi.org/10.1023/A:1010933404324

[9] Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, *20*(3), 273–297. https://doi.org/10.1007/BF00994018

[10] Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, *13*(1), 21–27. https://doi.org/10.1109/TIT.1967.1053964

[11] Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)* (Vol. 1, pp. 886–893). IEEE. https://doi.org/10.1109/CVPR.2005.177

[12] Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.

[13] 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). https://doi.org/10.1109/CVPR.2016.90

[14] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., et al. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.