VIETNAM NATIONAL UNIVERSITY HOCHIMINH CITY
UNIVERSITY OF INFORMATION TECHNOLOGY
ADVANCED PROGRAM IN INFORMATION SYSTEMS

TO LY TIEN DAT – TRINH VINH DAI

TRAFFIC SIGN DETECTION SYSTEM BASE ON DEEP LEARNING

CLASS: NT114.O21.MMCL

HO CHI MINH CITY, 2024

NATIONAL UNIVERSITY HOCHIMINH CITY
UNIVERSITY OF INFORMATION TECHNOLOGY
ADVANCED PROGRAM IN INFORMATION SYSTEMS

TO LY TIEN DAT - 21520712
TRINH VINH DAI - 21521915

TRAFFIC SIGN DETECTION SYSTEM BASE ON DEEP LEARNING

THESIS ADVISOR
MSC NGUYEN KHANH THUAT

HO CHI MINH CITY, 2024

ACKNOWLEDGMENTS

First and foremost, I express my heartfelt gratitude to MsC Nguyen Khanh Thuat, my thesis advisor, for her invaluable guidance, support, and mentorship throughout this research endeavor. MsC Nguyen Khanh Thuat's expertise, encouragement, and unwavering commitment to excellence have been instrumental in shaping the direction and success of this project. I am deeply thankful for her insightful feedback and dedication to my academic growth.

I extend my sincere appreciation to my co-author, Trinh Vinh Dai, whose collaborative efforts, expertise, and dedication have significantly enriched this thesis. Working alongside Trinh Vinh Dai has been both inspiring and rewarding, and I am grateful for the opportunity to collaborate with such a talented individual.

Special thanks are due to the faculty members of the Department of Computer Networks and Data Communications at the University of Information Technology for their support, encouragement, and provision of essential resources throughout this research journey.

I am also indebted to my family and friends for their unwavering support, understanding, and encouragement during the challenging phases of this project. Their love and encouragement have been my source of strength and motivation.

Lastly, I express my gratitude to everyone who has contributed to this endeavor, directly or indirectly. Your support, belief in our work, and collaborative efforts have been invaluable.

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

The thesis investigates the automatic recognition and classification of traffic signs in the distinctive traffic environment of Vietnam through the use of cutting-edge DL techniques, namely the YOLOv4 (You Only Look Once version 4) model in conjunction with the Darknet framework. Moreover, the project incorporates this paradigm into the NVIDIA Jetson Nano, an embedded device, to facilitate real-time processing and deployment in useful applications.

The first step in the study process is to compile a large dataset of photos of traffic signs that accurately depict the variety of difficult and demanding road conditions in Vietnam. To improve this dataset's resilience, the model is rigorously preprocessed using augmentation and normalization. YOLOv4 is a model that balances speed and accuracy in real-time object recognition, and it is used in the Darknet framework.

The trained YOLOv4 model is used to run on the NVIDIA Jetson Nano, a potent and reasonably priced embedded system built for AI apps. Through this connection, the model is tuned to operate optimally on the Jetson Nano, utilizing its GPU capabilities to expedite processing. YOLOv4, Darknet, and Jetson Nano work together to process images quickly and precisely, which makes the system ideal for real-time applications in traffic systems.

The thesis addresses a number of important issues, one of which is the variance in traffic sign appearances caused by varying weather, illumination, and physical impediments. Because of its sophisticated features, which include spatial pyramid pooling, path aggregation network, and CSPDarknet53 as its backbone, the YOLOv4 model can withstand these fluctuations and maintain excellent accuracy and dependability across a variety of scenarios.

The findings show that the YOLOv4 model can efficiently and accurately detect and categorize traffic indicators when used in conjunction with the Darknet framework on the Jetson Nano. With real-time sign identification for driver assistance systems, this capacity can improve traffic safety significantly. It can also aid in the development of autonomous vehicles that can navigate the intricate road conditions of Vietnam.

In summary, this thesis not only improves the field of DL-based traffic sign identification, but it also offers workable, scalable solutions that are customized to Vietnam's unique requirements. The initiative seeks to greatly increase traffic control and safety nationwide by merging YOLOv4 with Darknet and implementing it on the Jetson Nano.

LIST OF ACRONYMS

ITS: Intelligent Transport Systems

GPS: Global Positioning Systems

AI: Artificial Intelligence

DL: Deep Learning

DIY: Do It Yourself

PL: Packet Loss

Part 1: INTRODUCTION

1.1. Rationale for choosing the topic

Currently, self-driving cars are a trend in the world, they are gradually becoming popular in many places [1]-[3] and Vietnam is no exception. The need for an accurate traffic sign detection system is indispensable. Therefore, my group chose this as a research and development project in Vietnam

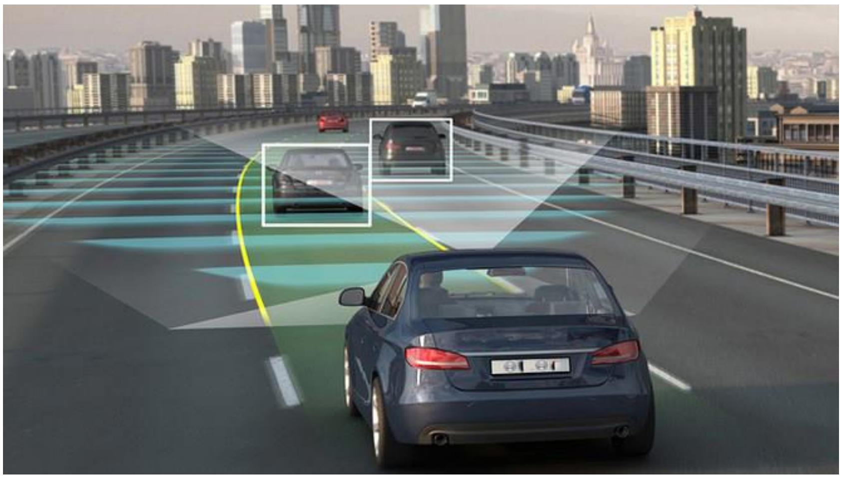

Figure 1. Current trend of self-driving cars

1.2. Research Objectives

- **Improving traffic safety:** Recognizing and identifying traffic signs is a crucial factor in ensuring road safety. By deploying ITS to automatically recognize traffic signs, the system can provide accurate and timely information to drivers about road rules and restrictions, thereby reducing the risk of traffic accidents. [4]
- **Enhancing compliance with traffic laws:** Traffic sign recognition helps drivers to recognize road rules and restrictions, ensuring compliance with traffic regulations. The ITS system can provide warnings or guidance to drivers when they do not comply with signs, ensuring adherence to traffic laws and improving safety for all road users. [5]
- **Optimizing traffic flow:** By recognizing traffic signs, the ITS system can analyze and collect information about traffic conditions. Based on this information, the system can make intelligent decisions regarding traffic signal adjustments, vehicle

routing, and traffic flow optimization. This helps reduce congestion and increase mobility on the roads.

- **Creating a smart traffic environment:** Deploying ITS for traffic sign recognition is a step towards building an intelligent traffic system. This system will integrate other technologies such as vehicle-to-everything communication (V2X) [6], GPS, and AI to provide smart traffic information and services to drivers and traffic operators.
- **Saving time and energy:** By providing accurate information about traffic signs, the ITS system helps drivers choose optimal routes, avoid traffic issues, and save travel time. This can also minimize fuel consumption and emissions, contributing to environmental protection.

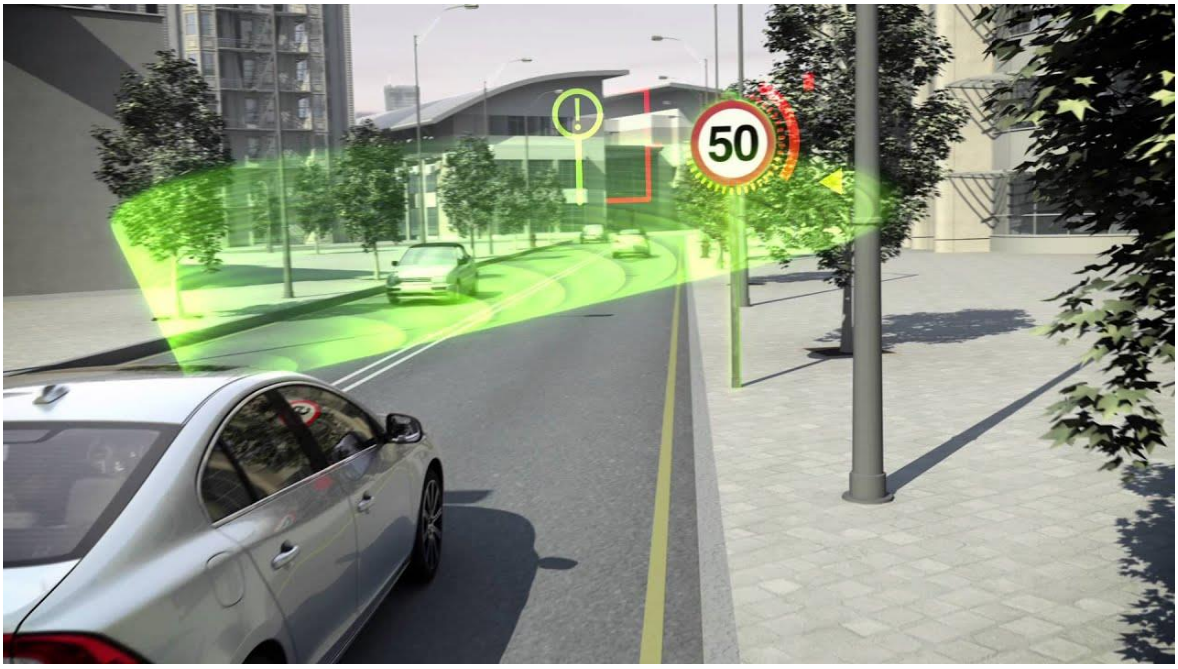

Figure 2 Traffic Sign Recognition using Pytorch and CNN [7]

Part 2: CONTENT

2.1. Used methodology

- Using machine learning technology: Let the machine learn automatically.
- DL technology [8]: Using machine artificial neural networks.
- Use Yolov4 with the open source darknet framework.

2.1.1. Darknet framework [9]

- DL Framework:
  - Darknet is a DL framework written in C and CUDA, optimized for high performance and fast processing on GPUs. This accelerates the training and inference processes of DL models.
- CUDA and GPU Support:
  - Darknet is optimized for CUDA and GPU usage, which helps speed up the training and deployment process. This support is crucial for applications requiring fast and efficient image processing.

2.1.2. YOLOv4

YOLOv4 stands for You Only Look Once version 4 [10] [11]. It is a real-time object detection model developed to address the limitations of previous YOLO versions such as YOLOv3 and other object detection models. other statue. YOLOv4 applies not only to recommender systems but also to independent process management and human input reduction. Its operation on conventional graphics processing units (GPUs) allows for affordable mass adoption, and it is designed to operate in real time on conventional GPUs while requiring only a Such GPU for training.

2.2. Project implementation

2.2.1. About dataset

- We use the Vietnam signage dataset on the roboflow website.[12]
- Dataset includes: 58 classes, 3680 images trong đó :
  - Train set: 2280 images
  - Valid set: 786 images

- Test set: 614 images

2.2.2. Training and evaluating the model

2.2.2.1. Training

- In this topic, we use Yolov4 technology to train this model.
- First, we used Google Colab [13] to train because this is an environment that already has the most popular machine learning packages and DL frameworks installed.
- But the downside of google colab is that it will limit GPU capacity, so training time will not be much, leading to PL still occurring, and many error signs being detected.
- For the above reason, my group used Kaggle [14] to train this model.

Figure 3. Detected a faulty sign

Figure 4. Detect signs accurately

Figure 5. Detect signs accurately with 2 signs

2.2.2.2. Evaluation [15]

- Our team tested the results of model training using Yolov4 on two frameworks: Darknet and Pytorch.
- On the other hand, we use WanDB [16] to monitor, manage and analyze DL models .
- The result of Pytorch:

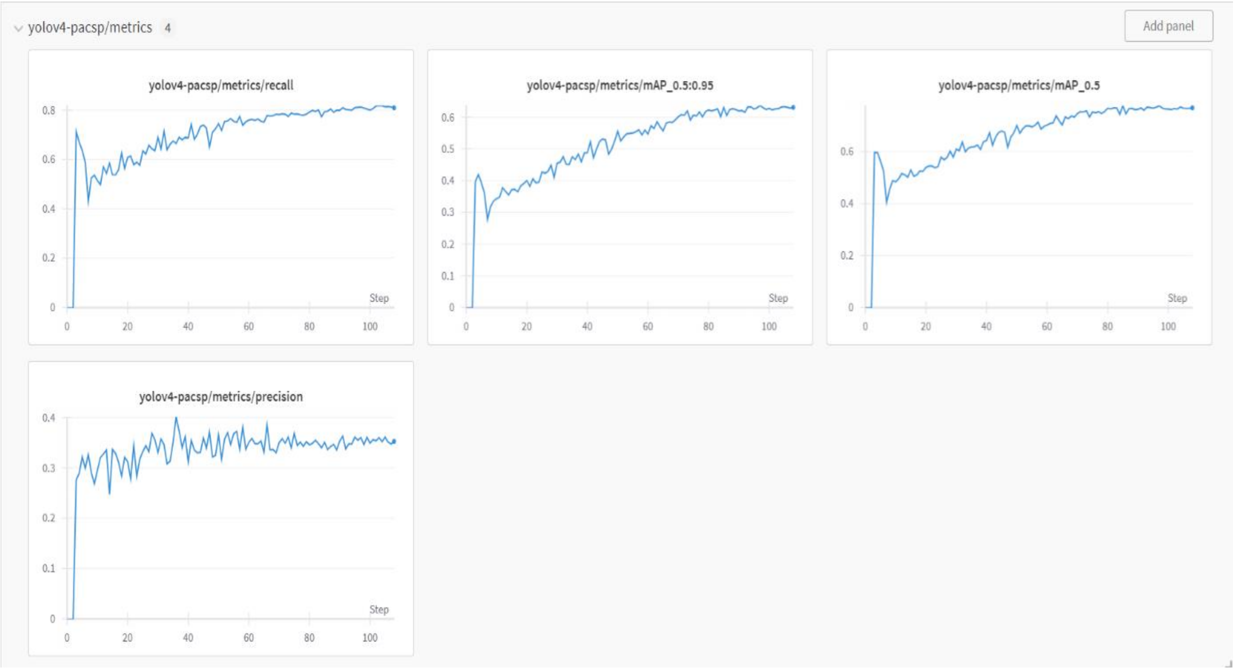

Figure 6. Metric – Result of Pytorch

Recall: The chart shows the change in Recall over steps. Recall gradually increases and reaches a stable level at around 0.75-0.8.

mAP@0.5:0.95: The chart shows Mean Average Precision (mAP) over steps, with an IoU threshold from 0.5 to 0.95. mAP also gradually increases and reaches around 0.5-0.6.

mAP@0.5: This chart shows mAP with an IoU threshold of 0.5. This value gradually increases and stabilizes at around 0.65-0.7.

Precision: This chart shows the Precision of the model over steps. Precision increases and stabilizes at around 0.35-0.4.

- From the above graphs, it can be observed that:
    - **Recall**: The high stable recall value indicates that the model can detect most objects in the test set.
    - **mAP**: mAP is a general metric for evaluating the object detection quality of the model. mAP@0.5:0.95 is usually lower than mAP@0.5 due to higher requirements for the bounding box accuracy.
    - **Precision**: Precision tends to stabilize, indicating that the model has a low confusion rate between object classes.

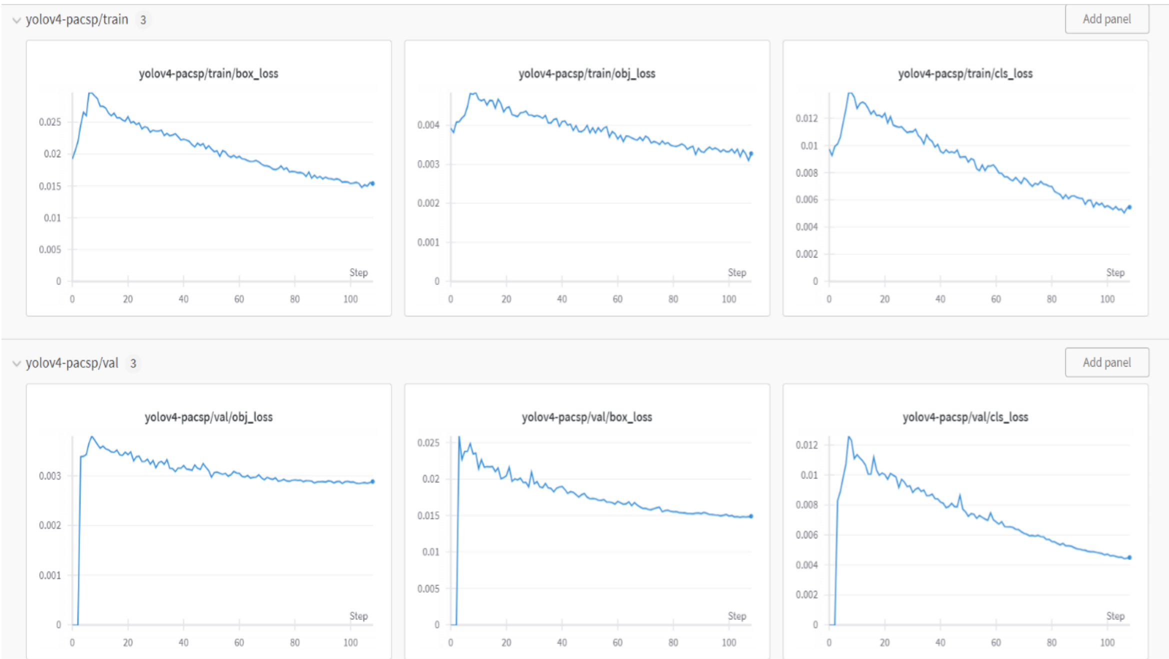

Figure 7. Train Loss and Validation Loss – Result of Pytorch

<u>Training Loss</u>

Box Loss: Initially high, then gradually decreases and stabilizes, indicating improvement in predicting bounding box positions.

Object Loss: Increases briefly, then consistently decreases, showing better object distinction over time.

Class Loss: Steadily decreases, reflecting improved object classification.

<u>Validation Loss</u>

Object Loss: Initial spike, then decreases and stabilizes, mirroring the training phase.

Box Loss: Decreases steadily, showing better bounding box predictions.

Class Loss: Consistently decreases, indicating improved classification on the validation set.

⇨ **Conclusion:** The declining loss values in both training and validation phases suggest that the model is learning effectively and stabilizing, indicating good performance in object detection and classification.

- The result of Darknet:

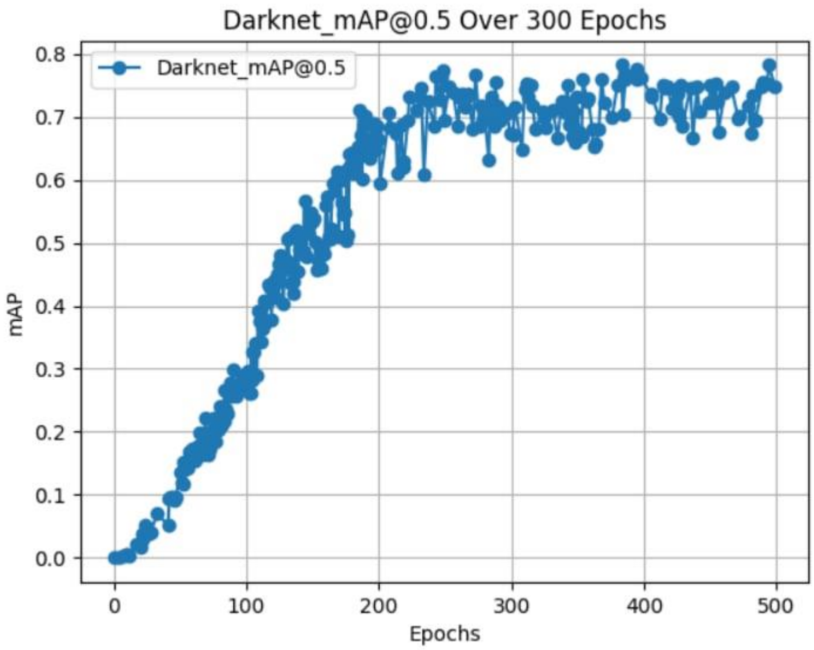

Figure 8. Epochs of Darknet

- This plot shows the performance of a model trained using Darknet, with the mean Average Precision (mAP) at an IoU threshold of 0.5 (mAP@0.5) over 500 epochs.
  - **x-axis (Epochs)**: The number of training epochs, ranging from 0 to 500.
  - **y-axis (mAP)**: The mean Average Precision, ranging from 0 to 0.8.
- The plot indicates that the mAP improves significantly during the initial epochs, with rapid gains up to around epoch 200. After this point, the improvements continue more slowly, with some fluctuations, and it appears to plateau around an mAP of 0.7 to 0.75.

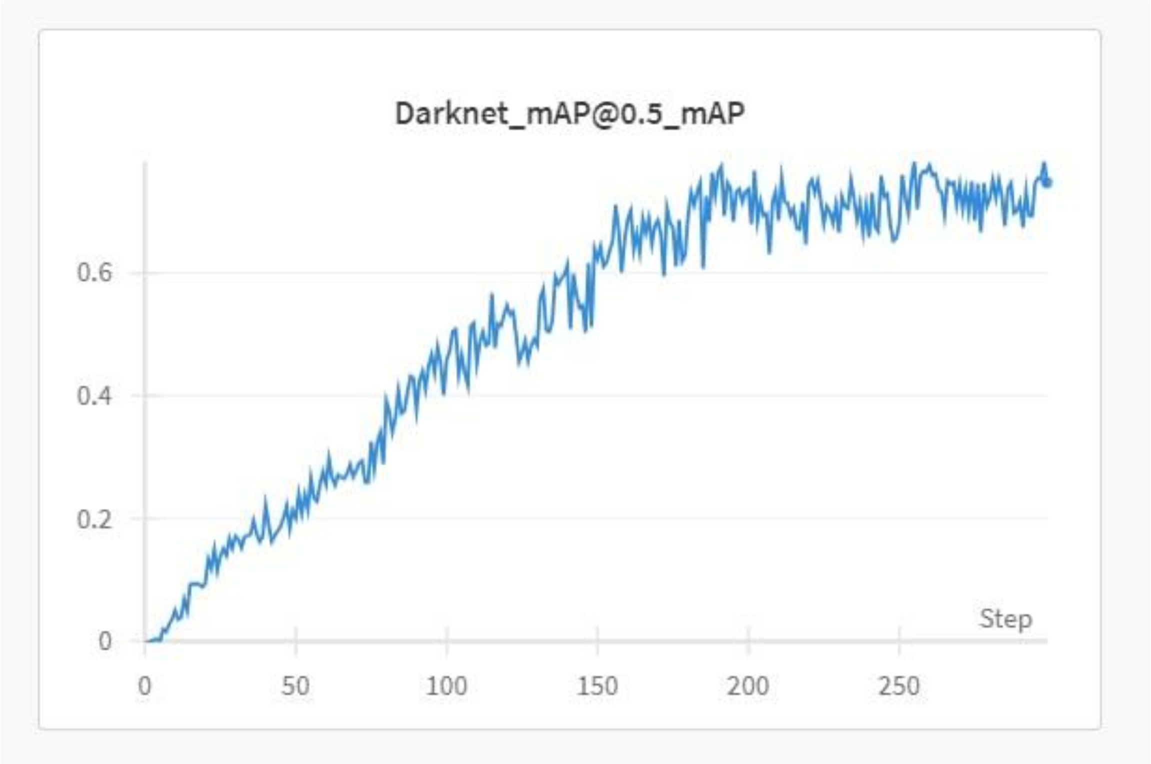

Figure 9. mAp at 0.5 – Result of Darknet

- **Evaluation:**
    - Early training phase (0-50 steps):
        - The mAP value quickly increases from around 0 to 0.2. This indicates that the model is learning basic features from the data.
    - Mid training phase (50-200 steps):
        - The mAP value continues to increase, but at a slower rate, reaching around 0.6. This is the phase where the model is learning and refining more complex features.
    - Late training phase (200-300 steps):
        - The mAP value tends to fluctuate around 0.6-0.7, indicating that the model has reached a stable state and is no longer making significant improvements with each step.

Part 3: DEVELOPMENT DIRECTION

3. Embed the model into the Jetson Nano device

3.1 Jetson Nano device

Jetson Nano is a compact, powerful computer developed by NVIDIA. It is designed to support artificial intelligence (AI) and machine learning applications. Here is some basic information about Jetson Nano:

Purpose:

Jetson Nano is designed to provide powerful AI processing for embedded devices and DIY projects. It is ideal for applications such as image recognition, video processing, and robot control.

Hardwareconfiguration:

CPU: 4-core ARM Cortex – A57 processor.

GPU: NVIDIA Maxwell with 128 CUDA cores.

RAM 4GB LPDDR4.

Memory: Supports microSD card to store operating system and data.

Connectivity: Includes USB, Ethernet, HDMI, and I/O interfaces for connecting to sensors and peripherals. [17]

Software:

Jetson Nano uses the NVIDIA JetPack software platform, which includes development kits and NVIDIA AI libraries such as TensorRT, cuDNN, and VisionWorks. Supports the Ubuntu operating system and can run popular AI frameworks such as TensorFlow, PyTorch, Caffe, and MXNet.

Application:

Jetson Nano is often used in robotics projects, drones, autonomous vehicles, and smart surveillance systems.

It is also used in education and research to develop and test AI and machine learning models.

In this project, we use Jetson Nano B01.

3.2 Embed the model

3.2.1 Optimizing the YOLOv4 Model

YOLOv4 is a computationally intensive DL model, so optimizing it to fit within the computational limits of Jetson Nano is crucial. This may involve using tools like TensorRT to compile the model into an optimized format suitable for Jetson Nano's GPU

3.2.2 Integration with CUDA and GPU Optimization

Jetson Nano supports CUDA and has an integrated GPU, enabling you to leverage these features to optimize both training and inference processes for the YOLOv4 model on Jetson Nano.

3.2.3 Building User Interface and System Integration

Once successfully embedded, consider building a user interface or integrating it into an automated system for object detection and control purposes.

3.2.4 Testing and Performance Evaluation

Finally, thorough testing and performance evaluation of the YOLOv4 model on Jetson Nano are essential to ensure proper functionality and meet project requirements.

REFERENCES

[1] H. Luo, Q. Kong, and F. Wu, ''Traffic sign image synthesis with generative adversarial networks,'' in Proc. Int. Conf. Pattern Recognit. (ICPR), Aug. 2018, pp. 2540–2545.

[2] C. Liu, S. Li, F. Chang, and Y. Wang, ''Machine vision based traffic sign detection methods: Review, analyses and perspectives,'' IEEE Access, vol. 7, pp. 86578–86596, 2019.

[3] C. Dewi, R.-C. Chen, and S.-K. Tai, ''Evaluation of robust spatial pyramid pooling based on convolutional neural network for traffic sign recognition system,'' Electronics, vol. 9, no. 6, p. 889, May 2020.

[4] Djarot Hindarto, "Enhancing Road Safety with Convolutional Neural Network Traffic Sign Classification," ResearchGate, Jul. 2023; https://www.researchgate.net/publication/375710164_Enhancing_Road_Safety_with_Convolutional_Neural_Network_Traffic_Sign_Classification.

[5] Yasmin Roper, Mark Rowland, Zoran Chakich, William McGIll, Vinuka Nanayakkara, David Young, Russell Whale, "Implications of Traffic Sign Recognition (TSR) Systems for Road Operators", p.51

[6] Syed Adnan Yusuf, "Vehicle-to-everything (V2X) in the autonomous vehicles domain – A technical review of communication, sensor, and AI technologies for road user safety", in ScienceDirect ,2023

[7] Sachin Sarkar, "Traffic Sign Recognition using Pytorch and CNN "; https://www.kaggle.com/code/sachinsarkar/traffic-sign-recognition-using-pytorch-and-cnn

[8] L. Abdi and A. Meddeb, ''DL traffic sign detection, recognition and augmentation,'' in Proc. Symp. Appl. Comput., Apr. 2017.

[9] Alexey AB, *darknet*; 2021; https://github.com/AlexeyAB/darknet

[10] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, ''YOLOv4: Optimal speed and accuracy of object detection,'' 2020, arXiv:2004.10934. [Online]. Available: http://arxiv.org/abs/2004.10934

[11] CHRISTINE DEWI, RUNG-CHING CHEN, (Member, IEEE), YAN-TING LIU,XIAOYI JIANG, (Senior Member, IEEE), KRISTOKO DWI HARTOMO. *Yolo V4 for Advanced Traffic Sign Recognition With Synthetic Training Data Generated by Various GAN* (Department of Information Management, Chaoyang University of Technology, Taichung 41349, Taiwan. Faculty of Information Technology, Satya Wacana Christian University, Central Java, Salatiga 50711, Indonesia. Department of Mathematics and Computer Science, University of Münster, 48149 Münster, Germany)

[12] Roboflow Universe; *Vietnam-Traffic-Sign-Dection*; https://universe.roboflow.com/vietnam-traffic-sign-detection/vietnam-traffic-sign-detection-2i2j8/dataset/6

[13] IEEE Access, " Performance Analysis of Google Colaboratory as a Tool for Accelerating DL Applications", 07 October 2018, p. 61677 – 61685.

[14] Thomhert S. Siadari, " Introduction to DL using Kaggle"; p.42.

[15] A.Radha Rani, Y.Anusha, S.K.Cherishama, S.Vijaya Laxmi. "Traffic sign detection and recognition using DL-based approach with haze removal for autonomous vehicle navigation"

[16] https://docs.wandb.ai/guides/hosting/self-managed/basic-setup

[17] https://developer.nvidia.com/embedded/jetson-nano