**Automatic Traffic Sign Recognition**

**17.11.2024**

**Applied Machine Learning**

**Aiymbay Sunggat**

**Adel Bulatova BDA-2206**

**Zhanna Saduakassova BDA-2203**

**Astana IT University**

**Overview**

This project involves the classification of traffic signs using deep learning models, leveraging transfer learning with pre-trained ResNet architectures. The process includes dataset preparation, training, evaluation, and iterative improvement of models. The final model is deployed using Streamlit for ease of testing through a user-friendly interface.

**Requirements Addressed**

**Dataset Details:**

* Used the [German Traffic Sign Recognition Benchmark (GTSRB)] dataset, containing images of 43 classes of traffic signs.
* Dataset is split into training (80%), validation (10%), and test (10%) sets.
* Augmented the dataset to include an "Unknown" class to handle out-of-scope inputs.
* Input images were resized to 64x64 during preprocessing.

**Model Training Process:**

* Employed ResNet34 architecture pre-trained on ImageNet.
* Fine-tuned the model to classify a subset of traffic signs ('главная дорога', 'уступи дорогу', 'стоп', and 'нет въезда').
* Incorporated transformations such as random cropping, rotation, and normalization for data augmentation.

**Metrics Documented:**

* Used accuracy, F1 score, precision, and recall for performance evaluation.
* Confusion matrices and classification reports were generated for each iteration of training and validation.

**Model Improvements:**

* Integrated a dropout layer to mitigate overfitting.
* Added an "Unknown" class for robustness.
* Trained a YOLOv5 model for object detection to demonstrate additional functionality.

**Deployment:**

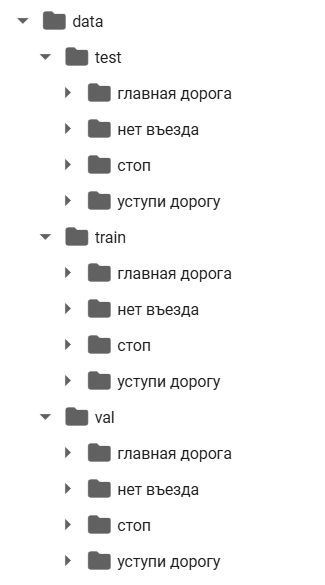
The final model was deployed using Streamlit, providing a web interface for users to upload and classify images of traffic signs.

**Code Submission:**

The project code, documentation, and deployment files are available on GitHub, with a detailed README file.

**Dataset Preparation**

* The GTSRB dataset was downloaded and unzipped.
* Traffic sign images were organized into folders corresponding to their labels.
* A subset of classes ('главная дорога', 'уступи дорогу', 'стоп', 'нет въезда') was initially selected for training.
* Images were resized to 64x64 pixels, normalized using the ImageNet mean and standard deviation values.
* An "Unknown" class was created by aggregating images from non-selected classes.



*Folder structure of* data/train*,* data/val*, and* data/test

**Model Training**

**Base Model**

**Architecture:** ResNet34 with the final fully connected layer replaced to classify 4 classes.

**Optimization:**

* Loss function: CrossEntropyLoss
* Optimizer: SGD with learning rate = 0.001 and momentum = 0.9
* Scheduler: StepLR with step size = 7, gamma = 0.1

**Performance Metrics:**

* Best Validation Accuracy: ~100%
* Confusion Matrix and Classification Report highlight some misclassification.

**Enhanced Model**

* Introduced an additional "Unknown" class.
* Added a dropout layer with a probability of 0.7 to improve generalization.
* Trained the model for three additional epochs.

**YOLOv5 Object Detection**

* YOLOv5s was used to perform object detection on test images.
* Demonstrated detection of specific traffic signs, providing bounding boxes and class predictions.

**Results**

**Enhanced Model Classification Metrics:**

* F1 Score: 0.99
* Precision: 0.99
* Recall: 0.99
* Improved performance, particularly for the "Unknown" class, reducing false positives.

**Confusion Matrix and Visualization:**

* Heatmaps of confusion matrices reveal accurate classification for most classes.
* Visualized predictions for randomly sampled test images.

**YOLOv5 Object Detection Results:**

* Successfully detected "Stop" and "No Truck Entry" signs in test images.
* Object detection adds functionality beyond classification.

**Deployment**

**Streamlit Application**

Features:

* Allows users to upload images for classification.
* Displays the predicted class along with confidence scores as a bar chart.
* Includes visualization of the uploaded image and its corresponding label.

**Instructions for Running the App:**

1. Clone the repository from GitHub.
2. Install the required dependencies using pip install -r requirements.txt.
3. Run the Streamlit app using streamlit run app.py.
4. Upload an image to view predictions and probabilities.

**GitHub Repository**

The project repository includes:

**Code:** Python scripts for training, evaluation, and deployment.

**Documentation:** Detailed README with setup instructions and explanation of the project.

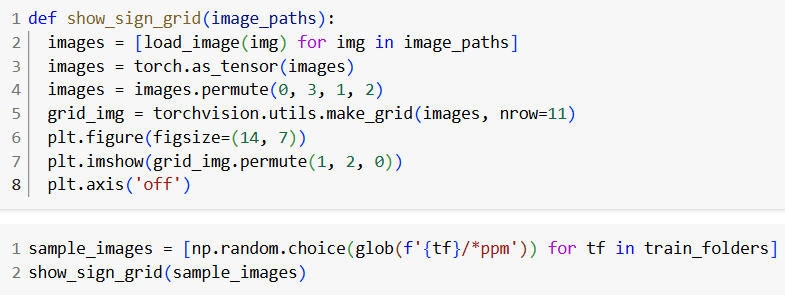
**Streamlit Deployment Files:** Code for the user interface and sample predictions.

**Conclusion**

This project demonstrates a robust pipeline for traffic sign classification and detection using deep learning. The enhanced model and deployment via Streamlit ensure practical applicability for real-world scenarios. Future improvements could include multi-language support, additional classes, and real-time detection capabilities.

**Screenshots**

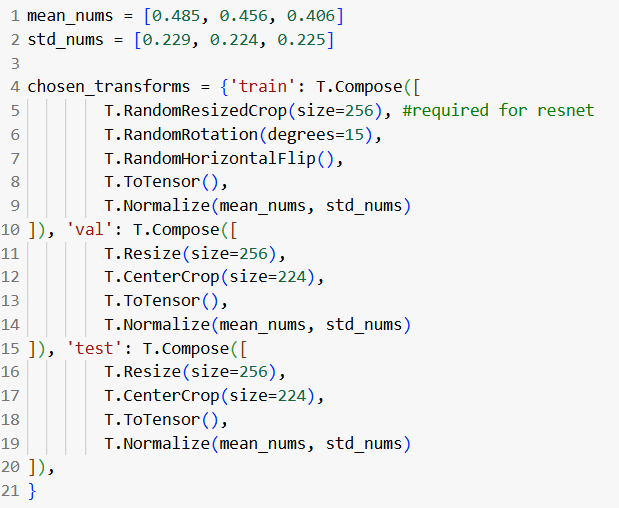
**Dataset Exploration**

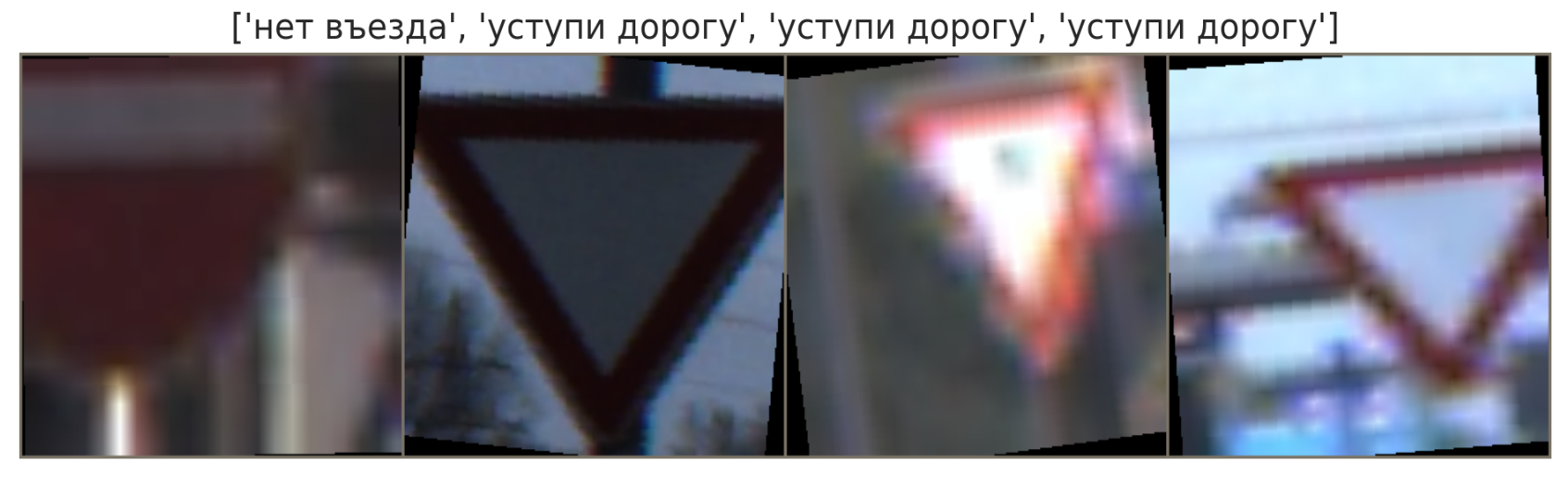
****

****

*Grid of traffic sign images displayed using* show\_sign\_grid*.*

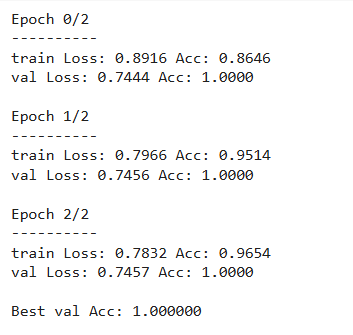
**Sample Visualizations**

****

****

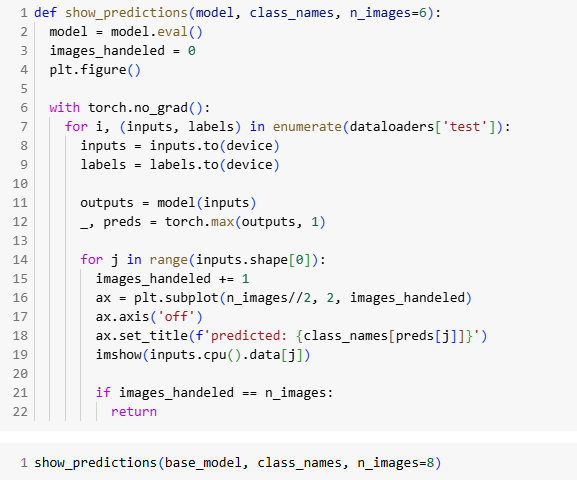
*Example traffic sign with augmented transformations (e.g., rotations, flips).*

**Training Process**



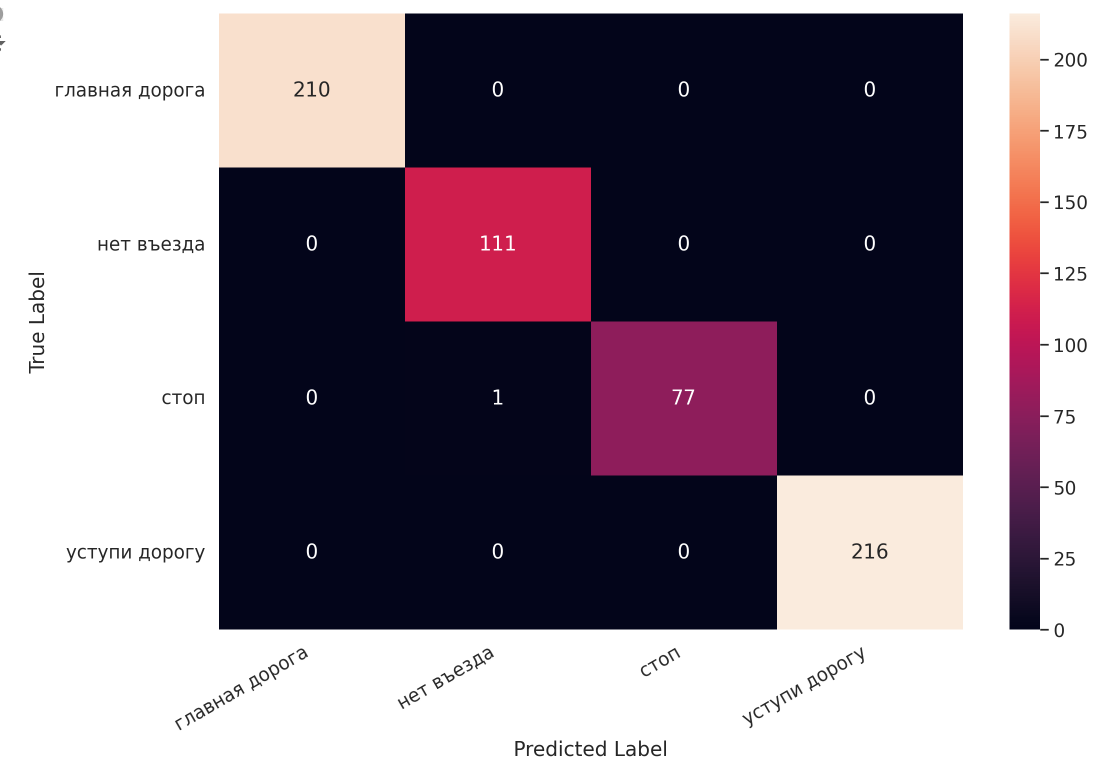
*Training and validation accuracy/loss logs for a few epochs.*

**Classification Results**

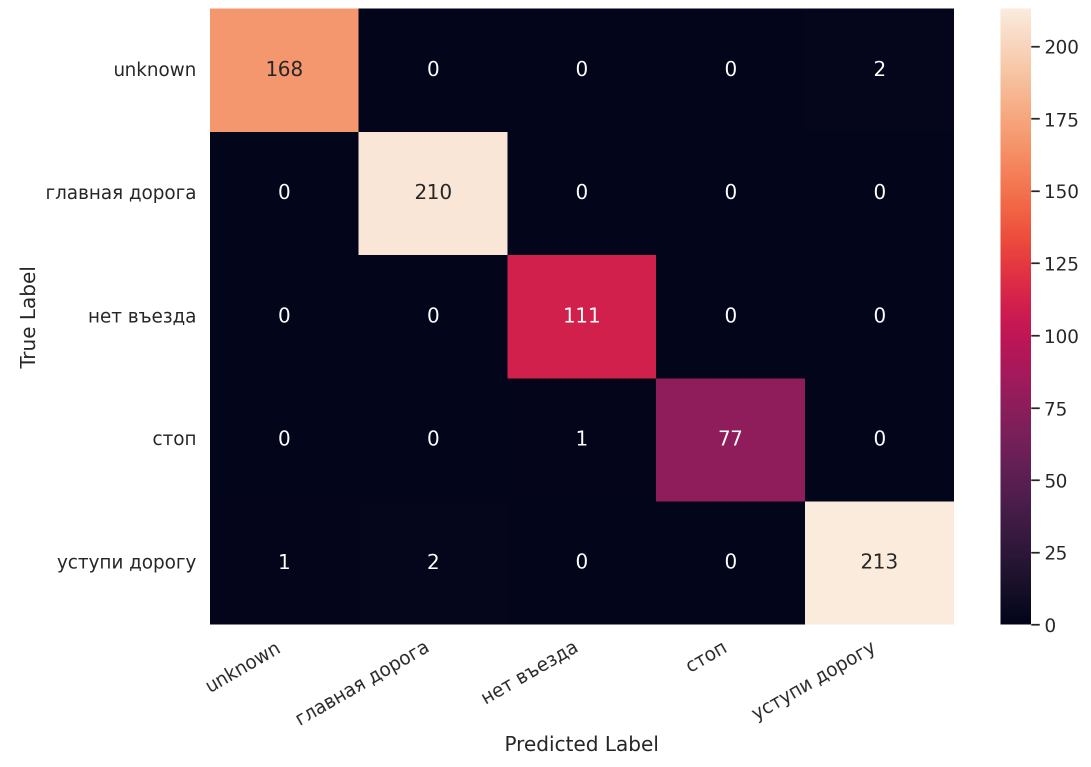
****

*Sample predictions from the show\_predictions function.*

**Confusion Matrix**

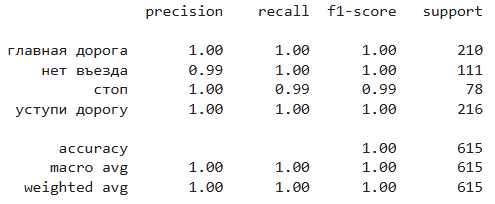
****

*Heatmap of the confusion matrix before adding unknown class.*

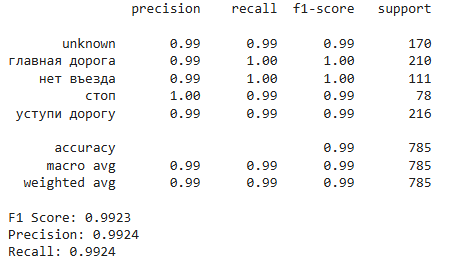
****

*Heatmap of the confusion matrix. after adding unknown class.*

**Evaluation Metrics**



*Classification report printed as a table before adding unknown class.*



*Classification report printed as a table after adding unknown class.*

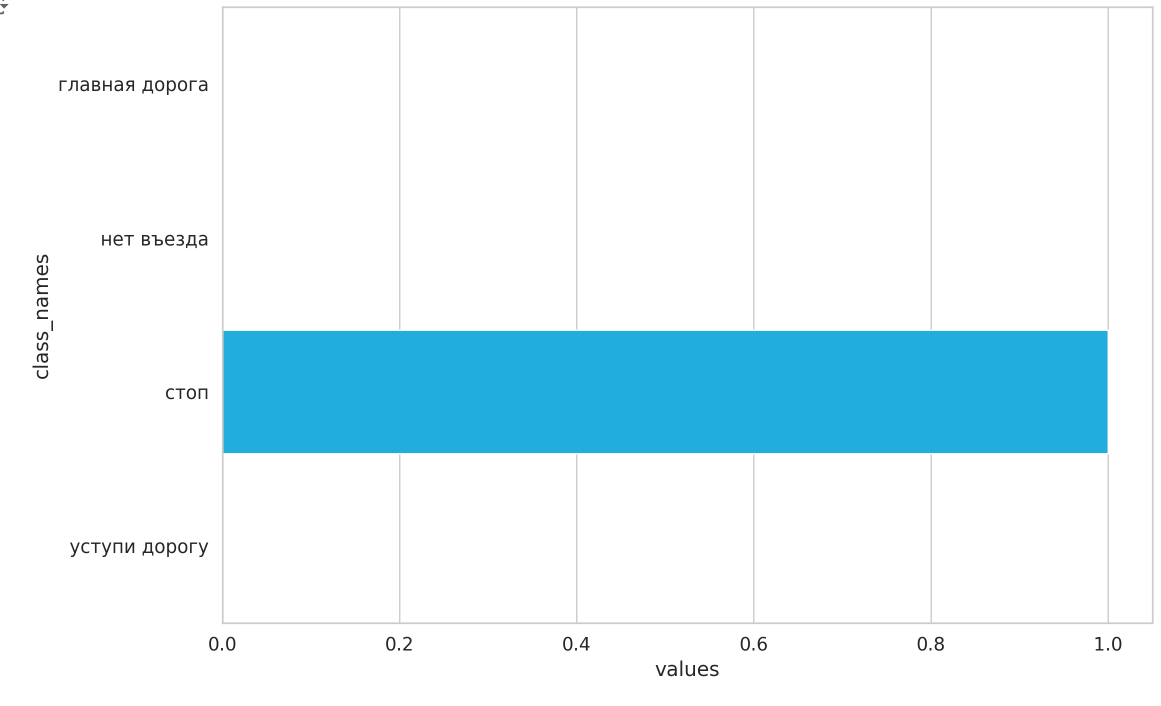
**YOLOv5 Detection Results**

****

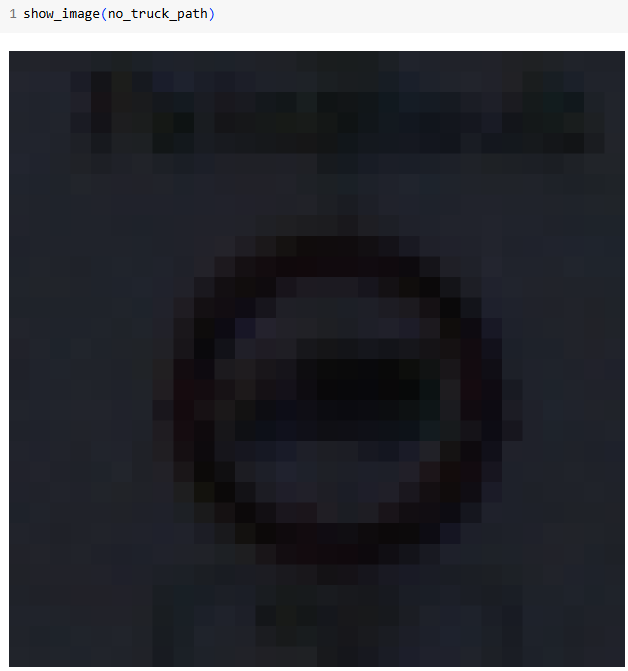
*Object detection output showing bounding boxes on a test image.*

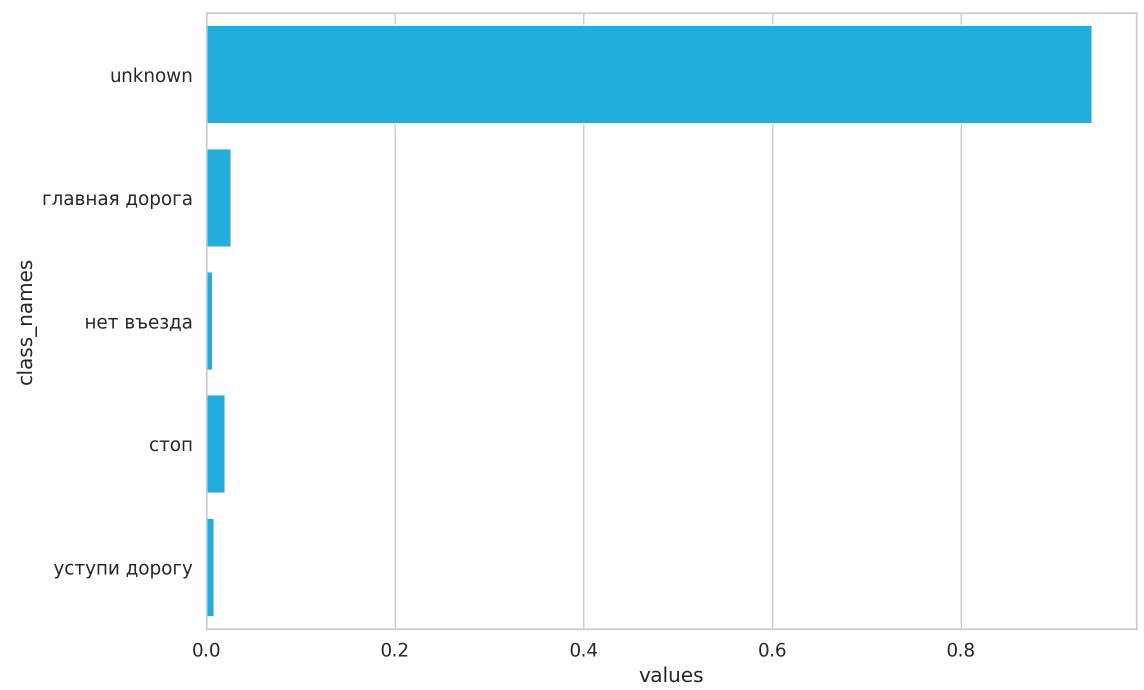
**Final Predictions**

****

****

*Bar chart visualizing class probabilities for a single image.*

****

****

*Bar chart visualizing class probabilities for a single image.*