# Commu-MNIST Revolution:

# Cyrillic Text Recognition With Different Machine Learning Models

Authored by Chris, Christian, and Owen

А Б В Г Д Е Ё Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ъ Ы Ь Э Ю Я

## Overview

We compared the efficacy of different machine learning methods in classifying Cyrillic characters. The models of interest were decision trees, all-vs-all aggregations of support vector machines, and convolutional neural networks.

Optimal models and hyperparameters for each respective model was reached through an iterative process of permuting hyperparameters and comparing results. Results were additionally compared with results from the ResNet API, which is a premade public model for use cases such as this.

Metrics of interest for all models included validation accuracy and training time of each model, with the aim being to identify the most effective method for hypothetical use cases given considerations of acceptable error threshold and finite compute resources.

## Results

Decision Trees were our worst-performing model in terms of both validation accuracy and validation accuracy per unit of training time. All permutations of criterions "entropy", "log_loss", and "gini" were tested, as well as splitters "random" and "best". These 6 permutations were tested 25 times each in order to avoid potential sampling biases in performance evaluations, but none produced worthwhile results.

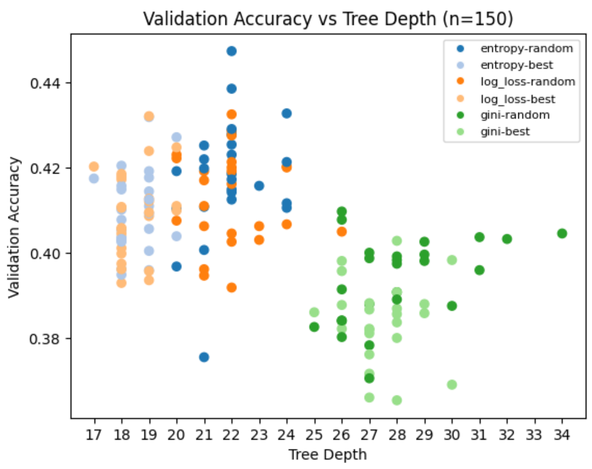
We had significantly better results with AVA SVM aggregations, which in our case, to generate binary classifiers for all pairs of each of the 34 cyrillic characters, meant generating 1122 binary classifiers for each AVA, and repeating this process for each permuted hyperparameter in the overall AVA. We tested 60 variations of the AVA hyperparameters, generating a total of 67320 binary classifiers. This rendered the SVM our by far most training-time intensive model, with the results typically taking approximately five hours to compute, but the resulting models were also highly accurate in validation.

Our third machine learning method was convolutional neural networks (CNNs). We approached this by varying the number of convolutional filters and dense outputs, and as CNNs are iteratively trained in epochs, we employed early stopping with a patience of five epochs to maximise validation accuracy.

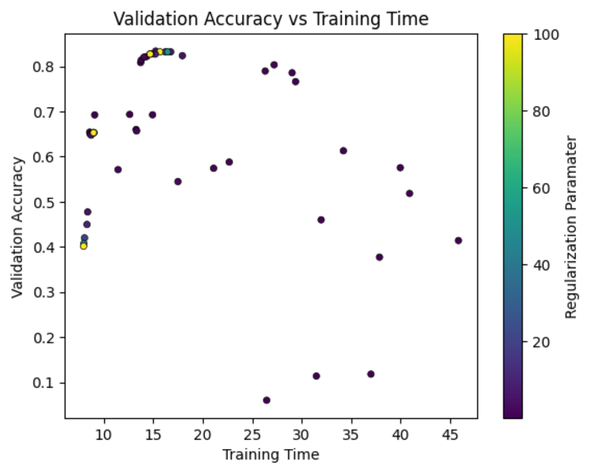
While we expected CNNs to outperform the other tested methods, we were nevertheless impressed with the results. CNNs drastically outperformed even our highly accurate AVA SVM classifiers across the board, for a fraction of the compute cost and training time.

## Data Visualisation

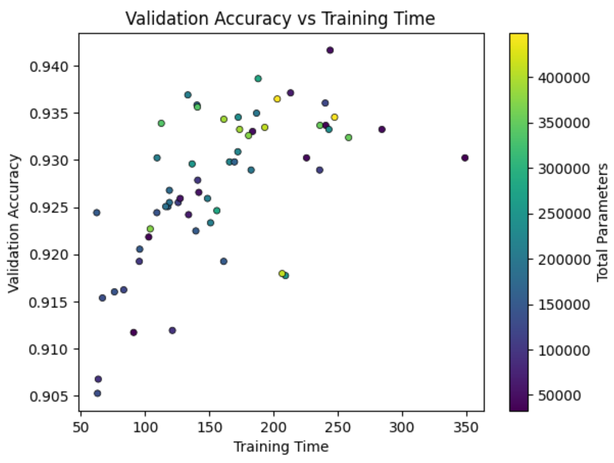
### *Decision Trees*




### *AVA SVMs*




### *CNNs*




### *Comparison*