# CS 145: Introduction to Data Mining
# Auto Plate Recognition

**Github:** https://github.com/dylanphe/auto-plate-detector.git

## I. Objective:

The goal of our project focuses on using machine learning algorithms that we have learned in class and through further research to classify the State's names that appear on all license plates we obtained from the 50 U.S. States including Washington D.C. We aim to construct and train models that can effectively perform this classification task. The project involves comparing performances between different machine learning models, as well as taking into account their complexity, training times, and memory efficiency. The metrics we intend to use for our models evaluation and comparison include F1 Scores and accuracy. In addition, we train and test the performance of these algorithms on a dataset of diverse license plate images with varying lighting conditions, quality and sizes.

## II. Obtaining and Preparing Dataset:

The dataset consists of 3,279 images of license plates from all 50 U.S. states plus Washington DC, which were obtained from sources 1 and 2 in our references.

We then organized this data into a table format of dimensions 3,279x2(file_list.csv). Each entry in the table contains two attributes, namely "Label" and "Image", which serve to identify each image file along with its corresponding label. In addition, we also split them into 3 different sets:

- The training dataset consists of 2951 images
- The validation dataset consists of 500 images
- The testing dataset consists of 328 images

In order to enhance the visibility of the State's name letters against the background in our image dataset, we employ several pre-processing steps, which include image resize, intensity enhancement, grayscale conversion, image segmentation, color inversion, and normalization. Specifics of these steps can be found in our code. With these preprocessing steps, we've improved our mode's best performance by around 40-50%.

## III. Models Implementation:

For our project, we have chosen to construct and compare three distinct machine learning models to address the task of license plate's States classification. The models we have selected offer different approaches to tackle the classification task, allowing us to gain insights into their strengths and weaknesses as shown below.

### 1. Model 1: Convolution Neural Networks (CNNs)

As we've learned in class, Convolutional Neural Networks or CNNs have proven to be highly effective in images classification tasks. These deep learning models are specifically designed to process visual data and capture complex patterns and features from the input images. With that said, we have decided to construct these deep neural networks as our first model for the task.

The architecture of the Convolutional Neural Network (CNN) was implemented using the Sequential class from the Keras library (model architecture in code). The CNN architecture consists of five convolutional blocks, each applying 2D convolution operations over the spatial dimensions of the input images. After the convolution operation, the ELU activation function is applied to introduce non-linearity into the model. Following each convolutional block, there are three additional operations to process the output before passing it to the next block:

- **Max-Pooling operation**: reduces the spatial dimensions of the feature maps by downsampling

- **Batch Normalization**: applied to normalize the activations of the previous layer
- **Dropout Operation**: A form of regularization that randomly selects a fraction of input

An additional operation is applied to the output of the fifth block to flatten the matrix before feeding it to the last fully connected dense layer with softmax as its activation function. The model is then trained on the training dataset to optimize its weight parameters. A learning rate of 0.001, a batch size of 3 were chosen to train our model for 10 epochs. It was able to achieve a training accuracy of over 98% and a validation accuracy of approximately 86%.

As a result, the trained CNNs model was able to predict the test dataset with a 80% accuracy. This indicates that the model performed well in classifying the test data, correctly predicting the states of license plate images with a high level of accuracy. Depending on the application and the desired level of performance, this testing accuracy may be satisfactory or require further improvements. Factors such as capturing plate images at random angles, in poor lighting conditions, or while in motion can significantly impact the image quality.

### 2. Model 2: Feed Forward Neural Networks (FFNNs)

As a contrast to the CNN model that was built, we now focused our attention on a slightly different but less complex algorithm for constructing our next model. The feedforward neural networks were also implemented using the sequential class in keras (model architecture in code).

The input of the basic two layer neural network is a flattened image, which is fed to the first Dense layer with 256 neurons that applies the ReLU activation function. To prevent overfitting, a Dropout process is also added to help prevent the model from relying too heavily on specific neurons, promoting better generalization to unseen data. Finally, the model ends with another Dense layer consisting of 51 neurons to produce the output for the 51 classes we have in our classification task using a softmax function.

The model is then trained in a similar manner to our first model, but achieves only around 3% of accuracy for both the training and the validation dataset. As expected, it also didn't perform well when predicting the testing data. One of the main reasons that lead to this poor prediction result of 4.26% is due to the fact that the model is too simple to effectively classify complex images. This FFNN model is a basic 2-layer fully connected neural network that lacks the ability to capture any meaningful spatial relationships and patterns within the images. These images as visualized earlier contained intricate details and local features that required more sophisticated models, such as convolutional neural networks (CNN) to extract and analyze.

### 3. Model 3: Pre-train CNNs - ResNet50

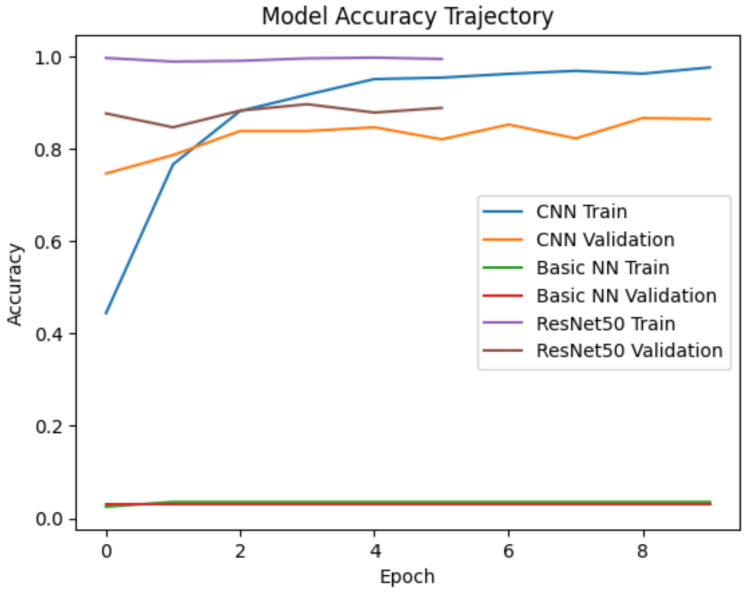
The third model that we have used is a "ResNet50 model", which is a deep convolutional neural network architecture best known for its excellent performance in image classification tasks. This model provided by Keras has been pre-trained on the large-scale ImageNet dataset with millions of labeled images across various categories. For this model, we define the shape of the input images to be (367, 1000, 3) since the ResNet50 model only works on images with more than 1 color channel. Similarly, we also applied an additional flattening layer to flatten the output from the base model followed by a dense layer that takes these flattened outputs and produces the final classification probabilities for the 51 classes.

Due to the size and the complexity of the model, we decided to train it for only 6 epochs and it is observed that this model still outdid the CNNs model at predicting the validation dataset with the highest accuracy achieved in the fourth epoch at 90%. One downside to this model is that the training process becomes such a daunting task to the CPU. A test accuracy of over 85% was achieved using this model for prediction.

## IV. Models Comparison and Evaluation:

We used train, validation, and test accuracy as well as F1 scores as our evaluation metrics. We will compare the performance of the models by comparing the mentioned evaluation metrics. The following grid displays the overall best accuracies, and do note that they are not from the same epochs as the accuracies are very similar to each other when they are at the maximum.

| Model | Train Accuracy | Validation Accuracy | Test Accuracy |
|---|---|---|---|
| CNNs | 98% | 86% | 83% |
| NNs | 3.5% | 3.2% | 4.2% |
| ResNet50 | 99% | 89% | 85% |




Although determining the best-performing model is challenging, we can conclude that Model 2 performed poorly due to its lack of layers. Networks with more layers can capture spatial and local features more effectively. Unfortunately, our basic NNs model had only two layers, leading to subpar performance in the task. We can also note that by a slight margin, ResNet outperformed CNN in case of both accuracy and F1 scores. CNN is known for its strong feature extraction capabilities in image processing tasks, while ResNet excels with large datasets and addresses the vanishing gradient problem. As our task involved image processing on a relatively large dataset, ResNet architecture contributed to its better performance compared to CNN but also at the cost of a longer training time and huge memory consumption.

## V. Contributions:


- Caleb Lee **UID:** 305 330 193 **Email:** bkcaleb45@g.ucla.edu
  - Models comparison, calculation of evaluation metric.
- Dylan Phe, **UID:** 505 834 475 **Email:** dylanphe@g.ucla.edu
  - Set-up project in the notebook, prepare the dataset, plotted out confusion matrices.
- Kevin Wang **UID**: 305-503-382 **Email**: kwang1083@g.ucla.edu
  - Construct and train all 3 models

## VI. References:

[1] Tolga. (2021). US License Plates Data - Quick Overview. Kaggle. https://www.kaggle.com/code/tolgadincer/us-license-plates-data-quick-overview/notebook.

[2] PlatesMania. (n.d.). License plates of the USA. Retrieved May 21, 2023, from https://platesmania.com/us/gallery-1.

[3] Angara, N. S. S. (2015). Automatic License Plate Recognition Using Deep Learning Techniques. Retrieved May 20, 2023, from https://scholarworks.uttyler.edu/ee_grad/30.