Deep Learning - CNN

Evgeny Manturov

March 27, 2024

1 Environment and Hardware

The environment used for this project is a **Python 3.11.8** installation with **IPython 8.20**. All the requirements of the virtual Anaconda environment are contained in **conda-requirements.txt** and can be installed along with the new venv using:

```
conda create -n env_name --file conda-requirements.txt
```

Training the model on CPU is lengthy and inefficient. In order to make the training orders of magnitude shorter, it is recommended to use an nVidia GPU with at least 12Gb of VRAM and CUDA capability. This requires CUDA and cuDNN installed.

2 Dataset

The dataset consists of 1260 images of different size and shape, each containing one (or several instances of one) of 27 different logos, which are presented as labels (classes) to be predicted. The validation set is a set of 135 real-world images of the logos, along with the respective validation labels.

3 Preprocessing

The tables containing image-related data and labels are loaded.

- "subset" column is dropped - it is of no use for the model.
- Images of classes not represented in both train and validation datasets are removed.
- Duplicate rows are dropped - 4536 rows are reduced to 1260 rows. This step may also include reducing image label representation to 1 per image.
- Image entries with negative logo dimensions are dropped - here it is unclear if such entries are wrong, or if [x1, y1], [x2, y2] should be swapped.
- Images without labels are removed from validation image list - they cannot be used for verification.

The images themselves are not too many and not too representative for the model to adequately predict the validation image set classes. Image augmentation can be used to generate new images from the existing ones. This includes:

- Resizing the image to the defined average size to feed the model - this step is required (the model inputs images with same shape only).
- Applying Color Jitter - randomly changing the brightness, contrast, saturation and hue of an image.

- Blurring the image to reduce its quality - will make the model more robust to low-quality images.
- Applying Random perspective - the image is 3D-rotated within its borders to emulate view angle.
- Posterizing the image by randomly reducing the number of bits in each channel.
- Normalizing the image tensor - is a conventional use as the last layer in the preprocessing stage.

Augmentation steps are not needed for the test batches - they are not used for backpropagation and therefore should be closer to the actual validation set.
Some augmentation layers are detrimental, e.g. Random flip (logos are almost always asymmetrical) and random rotation (some logos are too similar to others if rotated).

4 Model training procedure

Batch processing is used to train the CNN model to be used. The batch size selected is 128 (this can also be 64 or 32). The train image batch is loaded via augmentation-transform dataloader, and the model performance is tested on the transform dataloader test batch. Then, the model is tested on an entire validation dataset. Accuracy values for each epoch and each dataset are recorded. Training continues until the accuracy value running mean does not increase anymore.
Normally, the use of the same images to both train and test the model is detrimental and causes **overfitting**. To prevent overfitting, several techniques are used:

- 2 Dropout layers are added to the model (Section 5).
- Augment layers are applied in series and at random, which means that there are many different augmented images, which increases the set of unique images significantly, reducing the chance of the same image to train the model several times.
- Weight decay is added to the loss function.

5 The model

The model selected consists of:

1. 2D Convolution layer with activation function (gaussian error linear unit)
2. 2D pooling layer
   These are repeated twice
3. Flattening layer (converts a 2D tensor into a longer 1D tensor)
4. Random dropout layer
5. Hidden fully-connected layer with pooling tensor as input to 500 hidden perceptrons
6. Another Random dropout layer with activation function (rectified linear unit)
7. Output layer with Softmax activation function. This layer outputs the probability of the image being the image of a defined one-hot-encoded class.

The model is later compiled with Adam optimizer (/w weight decay and AMSGrad variant) and multiclass Cross entropy loss.

6 Experimentation and final results

The model has many different parameters to experiment with:

- Input image shape coefficient
- Size and stride of the convolution kernel
- The amount of perceptrons in 2D convolution hidden layers
- Types of activation functions
- The amount of perceptrons in the hidden layer
- The batch size

Accuracy is selected as the validation metric. The optimal model setup accuracy of train, test batches, and validation dataset, is displayed in Figure 1. By changing the parameters, the model accuracy may change significantly. For example, setting optimizer from Adam to SGD reduces the learning speed as shown in Figure 2.
In an attempt to increase the validation set accuracy, the neural network has been deepened:

- Each 2D Convolutional layer has been duplicated
- A third sequence of 2D conv - GELU - 2D conv - GELU - Pooling layers has been added

After some tweaks with image size, batch size and number of convolutional hidden layers, it has been observed that the model does not improve performance on the validation dataset when deepened, as displayed in Figure **??**.

7 Conclusions

The convolutional neural network is a sustainable way to create image identification algorithms. However, these networks require both a lot of diverse training data and data augmentation techniques to accurately predict images that are anyhow distorted or unclear.
In this case, it is safe to assume that the model lacks representative real-life logo examples and is only trained on augmented clear logo images. Since batch processing is used to train the model, it is not needed to ensure class balance in each batch individually, but it is highly recommended to ensure large enough batch size.
It is possible to use ensembles of several different CNN models, which may yield better results.
It may also be possible to improve the model significantly if larger image sizes and batch sizes are used, both of which require a lot of VRAM.

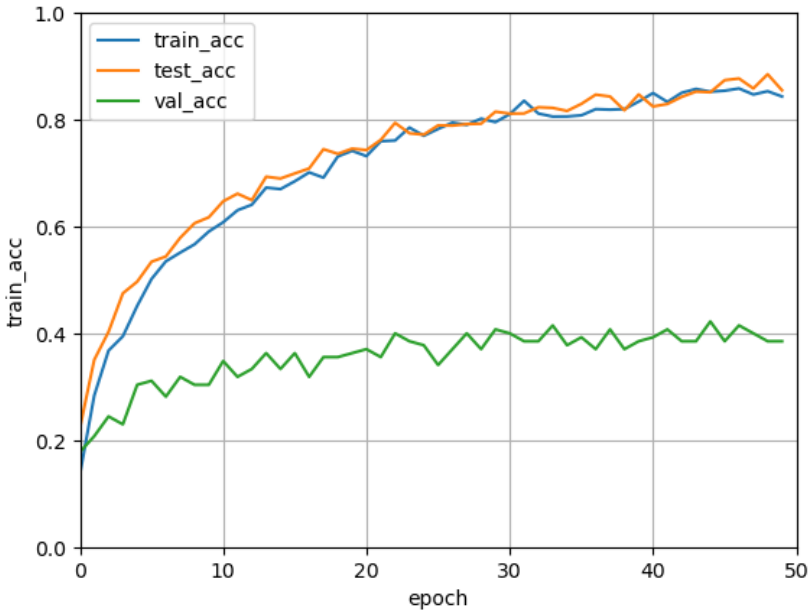

Figure 1: Model accuracy throughout epochs

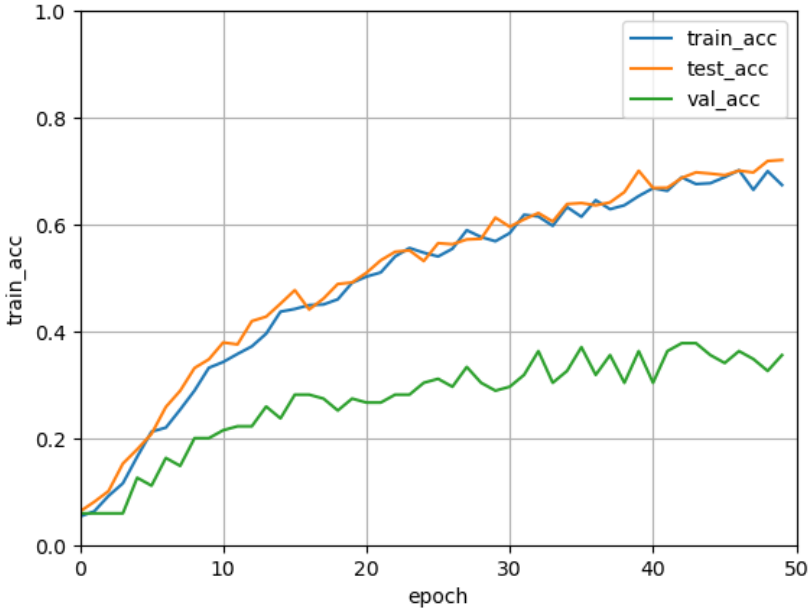

Figure 2: Model accuracy throughout epochs with SGD optimizer

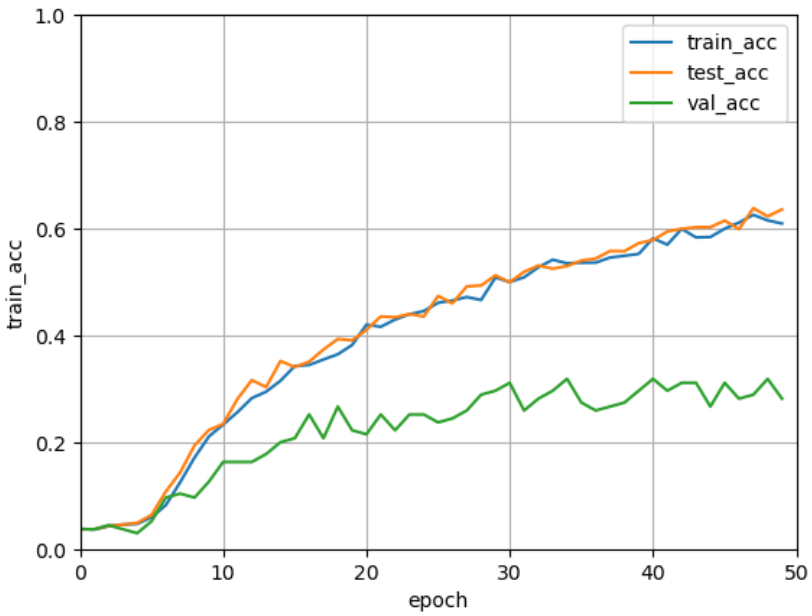

Figure 3: Accuracy of a deeper NN