CS337 Course Project
Face Recognition

Pranjal Kushwaha, Shashwat Garg, Vedang Asgaonkar, Virendra Kabra

Autumn 2022

Contents

1 Dataset

- We use the Labelled Faces in the Wild (LFW) dataset. It has been taken from Kaggle. There is an uneven distribution of images, with only 10 people having at least 53 images. Thus, we train the model to classify these 10 people, with our dataset containing 53 images for each.
- The train-test split is 80-20, and the train set is further split to create the validation set. Equal splits are created for each class.
- **Data Augmentation**: We add horizontally flipped images to the dataset. This acts as a regularizer, helping the model to generalize better.

2 Model Architecture

We use a Convolutional Neural Network (CNN) to create a multi-class image classifier. The network is inspired from FaceNet [3].

Layer	In	Out	Kernel	Params
conv1	$250 \times 250 \times 3$	$123 \times 123 \times 64$	$7 \times 7 \times 3, 2$	9K
batchnorm1	$123 \times 123 \times 64$	$123 \times 123 \times 64$		128
relu1	$123 \times 123 \times 64$	$123 \times 123 \times 64$		0
maxpool1	$123 \times 123 \times 64$	$61 \times 61 \times 64$	$2 \times 2 \times 64, 2$	0
dropout1	$61 \times 61 \times 64$	$61 \times 61 \times 64$		0
conv2	$61 \times 61 \times 64$	$61 \times 61 \times 128$	$3 \times 3 \times 64, 1$	74K
batchnorm2	$61 \times 61 \times 128$	$61 \times 61 \times 128$		256
relu2	$61 \times 61 \times 128$	$61 \times 61 \times 128$		0
maxpool2	$61 \times 61 \times 128$	$30 \times 30 \times 128$	$2 \times 2 \times 128, 2$	0
dropout2	$30 \times 30 \times 128$	$30 \times 30 \times 128$		0
conv3	$30 \times 30 \times 128$	$30 \times 30 \times 256$	$3 \times 3 \times 128, 1$	295K
batchnorm3	$30 \times 30 \times 256$	$30 \times 30 \times 256$		512
relu3	$30 \times 30 \times 256$	$30 \times 30 \times 256$		0
maxpool3	$30 \times 30 \times 256$	$15 \times 15 \times 256$	$2 \times 2 \times 128, 2$	0
dropout3	$15 \times 15 \times 256$	$15 \times 15 \times 256$		0
conv4	$15 \times 15 \times 256$	$15 \times 15 \times 64$	$3 \times 3 \times 256, 1$	148K
batchnorm4	$15 \times 15 \times 64$	$15 \times 15 \times 64$		128
relu4	$15 \times 15 \times 64$	$15 \times 15 \times 64$		0
dropout4	$15 \times 15 \times 64$	$15 \times 15 \times 64$		0
flatten	$15 \times 15 \times 64$	14400		0
fc1	14400	1024		14M
relu5	1024	1024		0
dropout5	1024	1024		0
fc2	1024	64		66K
relu5	64	64		0
dropout5	64	64		0
fc3	64	10		650
Total				15M

Table 1: Model Architecture

Notation for kernels: Size $\times$ Number of channels, Stride.

- **Convolutions**: To learn hierarchical representations of the input data, we use several convolutional layers. Given input of size $(N, C_{in}, H_{in}, W_{in})$ and output of size $(N, C_{out}, H_{out}, W_{out})$, the output value is

$$out(N_i, C_{out_j}) = \sum_{k=0}^{C_{in}-1} W(C_{out_j}, k) * input(N_i, k) + b(C_{out_j})$$

where $*$ denotes cross-correlation (flipped convolution), N is the batch size, C's are the number of channels, H and W are heights and widths. The parameters W and b are trainable. Cross-correlation is used to avoid flipping the filters. Since the weights are *learnt*, this is equivalent to using convolution.

- Batch Normalization: Adding these layers lead to faster convergence.
- ReLU: These are added for non-linearity, which is necessary for the universal approximation theorem to hold. The operation $ReLU(x) = \max(0, x)$ is applied element-wise.
- Max Pooling: Pooling involving down-sampling the feature maps by summarizing the features. In particular, max pooling outputs the maximum of input values over the specified region. This also helps make the representation become approximately invariant to small translations of the input.
- Dropout: It refers to dropping out random (non-output) neurons in the network, to prevent units from co-adapting too much. This prevents overfitting, thus acting as a regularizer. We use dropout with $p = 0.2$ after the maxpool layers [2], and with $p = 0.5$ after the fully-connected layers [1].
- Optimizer: Stochastic Gradient Descent (SGD) is used
  - Learning Rate: 10^{-3} is used. Convergence is achieved in less than 50 epochs. In comparison, smaller learning rates (order 10^{-4}) require many more epochs.
  - Weight Decay: 10^{-3} is used. For SGD, this is equivalent to using L_2 regularization.
- Loss: We use two losses in addition:
  - Cross Entropy Loss: This is the standard loss for multi-class classification.
  - Triplet Loss: This is introduced in [3]. We view the neural network as having an encoder and a decoder. The neural network until its penultimate layer is the encoder, which produces a latent representation for each image. The final layer acts as a decoder on this. This loss encourages clustering in the latent representation.

$$L(a, p, n) = \sum_{i}^{N} \left[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha \right]_+$$

where a is the anchor, p is a positive sample (having same class as a), n is a negative sample, and α is the margin (1 is used).

3 Analysis

- Batch Size

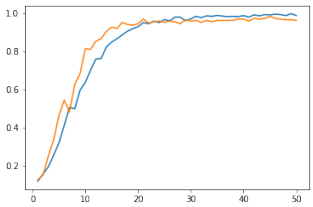

Figure 1: Batch Size 1

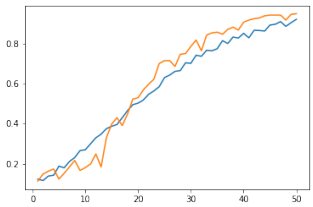

Figure 2: Batch Size 5

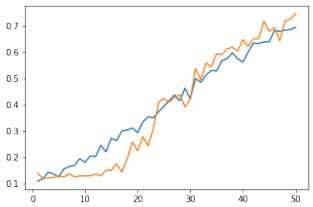

Figure 3: Batch Size 10

A batch size of 1 leads to better and faster convergence.

- Batch Normalization

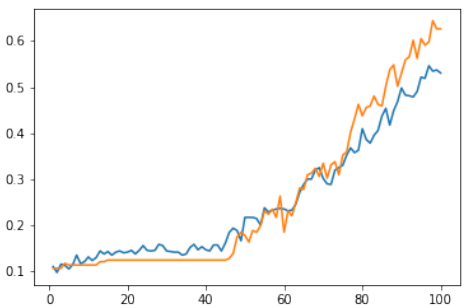

Figure 4: Without Batch Norm

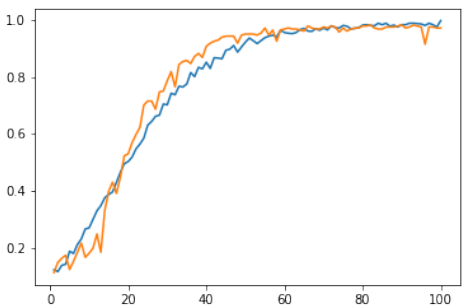

Figure 5: With Batch Norm

Faster convergence is observed with batch normalization.

- Optimizer

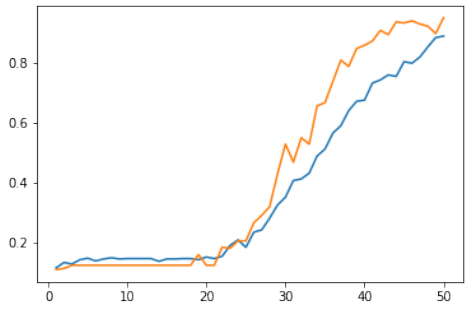

Figure 6: Adam

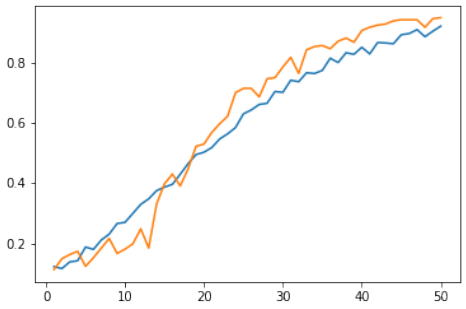

Figure 7: SGD

References

[1] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *CoRR*, abs/1207.0580, 2012.

[2] S. Park and N. Kwak. Analysis on the dropout effect in convolutional neural networks. In S.-H. Lai, V. Lepetit, K. Nishino, and Y. Sato, editors, *Computer Vision – ACCV 2016*, pages 189–204, Cham, 2017. Springer International Publishing.

[3] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 815–823, 2015.