Gradient-Based Learning Applied to Document Recognition

Team 20: Story of SMAI Life

Abhijeeth Reddy Singam (**2019101065**) Ainesh Sannidhi (**2019101067**)

Kunwar Maheep Singh (**2019101075**) Rishabh Khanna (**2019113025**)

Project for CS7.403 Statistical Methods in AI (Monsoon 2021)

Prof. Anoop M. Namboodiri

TA Mentor: Shanthika Naik

6th December 2021

Abstract

Lenet is a CNN-based character recognition system that takes a 32x32 resolution image and outputs the classification. It has three convolutional layers and two fully connected layers. The activation function used for all layers (excluding the last one) is tanh, and for the last layer, softmax was used. Average pooling is used between layers to decrease image resolution till we reach a 120-neuron fully connected layer.

Keywords: Neural Networks, OCR, Machine Learning, Gradient-Based Learning, Convolutional Neural Networks, Finite State Transducers

Github Link: https://github.com/inesane/Lenet

Gradient-Based Learning Applied to Document Recognition

Document recognition by virtue of character recognition is a problem that brings together the fields of computer vision, pattern recognition and machine learning. Character recognition is important in data extraction, especially from written text for the conversion of printed paper documents to machine-readable documents, such as bank cheques. Traditionally, machine learning techniques, especially neural networks, have been widely used when creating pattern recognition systems such as in optical character recognition and this paper aims to show how better pattern recognition systems can be made by focusing more on machine learning techniques as opposed to traditional hand-designed heuristics.

These techniques include using various Classifiers, Multilayer Neural Networks and Graph Transformer Networks which use Convolutional Neural Network character recognizers and global training techniques which give us a much better character recognition system than if we were to manually carry out the process of feature extraction.

Data Preprocessing

Dataset

For this project, we have chosen the MNIST dataset. It consists of 60,000 training set images and 10,000 test set images of handwritten digits from 0 to 9.

MNIST Dataset: http://yann.lecun.com/exdb/mnist/

Padding

A Padding of 2 has been added around the individual 28*28 images to get 32*32 images, which is then given to the Convolution Model.

Normalizing

As described in the paper, the input image values are normalized such that the background color is represented by -0.1 and the foreground is represented by 1.175. This ensures that the mean input is roughly 0 and the variance is 1, which accelerates learning.

Architecture

Layer	# Filters	Filter Size	Stride	Size of Feature Map	# Params	Activation Function	Connections
Input	-	-	-	32 * 32	-	-	-
Conv 1	6	5 * 5	1	6 * 28 * 28	156	tanh	122,304
Avg.Pooling 2	-	2 * 2	2	6 * 14 * 14	12	-	5,880
Conv 3	16	5 * 5	1	16 * 10 * 10	1,516	tanh	151,600
Avg.Pooling 4	-	2 * 2	2	16 * 5 * 5	32	-	151,600
Conv 5	120	5 * 5	1	120	1,516	tanh	48,120
Fully Connected 6	-	-	-	84	10,164	tanh	-
OUTPUT (Fully Connected)	-	-	-	10	1,850	sigmoid	-

Activation Function

In the paper, two activation functions are used. Tanh (specifically A*tanh(S*x) where A and S are constants) for all layers other than the output and Softmax for the final output layer. In modern times, newer activation functions like ReLU have proved to be more effective (as used in AlexNet) and could provide an improvement to LeNet-5's architecture.

Pooling Layers

In our implementation, a total of two average pooling layers are used.

Sparse Connectivity between Layer 2 and Layer 3

Sparse connectivity is implemented between layer 2 (pooling) and layer 3 (convolution). The image below describes the sparse connectivity scheme used in the original paper.

The sparse connectivity here forces a break in the symmetry of the network. It forces different feature maps to extract different features from the original images. Ideally, these different features would be complementary and would capture most of the useful information from the image.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	X				X	X	X			X	X	X	X		X	X
1	X	X				X	X	X			X	X	X	X		X
2	X	X	X				X	X	X			X		X	X	X
3		X	X	X			X	X	X	X			X		X	X
4			X	X	X			X	X	X	X		X	X		X
5				X	X	X			X	X	X	X		X	X	X

TABLE I

EACH COLUMN INDICATES WHICH FEATURE MAP IN S2 ARE COMBINED BY THE UNITS IN A PARTICULAR FEATURE MAP OF C3.

Weight Initialization

In our implementation, weights are initialized randomly between -2.4/fi to 2.4/fi where fi is the number of input parameters for the layer as mentioned in the paper.

Results

Outcome 1 (from using inbuilt functions)

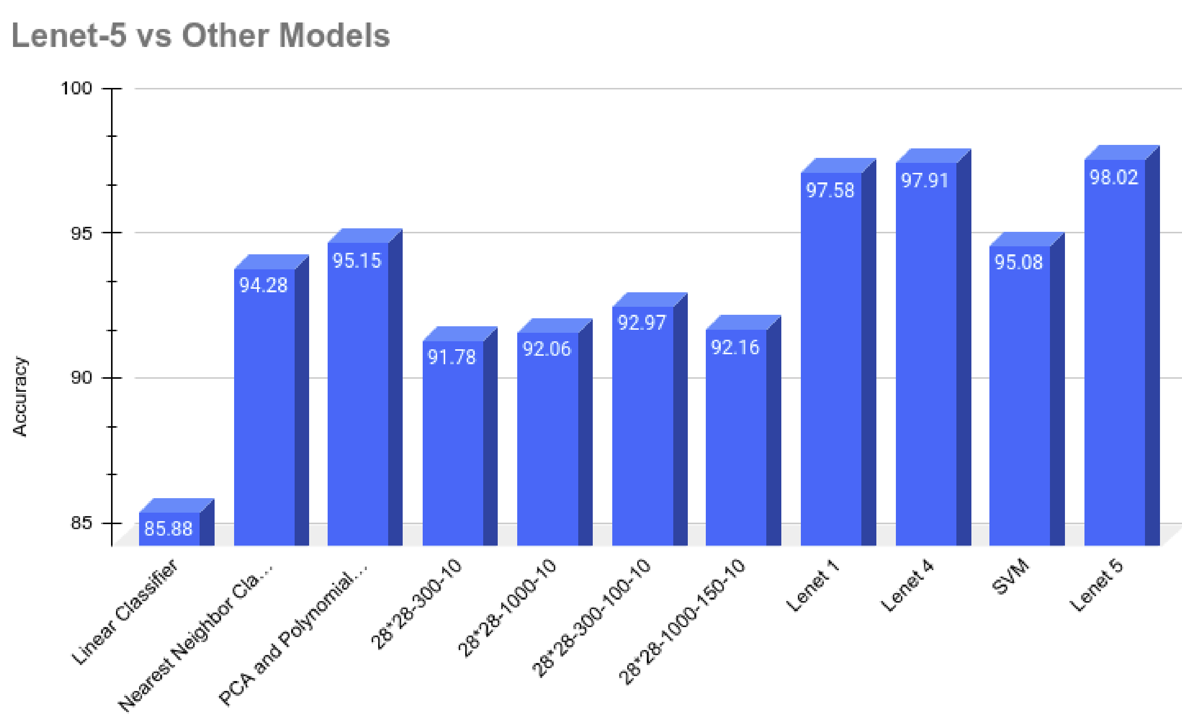
- **LeNet-5:** Out of the model implemented with inbuilt functions, this model performed the best by a small margin. This result goes to show LeNet-5's effectiveness.
- **LeNet-4:** The major difference between LeNet-5 and LeNet-4 is the presence of an additional fully connected layer and also a minor difference in the first fully connected layer. These differences allow LeNet-5 to outperform LeNet-4.
- **LeNet-1:** This model was originally implemented as a proof-of-concept that the CNN architecture was suitable for the problem of OCR. Even with a very small number of trainable parameters (~2600) it was able to achieve a very high accuracy.
- **SVC:** The Support Vector Classifier model performed comparably to other, slightly simpler methods like KNN and PCA+Polynomial (described later).
- **K-Nearest Neighbor:** The KNN classifier model's performance exceeded our expectations. Given that it is a very simple model and doesn't use complicated architecture like in LeNet-5, it was still able to put forth a competitive accuracy.
- **PCA and Polynomial:** In this model, PCA was used to bring the number of features down to 40, following which Singular Value Decomposition was used on the covariance matrices in order to diagonalize them. The 40-dimensional feature vector was then used as the input of a second degree polynomial classifier.
- **Fully Connected Multi-Layer Neural Nets:** Four models were implemented as a part of this architecture. Two of the models had one hidden layer and the other two had two hidden layers. These models generally performed worse than simpler methods like KNN.

- **Linear Classifier:** This model was both the simplest and worst performing models of the tested models. It's lack of accuracy can be attributed to its lack of complexity.

Outcome 2

Our implemented model without the use of any inbuilt functions apart from numpy reported an accuracy of 57%. This was after training the model on 10,000 data points for a single epoch.

Unfortunately, we were unable to run our custom implemented model fully on the dataset. The dataset consisted of 60,000 training images which would ideally be trained upon over multiple epochs. In our case, training took a very large amount of time and thus we were



unable to train the model for even a single epoch. The most we were able to train the model for was a single epoch over 10,000 samples. Any set of more elements than this was infeasible due our need for our systems throughout the duration of the project.

Conclusion

From the results obtained from both the custom-implemented model and the models implemented using inbuilt functions, we can see that the LeNet-5 architecture performs exceedingly well for the task of OCR. Especially from the comparison between models, we can clearly see how CNN models are better suited for our task when compared to other popular methods.

References

- Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," in Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, Nov. 1998, doi: 10.1109/5.726791.
- https://sh-tsang.medium.com/paper-brief-review-of-lenet-1-lenet-4-lenet-5-boosted-lenet-4-image-classification-1f5f809dbf17
- https://medium.com/@ngocson2vn/a-gentle-explanation-of-backpropagation-in-convolutional-neural-network-cnn-1a70abff508b
- https://www.jefkine.com/general/2016/09/05/backpropagation-in-convolutional-neural-networks/
- Handwritten Digit Recognition with a Back-Propagation Network
- https://www.kaggle.com/nishan192/mnist-digit-recognition-using-svm/notebook
-

Work Distribution

- Abhijeeth Reddy Singam - Implemented parts of forward and back propagation for all layers. Implemented convolution, sparse connectivity, activation function, forward passes. Conducted literature review.

- Ainesh Sannidhi - Implemented GUI for model testing. Implemented other models (with inbuilt functions) specified in the paper for comparison and outlined architecture of LeNet-5. Conducted literature review.
- Kunwar Maheep Singh - Implemented major parts of forward and back propagation for all layers. Implemented main code structure, pooling, forward passes and backward passes. Conducted literature review.
- Rishabh Khanna - Implemented outline of LeNet-5 along with other models (with inbuilt functions) specified in the paper for comparison. Implemented parts of forward propagation. Conducted literature review.