THE GEORGE WASHINGTON UNIVERSITY

WASHINGTON, DC

American Sign Language (ASL): Pattern Recognition Models using Neural Network

Machine Learning 2 – Final Project

December 2018

Presentation outline

1. Objective, Motivation and Background
2. Description of the data set
3. Methods
4. Results
5. Conclusion

Objective

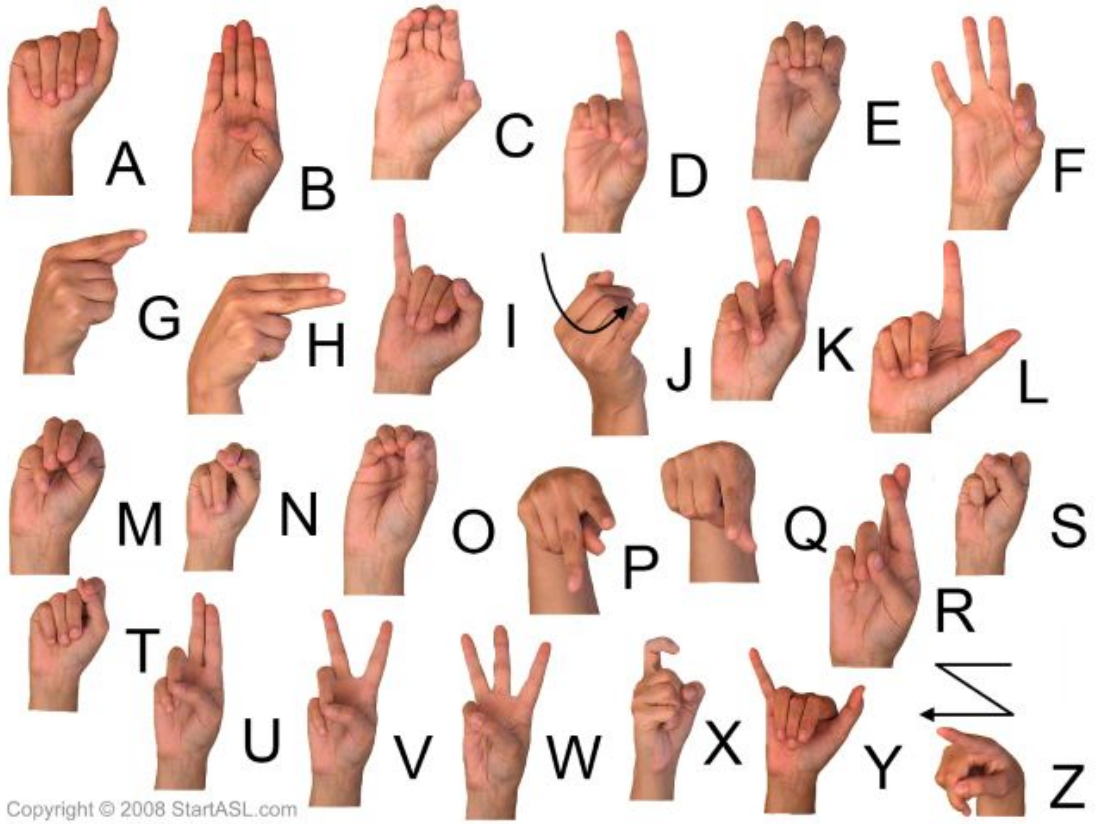
The objective of this work is to evaluate the performance of different Convolutional Neural Network architectures in the task of classifying twenty six hand signs of the American Sign Language representing the letters of the alphabet.

Hand gesture recognition

- Hand gesture recognition is the process of recognizing meaningful expressions of form and motion by a human involving only the hands.
- Many applications:
  - Robotic-assisted surgeries
  - Help deaf people to communicate
  - Any problem that can be benefited by augmenting the human-machine interaction …
- Mainly two ways: data-glove and vision.

American Sign Language (ASL)

- The American Sign Language is a visual language that incorporates hand signs as its foundation.
- When you use the hand signs for letters to spell out a word, you are “fingerspelling”.
- 26 hand positions for the letters.
- 10 positions for the digits.



ASL Vision Recognition

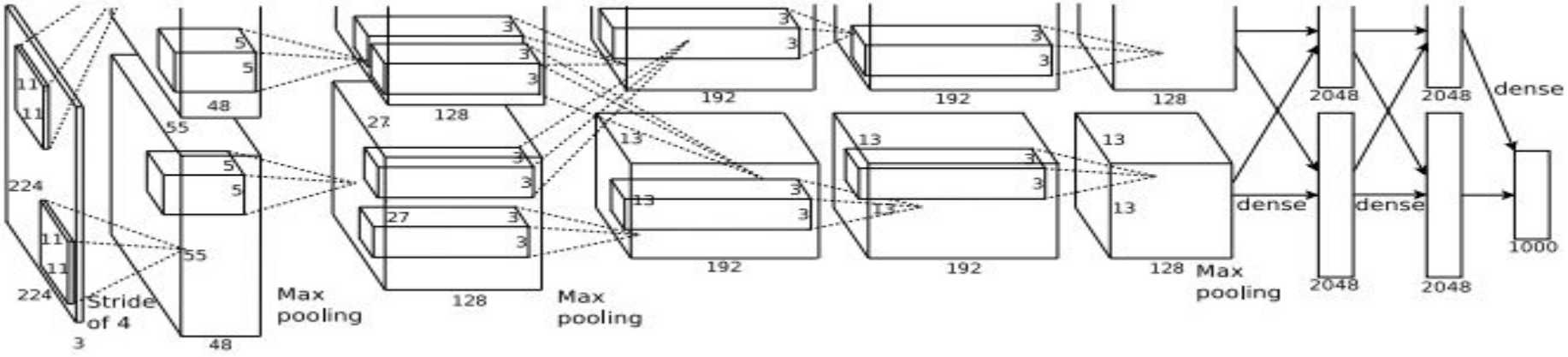
- Recent research in vision ASL recognition has employed deep CNN networks:
  - Garcia and Viesca (2016): real time ASL translator built upon pre-trained GoogleLeNet using transfer learning.
  - Bheda and Radpour (2017): CNN for images with depth-sensing technology.
  - Strezoski et al. (2018): test leader CNN architectures - LeNet, AlexNet, VGG net, GoogleLeNet - in the problem of identifying hand signs from the Marcel dataset.

CNN LeNet5 (1994)

- Seminal CNN model by Yann LeCun.
- Insight that image features are distributed across the entire image and that convolutions are an efficient way to extract similar features from multiple locations (parameters sharing).
- Much less parameters than conventional fully connected NN.
- Images are highly spatially correlated and convolutions take advantage of it.
- Specially designed to recognize handwritten digit

CNN AlexNet (2010)

- Winner of the ImageNet Large Scale Visual Recognition Challenge of 2010.
- Much larger network consisting of 11x11, 5x5, 3x3, convolutions, max pooling, dropout, and ReLU activations after every convolutional and fully-connected layer.
- Possible because the advent of GPU's

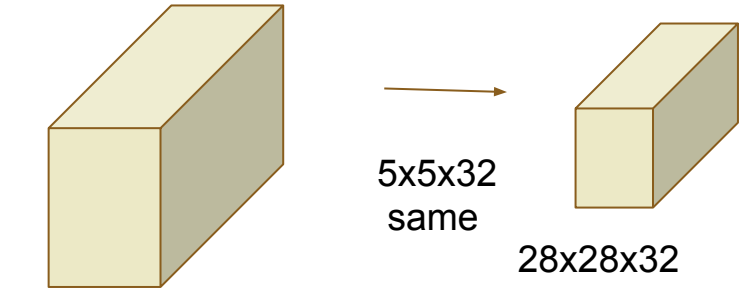


CNN VGG Net (2014)

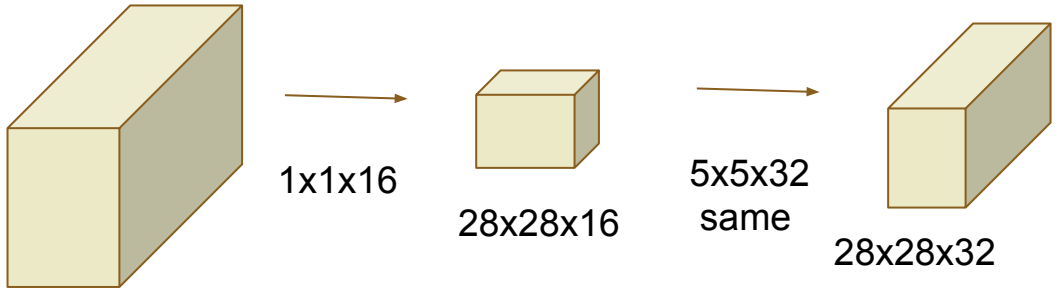
- Stacking **three** 3x3 convolutional layers without pooling between them, has an effective receptive field of using just a single 7x7 one.
- So, let's employ much more layers with smaller filter sizes.
- The advantage of incorporating three non-linear rectification layers instead of a single one is that makes the decision function more discriminative.
- This architecture is very appealing due its uniform structure and is the prefered solution for feature extraction.

CNN GoogleLeNet (2014)

- Creating even deeper architectures while keeping the computational cost constant.
- 22-layer deep architecture inspired by the Network In Network idea.
- The NiN method can be viewed as 1×1 convolutional layers.



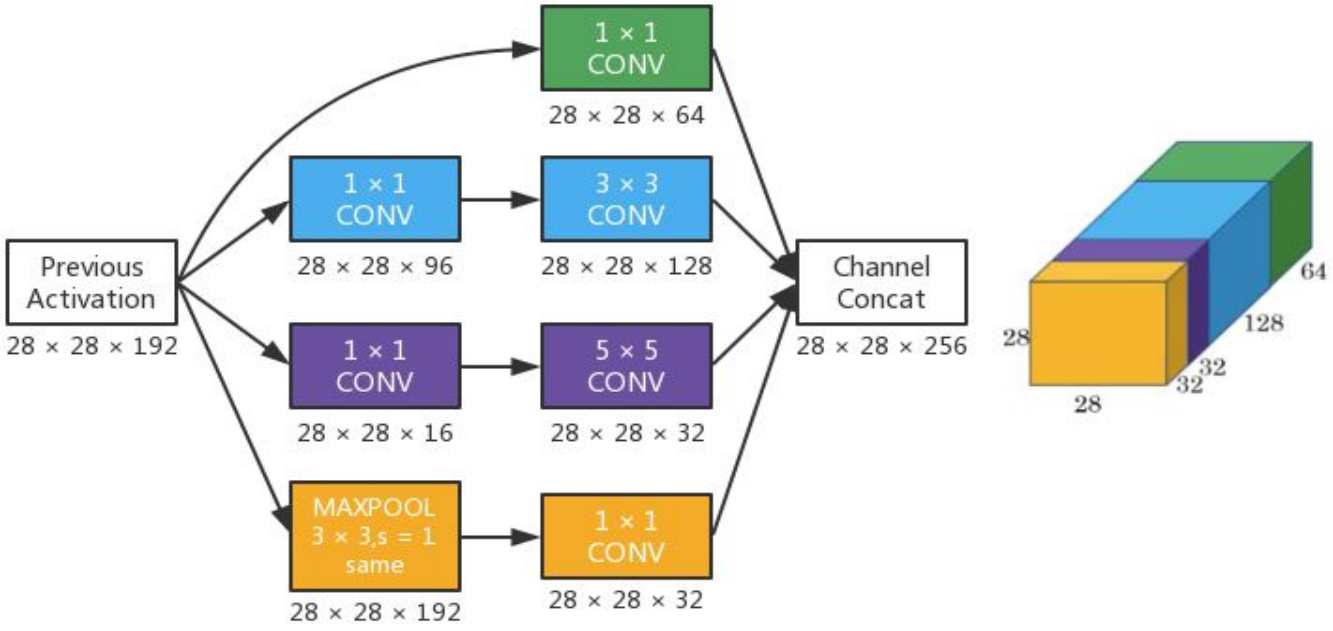
120 Million



12.4 Million

CNN GoogleLeNet (2014)

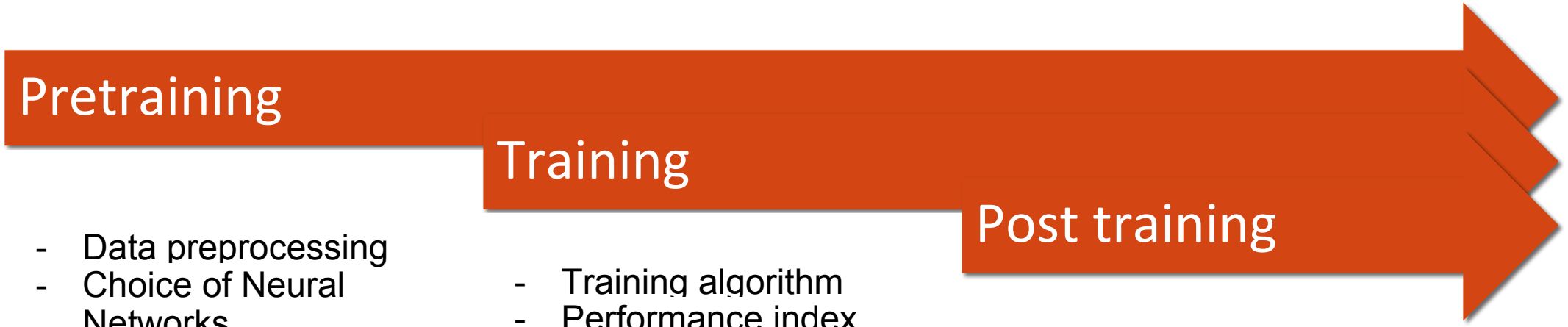
- Inception Modules.
- Using 1×1 convolutions to compute reductions before the expensive 3×3 and 5×5 convolutions.
- Error rate of 6.67% at ILSVRC.



Dataset Description

- The dataset ASL Alphabet is downloaded from Kaggle, which is a collection of images of alphabets from the American Sign Language.
- The dataset consists of 29 classes, of which 26 are for the letters A-Z and 3 classes for SPACE, DELETE and NOTHING, seperated in 29 different folders.
- The training data set contains 87,000 images which are 200x200 pixels. Each folder/letter has 3000 images.

Method



- Data preprocessing
- Choice of Neural Networks

- Training algorithm
- Performance index
- Weight initialization
- Stopping criteria
- Learning rate update
- Dropout nodes

- Confusion Matrix
- Precision/Recall/F-score
- AUC/ROC
- Error analysis
- Processing time

Method – Pretraining

- Data Preprocessing
  - Normalization (-1,1)
  - Train/Validation/Test split (70/15/15) --- Stratified by image labels
  - No missing values
- Choice of network for pattern recognition of images (CNN)
  - Baseline model
  - LeNet
  - AlexNet

Method – Pretraining (2)

Network Design

	Baseline	LeNet	AlexNet
Number of convolutional layers	2 conv (5, 5)	2 conv (5, 5)	5 conv (11, 5, 3, 3,3)
Number of fully connected layers	1 (250 neurons)	2 (120, 84 neurons)	2 (4096, 4096)
Layer output transfer function	LogSoftmax	None (Linear)	None (Linear)
Estimated number of parameters	11 M	4.2 M	103 M
Estimated number of feature maps	230	114	250K

Method – Training

Performance Index

Cross Entropy

Cross-entropy measures the performance of a classification model whose output is a probability value between 0 and 1.It is used to define a loss function, which is to be minimised by updating weights based on the loss function.

Optimizer

Adam Optimizer

Adam optimizer is a replacement optimization algorithm for stochastic gradient descent for training deep learning models. It provides an optimization algorithm that can handle sparse gradients on noisy problems and is relatively easy to configure where the default configuration parameters do well on most problems.

Method – Training (2)

Weight Initialization

Xavier Normal initialization

The xavier algorithm automatically determines the scale of initialization based on the number of input and output neurons and makes sure that the weights are not too small or too big.

Learning Rate Update

Reduce Learning Rate on Plateau

It is a scheduler in pytorch that updates the learning rate based on the value of a specific output (losses or accuracies) when that value is constant after a certain number of epochs.

Method – Training (3)

Stopping Criteria

Loss trends.- The number of epochs selection is based on the trends between the training and the validation losses.

Accuracy trends.- This criteria is used in addition to the loss trends.

Dropout Nodes

We use the default value (p=0.5) of dropout nodes during the training.

Method – Post-training

Classic Metric

- Confusion Matrix

- Precision/Recall/F-score

- AUC/ROC

Other metric

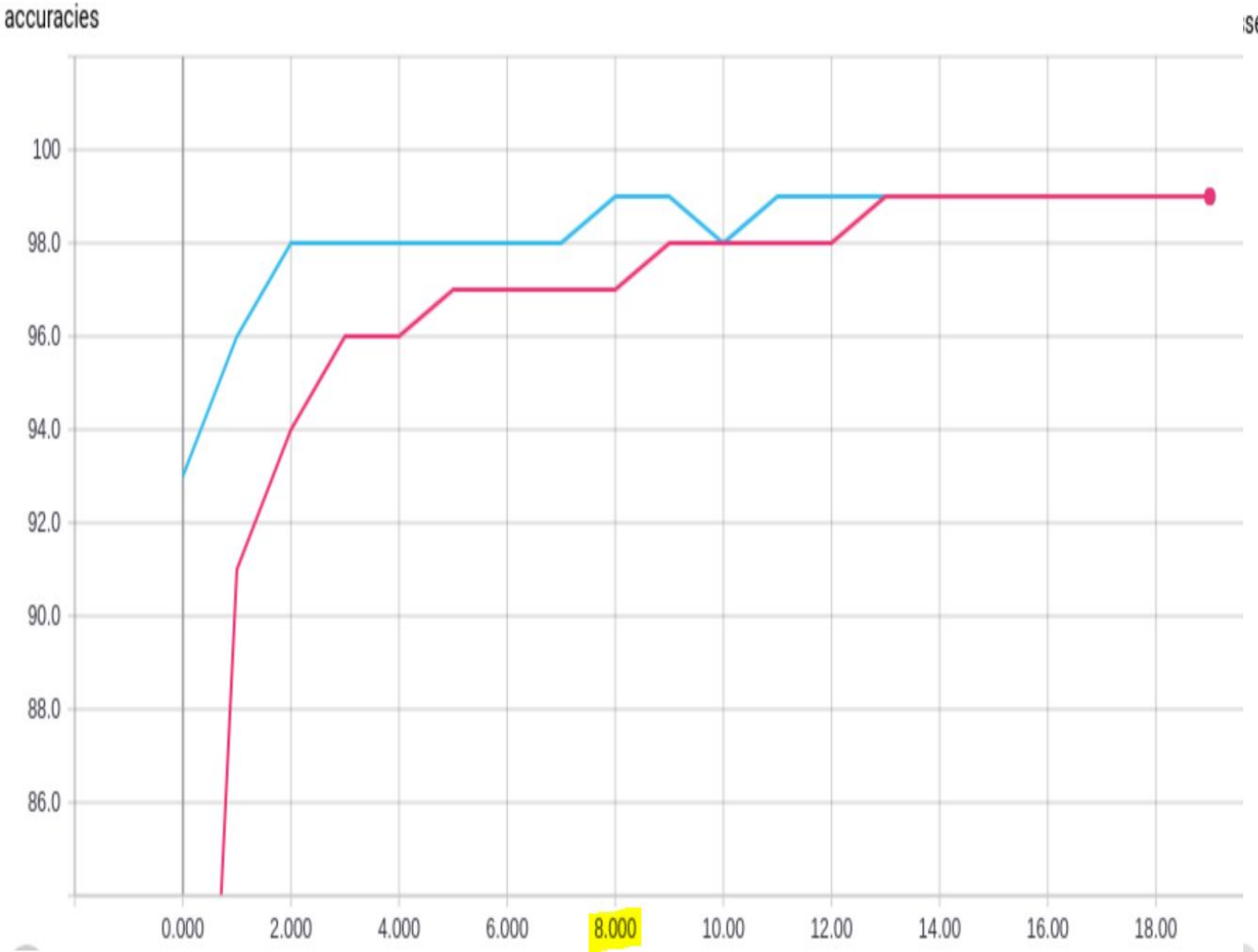
- Error analysis

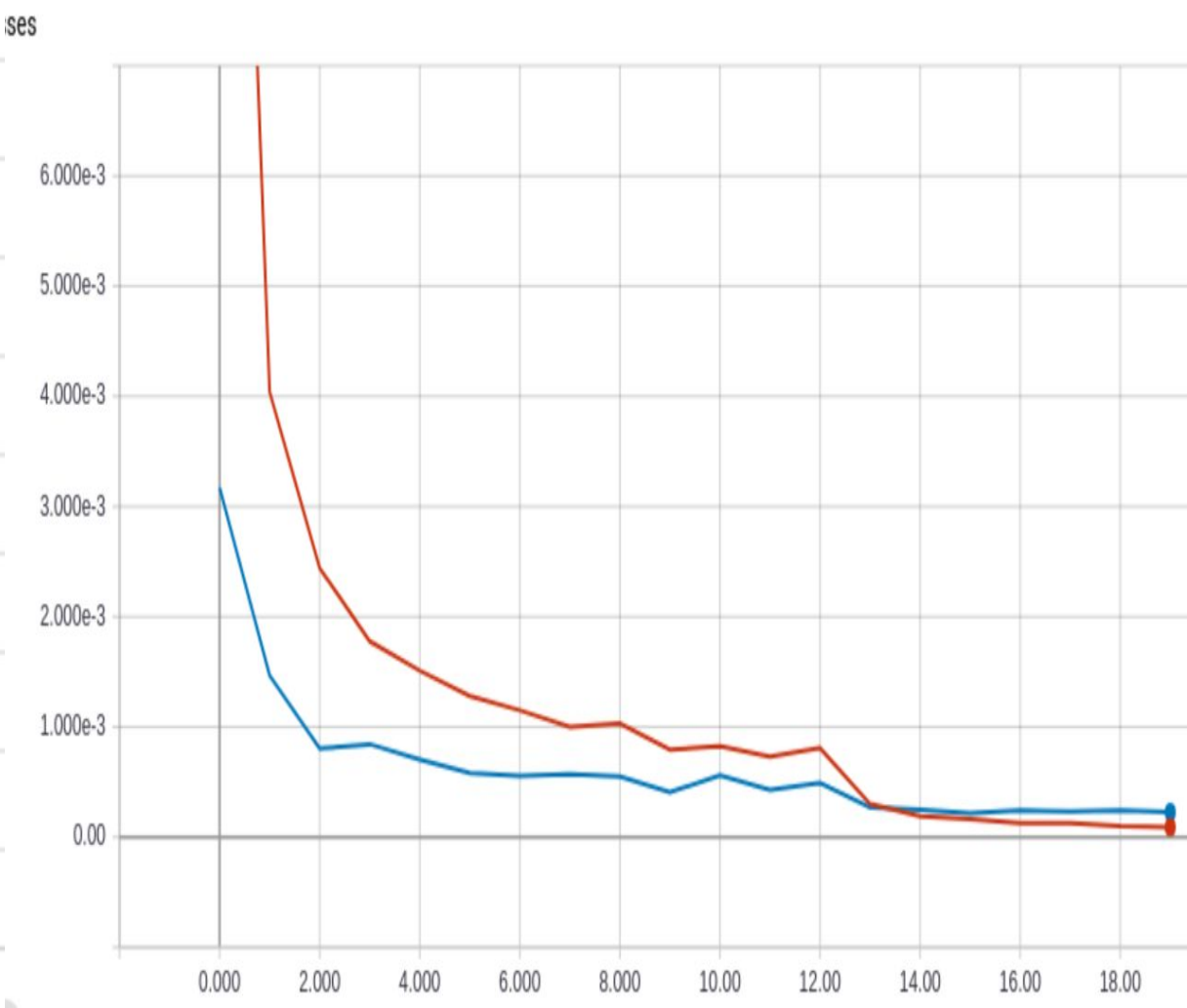
- Processing time

Results

1. Baseline model performance
2. LeNet model performance
3. AlexNet model performance
4. Models comparison and choice of the best model

Baseline Model





Baseline Model

Precision	0.9916846
Recall / Accuracy	0.9916475
F-score	0.9916470
AUC	All greater than 0.9998
Processing time	19 minutes
Error	std= 0.0007859 mean= 0.00096642
Update at epoch	None
Stopped at epoch	Epoch 10

Initial Accuracy: 0.99593

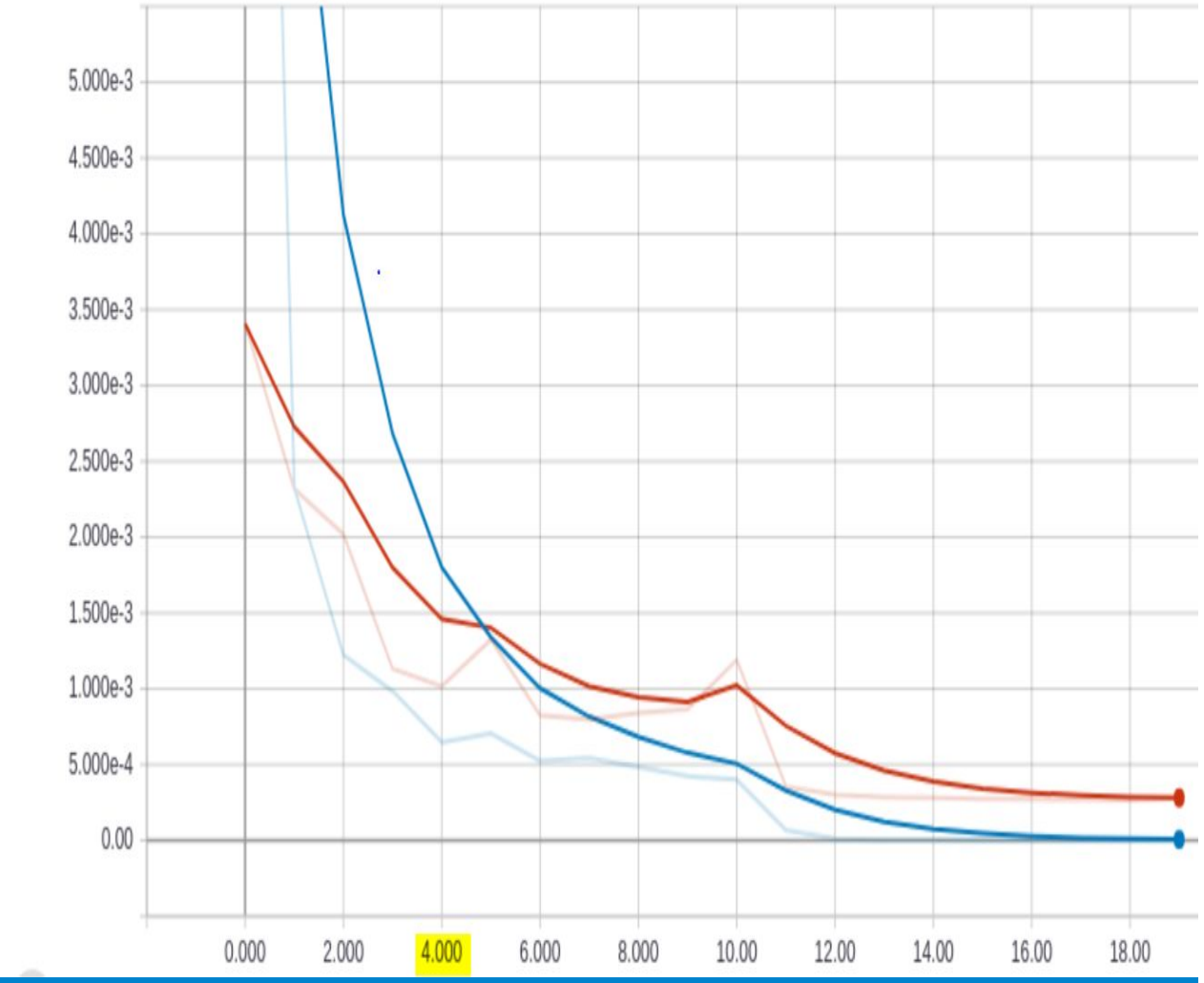
Initial time: 36 minutes

Initial epoch Updated: at 12 and 18

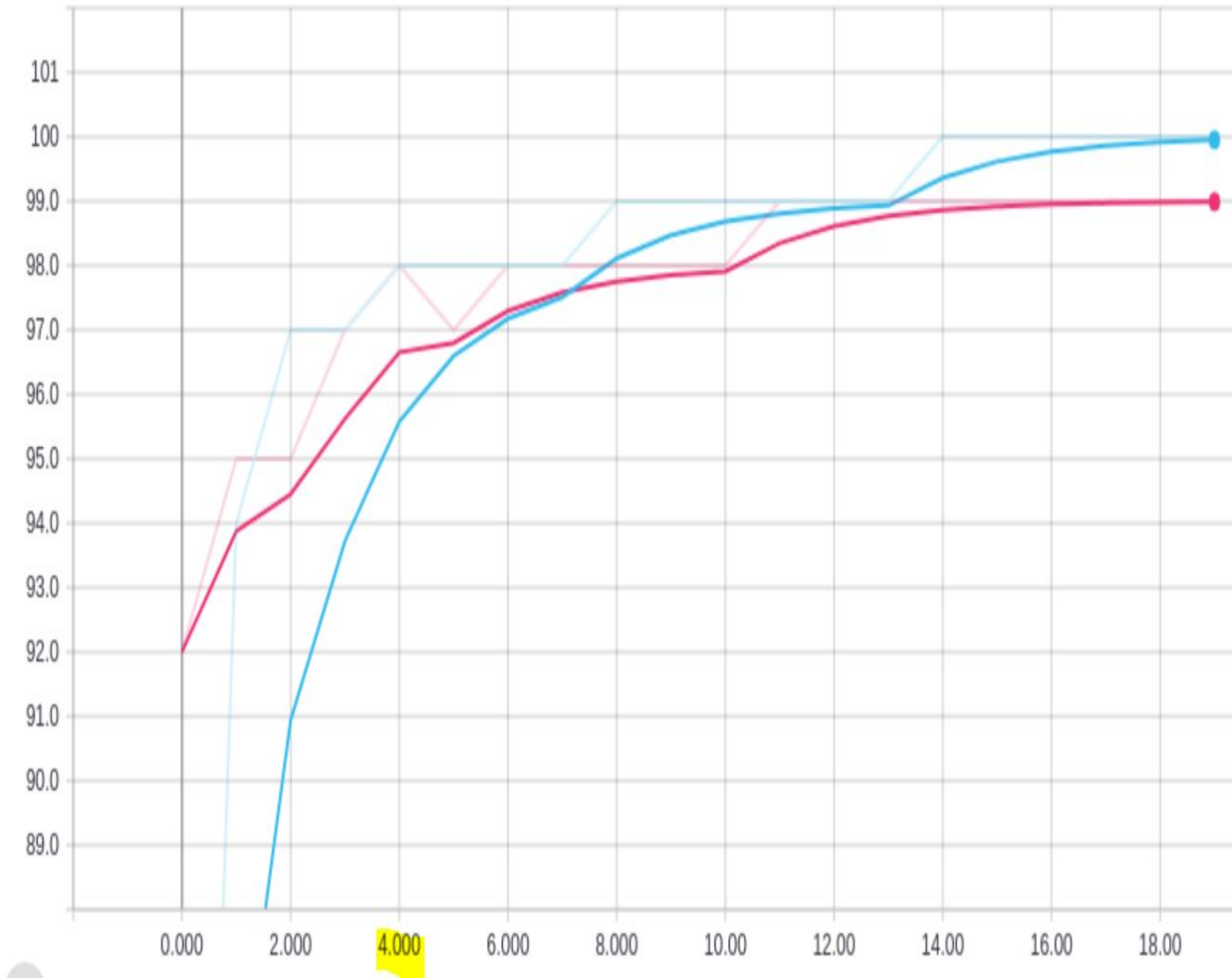
Classes with lowest accuracy (4): T, Space, S, V

LeNet Model

losses



accuracies



LeNet Model

Precision	0.976701
Recall / Accuracy	0.975785
F-score	0.975852
AUC	All greater than 0.9603
Processing time	7 minutes
Error	std= 0.0008129 mean= 0.002219
Update at epoch	None
Stopped at epoch	Epoch 4

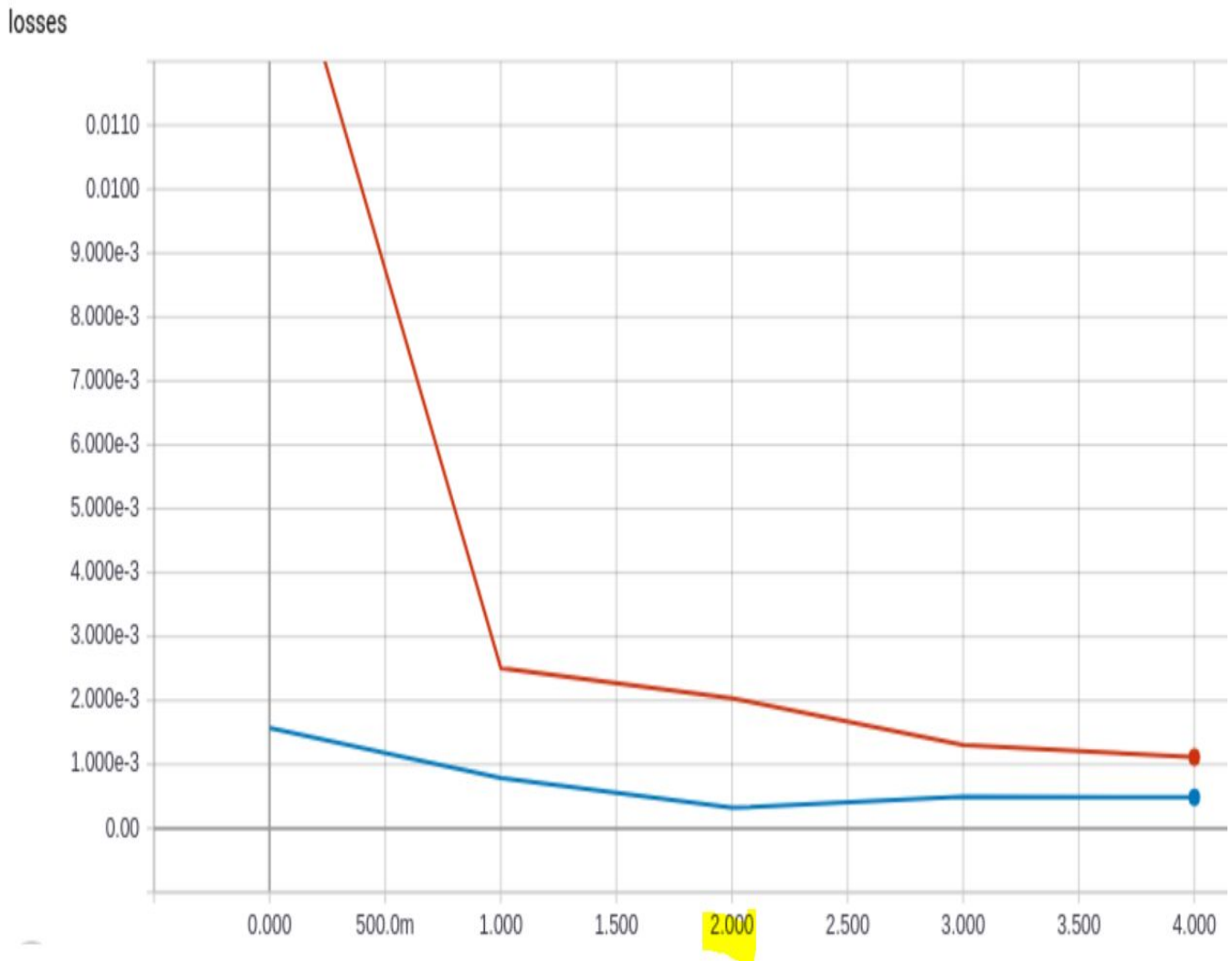
Initial Accuracy: 0.996091

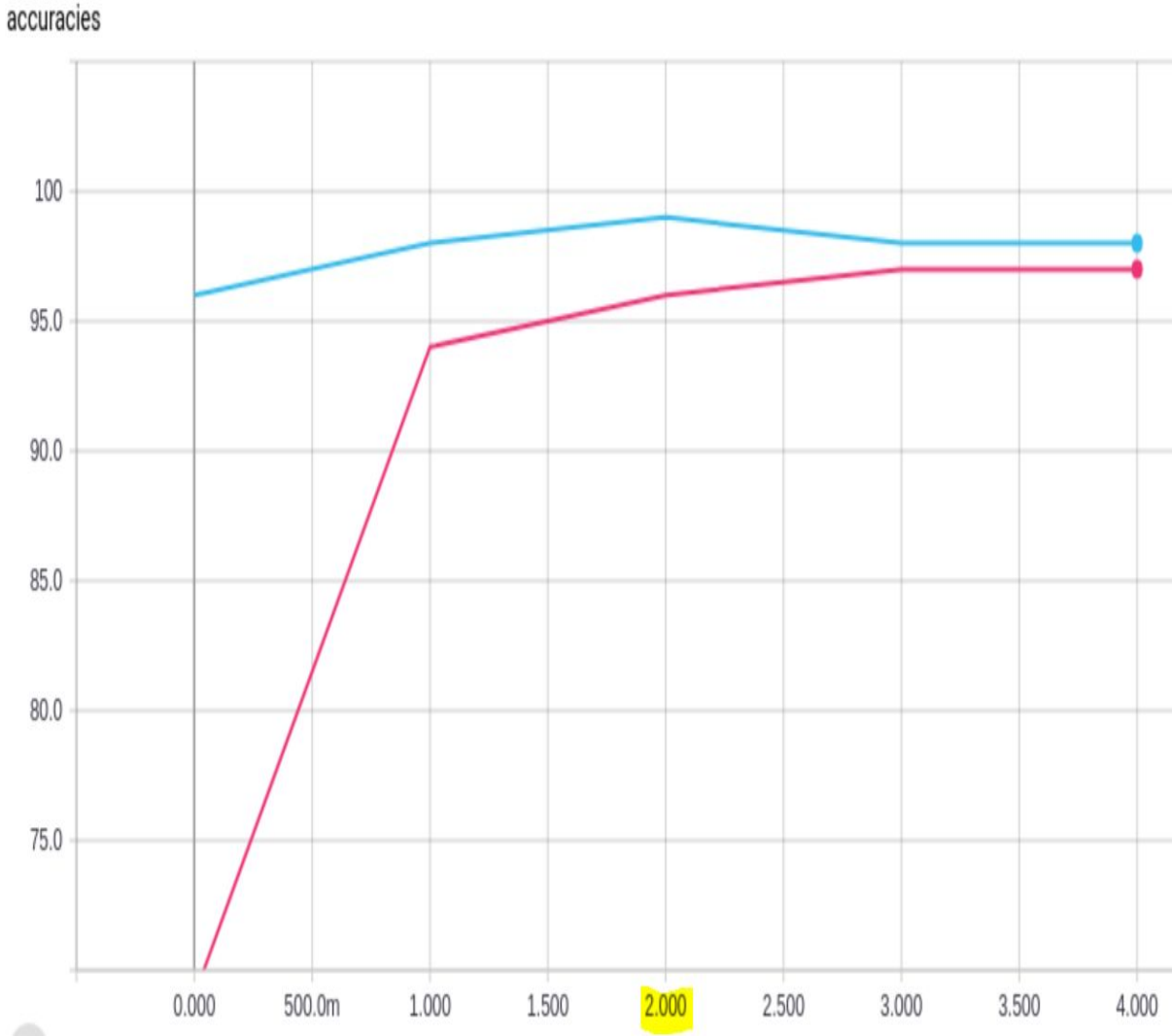
Initial time: 30 minutes

Initial epoch Updated: None

Classes with lowest accuracy (4): P, G, B, Space

AlexNet Model





AlexNet Model

Precision	0.989630
Recall / Accuracy	0.989272
F-score	0.989259
AUC	All greater than 0.9949
Processing time	44 minutes
Error	std = 0.0004457 mean = 0.0007320
Update at epoch	None
Stopped at epoch	Epoch 5

Initial Accuracy: NA

Initial time: NA

Initial epoch Updated: NA

Classes with lowest accuracy (4): P, J, H, M

Model Testing Summary

Metrics	Baseline- 10 epochs	LeNet – 4 epochs
Precision	0.9902369	0.9794684
Recall / Accuracy	0.9901915	0.9786973
F-score	0.9901984	0.9787579
AUC	All greater than 0.9998	All greater than 0.9596
Processing Time	18 minutes	7 minutes
Error	std = 0.0007808 mean = 0.000998	std = 0.000858553 mean = 0.00224975
Classes with lowest accuracy	S, A, B, del	G, P, Space, B

Conclusion

A two convolutional layer network seems to approximate the ground truth of the ASL classification problem.

The baseline model performs with 99% accuracy on the test set. Whereas the LeNet model is faster in time.

For instant translation application, time is important, LeNet is the choice.

Alternative to improve the model

- Do hyperparameters tuning (number of neurons, batch size, weight regularization)
- Design an ensemble model with majority of votes
- Keep the different models simple (not more than two convolutional layers).

Thank you and Happy Holiday