Unit-5

Pretrained Models & other architectures.

I. 1. Lenet
2. AlexNet
3. VGGNet
4. Densenet & Google Net (though not in syllabus)
5. Resnet
6. Transfer Learning

II. Improving Deep Neural Networks.
1. Hyperparameter tuning
2. Regularization & optimization.
3. Data Augmentation techniques.

III. 1. RCNN
1. GAN
2. Reinforcement Learning.

Questions from previous year SET

Q) Sketch the core components of a Reinforcement Learning paradigm (10m)

Q) Evaluate the following pre-trained deep learning models: (10m)
i) Google Net
ii) VGG-16.

(or)

Q) List different hyperparameters used in Deep learning models. Justify the effect of tuning atleast 4 hyperparameters to improvise the model performance. (10m)

Q) Explain the pros & cons of the transfer learning paradigm (10m).

5)

1. Lenet

a) Lenet (LeNet-5) is a CNN proposed by Yann LeCun in 1989 originally designed to demonstrate the task of handwritten digit classification by a CNN.

b) Architecture

i) The lenet CNN architecture has 7 layers : 3 convolutional layers, 2 subsampling layers & 2 fully connected layers. (Input : 32x32x1)

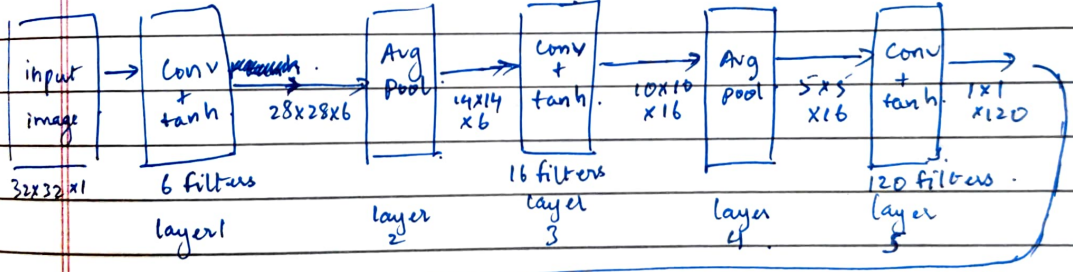
ii) The convolutional layers are used to extract spatial features.

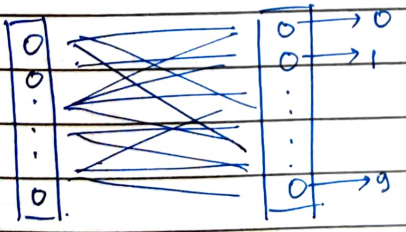
iii) Average pooling is used for subsampling.

iv) tanh function is used as the activation function.

v) The fully connected layers are nothing but Fully connected FFNN used for the final classification.

vi) The 7 layers are arranged as follows





layer 6
FC layer.
(84 neurons)

layer 7
output layer
(softmax)

2) AlexNet

a) AlexNet is a CNN architecture proposed by Alex Krizhevsky in 2012 to classify the images of the ImageNet dataset (more than 1 million images with 1000 class labels).

·) This was the first CNN architecture that used GPU to boost training performance

·) Architecture

i) the AlexNet architecture consists of 12 layers: 5 convolutional layers (ReLU activated), 3 max-pooling layers, 3 fully connected layers, & 1 soft max layer.

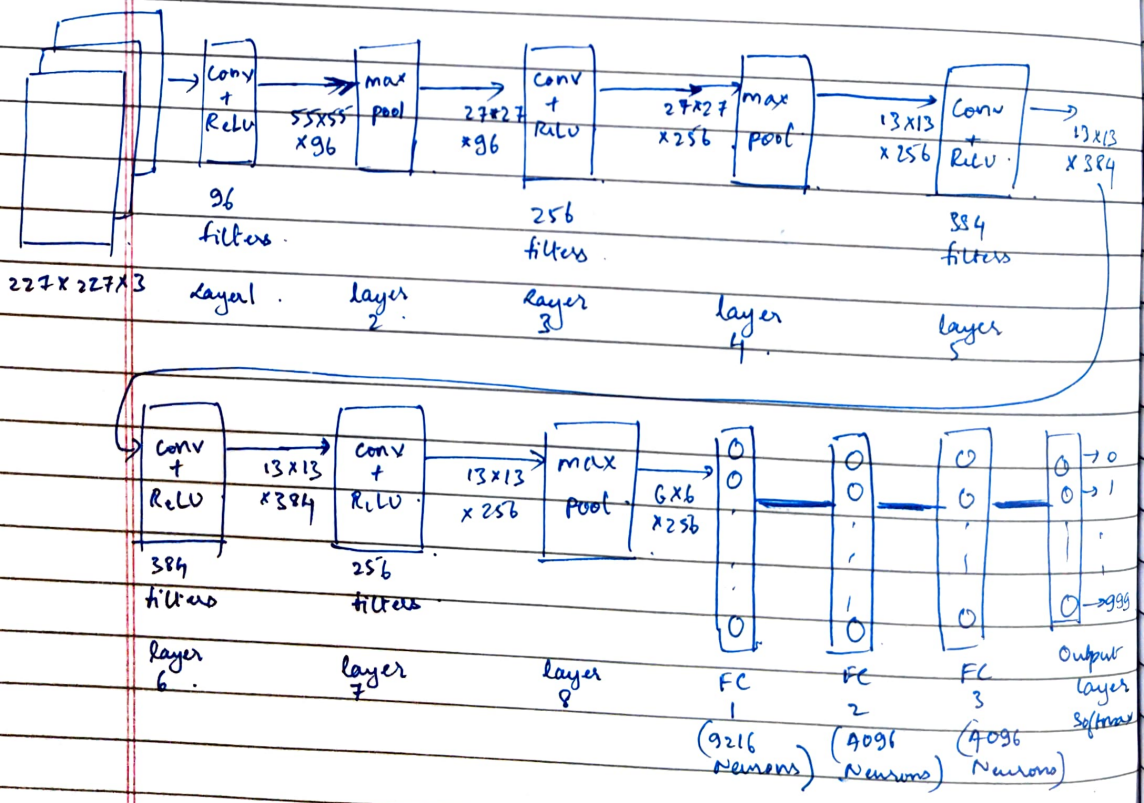
ii) Max pooling is used for subsampling.

iii) ReLU function is used as the activation function.

iv) Input dimension: 227 x 227 x 3 (after padding).

v) Stochastic Gradient Descent (SGD) is used as the optimizer.

vi) Data augmentation like flipping, jittering, cropping, color normalization etc are carried out.



i) C → MP → C → MP → C → C → C → MP → FC → FC → FC
5

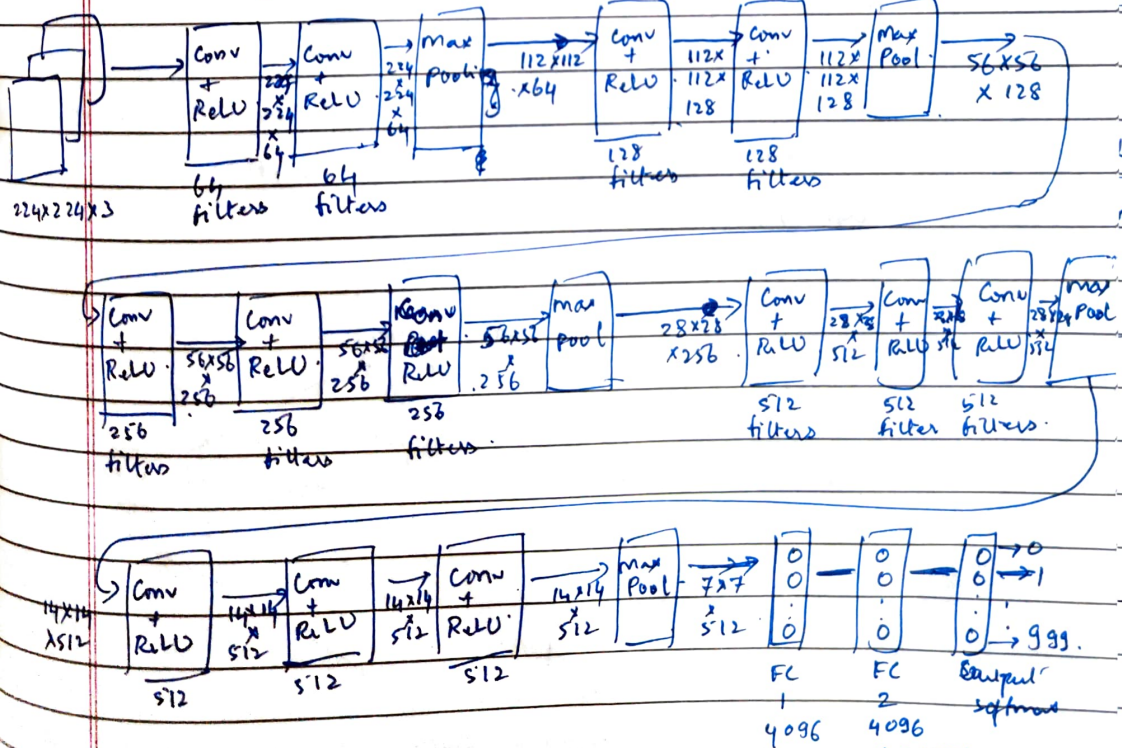
3) VGGNet

- The VGGnet is a deep CNN developed by the Visual Geometry Group in 2014 & one of the most liked models submitted at ILSVRC 2014 (i.e ImageNet competition).
- 2 types VGG architecture
  - i) VGG-16
  - ii) VGG-19
- The numbers 16 & 19 refers to the no. of weight layers of the model (Pooling layer is not considered a weight-layer).
- Both VGG 16 & 19 have a uniform architecture.

3.1) VGG-16 architecture

i) 16 weighted layers: 13 Conv layers (ReLU activation)
+3 FC layers (2 hidden layers → ReLU activation.
1 output layer → softmax activ.)

- 5 pooling layers (max pooling)



CCM → CCM → CCCM → CCCM → CCCM → FC1 → FC2 → S
64 128 256 512 512

3.2) VGG-19 architecture

i) Has the same basic idea as that of VGG 16, but has 3 additional conv layers.

CCM → CCM → CCCCM → CCCCM → CCCCM → FC1 → FC2 → S

4) Google Net

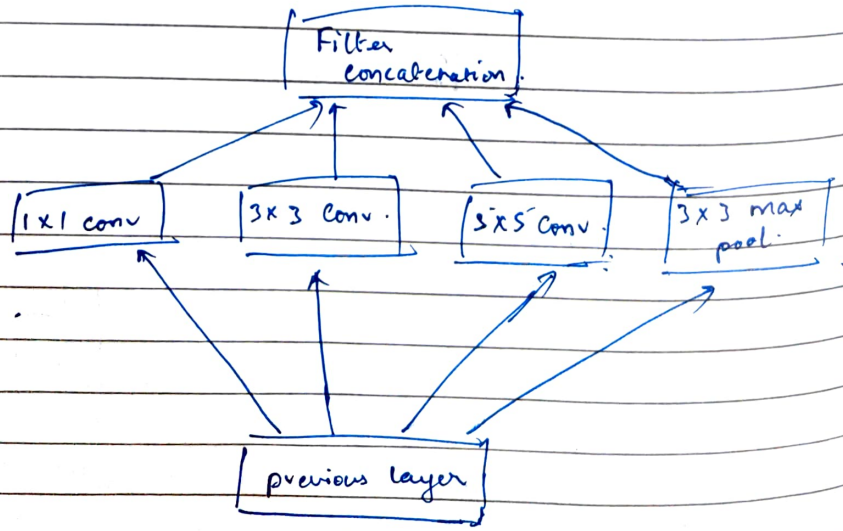
i) Google Net @ inception V1 was proposed by Google in 2014 in the research paper titled "Going Deeper with Convolutions"

ii) It was the winner of the ILSVRC 2014 image classification challenge.

iii) Architectural Features of Google Net

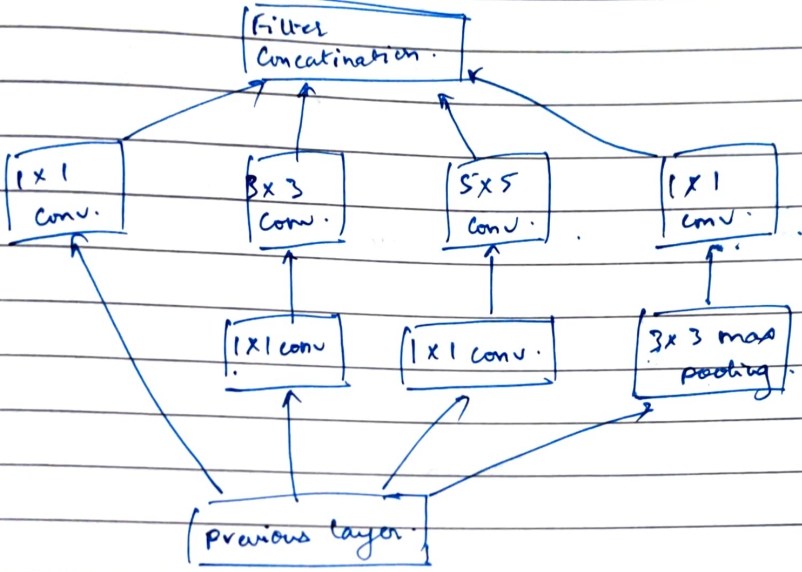
i) The overall architecture is 22 layers deep. The architecture was designed to improve computational efficiency (by decreasing the number of parameters) & allow parallel convolutions of different dimensions in the same block.

ii) To achieve parallel convolutions of different dimensions in the same block, Google Net uses an Inception Module (naive version).



Inception Module, naive version

iii) To decrease the number of parameters (dimension reductions) GoogleNet uses 3 additional 1×1 convolutions in the naive Inception module (i.e. each block) & a Global Average Pooling at the end of the architecture.



Inception module with dimension reductions

) Google net was the winner at the 2014 ILSVRC competition & had the top 5 error rate of 6.67 %. in classification task.

5) Resnet.

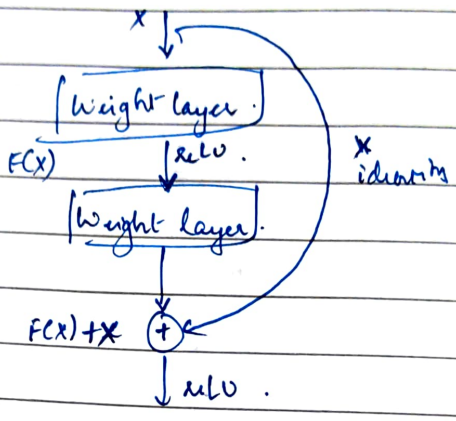
) Resnet also called Residual Network was developed by Microsoft research in 2015.

) In deep neural networks, due to the large no.. of layers, a problem called as vanishing/Exploding Gradients occur due to which the training & test errors increase.

) To solve the problem of vanishing/exploding gradients, the Resnet architecture introduced the concept of Residual Blocks

A residual block uses the concept of Skip Connection, wherein the activations of a layer is connected to further layers by skipping some layers in between.

- The ResNet architecture was built by stacking these residual blocks together



A residual block.

6) Transfer learning (TL)

- Transfer learning is ML technique wherein a model developed for a particular task is reused as the starting point to develop another model for a similar/related task.
- Instead of training a new model from scratch, TL allows you to leverage the learned parameters from a pretrained-model. (i.e use the knowledge gained eg: the knowledge of extracting certain features etc).
- Key concepts related to TL.

i) Pretrained models: There are deep CNNs previously trained on large datasets that require significant computational resources, such as image classification on large image dataset, video/Audio segmentation etc.

ii) Feature Extraction: In TL, the early layers of a pre-trained model are kept (which capture the general features like edges, textures, shapes etc) & only the final layers (usually the FC layers) are retrained as required by the new task.

iii) Fine-tuning: Sometimes, the weights of the early layers from a pre-trained model are fine-tuned (adjusted) helping the model to adjust to the specifics of the new task.

.) Pros of TL.

i) Reduced Training Time: Since the model uses the pretrained weights from a related task, the training time on the new task is significantly reduced.

ii) Improved Performance with limited data: TL is beneficial in tasks with a small dataset.
This is because the pretrained model already passes knowledge from a larger related dataset.

iii) Efficient use of Computational Resources: Since we don't have to train our model from scratch, TL facilitates efficient use of computational resources.

iv) Better Generalization: Models trained using TL often generalize better than models trained from scratch.

.) Cons of TL.

i) Negative Transfer: If the source & the target tasks are not sufficiently related, then TL can harm the performance. This phenomenon known as -ve transfer occurs when the pre-trained model's features are not applicable to the new task.

ii) Overfitting Risk: Fine tuning a pre-trained model on a small dataset can lead to overfitting.

iii) Complexity in model Selection: Determining which pre-trained model to choose, which layers to freeze & fine-tune can

iv) Data privacy concerns : Using a pre-trained model trained on sensitive @ proprietary data raises data privacy concerns.

II) Improving Deep Neural Networks.

i) Hyperparameter Tuning.

- Most dlp learning algorithms have several hyperparameters which control many aspects of the algorithms behavior.
- Some hyperparameters affect the time & space cost of running the algorithm.

  Some hyperparameters affect the quality of the model obtained by the training process
- The hyperparameters are set before the training process & unlike the model parameters (weights & biases), hyperparameters are not learned during the training process
- 2 ways to set hyperparameters.
  - i) Manual Hyperparameter tuning: Manually set the hyperparameters (trial & error) based on some best practice/heuristic.
  - ii) Automatic Hyperparameter Tuning: Based on some algorithm like Grid search, Random search.
- Few common hyperparameters of a dlp learning algorithm (ANN)
  - i) Learning rate (η).
  - ii) Depth of the Neural network: No/.. of layers of the net. (No/.. of hidden layers)
  - iii) Width of a layer: No/.. of neurons in a layer.
  - iv) Convolution Kernel width/dimension
  - v) Implicit zero padding.

vi) Dropout rate

c) Effect of these hyper parameters on the model's capacity.

Hyperparameter	Increases model capacity when...	Reason.
i) No/.. of hidden units (ie no/.. of hidden layers)	increased	Increasing the no/.. of hidden units increases the representational capacity of the model.
ii) Learning rate (η)	tuned optimally.	Too high or too low in' results in a model with low effective capacity due to optimization failure.
iii) No/.. of neurons in a layer	increased	directly proportional to the representational capacity of the model.
iv) Convolution kernel width	increased	Increasing the kernel width increases the no/.. of parameters in the model.
v) Implicit zero padding	increased	Padding before convolution keeps the representation size large.
vi) Dropout rate	decreased	

2) Regularization & Optimization.

2.1) Regularization

a) Regularization techniques are used to prevent the model from overfitting to the training data by adding a penalty term to the loss function.

Overfitting occurs when a model performs well on training data but poorly on unseen data.

·) Commonly used Regularization techniques.

i) L1 & L2 regularization.

ii) Dropout

iii) Batch normalization.

2.2) Optimization.

·) Optimizers (or) Optimization algorithms are techniques to minimize the value of the loss functions effectively as possible & thus return the model's parameters corresponding to that global/local minima.

i.e $W^* = \underset{\forall W}{\text{argmin}}(\text{Loss})$.

·) Commonly used Optimization techniques.

i) Gradient Descent

↳ Stochastic Gradient Descent (SGD).

↳ Mini-Batch Gradient Descent

ii) Adam.

3) Data Augmentation Techniques.

·) These are techniques used to increase the diversity & quantity of the training data for dep learning models.

·) Advantages of data augmentation

i) Improve model's performance.

ii) Reduce overfitting

iii) make models more robust to variations in i/p data.

·) Few Basic image data augmentation techniques

i) Flipping

ii) Rotating.

iii) Cropping.

iv) Scaling

.) Few advanced image data augmentation techniques.

i) Color Jitter

ii) Random Erasing.

iii) Mixup.

IV.

2) Reinforcement learning.

Q) Sketch the core components of a Reinforcement learning paradigm.

Ans.) Reinforcement Learning (RL) is an ML paradigm wherein a learner/agent learns to make decisions by interacting with an environment.

.) Every action of the agent is associated with a reward (either positive or negative). The agent's goal is to maximize cumulative reward overtime.

.) core components of a RL learning paradigm.

i) Agent: The learner / decision maker in the RL environment.
The agent interacts with the environment, takes actions & learns from the outcomes of the actions.

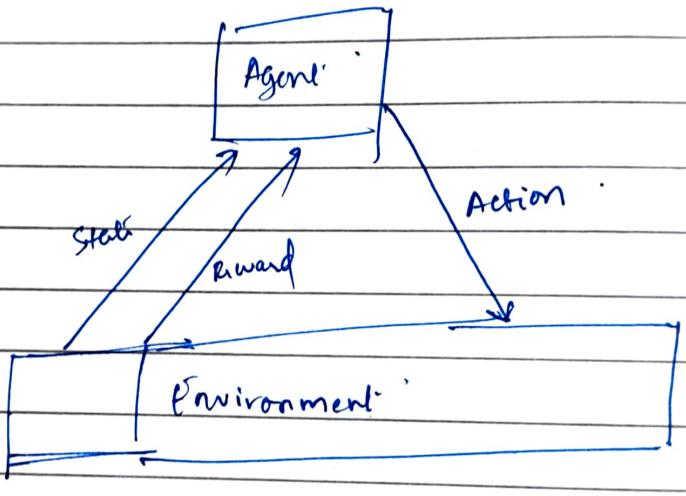
ii) Environment: The external system with which the agent interacts

iii) States (S): The set of all possible states of the environment. A current state provides the representation of the current situation of the environment.

iv) Actions (A): Set of all possible moves an agent can make. The agent selects an action based on the current state & its policy.

v) Policy (π): A strategy that the agent follows to decide actions based on the current state.

vi) Reward (R): A scalar value received after each action indicating the immediate benefit of the action.
Can either be a +ve reward (good action) or -ve reward (bad action).



$$s_0 \xrightarrow[r_0]{a_0} s_1 \xrightarrow[r_1]{a_1} s_2 \xrightarrow[r_2]{a_2} \cdots$$