CMSC 491/691: Computer Vision **Spring 2024**

Lecture 9: 2024-02-28 Neural Networks for Computer Vision

Lecturer: Tejas Gokhale *Scribe: Amanjot Singh*

9.1 Recap of Last Lecture

In the last lecture, we began with machine learning using computer vision. Here we got introduced to image classification. Image classification is based on colors, texture, shapes, edges and features.

Features has 3 main components are Detection , Description and Matching

Challenges in the image classification are Viewpoint Variation, Illumination, Background Clutter, Occlusion ,Pose and Deformation , Inter-Class Variation and Illusions

What is the concept of machine learning ?

It is the concept which includes collection of datasets of images and label. Training the classifier using machine learning algorithms and evaluating the classifier.

What is nearest neighbour classifier - We learnt about distance metric to compare the images and below is the formula-

$$\mathcal{D}(I_1, I_2) = \sum_p |I_1^P - I_2^P| \tag{9.1}$$

What is learning and inference ?

In the learning phase, a developer feeds their model with a dataset so that it can "learn" everything about the data it will be analyzing. In inference phase, the model can make predictions based on data to produce actionable results.

Supervised learning is the simple linear regression model which is represented by below equation -

$$f_\theta(x) = \theta_0 x + \theta_1 \tag{9.2}$$

Here x is the independent variable and θ_0 and θ_0 are the parameter for the best fit in the data

What is the best fit - the best fit is the function that minimizes the squared error between predictions and target value the least-squares

How to minimize the objective w.r.t θ_0 ?

We can do it using an optimizer. Below is the equation-

$$\theta^* = \arg\min_\theta \sum_{i=1}^{N} (f_\theta(x^{(i)}) - y^{(i)})^2$$

Empirical Risk Minimization

In the below figure hypothesis space is the space spanned by the possible number of parameters that define the model and optimizer is used to adjusting the model parameters to minimize a predefined loss function.

$$\theta^* = \arg\min_{\theta} \sum_{i=1}^{N} (\theta_1 x^{(i)} + \theta_0 - y^{(i)})^2$$

Figure 9.1: Linear least squares learning problems

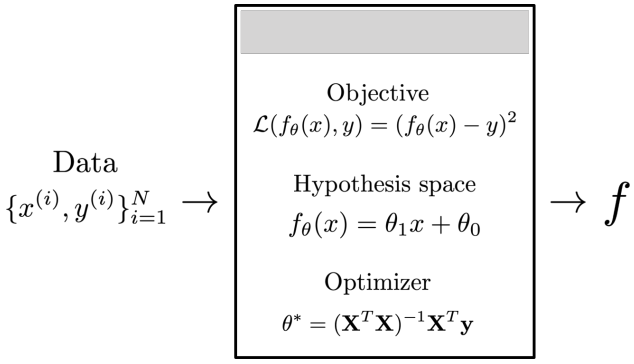


Figure 9.2: Case Study 1: Linear Least Squares

Image classification

In the below figure we have provided the input as the image of the duck. The classifier is trained on the labeled dataset. It recognized the features of a duck such as shape texture and color. It then output a label as duck.

Figure 9.3: Case Study-2: Image Classification

Softmax Regression

Softmax regression, also known as multinomial logistic regression, is a logistic regression model that is used for multi-class classification problems. The conversion of a vector of K scores (logits) into a probability mass function is typically done through the softmax function.It is a normalization function that transforms the raw scores (logits) into a vector of probabilities that sum to 1.

Generalization

The ability to perform well on the previously unobserved data is known as generalization

9.2 Neural Networks for Computer Vision

9.2.1 Linear Regression

Linear regression analysis is used to predict the value of a variable based on the value of another variable. They are used for tasks such as image classification, object detection, and image segmentation. They are inefficient when we have to capture relationships between non-linear relationships between the input features and the output. Below is the equation-

$$f_\theta(x) = \theta_0 x + \theta_1 \tag{9.3}$$

9.2.2 Polynomial Regression

In polynomial regression, an nth-degree polynomial is used to represent the relationship between the independent variable (or variables) and the dependent variable. Polynomial regression is a more flexible approach that can capture nonlinear relationships between the input features and the output.

Basis Function

Basis functions are building blocks for creating more complex functions. In other words, they are a set of k standard functions

What happens if we add more basis functions?

If we add more basis functions it will lead to phenomenon i.e called overfitting. It occurs when we have too high capacity a model, e.g., too many free parameters, too few data points to pin these parameters down.

What is underfitting ?

When the model does not have the capacity to capture the true function, we call this underfitting. An underfit model will have high error on the training points. This error is known as approximation error.

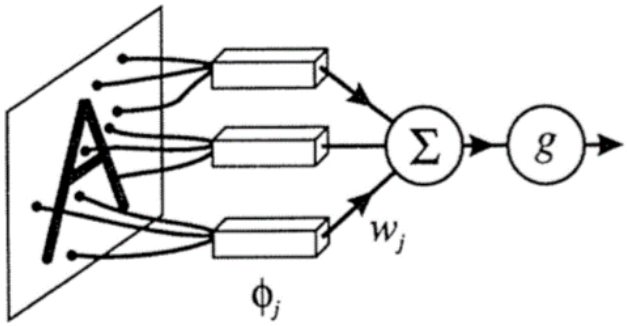
9.2.3 Parametric Approach

A linear function with a fixed set of parameters is used to model the relationship between the input features and the class labels in a parametric approach to linear classification. This method makes the assumption that the input feature-to-decision boundary , which divides classes, is a linear function.

In the linear classifier, the function $f(\mathbf{x}, \mathbf{W}) = \mathbf{W}\mathbf{x}$ maps the input feature vector W is the parametric weights and $f(\mathbf{x}, \mathbf{W})$ to the output, which can be interpreted as the scores or logits for each class."

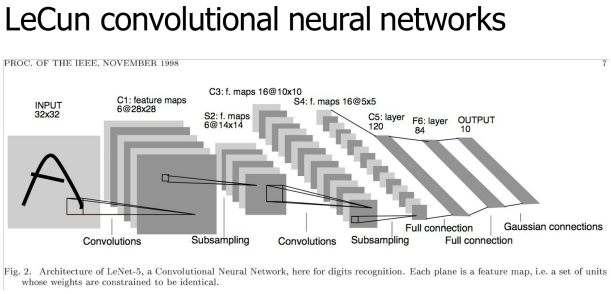
9.2.4 Perceptron

As a basic binary classifier with the ability to learn linear decision boundaries, the perceptron is a fundamental component of artificial neural networks. Frank Rosenblatt created the perceptron in 1957. It computes a weighted sum using a set of input features, each of which has a weight. After that, an activation function—typically a step function—is applied to this sum in order to determine the output class. The perceptron uses methods like the perceptron learning rule to modify its weights during training based on the discrepancy between the true labels and its predictions.



9.2.5 LeCun convolutional neural networks

LeCun convolutional neural networks (CNNs) are a subset of deep learning models that were developed primarily in the late 1980s and early 1990s by Yann LeCun and his associates. Computer vision and pattern recognition tasks have been revolutionized thanks in large part to these networks. The LeNet-5 architecture is a fundamental contribution that was created for challenges involving the recognition of handwritten digits.



9.2.6 ImageNet

ImageNet is a large-scale dataset containing millions of labeled images across thousands of object categories. It was introduced by Prof Princeton, Prof Standford. It has been instrumental in advancing the field of computer vision and machine learning, particularly through its use in training and evaluating deep learning models for tasks like image classification and object detection.

9.2.7 AlexNet

AlexNet is a seminal convolutional neural network (CNN) architecture developed by Krizhevsky, Sutskever, and Hinton in 2012. It has been trained on the large-scale imageNet dataset. It consists of 8 layers architecture but today we have around 100+ layers. It was one of the first architectures to utilize multi-GPU training. This capability allowed faster training times by dividing computations across multiple graphics processing units simultaneously.

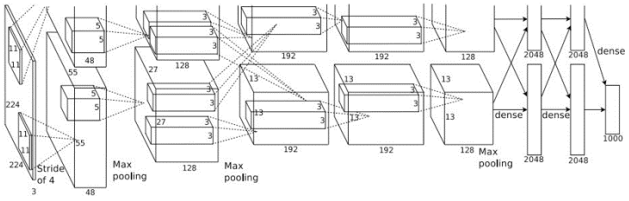
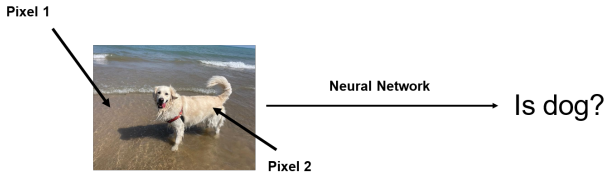


Figure 9.4: AlexNet

9.2.8 Object Recoginition

The convolutional neural networks (CNNs) are trained to recognize and categorize items in pictures or videos as part of the object recognition process in neural networks. CNNs use convolutional and pooling layers to extract pertinent characteristics from input images. In the above figure for the object recognition using



neural network we are using the picture of the dog, the neural network learns to identify as a dog by analyzing features. When presented the image of dog the neural network will extract the features like paw,fur etc and contrasts them with trained patterns. The neural network outputs the prediction that the image contains a dog if the extracted features closely reflect those in the feature space associated with dogs.

9.3 Computation in a neural network

In the neural network each neuron generates an output by combining the input features with matching weights and biases. Usually, one output neuron in linear regression produces the predicted value directly because of its linear decision boundary limitation, a single-layer linear neural network is unable to solve the XOR (exclusive OR) problem. Below is the equation-

$$y_j = \sum_i w_{ij} x_i + b_j \tag{9.4}$$

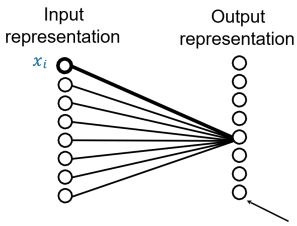


Figure 9.5: Linear Layer

In this equation:

- y_j represents the output of neuron j.
- i iterates over all neurons in the previous layer.
- x_i represents the output of neuron i in the previous layer.
- w_{ij} represents the weight of the connection between neuron i and neuron j.
- b_j represents the bias term for neuron j.

Activation Functions

In a neural network, activation functions are mathematical functions that are applied to each neuron's output. They give the network non-linearities, which help it understand and depict intricate relationships found in the data. The answer to issues like XOR in neural networks lies in non-linearities in the form of activation functions. Neural networks can model complex relationships in the data when non-linear activation functions are incorporated. This is crucial for solving tasks involving non-linear patterns or decision boundaries.

9.3.1 Step function

Step function output binary values(0 or 1) whether the input surpasses the threshold. Gradient Gradient descent is not used with the step function because the step it is not continuous and derivative g(y) is 0 .

$$g(y) = \begin{cases} 1, & \text{if} \quad y > 0 \\ 0, & \text{otherwise} \end{cases}$$

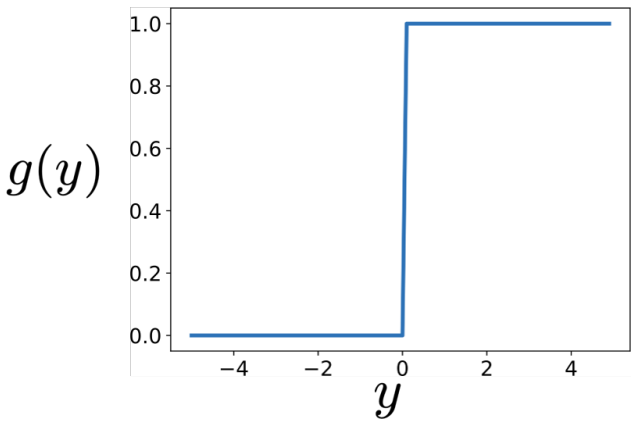


Figure 9.6: Step activation function

9.3.2 Sigmoid function

A smooth, S-shaped curve known as the sigmoid function maps real-valued inputs to the interval (0, 1); it is widely used in neural networks to introduce non-linearity and interpret outputs as probabilities. Large positive or large negative inputs cause it to saturate.As the optimization procedure goes on and the parameters converge towards the loss function's minimum, the gradient's magnitude might get closer to zero

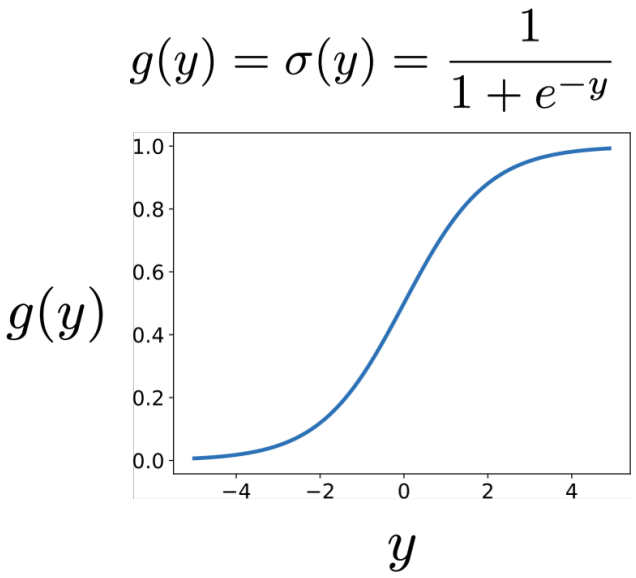


Figure 9.7: Sigmoid activation function

9.3.3 Rectified linear unit function

In neural networks, the Rectified Linear Unit, or ReLU, is a widely used activation function. It permits infinite output, which improves learning effectiveness. When compared to alternatives like tanh, ReLU is simpler to implement and frequently results in faster convergence. But in the negative region, it may have dead neurons, which could impair learning. ReLU's overall effectiveness keeps it the default option in many models, even with this drawback.

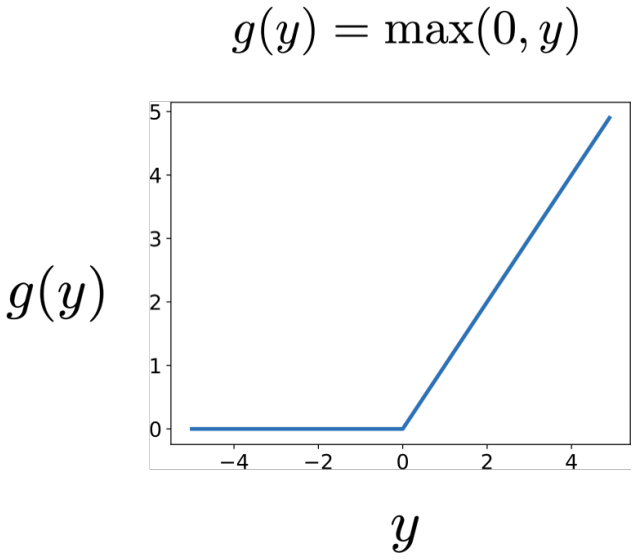


Figure 9.8: ReLu activation function

9.3.4 Leaky ReLU function

Leaky ReLU is a variant of the ReLU activation function. It permits a small, non-zero gradient for negative inputs, unlike ReLU, which sets them to zero. This gradient is usually defined by a small slope parameter "a" (e.g., 0.02). This help with problems brought on by zero gradients and keeps neurons from going dead in the negative region because Leaky ReLU guarantees non-zero gradients throughout the input space

$$g(y) = \begin{cases} \max(0, y), & \text{if} \quad y \geq 0 \\ a \min(0, y), & \text{if} \quad y < 0 \end{cases}$$

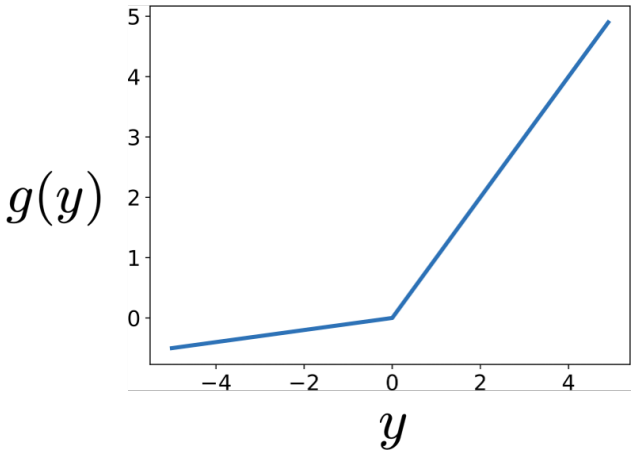
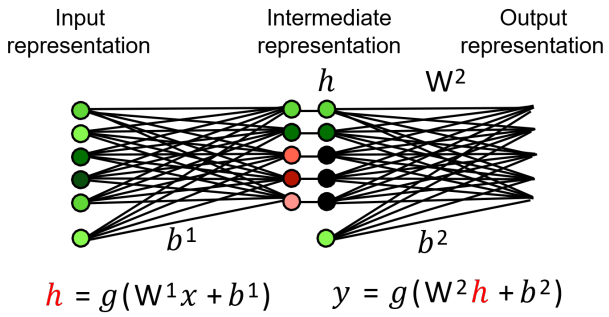


Figure 9.9: Enter Caption

9.3.5 Stacking Layers

In neural networks, stacking layers refers to the sequential arrangement of several layers of neurons. Each layer uses bias, activation functions, and weighted connections to process inputs. The output of the first layer (intermediate layer) is represented by the function $h = g(W_1 x + b_1)$, where g can be any activation function, including ReLU. The final output is provided by the function $y = g(W_2 h + b_2)$. During training, the network learns to map inputs to the appropriate outputs by modifying weights and biases. Thus using multiple layers introduces non-linearity as output of one layer serves as input of next layer.



9.3.6 Deep Nets

Deep Nets are multi-layered, complex architectures that facilitate learning from data and hierarchical feature extraction. For the case of dog image that we have used the deeper layers may identify more specialized features like the dog's paw shape or fur texture, while earlier layers may identify more general features like edges and colors. Through the analysis of these extracted features, the breed of the dog can be predicted by the network.

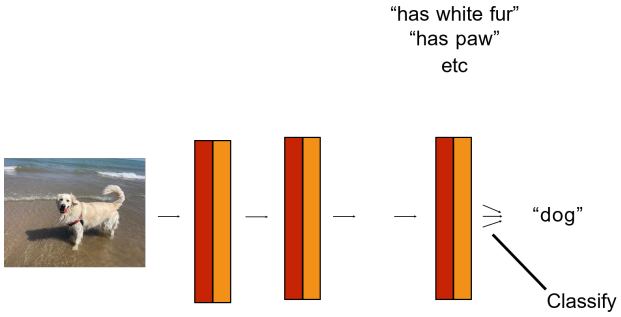


Figure 9.10: Enter Caption