Day 6: Neural Networks

Summer STEM: Machine Learning

Department of Electrical and Computer Engineering
NYU Tandon School of Engineering
Brooklyn, New York

Outline

Machine Learning Problem Pipeline

1. Formulate the problem: regression, classification, or others?
2. Gather and visualize the data
3. Design the model and the loss function
4. Train your model
    - (a) Perform feature engineering
    - (b) Construct the design matrix
    - (c) Choose regularization techniques
    - (d) Tune hyper-parameters using a validation set
    - (e) If the performance is not satisfactory, go back to step (a).
5. Evaluate the model on a test set

Outline

NYU TANDON SCHOOL OF ENGINEERING

Multiclass Classificaiton

- Previous model: $f(\mathbf{x}) = \sigma(\mathbf{w}^T \phi(\mathbf{x}))$
- Representing Multiple Classes:
  - One-hot / 1-of-K vectors, ex : 4 Class
  - Class 1 : $\mathbf{y} = [1, 0, 0, 0]$
  - Class 2 : $\mathbf{y} = [0, 1, 0, 0]$
  - Class 3 : $\mathbf{y} = [0, 0, 1, 0]$
  - Class 4 : $\mathbf{y} = [0, 0, 0, 1]$

Multiclass Classificaiton

- Previous model: $f(\mathbf{x}) = \sigma(\mathbf{w}^T \phi(\mathbf{x}))$
- Representing Multiple Classes:
  - One-hot / 1-of-K vectors, ex : 4 Class
  - Class 1 : $\mathbf{y} = [1, 0, 0, 0]$
  - Class 2 : $\mathbf{y} = [0, 1, 0, 0]$
  - Class 3 : $\mathbf{y} = [0, 0, 1, 0]$
  - Class 4 : $\mathbf{y} = [0, 0, 0, 1]$
- Multiple outputs: $f(\mathbf{x}) = \text{softmax}(W^T \phi(\mathbf{x}))$
- Shape of $W^T \phi(\mathbf{x})$: $(K, 1) = (K, D) \times (D, 1)$
- $\text{softmax}(\mathbf{z})_k = \frac{e^{z_k}}{\sum_j e^{z_j}}$

Multiclass Classificaiton

- Multiple outputs: $f(\mathbf{x}) = \text{softmax}(\mathbf{z})$ with $\mathbf{z} = W^T\phi(\mathbf{x})$
- $\text{softmax}(\mathbf{z})_k = \frac{e^{\mathbf{z}_k}}{\sum_j e^{\mathbf{z}_j}}$
- Softmax example: If $\mathbf{z} = \begin{bmatrix} -1 \\ 2 \\ 1 \\ -4 \end{bmatrix}$ then,

$$\text{softmax}(z) = \begin{bmatrix} \frac{e^{-1}}{e^{-1}+e^{2}+e^{1}+e^{-4}} \\ \frac{e^{2}}{e^{-1}+e^{2}+e^{1}+e^{-4}} \\ \frac{e^{1}}{e^{-1}+e^{2}+e^{1}+e^{-4}} \\ \frac{e^{-4}}{e^{-1}+e^{2}+e^{1}+e^{-4}} \end{bmatrix} \approx \begin{bmatrix} 0.035 \\ 0.704 \\ 0.259 \\ 0.002 \end{bmatrix}$$

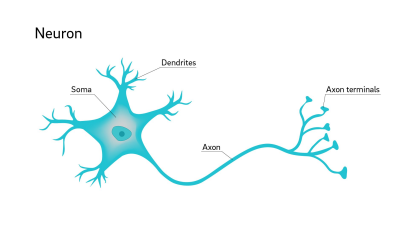
Cross-entropy

- Multiple outputs: $\hat{\mathbf{y}}_{\mathbf{i}} = \text{softmax}(W^T \phi(\mathbf{x}_i))$
- Cross-Entropy: $J(W) = -\sum_{i=1}^{N} \sum_{k=1}^{K} \mathbf{y}_{ik} log(\hat{\mathbf{y}}_{ik})$
- Example : K $= 4$

$$\text{If, } \mathbf{y}_i = [0, 0, 1, 0] \text{ then, } \sum_{k=1}^{K} \mathbf{y}_{ik} log(\hat{\mathbf{y}}_{ik}) = log(\hat{\mathbf{y}}_{i3})$$

Outline

Biological Neuron



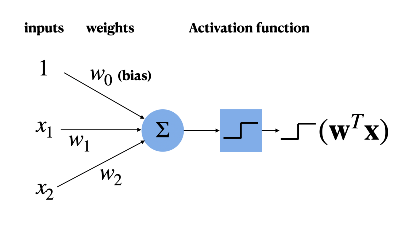
Source: David Baillot/UC San Diego

Mathematical Neuron: Perceptron

Biological Neuron:

- A neuron can receive electrochemical signals from other neurons;
- A neuron fires once its accumulated electric charge passes a certain threshold.
- Neurons that fire together wire together.

Mathematical Neuron (Perceptron)



Relation to Logistic Regression

What if we use the sigmoid function as the activation?

$$f(\mathbf{x}) = \sigma(\mathbf{w}^T \mathbf{x})$$

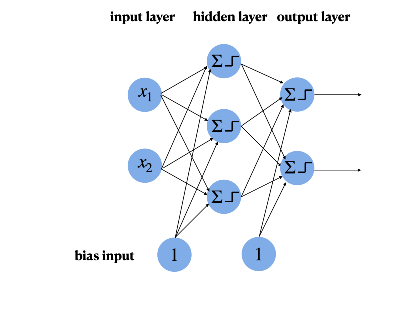
Decision boundary is a line: how is this supposed to revolutionize machine learning?

Multi-layer Perceptron (MLP)

We need more neurons and we need to connect them together!

- Many ways to do that...
- Today: multi-layer perceptron/fully connected feed-forward network.

MLP Example



- What is the shape of the input and output?
- How many parameters does this model have?
- What activation function would you use for the output layer? Why?

More about MLPs

- Many choices for the activation function: Sigmoid, Tanh, ReLU, Swish, etc.
- Many choices for the number of hidden layers and the number of neurons per layer.
- MLPs can approximate any continuous function given enough data.
- MLPs can overfit, but we know many effective ways of regularization.

Exercise: PyTorch Basics

Open `demo_pytorch_basics.ipynb`

Outline

Deep Learning

What does deep learning stand for?

- Deep: Neural network architectures with many hidden layers.
- Learning: Optimizing model parameters given a dataset.

In general, the deeper the model is, the more parameters we need to learn and the more data is needed.

NYU TANDON SCHOOL OF ENGINEERING

Large-Scale Machine Learning

For deep learning systems to perform well, large datasets are required

- COCO 330K images
- ImageNet 14 million images

Challenges:

- Memory limitation: GeForce RTX 2080 Ti has 11 GB memory, while ImageNet is about 300 GB.
- Computation: Calculating gradients for the whole dataset is computationally expensive (slow), and we need to do this many times.

Stochastic Gradient Descent

Idea: Instead of calculating the gradients from the whole dataset, do it only on a subset.

- Randomly select B samples from the dataset
- The loss for this subset

$$\tilde{J}(\mathbf{w}) = \frac{1}{B}\sum_{i=1}^{B} \|y - \hat{y}_i\|^2$$

- Update Rule

  $Repeat\{$

  $$\mathbf{w}_{new} = \mathbf{w} - \alpha \nabla \tilde{J}(\mathbf{w})$$

  $\}$

Yet Another Hyper-Parameter

This gives a noisy gradient

$$\nabla \tilde{J}(\mathbf{w}) = \nabla J(\mathbf{w}) + \epsilon$$

- **SGD**: $B = 1$, gives very noisy gradients
- **(batch) GD**: $B = N$, $\epsilon = 0$, expensive to compute
- **Mini-batch GD**: Pick a small B, typical values are $32, 64$, rarely more than 128 for image inputs

Some Noise Helps

Even if we can, we rarely set $B = N$. In fact, some noise in the gradients might help to

- escape from local minima,
- escape from saddle points, and
- improve generalization.

Outline

Overparameterized Models

- Modern deep learning models are heavily overparameterized, i.e. the number of learnable parameters is much larger than the number of the training samples.

Overparameterized Models

- Modern deep learning models are heavily overparameterized, i.e. the number of learnable parameters is much larger than the number of the training samples.
  - ResNet: State-of-the-art vision model, 10-60 million parameters

Overparameterized Models

- Modern deep learning models are heavily overparameterized, i.e. the number of learnable parameters is much larger than the number of the training samples.
  - ResNet: State-of-the-art vision model, 10-60 million parameters
  - GPT-3: State-of-the-art language model, 175 billion parameters

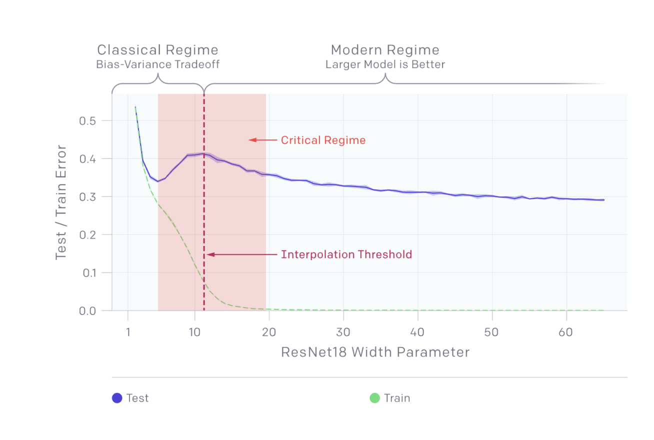
Overparameterized Models

- Modern deep learning models are heavily overparameterized, i.e. the number of learnable parameters is much larger than the number of the training samples.
  - ResNet: State-of-the-art vision model, 10-60 million parameters
  - GPT-3: State-of-the-art language model, 175 billion parameters
- Conventional wisdom: Such models overfit.

Overparameterized Models

- Modern deep learning models are heavily overparameterized, i.e. the number of learnable parameters is much larger than the number of the training samples.
  - ResNet: State-of-the-art vision model, 10-60 million parameters
  - GPT-3: State-of-the-art language model, 175 billion parameters
- Conventional wisdom: Such models overfit.
- It is not the case in practice!

Double Descent Curve



Source: OpenAI

Outline

Let's solve the mini-project with MLPs!

Open `lab_mlp_fish_market.ipynb`