CS230: Lecture 2
Deep Learning Intuition
Kian Katanforoosh
Recap
Kian Katanforoosh, Andrew Ng, Younes Bensouda Mourri

Model
= Architecture + Parameters

Input → Model → Output

Things that can change
- activation function
- optimizer
- Hyperparameters
- ...

Gradients → Loss → 0
Logistic Regression as a Neural Network

\[
\begin{pmatrix}
255 \\
231 \\
... \\
94 \\
142 \\
\end{pmatrix}
\]

\[
\begin{align*}
\frac{255}{255} & \rightarrow x_1^{(i)} \\
\frac{231}{255} & \rightarrow x_2^{(i)} \\
\frac{...}{255} & \rightarrow \ldots \\
\frac{94}{255} & \rightarrow x_{n-1}^{(i)} \\
\frac{142}{255} & \rightarrow x_n^{(i)}
\end{align*}
\]

\[
\sigma \left( w^T x^{(i)} + b \right) \rightarrow 0.73 > 0.5 \rightarrow \text{“it’s a cat”}
\]

Kian Katanforoosh, Andrew Ng, Younes Bensouda Mourri
The image shows a multi-class classification problem with three classes: Dog, Cat, and Giraffe. The input image is represented as a vector of pixel values, which are normalized by dividing each pixel by 255. The pixel values are as follows:

- Top left: 255
- Top middle: 231
- Top right: ...
- Middle left: 94
- Middle right: 142

Each pixel value is normalized and then fed into the input layer of a neural network. The network consists of linear transformations followed by a non-linear activation function, typically a sigmoid function, represented by \( \sigma \). The output for each class is determined by the activation function applied to the weighted sum of the input features, plus a bias term.

- For Dog: \( w^T x^{(i)} + b \rightarrow 0.12 \rightarrow \text{Dog?} \) with 0.12 > 0.5
- For Cat: \( w^T x^{(i)} + b \rightarrow 0.73 \rightarrow \text{Cat?} \) with 0.73 > 0.5
- For Giraffe: \( w^T x^{(i)} + b \rightarrow 0.04 \rightarrow \text{Giraffe?} \) with 0.04 < 0.5

The output probabilities indicate the likelihood of each class, and the decision is made by comparing these probabilities to a threshold (0.5 in this case).
Neural Network (Multi-class)

\[
\begin{pmatrix}
255 \\
231 \\
... \\
94 \\
142
\end{pmatrix}
\xrightarrow{/255}
\begin{align}
x_1^{(i)} \\
x_2^{(i)} \\
... \\
x_{n-1}^{(i)} \\
x_n^{(i)}
\end{align}
\xrightarrow{w^T x^{(i)} + b}
\begin{pmatrix}
\sigma \\
\sigma \\
\sigma
\end{pmatrix}
\]
Neural Network (1 hidden layer)

\[
\begin{pmatrix}
255 \\
231 \\
... \\
94 \\
142 \\
\end{pmatrix}
\]

\[
\frac{\mathbf{x}^{(i)}}{255}
\]

\[
\begin{pmatrix}
x_1^{(i)} \\
x_2^{(i)} \\
... \\
x_{n-1}^{(i)} \\
x_n^{(i)} \\
\end{pmatrix}
\]

\[
\begin{pmatrix}
a_1^{[1]} \\
a_2^{[1]} \\
a_3^{[1]} \\
a_1^{[2]} \\
\end{pmatrix}
\]

\[
\mathbf{a}^{[1]} \\
\mathbf{a}^{[2]}
\]

output layer

\[
0.73
\]

\[
\frac{0.73 > 0.5}{\text{Cat}}
\]

Kian Katanforoosh, Andrew Ng, Younes Bensouda Mourri
Deeper network: Encoding output layer

\[
x_1^{(i)} \rightarrow a_1^{[1]} \\
x_2^{(i)} \rightarrow a_2^{[1]} \\
x_3^{(i)} \rightarrow a_3^{[1]} \\
x_4^{(i)} \rightarrow a_4^{[1]}
\]

Hidden layer

\[
\hat{y}^{(i)}
\]

Technique called “encoding”
Let’s build intuition on concrete applications
Today’s outline

We will learn how to:

- Analyze a problem from a deep learning approach
- Choose an architecture
- Choose a loss and a training strategy

I. Day’n’Night classification
II. Face Recognition
III. Art generation
IV. Keyword Spotting
V. Image Segmentation
**Goal:** Given an image, classify as taken “during the day” (0) or “during the night” (1)

1. **Data?** 10,000 images
2. **Input?** Resolution? (64, 64, 3)
3. **Output?** y = 0 or y = 1
4. **Architecture?** Shallow network should do the job pretty well
5. **Loss?**
   \[
   L = y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})
   \]
**Face Verification**

**Goal**: A school wants to use Face Verification for validating student IDs in facilities (dinning halls, gym, pool …)

1. **Data?**
   Picture of every student labelled with their name

2. **Input?**
   Resolution?
   (412, 412, 3)

3. **Output?**
   \[ y = 1 \] (it’s you)
   or
   \[ y = 0 \] (it’s not you)
Goal: A school wants to use Face Verification for validating student IDs in facilities (dinning halls, gym, pool …)

4. What architecture?

Simple solution:

- Database image
- Input image
- Compute distance pixel per pixel
- If less than threshold then y=1

Issues:

- Background lighting differences
- A person can wear make-up, grow a beard…
- ID photo can be outdated
**Goal**: A school wants to use Face Verification for validating student IDs in facilities (dinning halls, gym, pool ...)

**4. What architecture?**

Our solution: encode information about a picture in a vector

We gather all student faces encoding in a database. Given a new picture, we compute its distance with the encoding of card holder.
Face Recognition

Goal: A school wants to use Face Verification for validating student IDs in facilities (dinning hall, gym, pool …)

4. Loss? Training?
We need more data so that our model understands how to encode:
Use public face datasets

What we really want: similar encoding
So let's generate triplets:

\[ L = \|\text{Enc}(A) - \text{Enc}(P)\|_2^2 - \|\text{Enc}(A) - \text{Enc}(N)\|_2^2 \]

Kian Katanforoosh, Andrew Ng, Younes Bensouda Mourri
Recap: Learning Process

Model = Architecture + Parameters

Input

anchor  positive  negative

Output

Loss

\[ L = \left\| \text{Enc}(A) - \text{Enc}(P) \right\|_2^2 - \left\| \text{Enc}(A) - \text{Enc}(N) \right\|_2^2 + \alpha \]

Enc(A)  Enc(P)  Enc(N)
**Face Recognition**

**Goal**: A school wants to use Face Identification for recognize students in facilities (dinning hall, gym, pool …)

K-Nearest Neighbors

**Goal**: You want to use Face Clustering to group pictures of the same people on your smartphone

K-Means Algorithm

Maybe we need to detect the faces first?
**Goal:** Given a picture, make it look beautiful

1. **Data?**
   - Let’s say we have any data

2. **Input?**
   - content image

3. **Output?**
   - style image
   - generated image
4. **Architecture?**

We want a model that **understands images** very well.

We load an **existing model trained on ImageNet** for example.

When this image forward propagates, we can get information about its content & its style by inspecting the layers.

5. **Loss?**

\[
L = \left\| \text{Content}_C - \text{Content}_G \right\|^2_2 + \left\| \text{Style}_S - \text{Style}_G \right\|^2_2
\]

We are not learning parameters by minimizing L. We are learning an image!
Art generation (Neural Style Transfer)

Correct Approach

\[ L = \|Content_C - Content_G\|^2 + \|Style_S - Style_G\|^2 \]

We are not learning parameters by minimizing L. We are learning an image!
Speech recognition: Keyword Spotting

Goal: Given an audio speech, detect the word “lion”.

1. Input?

2. Output?
   
   \[ y = 0 \text{ (there is “lion”) or } y = 1 \text{ (there isn’t “lion”) } \]
   
   \[ y = (0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0) \]

3. Data?

   Many audio recordings (“words”)
Speech recognition: Keyword Spotting

**Goal:** Given an audio speech, detect the word “lion”.

4. **What architecture?**

\[
L = \left\| \text{Enc}(A) - \text{Enc}(P) \right\|_2^2 - \left\| \text{Enc}(A) - \text{Enc}(N) \right\|_2^2 + \alpha
\]

Threshold: 0.6

0.12 0.01 0.27 ... ... ... ... ... 0.21 **0.92** 0.43 ... ... ... ... ... Threshold: 0.6

Kian Katanforoosh, Andrew Ng, Younes Bensouida Mourri
Duties for next week

For Thursday 01/25, 9am:

C1M3
• Quiz: Shallow Neural Networks
• Programming Assignment: Planar data classification with one-hidden layer

C1M4
• Quiz: Deep Neural Networks
• Programming Assignment: Building a deep neural network - Step by Step
• Programming Assignment: Deep Neural Network Application

Project
• Start the project!
• Fill-in AWS Form to get GPU credits