CS230: Lecture 2
Decision making in AI projects

Kian Katanforoosh
Recap of the week
Things that can change
- Activation function
- Optimizer
- Hyperparameters
- ...

Kian Katanforoosh
Logistic Regression as a Neural Network

\[
\begin{pmatrix}
255 \\
231 \\
\vdots \\
94 \\
142
\end{pmatrix}
\longrightarrow
\begin{array}{c}
x_1^{(i)} \\
x_2^{(i)} \\
\vdots \\
x_{n-1}^{(i)} \\
x_n^{(i)}
\end{array}
\xrightarrow{\text{/255}}
\begin{pmatrix}
x_1^{(i)} \\
x_2^{(i)} \\
\vdots \\
x_{n-1}^{(i)} \\
x_n^{(i)}
\end{pmatrix}
\xrightarrow{w^T x^{(i)} + b}
\sigma
\xrightarrow{0.73 > 0.5}
\text{“it’s a cat”}

Kian Katanforoosh
A diagram illustrating the process of classifying images using a multi-class classifier. The input image is represented by a vector of pixel values, which are then transformed into a set of features $x^{(i)}$, $x_1$, $x_2$, ..., $x_{n-1}$, $x_n$. Each feature is multiplied by a weight vector $w^T$, added a bias $b$, and passed through an activation function $\sigma$ (e.g., sigmoid). The output of the activation function is compared to thresholds to determine the class. For example, if $0.73 > 0.5$, the image is classified as a cat. The values $0.12$ and $0.04$ also indicate the likelihood of the image being a dog and a giraffe, respectively.
Neural Network (Multi-class)

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

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

\[
\begin{bmatrix}
x_{1}^{(i)} \\
x_{2}^{(i)} \\
... \\
x_{n-1}^{(i)} \\
x_{n}^{(i)}
\end{bmatrix}
\]

\[
\begin{align*}
a_1^{[1]} \\
a_2^{[1]} \\
a_3^{[1]}
\end{align*}
\]

\[
\begin{align*}
a_1^{[2]} &\rightarrow 0.73 \\
0.73 &> 0.5
\end{align*}
\]

Cat
Deeper network: Encoding output layer

Technique called “encoding”
Summary of learnings: Introduction

• A **model** is defined by its **architecture** and its **parameters**.

• The labelling strategy matters to successfully train your models. For example, if you’re training a 3-class (dog, cat, giraffe) classifier under the constraint of one animal per picture, you might use **one-hot vectors** to label your data.

• We introduced a set of **notations** to differentiate indices for neurons, layers and examples.

• In deep learning, **feature learning** replaces **feature engineering**.
Recap of the week

Let’s now talk about decision making and build intuition on concrete applications
What skills matter to carry out AI projects?

The AI project development lifecycle

[Workera (2020): Find out more about AI tasks, roles, and skills in the AI Career Pathways report: www.workera.ai/candidates/report/]

Kian Katanforoosh
What skills matter to carry out AI projects?

[Workera (2020): Find out more about AI tasks, roles, and skills in the AI Career Pathways report: www.workera.ai/candidates/report/]

Kian Katanforoosh
The necessary skills to carry out the tasks of the AI project development lifecycle are a combination of scientific, engineering, behavioral, and decision making skills.

In the rest of this presentation, we will illustrate AI decision making skills through real case studies. The goal is to learn war stories that you can refer to for your own AI projects.

We will learn to pose a ML problem, break down a complex ML project into pieces, choose a loss and a training strategy.

I. Day ’n’ Night classification
II. Face verification
III. Neural style transfer (Art generation)
IV. Trigger-word detection
Case study 1: Day ’n’ Night classification

**Goal**: Given an image, classify as taken “during the day” (0) or “during the night” (1)

1. **Data?** 10,000 images
   
   **Split? Bias?**

2. **Input?**
   
   **Resolution?** (64, 64, 3)

3. **Output?**
   
   y = 0 or y = 1

   **Last Activation?** sigmoid

4. **Architecture ?** A shallow CNN should do the job pretty well

5. **Loss?**
   
   \[ L = -[y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})] \]

   **Easy warm up**
Summary of learnings: Day ’n’ Night classification

• Use a known proxy project to evaluate how much data you need.

• Be scrappy. For example, if you’d like to find a good resolution of images to use for your data, but don’t have time for a large scale experiment, approximate human-level performance by testing your friends as classifiers.
Case study 2: 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
   - Bertrand

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

3. **Output?**
   - $y = 1$ (it’s you)
   - or
   - $y = 0$ (it’s not you)
Case study 2: Face Verification

**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
- compute distance pixel per pixel
- if less than threshold then y=1
- input image

Issues:

- Background lighting differences
- A person can wear make-up, grow a beard...
- ID photo can be outdated
Case study 2: Face Verification

**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.
**Case study 2: Face Verification**

**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
- different encoding

So let’s generate triplets:

- anchor
- positive
- negative

minimize encoding distance

maximize encoding distance

What we really want:

- similar encoding
- different encoding

So let’s generate triplets:

- anchor
- positive
- negative

Which loss should you minimize?

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

Which one should you minimize?

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

Case study 2: Face Verification

Input


Model = Architecture + Parameters

Output

Loss

\[ L = \| \text{Enc}(A) - \text{Enc}(P) \|_2^2 - \| \text{Enc}(A) - \text{Enc}(N) \|_2^2 + \alpha \]
Case study 2b: Face Identification and Face Clustering

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?
Summary of learnings: Face Verification

• In face verification, we have used an encoder network to learn a lower dimensional representation (called “encoding”) for a set of data by training the network to focus on non-noisy signals.

• Triplet loss is a loss function where an (anchor) input is compared to a positive input and a negative input. The distance from the anchor input to the positive input is minimized, whereas the distance from the anchor input to the negative input is maximized.

• You learned the difference between face verification, face identification and face clustering.
Case study 3: Art Generation

**Goal**: Given a picture, make it look beautiful

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

2. **Input?**
   content image
   style image

3. **Output?**
   generated image
Case study 3: Art Generation

4. Architecture? We use a pre-trained model because it extracts important information from images.

Leon A. Gatys, Alexander S. Ecker, Matthias Bethge: A Neural Algorithm of Artistic Style, 2015
Case study 3: Art Generation

Image generation process

$$Content_c = \begin{pmatrix} 0.29 \\ 0.31 \\ \vdots \\ 0.44 \end{pmatrix}$$

$$Style_s = \begin{pmatrix} 0.43 \\ 0.39 \\ \vdots \\ 0.53 \end{pmatrix}$$

$$Gram Matrix = \begin{pmatrix} 0.22 & 0.99 & \vdots & 0.43 \\ 0.99 & 0.12 & \vdots & 0.43 \\ \vdots & \vdots & \ddots & \vdots \\ 0.43 & 0.43 & \cdots & 0.92 \end{pmatrix}$$

Deep Network (retrained on ImageNet)

compute loss

After 2000 iterations

update pixels using gradients

$$\frac{\partial L}{\partial x}$$
Case study 3: Art Generation

Which loss should we minimize?

\[ L = \| \text{Content}_c - \text{Content}_g \|_2^2 \]
\[ - \| \text{Style}_s - \text{Style}_g \|_2^2 \]

\[ L = \| \text{Style}_s - \text{Style}_g \|_2^2 \]
\[ + \| \text{Content}_c - \text{Content}_g \|_2^2 \]

\[ L = \| \text{Style}_s - \text{Style}_g \|_2^2 \]
\[ - \| \text{Content}_c - \text{Content}_g \|_2^2 \]

A  B  C

Kian Katanforoosh
Case study 3: Art Generation

Image generation process

\[
\begin{align*}
\text{Content}_C &= \begin{pmatrix} 0.29 \\ 0.31 \\ \vdots \\ 0.44 \end{pmatrix} \\
\text{Style}_S &= \begin{pmatrix} 0.12 \\ 0.10 \\ \vdots \\ 0.92 \end{pmatrix}
\end{align*}
\]

Deep Network (pretrained on ImageNet)

After 2000 iterations

update pixels using gradients \( \frac{\partial L}{\partial x} \)

Compute loss

where

\[
L = \left\| \text{Content}_C - \text{Content}_G \right\|^2_2 + \left\| \text{Style}_S - \text{Style}_G \right\|^2_2
\]
Summary of learnings: Art Generation

- In the neural style transfer algorithm proposed by Gatys et al., you **optimize image pixels rather than model parameters**. Model parameters are pretrained and non-trainable.

- You leverage the “knowledge” of a pretrained model to extract the **content** of a content image and the **style** of a style image.

- The loss proposed by Gatys et al. aims to minimize the distances between the **content** of the generated and content images, and the **style** of the generated and style images.
Case study 4: Trigger word detection

**Goal**: Given a 10sec audio speech, detect the word “activate”.

1. **Data?**
   - A bunch of 10s audio clips

2. **Input?**
   - $x = A \text{ 10sec audio clip}$

3. **Output?**
   - $y = 0$ or $y = 1$
Let’s have an experiment!

\[ y = 1 \]
\[ y = 0 \]
\[ y = 1 \]
Case study 4: Trigger word detection

Goal: Given a 10sec audio speech, detect the word “activate”.

1. Data? A bunch of 10s audio clips

2. Input?
   - x = A 10sec audio clip
   - y = 0 or y = 1

3. Output?
   - y = 00..0000100000..000
   - y = 00..00001..10000..000

4. Architecture? Sounds like it should be a RNN

5. Loss?
   \[ L = -(y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})) \] (sequential)
Case study 4: Trigger word detection

What is critical to the success of this project?

1. Strategic data collection/labeling process

   Positive word  Negative word  Background noise

   000000..00001..10000..000

   Automated labelling

   + Error analysis

2. Architecture search & Hyperparameter tuning

   Fourier transform

   LSTM LSTM LSTM ... LSTM LSTM LSTM

   σ σ σ ... σ σ σ

   000000..00001..10000..000

   Never give up

   CONV + BN

   GRU + BN GRU + BN ... GRU + BN GRU + BN

   σ σ ... σ σ

   0..0001..100..0

   000000..00001..10000..000

Kian Katanforoosh
Summary of learnings: Trigger word detection

• Your **data collection strategy** is critical to the success of your project. (If applicable) Don’t hesitate to get out of the building.

• You can gain insights on your labelling strategy by using a **human experiment**.

• **Refer to expert advice** to earn time and be guided towards a good direction.
Featured among “the most beautiful loss functions of 2015”

\[
\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{i,j}^{\text{obj}} \left[ (x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]
\]

\[
+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{i,j}^{\text{obj}} \left[ (\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2 \right]
\]

\[
+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{i,j}^{\text{obj}} (C_i - \hat{C}_i)^2
\]

\[
+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{i,j}^{\text{noobj}} (C_i - \hat{C}_i)^2
\]

\[
+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2
\]
For next Tuesday, 9.45am PT:

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

Others:
• TA project mentorship (mandatory): Meet with a TA to discuss your proposal.
• Friday TA section
• Fill-in AWS Form to get GPU credits for your projects