CNNs: Face Verification

Recitation 5

Yizhuo Zhang, Bharat Gaind

HW2 Bootcamp

- link

What we will cover today

- Problem Statement
- Model Architectures
- Metrics during training & inference
- Code demo

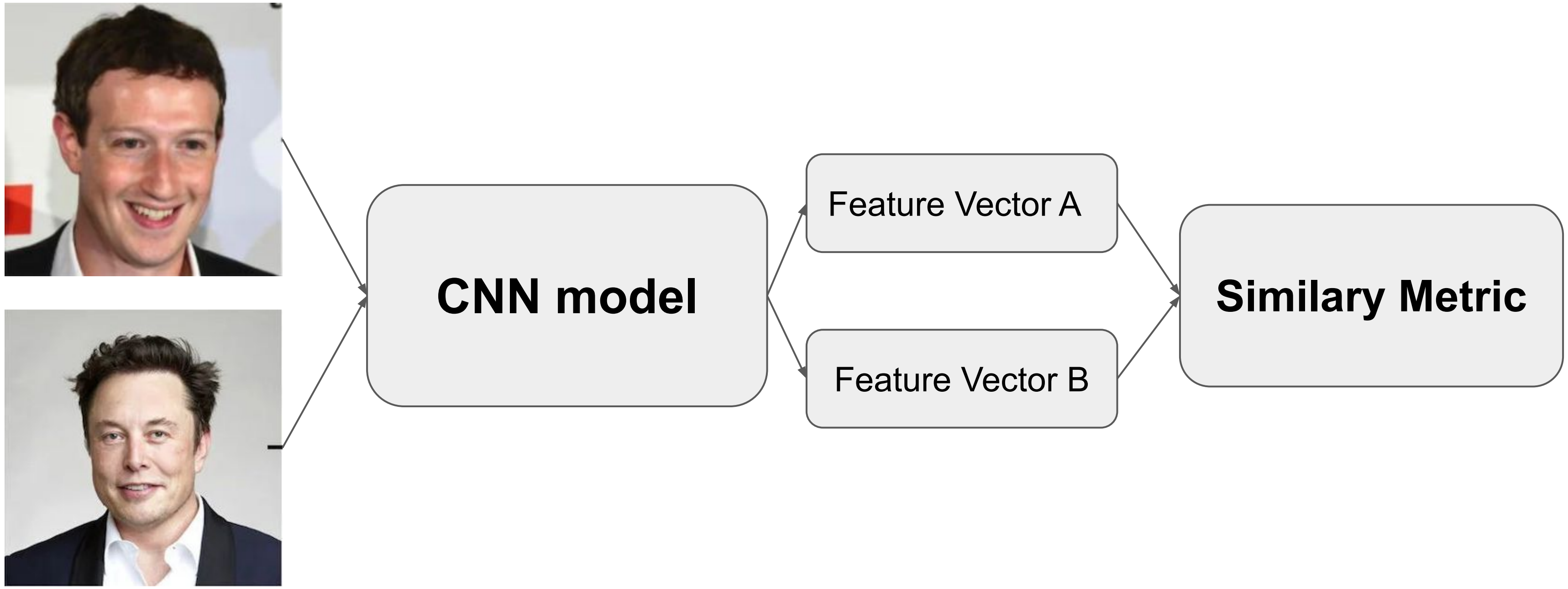
Problem Statement

Face Verification:

- Given a pair of face images, determine if they are of the same person

For each face image, you need to:

- Use a CNN to extract its features(embeddings)
- Compute a similarity score between the 2 feature vectors


CNN model
Feature Vector A
Feature Vector B
Similary Metric

Problem Statement

Use a CNN to extract its features(embeddings)

- What is the architecture of this CNN?
- How do we train the model? What is the loss function?

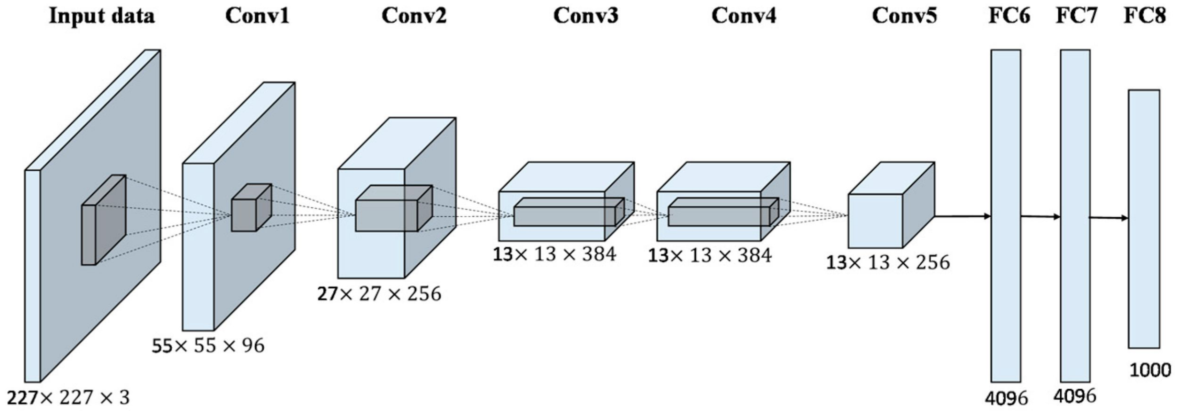
Compute a similarity score between the 2 feature vectors

- What is the metric of the similarity score?

Model Architectures: AlexNet

Published in 2012 and was the winner of the Imagenet LSVRC-2012

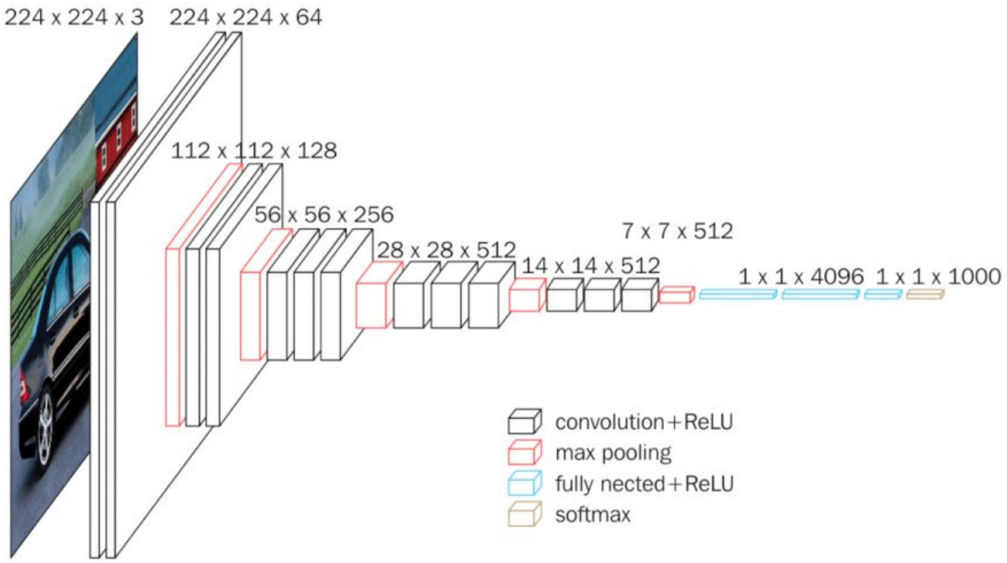
- Uses ReLU as it makes training faster
- Implements dropout between layers
- Uses data augmentation methods (Random Crop, Random Flip in PyTorch)
- The network is trained using SGD with momentum



Model Architectures: VGG Net (Recommended)

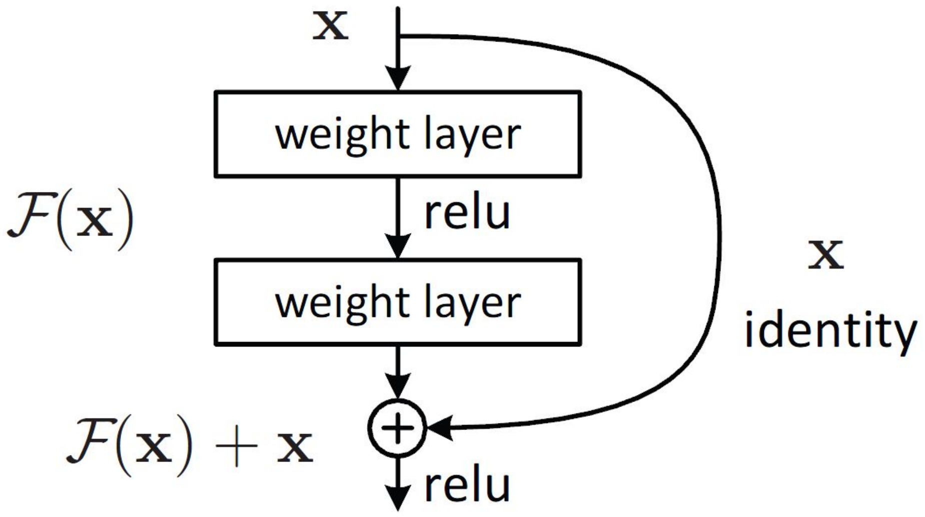
VGG (2014) architecture reduces the size of each layer yet increases the overall depth of it. reinforces the idea that CNNs must be deep in order to work well on visual data.

- VGG 16 ->
- Conv filters 3 * 3
- Max pooling 2 * 2



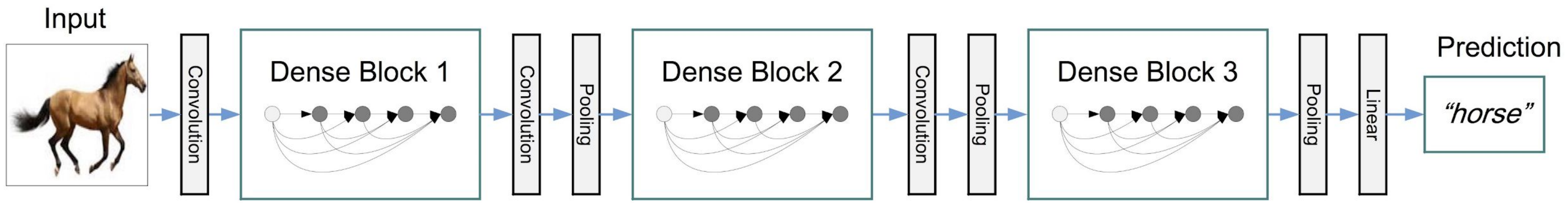
Model Architectures: ResNet (Recommended)

- Introduced in 2015, utilizes bottleneck architectures efficiently and learns them as residual functions
- Easier to optimize and can gain accuracy from increased depth due to skip connections



Model Architectures: Dense Nets

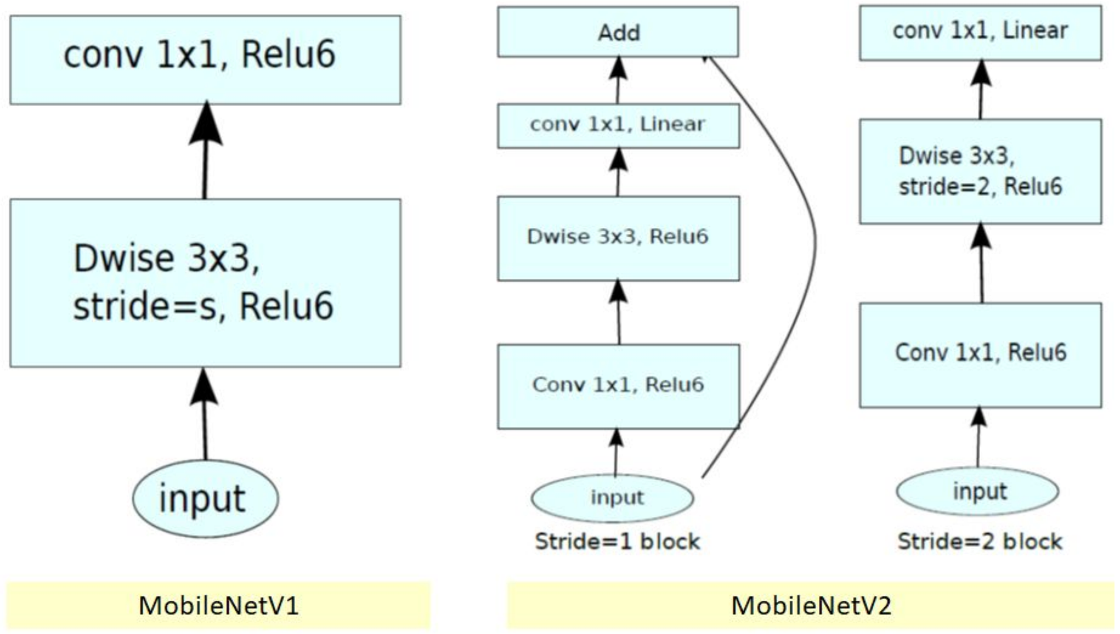
- It includes a stack of dense blocks followed by a transition layers
- Strengthen feature propagation, encourage feature reuse and decrease the number of parameters



Model Architectures: MobileNet and MobileNet V2

- Introduced in 2017, intended to run neural networks efficiently on mobile devices

- V1: Depth-wise separable convolution
- V2: Inverted residual block



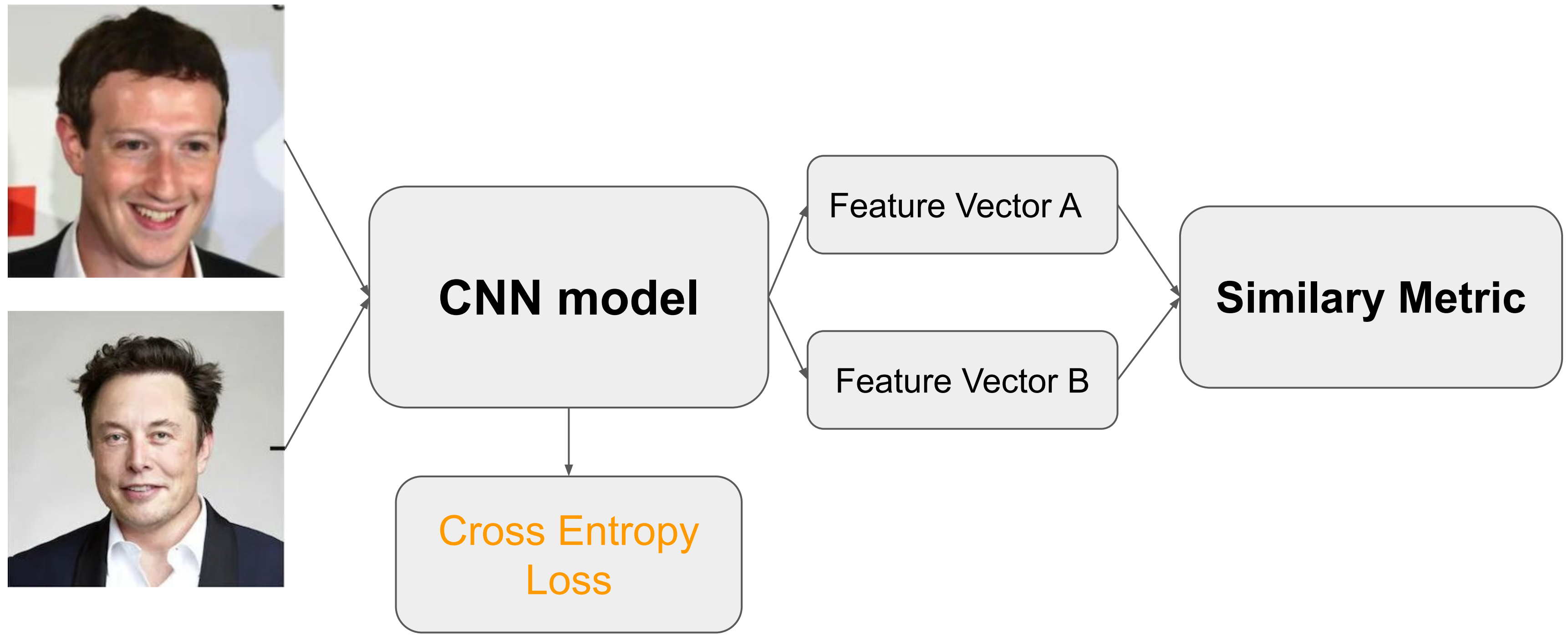
Metric for Training

How do we train the model to give us a good feature vector?

- **Train with classification** (recommended to start with)
- Metric learning

Classification:

- Cross entropy loss, same as hw1p2, recommended to start with
- Other losses in the writeup


CNN model
Feature Vector A
Feature Vector B
Similary Metric
Cross Entropy Loss

Metric for Verification

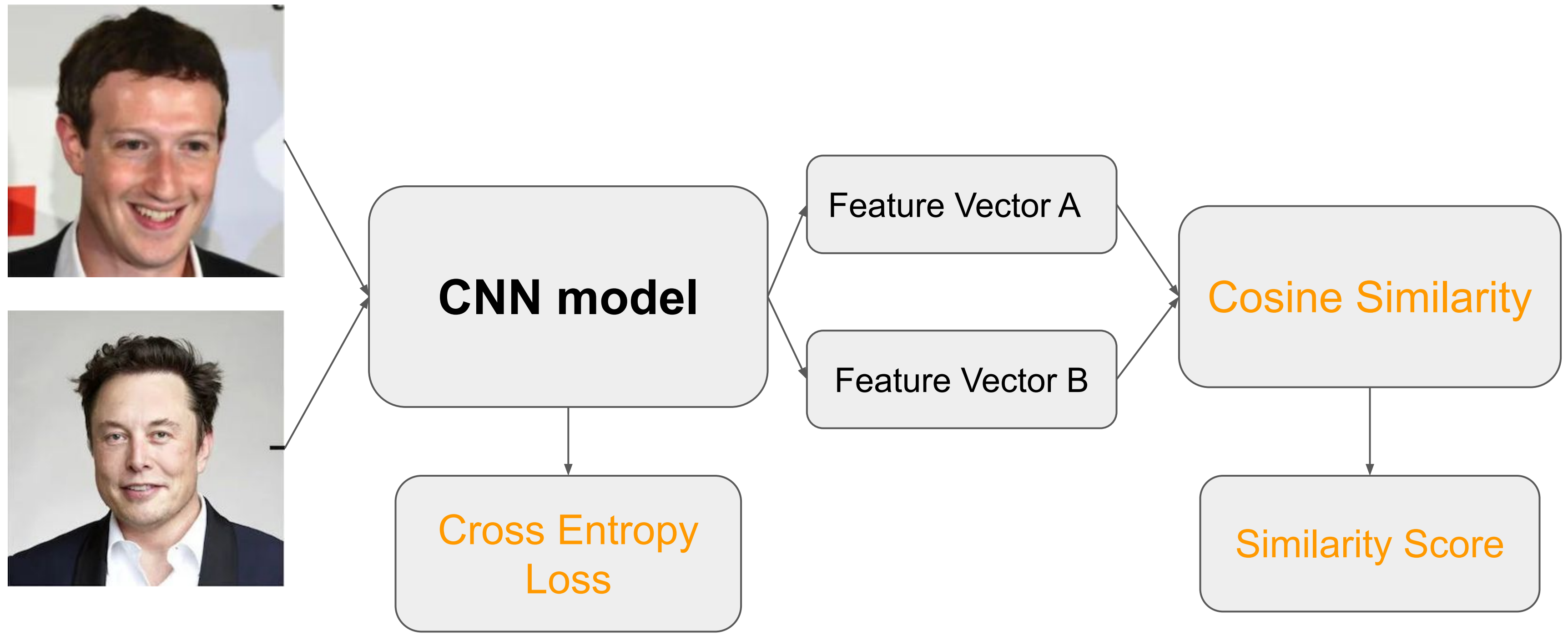
With two feature vectors, how do we compute the similarity score?

Cosine Similarity

- Wiki
- PyTorch implementation

Q:

- How does it compare to L2 similarity?
- In other applications, we compare the similarity score against a threshold to make a binary decision, but in hw2p2's kaggle submission, do you have to do that?


CNN model
Feature Vector A
Feature Vector B
Cosine Similarity
Cross Entropy Loss
Similarity Score