Approaching Face Classification & Verification

Hira Dhamyal
Hengrui Liu
Sarveshwaran Dhanasekar

11-785: Introduction to Deep Learning // Recitation 6 // February 22, 2019

Today’s Agenda

- Problem Statement
  - Verification vs Classification
  - Open set vs closed set
  - Transfer learning
- Data Explanation
  - Stats on the given Data
  - Torch Dataset (Notebook)
  - Torch DataLoader (Notebook)
- Model Architecture
  - Experimental process
  - 2D-CNN
  - Alex Net, VGG Net
  - ResNet, DenseNet
  - Shuffle Net and Mobile Net
- Learning objective
  - Area Under the Curve (AUC)
  - Transfer learning
  - Contrastive loss
  - Margin problem
  - Center loss
  - Triplet loss
  - Softmax (revisited)
  - Angular softmax

Problem Statement

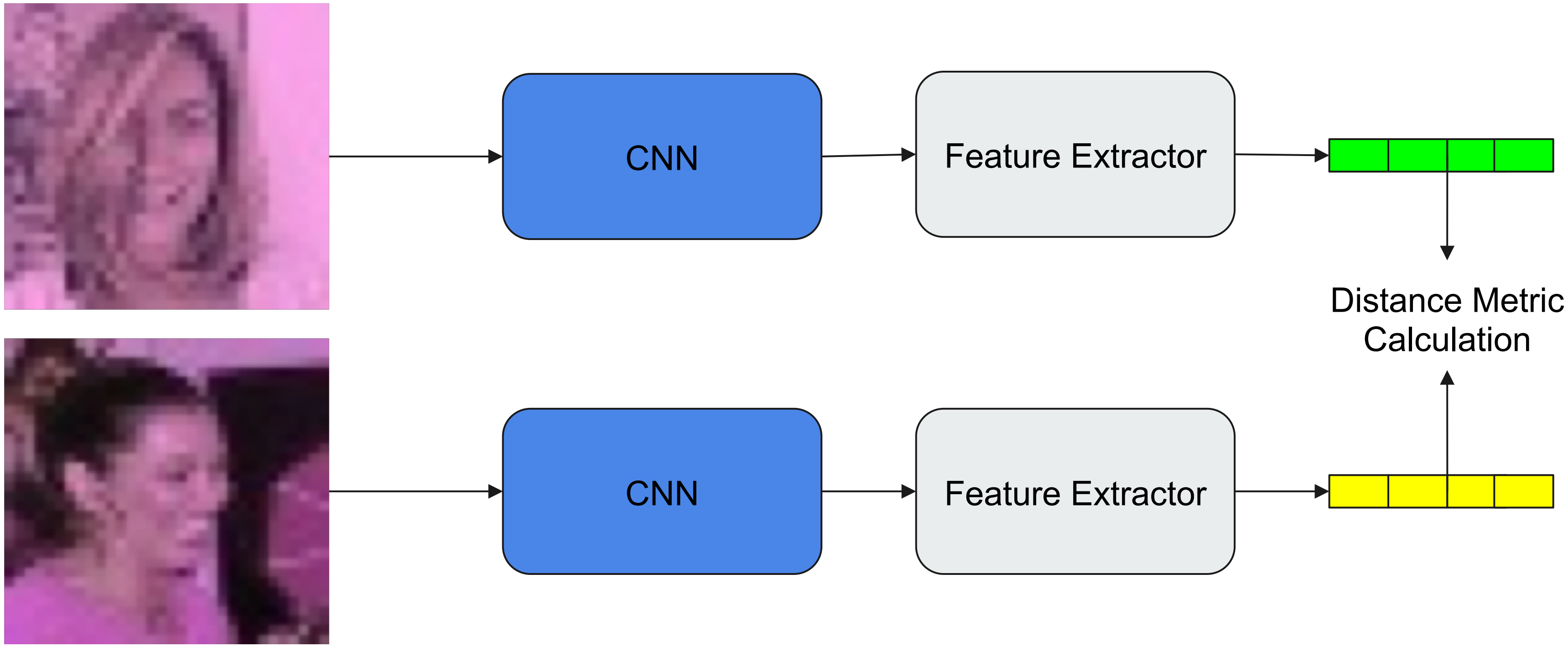
Face Classification

- Classifying the person id using the image of a person’s face.

Face Verification

- Use transfer learning to design a system for Face Verification
- The images contain a lot of facial features that vary across different people
- The main task is to train a CNN to extract and represent such important features from an image.
- The extracted features will be represented in a fixed-length vector of features, known as a face embedding
- Given a pair of face embeddings, we need to identify if they belong to the same person

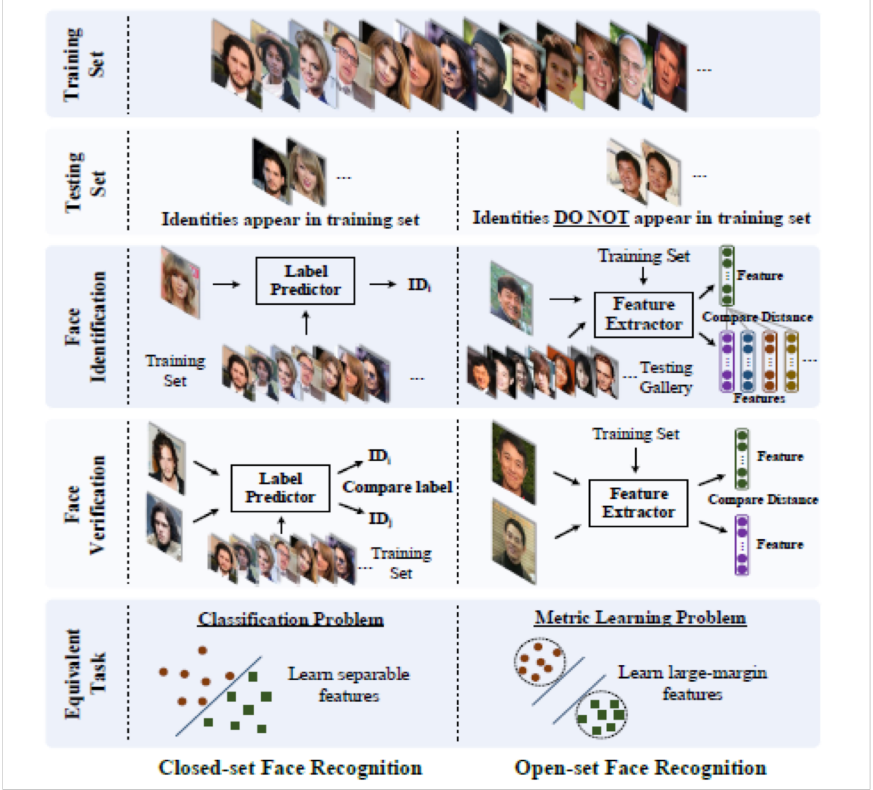
Face verification



Open Set vs Closed Set

- Closed set
    - Treated as a classification problem
    - Training dataset is assumed as exhaustive
    - All possible data classes are present in the training set
    - Identification is among previously defined classes
    - Face Classification is Closed set
- Open Set
    - Learn features instead of data classes
    - Training dataset is not exhaustive
    - Used as a metric learning problem
    - Face Verification is Open Set

Open Set vs Closed Set



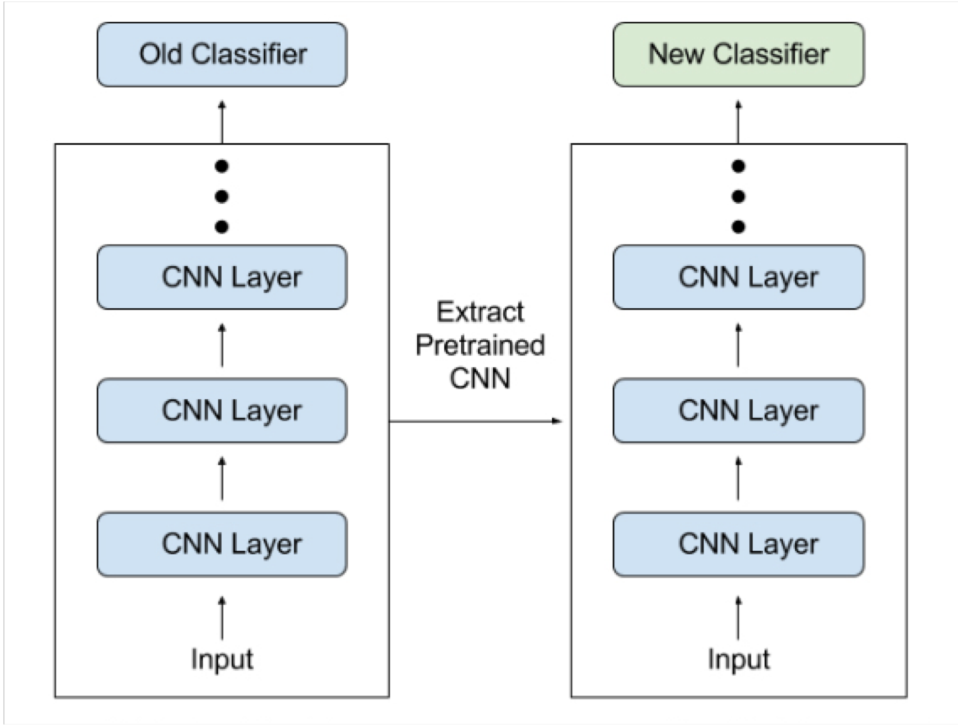
Classification versus Verification & Open versus Closed set for the facial recognition problem.

Classification vs Verification

- Classification
    - N-way classification task. Predict from a closed set of labels
- Verification
    - It’s a 1-to-n matching where given a sample, you match it to the closest sample among n other samples
    - Can also be performed as a 1-to-1 task where we verify if they belong to the same class of data or not

Transfer Learning

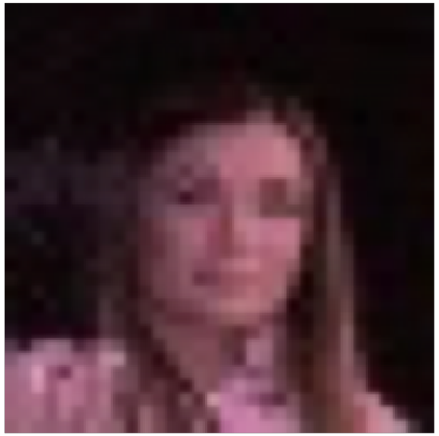
- Transfer learning is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task
- Develop Model Approach
  - Select a source task
  - Develop source model
  - Reuse model
  - Tune model
- Pre-trained Model Approach
  - Select source model (usually pre-trained)
  - Reuse model
  - Tune model

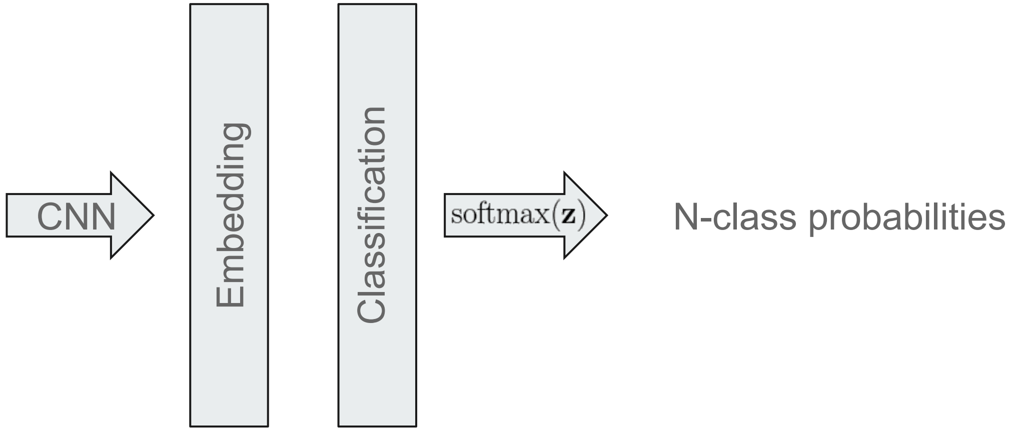


For the Homework...

- Transfer Learning to be used for the task of face verification
- We follow Develop Model Approach
- Train a convolutional network to perform N-way face classification
- Extract intermediate representations of face classifier
- Verify if two intermediate samples belong to same person
- Fine-tune the network to learn more discriminative features

Face classification


Input Image



Fully-connected layers

- N-way classification where N is the number of people present in the training set
  - Cross-Entropy objective
  - Applied per sample

Data Explanation -- Classification

Training	● Medium (2300 Face ids) ● Large (2000 Face ids)
Validation	● Medium ● Large
Testing	● Medium

Data Explanation -- Verification

Training	(Same as the classification)
Validation	● validation_verification ● validation_trials_verification.txt 100000 total trials True label given {0,1}
Testing	● Test_verification ● Test_trials_verification_student.txt 900000 total trials

Format of trial file

The trial file contains 2 columns.

1st column containing a pair of file

2nd is the label. 0 meaning they the pair is images of different people. 1 meaning otherwise

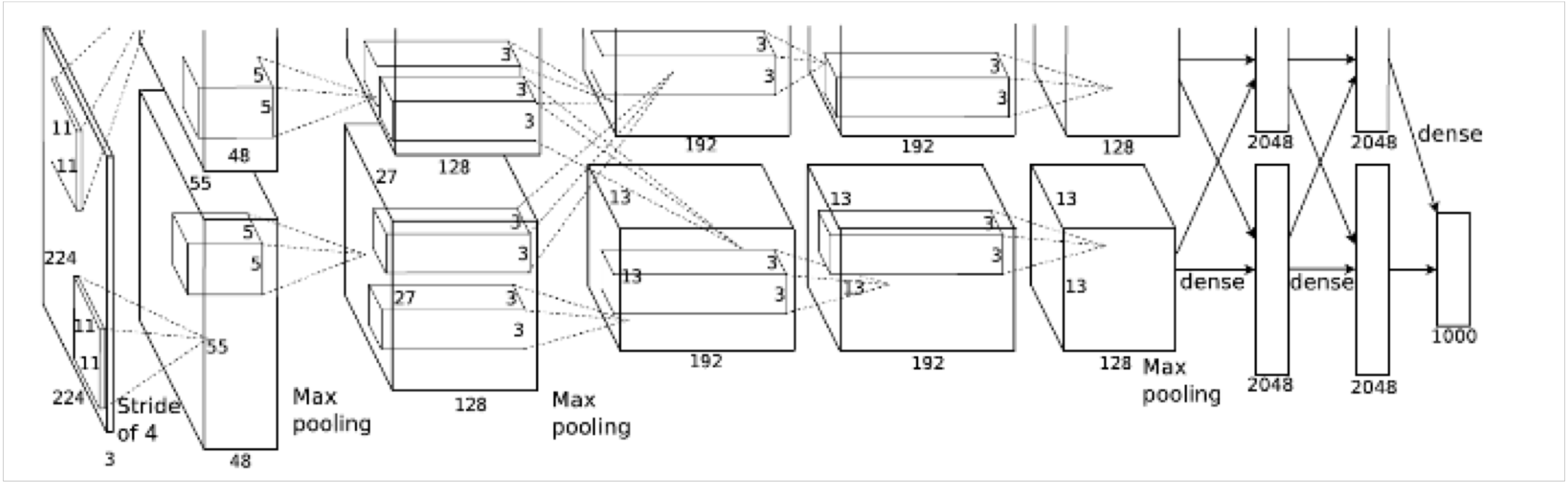
File name 1	File name 2	Label
fid_373/0111_01.jpg	fid_423/0170_01.jpg	0

CNNs in PyTorch

- Conv2d
  - Set number of channels to 3 (RGB)
  - While average pooling, average features across both dimension (H, W) to get a 2D vector
- Tapering
  - Memory Optimization Technique: Layers near the input can have higher filter sizes and larger strides
  - Reduce filter sizes and strides as the network goes deeper

CNN Architectures: AlexNet https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

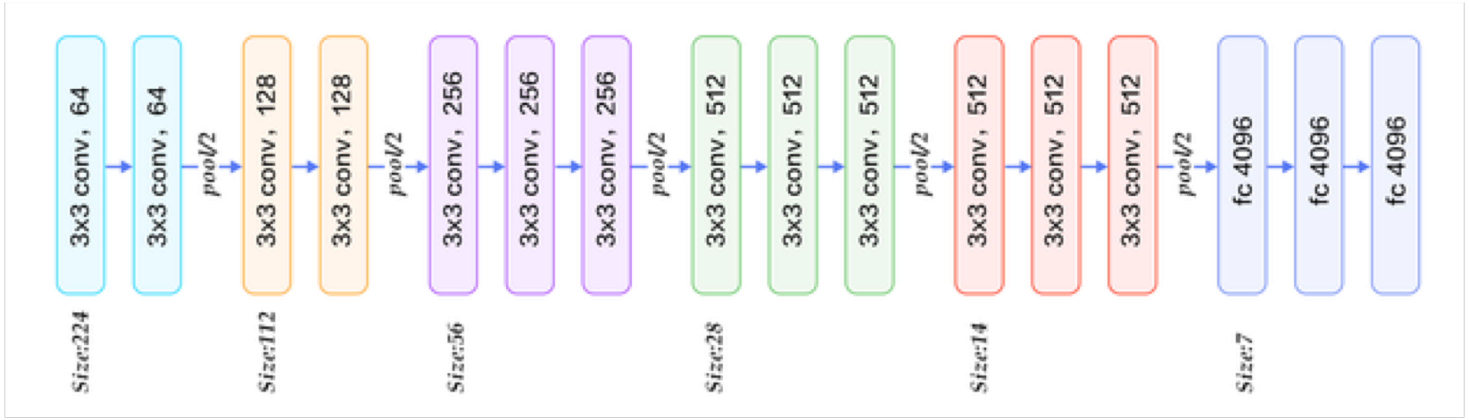
- Won the 2012 ImageNet LSVRC-2012
- Used ReLU, dropout



VGGNet

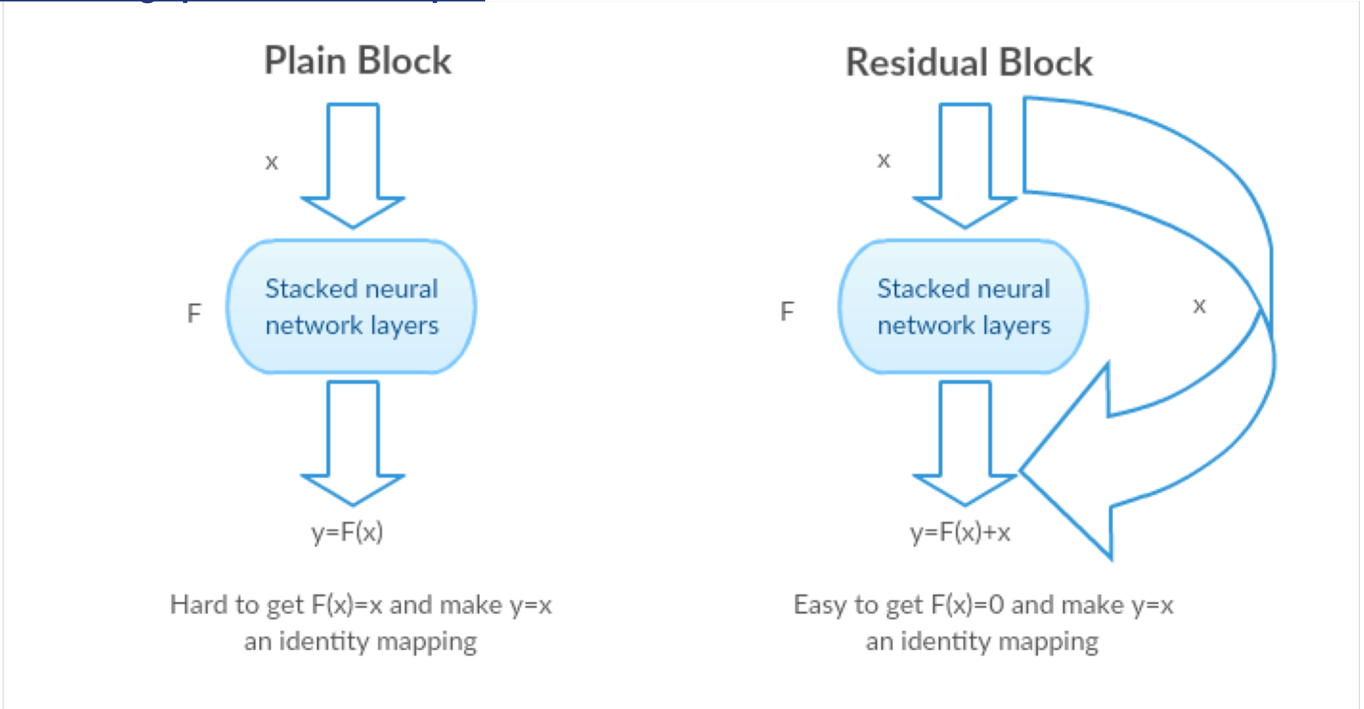
https://arxiv.org/pdf/1409.1556.pdf

- Convolutions layers (used only 3*3 size)
- Max pooling layers (used only 2*2 size)
- Fully connected layers at end
- Total 16 layers



ResNets

https://arxiv.org/pdf/1512.03385.pdf

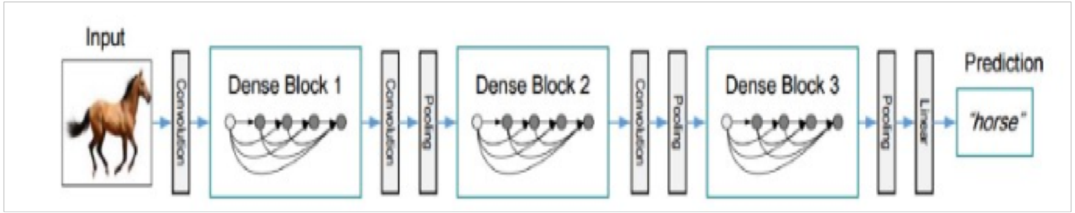

Plain Block
x
F
Stacked neural network layers
y=F(x)
Hard to get F(x)=x and make y=x an identity mapping
Residual Block
x
F
Stacked neural network layers
x
y=F(x)+x
Easy to get F(x)=0 and make y=x an identity mapping

Identity mapping in Residual blocks

DenseNets

https://arxiv.org/pdf/1608.06993.pdf

- Extension of ResNets
- Strengthen feature propagation, encourage feature reuse
- Fewer parameters than a ResNet



Other Newer Architectures to look into

- Mobile Net (https://arxiv.org/pdf/1801.04381.pdf)
- Shuffle Net (https://arxiv.org/pdf/1707.01083.pdf)

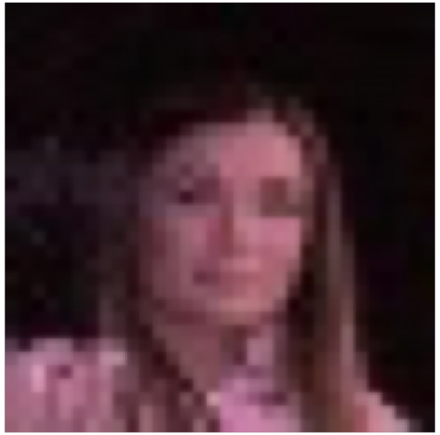
Recap

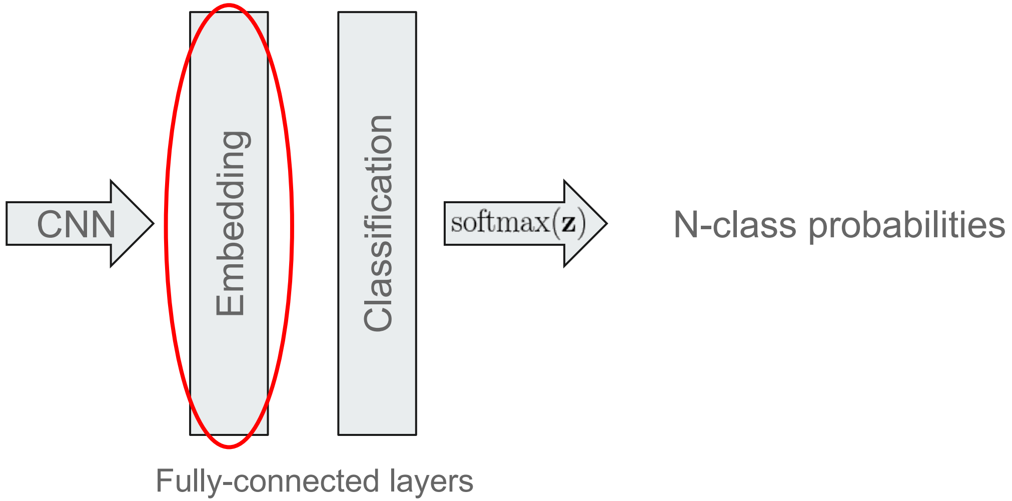
- ➔ The face verification problem
- ➔ Open set vs Closed set
- ➔ Transfer learning
- ➔ Classification vs verification
- ➔ Data stats and explanation
- ➔ CNNs for feature learning
- ➔ Architecture variations

What's Next

- ➔ Classification for verification (revisited)
- ➔ Area Under Curve (AUC)
- ➔ Explicitly optimizing for discriminative features
- ➔ Thinking about margins
- ➔ Triplet, center, contrastive losses
- ➔ Angular softmax

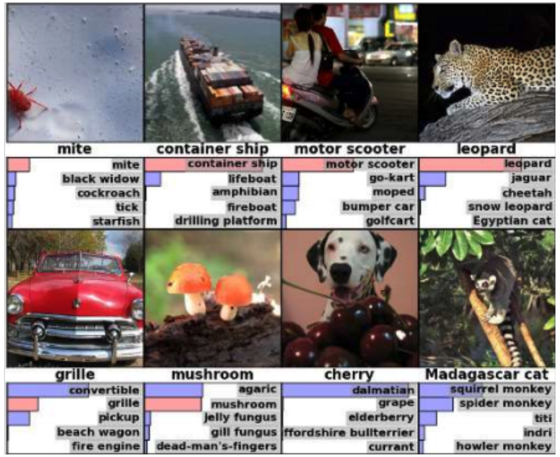
Face classification for verification


Input Image

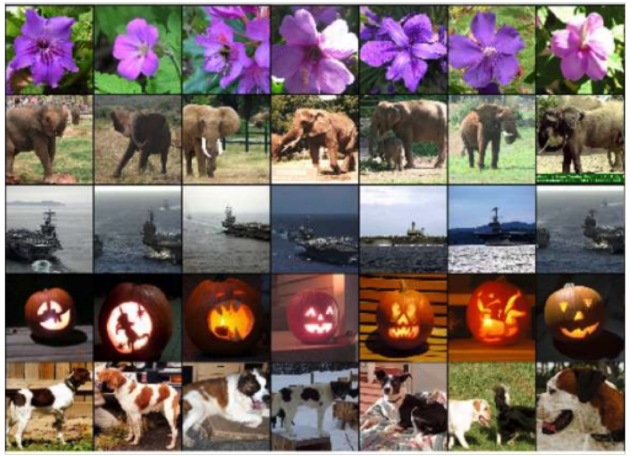


- Utilize the activations of the second to last fully-connected layer as a feature embedding.
- Estimate the similarity between two face images as the inverse distance between their embeddings.
- Consider the implications of different distance metrics (e.g. euclidean, cosine)

DNNs *can* learn meaningful features



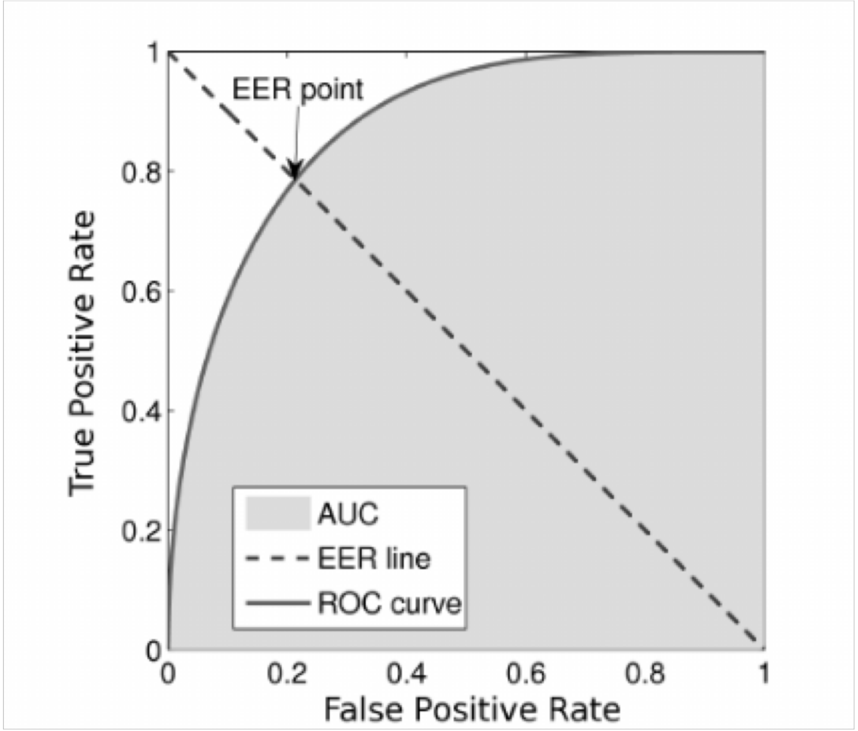
Eight ILSVRC-2010 test images and the five labels considered most probable by our model. The correct label is written under each image, and the probability assigned to the correct label is also shown with a red bar (if it happens to be in the top 5).



Five ILSVRC-2010 test images in the first column. The remaining columns show the six training images that produce feature vectors in the last hidden layer with the smallest Euclidean distance from the feature vector for the test image.

Area Under the Curve (AUC)

- The Receiver Operating Characteristic (ROC) curve is created by plotting the True Positive Rate (TPR) against the False Positive Rate (FPR) at various threshold settings
- The Area Under the Curve (AUC) for the ROC curve is equal to the probability that a classifier will rank a randomly chosen positive pair (images of same people) higher than a randomly chosen negative one (images from two different people) (assuming ’similar’ ranks higher than ’dissimilar’ in terms of similarity scores).



Source: https://www.webopedia.com/TERM/E/equal_error_rate.html

AUC

- AUC of 0.0 means that the model ranks all negatives above all positives. This is the worst model possible. In fact if you flip the scores, you will be better off.
- AUC between 0 and 0.5, means the model ranks a random positive example higher than a random negative example less than 50% of the time.
- AUC of 0.5, means it ranks a random positive example higher than a random negative example 50% of the time. It can be said that the predictive ability of the model is no better than random guessing.

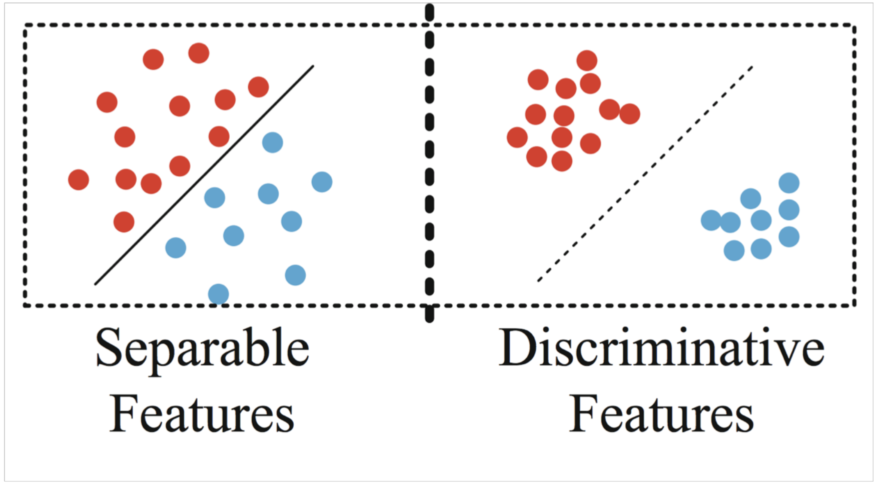
Aim for:

- **AUC between 0.5 and 1.0, means it ranks a random positive example higher than a random negative example more than 50% of the time.**
- **AUC of 1.0 means the model ranks all positives above all negatives.**

Cite: https://medium.com/datadriveninvestor/understanding-roc-auc-curve-7b706fb710cb

Discriminative features

- ➔ Classification task optimizes learning separable features
- ➔ Optimally we wish to learn discriminative features
  - ◆ **Maximal** *inter* class distance
  - ◆ **Minimal** *intra* class distance
- ➔ First define a measure of distance
- ➔ Most often consider Euclidean distance



Center loss

https://ydwen.github.io/papers/WenECCV16.pdf

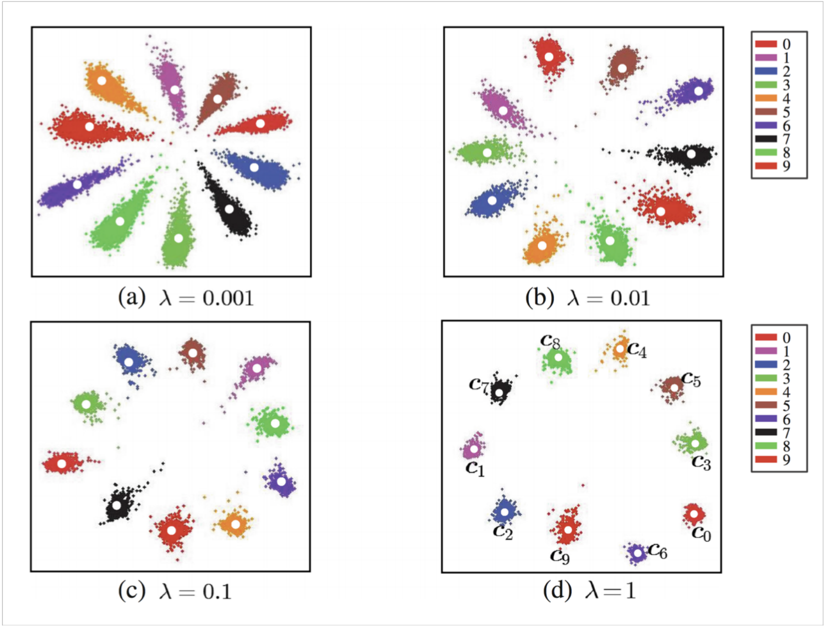
- ➔ Define a criterion that promotes minimal intra-class distance while maintaining inter-class separability
- ➔ Does **not** explicitly try to maximize inter-class class distance

Center Loss: $$\mathcal{L}_C = \frac{1}{2}\sum_{i=1}^{m}\|x_i - c_{y_i}\|_2^2$$

Joint Objective: $$\mathcal{L} = \mathcal{L}_S + \lambda\mathcal{L}_C$$

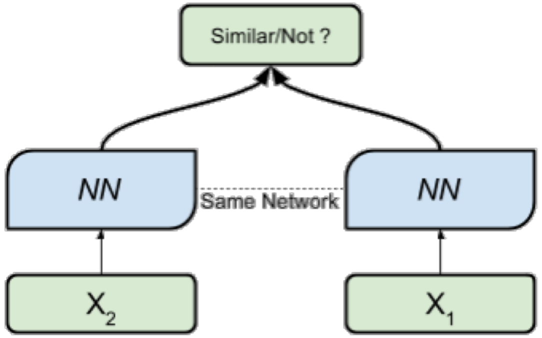
c_{y_i} feature center for class label

$\mathcal{L}_S$ softmax loss (softmax + xent)



Contrastive loss

- For every pair of training instances, try to push the features for intra-class samples together and maintain some margin *m* between inter-class ones.
- Feedforward two samples *i, j* and apply the contrastive loss function provided some distance metric of choice *D.*
- Generally only useful for fine tuning. Not an information rich criterion
  - 1 bit of information per sample pair
- Tricky to balance sample pairs and choose the margin

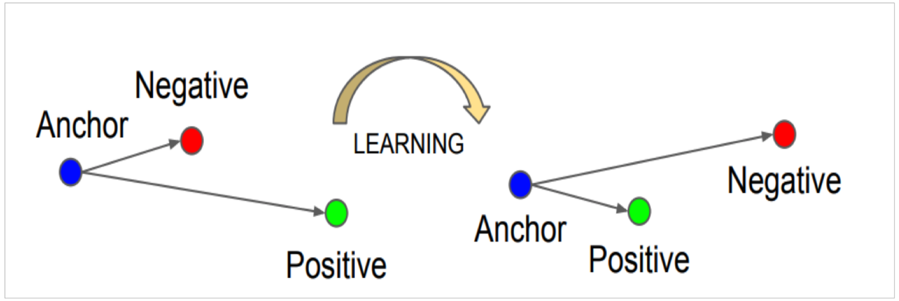


$$\mathcal{L}_{CT}(\mathbf{z}_i, \mathbf{z}_j) = 1(y_i = y_j)\left(\frac{1}{2}D(\mathbf{z}_i, \mathbf{z}_j)^2\right) + 1(y_i \neq y_j)\frac{1}{2}(\max(0, m - D(\mathbf{z}_i, \mathbf{z}_j)^2))$$

y_k class label of sample *k* $\mathbf{z}_k$ embedding of sample *k* m margin

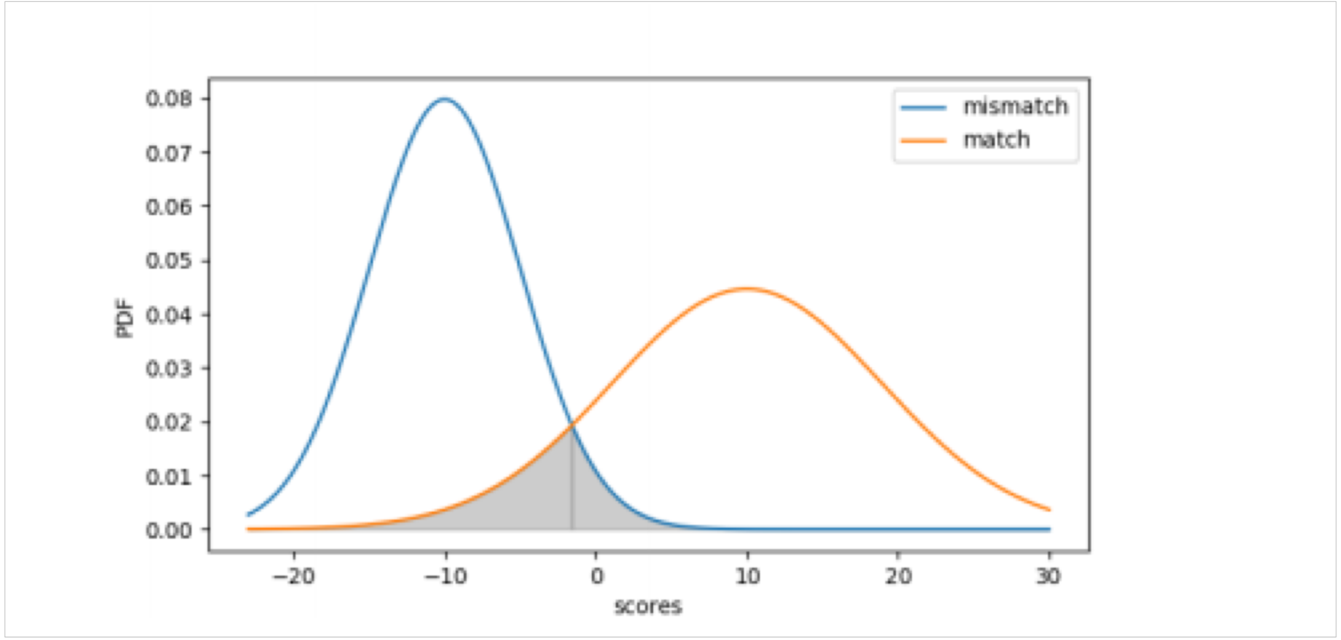
Triplet loss

- ➔ Motivated by nearest neighbor classification
- ➔ Try to ensure that the face embedding of a face *i* is closer to all the embeddings of that same face than it is to any other embedding that belongs to a face *i != j*
- ➔ Share parameters among 3 networks (similar to contrastive loss scenario)
- ➔ Good results when sampling triplets in an intelligent manner (sample mining)



Pair-Wise Loss

For verification task, this function explicitly tries to separate the distributions of similarity scores under similiar pairs and dis similar pairs.



Softmax loss (revisited)

$$p_1 = \frac{\exp(\mathbf{W}_1^T x + b_1)}{\exp(\mathbf{W}_1^T x + b_1) + \exp(\mathbf{W}_2^T x + b_2)}$$

$$p_2 = \frac{\exp(\mathbf{W}_2^T x + b_2)}{\exp(\mathbf{W}_1^T x + b_1) + \exp(\mathbf{W}_2^T x + b_2)}$$

$$(W_1 - W_2)x + b_1 - b_2 = 0$$

Resulting decision boundary (binary case)

➔ Define softmax *loss* as a combination of the final layer parameters, the softmax operator and cross-entropy loss.

➔ For this example, assume two classes (binary classification).

Angular softmax loss

➔ Idea: Combine Euclidean margin constraints with the softmax loss

➔ Formulate an angular interpretation to softmax that can be projected onto a hypersheric manifold

➔ No need for sample mining when identifying pairs / triplets for contrastive / triplet losses

➔ Interpretable as learning features that are discriminative

Angular softmax loss

Recall binary classification with softmax decision boundary:

$$(W_1 - W_2)x + b_1 - b_2 = 0$$

Angular softmax loss

Recall binary classification with softmax decision boundary:

$$(W_1 - W_2)x + b_1 - b_2 = 0$$

Note that,

$$W_i^T x_i + b_i = ||W_i^T||\ ||x_i|| \cos(\theta_i)$$

Where θ_i is the angle between the weight and feature vector.

Angular softmax loss

Recall binary classification with softmax decision boundary:

$$(W_1 - W_2)x + b_1 - b_2 = 0$$

Note that,

$$W_i^T x_i + b_i = ||W_i^T|| \, ||x_i|| \cos(\theta_i)$$

If $(||W_i|| = 1, b_i = 0)$, we can express the decision boundary equivalently as,

$$\cos(\theta_1) - \cos(\theta_2) = 0$$

Angular softmax loss

Recall binary classification with softmax decision boundary:

$$(W_1 - W_2)x + b_1 - b_2 = 0$$

Note that,

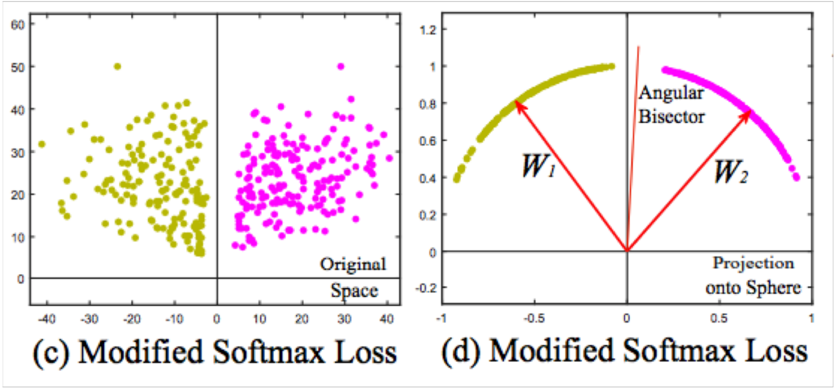
$$W_i^T x_i + b_i = ||W_i^T||\ ||x_i|| \cos(\theta_i)$$

If $(||W_i|| = 1, b_i = 0)$, we can express the decision boundary equivalently as,

$$\cos(\theta_1) - \cos(\theta_2) = 0$$

This is simply the angle bisector of the two parameter vectors

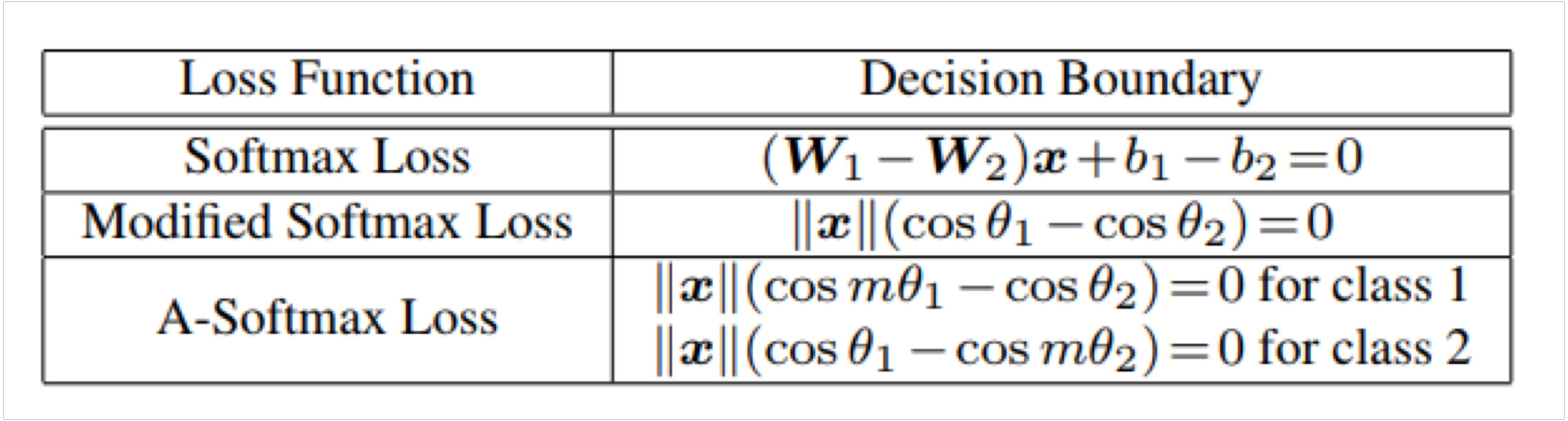
Angular softmax loss



Generalize to multi-class:

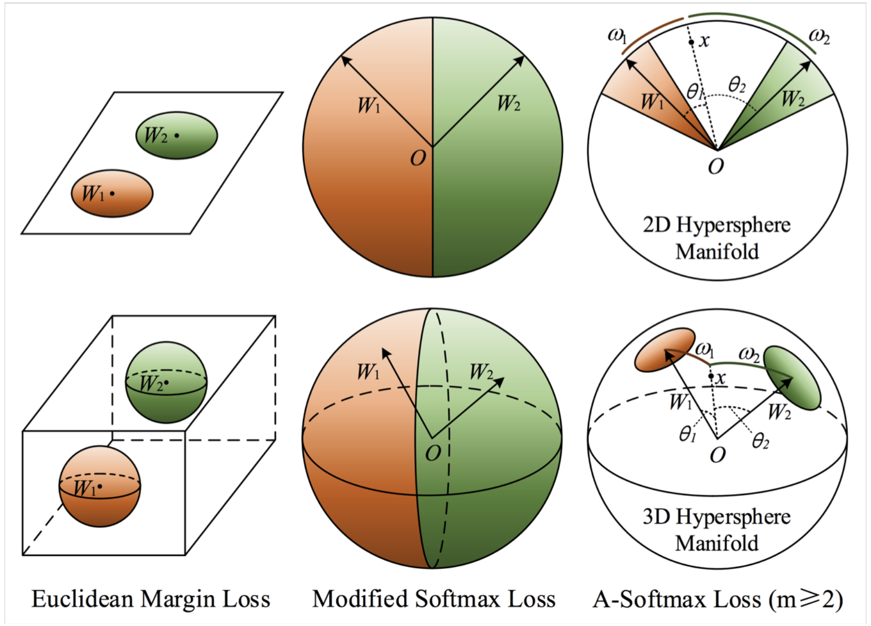
$$\theta_{j,i}\ (0 \leq \theta_{j,i} \leq \pi),\ \text{for}\ W_j, x_i$$

Angular softmax loss



Loss Function	Decision Boundary
Softmax Loss	$(\boldsymbol{W}_1 - \boldsymbol{W}_2)\boldsymbol{x} + b_1 - b_2 = 0$
Modified Softmax Loss	$\|\boldsymbol{x}\|(\cos\theta_1 - \cos\theta_2) = 0$
A-Softmax Loss	$\|\boldsymbol{x}\|(\cos m\theta_1 - \cos\theta_2) = 0$ for class 1 $\|\boldsymbol{x}\|(\cos\theta_1 - \cos m\theta_2) = 0$ for class 2

Angular softmax loss



➔ Nice geometric interpretation when weights are normalized and biases set to zero (in the last fully-connected layer)

Sources

- https://arxiv.org/pdf/1512.03385.pdf
- https://medium.com/@14prakash/understanding-and-implementing-architectures-of-resnet-and-resnext-for-state-of-the-art-image-cf51669e1624
- https://towardsdatascience.com/densenet-2810936aeebb
- https://arxiv.org/pdf/1608.06993v3.pdf
- https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
- https://arxiv.org/pdf/1409.1556.pdf
- https://www.ece.rice.edu/~dhj/courses/elec241/spectrogram.html
- https://arxiv.org/pdf/1704.08063.pdf
- http://ydwen.github.io/papers/WenECCV16.pdf
- https://arxiv.org/pdf/1503.03832v3.pdf
- http://yann.lecun.com/exdb/publis/pdf/hadsell-chopra-lecun-06.pdf
- https://www.cs.cmu.edu/~rsalakhu/papers/oneshot1.pdf