Компьютерное зрение

Лекция 8.
Faces classification & verification. Metric Learning

25.06.2020

Руслан Рахимов

Постановка задачи

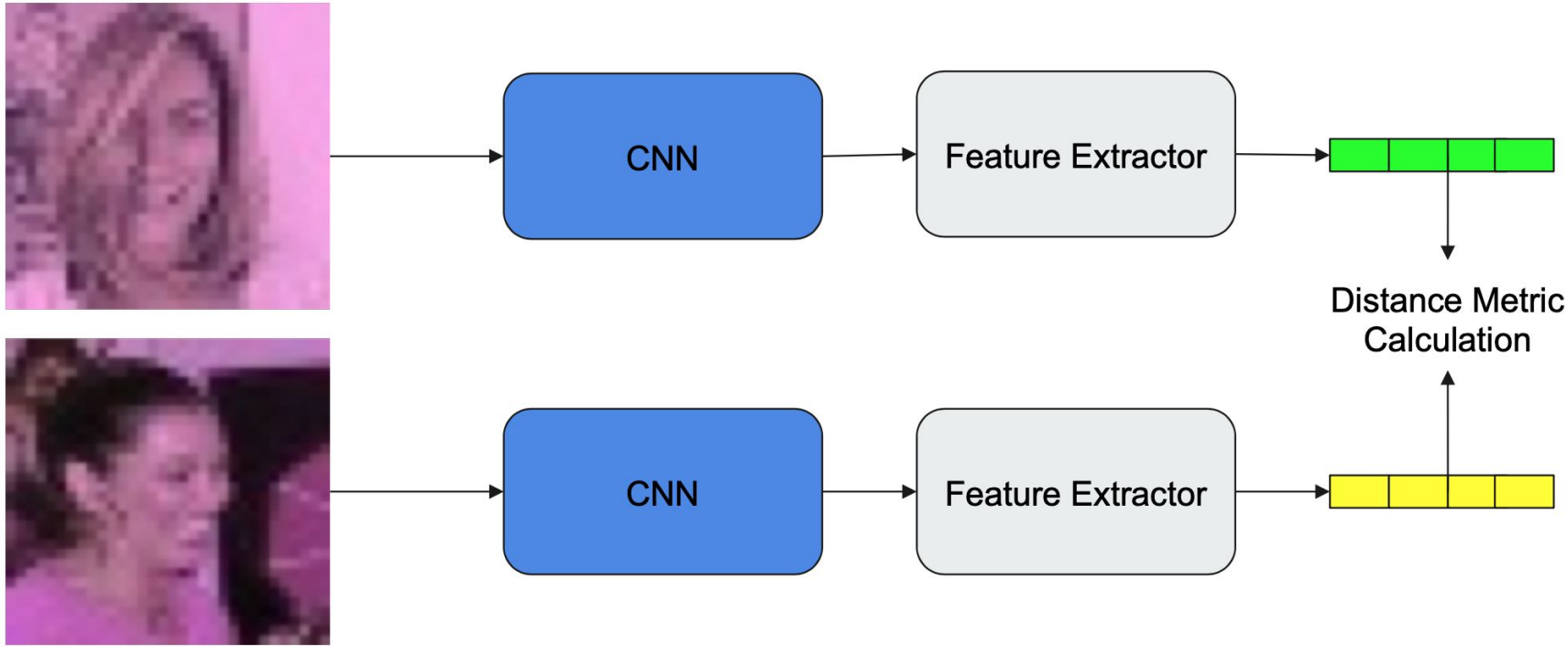
Face Classification

- Определить person_id имея базу лиц людей

Face Verification

- Используем **transfer learning** для создания системы для Face Verification
- Изображения содержат много **“facial features”**, которые отличаются от человека к человеку
- Основная задача - натренировать CNN извлекать хорошие признаки
- Извлеченные признаки должны представлять собой вектор фиксированный длины, известный также как **face embedding**
- Имея пару face эмбеддингов, нужно определить принадлежат ли они одному и тому же человеку

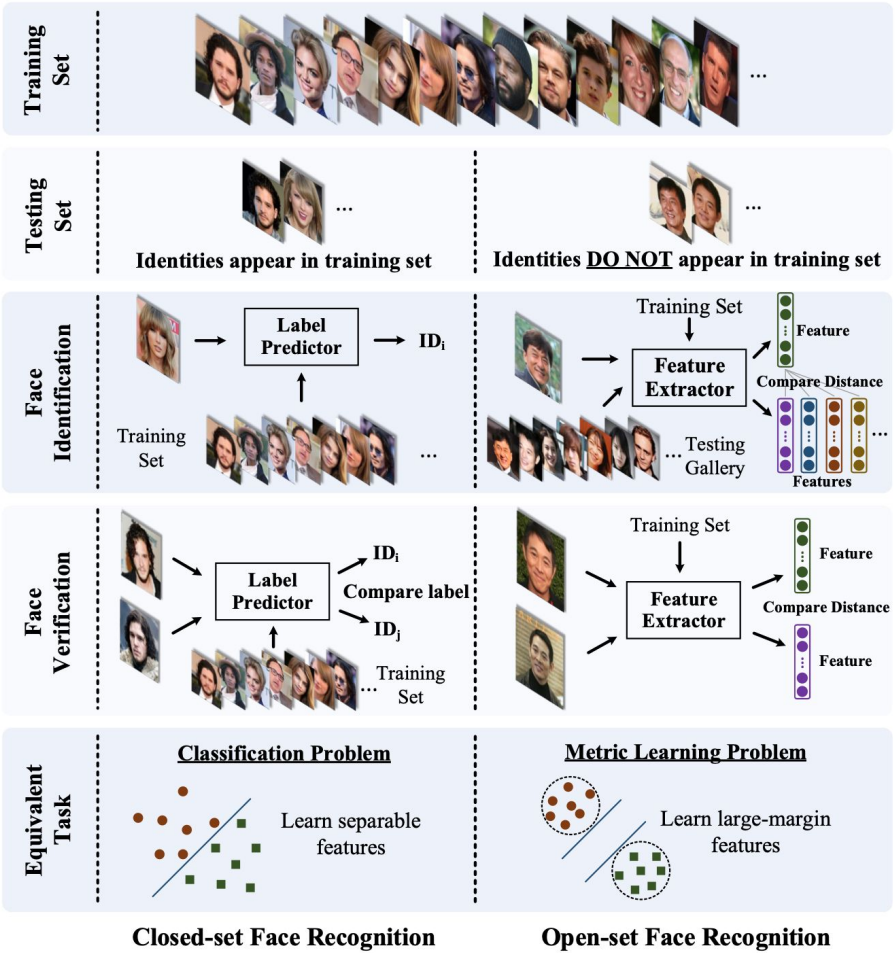
Face verification



Open Set vs Closed Set

- **Closed set**
  - Рассматривается как задача классификации (идентификации)
  - Обучающая выборка считается “полной”
  - Т.е. все классы представлены в ней
  - Идентификация происходит среди людей, представленный в трейне
  - **Face Classification** это **closed set** задача

- **Open Set**
  - Учим представление данных вместо классов
  - Обучающая выборка “неполная”
  - Задача определяется как metric learning problem
  - **Face Verification** это **open Set** задача

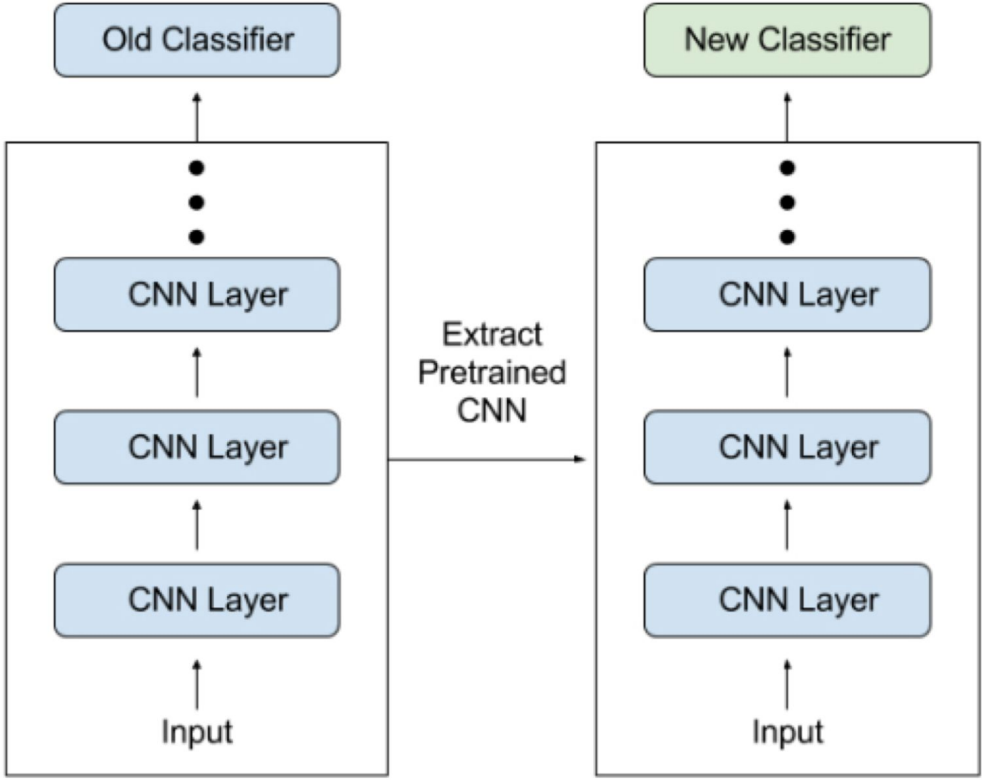

Training Set
Testing Set
Identities appear in training set
Identities DO NOT appear in training set
Face Identification
Label Predictor
ID_i
Training Set
Training Set
Feature Extractor
Feature
Compare Distance
Features
Testing Gallery
Face Verification
Label Predictor
ID_i
Compare label
ID_j
Training Set
Training Set
Feature Extractor
Feature
Compare Distance
Feature
Equivalent Task
Classification Problem
Learn separable features
Metric Learning Problem
Learn large-margin features
Closed-set Face Recognition
Open-set Face Recognition

Classification vs Verification

- **Classification**
  - Классификация на N классов. Предсказываем класс из ограниченного множества.

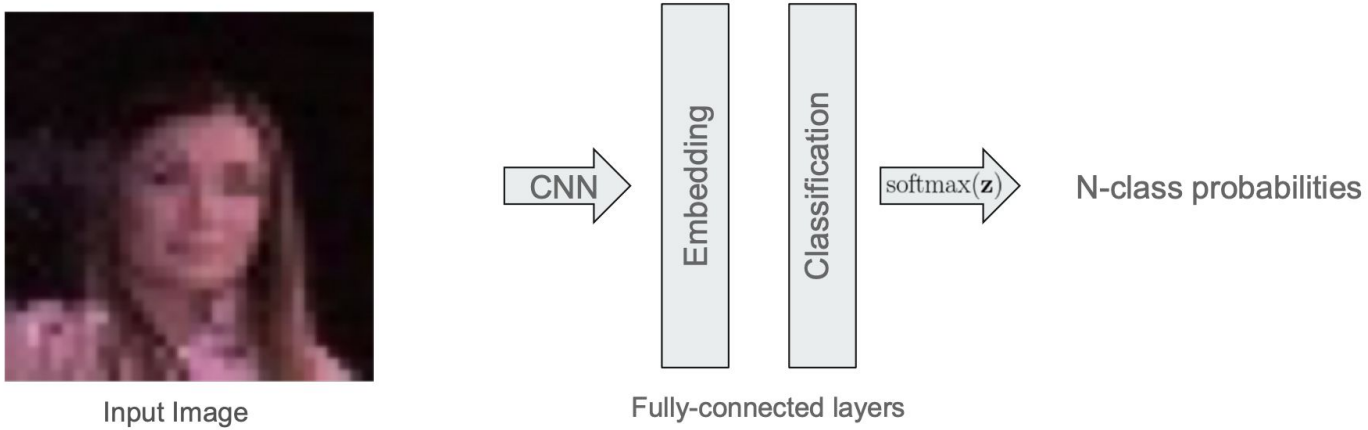
- **Verification**
  - Это 1-to-n матчинг, когда имея объект, нужно найти “ближайший” к нему среди n других объектов
  - Также можно представить в виде 1-to-1 задачи, когда нужно сказать принадлежат ли два объекта одному классу (например два изображения одной личности)

Transfer Learning



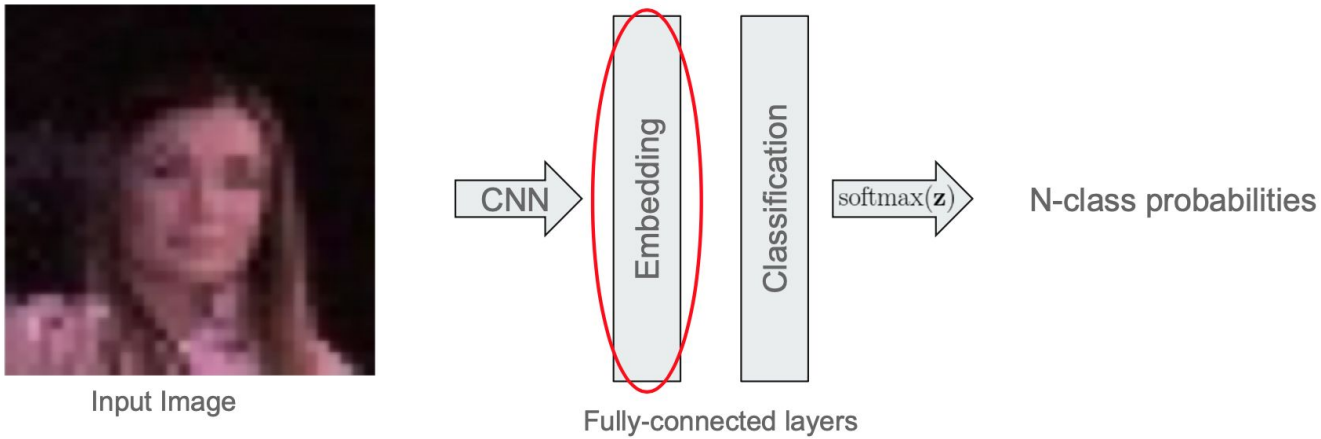
Face classification

Классификация на N классов, где N - число людей в тренировочной выборке

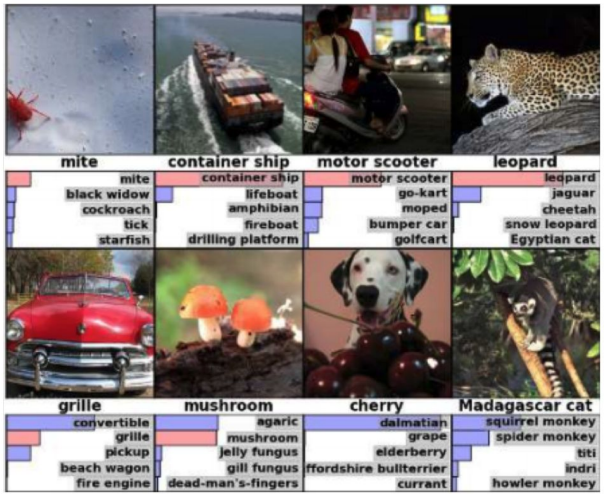


Face classification for verification

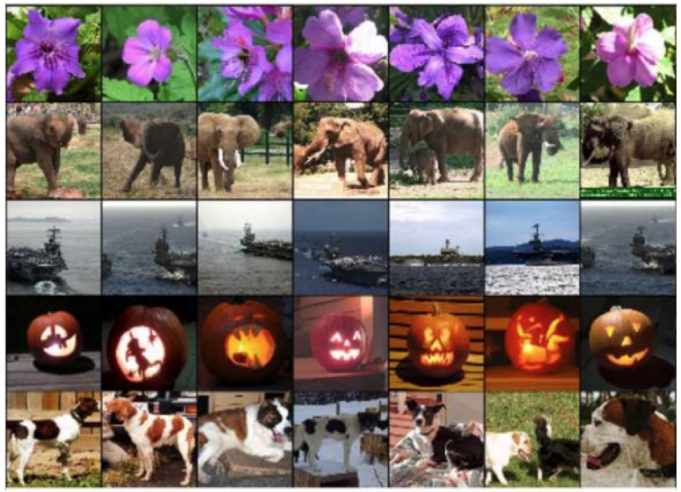
- Используем выходы предпоследнего слоя (перед последним Linear) как feature embedding
- Оцениваем похожесть двух лиц как функцию обратную от расстояния между эмбеддингами
- Можно рассматривать разные функции расстояния (н. евклидовое, косинусное)



Нейросети выучивает полезные признаки



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).

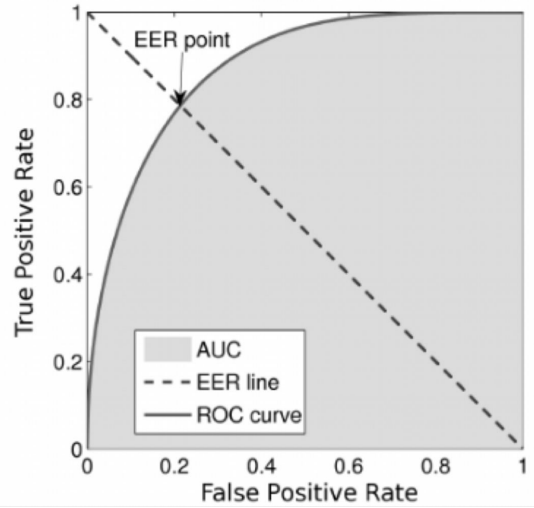


Five ILSVRC-2010 test images in the first column. The remaining columns show the six training images that produce feature vectors in the last hidden layer with the smallest Euclidean distance from the feature vector for the test image.

Area Under the Curve (AUC)

AUC кривая строится отображает True Positive Rate по оси y и False Positive Rate по оси x на различных значениях порога.

Площадь под AUC кривой равна вероятности что классификатор выше проранжирует случайно выбранную пару изображений одного человека чем случайно выбранную пару изображений двух разных людей.



AUC

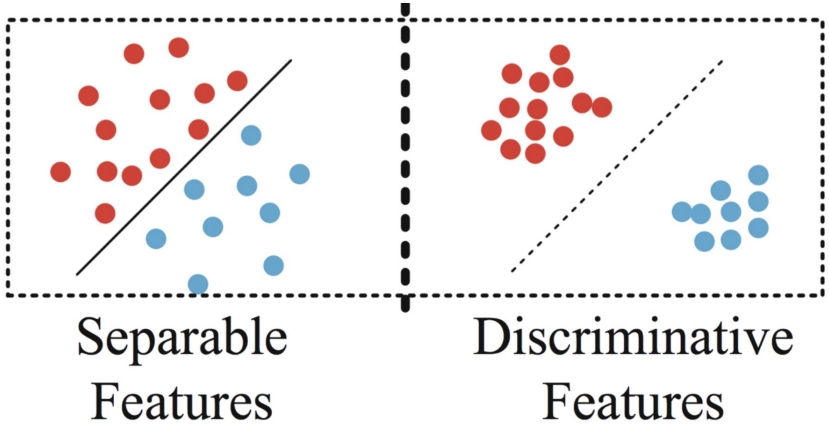
AUC = 0 значит что модель ранжирует негативные примеры выше положительных. Это худший случай.

AUC на уровне 0.5 соответствует случайному угадыванию.

Наша цель:

- AUC между 0.5 и 1
- AUC на уровне 1 означает, что модель ранжирует все положительные примеры выше отрицательных.

Discriminative features



- При классификации мы пытаемся найти separable features, такие признаки по которым можно разделить объекты на два или более классов

- В идеале мы пытаемся выучить discriminative features, т.е. признаки, которые имеют
  - Максимальное различие между объектам разных классов (inter class distance)
  - Минимальную различие между объектами одного класса (intra class distance)

- Для начала нужно определить меру расстояния (distance)
- Наиболее часто используемое - евклидово

Center loss

- Определить критерий, который минимизирует **intra-class distance**, сохраняя при этом разделимость классов (**inter-class separability**)
- Неявно максимизируем **inter-class distance**

Center Loss:

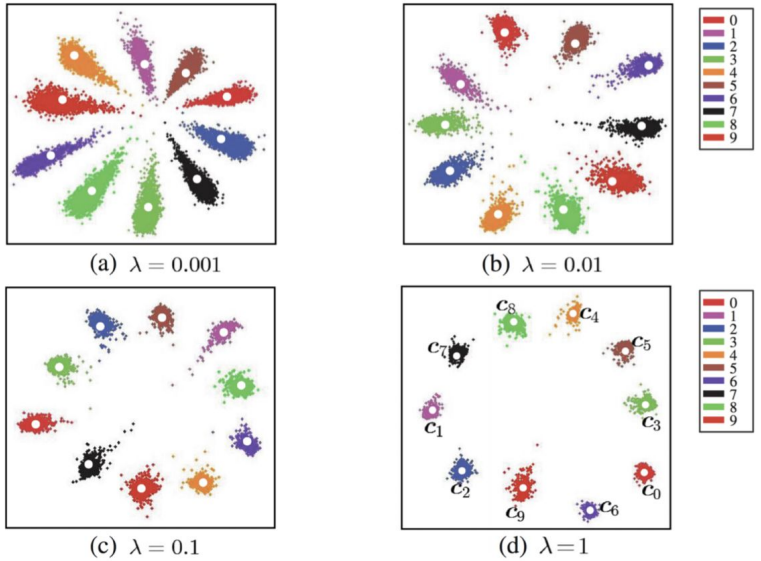
$$\mathcal{L}_c = \frac{1}{2}\sum_{i=1}^{m}\left\|x_i - c_{y_i}\right\|_2^2$$

Joint Objective:

$$\mathcal{L} = \mathcal{L}_s + \lambda\mathcal{L}_c$$

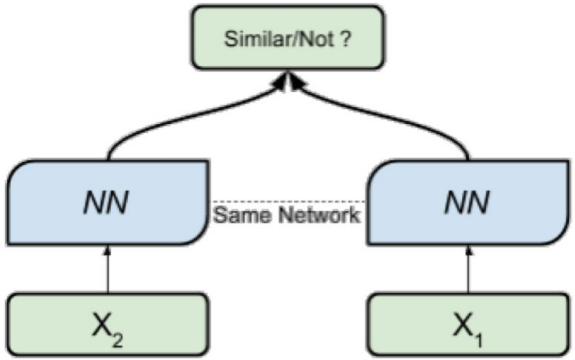
c_{y_i} feature center for class label

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



Contrastive loss

- Для каждой пары объектов из train выборки, пытаемся уменьшить intra-class distance, сохраняя определенный margin m между объектами из разных классов
- Подаем на вход два объекта i, j и применяем contrastive loss, выбрав distance metric D



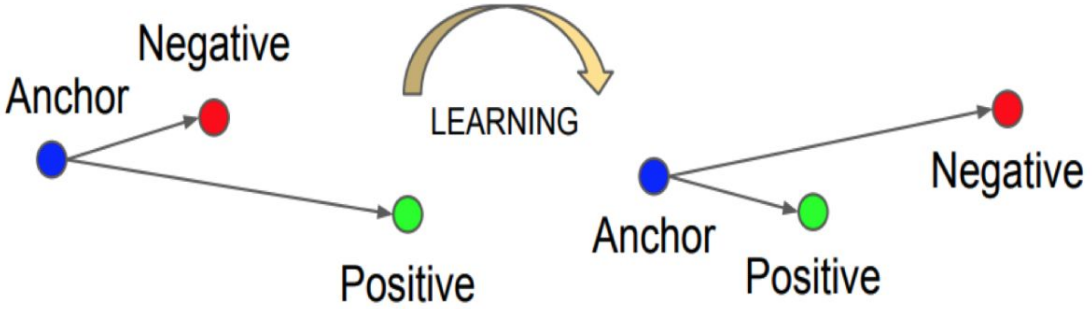
- Обычно хорошо подходит для файнтюнинга, а не как хороший информационный критерий.
  - 1 бит информации на 1 пару
- Сложно подбирать пары и выбирать margin

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

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

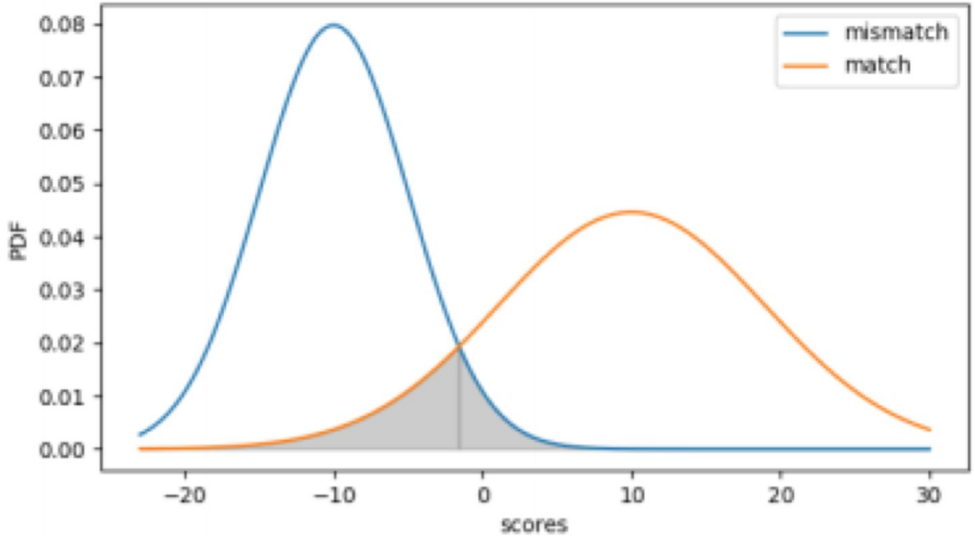
Triplet loss

- Мотивация: kNN алгоритм для классификации
- Имея лицо i и j, стараемся сделать эмбеддинг лица человека i ближе к другим эмбеддингам того же человека, чем к эмбеддингам другого
  человека j != i
- Хорошие результаты, когда семплим триплеты в умной манере (sample mining)



Pair-Wise loss

Для задачи верификации эта функция пытается разделить распределения похожести внутри похожих пар и не похожих пар



Softmax loss

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

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

- Определяем softmax loss, как комбинацию параметров последнего слоя, softmax оператора и cross-entropy loss
- Для примера рассмотрим два класса (бинарная классификация)

Softmax loss

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

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

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

- Определяем softmax loss, как комбинацию параметров последнего слоя, softmax оператора и cross-entropy loss
- Для примера рассмотрим два класса (бинарная классификация)

Angular softmax loss (SphereFace, 2017)

- Идея: объединить евклидову дистанцию и softmax
- Нет необходимости в sample mining при определении пар / триплетов для contrastive / triplet loss
- Можно интерпретировать, как поиск discriminative features

Angular softmax loss

Вспомним бинарную классификацию с softmax decision boundary

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

Angular softmax loss

Вспомним бинарную классификацию с softmax decision boundary

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

Заметим, что

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

Где θ_i угол между весами и признаками

Angular softmax loss

Вспомним бинарную классификацию с softmax decision boundary

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

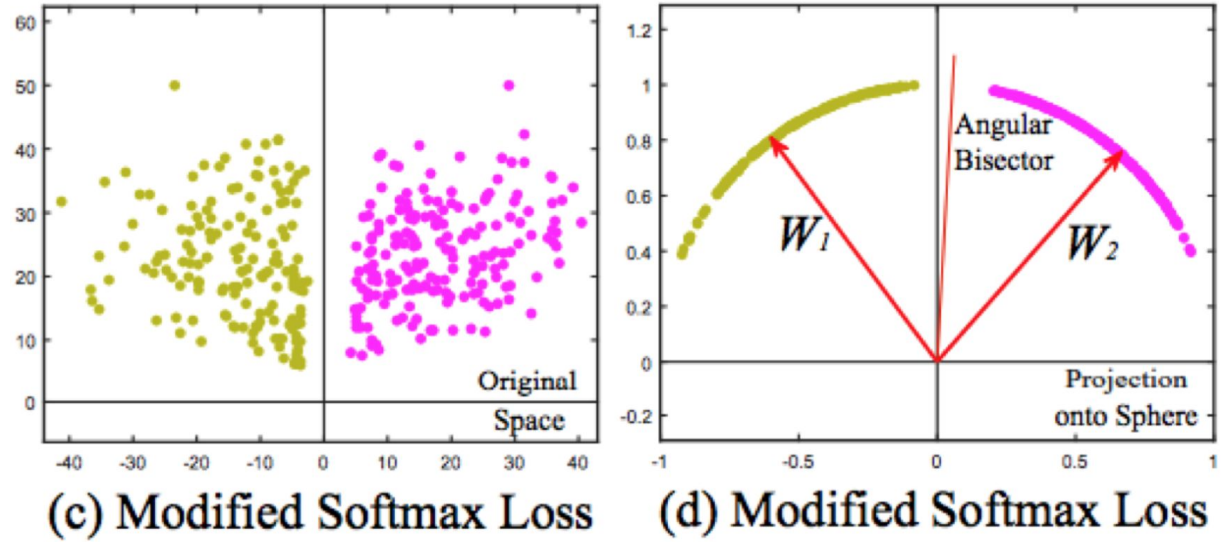
Заметим, что

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

Если $(||W_i|| = 1, b_i = 0)$ можно записать в следующем виде:

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

Angular softmax loss



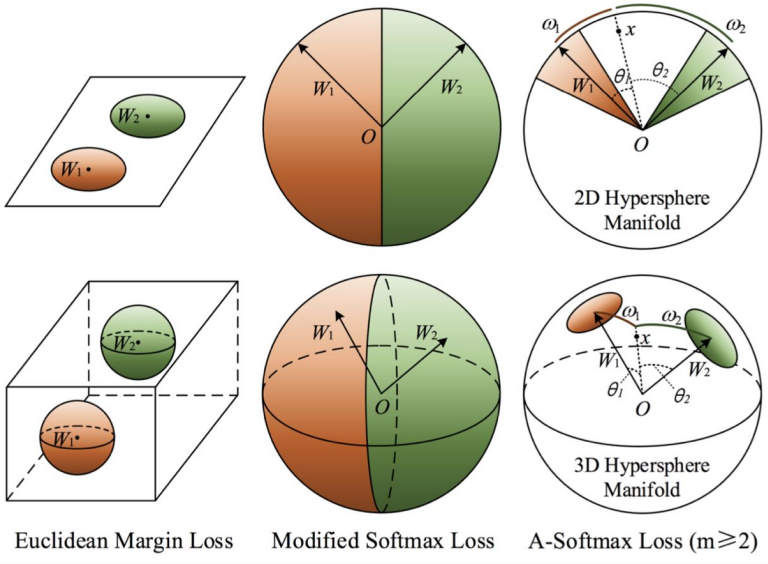
(c) Modified Softmax Loss (d) Modified Softmax Loss

Можно обобщить на multiclass

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



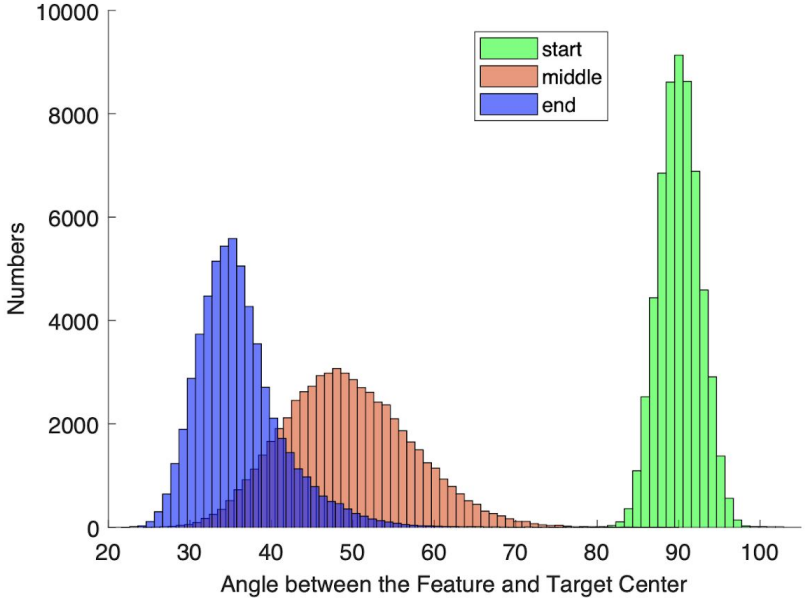
Хорошая геометрическая интерпретация, когда в последнем слое веса нормализованы и bias=0

Arcface (2018)

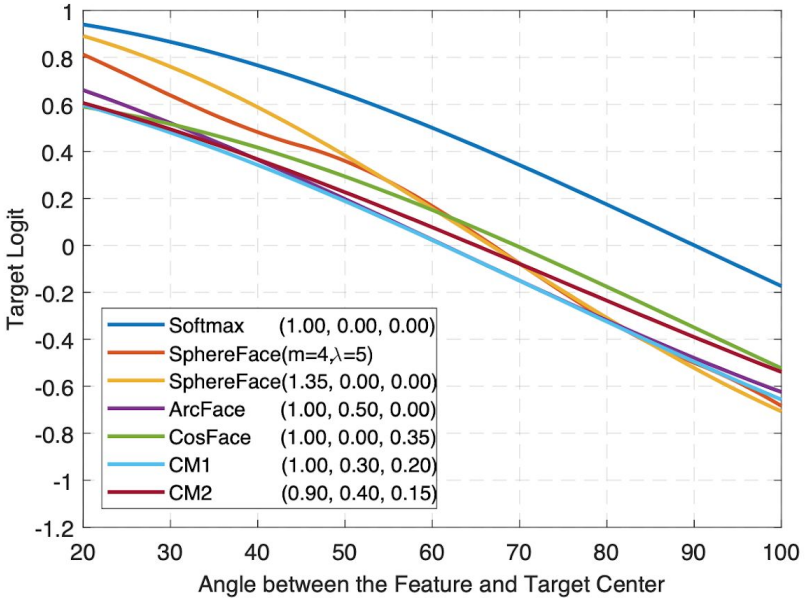
By combining all of the margin penalties, we implement SphereFace, ArcFace and CosFace in an united framework with m_1, m_2 and m_3 as the hyper-parameters.

$$L_4 = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s(\cos(m_1\theta_{y_i}+m_2)-m_3)}}{e^{s(\cos(m_1\theta_{y_i}+m_2)-m_3)}+\sum_{j=1,j\neq y_i}^{n}e^{s\cos\theta_j}}$$

Arcface

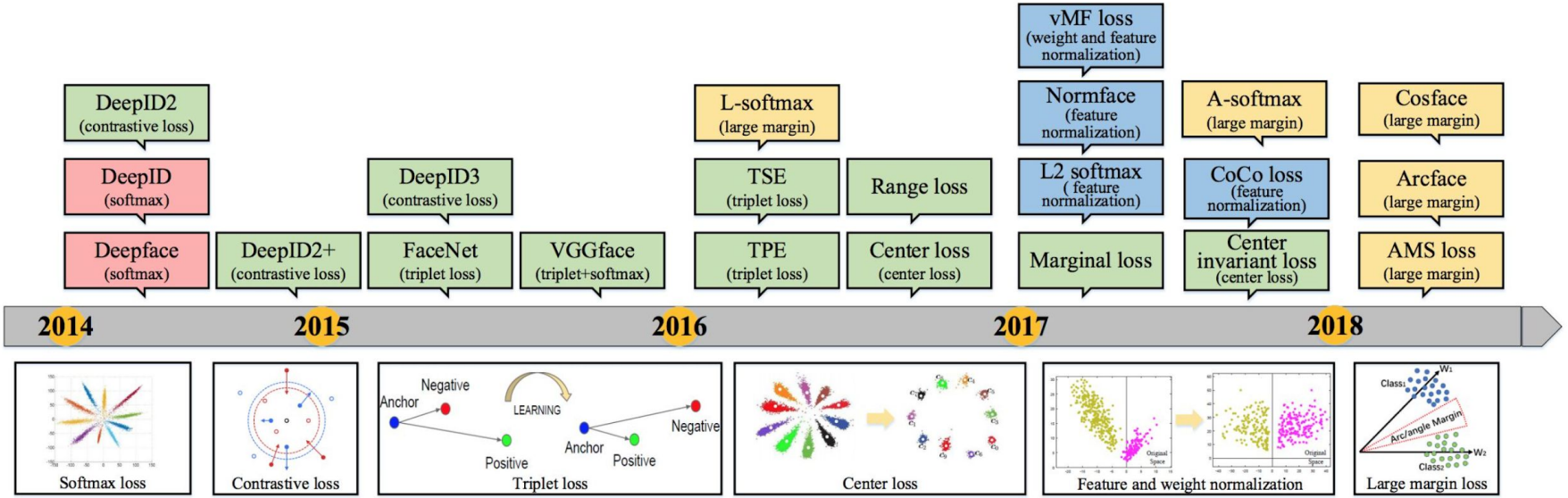


(a) θ_j Distributions



(b) Target Logits Curves

История развития подходов



полезное на github

- https://github.com/KevinMusgrave/pytorch-metric-learning