

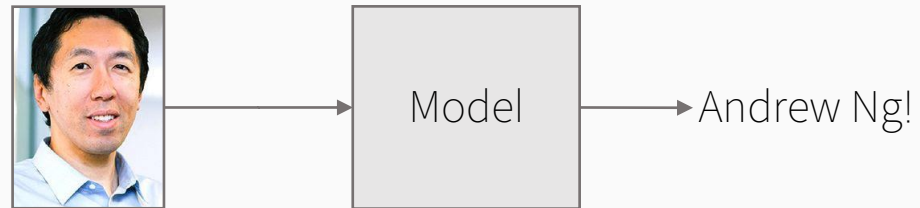
DeepIntoDeep

Deep Metric Learning

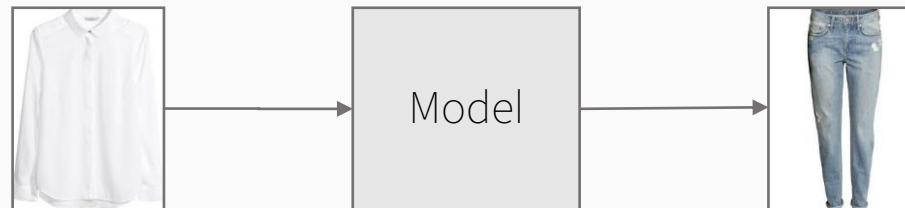
발표자: 오원준

Introduction

Face Recognition Task

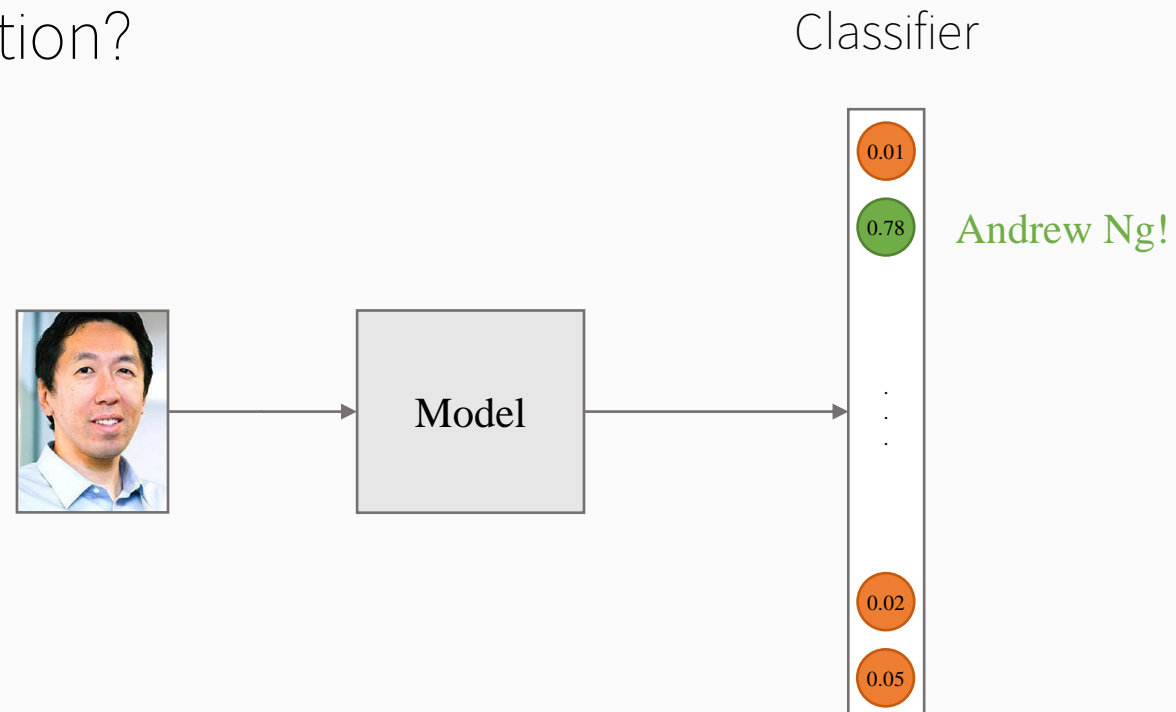


Fashion Compatibility, Recommendation Task



Introduction

How to solve? Classification?



Introduction

Problems...

[-] Size of model

4Byte \times previous hidden dimension \times # of total faces

[-] Overfitting

Not enough data for each class

[-] Not generalizable for unseen classes

What if a new baby is born?

[-] Inference computation cost

Imagine Face-ID takes 5 minutes to recognize your face

Problem Define

Closed-Set

- Data used in training environment
- Kind of classification using input identities

Open-Set

- Data used in testing environment
- $\{\text{Testing data}\} \cap \{\text{Training data}\} = \emptyset$
- Unseen classes
- Can change everyday

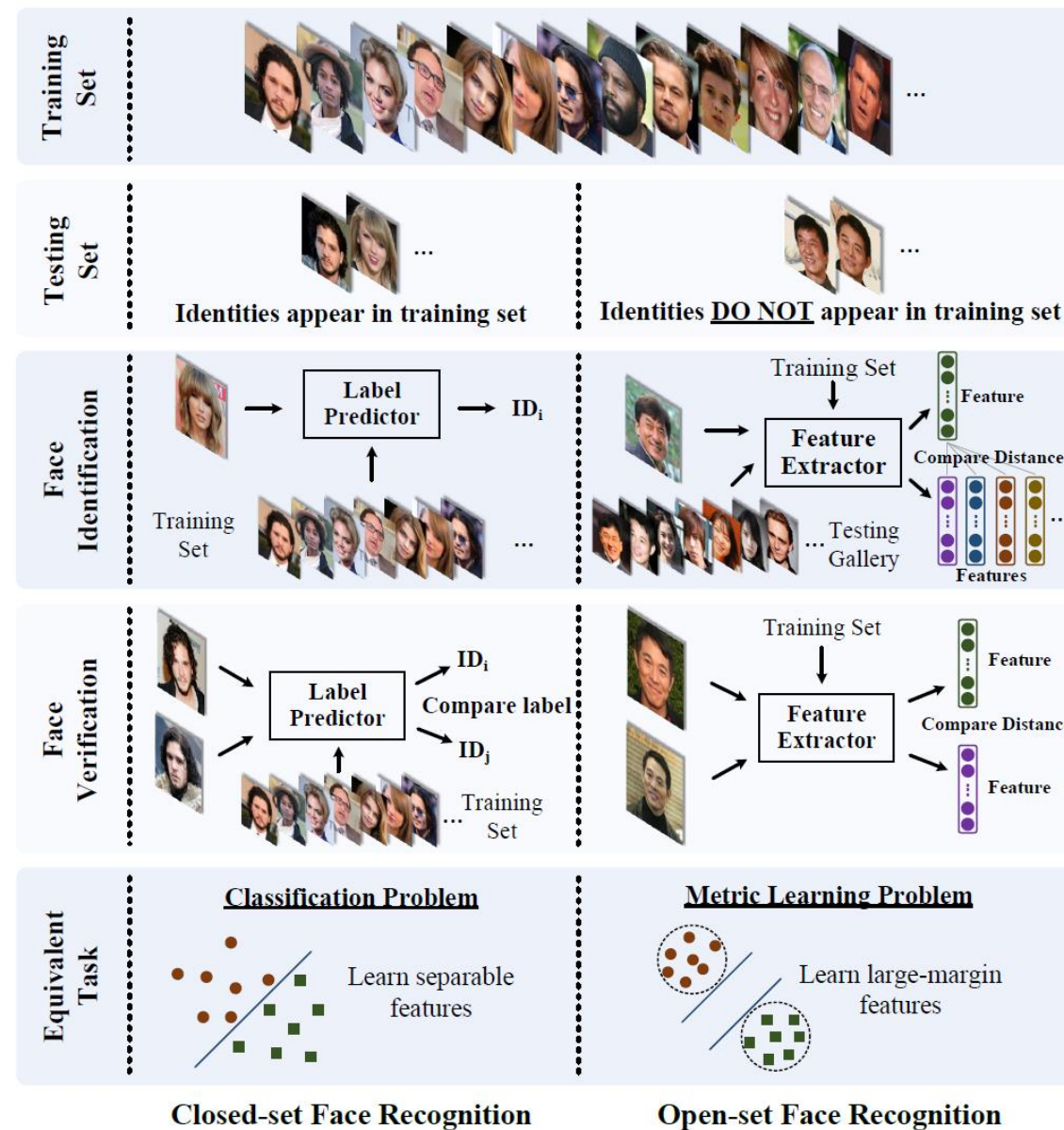


Figure 1: Comparison of open-set and closed-set face recognition.

Metric Learning is All You Need

Keypoints of Metric Learning

Rather directly solving the classification problems...

Make an Expression that best describes the items

→ Use Expression to solve other problems

How to make Best Expression?

→ Use Metric Learning!

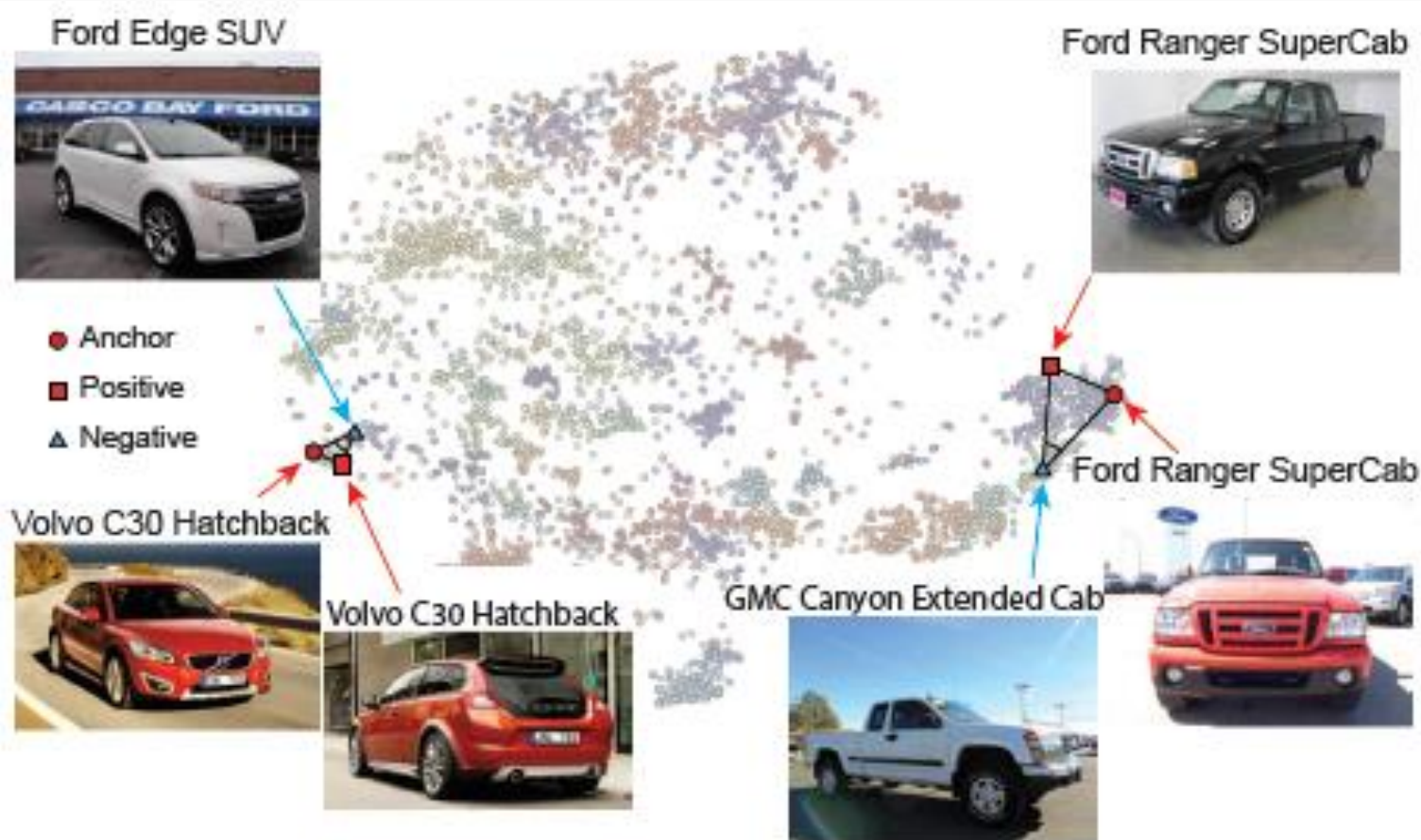


Figure 1. Example of feature embedding computed by t-SNE [32] for the Stanford car dataset [18], where the images of Ford Ranger SuperCab (right) have a more diverse distribution than Volvo C30 Hatchback (left). Conventional triplet loss has difficulty in dealing with such unbalanced intra-class variation. The proposed angular loss addresses this issue by minimizing the scale-invariant angle at the negative point.

3 Main Themes for Metric learning

1. How to create features of the input
(Model Selection)
2. Quantify loss and corresponding the similarity measure
(Triplet-L2 dist, NTXent-Cosine, etc...)
3. Select Sampling Method
(offline/online, batch-hard/batch-all, etc...)

How to create features of the input

Create embeddings for similarity

(Expression \doteq Embedding \doteq Feature \doteq Hidden Representation)

Past: PCA, LDA

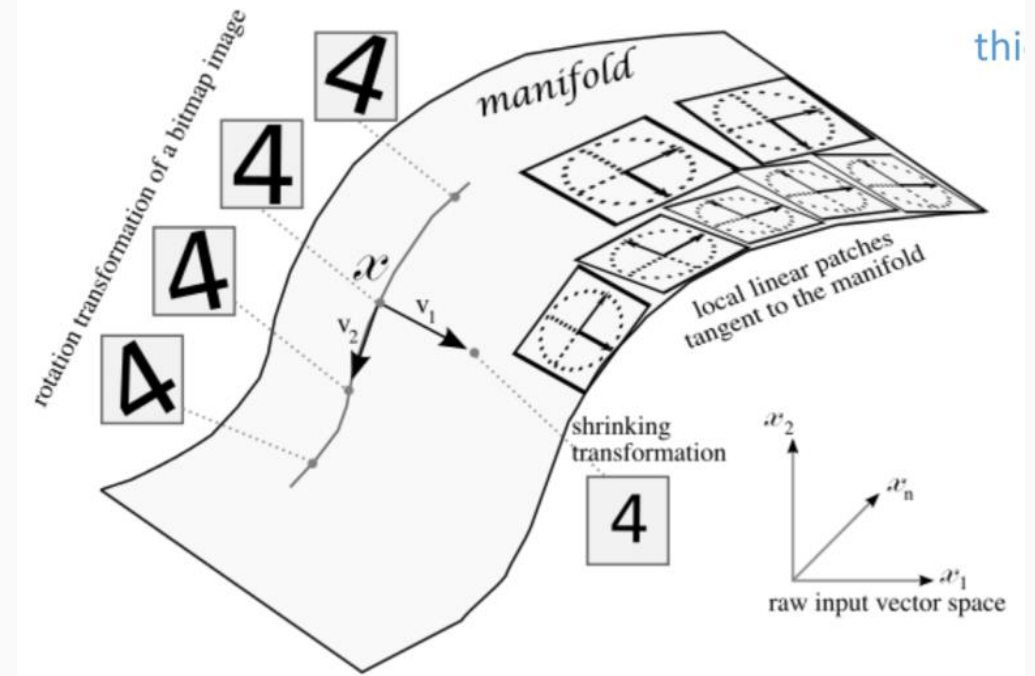
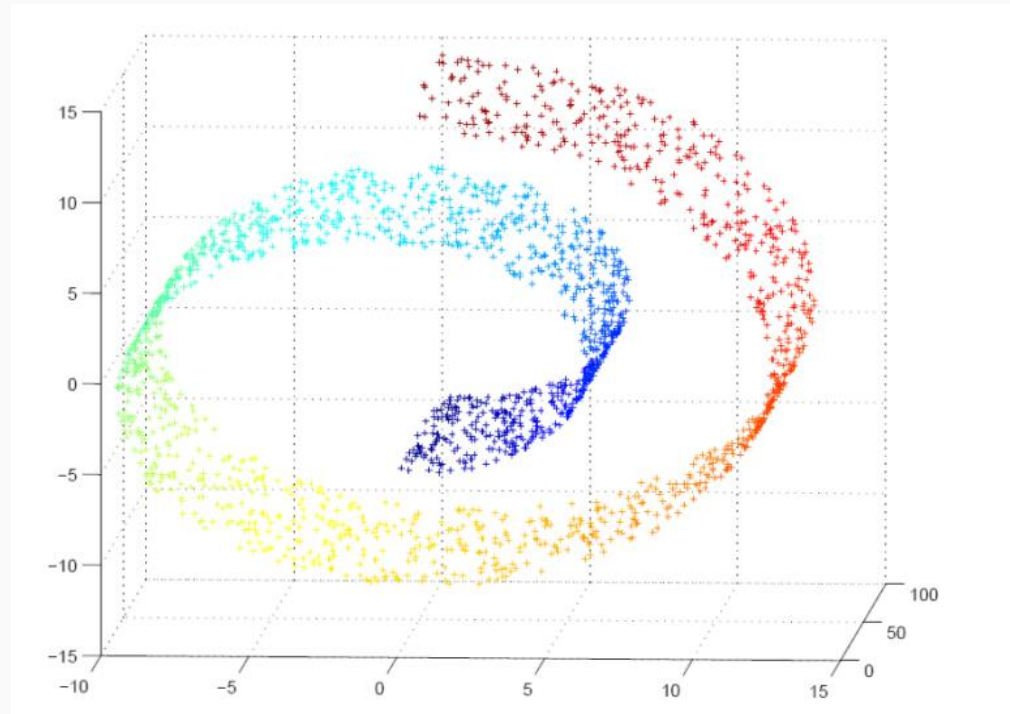
Now: Deep Learning Models(CNN, RNN, Transformer etc...)

Deep Learning Models much better performs than previous methods

How to create features of the input

Deep learning models map high dimensional inputs to low dimensional space.

→ Find good manifold!



Quantify loss and corresponding the similarity measure

similarity measure = 'Metric'

(Reason why we call method as 'metric learning')

Good Metric?

- $d(x, y) \geq 0$ (*non - negativity*)
- $d(x, y) = 0$ *iff* $x = y$
- $d(x, y) \leq d(x, z) + d(x, y)$ (*trangular inequality*)
- $d(x, y) = d(y, x)$ (*symmetry*)

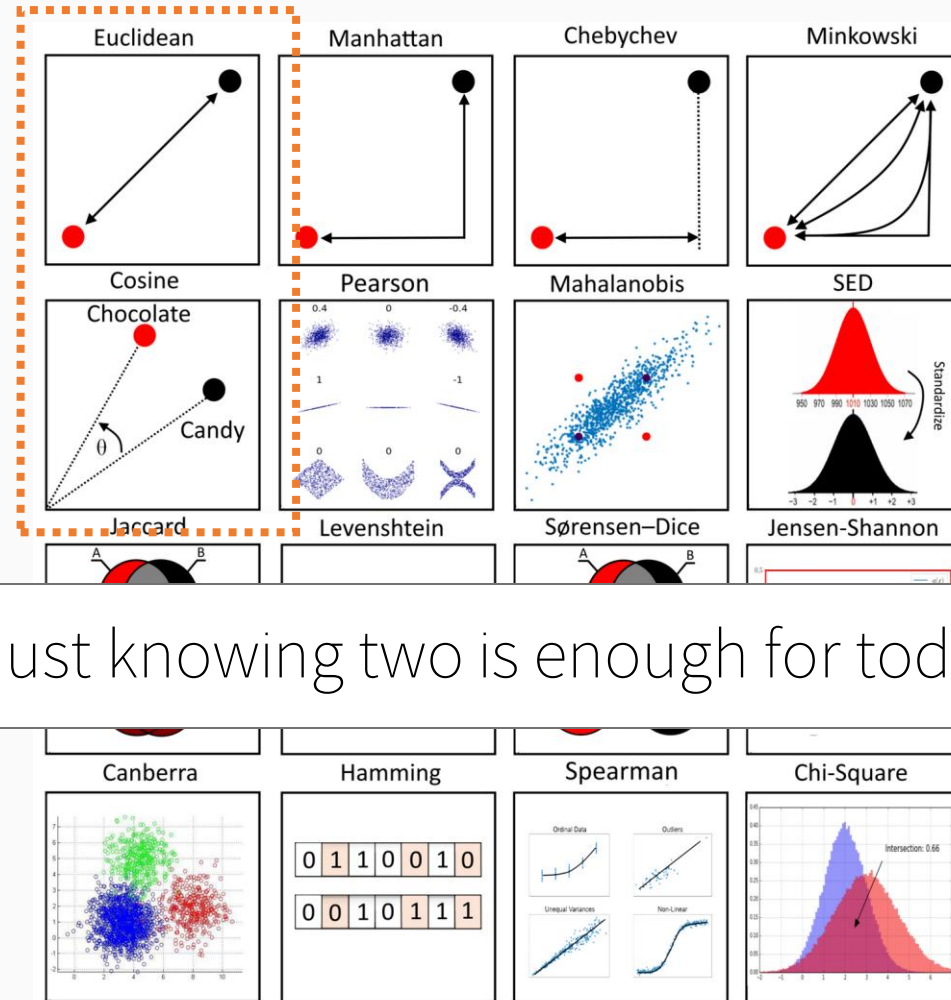
Quantify loss and corresponding the similarity measure

By using similarity measure... We need **LOSS FUNCTION!**

- Contrastive Loss
 - Triplet Loss
 - InfoNCE
- Etc...

Talk deeply later...

Quantify loss and corresponding the similarity measure



Just knowing two is enough for today!

Select Sampling Method

How to sample Negative, Positive samples...

Also, Talk deeply later...

3 Main Themes for Metric learning

3 Main Themes for Metric learning

- Model Structure
- Loss
- Sampling Method

Please Think about...

Which category each part belongs to!

Deep dive into Metric Learning!

Softmax Loss

Classification loss

Softmax Loss

- Start of Face Recognition
- Consider Decision Boundary of BCE

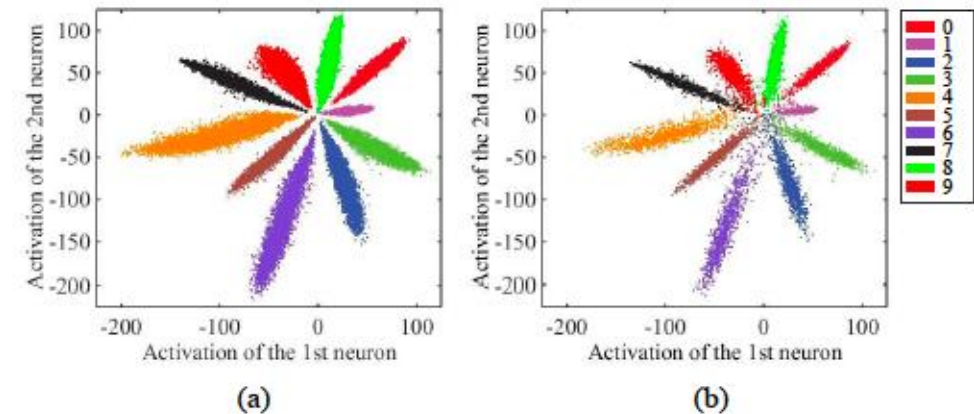
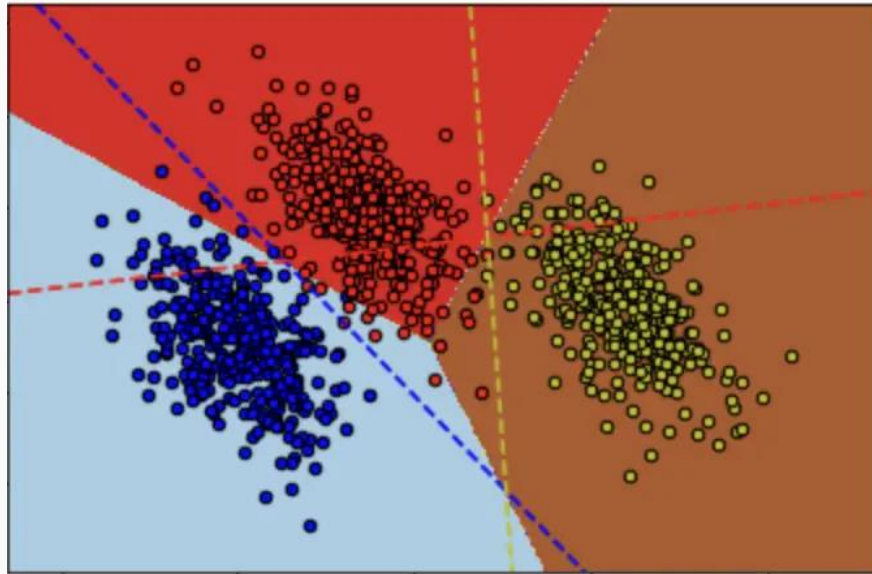
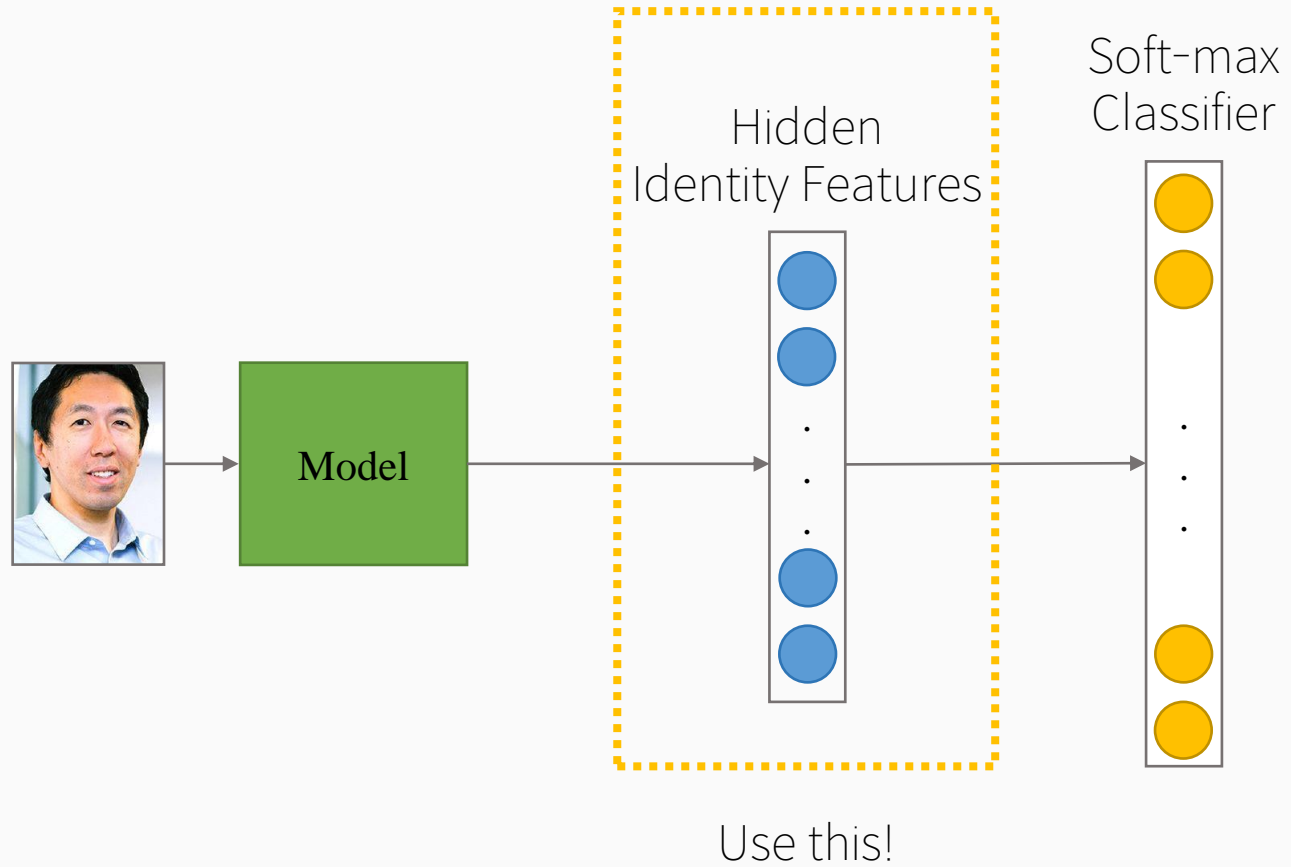


Fig. 2. The distribution of deeply learned features in (a) training set (b) testing set, both under the supervision of softmax loss, where we use 50K/10K train/test splits. The points with different colors denote features from different classes. **Best viewed in color.**

(1) https://programmatically.com/the-softmax-function-and-multiple-logistic-regression/#google_vignette

(2) Yandong Wen, Kaipeng Zhang et al. (2016). A discriminative feature learning approach for deepface recognition.



Softmax Loss: Pros and Cons

[+] Easy to use

[+] Easy to train

[−] Too much parameter for FC layer

[−] Soft-max classifier doesn't guarantee good manifold

[−] Indirectness(Pray...)

Good Feature?

Good Feature?

What you want to do with good features

- Clustering
- Verification
- Recognition

Softmax is Insufficient

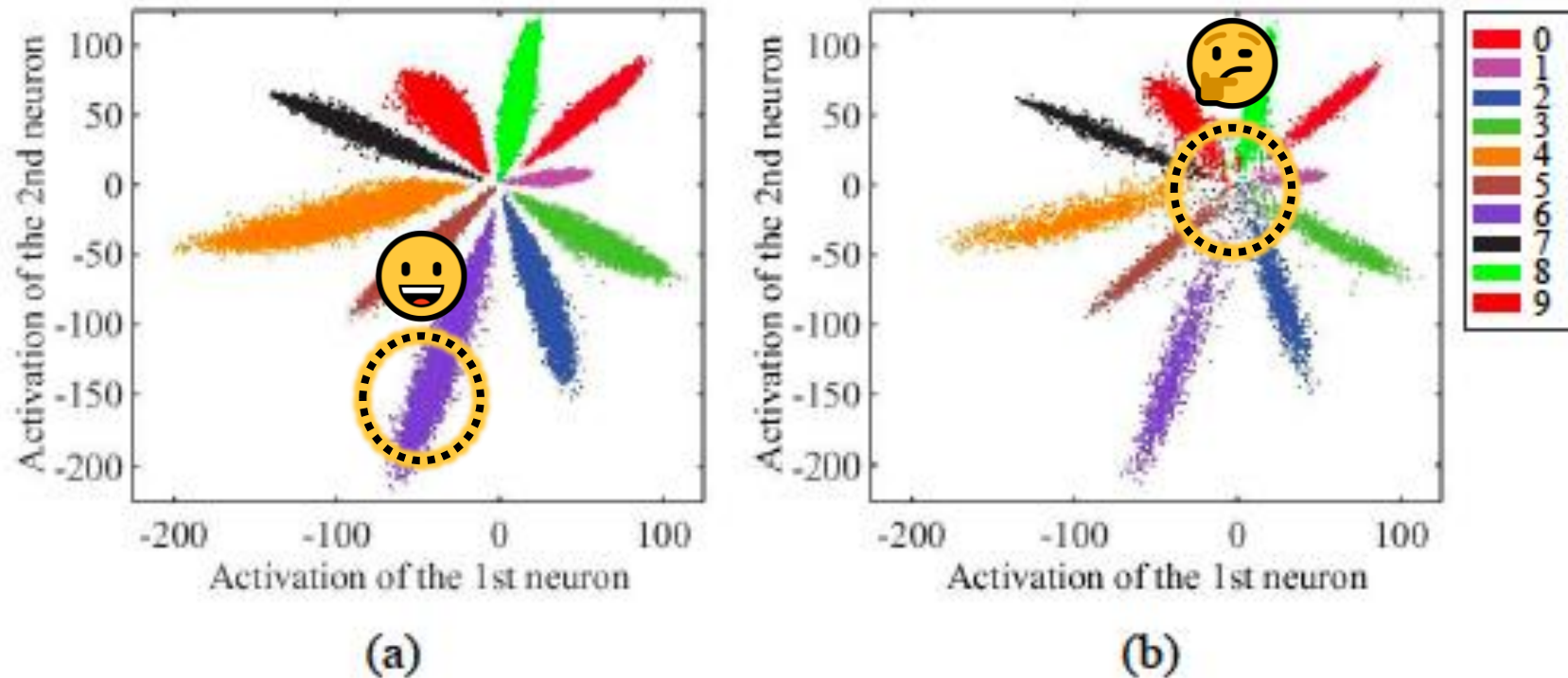


Fig. 2. The distribution of deeply learned features in (a) training set (b) testing set, both under the supervision of softmax loss, where we use 50K/10K train/test splits. The points with different colors denote features from different classes. **Best viewed in color.**

What we want is...

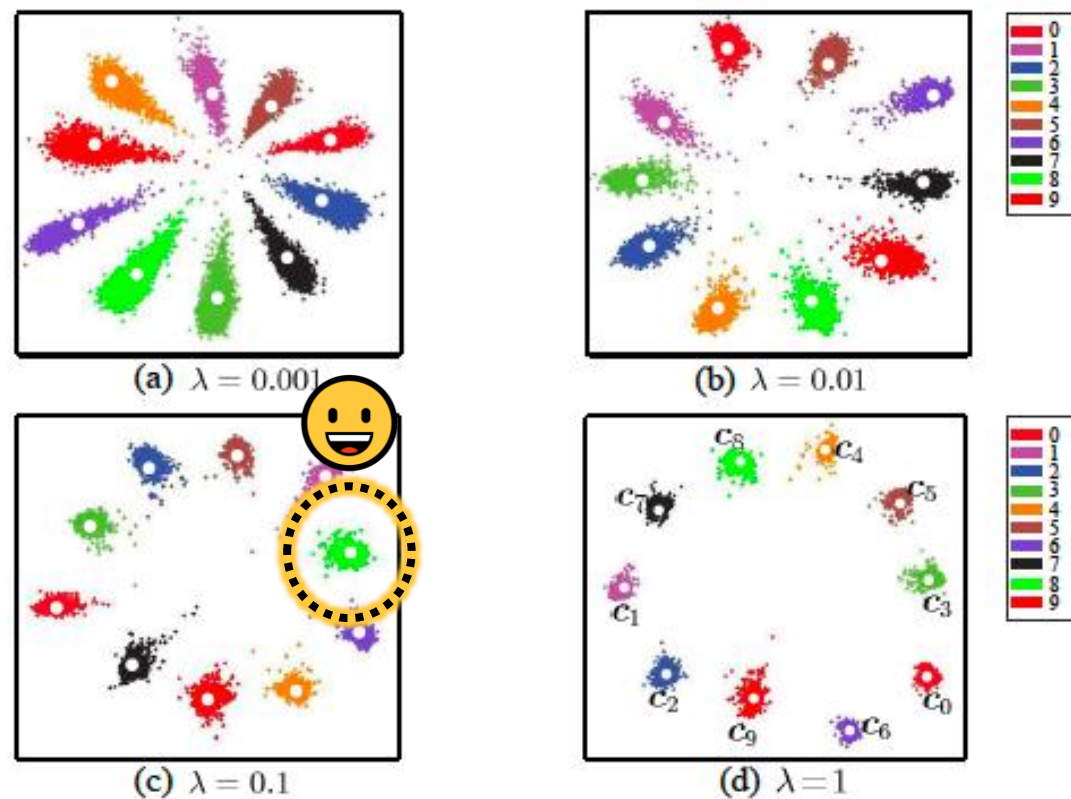
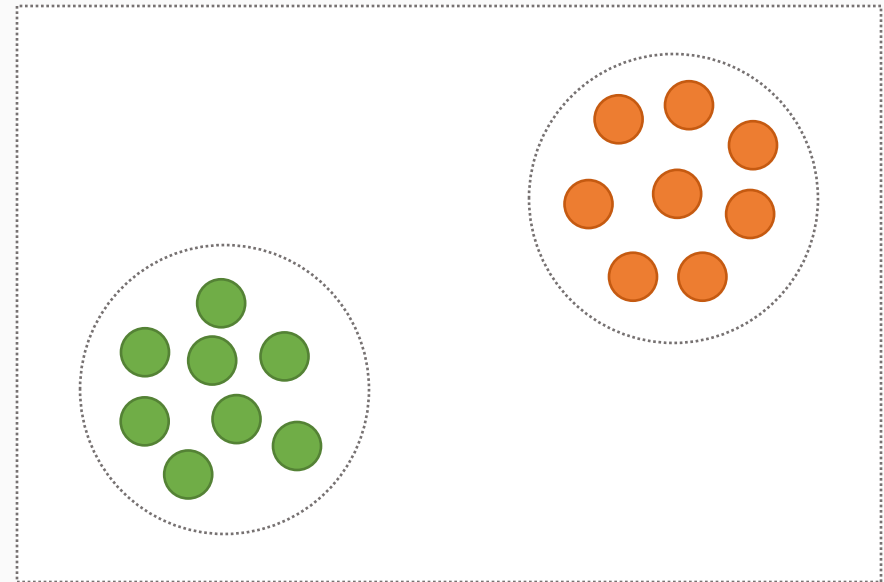
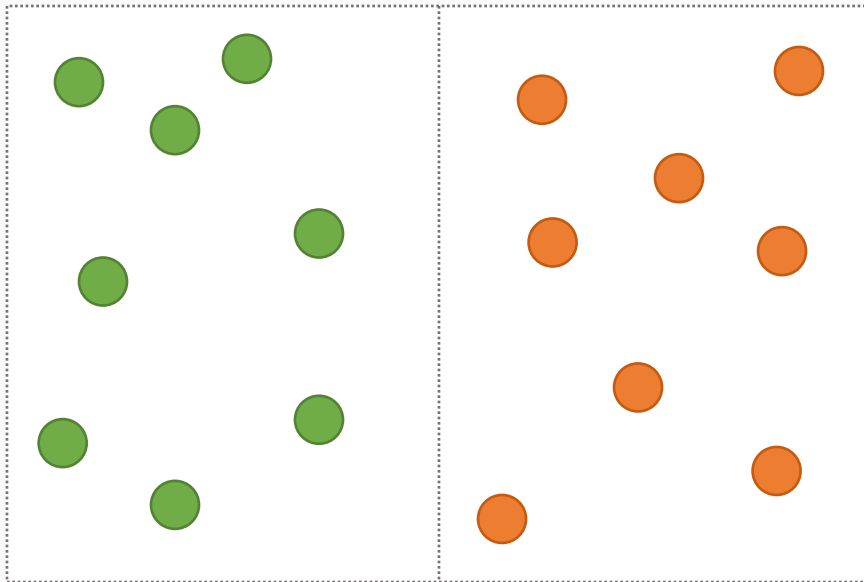


Fig. 3. The distribution of deeply learned features under the joint supervision of softmax loss and center loss. The points with different colors denote features from different classes. Different λ lead to different deep feature distributions ($\alpha = 0.5$). The white dots (c_0, c_1, \dots, c_9) denote 10 class centers of deep features. **Best viewed in color.**

3 conditions of good expression

- **Separable** (Inter-class variation \uparrow)
- Discriminable (Intra-class variation \downarrow)
- **Generalizable**



To improve softmax...

A lot of ways

- Center loss...
- Angle margin to Softmax...

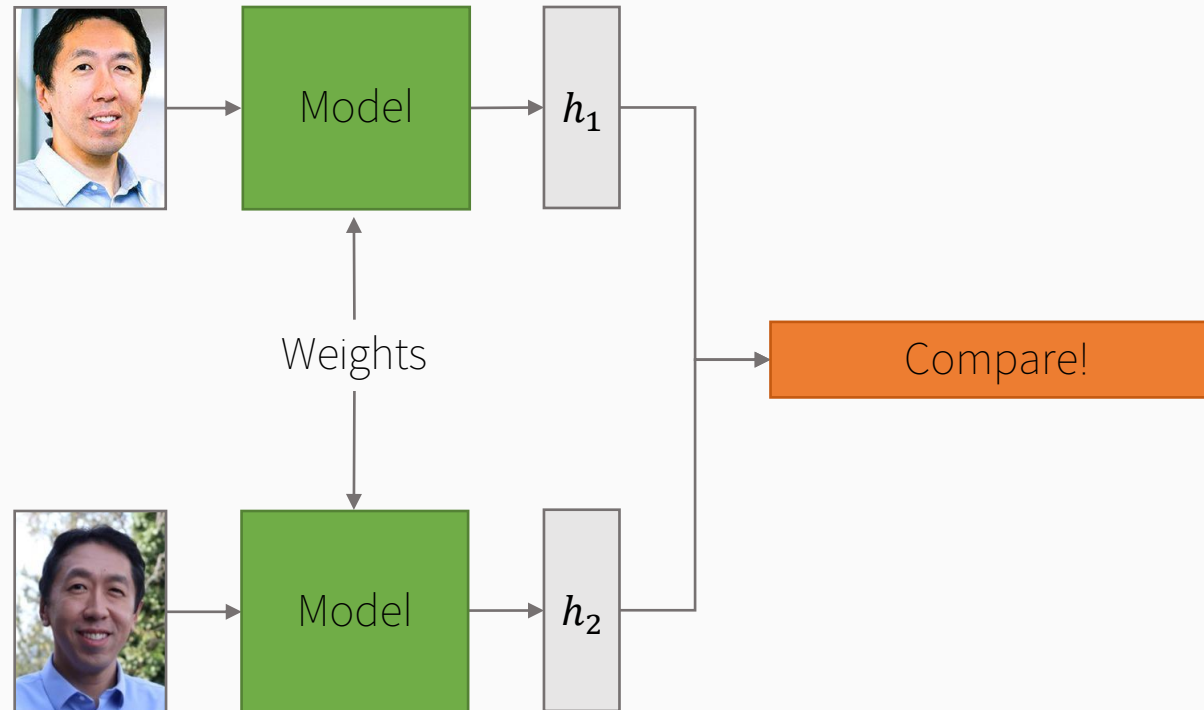
But we will just concentrate on “Contrastive Learning”

Siamese Network

Siamese Network

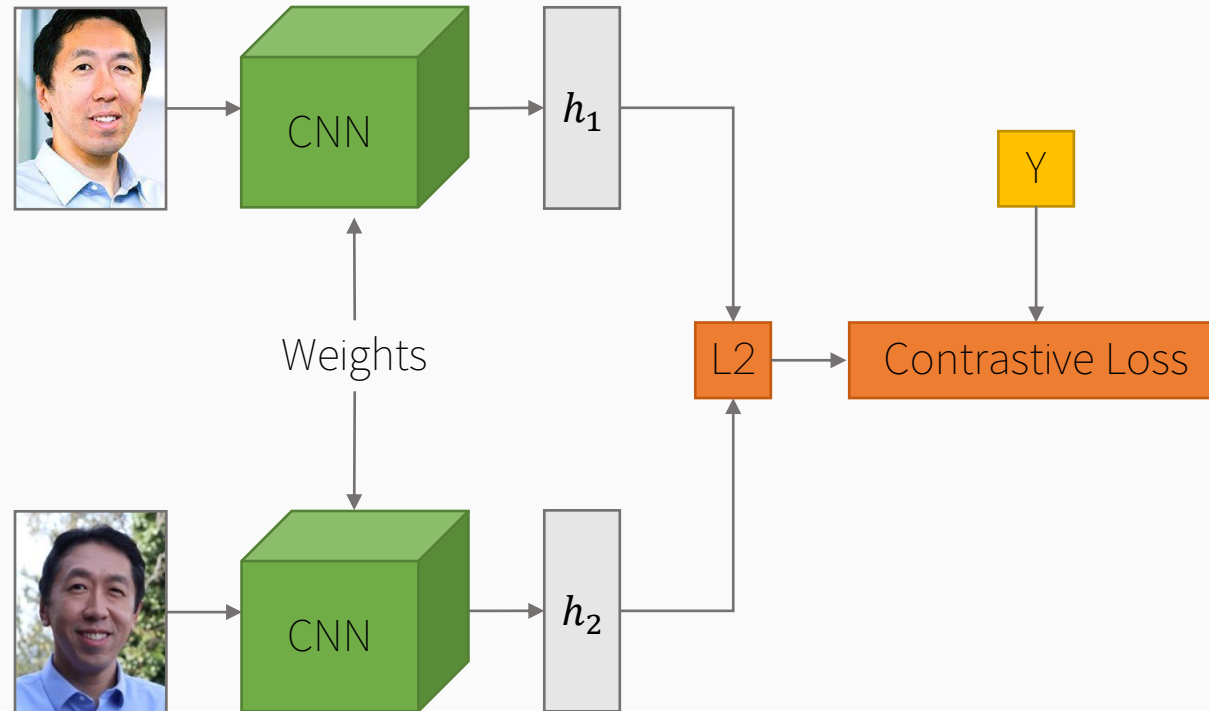
A **Siamese neural network** (sometimes called a **twin neural network**) is an [artificial neural network](#) that uses the same weights while working in tandem on two different input vectors to compute comparable output vectors.^{[1][2][3][4]} Often one of the output vectors is precomputed, thus forming a baseline against which the other output vector is compared. This is similar to comparing [fingerprints](#) but can be described more technically as a distance function for [locality-sensitive hashing](#).^[citation needed]

Siamese Network



Contrastive Loss

Contrastive Loss



Contrastive Loss

Directly learn metrics!

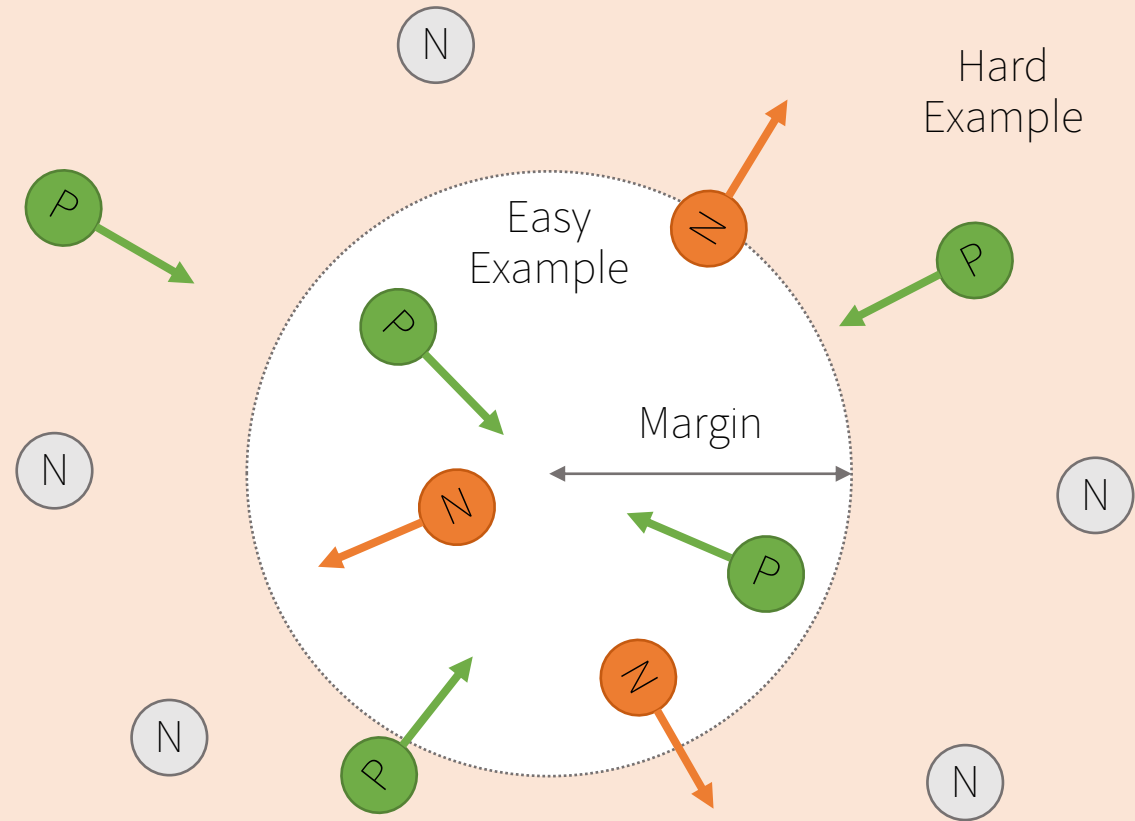
Main idea:

- (same class) attract: the distance between embeddings goes to 0
- (different class) repel: learn so that the distance is greater than α (margin)

$$\mathcal{L}_{contrastive} = \frac{1}{2}[(1 - Y) \cdot D_{ij}^2 + Y \cdot \max(0, m - D_{ij})^2]$$

$$D_{ij} = \| f(x_i) - f(x_j) \|_2$$

Contrastive Loss



Contrastive Loss



Nothing to Learn

Contrastive Loss: Pros and Cons

[+] Directly optimize distance between samples

[+] Relatively easy to organize a dataset

[−] No tolerance for Intra-class variation

[−] Tuning 'margin' requires a lot of efforts

[−] Same margin for any negative samples

[−] Needs Hard Mining

Vector collapse in Contrastive Loss

Kind of Overfitting problem

- Positive pair is continuously mapped to one point,
- the points are not smoothly sprayed

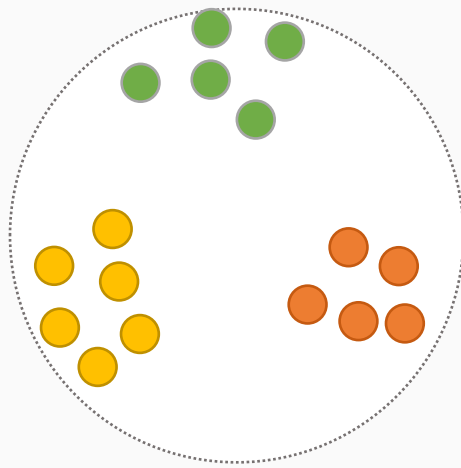
Instead of seeing the general characteristics of the whole,
it becomes **overfitted to a specific feature**.

Generalization performance ↓

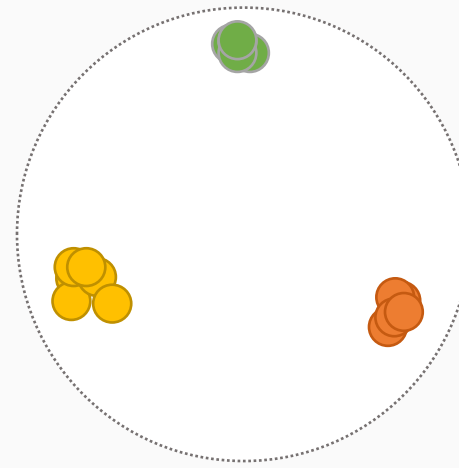
How to Prevent? Hard sample mining, Early stopping etc...

Vector collapse in Contrastive Loss

W/O Collapse

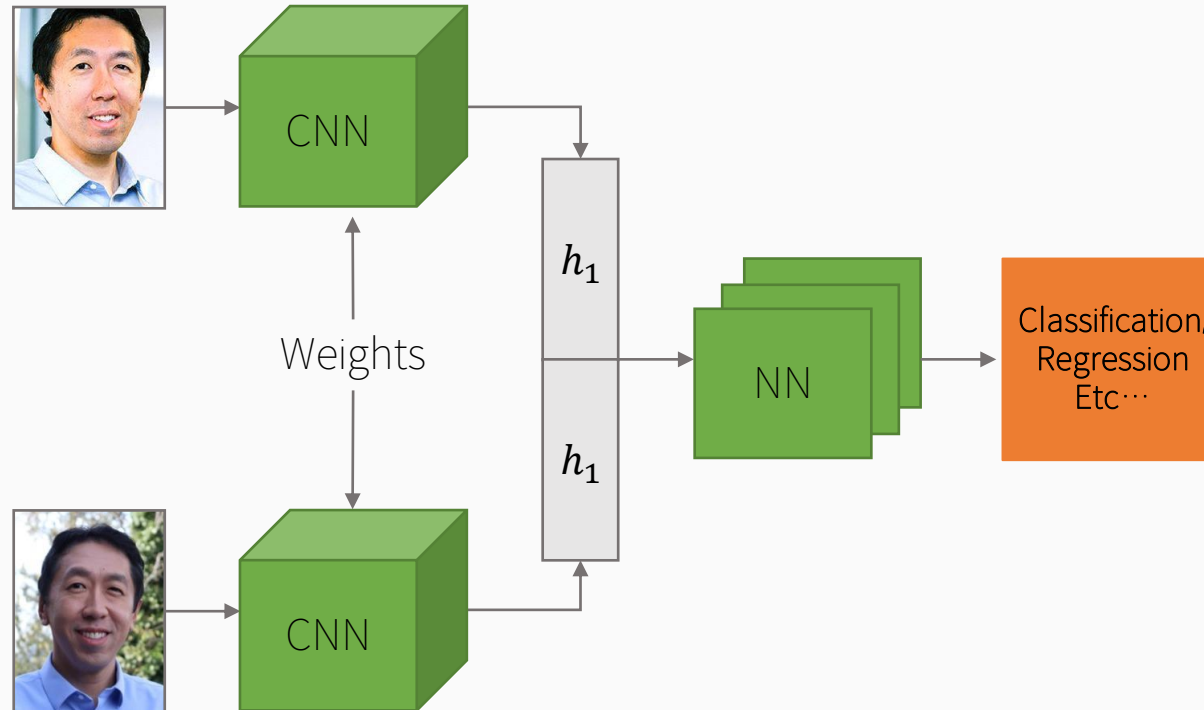


With Collapse



(Optional) Siamese for one-shot classification

Siamese for one-shot classification



Siamese for one-shot classification: Pros and Cons

[+] Higher performance than Contrastive Loss(Maybe?)

[+] *Discussion!*

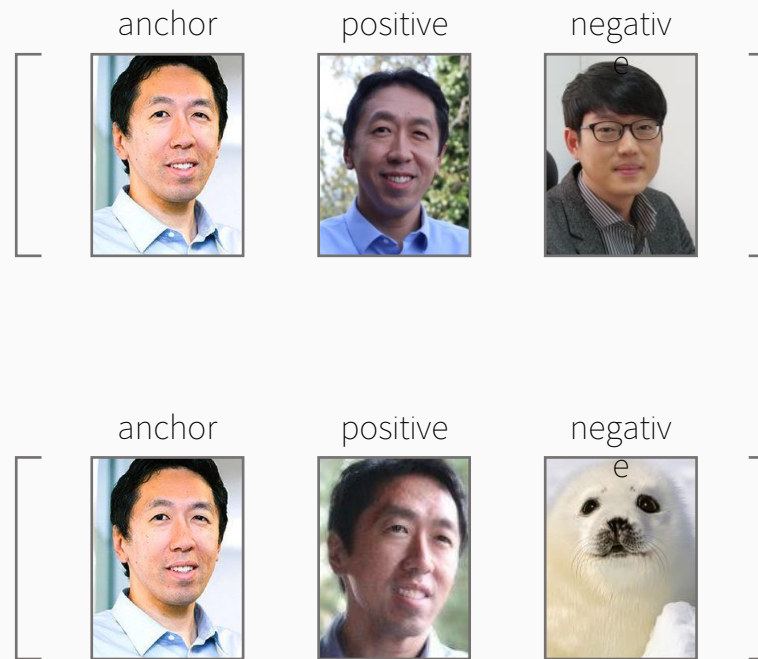
[−] *Discussion!*

Discussion1(개인)

1. Why does a one-shot mechanism produce better results than contrastive methods?
2. Why is it good to use this kind of structure? (Data Perspective)
3. What are some practical disadvantages compared to Contrastive loss?

Triplet Loss

Triplet Loss



Triplet Loss

Main idea:

- Find distances for the same class (Anchor-Positive)
- Find distances for the different class (Anchor-Negative)
- Learn so that the difference between the two distances is more than m (margin)

$$\mathcal{L}_{triplet} = \max(0, D_{anc,pos}^2 - D_{anc,neg}^2 + m)$$

$$D_{i,j} = \| f(x_i) - f(x_j) \|_2$$

Triplet Loss

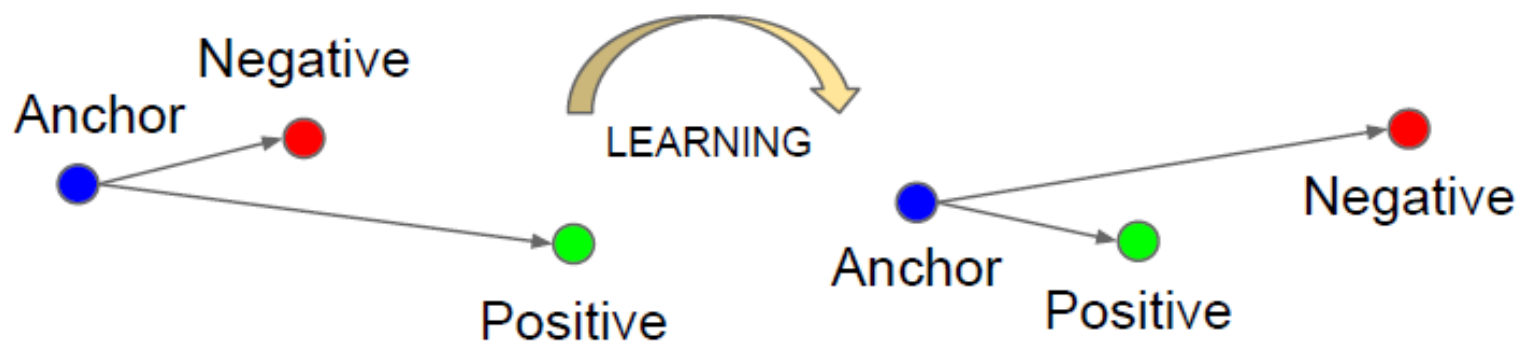
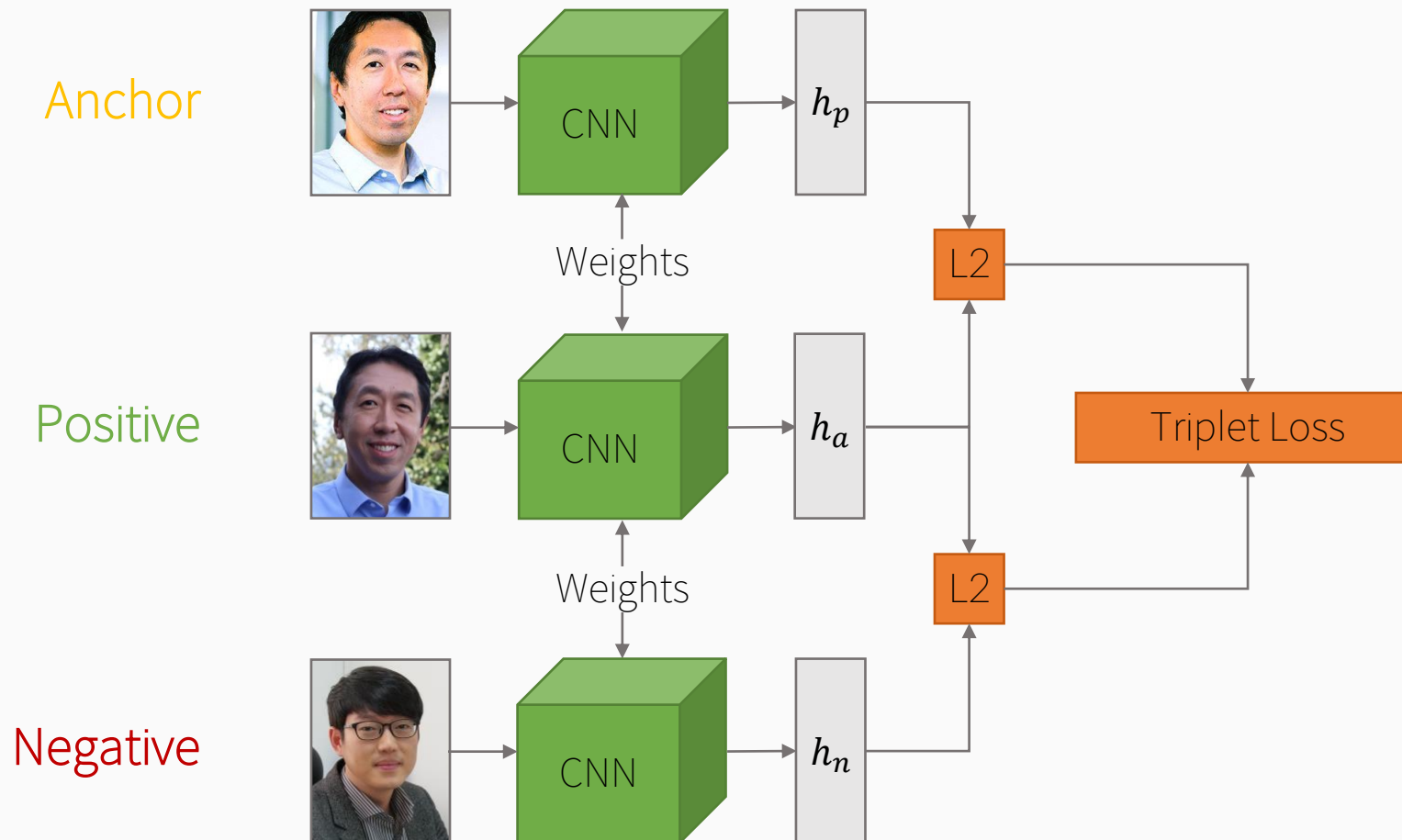
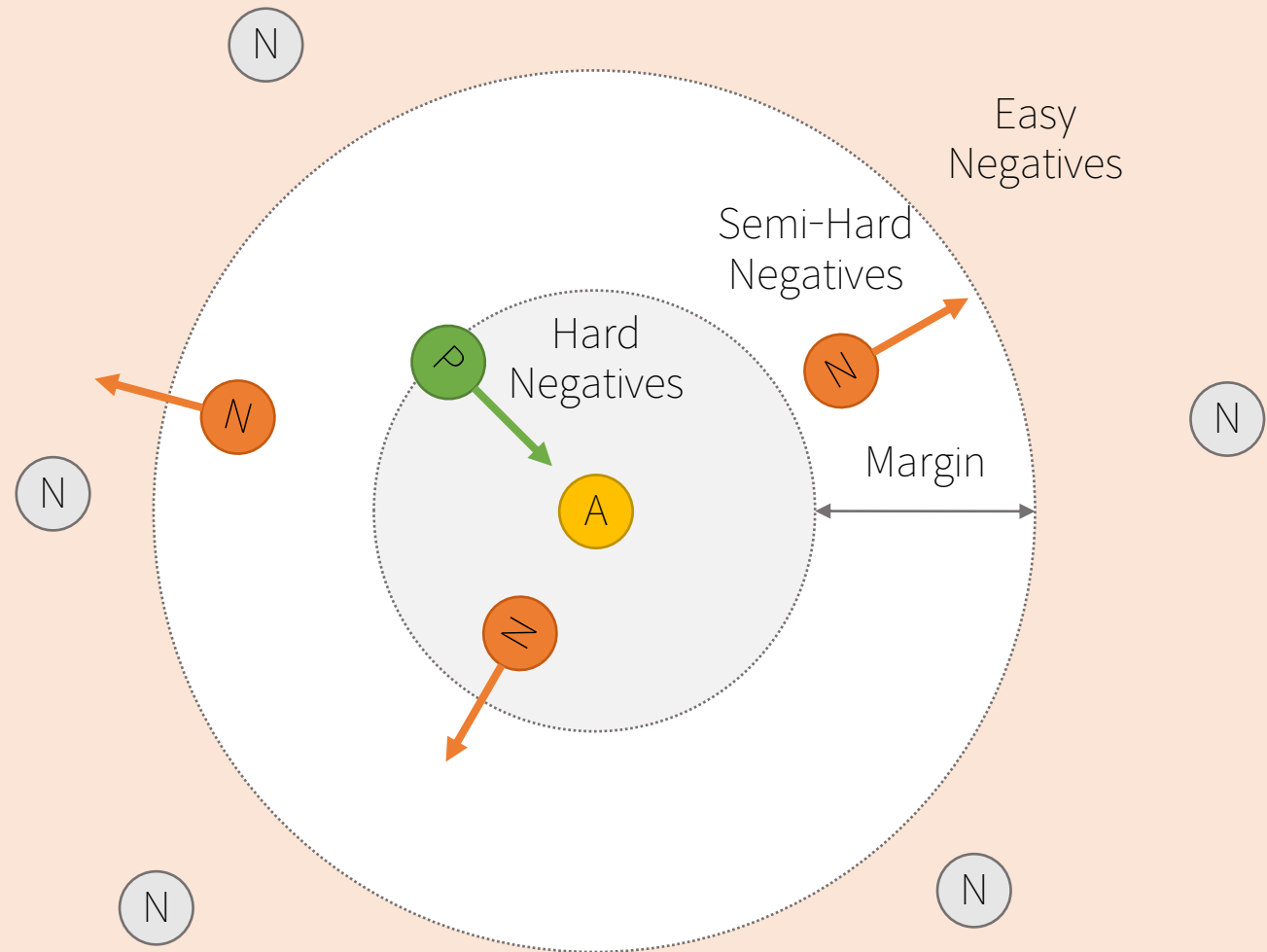


Figure 3. The **Triplet Loss** minimizes the distance between an *anchor* and a *positive*, both of which have the same identity, and maximizes the distance between the *anchor* and a *negative* of a different identity.

Triplet Loss



Triplet Loss



Triplet Loss: Pros and Cons

[+] Ranking(Less greedy)

[+] Tolerance for Intra-class variation

[+] Efficient Semi-hard mining

[−] (Still) Tuning ‘margin’ requires a lot of efforts

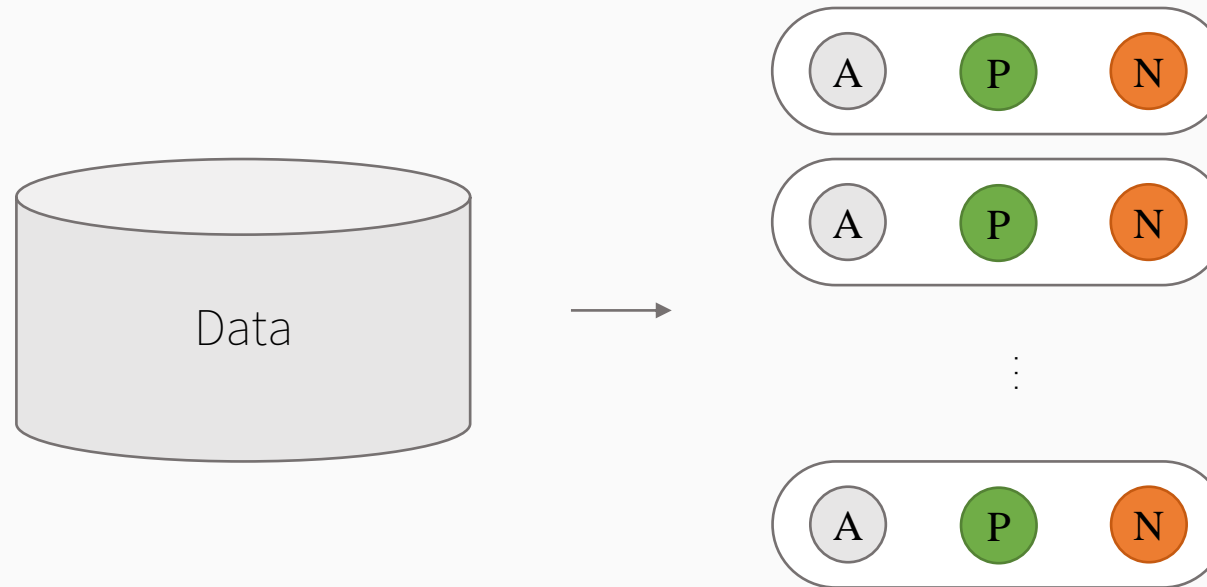
[−] (Still) Same margin for any pairs

[−] Too much computation[$O(N^3)$] + Needs hard-mining

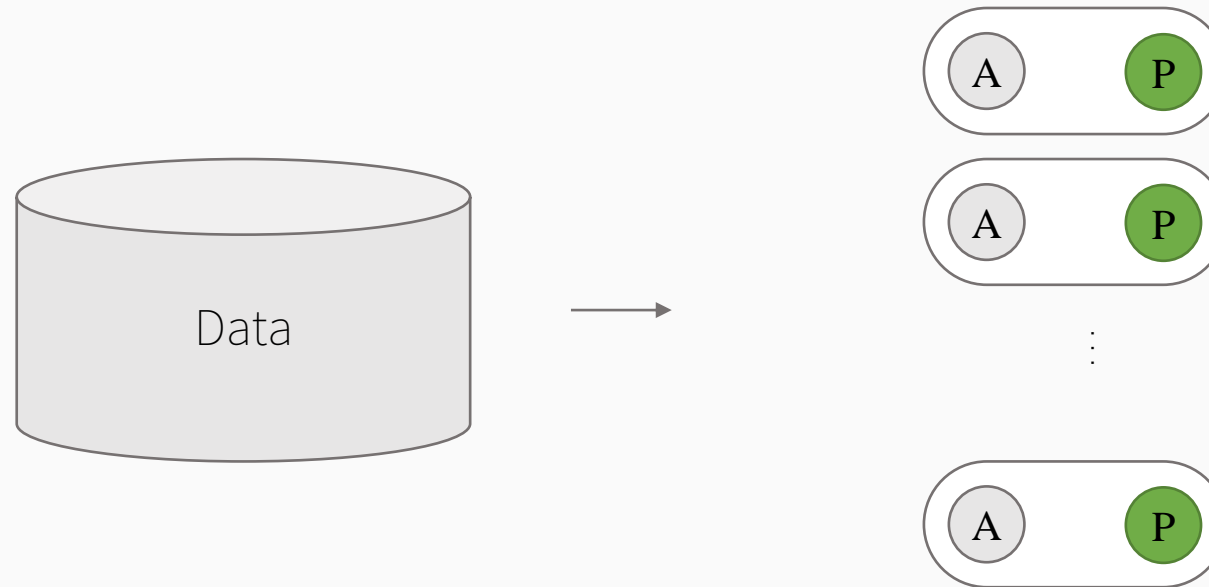
Efficient Batch Mining Techniques

For Triplet Loss

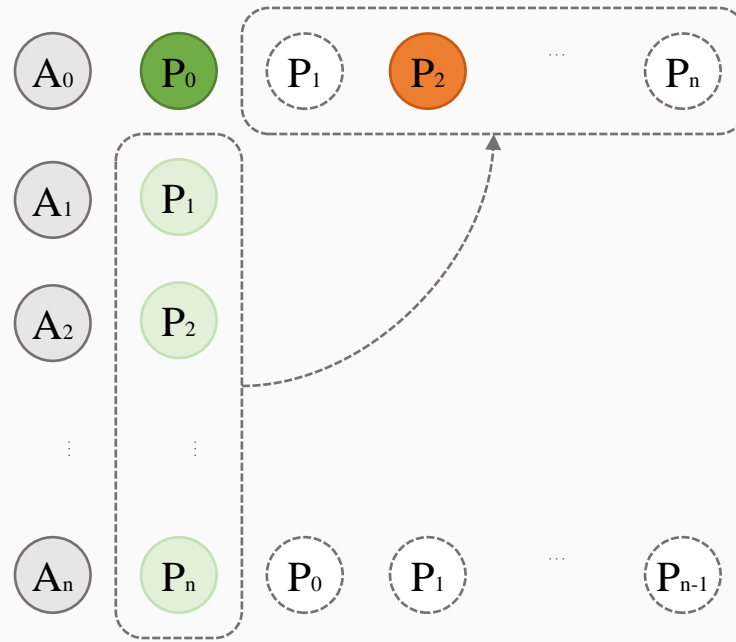
Offline mining



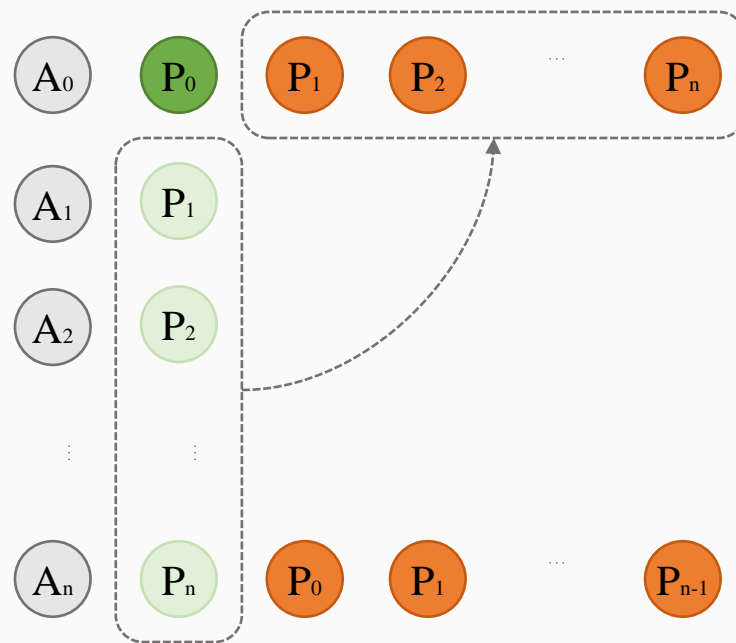
Online mining



Batch-hard: Choose Hardest 'Semi-hard' Negative



Batch-all: Average Negatives



Hypersphere manifold / Feature normalization

Hypersphere Manifold / Feature normalization

Back to Triplet loss...

- N dimension vector space
- But **not enough** information to find optimal structure
(Triplet just use simple distance)

Do Feature normalization!

- Kind of Inductive bias
- Make manifold as Hypersphere!

Hypersphere Manifold / Feature normalization

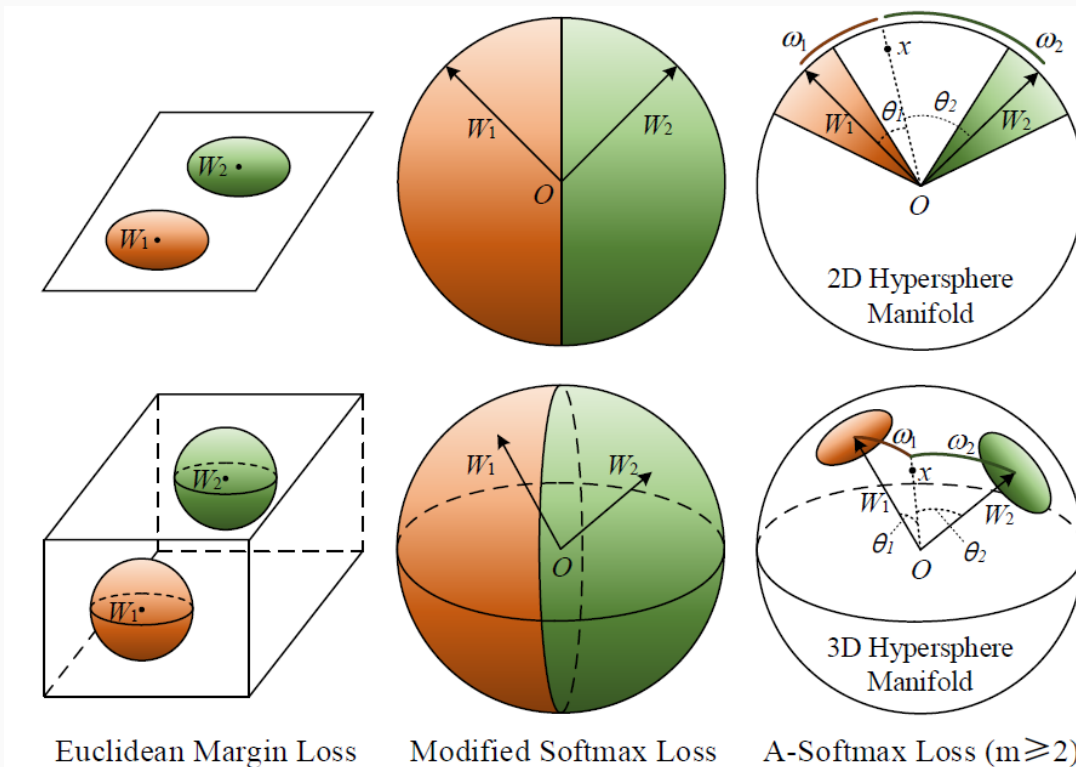


Figure 3: Geometry Interpretation of Euclidean margin loss (e.g. contrastive loss, triplet loss, center loss, etc.), modified softmax loss and A-Softmax loss. The first row is 2D feature constraint, and the second row is 3D feature constraint. The orange region indicates the discriminative constraint for class 1, while the green region is for class 2.

Discussion2(팀)

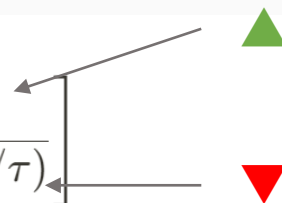
We learned that Metric Learning can be applied to various fields such as Face Recognition, Information Retrieval, and Recommendation etc...

Among the various data, please select data type that you think would be good to assume Hyperspace manifold or not, explain why you think it would be better.

InfoNCE(Noise Contrastive Estimator)

more Probabilistic-like and more effective approach for contrastive learning

Similar to Log Loss(Cross-Entropy Loss) form...

$$\mathcal{L}_{InfoNCE} = \mathbb{E}_X \left[-\log \frac{\exp(\text{sim}(x, x_+)/\tau)}{\sum_{x_j \in X} \exp(\text{sim}(x, x_j)/\tau)} \right] \quad (1)$$


$$= \mathbb{E}_X \left[\log \left(1 + \sum_{x_j \in X} \exp\left(\frac{\text{sim}(x, x_j) - \text{sim}(x, x_+)}{\tau}\right) \right) \right] \quad (2)$$

$$\text{sim}(u, v) = u^\top v / \|u\| \|v\|$$

LogSumExp

Approximation of “max” function

→ but Continuous, Convex!

$$\begin{aligned} & \log(1 + \exp(\{sim(x, x_-) - sim(x, x_+)\})) \\ &= \max(0, \{sim(x, x_-) - sim(x, x_+)\}) \end{aligned}$$

Discussion(팀)

In many areas, InfoNCE performs better than Triplet.

Discuss why InfoNCE loss produces better results than Triplet loss.

(Find Structural Advantages of InfoNCE!)

There are various answers!

One way is...

- Transform Triplet into LogSumExp form and show relevancy between “Triplet” and “InfoNCE”

NTXent: Pros and Cons

[+] Discussed earlier!

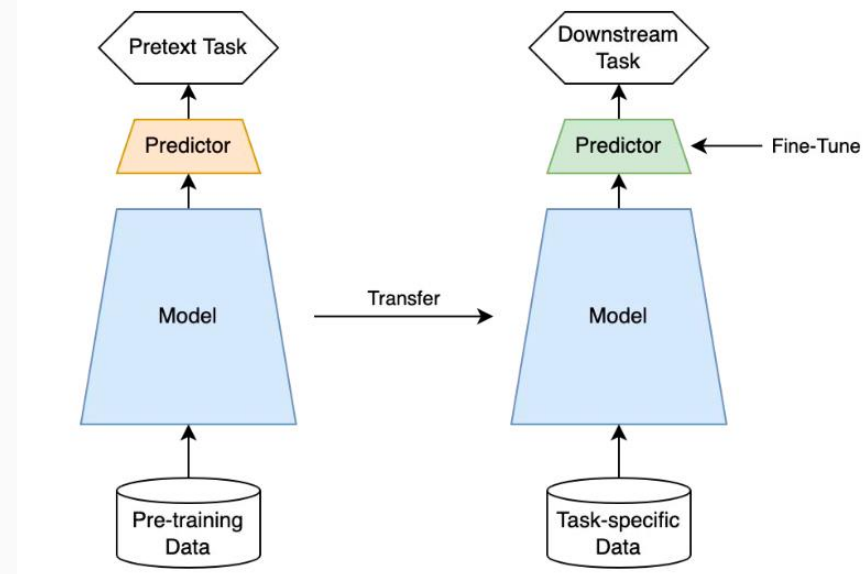
[−] Tuning ‘Temperature’ requires a lot of efforts

(In original paper) the wrong temperature leads to much worse performance than other methods.

Metric Learning in SSL

SSL(Self-Supervised-Learning)

- Kind of training methods that focused on creating good representations from unlabeled dataset
- Model trained by SSL can be used for various downstream tasks by transfer learning
- i.e. BERT, SimCSE, SimCLR, MoCo, etc...



SimCLR

SimCLR

- Use augmented image as positive sample
- Use InfoNCE loss

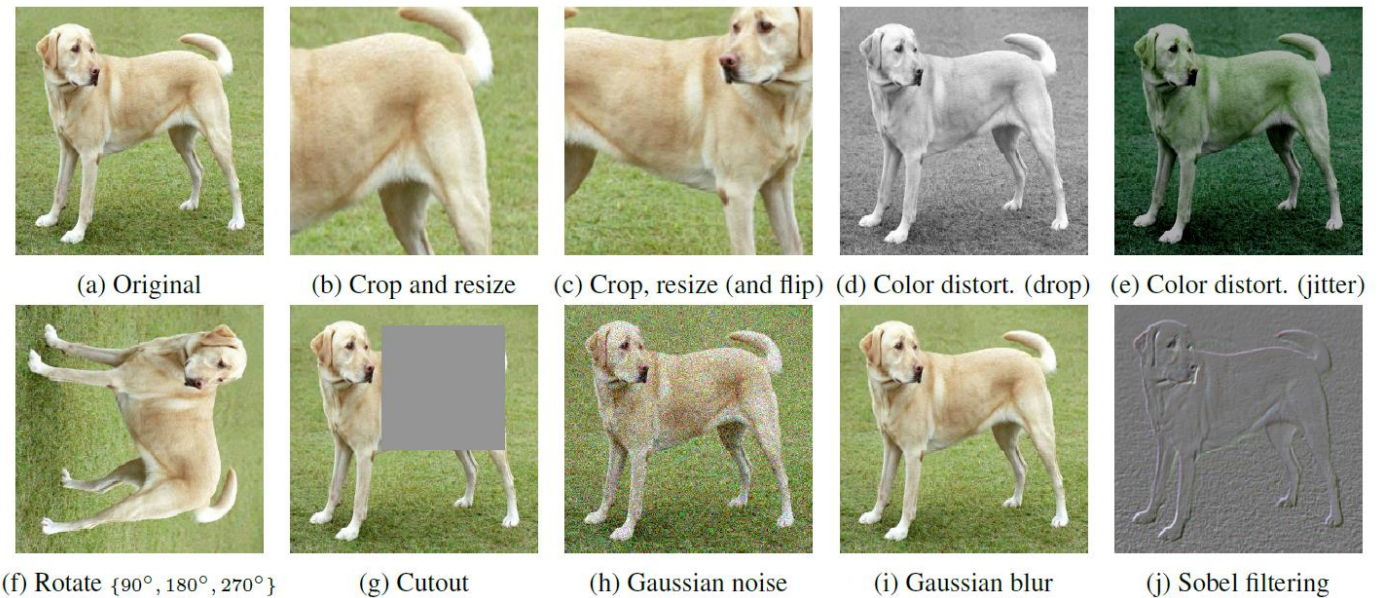
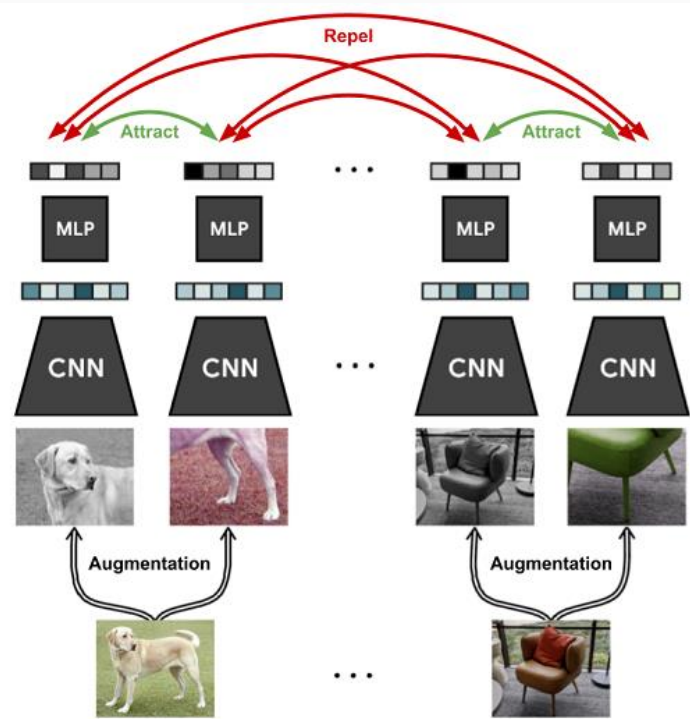


Figure 4. Illustrations of the studied data augmentation operators. Each augmentation can transform data stochastically with some internal parameters (e.g. rotation degree, noise level). Note that we *only* test these operators in ablation, the *augmentation policy* used to train our models only includes *random crop* (with *flip* and *resize*), *color distortion*, and *Gaussian blur*. (Original image cc-by: Von.grzanka)

Metric Learning in NLP

SBERT

Sentence-BERT

- powerful methods to create Sentence Embedding
- Use Siamese, Triplet loss etc...
- Various uses... Retrieval, RAG(LLM), BERTopic etc..

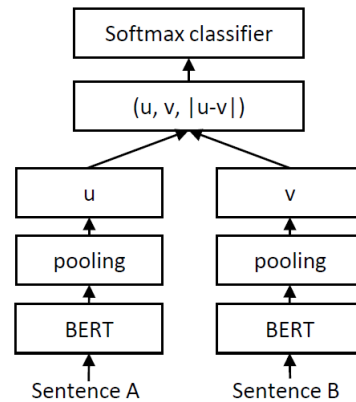


Figure 1: SBERT architecture with classification objective function, e.g., for fine-tuning on SNLI dataset. The two BERT networks have tied weights (siamese network structure).

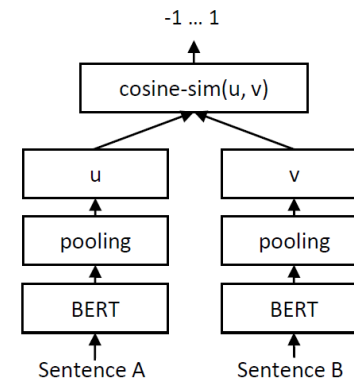


Figure 2: SBERT architecture at inference, for example, to compute similarity scores. This architecture is also used with the regression objective function.

Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks

Multimodality, Alignment

Multimodality, Alignment

Image, Audio, Text etc...

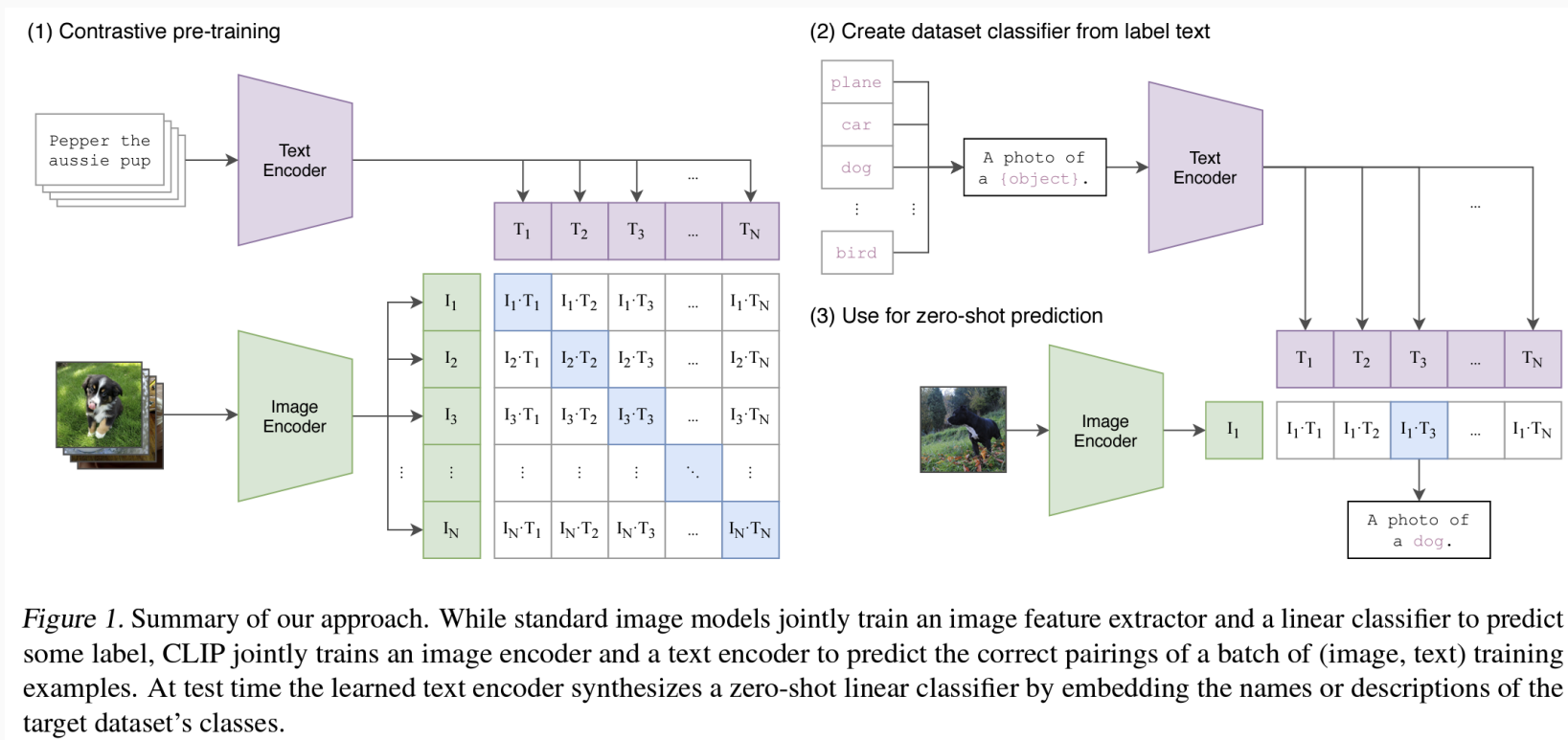
Can't we learn the connection between each other?

How can we map to a one vector space?

-> Metric Learning!

CLIP

CLIP loss(Type of InfoNCE)



Any Questions? 🤔

References(paper)

- Mahmut KAYA. (2019). Deep Metric Learning: A Survey
- R. Hadsell, S. Chopra et al. (2006). Dimensionality Reduction by Learning an Invariant Mapping(DrLIM)
- Gregory Koch, Richard Zemel et al. (2015). Siamese Neural Networks for One-shot Image Recognition
- Florian Schoroff, Dmitry Kalenichenko et al. (2015). FaceNet: A Unified Embedding for Face Recognition and Clustering
- Kihyuk Sohn. (2016). Improved Deep Metric Learning with Multi-class N-pair Loss Objective
- Yandong Wen, Kaipeng Zhang et al. (2016). A discriminative feature learning approach for deepface recognition
- Jian Wang, Feng Zhou et al. (2017). Deep Metric Learning with Angular Loss
- Weiyang Liu, Yandong Wen et al. (2017). SphereFace: Deep Hypersphere Embedding for Face Recognition
- Nils Reimers, Iryna Gurevych. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks
- Ting Chen, Simon Kornblith et al. (2020). SimCLR: A Simple Framework for Contrastive Learning of Visual Representations
- Mikolaj Wiecek, Barbara Rychalska et al. (2021). On the Unreasonable Effectiveness of Centroids in Image Retrieval
- Tianyu Gao, Xingcheng Yao et al. (2021). SimCSE: Simple Contrastive Learning of Sentence Embeddings
- Alec Radford, Jong Wook Kim et al. (2021). Learning Transferable Visual Models From Natural Language Supervision

References(others)

- 2021 VeA Deep Metric Learning
- 시각적 이해를 위한 머신러닝(2023년 봄; 한국어)
- https://quaternion.qdrant.tech/tutorials/triplet_loss_trick

Thank you!