EECS 498/598: Deep Learning
Lecture 7. Embedding Methods

Honglak Lee

02/22/2019

Outline

- **Embedding Methods Basics**
- Case Study 1: Embeddings for Text
- Case Study 2: Embedding for Images
- Case Study 3: Embeddings from Sequential Data
- Case Study 4: Embedding for Multimodal Data

Learning Embedding

Problem definition:

- Given set X of input samples, find a mapping from input to embedding X→ Y such that Y reflects semantics or similarity/neighborhood structures.

$$\mathcal{X} = \{x_1, x_2, \ldots, x_n \in \mathbb{R}^h\} \rightarrow \mathcal{Y} = \{y_1, y_2, \ldots, y_n \in \mathbb{R}^l\}$$

- The embedding may be a parameterized function (e.g., y = f(x) by a neural network). If not, you can directly optimize Y (i.e., all embedding coordinates) corresponding to the entire input dataset X.
- The specific formulation differs depending on the objective function.

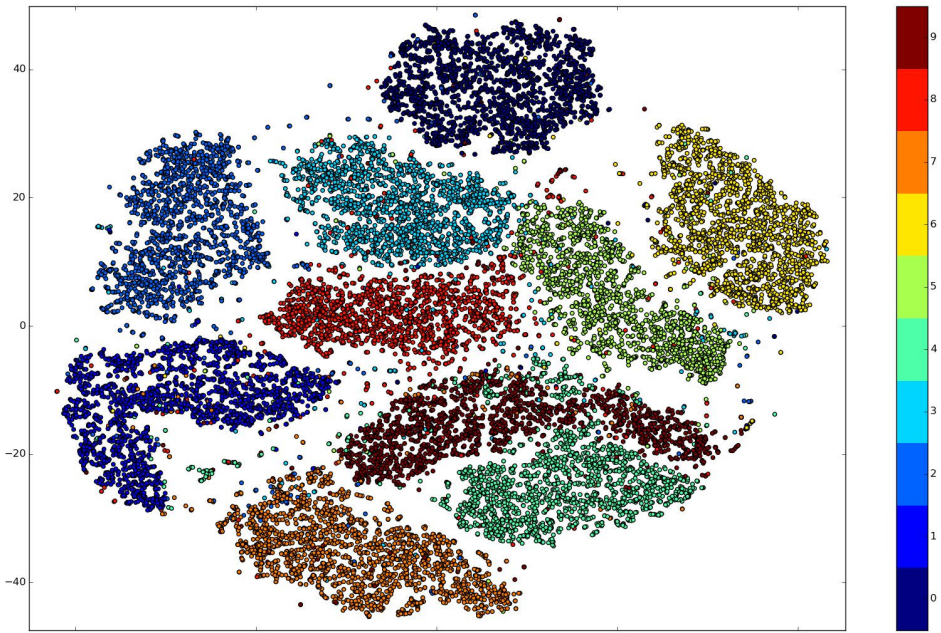
Applications of embedding methods

Embeddings can be useful for

- Data visualization (e.g., tSNE)
- Intermediate representation for downstream tasks; i.e., initializing deep/recurrent neural networks with pretrained embeddings (e.g., word2vec)
- Similarity metric (e.g., Siamese network or joint embedding)

Data visualization with embeddings

- Embeddings are vector representations that reflect semantic meaning in feature space (i.e., feature vectors live in neighborhoods based on meaning)
- We can visualize these with techniques such as t-SNE

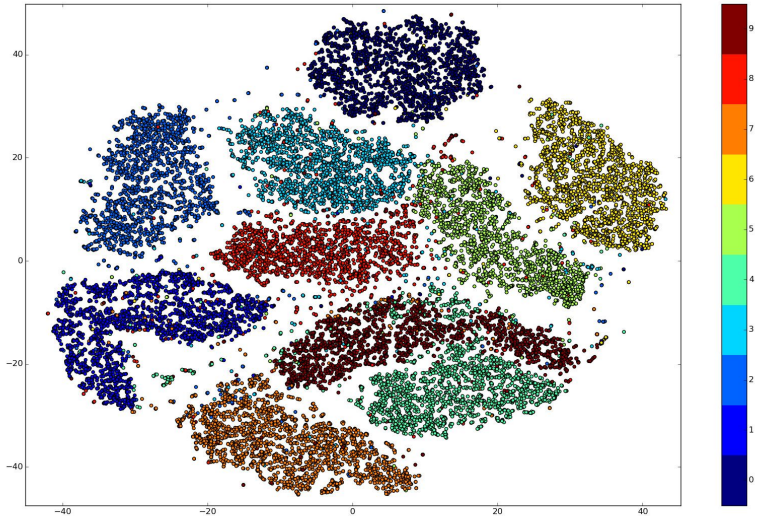


tSNE visualization of handwritten digits (MNIST) in 2d space.

Maaten, L. V. D., & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of machine learning research*, *9*(NovVisualizing data using

t-Distributed Stochastic Neighbor Embedding (t-SNE)

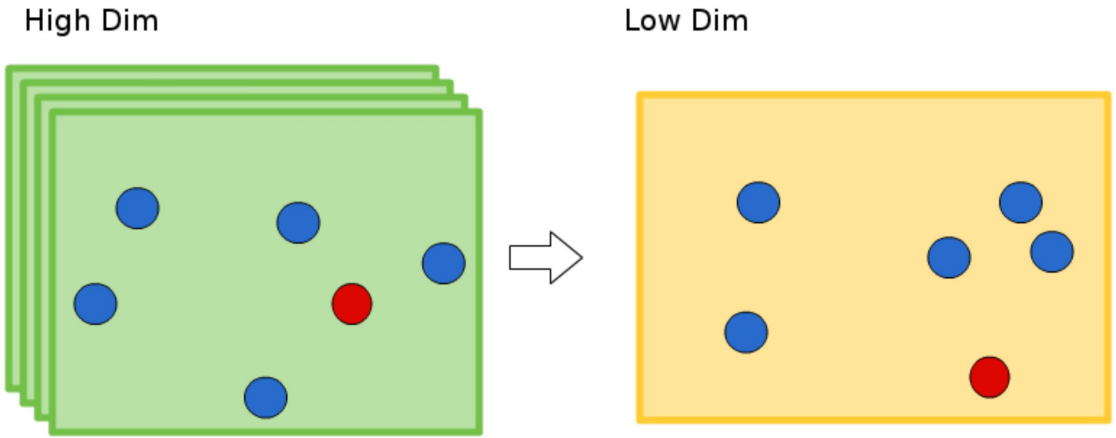
- Given a collection of N high-dimensional data x_1, x_2, … x_n, how can we get a sense of how they are arranged in data space?



Slide credit: Kai-Wen Zhao

t-Distributed Stochastic Neighbor Embedding (t-SNE)

- Compress the high dimensional data into a lower dimensional space
- We want to preserve distance and neighborhood structure



Slide credit: Kai-Wen Zhao

Preliminary: Stochastic Neighbor Embedding (SNE)

SNE converts euclidean distances to similarities, that can be interpreted as probabilities P_i over neighbors of x_i. Then we find embedding Q_i that approximates P_i.

$P_i = \{p_{1|i}, p_{2|i}, ..., p_{n|i}\}$ and $Q_i = \{q_{1|i}, q_{2|i}, ..., q_{n|i}\}$ are the distributions on the neighbors of datapoint i.

$$p_{j|i} = \frac{\exp(- \| x_i - x_j \|^2 / 2\sigma_i^2)}{\sum_{k \neq i} \exp(- \| x_i - x_k \|^2 / 2\sigma_i^2)}$$

$$q_{j|i} = \frac{\exp(- \| y_i - y_j \|^2)}{\sum_{k \neq i} \exp(- \| y_i - y_k \|^2)}$$

$$p_{i|i} = 0, q_{i|i} = 0$$

SNE minimizes the following cost:

$$C = \sum_i KL(P_i || Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$$

Slide credit: Simon Carbonnelle

t-Distributed Stochastic Neighbor Embedding (t-SNE)

SNE

Modelisation:

$$p_{j|i} = \frac{\exp(-\|x_i - x_j\|^2 / 2\sigma_i^2)}{\sum_{k \neq i} \exp(-\|x_i - x_k\|^2 / 2\sigma_i^2)}$$

$$q_{j|i} = \frac{\exp(-\|y_i - y_j\|^2)}{\sum_{k \neq i} \exp(-\|y_i - y_k\|^2)}$$

Cost Function:

$$C = \sum_i KL(P_i || Q_i)$$

Derivatives:

$$\frac{dC}{dy_i} = 2 \sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j)$$

t-Distributed Stochastic Neighbor Embedding (t-SNE)

SNE

Modelisation:

$$p_{j|i} = \frac{\exp(-\|x_i - x_j\|^2/2\sigma_i^2)}{\sum_{k \neq i} \exp(-\|x_i - x_k\|^2/2\sigma_i^2)}$$

$$q_{j|i} = \frac{\exp(-\|y_i - y_j\|^2)}{\sum_{k \neq i} \exp(-\|y_i - y_k\|^2)}$$

Cost Function:

$$C = \sum_i KL(P_i||Q_i)$$

Derivatives:

$$\frac{dC}{dy_i} = 2\sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j)$$

$\Rightarrow$

Symmetric SNE

Modelisation:

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$$

$$q_{ij} = \frac{\exp(-\|y_i - y_j\|^2)}{\sum_{k \neq l} \exp(-\|y_k - y_l\|^2)}$$

Cost Function:

$$C = KL(P||Q)$$

Derivatives:

$$\frac{dC}{dy_i} = 4\sum_j (p_{ij} - q_{ij})(y_i - y_j)$$

$\rightarrow$ Faster optimization!

t-Distributed Stochastic Neighbor Embedding (t-SNE)

SNE ⇒ Symmetric SNE ⇒ t-SNE

Modelisation:

$$p_{j|i} = \frac{\exp(-\|x_i - x_j\|^2/2\sigma_i^2)}{\sum_{k\neq i}\exp(-\|x_i - x_k\|^2/2\sigma_i^2)}$$

$$q_{j|i} = \frac{\exp(-\|y_i - y_j\|^2)}{\sum_{k\neq i}\exp(-\|y_i - y_k\|^2)}$$

Cost Function:

$$C = \sum_i KL(P_i||Q_i)$$

Derivatives:

$$\frac{dC}{dy_i} = 2\sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j)$$

Modelisation:

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$$

$$q_{ij} = \frac{\exp(-\|y_i - y_j\|^2)}{\sum_{k\neq l}\exp(-\|y_k - y_l\|^2)}$$

Cost Function:

$$C = KL(P||Q)$$

Derivatives:

$$\frac{dC}{dy_i} = 4\sum_j (p_{ij} - q_{ij})(y_i - y_j)$$

→ Faster optimization!

Modelisation:

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$$

$$q_{ij} = \frac{(1+\|y_i - y_j\|^2)^{-1}}{\sum_{k\neq l}(1+\|y_k - y_l\|^2)^{-1}}$$

Cost Function:

$$C = KL(P||Q)$$

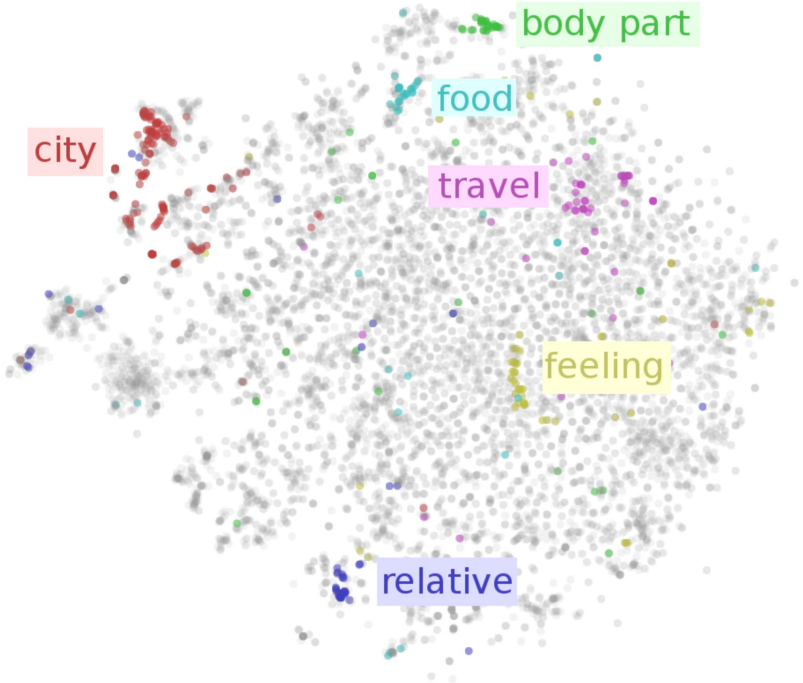
Derivatives:

$$\frac{dC}{dy_i} = 4\sum_j (p_{ij} - q_{ij})(y_i - y_j)(1+\| y_i - y_j \|^2)^{-1}$$

→ Improved robustness
(to noise/uncertainties)

Embedding Methods Basics

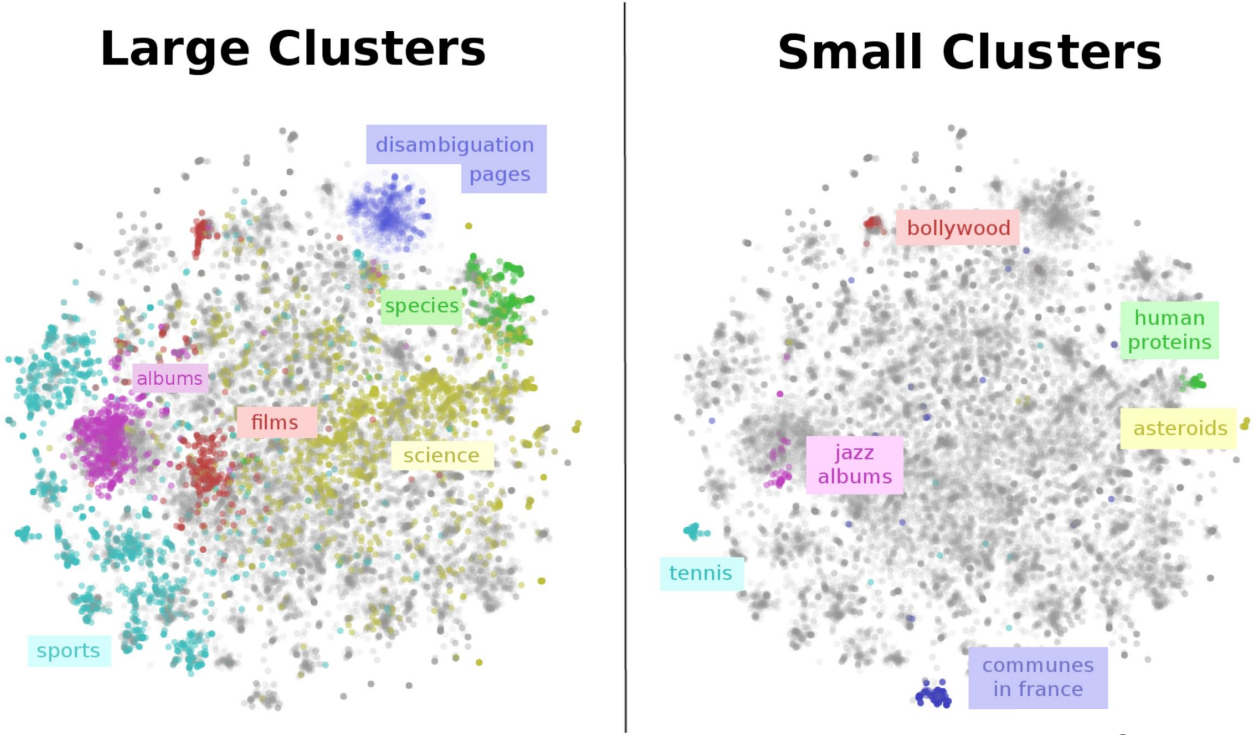
Using t-SNE to explore word embeddings



Slide credit: Simon Carbonelle

Embedding Methods Basics

Using t-SNE to explore wikipedia article embeddings

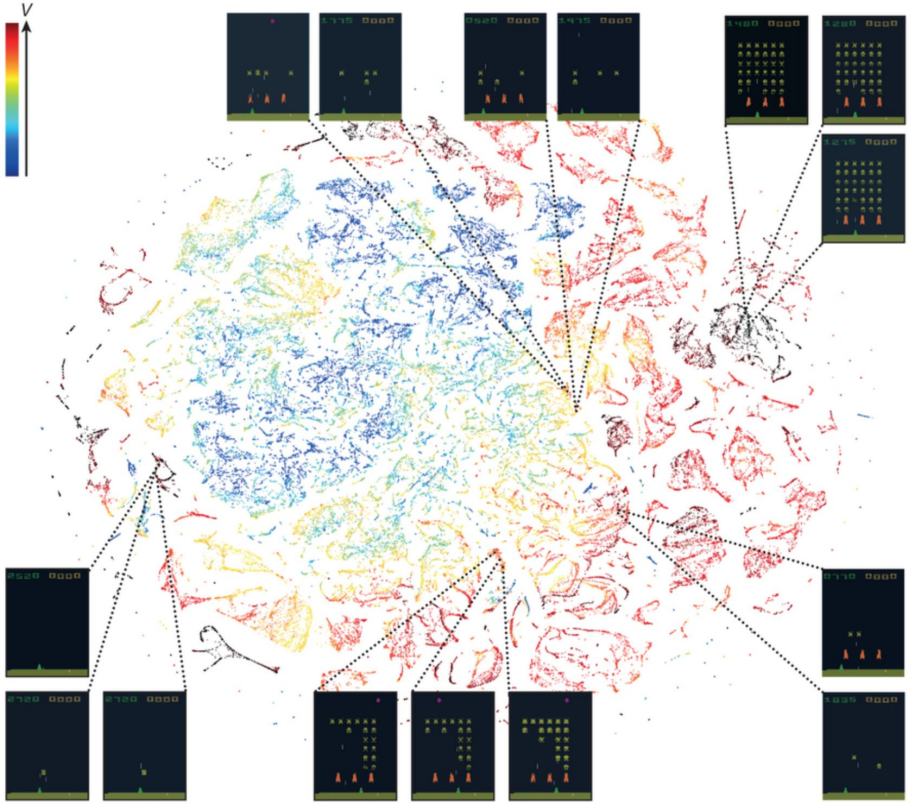


Slide credit: Simon Carbonelle

Embedding Methods Basics

Using t-SNE to explore game state representations

- DQN network that learns to play video games learns embeddings that group game states that look similar



Human-level control through deep reinforcement learning, V. Mnih et Al. (Nature,2015)

Slide credit: Simon Carbonelle

Outline

- Embedding Methods Basics
- **Case Study 1: Embeddings for Text**
- Case Study 2: Embedding for Images
- Case Study 3: Embeddings from Sequential Data
- Case Study 4: Embedding for Multimodal Data

Discrete representation of words

- Words as atomic symbols
  - In vector space terms, this is a vector with one 1 and a lot of zeros

[0 0 0 0 0 0 0 0 1 0 0 0 0 0]

- Dimensionality
  - 20k (speech), 50k (Penn Tree Bank), 500k (Big vocab), 13M (Google 1T)
- We call this a **one-hot** representation.

Its problem:

motel [0 0 0 0 0 0 0 0 1 0 0 0 0 0] AND
hotel [0 0 0 0 0 0 0 0 0 0 0 1 0 0] = 0

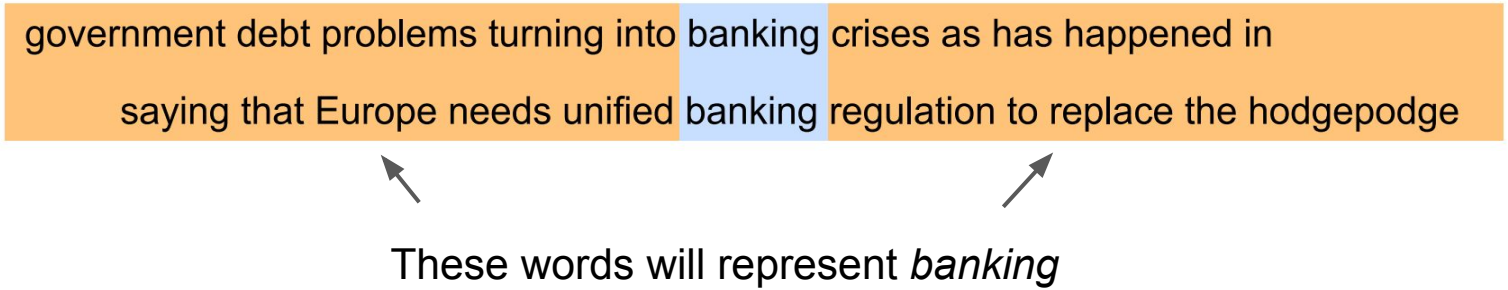
Slide credit: Richard Socher

Distributional similarity based representations

- You can get a lot of value by representing a words by means of its neighbors

  “You shall know a word by the company it keeps” - J. R. Firth 1957: 11

- One of the most successful ideas of modern statistical NLP



Slide credit: Richard Socher

How to make neighbors represent words?

- With a co-occurrence matrix ?
- Problems
  - Increase in size with vocabulary
  - Very high dimensional: require a lot of storage
  - Subsequent classification models have sparsity issues
  - Models are less robust

Example corpus:

- I like deep learning.
- I like NLP.
- I enjoy flying.

counts	I	like	enjoy	deep	learning	NLP	flying	.
I	0	2	1	0	0	0	0	0
like	2	0	0	1	0	1	0	0
enjoy	1	0	0	0	0	0	1	0
deep	0	1	0	0	1	0	0	0
learning	0	0	0	1	0	0	0	1
NLP	0	1	0	0	0	0	0	1
flying	0	0	1	0	0	0	0	1
.	0	0	0	0	1	1	1	0

Slide credit: Richard Socher

How to make neighbors represent words?

- How about low-dimensional vectors ?
  - Idea: store “most” of the important information in a fixed, small number of dimensions: a dense vector
- How to reduce the dimensionality?
  - SVD of co-occurrence matrix
- Problems
  - Computational cost scales cubically for n matrix: $O(n^3)$ flops
  - Bad for millions of words or documents
  - Hard to incorporate new words or documents
  - Different learning regime than other DL models

Slide credit: Richard Socher

Idea: Directly learn low-dimensional word vectors

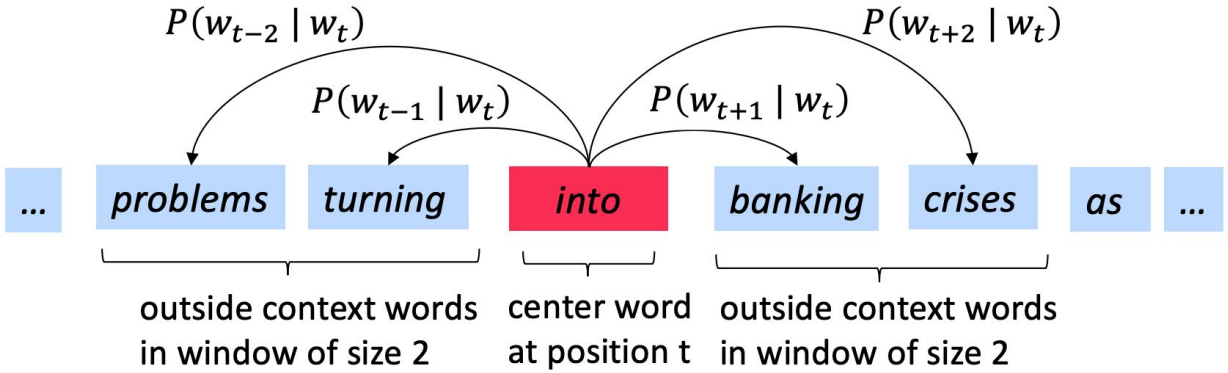
Main idea of Word2Vec

- Instead of capturing co-occurrence counts directly, predict surrounding words of every word (related work: Pennington et al. (2014) and Levy and Goldberg (2014))
- Faster and can easily incorporate a new sentence/document or add a word to the vocabulary

Slide credit: Richard Socher

Main idea of Word2Vec

- Iterate through each word of the whole corpus
- Predict surrounding words using word vectors



$$p(o|c) = \frac{\exp\left(u_o^T v_c\right)}{\sum_{w=1}^{W} \exp\left(u_w^T v_c\right)}$$

- Update vectors so you can predict well

Slide credit: Richard Socher

Details of Word2Vec

- Predict surrounding words in a window of length m of every word
- For $p(w_{t+j}|w_t)$ the simplest first formulation is

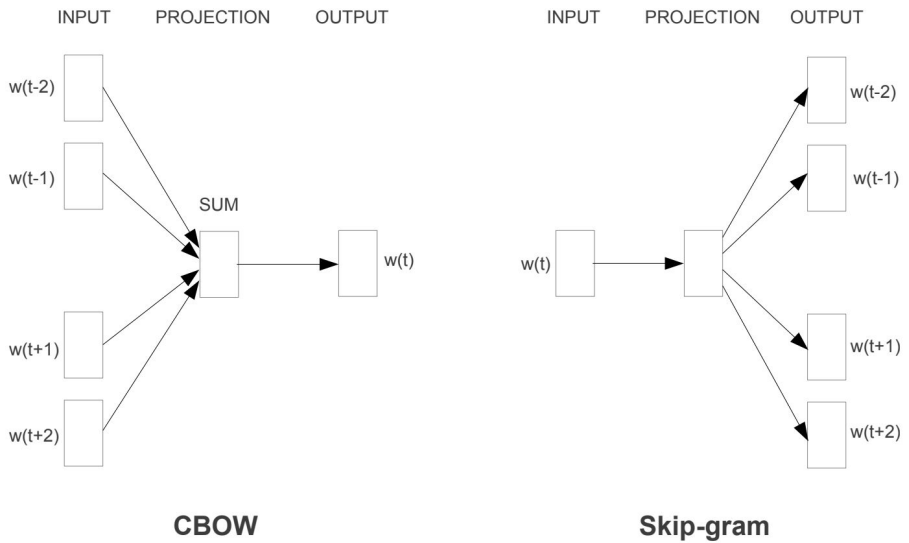
$$p(o|c) = \frac{\exp\left(u_o^T v_c\right)}{\sum_{w=1}^{W} \exp\left(u_w^T v_c\right)}$$

- o is the outside (or output) word id, c is the center word id, u and v are ‘center’ and ‘outside’ vectors of o and c
- Every word has two vectors
- This is essentially ‘dynamic’ logistic regression

Slide credit: Richard Socher

Word2Vec: Two model variants

1. Skip-gram (SG)
   - Predict context (“outside”) words (position independent) given center word
2. Continuous Bag of Words (CBOW)
   - Predict center word from (bag of) context words



[Mikolov et al.,2013, Efficient Estimation of Word Representations in Vector Space]

Slide credit: Richard Socher

Skip-gram with negative sampling

- Normalization factor is too computationally expensive.
- Negative sampling: Train binary logistic regressions for a true pair (center word and word in its context window) versus several noise pairs (center word paired with a random word)
- Training objective:

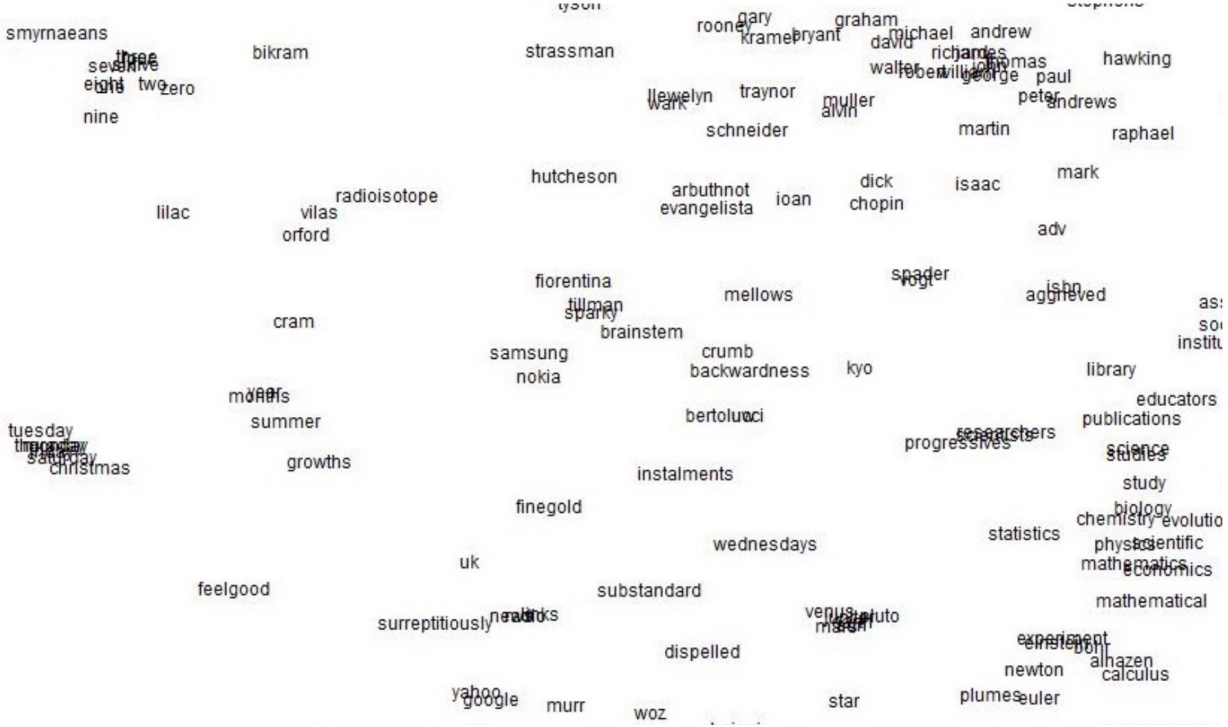
$$J(\theta) = \frac{1}{T}\sum_{t=1}^{T} J_t(\theta)$$

$$J_t(\theta) = \log \sigma\left(u_o^T v_c\right) + \sum_{i=1}^{k} \mathbb{E}_{j\sim P(w)}\left[\log \sigma\left(-u_j^T v_c\right)\right]$$

- P(w) = $U(w)^{3/4}/Z$ where U(w) is the unigram distribution
  - The power makes less frequent words to be sampled more often

Slide credit: Richard Socher

Visualizing word embeddings using t-SNE



Slide credit: Richard Socher

Linear relationships in Word2Vec

These representations are very good at encoding dimensions of similarity!

- Analogies testing dimensions of similarity can be solved quite well just by doing vector subtraction in the embedding space

  - Syntactically

$$x_{apple} - x_{apples} \approx x_{car} - x_{cars} \approx x_{family} - x_{families}$$

  - Semantically

$$x_{shirt} - x_{clothing} \approx x_{chair} - x_{furniture}$$

$$x_{king} - x_{man} \approx x_{queen} - x_{woman}$$

Slide credit: Richard Socher

Word Analogies

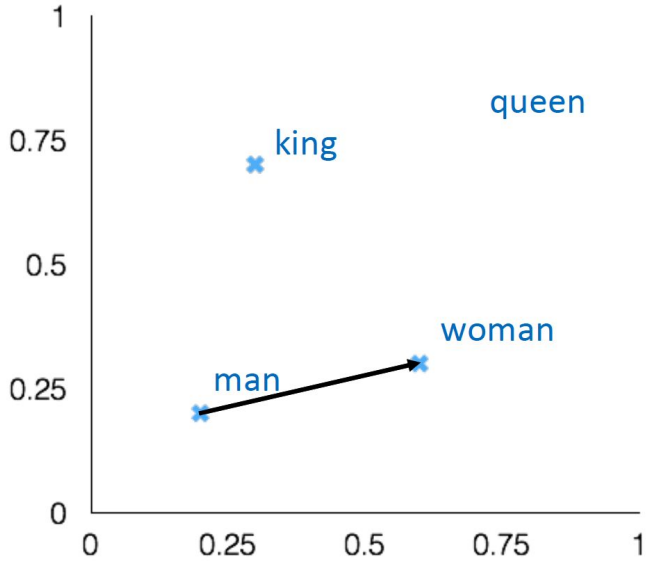
Test for linear relationships, examined by Mikolov et al. (2014)

a:b :: c:?

$$d = \arg\max_x \frac{(w_b - w_a + w_c)^T w_x}{||w_b - w_a + w_c||}$$

man:woman :: king:?

+	king	[0.30 0.70]
-	man	[0.20 0.20]
+	woman	[0.60 0.30]
	queen	[0.70 0.80]



Slide credit: Richard Socher

Word2Vec: Pros and Cons

- Pros
  - \+ Can capture complex patterns beyond word similarity
  - \+ Generate improved performance on other tasks (compared to one-hot representations)

- Cons
  - \- Training time scales with corpus size
  - \- Inefficient usage of statistics

Slide credit: Richard Socher

Learning embedding with word co-occurrences: GloVe

$$J(\theta) = \frac{1}{2} \sum_{i,j=1}^{W} f(P_{ij})(u_i^T v_j - \log P_{ij})^2$$

- Fast training
- Scalable to huge corpora
- Good performance even with small corpus, and small vectors

Slide credit: Richard Socher

GloVe results

Nearest words to frog:

1. frogs
2. toad
3. litoria
4. leptodactylidae
5. rana
6. lizard
7. eleutherodactylus

litoria

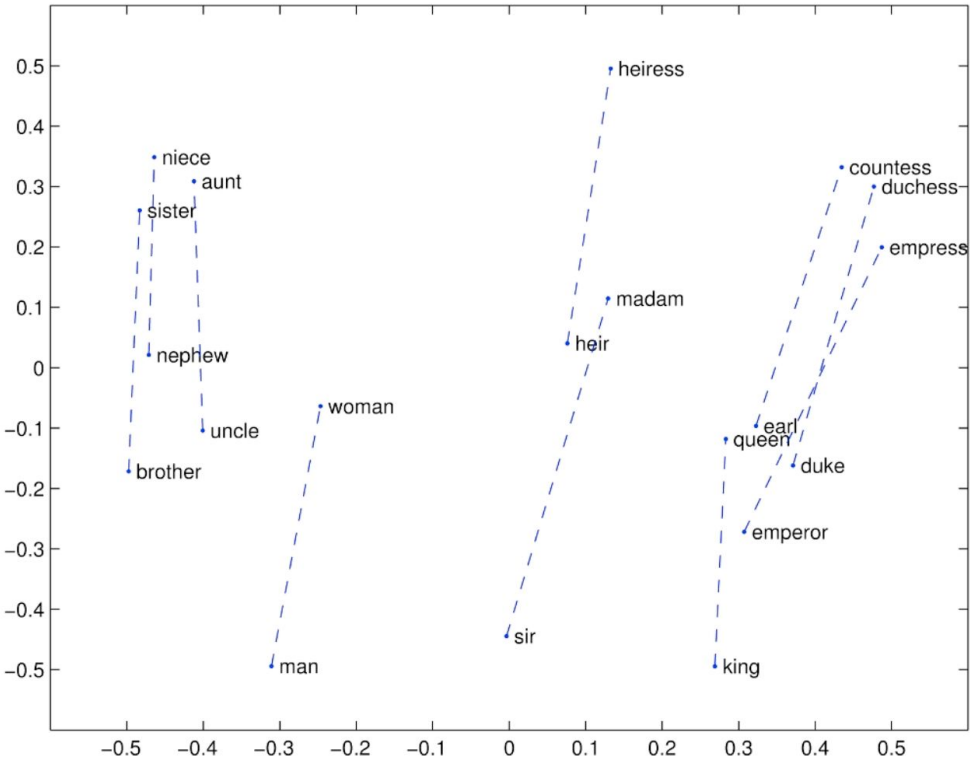
leptodactylidae

rana

eleutherodactylus

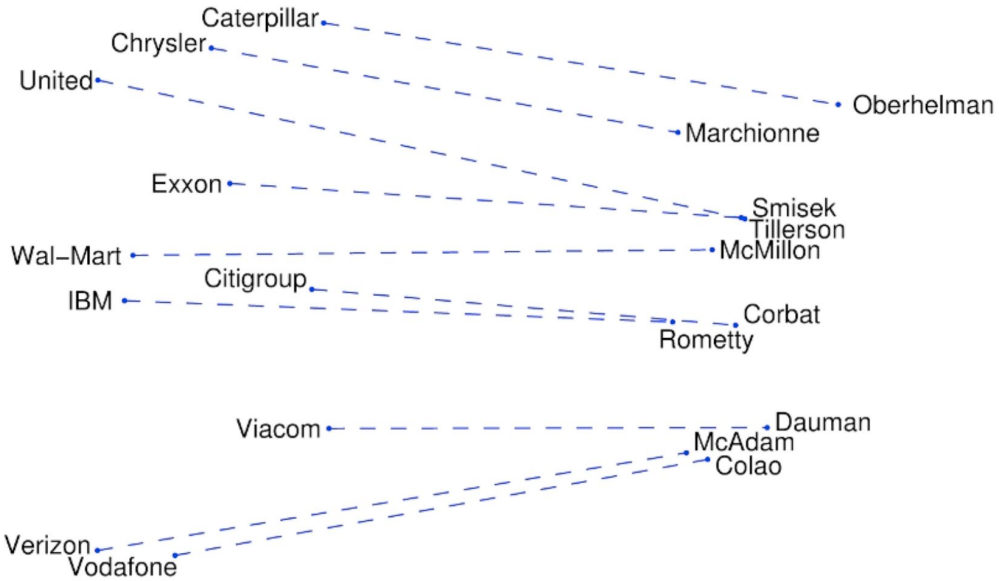
Slide credit: Richard Socher

GloVe visualizations



Slide credit: Richard Socher

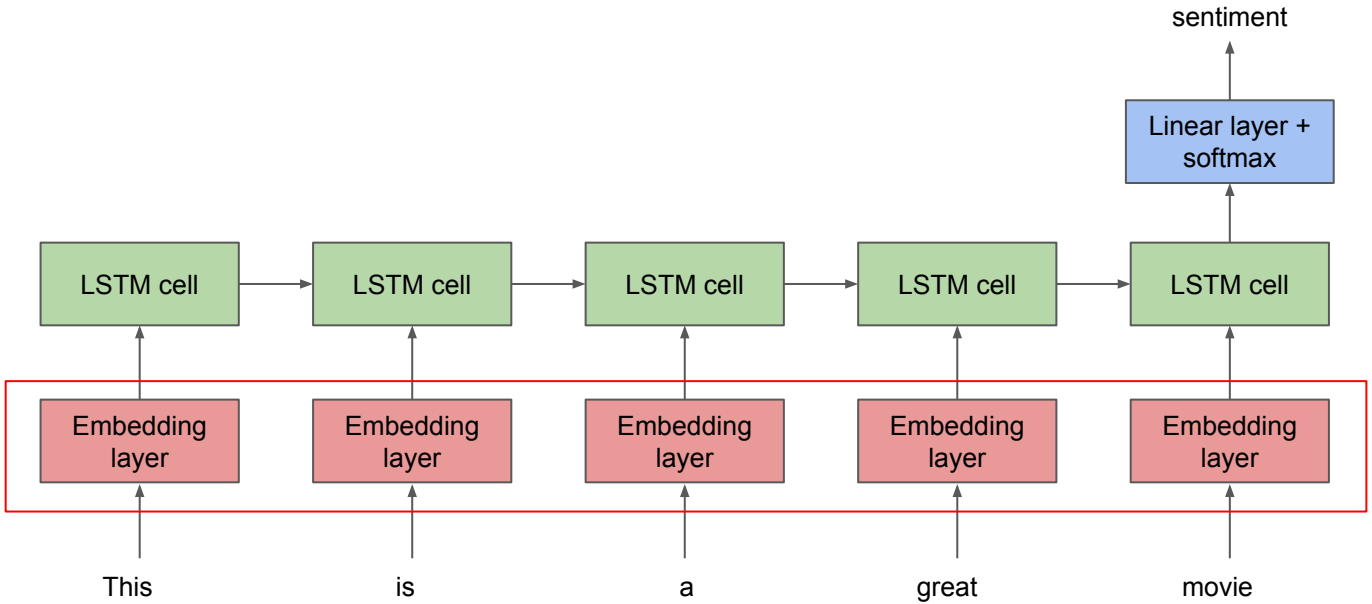
GloVe visualizations: Company - CEO



Slide credit: Richard Socher

Application of word embeddings for downstream tasks

- Pretrained word embedding can be used for initializing LSTM/RNN
- Example: Sentiment Analysis with LSTM
  - Embedding layer can be initialized with pre-trained word embeddings (or trained from scratch).



Outline

- Embedding Methods Basics
- Case Study 1: Embeddings for Text
- **Case Study 2: Embedding for Images**
- Case Study 3: Embeddings from Sequential Data
- Case Study 4: Embedding for Multimodal Data

Learning image similarity

Input: Given a pair of input images, we want to know how “similar” they are to each other.

Output: The output can take a variety of forms:

- Either a binary label, i.e. 0 (same) or 1 (different).
- A Real number indicating how similar a pair of images are.

Learning image similarity

Given a pair of input images, we want to know how “similar” they are to each other.

Application: Face verification

Matched pairs

Mismatched pairs

Huang, G. B., Mattar, M., Berg, T., & Learned-Miller, E. (2008, October). Labeled faces in the wild: A database forstudying face recognition in unconstrained environments. In *Workshop on faces in'Real-Life'Images: detection, alignment, and recognition*.

Learning image similarity

Problem: Given a pair of input images, we want to know how “similar” they are to each other.

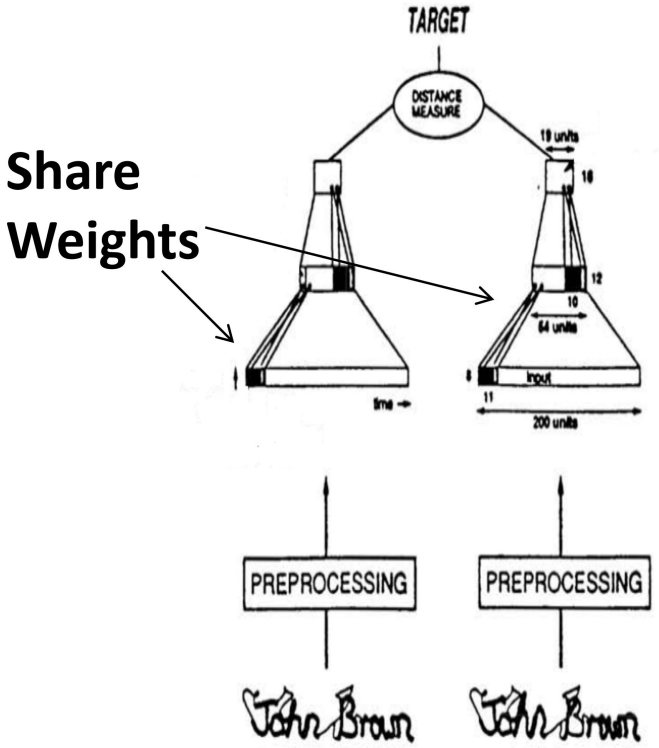
The output can take a variety of forms:

- Either a binary label, i.e. 0 (same) or 1 (different).
- A Real number indicating how similar a pair of images are.

Typical Siamese CNN

Input: A pair of input images (e.g., faces, signatures, etc.)

Output: binary class labels “same” or “different”

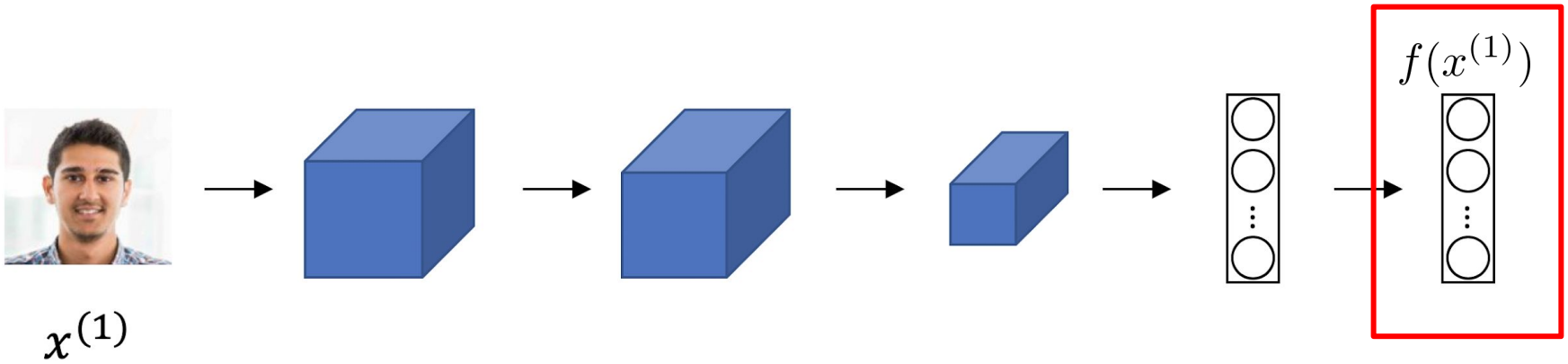


Bromley, J., Bentz, J.W., Bottou, L., Guyon, I., LeCun, Y., Moore, C., Säckinger, E. and Shah, R., 1993. Signature Verification Using A "Siamese" Time Delay Neural Network. IJPRAI, 7(4), pp.669-688.

Slide credit: Moitreya Chatterjee, Yunan Luo

Embeddings for Images

Siamese network

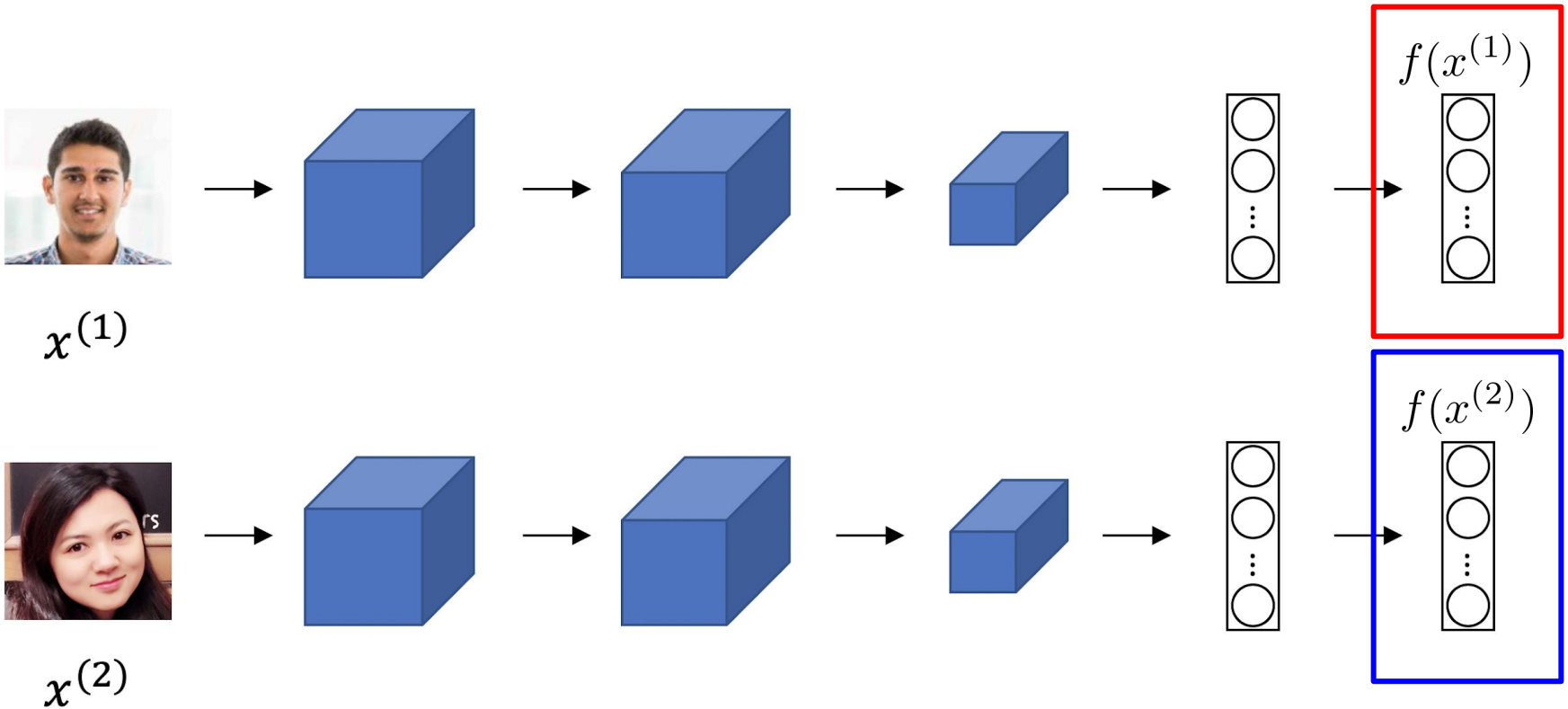


[Taigman et. al., 2014. DeepFace closing the gap to human level performance]

Slide credit: Andrew Ng

Embeddings for Images

Siamese network

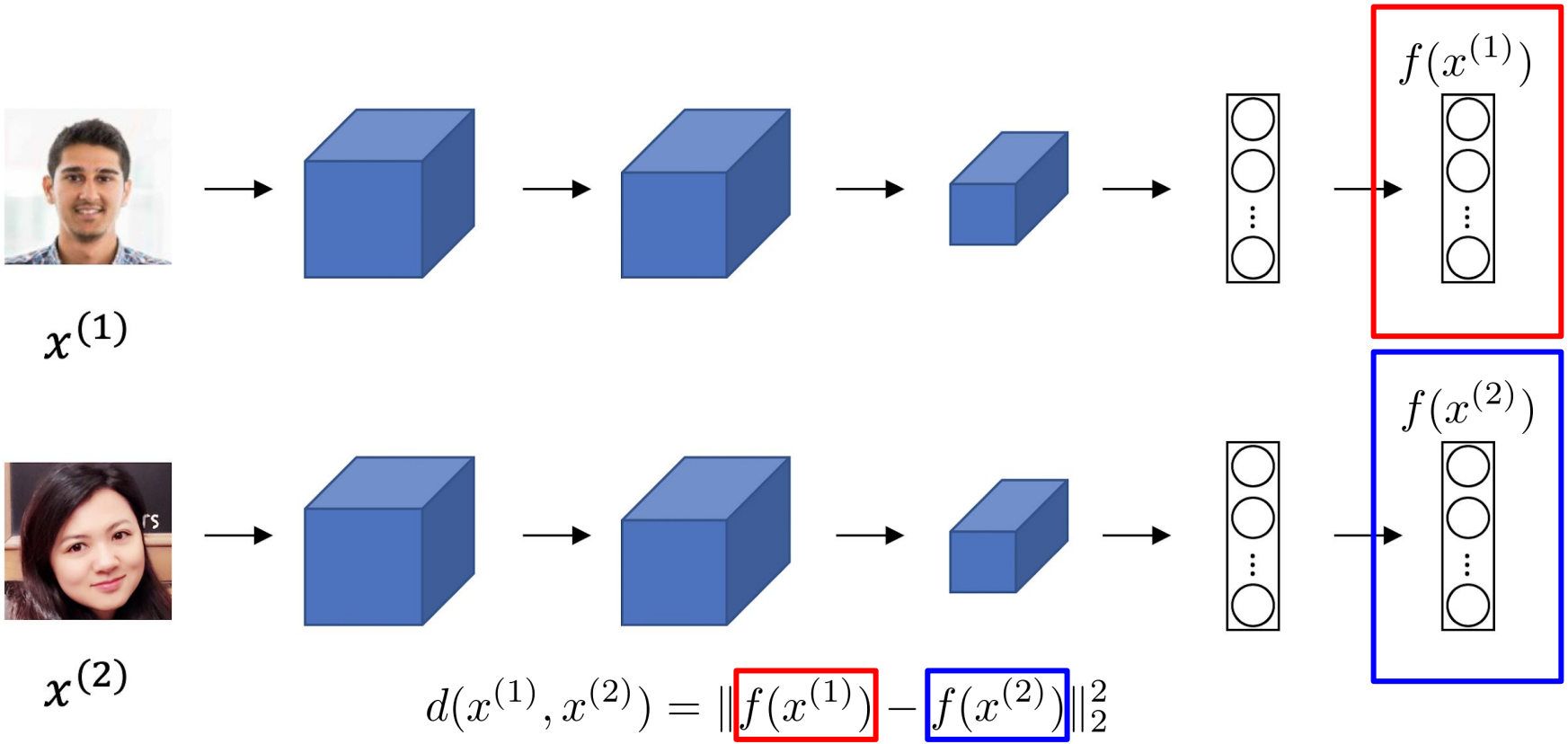


[Taigman et. al., 2014. DeepFace closing the gap to human level performance]

Slide credit: Andrew Ng

Embeddings for Images

Siamese network

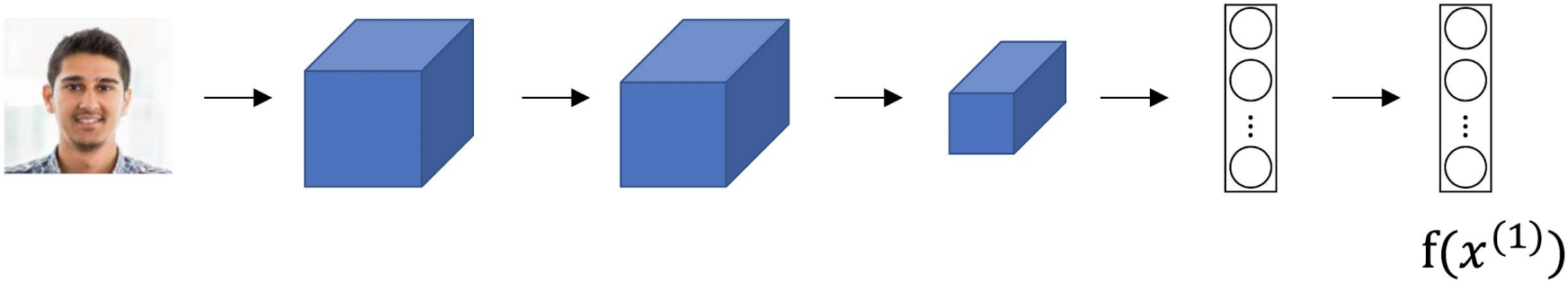


[Taigman et. al., 2014. DeepFace closing the gap to human level performance]

Slide credit: Andrew Ng

Embeddings for Images

Goal of learning

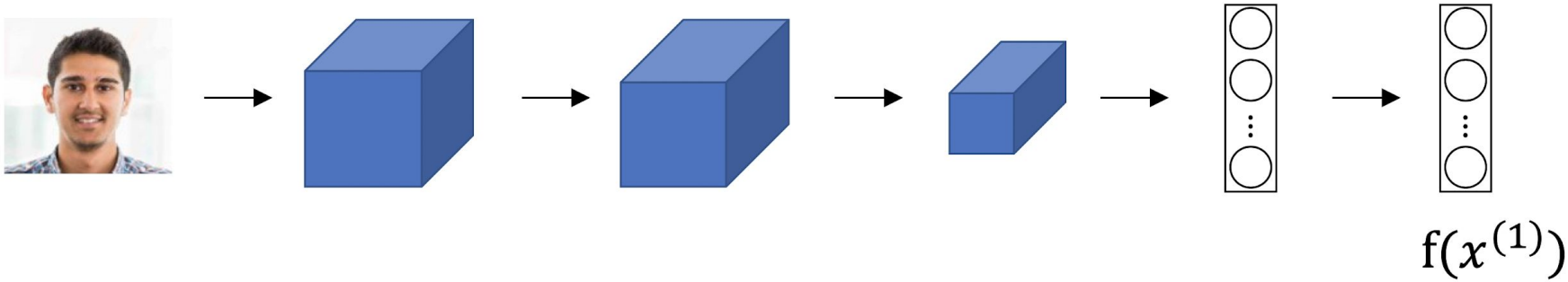


Parameters of NN define an encoding $f(x^{(i)})$

Slide credit: Andrew Ng

Embeddings for Images

Goal of learning



Parameters of NN define an encoding $f(x^{(i)})$

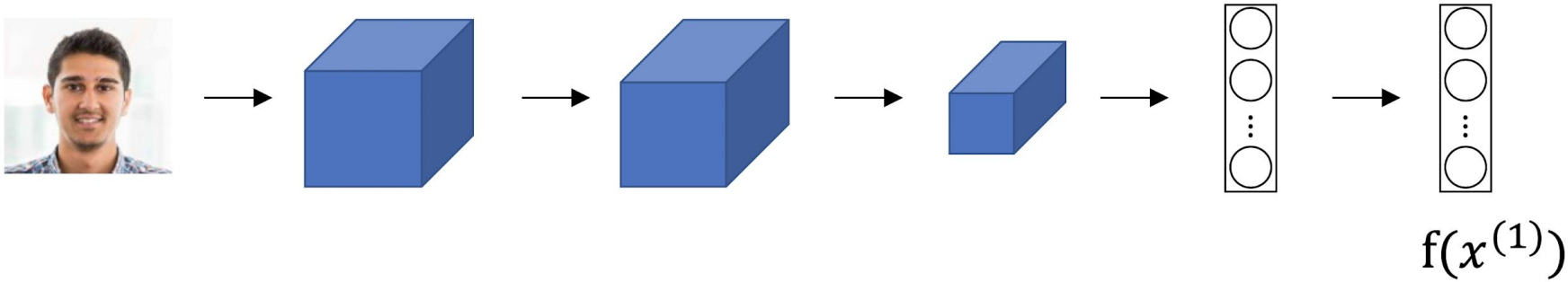
Learn parameters so that:

If $x^{(i)}, x^{(j)}$ are the same person, $\left\|f(x^{(i)}) - f(x^{(j)})\right\|^2$ is small.

Slide credit: Andrew Ng

Embeddings for Images

Goal of learning



Parameters of NN define an encoding $f(x^{(i)})$

Learn parameters so that:

If $x^{(i)}, x^{(j)}$ are the same person, $\left\|f(x^{(i)}) - f(x^{(j)})\right\|^2$ is small.

If $x^{(i)}, x^{(j)}$ are different persons, $\left\|f(x^{(i)}) - f(x^{(j)})\right\|^2$ is large.

Slide credit: Andrew Ng

Embeddings for Images

Triplet loss

Anchor

Positive

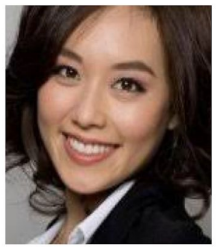
[Schroff et al.,2015, FaceNet: A unified embedding for face recognition and clustering]

Slide credit: Andrew Ng

Embeddings for Images

Triplet loss

Anchor



Positive

Anchor

Negative

[Schroff et al.,2015, FaceNet: A unified embedding for face recognition and clustering]

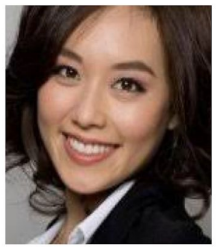
Slide credit: Andrew Ng

Embeddings for Images

Triplet loss

Anchor

A



Positive

P

Anchor

A

Negative

N

[Schroff et al.,2015, FaceNet: A unified embedding for face recognition and clustering]

Slide credit: Andrew Ng

Embeddings for Images

Triplet loss

Anchor
A

Positive
P

$$\underbrace{\|f(A) - f(P)\|_2^2}_{d(A,P)}$$

Anchor
A

Negative
N

[Schroff et al.,2015, FaceNet: A unified embedding for face recognition and clustering]

Slide credit: Andrew Ng

Embeddings for Images

Triplet loss




Anchor A Positive P Anchor A Negative N

$$\underbrace{\|f(A) - f(P)\|_2^2}_{d(A,P)} + \alpha \leq \underbrace{\|f(A) - f(N)\|_2^2}_{d(A,N)}$$

[Schroff et al.,2015, FaceNet: A unified embedding for face recognition and clustering]

Slide credit: Andrew Ng

Embeddings for Images

Triplet loss




Anchor A Positive P Anchor A Negative N

$$\underbrace{\|f(A) - f(P)\|_2^2}_{d(A,P)} + \alpha \leq \underbrace{\|f(A) - f(N)\|_2^2}_{d(A,N)}$$

margin

$$\|f(A) - f(P)\|_2^2 - \|f(A) - f(N)\|_2^2 + \alpha \leq 0$$

[Schroff et al.,2015, FaceNet: A unified embedding for face recognition and clustering]

Slide credit: Andrew Ng

Embeddings for Images

Triplet loss: Training

- Given 3 images A, P, N

$$\mathcal{L}(A, P, N) = \max(\|f(A) - f(P)\|_2^2 - \|f(A) - f(N)\|_2^2 + \alpha, 0)$$

[Schroff et al.,2015, FaceNet: A unified embedding for face recognition and clustering]

Slide credit: Andrew Ng

Embeddings for Images

Triplet loss: Training

- Given 3 images A, P, N

$$\mathcal{L}(A, P, N) = \max(\|f(A) - f(P)\|_2^2 - \|f(A) - f(N)\|_2^2 + \alpha, 0)$$

[Schroff et al.,2015, FaceNet: A unified embedding for face recognition and clustering]

Slide credit: Andrew Ng

Embeddings for Images

Triplet loss: Training

- Given 3 images A, P, N

$$\mathcal{L}(A, P, N) = \max(\|f(A) - f(P)\|_2^2 - \|f(A) - f(N)\|_2^2 + \alpha, 0)$$

$$J = \sum_{i=1}^{m} \mathcal{L}(A^{(i)}, P^{(i)}, N^{(i)})$$

- Training set: 10k pictures of 1k persons

[Schroff et al.,2015, FaceNet: A unified embedding for face recognition and clustering]

Slide credit: Andrew Ng

Embeddings for Images

Triplet loss: Choosing the triplets

- During training, if A, P, N are chosen randomly, $d(A, P) + \alpha \leq d(A, N)$ is easily satisfied.

[Schroff et al.,2015, FaceNet: A unified embedding for face recognition and clustering]

Slide credit: Andrew Ng

Embeddings for Images

Triplet loss: Choosing the triplets

- During training, if A, P, N are chosen randomly, $d(A, P) + \alpha \leq d(A, N)$ is easily satisfied.

$$\|f(A) - f(P)\|_2^2 + \alpha \leq \|f(A) - f(N)\|_2^2$$

[Schroff et al.,2015, FaceNet: A unified embedding for face recognition and clustering]

Slide credit: Andrew Ng

Embeddings for Images

Triplet loss: Choosing the triplets

- During training, if A, P, N are chosen randomly, *d(A, P)* + $\alpha \leq$ *d(A, N)* is easily satisfied.

$$\|f(A) - f(P)\|_2^2 + \alpha \leq \|f(A) - f(N)\|_2^2$$

[Schroff et al.,2015, FaceNet: A unified embedding for face recognition and clustering]

Slide credit: Andrew Ng

Embeddings for Images

Triplet loss: Choosing the triplets

- During training, if A, P, N are chosen randomly, *d(A, P)* + α ≤ *d(A, N)* is easily satisfied.

$$\|f(A) - f(P)\|_2^2 + \alpha \leq \|f(A) - f(N)\|_2^2$$

- Choose triplets that are "hard" to train on.

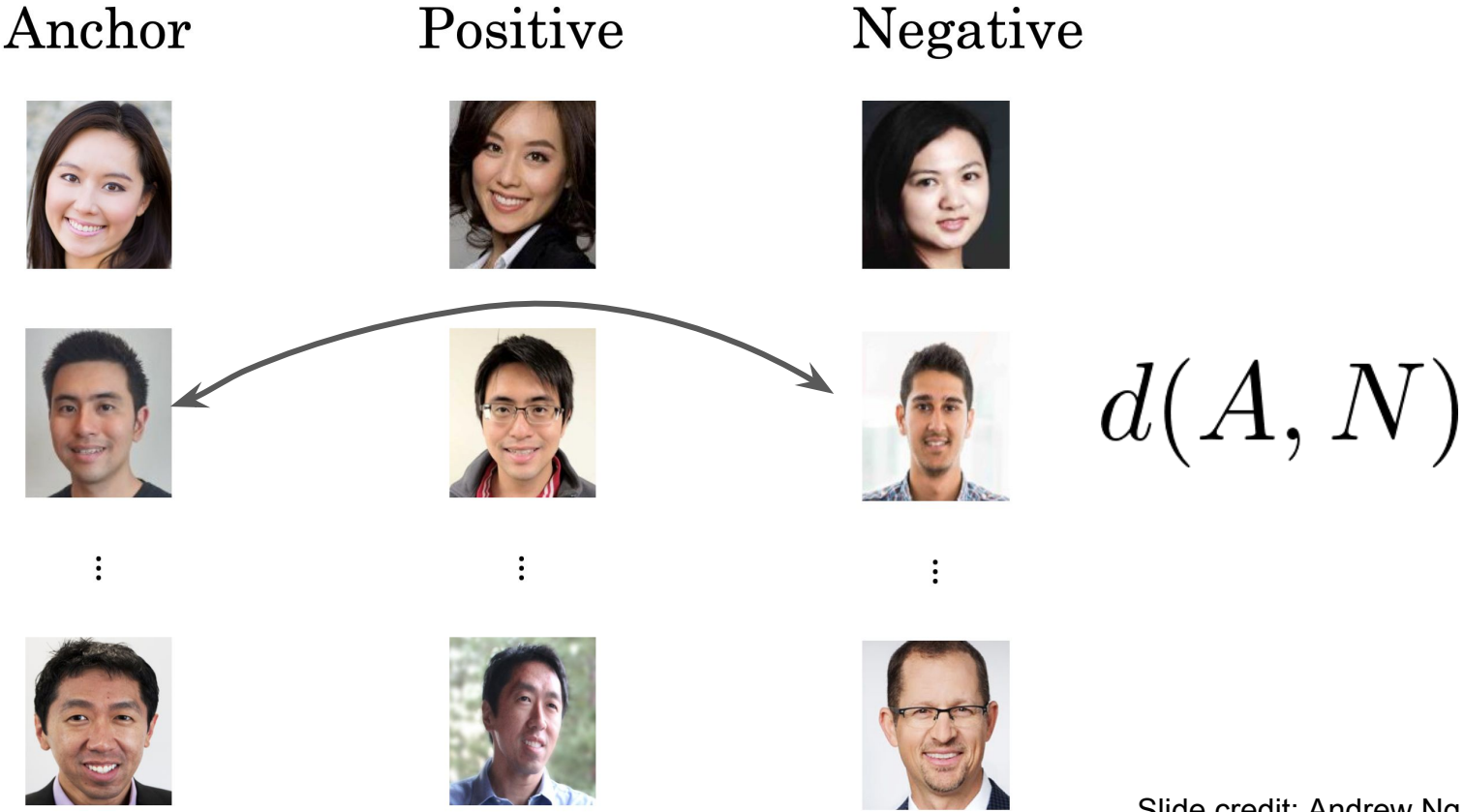
$$d(A, P) + \alpha \leq d(A, N)$$
$$d(A, P) \approx d(A, N)$$

[Schroff et al.,2015, FaceNet: A unified embedding for face recognition and clustering]

Slide credit: Andrew Ng

Embeddings for Images

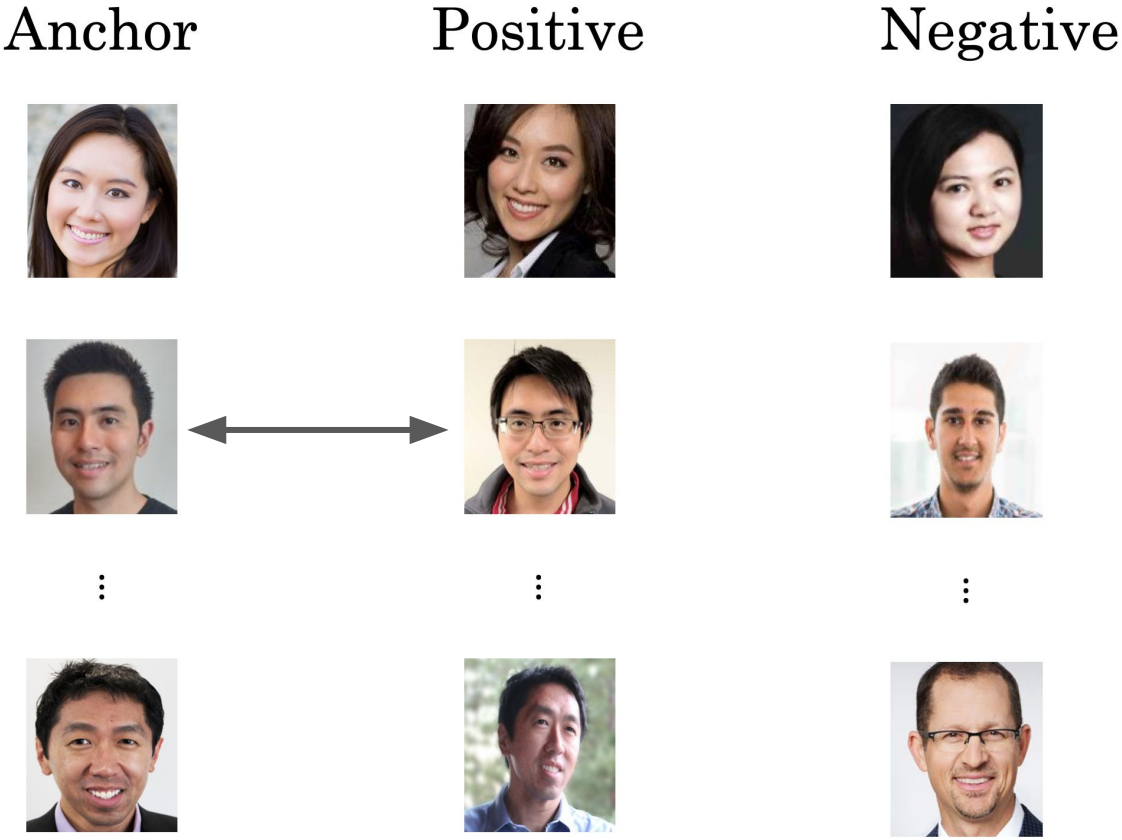
Triplet loss: Training using triplets



Slide credit: Andrew Ng

Embeddings for Images

Triplet loss: Training using triplets



Slide credit: Andrew Ng

Outline

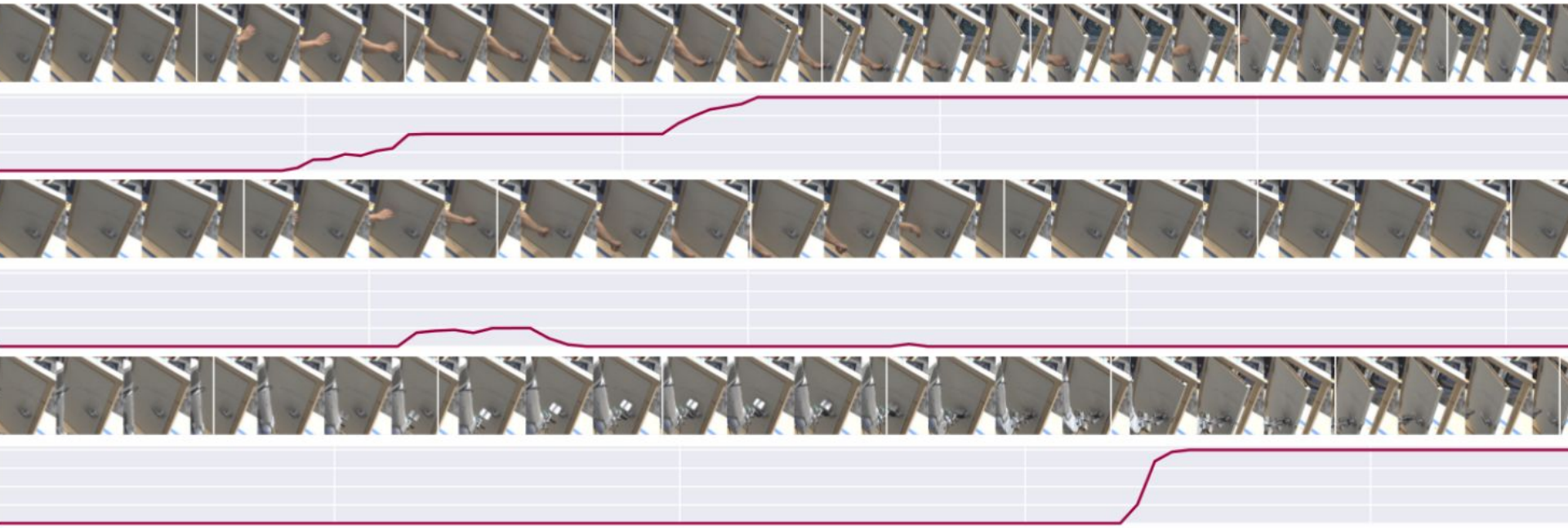
- Embedding Methods Basics
- Case Study 1: Embeddings for Text
- Case Study 2: Embedding for Images
- **Case Study 3: Embeddings from Sequential Data**
- Case Study 4: Embedding for Multimodal Data

Challenges: kinesthetic demonstrations

[Sermanet, Xu, Levine RSS'17] "Unsupervised Perceptual Rewards" arxiv.org/abs/1612.06699
sermanet.github.io/rewards

Slide credit: Pierre Sermanet

Unsupervised Perceptual Rewards



Can we learning embedding of images/videos that can be useful for robotic tasks?

Slide credit: Pierre Sermanet

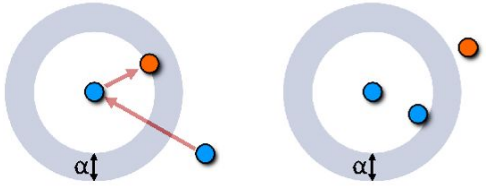
Multi-view videos

How can we learn a view-invariant embedding of demonstrations/actions?

Slide credit: Pierre Sermanet

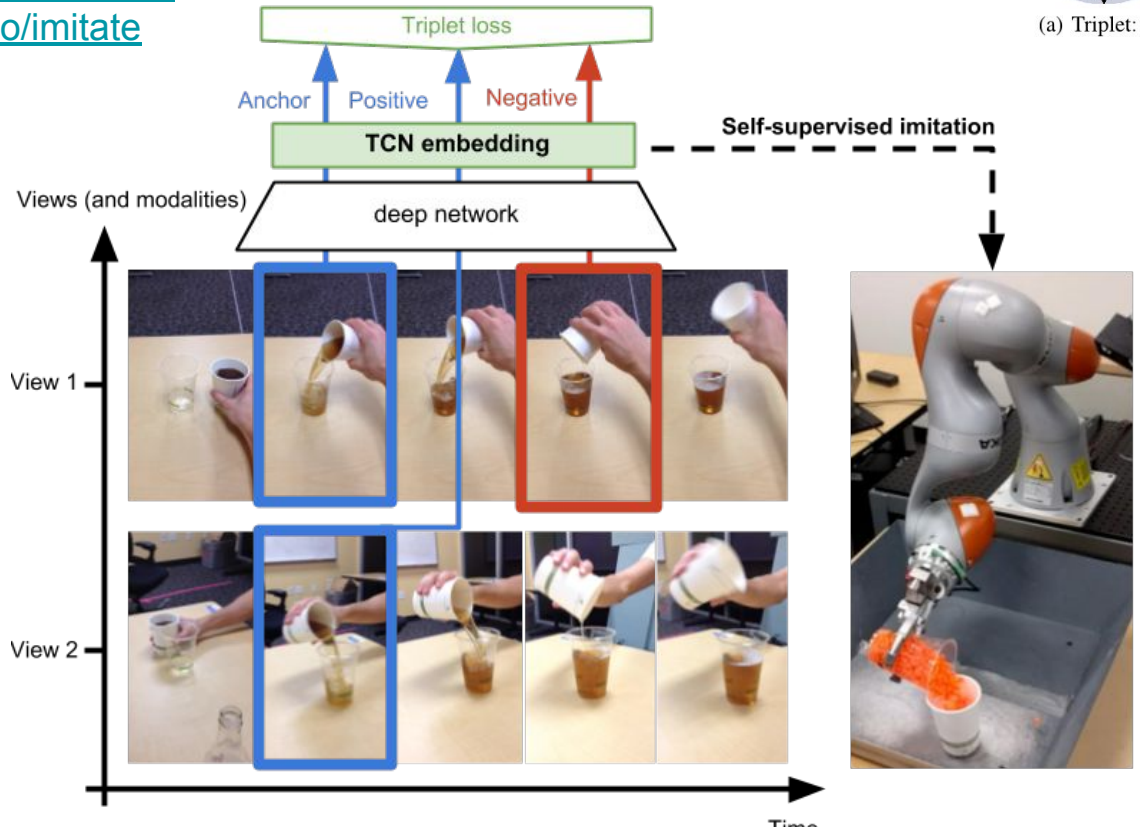
Time-Contrastive Networks (TCN)

arxiv.org/abs/1704.06888v2
sermanet.github.io/imitate



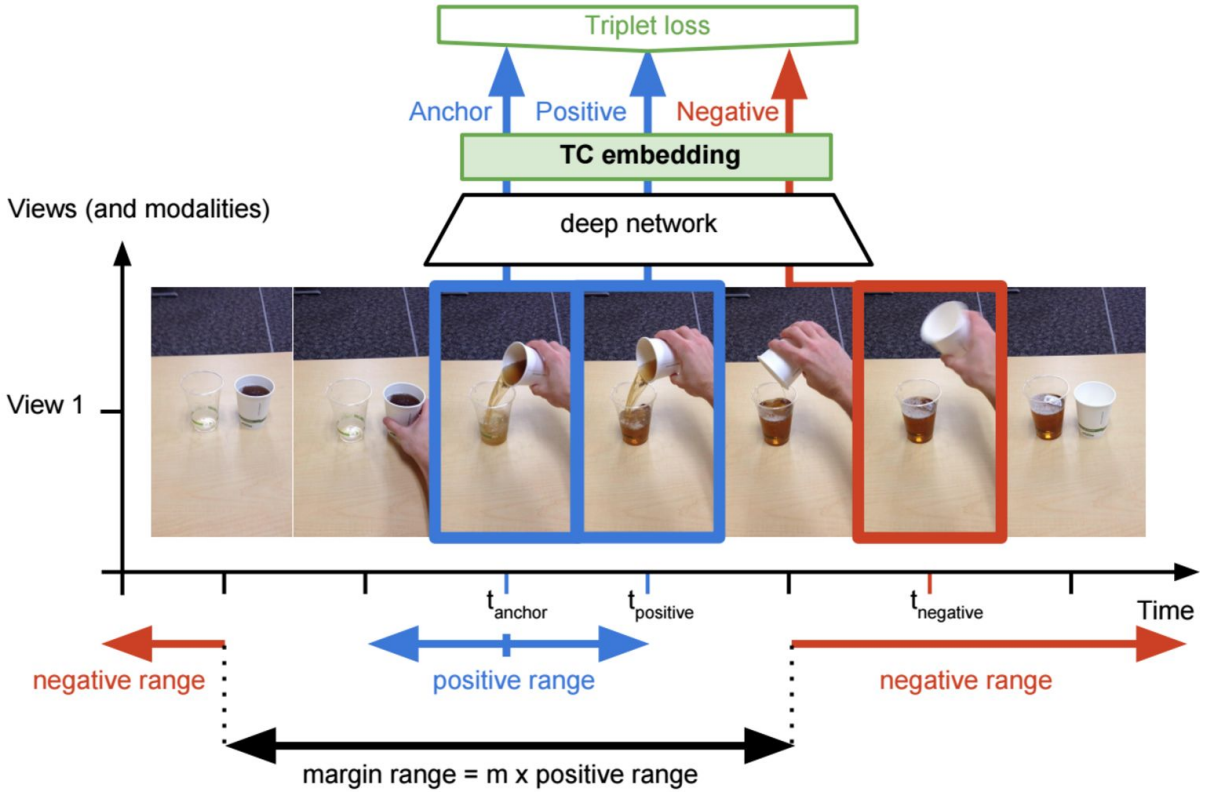
(a) Triplet: before. (b) Triplet: after.

(source: [Rippel et al 2015])



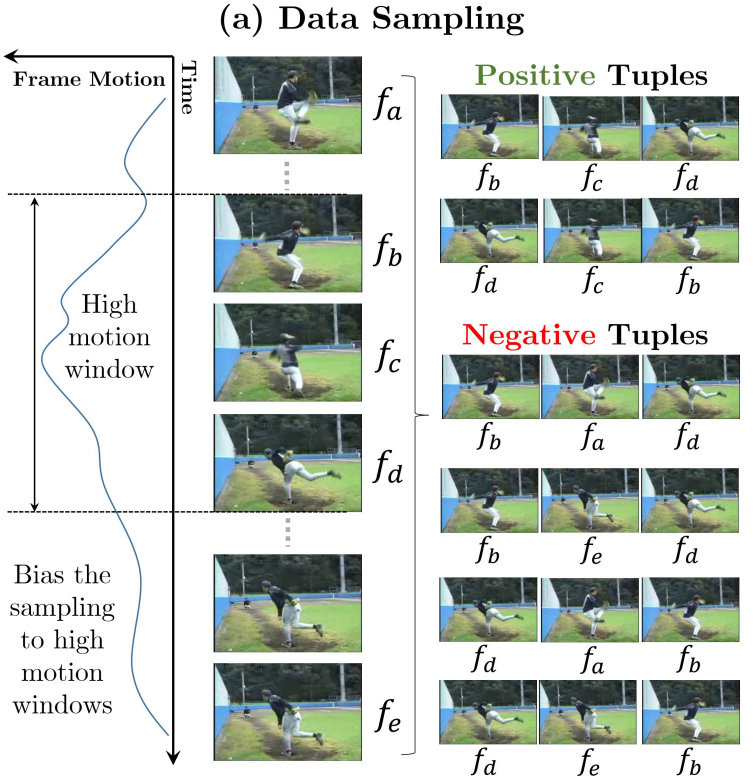
Slide credit: Pierre Sermanet

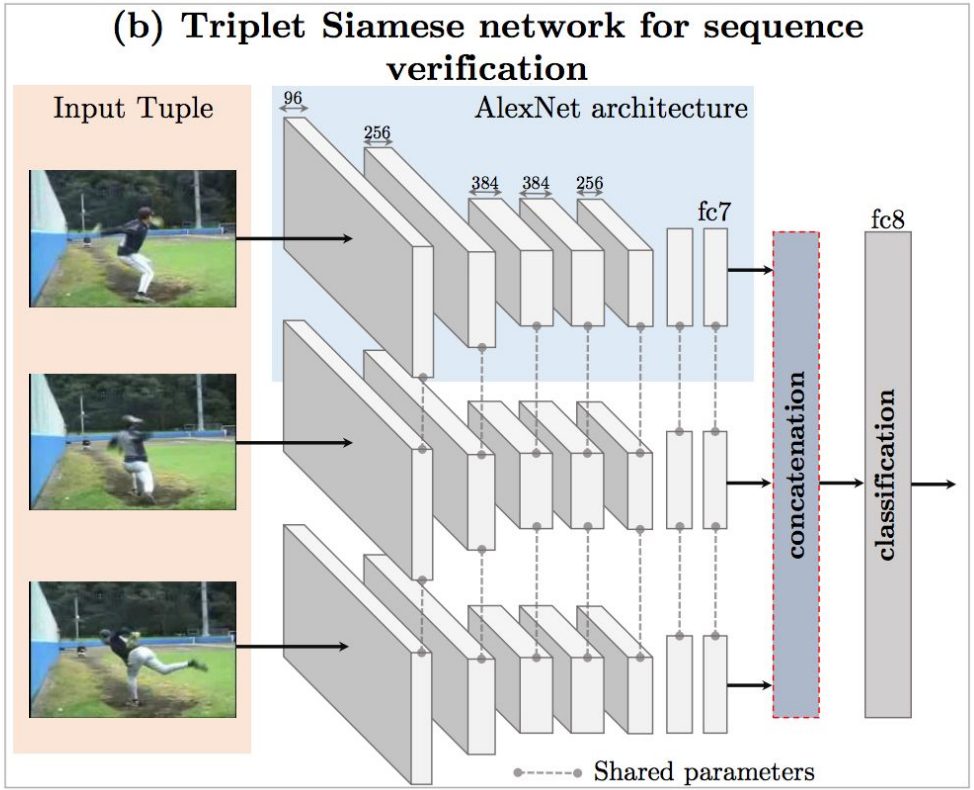
Single-View TCN



Slide credit: Pierre Sermanet

Related Work: Shuffle&Learn [Misra et al, 16]





Shuffle and Learn: Unsupervised Learning using Temporal Order Verification Ishan Misra, C. Lawrence Zitnick, Martial Hebert, 2016

Slide credit: Pierre Sermanet

Nearest Neighbor Imitation

3rd-person | Imitating | 1st-person | Nearest Neighbors

Observation | **Multi-view TCN** | **Single-view TCN** | **Shuffle & Learn [Misra et al. 16]**

Slide credit: Pierre Sermanet

Nearest Neighbor Imitation

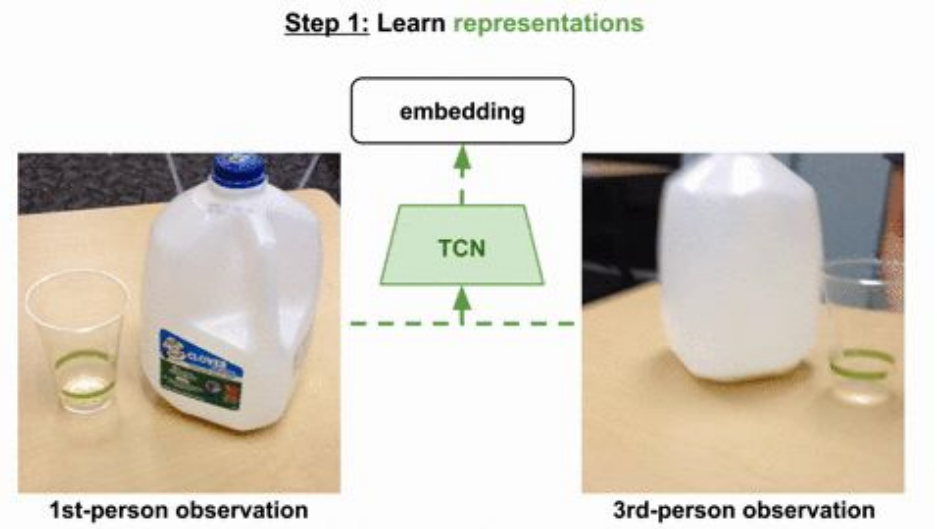
3rd-person ← Imitating → 1st-person

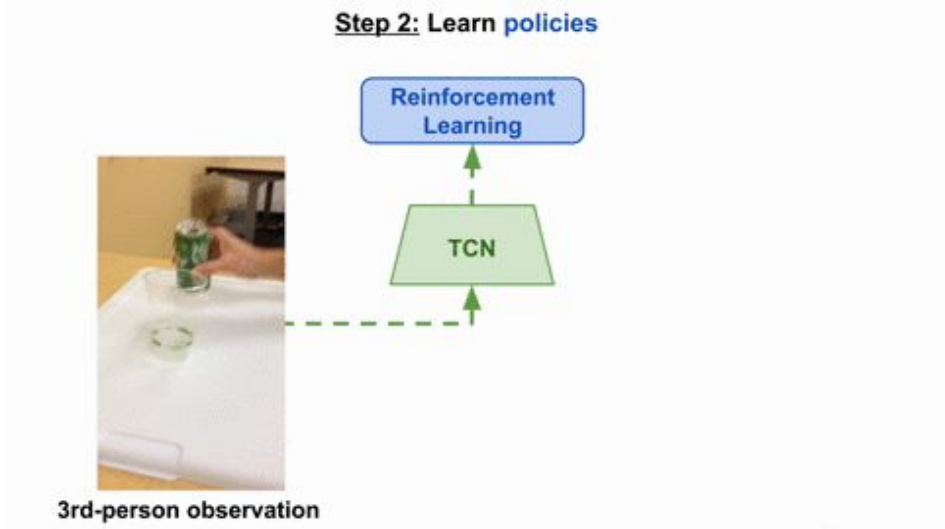
Nearest Neighbors

Observation | **Multi-view TCN** | **Single-view TCN** | **Shuffle & Learn [Misra et al. 16]**

Slide credit: Pierre Sermanet

Approach (pouring, real)





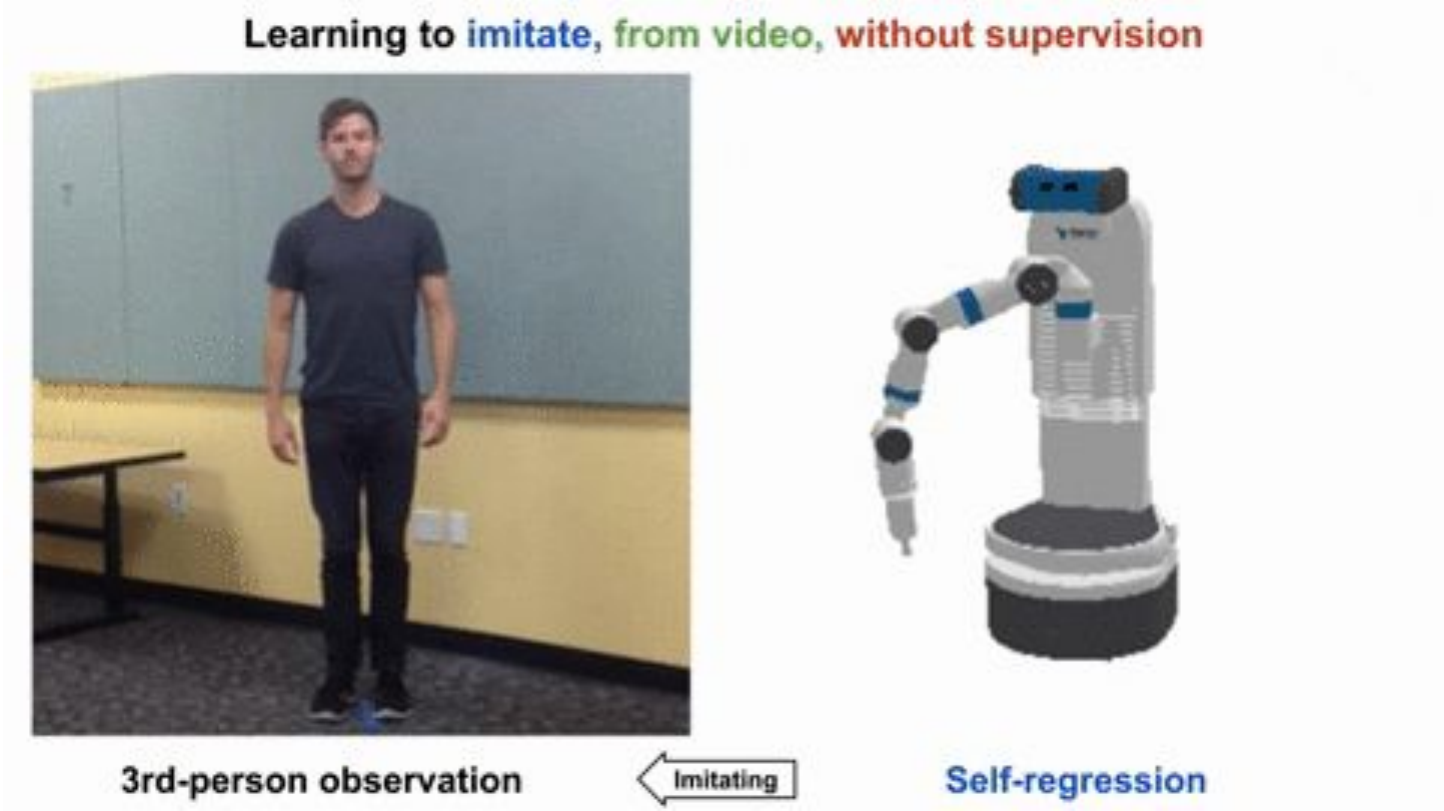
*** RL used: PILQR** Combining Model-Based and Model-Free Updates for Trajectory-Centric Reinforcement Learning, Chebotar, Y., Hausman, K., Zhang, M., Sukhatme, G., Schaal, S., Levine, S. [ICML 17]

Slide credit: Pierre Sermanet

Resulting policies

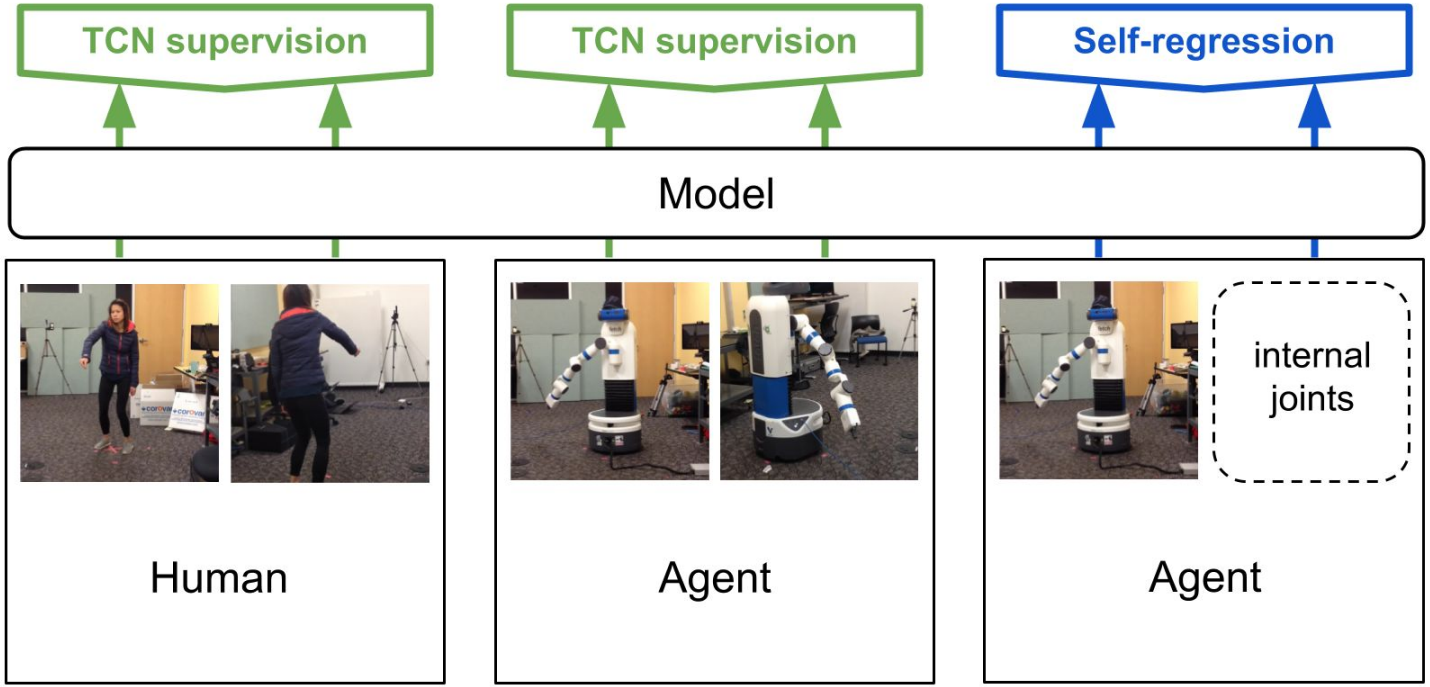
Slide credit: Pierre Sermanet

Pose imitation



Slide credit: Pierre Sermanet

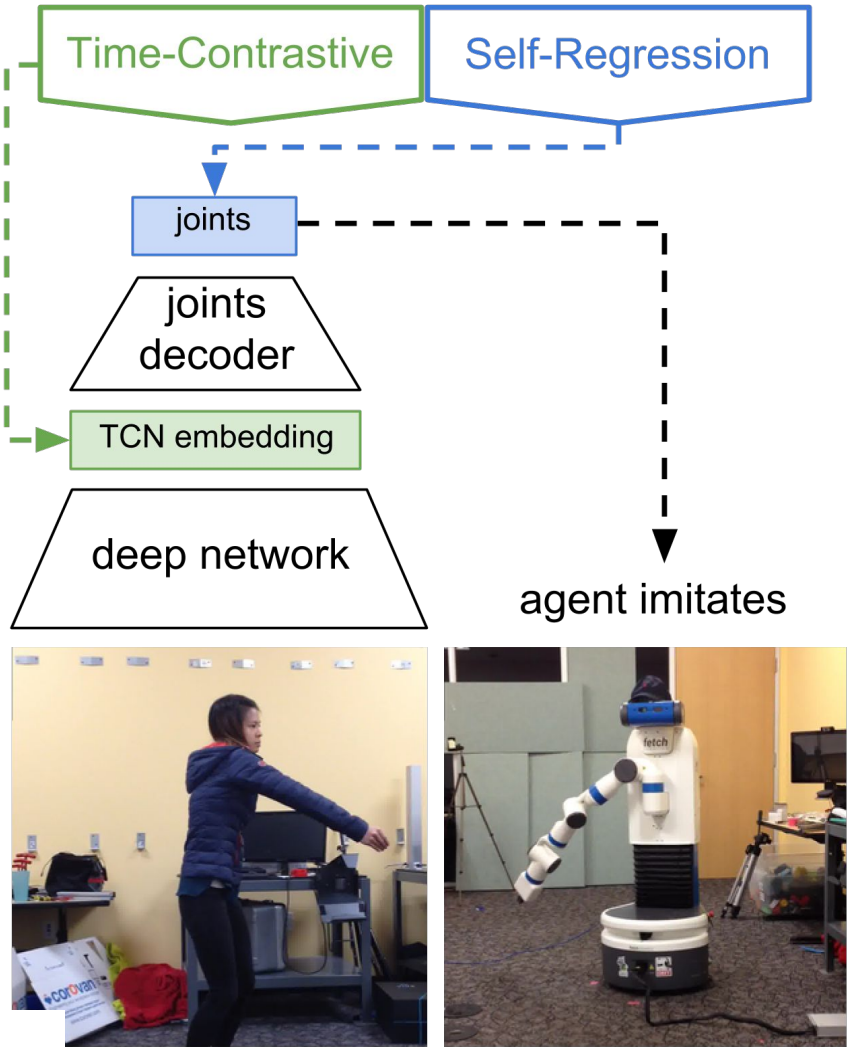
Self-supervision signals



Slide credit: Pierre Sermanet

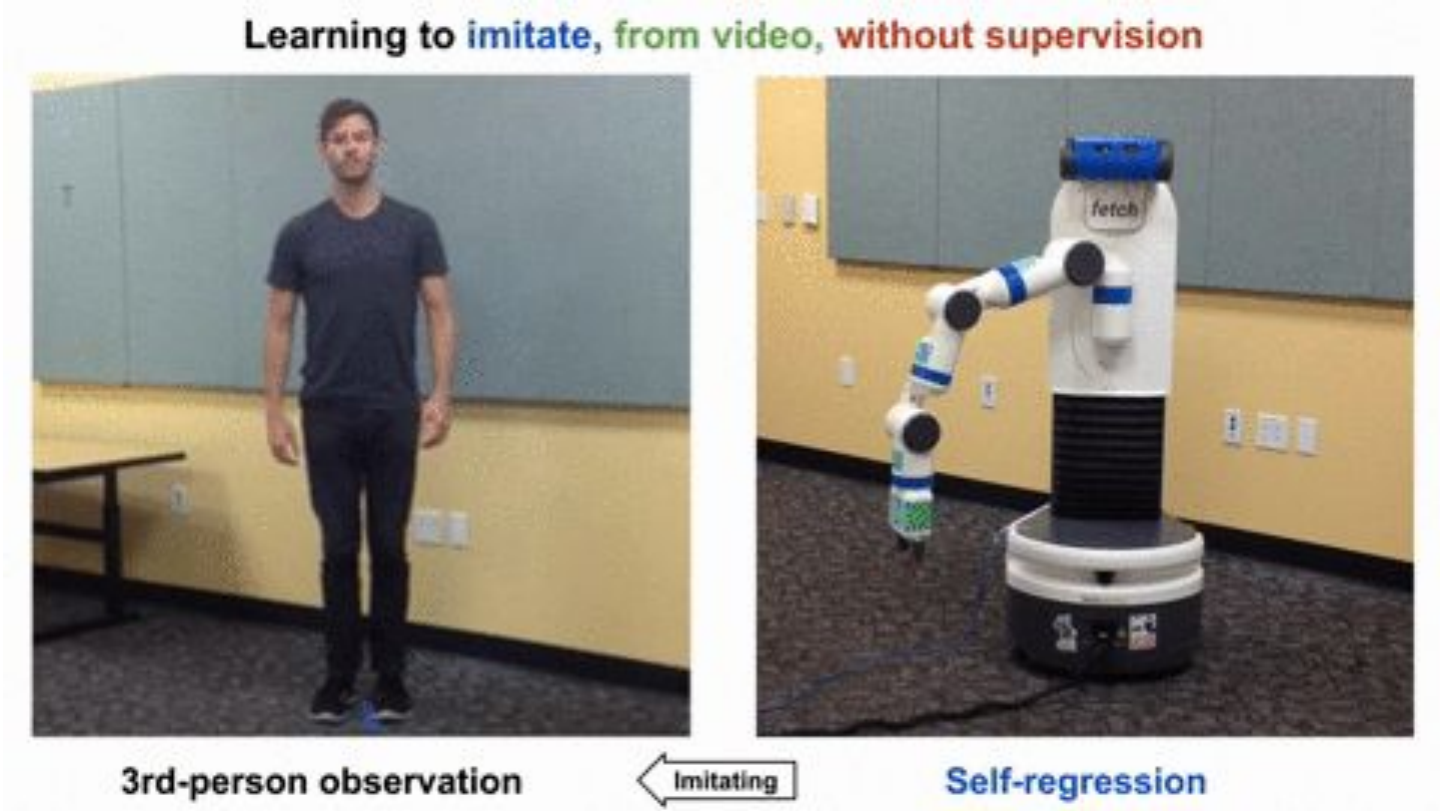
Pose imitation

- Entirely self-supervised
- End-to-end (no explicit pose)



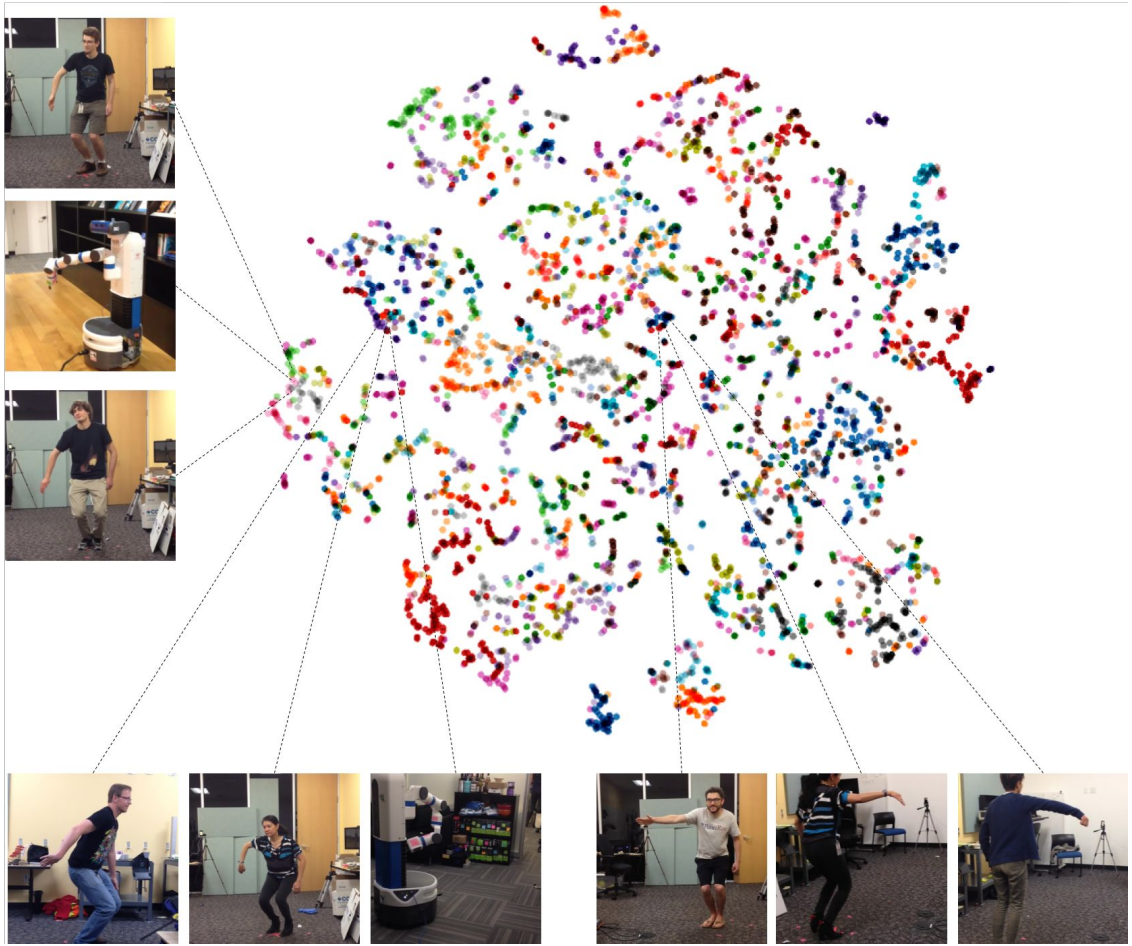
Slide credit: Pierre Sermanet

Pose imitation (real robot)



Slide credit: Pierre Sermanet

t-SNE



Slide credit: Pierre Sermanet

Outline

- Embedding Methods Basics
- Case Study 1: Embeddings for Text
- Case Study 2: Embedding for Images
- Case Study 3: Embeddings from Sequential Data
- **Case Study 4: Embedding for Multimodal Data**

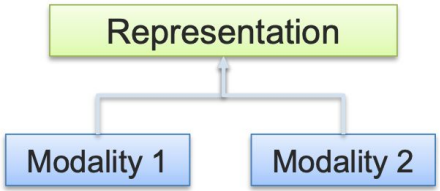
Multimodal learning

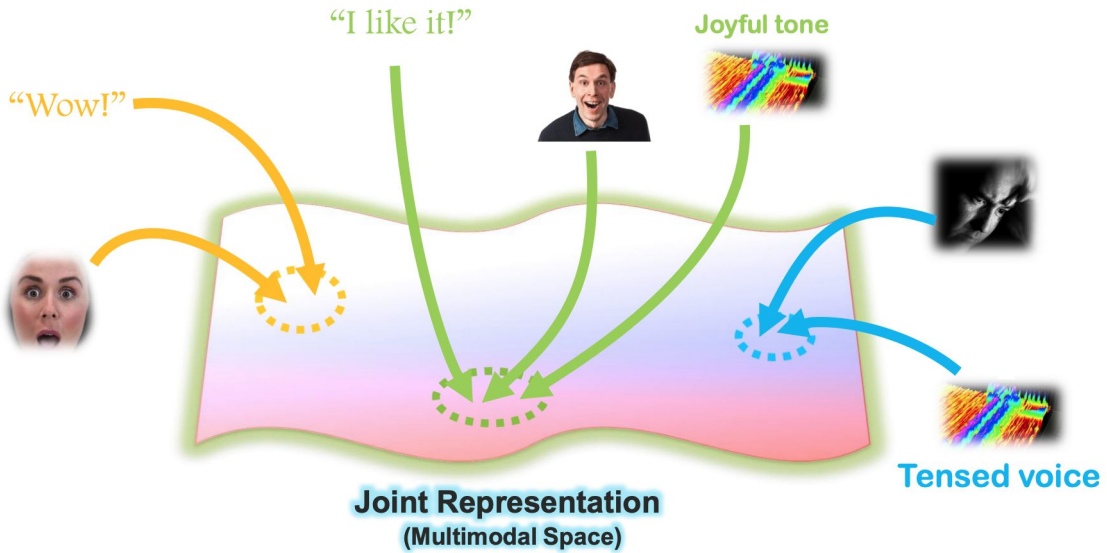
- What is multi-modal learning ?
  - Learning that involves multiple modalities
  - This can manifest itself in different ways
    - Input is one modality, output is another
    - Multiple modalities are learned jointly
    - One modality assists in the learning of another
- Examples of modalities
  - Natural Language, Visual, Auditory, Haptics/touch
  - Physiological signals (Eg. ECG)
  - Other modalities (Infrared images, fMRI, depth images)

Slide credit: Louis-Philippe Morency, Tadas Baltrusaitis

Multimodal learning

- Core challenge: Representation
- How do we represent and summarize multimodal data in a way that exploits the complementarity and redundancy

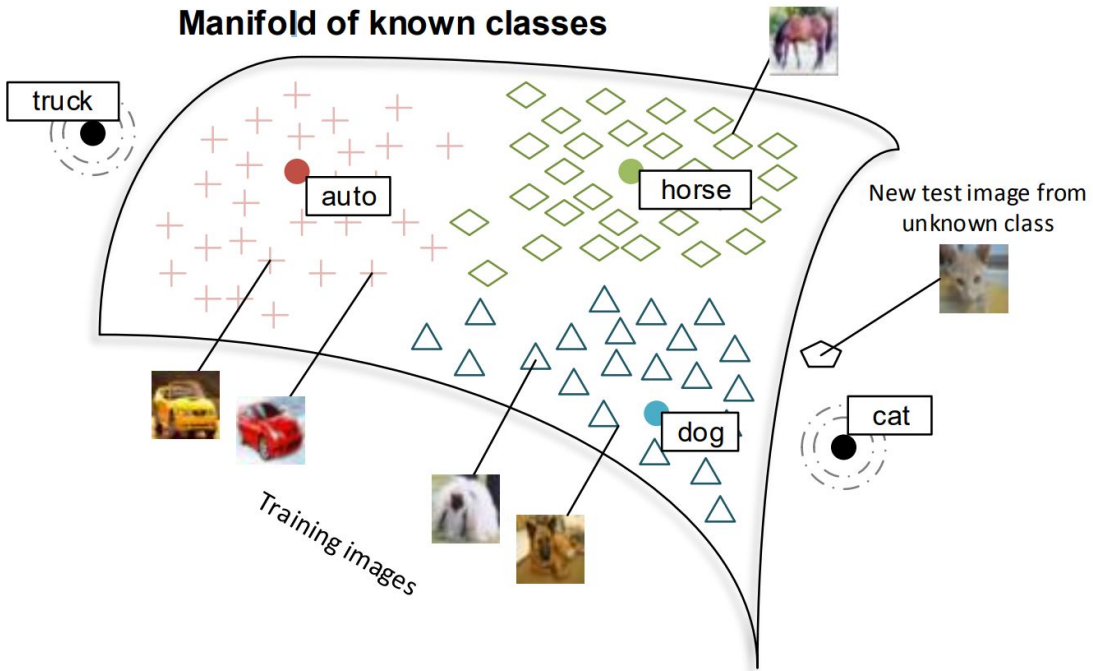




Slide credit: Louis-Philippe Morency, Tadas Baltrusaitis

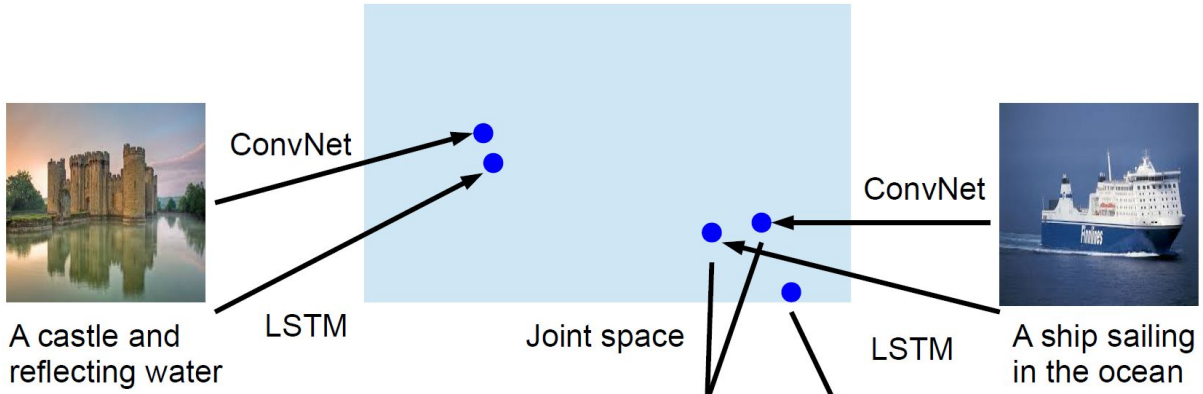
Zero-shot learning using cross-modal embeddings

Idea: learning a mapping from an input image into a word vector (embedding space)



[Socher et al.,2013, Zero-Shot Learning Through Cross-Modal Transfer]

Joint image-text embedding



Minimize the following objective:

$$\text{images} \rightarrow \sum_{\mathbf{x}} \sum_{k} \max\{0, \alpha - s(\mathbf{x}, \mathbf{v}) + s(\mathbf{x}, \mathbf{v}_k)\} +$$

$$\text{text} \rightarrow \sum_{\mathbf{v}} \sum_{k} \max\{0, \alpha - s(\mathbf{v}, \mathbf{x}) + s(\mathbf{v}, \mathbf{x}_k)\}$$

[Kiros et al.,2014, Unifying Visual-Semantic Embeddings with Multimodal Neural Language Models]

Slide credit: Ryan Kiros

Retrieval

tower, building, cathedral, dome, castle

bowl, cup, soup, cups, coffee

kitchen, stove, oven, refrigerator, microwave

ski, skiing, skiers, skiiers, snowmobile

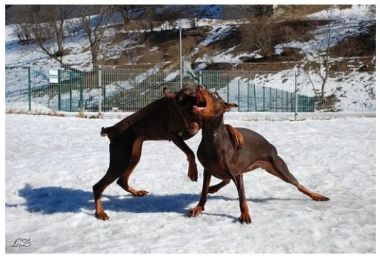
beach

snow

[Kiros et al.,2014, Unifying Visual-Semantic Embeddings with Multimodal Neural Language Models]

Slide credit: Ryan Kiros

Retrieval



The dogs are in the snow in front of a fence .

Four men playing basketball , two from each team .

A boy skateboarding

Two men and a woman smile at the camera .

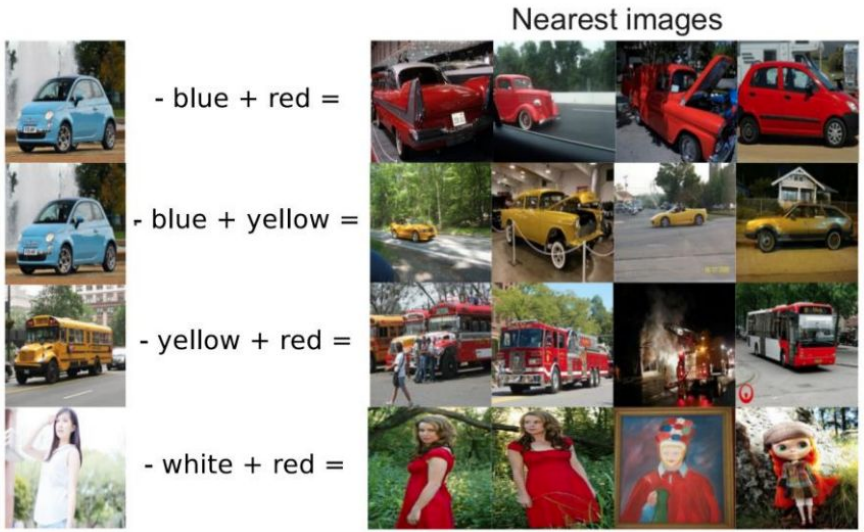
Women participate in a skit onstage .

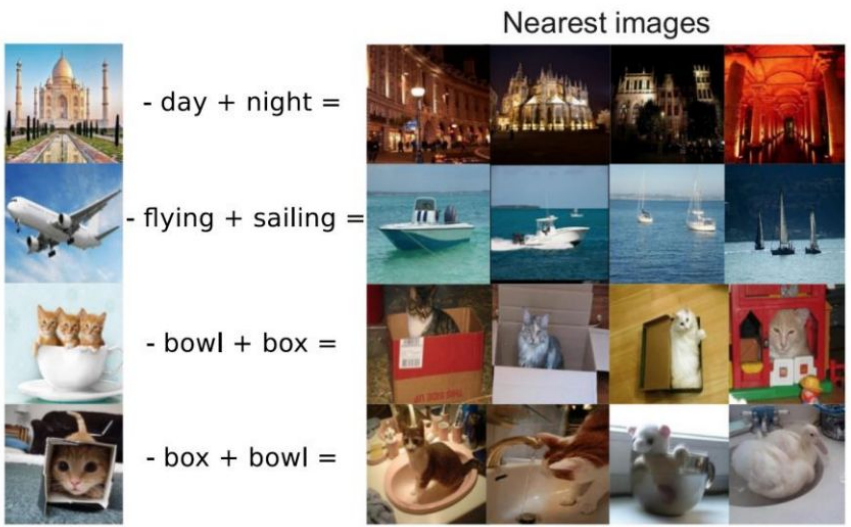
A man is doing tricks on a bicycle on ramps in front of a crowd .

[Kiros et al.,2014, Unifying Visual-Semantic Embeddings with Multimodal Neural Language Models]

Slide credit: Ryan Kiros

Multimodal Vector Space Arithmetic



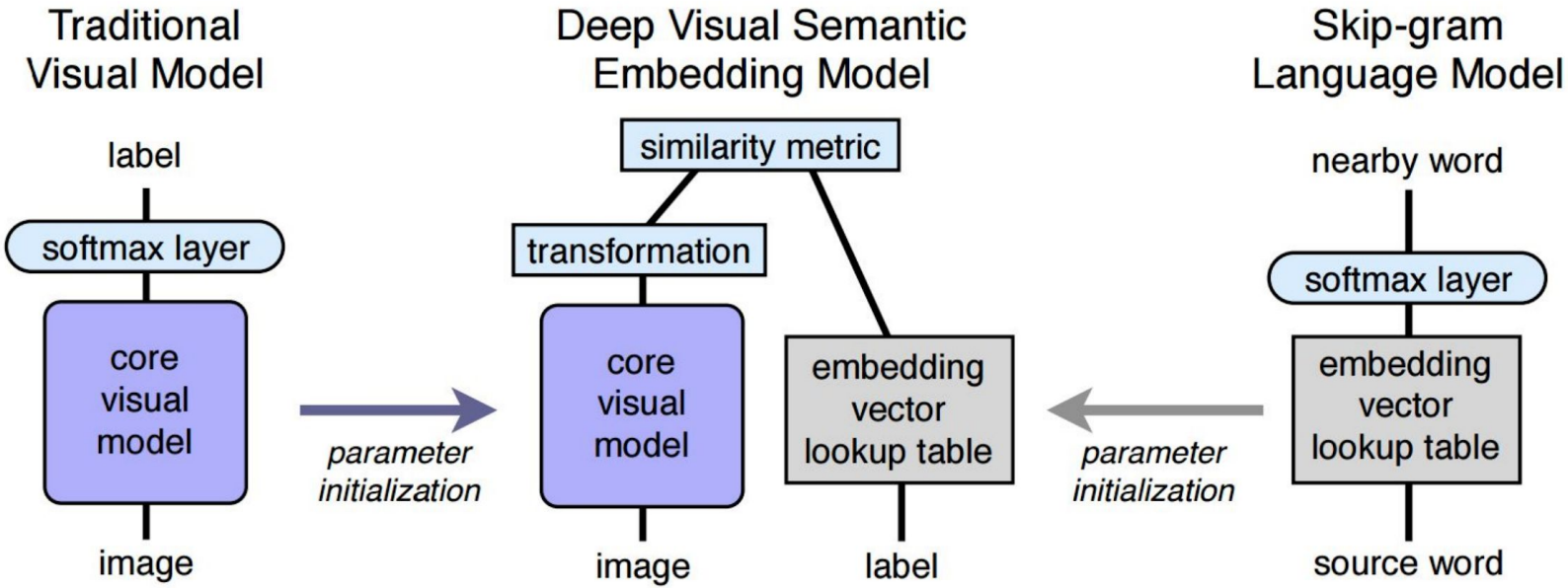


[Kiros et al.,2014, Unifying Visual-Semantic Embeddings with Multimodal Neural Language Models]

Slide credit: Ryan Kiros

Deep Visual-Semantic embeddings

- Learn to map images and words to the same space

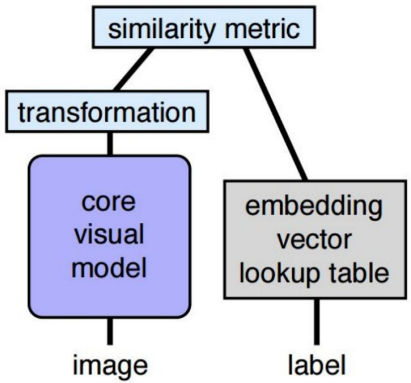


[Frome et al.,2013, DeViSE: A Deep Visual-Semantic Embedding Model]

Deep Visual-Semantic embeddings

- Pre-train visual model and language model independently
- Use a linear projection layer to map features to a shared space
- Loss function:

$$loss(image, label) = \sum_{j \neq label} \max[0, margin - \vec{t}_{label} M\vec{v}(image) + \vec{t}_j M\vec{v}(image)]$$



[Frome et al.,2013, DeViSE: A Deep Visual-Semantic Embedding Model]

Zero shot learning with DeViSE

- Can infer about candidate labels it has never visually observed.
- Model trained on images labeled tiger shark, bull shark, and blue shark, but never with images labeled shark, would likely learn a representation of the general concept of shark.

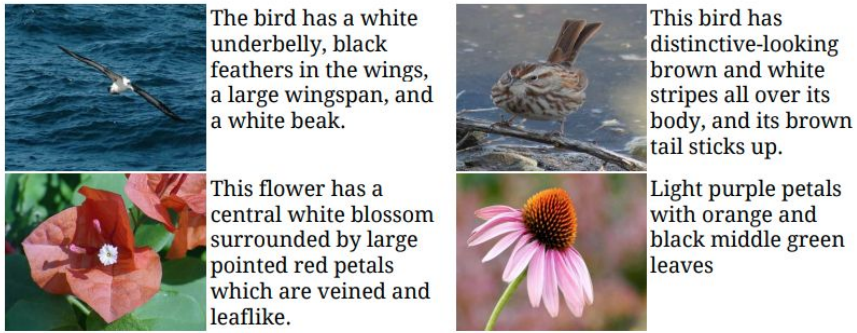
	Our model	Softmax over ImageNet 1K
A	eyepiece, ocular	typewriter keyboard
	Polaroid	tape player
	compound lens	reflex camera
	telephoto lens, zoom lens	CD player
	rangefinder, range finder	space bar
B	oboe, hautboy, hautbois	reel
	bassoon	punching bag, punch bag, ...
	English horn, cor anglais	whistle
	hook and eye	bassoon
	hand	letter opener, paper knife, ...

	Our model	Softmax over ImageNet 1K
D	fruit	pineapple, ananas
	pineapple	coral fungus
	pineapple plant, Ananas ..	artichoke, globe artichoke
	sweet orange	sea anemone, anemone
	sweet orange tree, ...	cardoon
E	comestible, edible, ...	pot, flowerpot
	dressing, salad dressing	cauliflower
	Sicilian pizza	guacamole
	vegetable, veggie, veg	cucumber, cuke
	fruit	broccoli

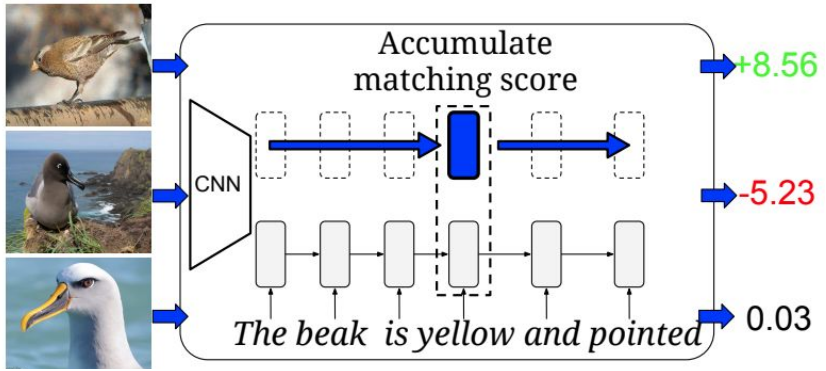
[Frome et al.,2013, DeViSE: A Deep Visual-Semantic Embedding Model]

Embeddings for fine-grained descriptions

Fine-grained descriptions



Learn scoring function between image and text descriptions



[Reed et al.,2017, Learning Deep Representations of Fine-Grained Visual Descriptions]

Embeddings for fine-grained descriptions

Training Objective $\frac{1}{N}\sum_{n=1}^{N} \ell_v(v_n, t_n, y_n) + \ell_t(v_n, t_n, y_n)$

$$\ell_v(v_n, t_n, y_n) = \max_{y \in \mathcal{Y}}(0, \Delta(y_n, y) + \mathbb{E}_{t \sim \mathcal{T}(y)}[F(v_n, t) - F(v_n, t_n)])$$

$$\ell_t(v_n, t_n, y_n) = \max_{y \in \mathcal{Y}}(0, \Delta(y_n, y) + \mathbb{E}_{v \sim \mathcal{V}(y)}[F(v, t_n) - F(v_n, t_n)])$$

visual information $v \in \mathcal{V}$

text descriptions $t \in \mathcal{T}$

class labels $y \in \mathcal{Y}$

$\Delta : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ is the 0-1 loss

$F : \mathcal{V} \times \mathcal{T} \to \mathbb{R}$

$F(v, t) = \theta(v)^T \varphi(t)$

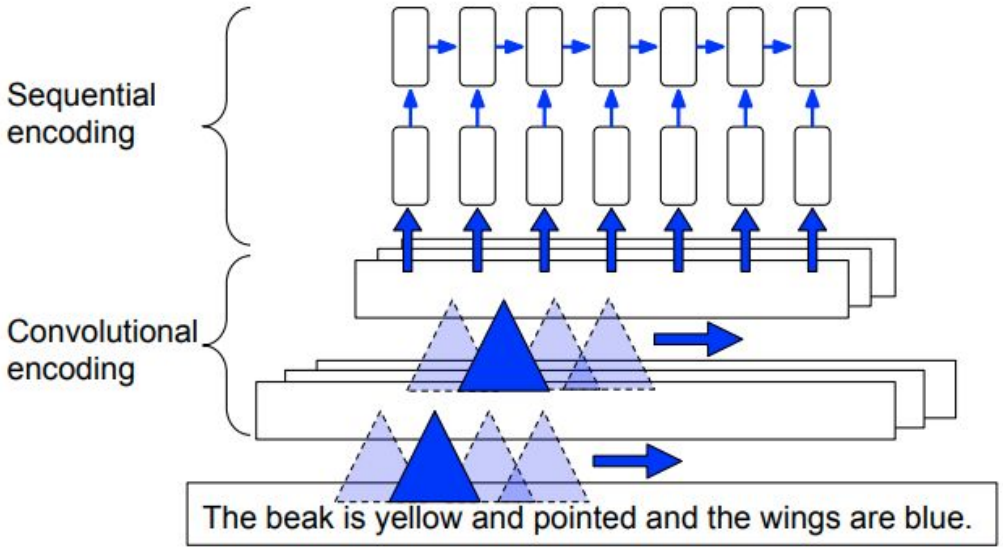
learnable encoder functions

$\theta(v)$ for images and $\varphi(t)$ for text

[Reed et al.,2017, Learning Deep Representations of Fine-Grained Visual Descriptions]

Embeddings for fine-grained descriptions

- Text-encoder: CNN-RNN hybrid model



[Reed et al.,2017, Learning Deep Representations of Fine-Grained Visual Descriptions]

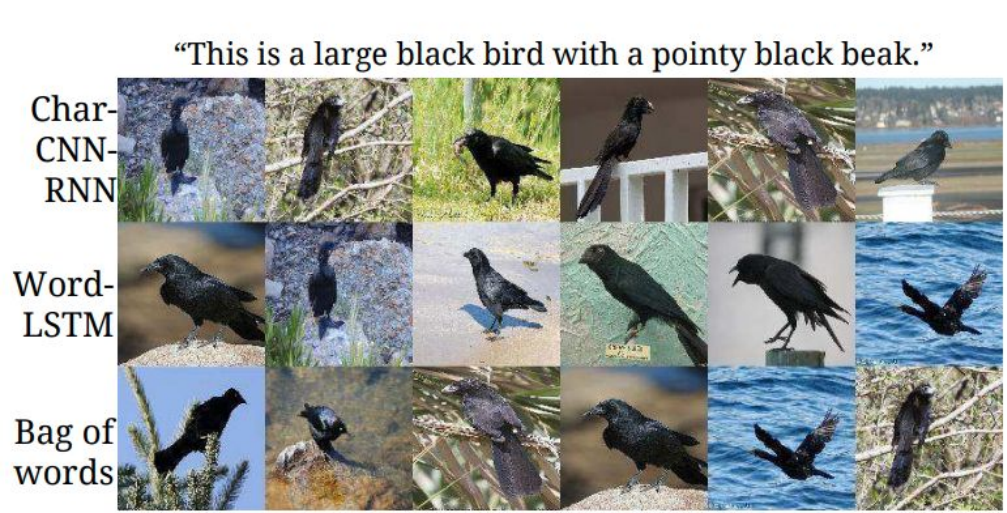
Embeddings for fine-grained descriptions

- t-SNE embeddings of *test class* description embeddings from Oxford-102 and CUB datasets, marked with corresponding images.

[Reed et al.,2017, Learning Deep Representations of Fine-Grained Visual Descriptions]

Embeddings for fine-grained descriptions

- Zero-shot retrieval with different text encoders



[Reed et al.,2017, Learning Deep Representations of Fine-Grained Visual Descriptions]

Summary

- Embedding methods can be useful for
  - Data visualization (e.g., tSNE)
  - Initializing DNN/RNN for downstream tasks (e.g., word2vec, Glove)
  - Similarity measure (e.g., Siamese networks)
- Case Study 1: Embeddings for Text (skip-gram/CBOW word2vec, Glove)
- Case Study 2: Embedding for Images (Siamese networks; pairwise/triplet losses)
- Case Study 3: Embeddings from Sequential Data (time-contrastive nets)
- Case Study 4: Embedding for Multimodal Data (image/text joint embedding)