One Shot Learning with Siamese Network

Nisarg Doshi 202111033

Hemani Bharadwaj 202111036

Dec 9, 2022

Introduction and Motivation

- Deep CNN for image classification
- Limitation of Deep CNN: Lots of labelled data required
- In many applications, not feasible!
- Eg. Face Recognition for all employees of a big organization
- One shot learning aims to solve this problem
- It makes predictions with just a single training example of each new class
- It uses a supervised training approach to learn generic input features based on the training data and then it makes predictions about unknown class distributions.

Problem Statement

- We explore a method called One Shot Learning by learning siamese neural networks which employ a unique structure to naturally rank similarity between inputs.

Figure: One Shot Prediction Task [1]

[1] Koch, Gregory R.. "Siamese Neural Networks for One-Shot Image Recognition." (2015).

Classification vs One Shot Learning

- CNN-based classification has two limitations:

- During training, large number of images required
- If we add a new class, need to retrain the model

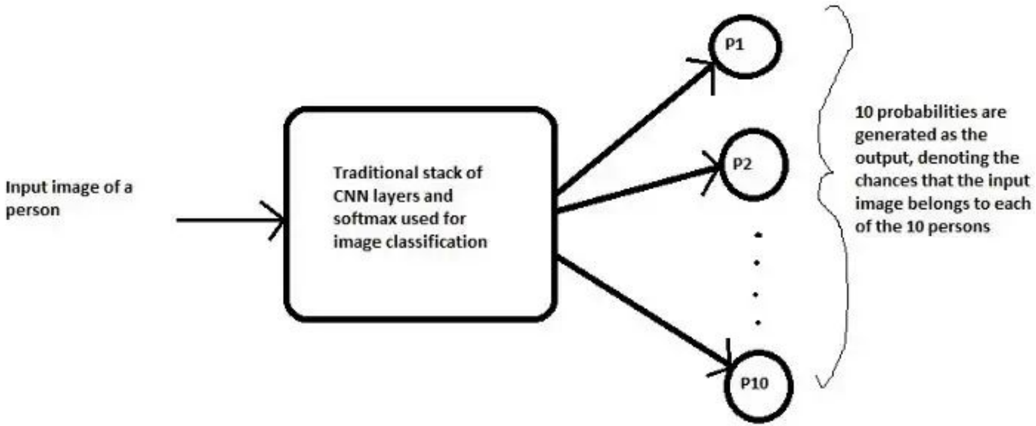


Figure: Standard CNN classification [2]

[2] https://towardsdatascience.com/one-shot-learning-with-siamese-networks-using-keras-17f34e75bb3d

Classification vs One Shot Learning

- One Shot Classification only requires one training example for each class
- New class can be easily added by just taking one training sample

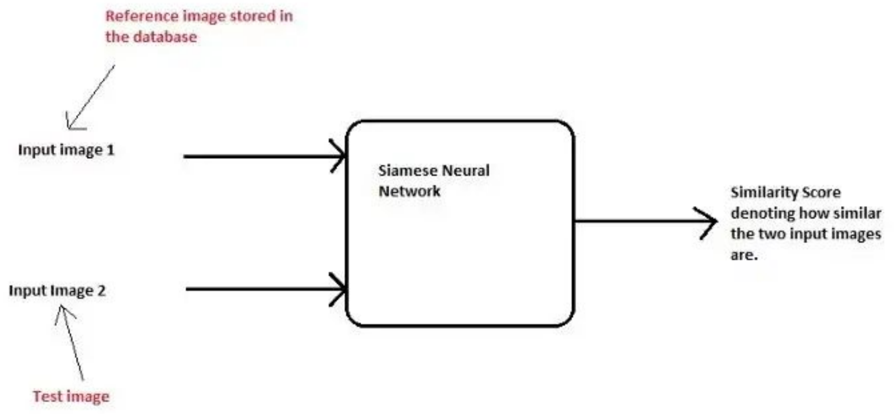


Figure: One Shot classification [2]

[2] https://towardsdatascience.com/one-shot-learning-with-siamese-networks-using-keras-17f34e75bb3d

Siamese Network: Figure

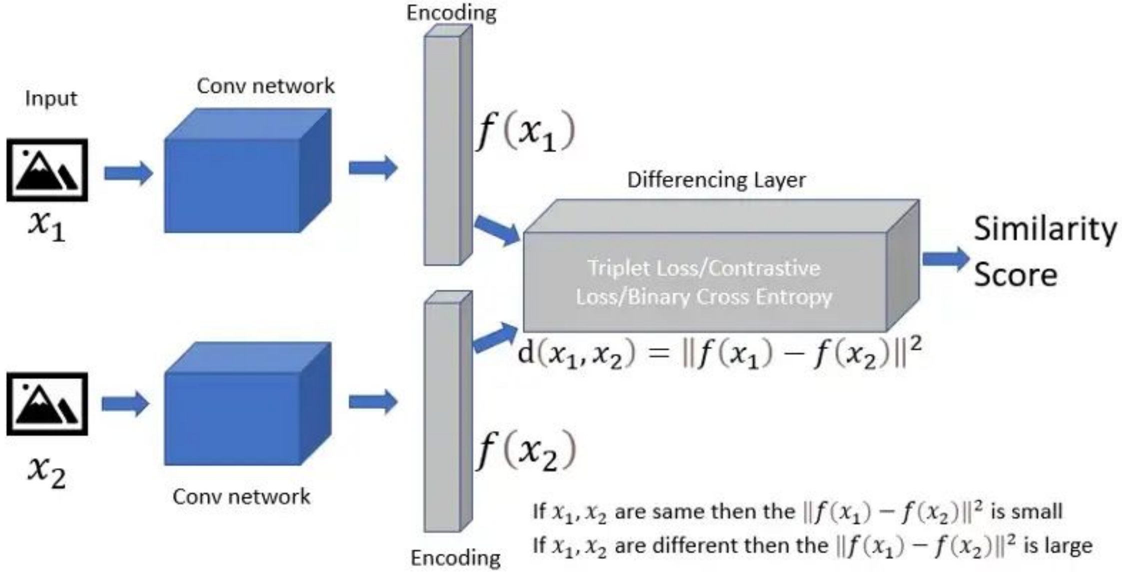
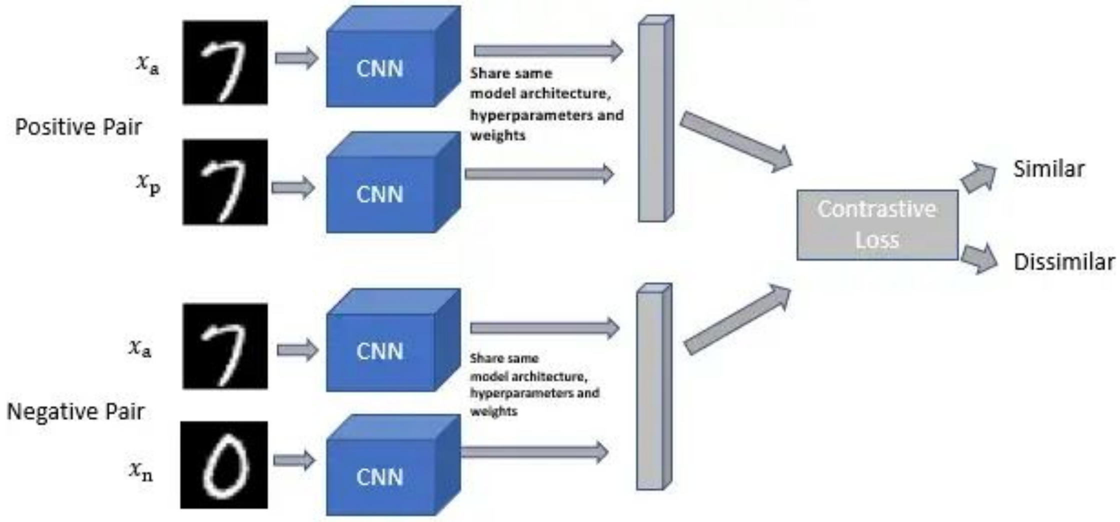


Figure: Siamese Network [3]

[3] https://medium.com/swlh/one-shot-learning-with-siamese-network-1c7404c35fda

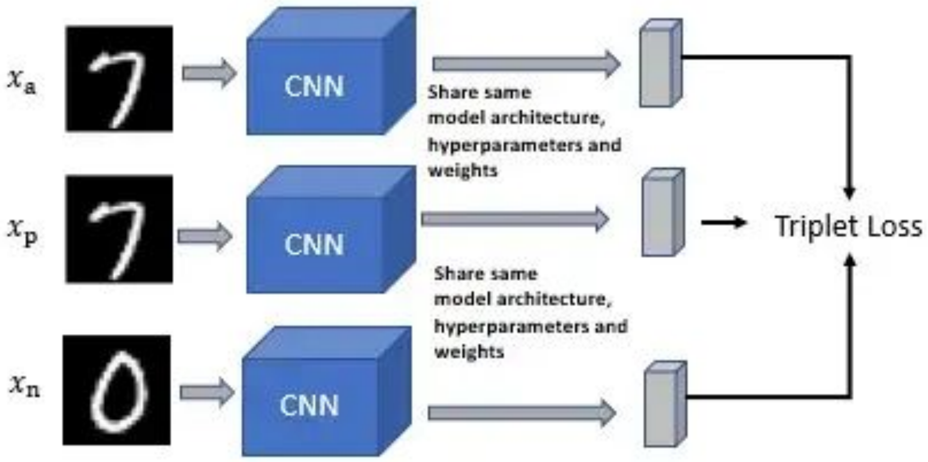
Loss Functions: Contrastive Loss



$$Contrastive\ Loss = (1-Y)\frac{1}{2}D_w^2 + (Y)\frac{1}{2}\{\max(0, m - D_\omega^2)\}$$

[3] https://medium.com/swlh/one-shot-learning-with-siamese-network-1c7404c35fda

Loss Functions: Triplet Loss



$$L = \max(\mathrm{d}(a,p) - \mathrm{d}(a,n) + margin, 0)$$

[3] https://medium.com/swlh/one-shot-learning-with-siamese-network-1c7404c35fda

Siamese Network: Training

- Load the dataset containing the different classes
- Create positive and negative data pairs.
- Build the Convolutional neural network, which outputs the feature encoding using a fully connected layer.
- Build the differencing layer to calculate the Euclidean distance between the two sister CNN networks encoding output.
- The final layer is a fully-connected layer with a single node using the sigmoid activation function to output the Similarity score.
- Compile the model using binary cross-entropy

Siamese Network: Testing

- Send two inputs to the trained model to output the Similarity score

- As the last layer uses the sigmoid activation function, it outputs a value in the range 0 to 1.

- A Similarity score close to 1 implies that the two inputs are similar. A Similarity score close to 0 implies that the two inputs are dissimilar.

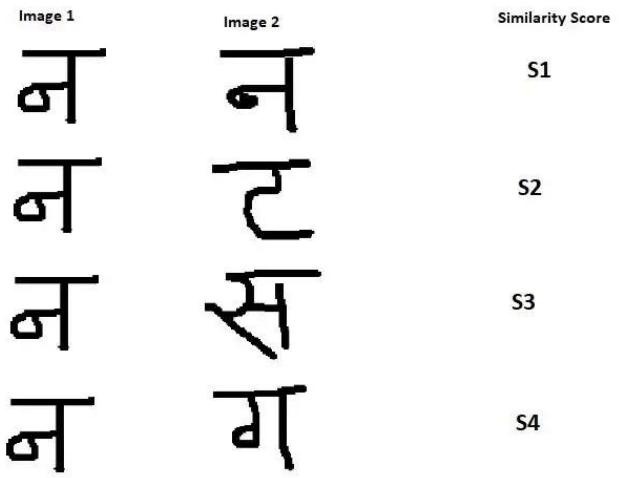
- Threshold usually kept: 0.5

Siamese Network: Validation

- Just looking at the score it's difficult to observe the results.
- N-way one shot learning
- Repeating this for k times,

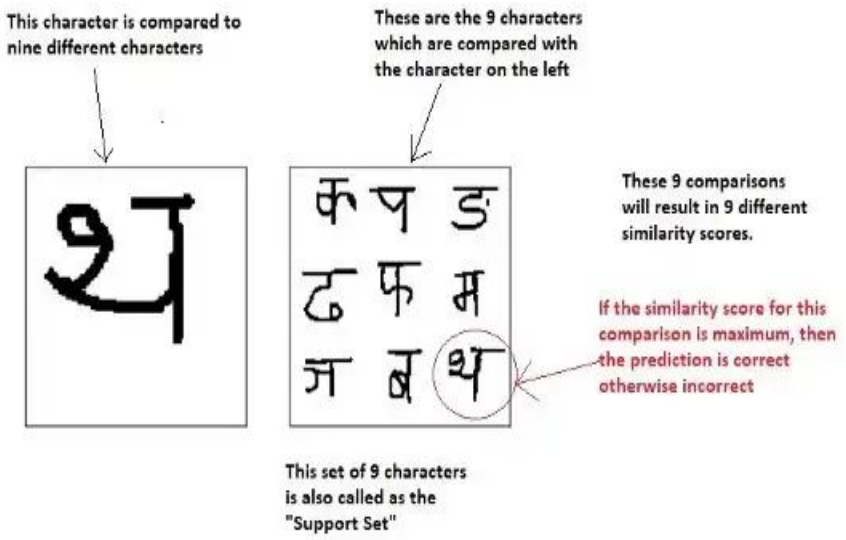
  percent_correct = (100 * n_correct) / k

  where k: total no. of trials
  n_correct: no. of correct predictions out of k trials.



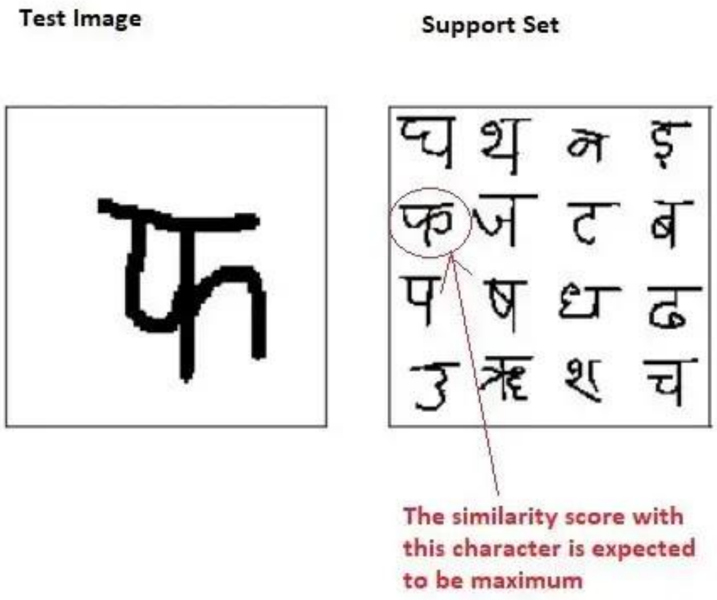
[3] https://medium.com/swlh/one-shot-learning-with-siamese-network-1c7404c35fda

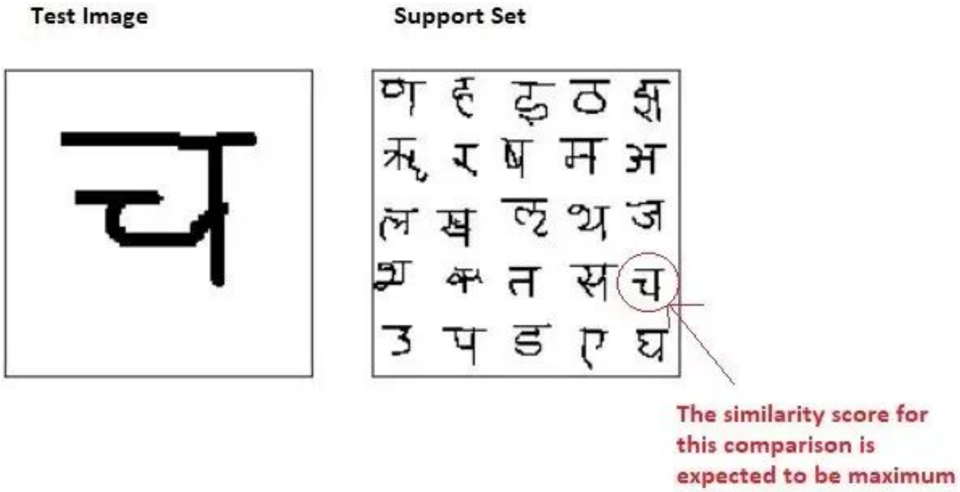
Siamese Network: Validation



[3] https://medium.com/swlh/one-shot-learning-with-siamese-network-1c7404c35fda

Siamese Network: Validation

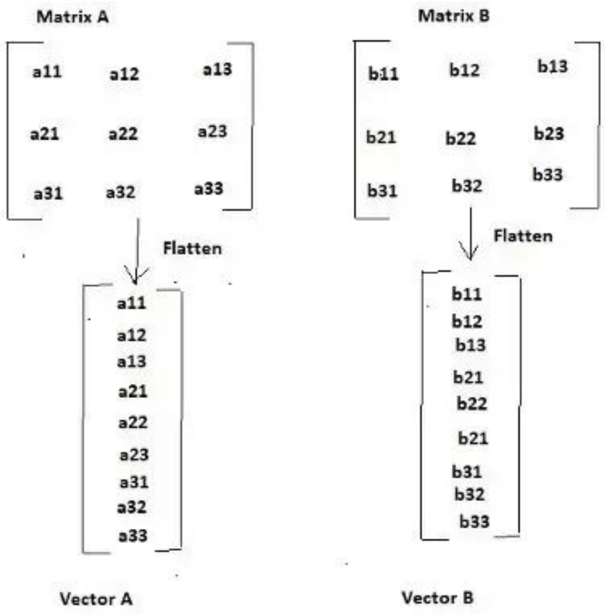




[3] https://medium.com/swlh/one-shot-learning-with-siamese-network-1c7404c35fda

Baseline Model: Nearest Neighbour

- First, flatten the matrix into vectors:



- Then calculating L2 norm (Euclidean distance):

$$||a-b||_2^2 = \sum_{i=1}^{n}(a_i - b_i)^2$$

[3] https://medium.com/swlh/one-shot-learning-with-siamese-network-1c7404c35fda

Dataset

- Omniglot Dataset[4]
- 1623 hand drawn characters across 50 alphabets
- 20 examples for every character
- Each image is grayscale of 105x105 size
- Training: 30 alphabets
- Testing: 20 alphabets

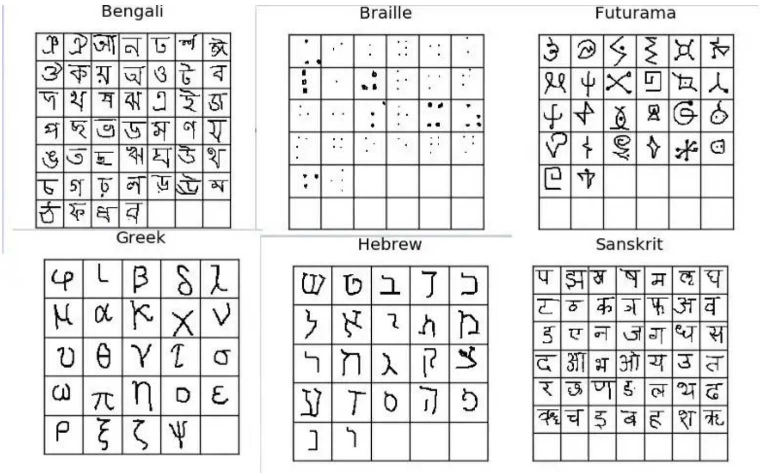
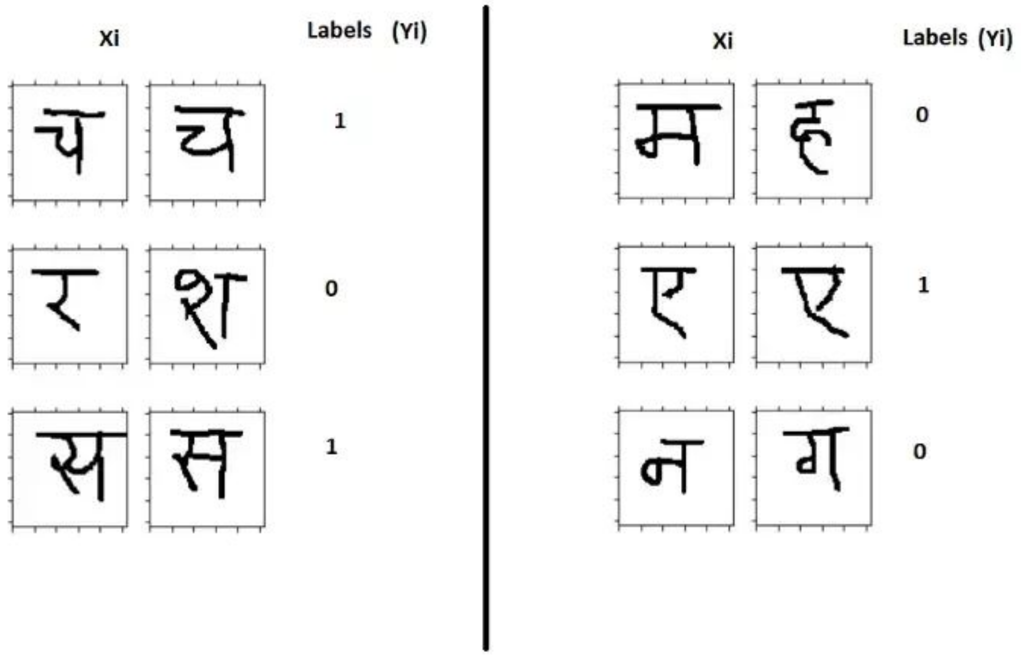


Figure: Omniglot dataset

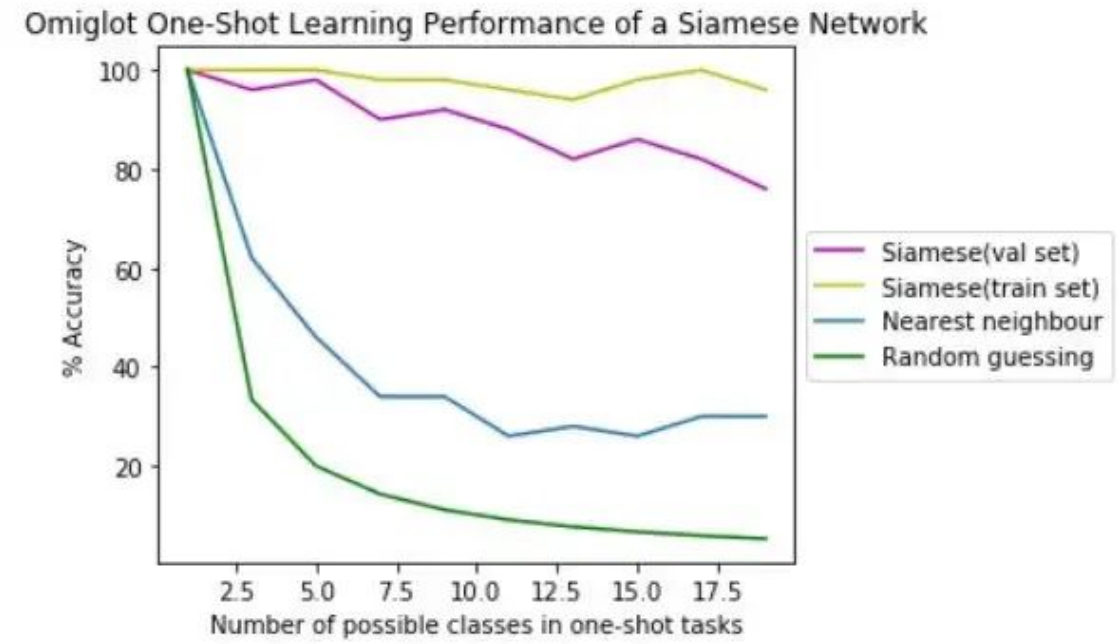
[4] https://sorenbouma.github.io/blog/oneshot/

Dataset visualization for our problem



https://sorenbouma.github.io/blog/oneshot/

Results



[3] https://medium.com/swlh/one-shot-learning-with-siamese-network-1c7404c35fda

References

[1] Koch, Gregory R.. “Siamese Neural Networks for One-Shot Image Recognition.” (2015).

[2] https://towardsdatascience.com/one-shot-learning-with-siamese-networks-using-keras-17f34e75bb3d

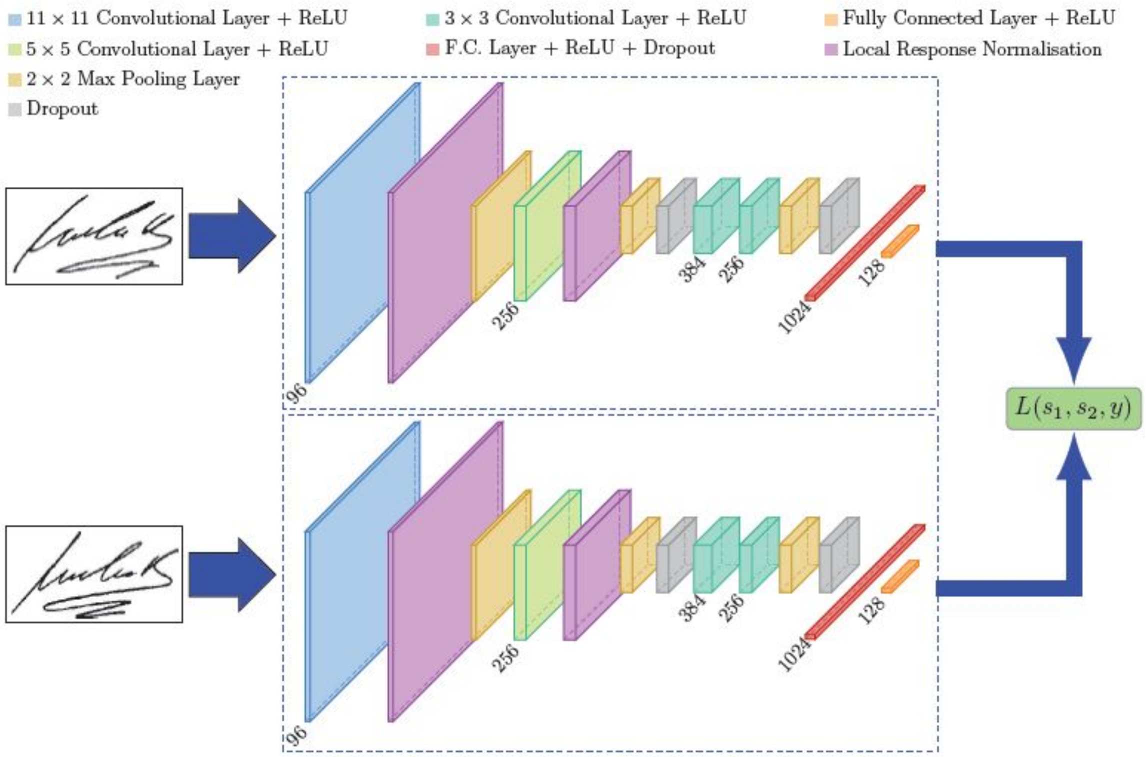
[3] https://medium.com/swlh/one-shot-learning-with-siamese-network-1c7404c35fda

[4] https://sorenbouma.github.io/blog/oneshot/

Thank You

Backup Slides

Internal Architecture

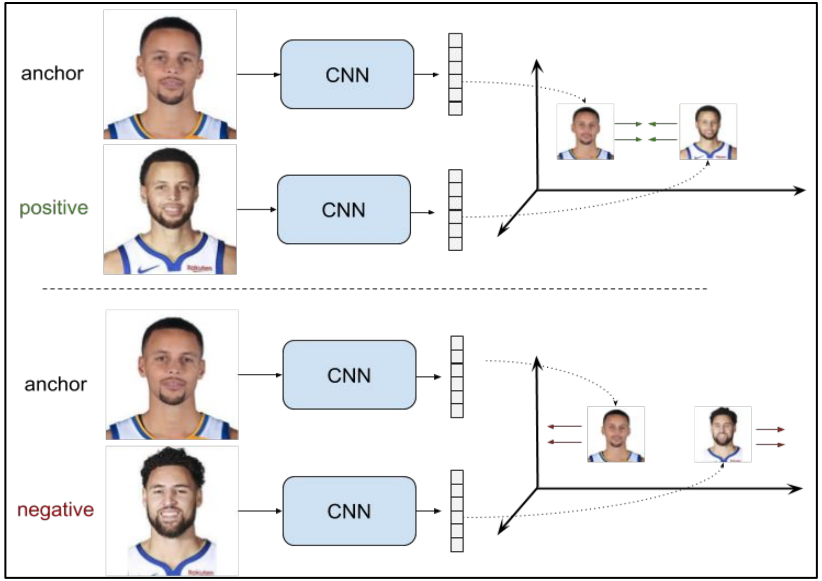


[3] https://medium.com/swlh/one-shot-learning-with-siamese-network-1c7404c35fda

Loss functions:

- Contrastive Loss function

(1-Y) x 0.5 x X² + Y x 0.5 x (max(0,m-X))²



- **Triplet loss**

max(0,d(A,P) + d(A,B) + alpha)

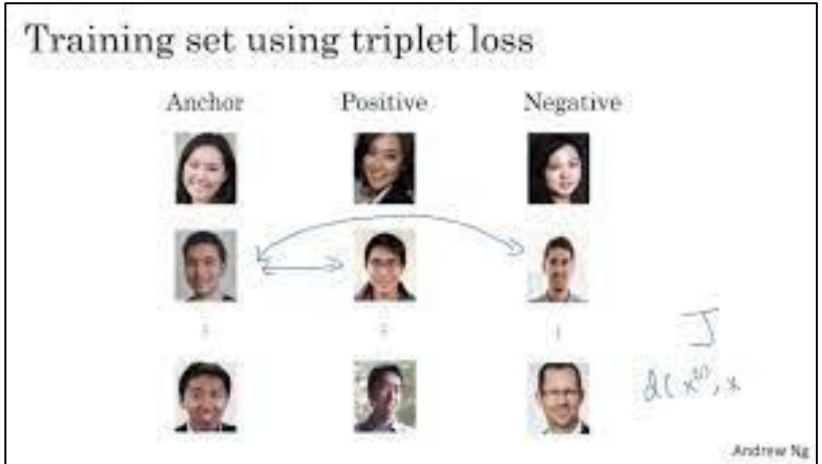


Image source : https://gombru.github.io/2019/04/03/ranking_loss/

Anchor $\quad$ Positive $\qquad\qquad$ Anchor $\quad$ Negative

$A \qquad P \qquad\qquad\qquad A \qquad N$

$d(A,P) = 0.5 \qquad \alpha \to 0.2 \qquad d(A,N) = \cancel{0.51}\ 0.7$

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

$$\boxed{\underbrace{\| f(A) - f(P) \|^2}_{0} - \underbrace{\| f(A) - f(N) \|^2}_{0} + \alpha \le 0}$$

α — margin

$f(\text{img}) = \vec{0}$

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

Andrew Ng