Siamese Network and Triplet Loss

Lukas Lamminger, Filip Vecek

January 31,2020

Schedule

1 Neural Network

2 Convolutional Network

3 One Shot Learning

4 Triplet Loss

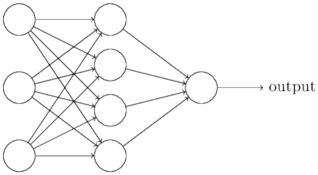
5 Face Net

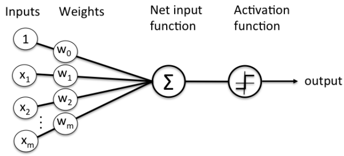
Neural Network

Neural Network

- Neural networks are a set of algorithms, that are designed to recognize patterns.
- The patterns they recognize are numerical, contained in vectors into which all the real-life data must be translated.
- They are composed of several layers, which in turn are composed out of several nodes.

Structure





Schematic of Rosenblatt's perceptron.

- The neural network receives the input data in the first so-called input layer. The data is then processed in the subsequent hidden layers until the final result is received in the output layer.
- Each specific node acts as a function to process the received data, before it sends the data to the next node to be processed.

https://sebastianraschka.com/Articles/2015_singlelayer_neurons.html

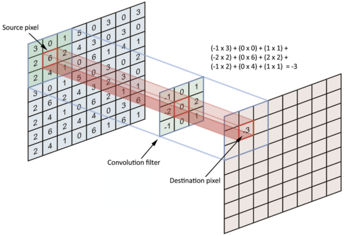
Convolutional Network

Convolutional Network

- Several drawbacks to traditional neural networks (multilayer perceptron or MLP)
- Since we use one perceptron for each input, the amount of weights quickly becomes unimaginably large
- Another problem is that the MLP will react differently to an input and its shifted version

As a result one will want to use a different kind of network for image processing

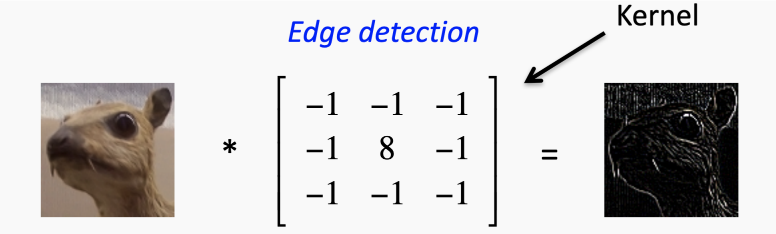
Convolutional Network



- Convolutional Networks are the best choice for image processing since they are translation invariant and each pixel position and neighborhood has semantic meaning
- This influence of nearby pixels is mainly done through the use of a filter, that we move across the picture from the top left to bottom right

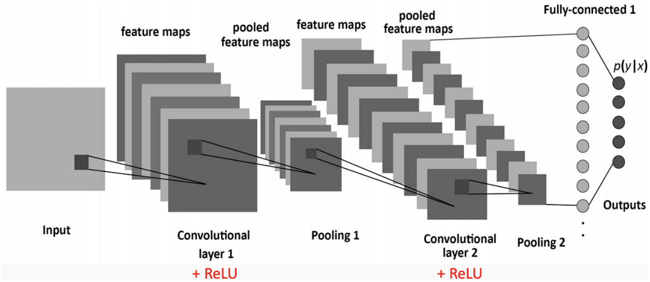
https://towardsdatascience.com/simple-introduction-to-convolutional-neural-networks-cdf8d3077bac

- As result we significantly reduce the number of weights the neural network must learn in comparison to a MLP
- The change in location for the key features also doesn't throw the network off
- The specific filters for different features also get continuously updated through the training process, so the chance of two features having the same filter is extremely low



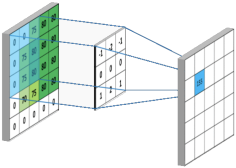
https://towardsdatascience.com/simple-introduction-to-convolutional-neural-networks-cdf8d3077bac

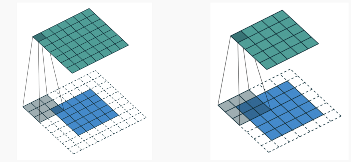
- After the filters have passed through the image a feature map is created, each corresponding to the results of a filter
- In turn these feature maps can be used as input for the next convolutional layer again
- Another option would be to create a pooling layer, where the "best" feature maps are pooled together to be used as the next input



https://towardsdatascience.com/simple-introduction-to-convolutional-neural-networks-cdf8d3077bac

Padding





Full padding. Introduces zeros such that all pixels are visited the same amount of times by the filter. Increases size of output.

Same padding. Ensures that the output has the same size as the input.

- Since we apply the filters through convolutions the output would obviously be downsized compared to the input - to prevent this one can use padding
- Padding is usually used to retain the size of the input in the feature maps or at least keep them from being too small in very deep networks

https://towardsdatascience.com/simple-introduction-to-convolutional-neural-networks-cdf8d3077bac

One Shot Learning

One Shot Learning

- One shot learning is the technique of learning representations from a single image.
- In the case of face verification, a model or system may only have one example of a persons face on record and must correctly verify new photos of that person, perhaps each day.
- Therefore, face recognition is a common example of an one-shot learning task.

One Shot Learning





- Since there isn't enough data to build a CNN, one could build a similarity function that compares the images on the right with all images on the left
- The similarity function will return a value and if that value is lesser than or equal to a threshold value the images are similar, else they are not

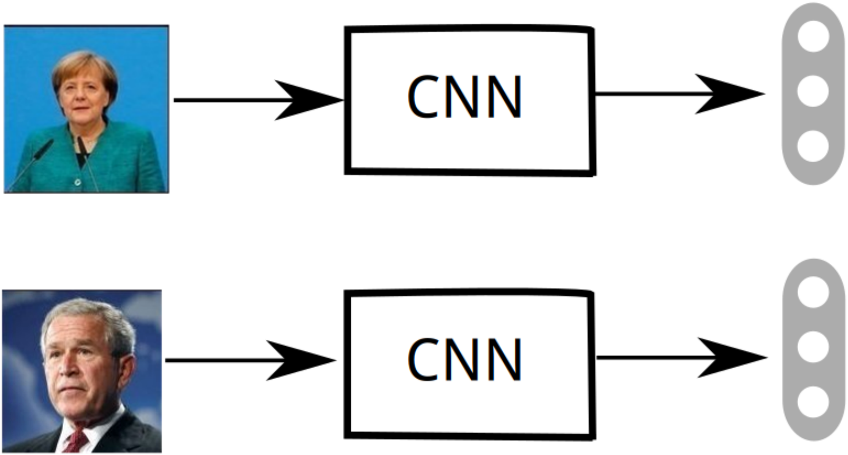
Siamese Network

Siamese network

In Siamese networks, we take an input image of a person and find out the encodings of that image, then, we take the same network without performing any updates on weights or biases and input an image of a different person and again predict it's encodings.

Now, we compare these two encodings to check whether there is a similarity between the two images

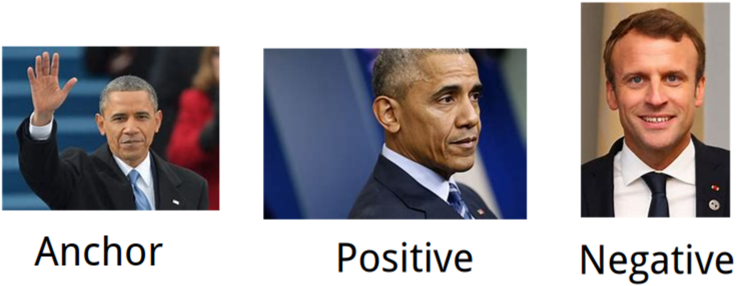
Siamese Network



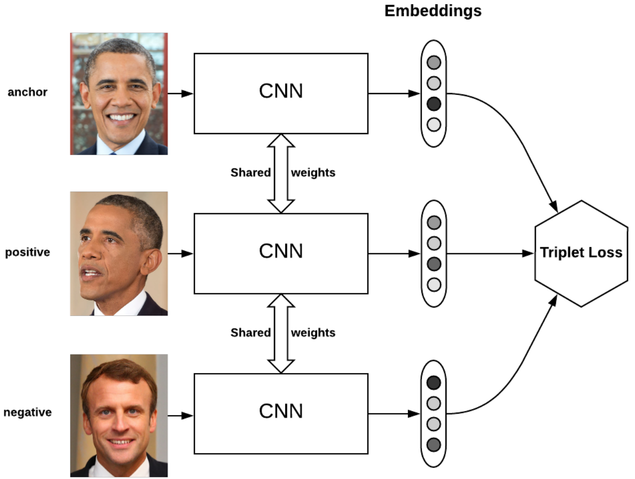
Triplet Loss

Triplet Loss

- The network gets trained by taking an anchor image and comparing it with a positive and a negative sample
- The similarity between the anchor and positive sample must be high
- The similarity between the anchor and the negative sample must be low



Triplet Loss



https://omoindrot.github.io/assets/triplet_loss/triplet_loss.png

Triplet Loss

Requirements of the Loss Function

- distance(A,P) < distance(A,N)
- distance(A,P) - distance(A,N) < 0
- In order to avoid trivial solutions we will add a margin

Triplet Loss Function

$$L = max(d(a,p) - d(a,n) + margin, 0)$$

Triplet Loss

FaceNet: A Unified Embedding for Face Recognition and Clustering, 2015.

Triplet Loss Code

```
import tensorflow as tf
```

```
def triplet_loss(y_true, y_pred, N=3):
  anchor_output   = tf.convert_to_tensor(y_pred[:,0:N])
  positive_output = tf.convert_to_tensor(y_pred[:,N:N*2])
  negative_output = tf.convert_to_tensor(y_pred[:,N*2:N*3])

  d_pos = tf.reduce_sum(tf.square(tf.subtract(anchor_output , positive_output)), 1)
  d_neg = tf.reduce_sum(tf.square(tf.subtract(anchor_output , negative_output)), 1)


  loss = tf.maximum(0., 0.5 + d_pos - d_neg)
  loss = tf.reduce_mean(loss)
  return loss
```

Face Net

Face Net

- FaceNet is a face recognition system that was described by Florian Schroff, et al. at Google in their 2015 paper titled “FaceNet: A Unified Embedding for Face Recognition and Clustering.”
- It is a system that, given a picture of a face, will extract high-quality features from the face and predict a 128 element vector representation these features, called a face embedding.

Labelled Faces in the Wild

- Labeled Faces in the Wild (LFW) is a database of face photographs designed for studying the problem of unconstrained face recognition.
- This database was created and maintained by researchers at the University of Massachusetts, Amherst.
- 13,233 images of 5,749 people were detected and centered by the Viola Jones face detector and collected from the web.
- 1,680 of the people pictured have two or more distinct photos in the dataset.

Labeled Faces in the Wild

Positive Pairs in LFW

http://vis-www.cs.umass.edu/lfw/

Face Verification on Labeled Faces in the Wild

RANK	METHOD	ACCURACY	PAPER TITLE	YEAR
1	ArcFace + MS1MV2 + R100,	99.83%	ArcFace: Additive Angular Margin Loss for Deep Face Recognition	2018
2	CosFace	99.73%	CosFace: Large Margin Cosine Loss for Deep Face Recognition	2018
3	FaceNet	99.63%	FaceNet: A Unified Embedding for Face Recognition and Clustering	2015
4	Ring loss	99.52%	Ring loss: Convex Feature Normalization for Face Recognition	2018
5	DeepId2+	99.47%	Deeply learned face representations are sparse, selective, and robust	2014
6	SphereFace	99.42%	SphereFace: Deep Hypersphere Embedding for Face Recognition	2017

https://paperswithcode.com/sota/face-verification-on-labeled-faces-in-the

Sources

- https://algorithmia.com/blog/introduction-to-loss-functions
- https://towardsdatascience.com/siamese-network-triplet-loss-b4ca82c1aec8
- https://medium.com/@susmithreddyvedere/triplet-loss-b9da35be21b8
- https://medium.com/@crimy/one-shot-learning-siamese-networks-and-triplet-loss-with-keras-2885ed022352
- FaceNet: A Unified Embedding for Face Recognition and Clustering, 2015.
- https://towardsdatascience.com/simple-introduction-to-convolutional-neural-networks-cdf8d3077bac