Unsupervised Learning

This section was written by Baris Can Durak and Can Aykin. (last modified on 06.02.2017)

Unsupervised Learning

Similar to supervised learning, a neural network can be used in a way to train on unlabeled data sets. This type of algorithms are categorized under unsupervised learning algorithms and are useful in a multitude of tasks such as clustering.

Convolutional neural networks are generally trained as supervised methods which means both the inputs (i.e. images in an image recognition task) and their labels (i.e. the objects depicted in the images) are available within the training data. On the other hand, specific unsupervised learning methods are developed for convolutional neural networks to pre-train them. These methods were employed in the past in order to overcome the computational limits during the training of the network and are still in use to generally speed up the training process.(1)

Autoencoders

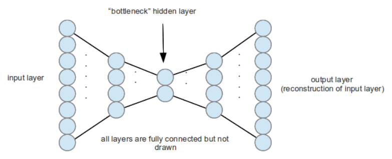
"bottleneck" hidden layer
input layer
output layer
(reconstruction of input layer)
all layers are fully connected but not drawn

***Figure 1**: the neuronal structure of an autoencoder*(3)

The unsupervised learning in convolutional neural networks is employed via autoencoders. The autoencoder structure consists of two layers, an encoding and a decoding layer. The goal of an autoencoder is to achieve identity function within its whole structure. In more detail, the input of size n is represented with m number of parameters within the input layer (encoding) and the output layer aims to recreate the original input of size n from the size m representation (decoding).(2)

- **Encoding:** $f: \mathbb{R}^n \to \mathbb{R}^m, x \mapsto f(x)$
- **Decoding:** $g: \mathbb{R}^m \to \mathbb{R}^n, f(x) \mapsto g(f(x))$
- **Minimizing the loss:** $L(x, g(f(x)))$

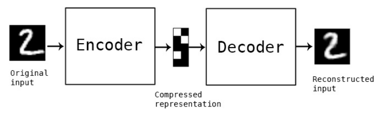

***Figure 2:** the basic functionality of an autoencoder* (4)

In case $m \geq n$ the system can learn identity without any difficulty since expanding the input space into a larger size can easily be achieved without loss. The more challenging task is to create a reduced representation $m < n$ which would encode the input in a smaller, compressed form. While this compression is lossy, it is still possible to recreate the original input as closely as possible, which is the purpose of autoencoder training in general.

As the system aims to achieve identity or recreating the original input as close as possible in case of a reduced representation, the input and the labels of the data are one and the same during training. This allows the structure to learn using only the non-labeled inputs making the method self-governing and hence, an unsupervised learning method.

Deep neural net structures such as convolutional neural networks are computationally expensive to train with simple backpropagation. Therefore, the autoencoders can be utilized to perform layerwise training improving the initialization of individual layers using only non-labeled inputs. Even though this solution alone would not create a fully accurate network, when used as a pre-training method it can accelerate the network's convergence speed as well as reduce the computational requirements of the problem and can be combined with backpropagation in order to meet the accuracy requirements.

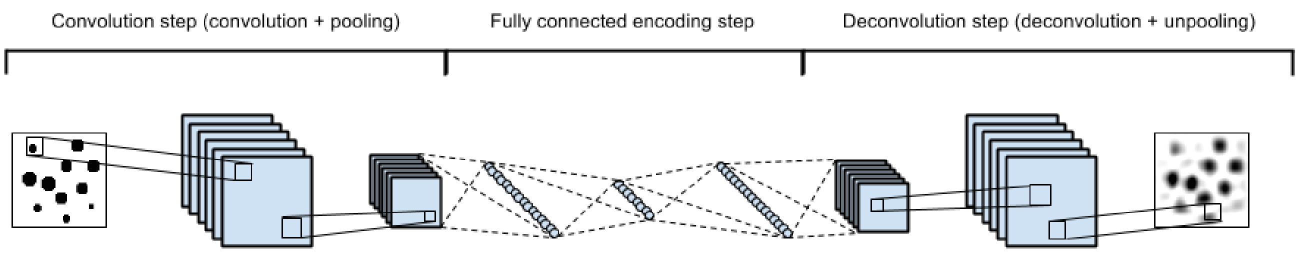
Convolution step (convolution + pooling)
Fully connected encoding step
Deconvolution step (deconvolution + unpooling)

Figure 3: *layer representation of an autoencoder*[5]

Literature

[1] Why does unsupervised pre-training help deep learning? (2010, Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, & Samy Bengio)

[2] Learning deep architectures for AI (2009, Yoshua Bengio)

Weblinks

[3] http://nghiaho.com/?p=1765

[4] https://blog.keras.io/building-autoencoders-in-keras.html

[5] https://swarbrickjones.wordpress.com/2015/04/29/convolutional-autoencoders-in-pythontheanolasagne/