Evolution and Uses of CNNs and Why Deep Learning?

🎙️ *Yann LeCun*

Evolution of CNNs

In animal brains, neurons react to edges that are at particular orientations. Groups of neurons that react to the same orientations are replicated over all of the visual field.

Fukushima (1982) built a neural net (NN) that worked the same way as the brain, based on two concepts. First, neurons are replicated across the visual field. Second, there are complex cells that pool the information from simple cells (orientation-selective units). As a result, the shift of the picture will change the activation of simple cells, but will not influence the integrated activation of the complex cell (convolutional pooling).

LeCun (1990) used backprop to train a CNN to recognize handwritten digits. There is a demo from 1992 where the algorithm recognizes the digits of any style. Doing character/pattern recognition using a model that is trained end-to-end was new at that time. Previously, people had used feature extractors with a supervised model on top.

These new CNN systems could recognize multiple characters in the image at the same time. To do it, people used a small input window for a CNN and swiped it over the whole image. If it activated, it meant there was a particular character present.

Later, this idea was applied to faces/people detection and semantic segmentation (pixel-wise classification). Examples include Hadsell (2009) and Farabet (2012). This eventually became popular in industry, used in autonomous driving applications such as lane tracking.

Special types of hardware to train CNN were a hot topic in the 1980s, then the interest dropped, and now it has become popular again.

The deep learning (though the term was not used at that time) revolution started in 2010-2013. Researchers focused on inventing algorithms that could help train large CNNs faster. Krizhevsky (2012) came up with AlexNet, which was a much larger CNN than those used before, and trained it on ImageNet (1.3 million samples) using GPUs. After running for a couple of weeks AlexNet beat the performance of the best competing systems by a large margin – a 25.8% *vs.* 16.4% top-5 error rate.

After seeing AlexNet's success, the computer vision (CV) community was convinced that CNNs work. While all papers from 2011-2012 that mentioned CNNs had been rejected, since 2016 most accepted CV papers use CNNs.

Over the years, the number of layers used has been increasing: LeNet – 7, AlexNet – 12, VGG – 19, ResNet – 50. However, there is a trade-off between the number of operations needed to compute the output, the size of the model, and its accuracy. Thus, a popular topic now is how to compress the networks to make the computations faster.

Deep Learning and Feature Extraction

Multilayer networks are successful because they exploit the compositional structure of natural data. In compositional hierarchy, combinations of objects at one layer in the hierarchy form the objects at the next layer. If we mimic this hierarchy as multiple layers and let the network learn the appropriate combination of features, we get what is called Deep Learning architecture. Thus, Deep Learning networks are hierarchical in nature.

Deep learning architectures have led to an incredible progress in computer vision tasks ranging from identifying and generating accurate masks around the objects to identifying spatial properties of an object. Mask-RCNN and RetinaNet architectures mainly led to this improvement.

Mask RCNNs have found their use in segmenting individual objects, *i.e.* creating masks for each object in an image. The input and output are both images. The architecture can also be used to do instance segmentation, *i.e.* identifying different objects of the same type in an image. Detectron, a Facebook AI Research (FAIR) software system, implements all these state-of-the-art object detection algorithms and is open source.

Some of the practical applications of CNNs are powering autonomous driving and analysing medical images.

Although the science and mathematics behind deep learning is fairly understood, there are still some interesting questions that require more research. These questions include: Why do architectures with multiple layers perform better, given that we can approximate any function with two layers? Why do CNNs work well with natural data such as speech, images, and text? How are we able to optimize non-convex functions so well? Why do over-parametrised architectures work?

Feature extraction consists of expanding the representational dimension such that the expanded features are more likely to be linearly separable; data points in higher dimensional space are more likely to be linearly separable due to the increase in the number of possible separating planes.

Earlier machine learning practitioners relied on high quality, hand crafted, and task specific features to build artificial intelligence models, but with the advent of Deep Learning, the models are able to extract the generic features automatically. Some common approaches used in feature extraction algorithms are highlighted below:

- Space tiling
- Random Projections
- Polynomial Classifier (feature cross-products)

- Radial basis functions
- Kernel Machines

Because of the compositional nature of data, learned features have a hierarchy of representations with increasing level of abstractions. For example:

- Images - At the most granular level, images can be thought of as pixels. Combination of pixels constitute edges which when combined forms textons (multi-edge shapes). Textons form motifs and motifs form parts of the image. By combining these parts together we get the final image.
- Text - Similarly, there is an inherent hierarchy in textual data. Characters form words, when we combine words together we get word-groups, then clauses, then by combining clauses we get sentences. Sentences finally tell us what story is being conveyed.
- Speech - In speech, samples compose bands, which compose sounds, which compose phones, then phonemes, then whole words, then sentences, thus showing a clear hierarchy in representation.

Learning representations

There are those who dismiss Deep Learning: if we can approximate any function with 2 layers, why have more?

For example: SVMs find a separating hyperplane “in the span of the data”, meaning predictions are based on comparisons to training examples. SVMs are essentially a very simplistic 2 layer neural net, where the first layer defines “templates” and the second layer is a linear classifier. The problem with 2 layer fallacy is that the complexity and size of the middle layer is exponential in N (to do well with a difficult task, need LOTS of templates). But if you expand the number of layers to $\log(N)$, the layers become linear in N. There is a trade-off between time and space.

An analogy is designing a circuit to compute a boolean function with no more than two layers of gates – we can compute **any boolean function** this way! But, the complexity and resources of the first layer (number of gates) quickly becomes infeasible for complex functions.

What is “deep”?

- An SVM isn’t deep because it only has two layers
- A classification tree isn’t deep because every layer analyses the same (raw) features
- A deep network has several layers and uses them to build a **hierarchy of features of increasing complexity**

How can models learn representations (good features)?

Manifold hypothesis: natural data lives in a low-dimensional manifold. Set of possible images is essentially infinite, set of “natural” images is a tiny subset. For example: for an image of a person, the set of possible images is on the order of magnitude of the number of face muscles they can move (degrees of freedom) ~ 50. An ideal (and unrealistic) feature extractor represents all the factors of variation (each of the muscles, lighting, *etc.*).

Q&A from the end of lecture:

- For the face example, could some other dimensionality reduction technique (*i.e.* PCA) extract these features?
    - Answer: would only work if the manifold surface is a hyperplane, which it is not

---

📝 *Marina Zavalina, Peeyush Jain, Adrian Pearl, Davida Kollmar*
27 Jan 2020