Introduction to Deep Learning Algorithms

See the following article for a recent survey of deep learning:

Yoshua Bengio, Learning Deep Architectures for AI, Foundations and Trends in Machine Learning, 2(1), 2009

Depth

The computations involved in producing an output from an input can be represented by a **flow graph**: a flow graph is a graph representing a computation, in which each node represents an elementary computation and a value (the result of the computation, applied to the values at the children of that node). Consider the set of computations allowed in each node and possible graph structures and this defines a family of functions. Input nodes have no children. Output nodes have no parents.

The flow graph for the expression $sin(a^2 + b/a)$ could be represented by a graph with two input nodes a and b, one node for the division b/a taking a and b as input (i.e. as children), one node for the square (taking only a as input), one node for the addition (whose value would be $a^2 + b/a)$ and taking as input the nodes $a^2$and b/a, and finally one output node computing the sinus, and with a single input coming from the addition node.

A particular property of such *flow graphs* is **depth**: the length of the longest path from an input to an output.

Traditional feedforward neural networks can be considered to have depth equal to the number of layers (i.e. the number of hidden layers plus 1, for the output layer). Support Vector Machines (SVMs) have depth 2 (one for the kernel outputs or for the feature space, and one for the linear combination producing the output).

Motivations for Deep Architectures

The main motivations for studying learning algorithms for deep architectures are the following:

- Insufficient depth can hurt

- The brain has a deep architecture
- Cognitive processes seem deep

Insufficient depth can hurt

Depth 2 is enough in many cases (e.g. logical gates, formal [threshold] neurons, sigmoid-neurons, Radial Basis Function [RBF] units like in SVMs) to represent any function with a given target accuracy. But this may come with a price: that the required number of nodes in the graph (i.e. computations, and also number of parameters, when we try to learn the function) may grow very large. Theoretical results showed that there exist function families for which in fact the required number of nodes may grow exponentially with the input size. This has been shown for logical gates, formal neurons, and RBF units. In the latter case Hastad has shown families of functions which can be efficiently (compactly) represented with $O(n)$ nodes (for n inputs) when depth is d, but for which an exponential number ($O(2^n)$) of nodes is needed if depth is restricted to $d - 1$.

One can see a deep architecture as a kind of factorization. Most randomly chosen functions can't be represented efficiently, whether with a deep or a shallow architecture. But many that can be represented efficiently with a deep architecture cannot be represented efficiently with a shallow one (see the polynomials example in the Bengio survey paper). The existence of a compact and deep representation indicates that some kind of structure exists in the underlying function to be represented. If there was no structure whatsoever, it would not be possible to generalize well.

The brain has a deep architecture

For example, the visual cortex is well-studied and shows a sequence of areas each of which contains a representation of the input, and signals flow from one to the next (there are also skip connections and at some level parallel paths, so the picture is more complex). Each level of this feature hierarchy represents the input at a different level of abstraction, with more abstract features further up in the hierarchy, defined in terms of the lower-level ones.

Note that representations in the brain are in between dense distributed and purely local: they are **sparse**: about 1% of neurons are active simultaneously in the brain. Given the huge number of neurons, this is still a very efficient (exponentially efficient) representation.

Cognitive processes seem deep

- Humans organize their ideas and concepts hierarchically.
- Humans first learn simpler concepts and then compose them to represent more abstract ones.
- Engineers break-up solutions into multiple levels of abstraction and processing

It would be nice to learn / discover these concepts (knowledge engineering failed because of poor introspection?). Introspection of linguistically expressible concepts also suggests a *sparse*representation: only a small fraction of all possible words/concepts are applicable to a particular input (say a visual scene).

Breakthrough in Learning Deep Architectures

Before 2006, attempts at training deep architectures failed: training a deep supervised feedforward neural network tends to yield worse results (both in training and in test error) then shallow ones (with 1 or 2 hidden layers).

Three papers changed that in 2006, spearheaded by Hinton's revolutionary work on Deep Belief Networks (DBNs):

- Hinton, G. E., Osindero, S. and Teh, Y., A fast learning algorithm for deep belief nets Neural Computation 18:1527-1554, 2006
- Yoshua Bengio, Pascal Lamblin, Dan Popovici and Hugo Larochelle, Greedy Layer-Wise Training of Deep Networks, in J. Platt et al. (Eds), Advances in Neural Information Processing Systems 19 (NIPS 2006), pp. 153-160, MIT Press, 2007
- Marc'Aurelio Ranzato, Christopher Poultney, Sumit Chopra and Yann LeCun Efficient Learning of Sparse Representations with an Energy-Based Model, in J. Platt et al. (Eds), Advances in Neural Information Processing Systems (NIPS 2006), MIT Press, 2007

The following key principles are found in all three papers:

- Unsupervised learning of representations is used to (pre-)train each layer.

- Unsupervised training of one layer at a time, on top of the previously trained ones. The representation learned at each level is the input for the next layer.
- Use supervised training to fine-tune all the layers (in addition to one or more additional layers that are dedicated to producing predictions).

The DBNs use RBMs for unsupervised learning of representation at each layer. The Bengio et al paper explores and compares RBMs and*auto-encoders* (neural network that predicts its input, through a bottleneck internal layer of representation). The Ranzato et al paper uses sparse auto-encoder (which is similar to *sparse coding*) in the context of a *convolutional* architecture. Auto-encoders and convolutional architectures will be covered later in the course.

Since 2006, a plethora of other papers on the subject of deep learning has been published, some of them exploiting other principles to guide training of intermediate representations. SeeLearning Deep Architectures for AI for a survey.