Statistical Machine Learning
Reading Assignment 4 Report
Deep Learning / DBNs

Alireza Sadeghi - Seyed Mohsen Shojaee

July 20, 2016

1 Introduction

There are a number of ways that the field of deep learning has been characterized. For example, in 1986, Rina Dechter introduced the concepts of first order deep learning and second order deep learning in the context of constraint satisfaction. Later, deep learning was characterized as a class of machine learning algorithms that

- use a cascade of many layers of nonlinear processing units for feature extraction and transformation. Each successive layer uses the output from the previous layer as input. The algorithms may be supervised or unsupervised and applications include pattern analysis (unsupervised) and classification (supervised).
- are based on the (unsupervised) learning of multiple levels of features or representations of the data. Higher level features are derived from lower level features to form a hierarchical representation.
- are part of the broader machine learning field of learning representations of data.
- learn multiple levels of representations that correspond to different levels of abstraction; the levels form a hierarchy of concepts.

These definitions have in common (1) multiple layers of nonlinear processing units and (2) the supervised or unsupervised learning of feature representations

in each layer, with the layers forming a hierarchy from low-level to high-level features. The composition of a layer of nonlinear processing units used in a deep learning algorithm depends on the problem to be solved. Layers that have been used in deep learning include hidden layers of an artificial neural network and sets of complicated propositional formulas. They may also include latent variables organized layer-wise in deep generative models such as the nodes in Deep Belief Networks and Deep Boltzmann Machines which will be discussed further in the report.

Deep learning algorithms transform their inputs through more layers than shallow learning algorithms. At each layer, the signal is transformed by a processing unit, like an artificial neuron, whose parameters are 'learned' through training. A chain of transformations from input to output is a credit assignment path (CAP). CAPs describe potentially causal connections between input and output and may vary in length. For a feedforward neural network, the depth of the CAPs, and thus the depth of the network, is the number of hidden layers plus one (the output layer is also parameterized). For recurrent neural networks, in which a signal may propagate through a layer more than once, the CAP is potentially unlimited in length. There is no universally agreed upon threshold of depth dividing shallow learning from deep learning, but most researchers in the field agree that deep learning has multiple nonlinear layers (CAP ¿ 2) and Schmidhuber considers CAP ¿ 10 to be very deep learning.

2 Concepts

Deep learning algorithms are based on distributed representations. The underlying assumption behind distributed representations is that observed data are generated by the interactions of factors organized in layers. Deep learning adds the assumption that these layers of factors correspond to levels of abstraction or composition. Varying numbers of layers and layer sizes can be used to provide different amounts of abstraction.

Deep learning exploits this idea of hierarchical explanatory factors where higher level, more abstract concepts are learned from the lower level ones. These architectures are often constructed with a greedy layer-by-layer method. Deep learning helps to disentangle these abstractions and pick out which features are useful for learning.

For supervised learning tasks, deep learning methods obviate feature engi-

neering, by translating the data into compact intermediate representations akin to principal components, and derive layered structures which remove redundancy in representation.

Many deep learning algorithms are applied to unsupervised learning tasks. This is an important benefit because unlabeled data are usually more abundant than labeled data. Examples of deep structures that can be trained in an unsupervised manner are neural history compressors and deep belief networks.

3 Restricted Boltzman Machines

A restricted Boltzmann machine (RBM) is a generative stochastic artificial neural network that can learn a probability distribution over its set of inputs.

RBMs were initially invented under the name Harmonium by Paul Smolensky in 1986, and rose to prominence after Geoffrey Hinton and collaborators invented fast learning algorithms for them (mid-2000s). RBMs have found applications in dimensionality reduction, classification, collaborative filtering, feature learning and topic modeling. They can be trained in either supervised or unsupervised ways, depending on the task.

As their name implies, RBMs are a variant of Boltzmann machines, with the restriction that their neurons must form a bipartite graph: a pair of nodes from each of the two groups of units (commonly referred to as the "visible" and "hidden" units respectively) may have a symmetric connection between them; and there are no connections between nodes within a group. By contrast, "unrestricted" Boltzmann machines may have connections between hidden units. This restriction allows for more efficient training algorithms than are available for the general class of Boltzmann machines, in particular the gradient-based contrastive divergence algorithm.

Restricted Boltzmann machines can be used in deep learning networks. In particular, *deep belief networks* can be formed by "stacking" RBMs and optionally fine-tuning the resulting deep network with gradient descent and backpropagation. We conclude this report by an overview on *deep belief networks*.

4 Deep Belief Networks

Deep belief nets are probabilistic generative models that are composed of multiple layers of stochastic, latent variables. The latent variables typically have

binary values and are often called hidden units or feature detectors. The top two layers have undirected, symmetric connections between them and form an associative memory. The lower layers receive top-down, directed connections from the layer above. The states of the units in the lowest layer represent a data vector.

The two most significant properties of deep belief nets are:

- There is an efficient, layer-by-layer procedure for learning the top-down, generative weights that determine how the variables in one layer depend on the variables in the layer above.
- After learning, the values of the latent variables in every layer can be inferred by a single, bottom-up pass that starts with an observed data vector in the bottom layer and uses the generative weights in the reverse direction.

In particular with DBNs, we are interested in solving two main problems

- Learning Problem (Adjust parameters to increase likelihood of data generation)
- Inference Problem (Infer state of unobserved variables - not covered in this report)

4.1 Learning in DBNs

Deep belief nets are learned one layer at a time by treating the values of the latent variables in one layer, when they are being inferred from data, as the data for training the next layer. This efficient, greedy learning can be followed by, or combined with, other learning procedures that fine-tune all of the weights to improve the generative or discriminative performance of the whole network.

We take an iterative, greedy approach to learning, since it's nearly impossible to infer the posterior distribution over all possible configurations of hidden variables, since V-Sturctures introduce new types of dependencies that in turn, make exact inference intractable (We call this the "explaining away" phenomena). Also the posterior depends on both the likelihood and the prior, therefore we need to know the values of W in higher layers. And finally, we need to marginalize the distribution on all configurations of variables in higher levels which in turn leaves us with an extremely hard to compute integral.

In order to overcome the challenges mentioned above, we propose the following algorithm for DBN learning:

- Train a restricted Boltzmann machine on X to obtain its weight matrix, W. Use this as the weight matrix between the lower two layers of the network.
- Transform X by the RBM to produce new data X', either by sampling or by computing the mean activation of the hidden units.
- Repeat this procedure with $X \leftarrow X'$ for the next pair of layers, until the top two layers of the network are reached.
- Fine-tune all the parameters of this deep architecture with respect to a proxy for the DBN log- likelihood, or with respect to a supervised training criterion (after adding extra learning machinery to convert the learned representation into supervised predictions, e.g. a linear classifier).

Discriminative fine-tuning can be performed by adding a final layer of variables that represent the desired outputs and backpropagating error derivatives. When networks with many hidden layers are applied to highly-structured input data, such as images, backpropagation works much better if the feature detectors in the hidden layers are initialized by learning a deep belief net that models the structure in the input data (Hinton and Salakhutdinov, 2006).

4.2 DBNs as a composition of simple learning modules

A deep belief net can be viewed as a composition of simple learning modules each of which is a restricted type of Boltzmann machine that contains a layer of visible units that represent the data and a layer of hidden units that learn to represent features that capture higher-order correlations in the data. The two layers are connected by a matrix of symmetrically weighted connections, W , and there are no connections within a layer. Given a vector of activities v for the visible units, the hidden units are all conditionally independent so it is easy to sample a vector, h , from the factorial posterior distribution over hidden vectors, $p(h|v, W)$. It is also easy to sample from $p(v|h, W)$. By starting with an observed data vector on the visible units and alternating several times between sampling from $p(h|v, W)$ and $p(v|h, W)$, it is easy to get a learning signal. This signal is simply the difference between the pairwise correlations of the visible and hidden units at the beginning and end of the sampling.