Deep Learning and Natural Language Processing (NLP)

Saba Sabrin

Why Deep learning?

Earlier use of machine learning problems was mostly handled by human designed features and through different classification algorithms. Raw data processing has always been troublesome because of low dimensionality, less structure and noise. It wasn't easy to come up with a model for solving image or text processing. Deep learning is the new era to solve these machine learning problems with the help of neural networks.

Deep Neural Networks (DNN)

Neural networks are simply biologically inspired models of brain neurons. It has the mechanism of learning and adapting with different models. Based on specially that criteria, neural networks are used for solving competitive classification, regression and prediction based problems. A simple neural network takes some inputs, transforms the inputs, then adjust different parameters to generate correct results. The adjusting part is the most critical part overall. This part normally helps the network to learn. A network having multiple hidden layers are normally considered as Deep Neural Networks. These hidden layers are the key to learn representations from the data through a training process. Basically, deep neural networks can model complex non-linear relationships by composing set of lower level features to generate higher level features. After an efficient training, these networks behave well in real-time. Below is some discussion about different types of neural networks and how they learn, trained and used in real world.

Backpropagation is the main technique behind the adjustment of input parameters and to modify them with some added values called weights. The weight values are calculated depending on different activation functions. There are different types of activation functions such as, "Sigmoid", "Tanh", "Relu" etc. Depending on the problem, the functions are used and weights are generated through different algorithms. One such popular technique is Stochastic gradient descent. Initially, for every input, weight values are summed up and passed through an activation function. For a single pass through the network with all the inputs and weights, an output is calculated. This output is then compared with the target output, normally for supervised learning. A cost optimization or error function is used to measure the error between those outputs and finally, for every iteration through the network, it tries to low down the loss gradually. Though it's a simple procedure, but the calculation normally take time depending on the training set.

Feed-Forward Neural Network is normally fully connected and can be used for classification or regression problems. There can be multiple hidden units which can transform the data at every layer. The reason of naming this network as feed-forward is that, this network only passes data in a forward direction through the network. Being the simplest network, it has all the features of a standard neural network such, activation functions & cost optimization. The input and output dimensions are fixed for this type of networks. Though, the hidden units might differ. This network doesn't perform well when the input size is unknown, especially for sequences.

Recurrent Neural Network, LSTMs are good for processing sequence type data. As we know normal feed-forward networks cannot come back to their previous state as it only works in forwarding direction. Unlike to this, RNNs shares some recurrent weight through the network which behaves like it's internal memory and based on that it calculated the activation functions, weights and outputs at every time step. It's an acyclic process for unknown sequence of data. There are also some problems related to this work as it shares a large amount of data throughout the training phase. Training is also a little tricky because of vanishing gradient problem or exploding gradient problems. Long Short Term Memory (LSTM) technique has been proposed to overcome the problems of RNN architectures. LSTM has recurrent gates which helps the errors to flow backwards even for unlimited number of time steps/layers. This helps to learn very deep learning tasks for sequences or events happened before long back. One more variant of RNN architecture is Bi-directional RNN which works both with past and future context. Outputs of two RNNs are joined in a way so that the combined output could be used for predictions. This can also be trained jointly with LSTM.

NLP applications are highly depended on deep learning architectures. There are various areas such as, language modelling, domain adaptation, language translation, speech recognition, part-of-speech tagging, random text generation and more. One important use of DNNs are also to generate efficient word embeddings for NLP based tasks. There are different models like, Skip-Gram or Continuous Bag-of-words for generating efficient and useful word vectors. But, the question is why do we need word vectors for NLP task. The reason is to capture the dimensionality and relation between different words. It could be a syntactic or semantic relationship within a vocabulary of words. Nowadays, Google has introduced the term Thought vectors which points towards the distributed representation of the words. These representations are learned through simple neural networks without hidden layers and by adding a projection layer instead.

There are numerous research going on in this area specially, LSTMs are showing great results these days solving different NLP based issues and can be also combined with other network architectures.

References

1. DL4NLP course content, TU Darmstadt, https://moodle.informatik.tu-darmstadt.de/course/view.php?id=19