Report

Egirin Gega & Samuli Romo

January 19, 2020

Contents

List of Figures

List of Tables

1 Convolutional Neural Network

A Convolutional Neural Network also known as a CNN or COMV NET is an artificial neural network that is so far been most popularly used for analyzing images. Although image analysis has been the most widespread use of CNN's they can also be used for other data analysis or classification problems as well most generally.

1.1 How does it Work

We can think of a CNN as an artificial neural network that has some type of specialization for being able to pick out or detect patterns and make sense of them. This pattern detection is what makes CNN so useful for image analysis so if a CNN is just some form neural network what differentiates it from just a standard multi-layer perceptron or MLP well these layers are precisely what makes a CNN a CNN.

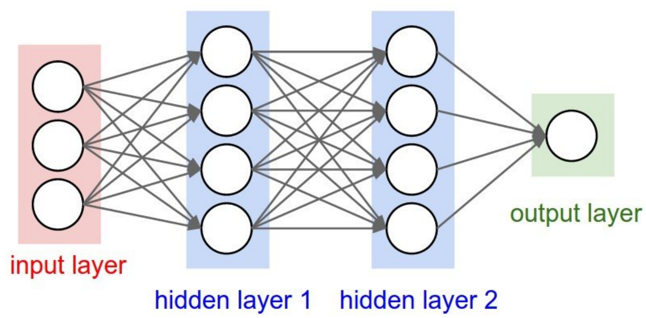

Figure 1: Simple Representation.

CNN's can and usually do have other non convolutional layers as well but the basis of a CNN is the convolutional layers all right so what do these convolutional layers do just like any other layer a convolutional layer receives input then transforms the input in some way and then outputs the transform input to the next layer with a convolutional layer this transformation is a convolution operation.

Most generally we can think of a CNN as an artificial neural network that has some type of specialization for being able to pick out or detect patterns and make sense of them. This pattern detection is what makes CNN so useful for image analysis.So if a CNN is just some form of an artificial neural network what differentiates it from just a standard multi-layer perceptron or MLP?

Well a CNN has hidden layers called convolutional layers and these layers are precisely what makes a CNN well a CNN. Now CNN's can and usually do have other non convolutional layers as well but the basis of a CNN is the convolutional layers.

2 Recurrent Neural Network

Recurrent neural networks are learning model with a simple structure and a built-in feedback loop, allowing it to act as a forecasting engine. Recurrent neural networks, or RNNs, have a long history, but their recent popularity is mostly due to the works of Juergen Schmidhuber, Sepp Hochreiter, and Alex Graves. Their applications are extremely versatile, ranging from speech recognition to driverless cars.

2.1 How does it Work

All the nets we've seen up to this point have been feedforward neural networks. In a feedforward neural network, signals flow in only one direction from input to output, one layer at a time. In a recurrent net, the output of a layer is added to the next input and fed back into the same layer, which is typically the only layer in the entire network. You can think of this process as a passage through time – shown here are 4 such time steps. At t = 1, the net takes the output of time t = 0 and sends it back into the net along with the next input. The net repeats this for t = 2, t = 3, and so on. Unlike feedforward nets, a recurrent net can receive a sequence of values as input, and it can also produce a sequence of values as output. The ability to operate with sequences opens up these nets to a wide variety of applications.

Here are a few examples. When the input is singular and the output is a sequence, a potential application is image captioning. A sequence of inputs with a single output can be used for document classification. When both the input and output are sequences, these nets can classify videos frame by frame. If a time delay is introduced, the net can statistically forecast the demand in supply chain planning. Have you ever used an RNN for one of these applications? If so, please comment and share your experiences. Like we've seen with previous deep learning models, by stacking RNNs on top of each other, you can form a net capable of more complex output than a single RNN working alone.

Typically, an RNN is an extremely difficult net to train. Since these nets use backpropagation, we once again run into the problem of the vanishing gradient.

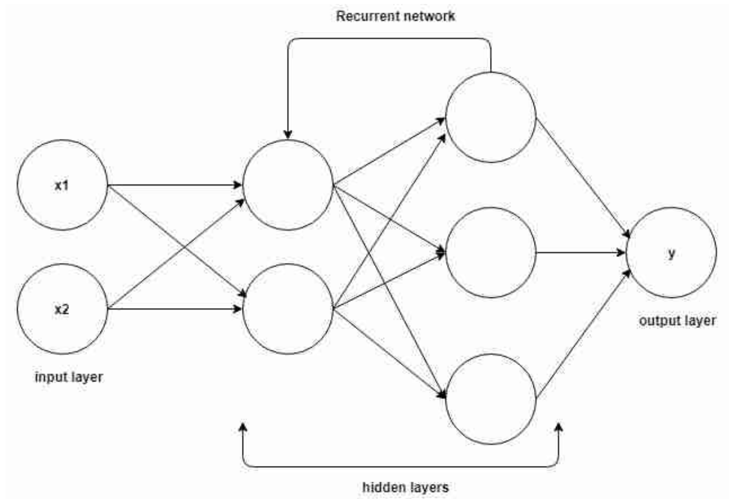

Figure 2: Recurrent Neural Network.

Unfortunately, the vanishing gradient is exponentially worse for an RNN. The reason for this is that each time step is the equivalent of an entire layer in a feedforward network. So training an RNN for 100 time steps is like training a 100-layer feedforward net – this leads to exponentially small gradients and a decay of information through time. There are several ways to address this problem - the most popular of which is gating. Gating is a technique that helps the net decide when to forget the current input, and when to remember it for future time steps. The most popular gating types today are GRU and LSTM. Besides gating, there are also a few other techniques like gradient clipping, steeper gates, and better optimizers.

When it comes to training a recurrent net, GPUs are an obvious choice over an ordinary CPU. This was validated by a research team at Indico, which uses these nets on text processing tasks like sentiment analysis and helpfulness extraction. The team found that GPUs were able to train the nets 250 times faster! That's the difference between one day of training, and over eight months! So under what circumstances would you use a recurrent net over a feedforward net? We know that a feedforward net outputs one value, which in many cases was a class or a prediction. A recurrent net is suited for time series data, where an output can be the next value in a sequence, or the next several values. So the answer depends on whether the application calls for classification, regression, or forecasting.

3 Time series problem

In order of increasing complexity we can think of time series data as a sequence of data points that measure the same thing over an ordered period of time. Another way of thinking about it, is that it's a series of numeric values, each with its own timestamp defined by a name and a set of label dimensions.

We're seeing this type of data set become more common, in fact if we look at developer usage patterns in the past two years time series databases have emerged as the fastest growing category of databases.

Time series forecasting is performed in nearly every organization that works with quantifiable data. Retail stores forecast sales. Energy companies forecast reserves, production, demand, and prices. Educational institutions forecast enrollment. Goverments forecast tax receipts and spending. International financial organizations forecast inflation and economic activity. The list is long but the point is short - forecasting is a fundamental analytic process in every organization. The purpose of this example is to help with visualization of a Time Series Graph of an airline Company.

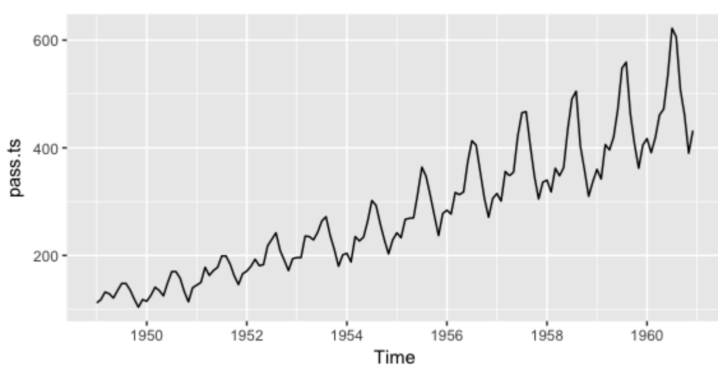

Figure 3: TimeSeries Data Example.

Well smart homes monitor data points like the temperature, number of people in the house and energy usage, Amazon monitors how its many assets are moving across the world with a fine-grained level of precision and efficiency that makes same-day delivery possible. All of these applications have one thing in common, they rely on a form of data that measures how things change over time.

Usually time series data sets have three commonalities. The data that does arrive is recorded as a new entry, it arrives in time order and time is a primary axis. These data sets are generally append-only the data that has been reported doesn't change. Since it was recorded at some point in the past, time series data sets are different than just having a time field as a column in a data set. In that, when we collect a new data point for time series data we have to create a brand new row for it. Only by doing this will we be able to track all changes to a system over time. Recording data points over a series of time allows us to analyze how something has changed in the past monitor, how is changing in the presence and even predict how it could change in the future.

Recurrent neural networks are able to learn how to make predictions in sequences of data and a variance of recurrent networks called long-short term memory networks are able to learn from even longer sequences of data. We can treat our data set as a supervised problem if we'd like and use an LS TM network to predict our target.