Crash Course in Recurrent Neural Networks for Deep Learning

by **Jason Brownlee** on July 7, 2016 in **Long Short-Term Memory Networks**

There is another type of neural network that is dominating difficult machine learning problems that involve sequences of inputs called recurrent neural networks.

Recurrent neural networks have connections that have loops, adding feedback and memory to the networks over time. This memory allows this type of network to learn and generalize across sequences of inputs rather than individual patterns.

A powerful type of Recurrent Neural Network called the Long Short-Term Memory Network has been shown to be particularly effective when stacked into a deep configuration, achieving state-of-the-art results on a diverse array of problems from language translation to automatic captioning of images and videos.

In this post you will get a crash course in recurrent neural networks for deep learning, acquiring just enough understanding to start using LSTM networks in Python with Keras.

After reading this post, you will know:

- The limitations of Multilayer Perceptrons that are addressed by recurrent neural networks.
- The problems that must be addressed to make Recurrent Neural networks useful.
- The details of the Long Short-Term Memory networks used in applied deep learning.

Let’s get started.

Crash Course in Recurrent Neu
Photo by Martin Fisch,

Support For Sequences in Neural Net

There are some problem types that are best framed involving either a sequence as an input or an output.

For example, consider a univariate time series problem, like the price of a stock over time. This dataset can be framed as a prediction problem for a classical feedforward multilayer Perceptron network by defining a windows size (e.g. 5) and training the network to learn to make short term predictions from the fixed sized window of inputs.

This would work, but is very limited. The window of inputs adds memory to the problem, but is limited to just a fixed number of points and must be chosen with sufficient knowledge of the problem. A naive window would not capture the broader trends over minutes, hours and days that might be relevant to making a prediction. From one prediction to the next, the network only knows about the specific inputs it is provided.

Univariate time series prediction is important, but there are even more interesting problems that involve sequences.

Consider the following taxonomy of sequence problems that require a mapping of an input to an output (taken from Andrej Karpathy).

- **One-to-Many**: sequence output, for image captioning.
- **Many-to-One**: sequence input, for sentiment clas
- **Many-to-Many**: sequence in and out, for machine

- **Synched Many-to-Many**: synced sequences in and out, for video classification.

We can also see that a one-to-one example of input to output would be an example of a classical feedforward neural network for a prediction task like image classification.

Support for sequences in neural networks is an important class of problem and one where deep learning has recently shown impressive results State-of-the art results have been using a type of network specifically designed for sequence problems called recurrent neural networks.

Recurrent Neural Networks

Recurrent Neural Networks or RNNs are a special typ ms.

Given a standard feed-forward multilayer Perceptron network, a recurrent neural network can be thought of as the addition of loops to the architecture. For example, in a given layer, each neuron may pass its signal latterly (sideways) in addition to forward to the next layer. The output of the network may feedback as an input to the network with the next input vector. And so on.

The recurrent connections add state or memory to the network and allow it to learn broader abstractions from the input sequences.

The field of recurrent neural networks is well established with popular methods. For the techniques to be effective on real problems, two major issues needed to be resolved for the network to be useful.

1. How to train the network with Backpropagation.
2. How to stop gradients vanishing or exploding during training.

1. How to Train Recurrent Neural Networks

The staple technique for training feedforward neural networks is to back propagate error and update the network weights.

Backpropagation breaks down in a recurrent neural network, because of the recurrent or loop connections.

This was addressed with a modification of the Backpropagation technique called Backpropagation Through Time or BPTT.

Instead of performing backpropagation on the recurrent network as stated, the structure of the network is unrolled, where copies of the neurons that have recurrent connections are created. For example a single neuron with a connection to itself (A->A) could be represented as two neurons with the same weight values (A->B).

This allows the cyclic graph of a recurrent neural network to be turned into an acyclic graph like a classic feed-forward neural network, and Backpropagation can be applied.

2. How to Have Stable Gradients During Training

When Backpropagation is used in very deep neural ne the
gradients that are calculated in order to update the we

They can become very large numbers called explodin
vanishing gradient problem. These large numbers in t ,
making training unstable and the network unreliable.

This problem is alleviated in deep multilayer Perceptro er
function, and even more exotic but now less popular a
layers.

In recurrent neural network architectures, this problem ture
called the Long Short-Term Memory Networks that all

Long Short-Term Memory Networks

The Long Short-Term Memory or LSTM network is a recurrent neural network that is trained using Backpropagation Through Time and overcomes the vanishing gradient problem.

As such it can be used to create large (stacked) recurrent networks, that in turn can be used to address difficult sequence problems in machine learning and achieve state-of-the-art results.

Instead of neurons, LSTM networks have memory blocks that are connected into layers.

A block has components that make it smarter than a classical neuron and a memory for recent sequences. A block contains gates that manage the block's state and output. A unit operates upon an input sequence and each gate within a unit uses the sigmoid activation function to control whether they are triggered or not, making the change of state and addition of information flowing through the unit conditional.

There are three types of gates within a memory unit:

- **Forget Gate**: conditionally decides what information to discard from the unit.
- **Input Gate**: conditionally decides which values fr

- **Output Gate**: conditionally decides what to output based on input and the memory of the unit.

Each unit is like a mini state machine where the gates of the units have weights that are learned during the training procedure.

You can see how you may achieve a sophisticated learning and memory from a layer of LSTMs, and it is not hard to imagine how higher-order abstractions may be layered with multiple such layers.

Resources

We have covered a lot of ground in this post. Below are some resources that you can use to go deeper into the topic of recurrent neural networks for deep learning.

Resources to learn more about Recurrent Neural Net

- Recurrent neural network on Wikipedia
- Long Short-Term Memory on Wikipedia
- The Unreasonable Effectiveness of Recurrent Ne
- Understanding LSTM Networks
- Deep Dive into Recurrent Neural Nets
- A Beginner’s Guide to Recurrent Networks and L

Popular tutorials for implementing LSTMs.

- LSTMs for language modeling with TensorFlow
- RNN for Spoken Word Understanding in Theano
- LSTM for sentiment analysis in Theano

Primary sources on LSTMs.

- Long short-term memory [pdf], 1997 paper by Hochreiter and Schmidhuber
- Learning to forget: Continual prediction with LSTM, 2000 by Schmidhuber and Cummins that add the forget gate
- On the difficulty of training Recurrent Neural Networks [pdf], 2013

People to follow doing great work with LSTMs.

- Alex Graves
- Jürgen Schmidhuber
- Ilya Sutskever
- Tomas Mikolov

Summary

In this post you discovered sequence problems and recurrent neural networks that can be used to address them.

Specifically, you learned:

- The limitations of classical feedforward neural networks and how recurrent neural networks can overcome these problems.
- The practical problems in training recurrent neural networks and how they are overcome.
- The Long Short-Term Memory network used to create deep recurrent neural networks.

Do you have any questions about deep recurrent neural networks, LSTMs or about this post? Ask your question in the comments and I will do my best to answer.

About Jason Brownlee

Dr. Jason Brownlee is a husband, proud father, academic researcher, author, professional developer and a machine learning practitioner. He is dedicated to helping developers get started and get good at applied machine learning. Learn more.

View all posts by Jason Brownlee →

< Applied Deep Learning in Python Mini-Course

How to Transform Your Machine Learning Data in Weka >

4 Responses to *Crash Course in Recurrent Neural Networks for Deep Learning*

Marcel August 8, 2016 at 8:34 pm # REPLY

Thanks a million Jason! Machine Learning Mastery is exactly what I need! 🙂

Jason Brownlee August 15, 2016 at 9:38 am # REPLY

You're welcome Marcel.

Supriya June 16, 2017 at 9:29 pm #

I am doing chatbot implementation using Kera
added in sequential model. Can you guide me how ma
implementation? Is it good to have more LSTM nodes

Jason Brownlee June 17, 2017 at 7:26 am #

We cannot know what model will be best for a given problem analytically. You must discover it analytically.

My best advice on evaluating deep learning algorithms is here:
http://machinelearningmastery.com/evaluate-skill-deep-learning-models/

Leave a Reply

Name (required)

Email (will not be published) (required)

Website

SUBMIT COMMENT

Welcome to Machine Learning Mastery

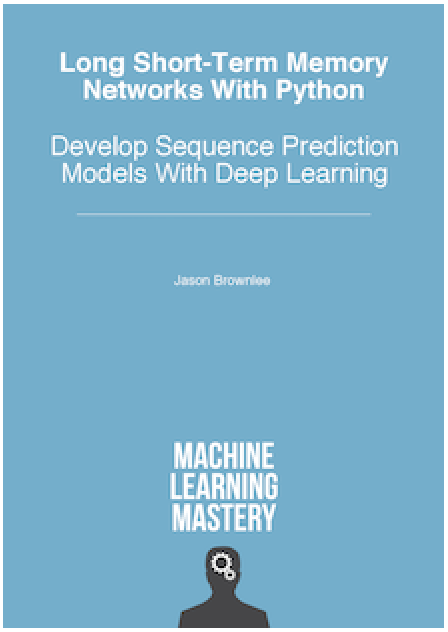
Hi, I'm Dr. Jason Brownlee.
My goal is to make practitioners like YOU

Read More

Deep Learning for S

Cut through the math
Discover 4 Models, 6 Arch

Get Started With LSTMs in Python Today!


Long Short-Term Memory Networks With Python
Develop Sequence Prediction Models With Deep Learning
Jason Brownlee
MACHINE LEARNING MASTERY

POPULAR

Time Series Prediction with LSTM Recurrent Neural Network

JULY 21, 2016

Your First Machine Learning Project in Python Step-By-Step
JUNE 10, 2016

Develop Your First Neural Network in Python With Keras Step-By-Step
MAY 24, 2016

Sequence Classification with LSTM Recurrent Neural Networks in Python with Keras
JULY 26, 2016

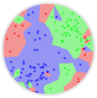
Multi-Class Classification Tutorial with the Keras D
JUNE 2, 2016

How to Setup a Python Environment for Machine L
MARCH 13, 2017

Regression Tutorial with the Keras Deep Learning I
JUNE 9, 2016

How to Run Your First Classifier in Weka
FEBRUARY 17, 2014

Time Series Forecasting with the Long Short-Term Memory Network in Python
APRIL 7, 2017


Tutorial To Implement k-Nearest Neighbors in Python From Scratch
SEPTEMBER 12, 2014

Privacy | Contact | About