Sequence modelling using neural networks

RNN's, GRU's and LSTM's

Exebit-2018

Indian Institute of Techonology Madras

Apr 14^{th}

Outline

Introduction

Motivation

- Fully connected networks and convolution networks cannot handle dynamic inputs
- All inputs have the exact same size.
- How do we handle audio or text?
- Use RNN's, LSTM's and GRU's[1]

[1] Ronald J Williams and David Zipser. "A learning algorithm for continually running fully recurrent neural networks". In: *Neural computation* 1.2 (1989), pp. 270–280.

Motivation

- Fully connected networks and convolution networks cannot handle dynamic inputs
- All inputs have the exact same size.
- How do we handle audio or text?
- Use RNN's, LSTM's and GRU's[1]

[1] Ronald J Williams and David Zipser. "A learning algorithm for continually running fully recurrent neural networks". In: *Neural computation* 1.2 (1989), pp. 270–280.

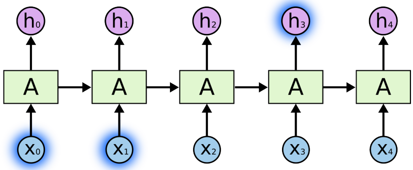
Motivation

- Fully connected networks and convolution networks cannot handle dynamic inputs
- All inputs have the exact same size.
- How do we handle audio or text?
- Use RNN's, LSTM's and GRU's[1]

[1] Ronald J Williams and David Zipser. "A learning algorithm for continually running fully recurrent neural networks". In: *Neural computation* 1.2 (1989), pp. 270–280.

Motivation

- Fully connected networks and convolution networks cannot handle dynamic inputs
- All inputs have the exact same size.
- How do we handle audio or text?
- Use RNN's, LSTM's and GRU's[1]



[1] Ronald J Williams and David Zipser. "A learning algorithm for continually running fully recurrent neural networks". In: *Neural computation* 1.2 (1989), pp. 270–280.

Natural language

- Natural language processing(NLP) is one domain with vast applications for sequence modelling.
- DL has completely changed the landscape for computer vision
- Images are comparatively easy to handle since
  - Images have symmetries to exploit
  - Pixels can take only 256 values, the English vocabulary has nearly 1 million words with varied meanings.
  - Inputs have a fixed size
  - Language has idioms, expressions, collocations and has evolved over time into a complex system.

Natural language

- Natural language processing(NLP) is one domain with vast applications for sequence modelling.
- DL has completely changed the landscape for computer vision
- Images are comparatively easy to handle since
  - Images have symmetries to exploit
  - Pixels can take only 256 values, the English vocabulary has nearly 1 million words with varied meanings.
  - Inputs have a fixed size
  - Language has idioms, expressions, collocations and has evolved over time into a complex system.

Applications of DL in NLP

- *Classification of text* : Utility for sentiment/emotion analysis.
- *Sequence-2-Sequence Models* : Transliteration of text or extractive summarization.
- *Text generation* : Generating a corpus of text.
- *Audio recognition/processing* : Converting audio output to text.
- *Video processing* : Processing sequence of images.

Applications of DL in NLP

- *Classification of text* : Utility for sentiment/emotion analysis.
- *Sequence-2-Sequence Models* : Transliteration of text or extractive summarization.
- *Text generation* : Generating a corpus of text.
- *Audio recognition/processing* : Converting audio output to text.
- *Video processing* : Processing sequence of images.

Applications of DL in NLP

- *Classification of text* : Utility for sentiment/emotion analysis.
- *Sequence-2-Sequence Models* : Transliteration of text or extractive summarization.
- *Text generation* : Generating a corpus of text.
- *Audio recognition/processing* : Converting audio output to text.
- *Video processing* : Processing sequence of images.

Applications of DL in NLP

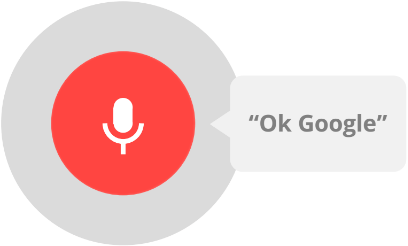
- *Classification of text* : Utility for sentiment/emotion analysis.
- *Sequence-2-Sequence Models* : Transliteration of text or extractive summarization.
- *Text generation* : Generating a corpus of text.
- *Audio recognition/processing* : Converting audio output to text.
- *Video processing* : Processing sequence of images.

Applications of DL in NLP

- *Classification of text* : Utility for sentiment/emotion analysis.
- *Sequence-2-Sequence Models* : Transliteration of text or extractive summarization.
- *Text generation* : Generating a corpus of text.
- *Audio recognition/processing* : Converting audio output to text.
- *Video processing* : Processing sequence of images.

Applications of DL in NLP

Google's Speech recognition



MS-Coco dataset for visual question answering

Figure: How many leftover donuts is the Red bicycle holding?

Recurrent Neural Networks

The Intuition

- RNN consists of a state S_t at any point t
- The input at time t is denoted by X_t
- The output is denoted by H_t

- Note that the recurrent core($\mathcal{A}$) is the same for all t

[2] *Colah's Blog on Understanding LSTM's.* https://colah.github.io/posts/2015-08-Understanding-LSTMs.

The Intuition

- RNN consists of a state S_t at any point t
- The input at time t is denoted by X_t
- The output is denoted by H_t

- Note that the recurrent core($\mathcal{A}$) is the same for all t

[2] *Colah's Blog on Understanding LSTM's.* https://colah.github.io/posts/2015-08-Understanding-LSTMs.

The Intuition

- RNN consists of a state S_t at any point t
- The input at time t is denoted by X_t
- The output is denoted by H_t

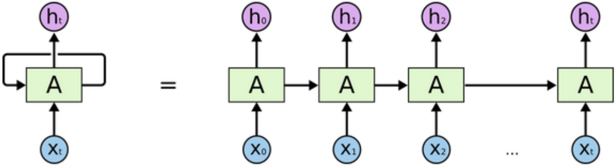


Figure: The RNN structure[2]

- Note that the recurrent core($\mathcal{A}$) is the same for all t

[2] *Colah's Blog on Understanding LSTM's.* https://colah.github.io/posts/2015-08-Understanding-LSTMs.

The Intuition

- RNN consists of a state S_t at any point t
- The input at time t is denoted by X_t
- The output is denoted by H_t

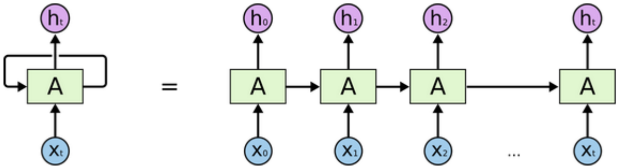


Figure: The RNN structure[2]

- Note that the recurrent core($\mathcal{A}$) is the same for all t

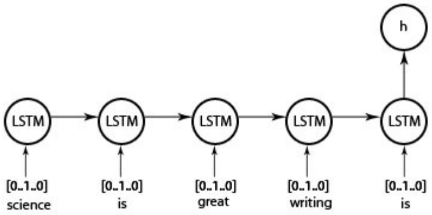
[2] *Colah's Blog on Understanding LSTM's.* https://colah.github.io/posts/2015-08-Understanding-LSTMs.

The Intuition

- The output H_t need not be utilized at all steps.
    -

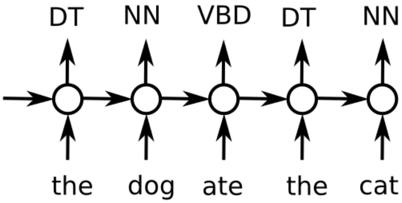
The Intuition

- The output H_t need not be utilized at all steps.
  - Only the last H_T is utilized and trained on(Grammatical correctness of sentence)

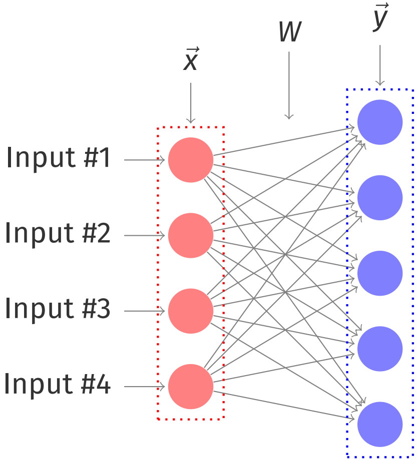


The Intuition

- The output H_t need not be utilized at all steps.
  - $h_0 \cdots h_T$ are are utilized to generate an output.(Finding POS tags for every word)



The Notation - Fully Connected Layer

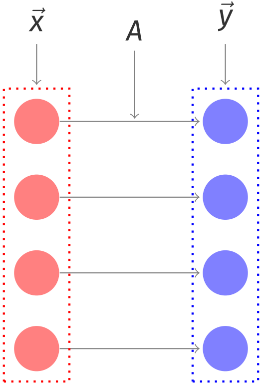


Let $\vec{x}$ be a vector of size n_1 and $\vec{y}$ be a vector of size n_2. Let W be a matrix of dimensions $n_2 \times n_2$. Consider the following relation between $\vec{x}$ and $\vec{y}$:

$$\vec{y} = W\vec{x}$$

This is represented by the feed-forward layer presented on the left.

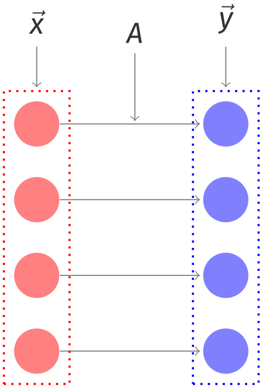
The Notation - Activation Layer



Let $\vec{x}$ be a vector of size n and $\vec{y}$ be a vector of size n. Let A be an activation function from $n \to n$. Consider the following relation between $\vec{x}$ and $\vec{y}$:

$$\vec{y} = A(\vec{x})$$

The Notation - Activation Layer



Let $\vec{x}$ be a vector of size n and $\vec{y}$ be a vector of size n. Let A be an activation function from $n \rightarrow n$. Consider the following relation between $\vec{x}$ and $\vec{y}$:

$$\vec{y} = A(\vec{x})$$

The activation function is typically ReLu sigmoid or TanH. For example, if A was ReLu we have :

$$\vec{y} = [y_1, y_2, \cdots, y_n]$$
$$\vec{x} = [x_1, x_2, \cdots, x_n]$$
$$y_i = max(x_i, 0) \quad \forall i \in [1, \cdots, n]$$

The Equations of an RNN[3]

The the primary idea behind RNN's is encapsulated by the following equation

$$s^{(t)} = f(s^{(t-1)}, x^{(t)}; \theta)$$

The Neural Network equations

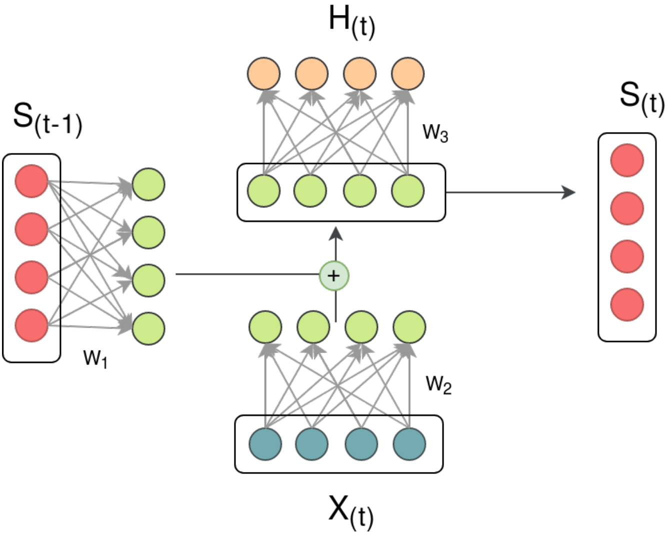
Consistent with the notation earlier, S_t is the state, X_t is the input and H_t is the output.

$$\vec{s}_{(t)} = tanH\left(\mathbf{W_1}\vec{s}_{(t-1)} + \mathbf{W_2}\vec{x}_{(t)} + \mathbf{b_1}\right)$$
$$\vec{o}_{(t)} = \mathbf{W_3}\vec{s}_{(t)} + \mathbf{b_2}$$

The parameters to train/tune are $\mathbf{W_1}, \mathbf{W_2}, \mathbf{W_3}, \mathbf{b_1}, \mathbf{b_2}$

[3] Ian Goodfellow, Yoshua Bengio and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

A Pictorial Representation



Points to ponder

- Note that the parameters are shared(like a CNN) which contributes to the sucess of this model.
- Some unanswered questions :
  1. How do we use the outputs $H_{(t)}$?
  2. What are the complications with gradient descent?
  3. What loss functions do we train with ?
- RNN, like CNNs can be used as a modular components. One can generate another sequence or feed the output to another neural network.

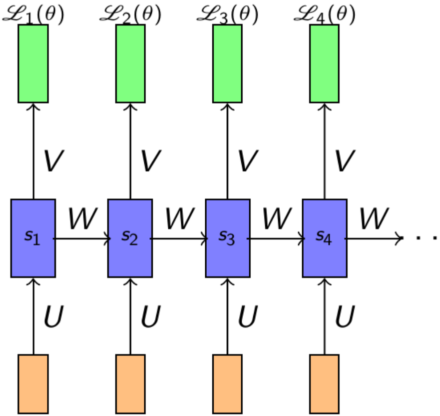
Points to ponder

- Note that the parameters are shared(like a CNN) which contributes to the sucess of this model.
- Some unanswered questions :
  1. How do we use the outputs $H_{(t)}$?
  2. What are the complications with gradient descent?
  3. What loss functions do we train with ?
- RNN, like CNNs can be used as a modular components. One can generate another sequence or feed the output to another neural network.

Points to ponder

- Note that the parameters are shared(like a CNN) which contributes to the sucess of this model.
- Some unanswered questions :
  1. How do we use the outputs $H_{(t)}$?
  2. What are the complications with gradient descent?
  3. What loss functions do we train with ?
- RNN, like CNNs can be used as a modular components. One can generate another sequence or feed the output to another neural network.

The Loss Function



The loss function $\mathcal{L}$ is given by

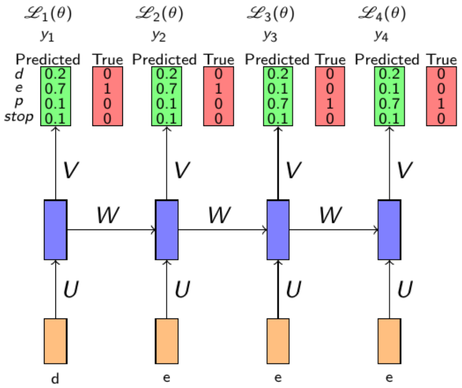
$$\mathcal{L} = \mathcal{L}_1(\theta) + \mathcal{L}_2(\theta) + \cdots + \mathcal{L}_n(\theta)$$

Consider a task where we have to predict a sequence of characters, given an input of characters.(ex : Spell correction)

How do you utilize the output layer $H_{(t)}$?

Note that unlike this example, $\mathcal{L}_i$ can also be zero and we have only one non-zero $\mathcal{L}_T$.

The Loss Function - Sequence prediction



Transform the output $H_{(t)}$ into a probability distribution using a softmax layer i.e :

$$Q_{(t)} = \text{softmax}(H_{(t)})$$

The loss function is the cross-entropy ($\sum_n -p_n log(q_n)$). Hence the loss is given by.

$$\begin{aligned}\mathcal{L} &= \mathcal{L}_1(\theta) + \mathcal{L}_2(\theta) + \mathcal{L}_3(\theta) + \mathcal{L}_4(\theta) \\ &= \sum_{t=1}^{4} -\ln(Q_{(t)}(\text{True class})_t) \\ &= -\ln(0.7) - \ln(0.7) \\ &\quad - \ln(0.7) - \ln(0.7)\end{aligned}$$

Backkpropagation through time

- The expression for the gradients is complicated. Why?

$$s_{(t)} \propto Ws_{(t-1)} + W^2 s_{(t-2)} \cdots + W^t s_{(0)}$$

- Hence, the expression for $\frac{\partial \mathcal{L}}{\partial W}$ has terms involving derivative of power of matrix.
- A good reference for more of the math : https://www.deeplearningbook.org/contents/rnn.html
- We let TensorFlow handle derivatives with it's auto-differentiation :)
- The derivatives and above equation expose a major drawback of RNN's

Backkpropagation through time

- The expression for the gradients is complicated. Why?

$$s_{(t)} \propto W s_{(t-1)} + W^2 s_{(t-2)} \cdots + W^t s_{(0)}$$

- Hence, the expression for $\frac{\partial \mathcal{L}}{\partial W}$ has terms involving derivative of power of matrix.
- A good reference for more of the math : `https://www.deeplearningbook.org/contents/rnn.html`
- We let TensorFlow handle derivatives with it's auto-differentiation :)
- The derivatives and above equation expose a major drawback of RNN's

Backkpropagation through time

- The expression for the gradients is complicated. Why?

$$s_{(t)} \propto Ws_{(t-1)} + W^2 s_{(t-2)} \cdots + W^t s_{(0)}$$

- Hence, the expression for $\frac{\partial \mathcal{L}}{\partial W}$ has terms involving derivative of power of matrix.
- A good reference for more of the math : `https://www.deeplearningbook.org/contents/rnn.html`
- We let TensorFlow handle derivatives with it's auto-differentiation :)
- The derivatives and above equation expose a major drawback of RNN's

Long-Short Term Memory Networks

Drawback's of RNN

- The expressions for the derivative diminish with time. Hence the effect of derivative of $\mathcal{L}_{\mathcal{T}}$ has a smaller effect on $S_{(k)}$ if k is farther away.
- The state information from earlier time steps are progressively diluted. There is no mechanism for retaining state information from earlier time steps.

$$S_{(t)} \propto Ws_{(t-1)} + W^2 s_{(t-2)} \cdots + W^t s_{(0)}$$

- There is no ability to reject the information from the newest time-step, even if it is irrelevant.

Drawback's of RNN

- The expressions for the derivative diminish with time. Hence the effect of derivative of $\mathcal{L}_{\mathcal{T}}$ has a smaller effect on $S_{(k)}$ if k is farther away.
- The state information from earlier time steps are progressively diluted. There is no mechanism for retaining state information from earlier time steps.

$$S_{(t)} \propto Ws_{(t-1)} + W^2 s_{(t-2)} \cdots + W^t s_{(0)}$$

- There is no ability to reject the information from the newest time-step, even if it is irrelevant.

Drawback's of RNN

- The expressions for the derivative diminish with time. Hence the effect of derivative of $\mathcal{L}_{\mathcal{T}}$ has a smaller effect on $S_{(k)}$ if k is farther away.
- The state information from earlier time steps are progressively diluted. There is no mechanism for retaining state information from earlier time steps.

$$S_{(t)} \propto Ws_{(t-1)} + W^2 s_{(t-2)} \cdots + W^t s_{(0)}$$

- There is no ability to reject the information from the newest time-step, even if it is irrelevant.

Motivating LSTM's[4]

- i_t : Input gate that filters the input x_t.
- o_t : Output gate that determines the information to be passed on to the next time step.
- f_t : Forget gate that retains only relevant information from the information passed on.
- All the “gates” are vectors with same size as state vector.

[4] Sepp Hochreiter and Jürgen Schmidhuber. “Long short-term memory”. In: *Neural computation* 9.8 (1997), pp. 1735–1780.

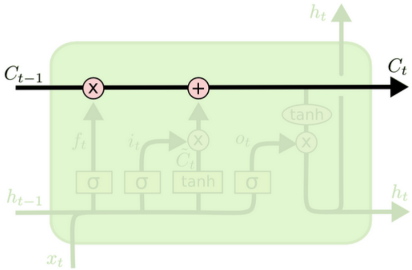
Motivating LSTM's[4]

- i_t : Input gate that filters the input x_t.
- o_t : Output gate that determines the information to be passed on to the next time step.
- f_t : Forget gate that retains only relevant information from the information passed on.
- All the "gates" are vectors with same size as state vector.

[4] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780.

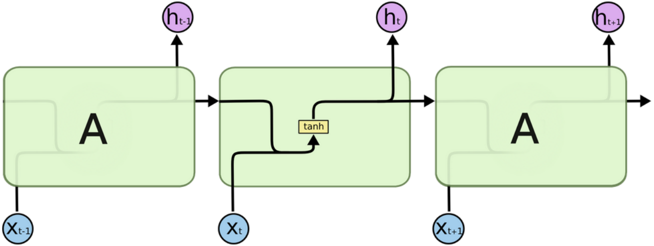
The Notation

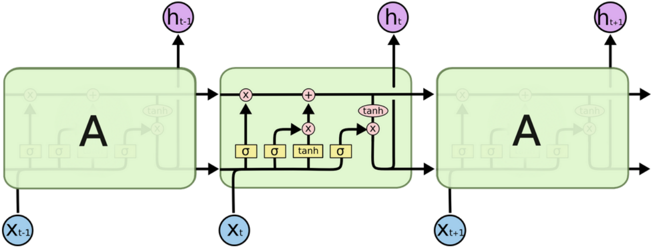
- Instead of a single state s_t, we utilize two states. One maintaining the long-term state(c_t), while the other maintains the short-term context(s_t).



- The long-term context C_t allows information to be easily passed on. This also allows the gradients to backpropagate for a longer period of time.

Comparing RNNs and LSTMs





The Equations

The LSTM structure

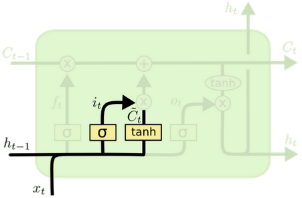
The equation for the gates are :

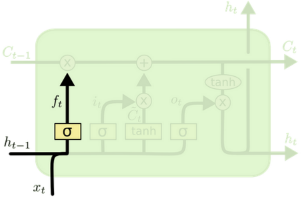
$$\begin{aligned} o_t &= \sigma(W_o h_{t-1} + U_o x_t + b_o) \\ i_t &= \sigma(W_i h_{t-1} + U_i x_t + b_i) \\ f_t &= \sigma(W_f h_{t-1} + U_f x_t + b_f) \end{aligned}$$

The equation for the states are :

$$\begin{aligned} \bar{c}_t &= \sigma(W_1 h_{t-1} + W_2 x_t + b_1) \\ c_t &= f_t \odot c_{t-1} + i_t \odot \bar{c}_t \\ s_t &= o_t \odot \sigma(c_t) \\ h_t &= \begin{cases} s_t \\ W_3 s_t + b_2 \end{cases} \end{aligned}$$

Visualizing the Equations



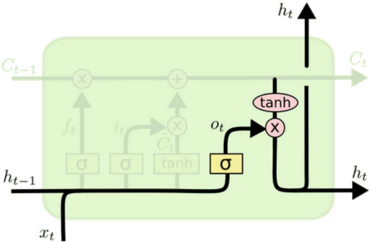


The input gate is used to filter the input content X_t and S_t and write into the long term context C_t.

The forget gate is used to filter the long term context C_t based on the short term context and current input X_t.

The two are used to modify the long term context.

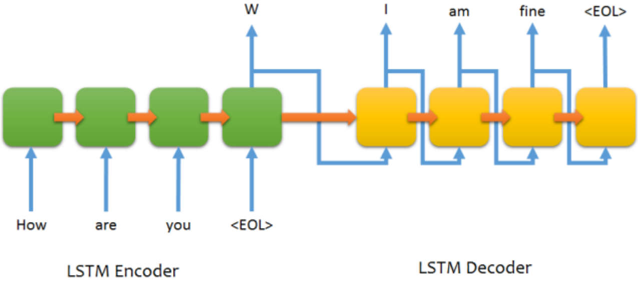
Visualizing the Equations



The output gate is used to determine the new local context, output vectors(h_t and s_t).

Encoder decoder Models

Two LSTM's. The first LSTM generate an output H_T which is passed to the second LSTM.
The S_0 of the second LSTM is set to be H_T. Then the first output is generated.



This first output is fed as input to the next step. This process is repeated till a <STOP> token is an output of the model.

How do you feed text to the model?

Let the vocabulary be of size $|\mathcal{V}|$. Assign every word a unique ID. For example, let the entire vocabulary be : ['Hello', 'World', 'Deep', 'Learning'].

We cannot use this unique ID as the reprentation, this assumes ordering among words. Instead use a one-hot vector representation.

Let the ID('hello') = 0 and ID('Learning') = 3. Then the representations are :

$$\vec{W}(\text{Hello}) = [1, 0, 0, 0]$$
$$\vec{W}(\text{Learning}) = [0, 0, 0, 1]$$

Some Resources

1. A gentle introduction to RNN[5]
2. A nice intro to LSTM's[6]
3. Some overview on RNN's and encoder models[7]
4. Attention Mechanisms[8]

---

[5] *Jason Brownlee's Blog on RNN's.* https://machinelearningmastery.com/rnn-unrolling/.

[6] *Colah's Blog on Understanding LSTM's.* https://colah.github.io/posts/2015-08-Understanding-LSTMs.

[7] *Jason Brownlee's Blog on Encoder-decoders.* https://machinelearningmastery.com/encoder-decoder-recurrent-neural-network-models-neural-machine-translation/.

[8] *WildML's Blog on Attention Mechanisms.* http://www.wildml.com/2016/01/attention-and-memory-in-deep-learning-and-nlp/.

References

- *Colah's Blog on Understanding LSTM's.* https://colah.github.io/posts/2015-08-Understanding-LSTMs.
- Goodfellow, Ian, Yoshua Bengio and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.
- Hochreiter, Sepp and Jürgen Schmidhuber. "Long short-term memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780.
- *Jason Brownlee's Blog on Encoder-decoders.* https://machinelearningmastery.com/encoder-decoder-recurrent-neural-network-models-neural-machine-translation/.
- *Jason Brownlee's Blog on RNN's.* https://machinelearningmastery.com/rnn-unrolling/.
- *WildML's Blog on Attention Mechanisms.* http://www.wildml.com/2016/01/attention-and-