Natural Language Processing: Predictive Models Using Recurrent Neural Networks

W. Andy Holst

School of Professional Studies, Northwestern University

MSDS 458: Artificial Intelligence and Deep Learning

Dr. Syamala Srinivasan

August 8, 2021

Abstract

This paper addresses how the Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) RNN concepts provide exceptional performance in sequence-based data such as natural language (text). In our experiments we were able to achieve **90.2% Test Accuracy** with LSTM neural network techniques. Building and deploying predictive models using RNN and LSTM neural networks, which benefits today's information retrieval systems such as Google, has developed remarkably quickly in the past few years and is an ever important facet of machine learning and artificial intelligence.

Natural Language Processing: Predictive Models Using Recurrent Neural Networks

With the advent of computers in the mid twentieth century, the idea of using these machines to "perform useful tasks involving human language" was simultaneously conceived (Jurafsky & Martin, 2014 p. 1). Two almost uniquely human traits are our ability to communicate information and knowledge to our fellow humans, as well as to use tools and technology to enhance our own brains beyond the singular. It is through all of the different human languages that exist that our communication occurs. So, it naturally follows that teaching computers how to communicate and perform useful tasks as a tool is a desirable outcome. That, in a nutshell, is what Natural Language Processing (NLP) is all about.

According to Jurafsky and Martin, there are many aspects of NLP which must be considered. Within just four short years since the publication of "Attention Is All You Need" by researchers from Google Brain (Vaswani et al., 2017), Natural Language Processing (NLP) has entered the mainstream of human society. The paper has been cited over 23,000 times on Google Scholar. Access to information and text documents on the Internet has exploded in speed and popularity thanks to the ability of search engines to quickly and accurately categorize information so that it can be found and consumed. In this research paper, I analyze a specific subset of Neural Networks - called Recurrent Neural Networks - which prove effective for conducting multi-class classification of news articles. This type of advance in NLP has allowed Google (and others) to place information at our fingertips through extremely fast information retrieval and organization of information across the internet.

There will obviously be more significant developments in Natural Language Processing in the coming years and decades. While we may not yet have achieved full intelligence in computers, we are exceedingly close to the vision of having machines "perform useful tasks involving human language".

Literature Review

The business case for natural language processing is quite obvious, with the largest technology companies -- Apple, Google, Amazon, Facebook -- all leading developments in the processing of text and enriching our lives through technology. As mentioned above, the publication of “Attention is All You Need” (Vaswani et al., 2017) was pivotal for NLP, so I reviewed this publication and have considered many aspects of it in this research paper.

Data

The dataset used for this research -- AG News Subset -- is a collection of more than 1 million news articles. According to Tensorflow, news articles have been gathered from more than 2000 news sources by ComeToMyHead in more than 1 year of activity. ComeToMyHead is an academic news search engine which has been running since July, 2004. The dataset is provided by the academic community for research purposes in data mining (clustering, classification, etc), information retrieval (ranking, search, etc), xml, data compression, data streaming, and any other non-commercial activity (Tensorflow, 2021).

The data preparation, exploratory data analysis and visualization steps undertaken to prepare the dataset for subsequent research, analysis, and modeling were as follows:

- A Google Colaboratory Notebook (Python 3.7.11) was created, with GPU enabled.
- Python Packages imported as necessary for pre-processing and modeling, including: Numpy (1.19.5), Pandas (1.1.5), Tensorflow (2.5.0), Sci-kit Learn (0.22.2) (*for Confusion Matrix*), Matplolib (3.2.2). Seaborn (0.11.1), and Time.
- Importing the AG News Subset database of news articles (127,600 in total), which is available using the Tensorflow API: `tfds.load('ag_news_subset'...`
- Splitting the dataset into Training (114,000 articles), Validation (6,000), Test (7,600).

Methods: Research Design and Modeling

Over the course of the fifth and sixth weeks of the Summer Term 2021, I conducted a total of thirty (30) experiments on the AG News Subset dataset, all of which consisted of a multiclass classification problem that categorizes news articles into four different types. The dataset was split into a Training Set of `114,000` news articles, a Validation Set of `6,000` news articles, and a Test Set of `7,600` news articles. Each article is labeled as one of 4 different types of news articles: `['World', 'Sports', 'Business', 'Sci/Tech']`, and are all evenly distributed in count. The neural network models explored in these experiments consisted initially of single layer recurrent neural networks, exploring a range of hyperparameters. Later models utilized the LSTM concept, and different hyperparameters explored included: varying vocabulary size (1000 words, 2000 words, 3000 words, 10000 words), choosing to keep or eliminate stop words (*Note: I intended to incorporate this feature but I could not find documentation for TextVectorizer that allowed for stop words*), varying output sequence length (64, 128, and 256), using single- versus 2-layers of LSTM, and using a unidirectional versus a bidirectional approach

All models were constructed in a `Sequential` layer-fashion using Keras (Python library was built by and described eloquently by Francois Chollet in “Deep Learning with Python”) (Chollet, 2018). The input layer consists of an `Embedding` layer with the size of the layer equal to the vocabulary size (1000, 2000, 3000 and 10000). The single hidden layers are all recurrent or LSTM (`SimpleRNN / LSTM`) layers with `relu` activation function. The last layer uses a `softmax` activation with `4` nodes, outputting a probability distribution over the 4 different output classes. The loss function used in these experiments was `sparse_categorical_crossentropy`, and the optimization algorithm used was `Adam`. A

validation set was created by setting apart `7,600` samples from the original training data, to minimize overfitting.

Because this problem involves news articles, and the evolution of this type of data is expected to be frequently updated as more and more news proliferates the Internet, it is expected that this model will require periodic updating. However, as Chollet outlines in his text *Deep Learning with Python,* there are several means of “freezing” portions of models and utilizing them for “transfer learning.” Thus, updating of these types of news classification models is expected to be infrequent and will only enhance performance of future LSTMs.

NATURAL LANGUAGE PROCESSING

The specifications of each of the twenty-seven (27) Experiments conducted is listed below:

Experiment	Vocab Size	Output Sequence Length	Model Type	Layers	Directionality	Regularization
1	1000	64	SimpleRNN	1	Uni	ES
2	1000	128	SimpleRNN	1	Uni	ES
3	1000	256	SimpleRNN	1	Uni	ES
4	1000	64	SimpleRNN	1	Bi	ES
5	1000	128	SimpleRNN	1	Bi	ES
6	1000	256	SimpleRNN	1	Bi	ES
7	1000	64	LSTM	1	Uni	ES
8	1000	128	LSTM	1	Uni	ES
9	1000	256	LSTM	1	Uni	ES
10	1000	64	LSTM	1	Bi	ES
11	1000	128	LSTM	1	Bi	ES
12	1000	256	LSTM	1	Bi	ES
13	1000	64	LSTM	2	Uni	ES/Drop
14	1000	128	LSTM	2	Uni	ES/Drop
15	1000	256	LSTM	2	Uni	ES/Drop
16	1000	64	LSTM	2	Bi	ES/Drop
17	1000	128	LSTM	2	Bi	ES/Drop
18	1000	256	LSTM	2	Bi	ES/Drop
22	2000	64	LSTM	2	Bi	ES/Drop
23	2000	128	LSTM	2	Bi	ES/Drop
24	2000	256	LSTM	2	Bi	ES/Drop
25	3000	64	LSTM	2	Bi	ES/Drop
26	3000	128	LSTM	2	Bi	ES/Drop
27	3000	256	LSTM	2	Bi	ES/Drop
28 - Best	**10000**	**64**	**LSTM**	**2**	**Bi**	**ES/Drop**
29	10000	128	LSTM	2	Bi	ES/Drop
30	10000	64	LSTM	3	Bi	ES/Drop

Note: Experiments 19-21 omitted due to error code with CNN model.

Results

The following Table produces final results for all twenty-seven (27) Experiments:

Experiment	Build Time (sec)	Train Accuracy (%)	Train Loss	Valid. Accuracy (%)	Valid. Loss	Test Accuracy (%)	Test Loss
1	1721	86.3	0.38	85.0	0.41	84.4	0.44
2	931	86.2	0.38	85.4	0.41	85.0	0.42
3	1908	89.0	0.30	84.5	0.45	83.3	0.47
4	2569	86.9	0.36	86.3	0.39	85.2	0.40
5	2632	88.4	0.32	85.4	0.41	85.0	0.42
6	1971	87.6	0.34	85.5	0.41	84.4	0.43
7	324	86.3	0.38	86.0	0.39	85.3	0.41
8	345	86.3	0.38	85.9	0.39	84.7	0.41
9	302	86.5	0.37	86.0	0.38	85.3	0.40
10 *	362	86.4	0.38	85.8	0.40	84.7	0.41
11	294	86.4	0.37	86.5	0.38	85.2	0.40
12	872	88.0	0.33	86.7	0.37	86.1	0.38
13	265	87.1	0.36	85.9	0.41	85.1	0.40
14	131	85.6	0.42	85.5	0.42	85.0	0.42
15	428	88.0	0.34	86.0	0.40	85.9	0.40
16	354	86.7	0.37	86.3	0.37	85.8	0.39
17	602	89.3	0.30	87.3	0.35	86.1	0.41
18	252	86.3	0.39	87.4	0.35	85.2	0.40
22	240	88.5	0.34	86.8	0.36	87.8	0.34
23	476	91.1	0.25	88.3	0.34	88.4	0.35
24	483	90.1	0.26	88.6	0.32	88.4	0.34
25	442	91.8	0.23	88.9	0.33	89.3	0.34
26	407	91.9	0.23	88.9	0.33	88.8	0.34
27	326	90.2	0.28	88.7	0.32	88.7	0.33
28 - BEST	**238**	**92.8**	**0.22**	**90.2**	**0.31**	**90.2**	**0.30**
29	704	96.7	0.09	88.6	0.49	88.4	0.49
30	366	92.4	0.23	89.0	0.33	89.5	0.31

Note: Experiments 19-21 omitted due to error code with CNN model.

Note 1: Due to an error code while attempting to fit the CNN model (see below) I was unable to get a CNN to properly fit the data:

```
InvalidArgumentError: 2 root error(s) found.
  (0) Invalid argument:  assertion failed: [Condition x == y did not hold element-wise:] [x (sparse_categorical_crossentropy/SparseSoftmaxCrossEntropyWithLogits/Shape_1:0) = ] [64 1] [y
(sparse_categorical_crossentropy/SparseSoftmaxCrossEntropyWithLogits/strided_slice:0) = ] [64 27]
         [[node sparse_categorical_crossentropy/SparseSoftmaxCrossEntropyWithLogits/assert_equal_1/Assert/Assert (defined at <ipython-input-69-c347a707ca2f>:6) ]]
         [[gradient_tape/sequential_11/embedding_14/embedding_lookup/Reshape/_76]]
  (1) Invalid argument:  assertion failed: [Condition x == y did not hold element-wise:] [x (sparse_categorical_crossentropy/SparseSoftmaxCrossEntropyWithLogits/Shape_1:0) = ] [64 1] [y
(sparse_categorical_crossentropy/SparseSoftmaxCrossEntropyWithLogits/strided_slice:0) = ] [64 27]
         [[node sparse_categorical_crossentropy/SparseSoftmaxCrossEntropyWithLogits/assert_equal_1/Assert/Assert (defined at <ipython-input-69-c347a707ca2f>:6) ]]
```

Note 2: Due to an error encountered while attempting to produce a visualization of the hidden layers of the LSTM Model using a technique called t-SNE .

```
ValueError: Graph disconnected: cannot obtain value for tensor KerasTensor(type_spec=TensorSpec(shape=(None, 1), dtype=tf.string, name='text_vectorization_2_input'),
name='text_vectorization_2_input', description="created by layer 'text_vectorization_2_input'") at layer "text_vectorization_2". The following previous layers were accessed without issue:
['text_vectorization_2']
```

Analysis and Interpretation

The following steps were instrumental towards increasing model performance: First of all, increasing the vocabulary size resulted in a decent increase in model performance. For experiments conducted, the best performance occurred with a vocabulary size of 10,000 words. Output sequence length did not appear to have any substantial impact on performance. The most prominent improvement noticed was the use a 2-layer LSTM Neural Network, with Bidirectional Learning. Moving from an RNN to an LSTM Model both boosted performance as well as ***dramatically*** reduced the model build time. The best performing models used Early Stopping and one 50% Drop Out Layer to prevent overfitting. The best model performance was achieved in Experiment 28 with a **Test Accuracy of 90.2%**.

With Model 28, Figure 1 below shows the loss and accuracy graphs for this model. The model converges very quickly (in 238 secords, after only 2 epochs). *There may be some improvements to this model to increase performance and better control overfitting, perhaps by changing batch size, which should be considered in future experiments.*

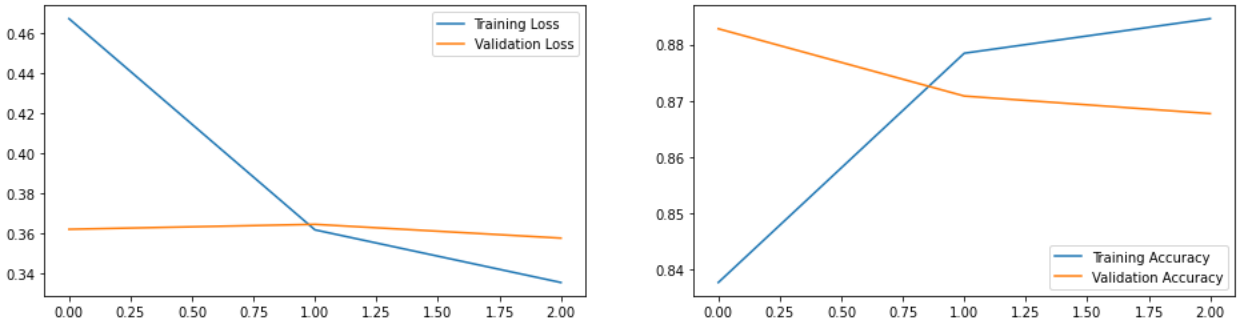

Figure 1 - Training and Validation Loss and Accuracy Plots for Model 28 “BEST”

The Confusion Matrix for Model 28, depicting the predicted results for the first 15 news articles from the Test Dataset, is shown in Figure 2 below. This model is achieving quite good results in the Confusion Matrix, with only one of the 15 results dropping below 95% (Test #5).

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
World	0.01%	99.99%	0.01%	0.53%	99.99%	0.08%	2.93%	0.12%	99.89%	0.07%	100.00%	0.11%	0.31%	97.32%	99.99%
Sports	99.99%	0.00%	0.00%	99.46%	0.00%	0.01%	96.72%	99.88%	0.00%	0.00%	0.00%	0.00%	0.00%	0.07%	0.00%
Business	0.00%	0.01%	0.17%	0.01%	0.00%	5.63%	0.14%	0.00%	0.05%	99.89%	0.00%	0.04%	99.64%	0.09%	0.00%
Sci/Tech	0.00%	0.00%	99.83%	0.01%	0.00%	94.28%	0.22%	0.00%	0.05%	0.04%	0.00%	99.85%	0.05%	2.52%	0.00%

Figure 2 - Confusion Matrix (first 15 results) for Model 28 “BEST”

Conclusions

This paper addressed how the Recurrent Neural Network (RNN) and an extension of this type of model -- the Long Short-Term Memory RNN (LSTM) provides exceptional performance in computer vision multi-class classification models over Deep Neural Networks. In our experiments we were able to achieve **90.2% Test Accuracy** with relatively convolutional neural network techniques. Building and deploying predictive models using RNN and LSTM neural networks, which benefits today’s information retrieval systems such as Google, has developed remarkably quickly in the past few years and is an ever important facet of machine learning and artificial intelligence.

References

Jurafsky, D., & Martin, J. H. (2014). Chapter 1: Introduction. In *Speech and language processing* (Second Edition, pp. 1–16). essay, Pearson Prentice Hall.

Tensorflow, (2021, June 30). *Ag_News_Subset : Tensorflow datasets*. TensorFlow. https://www.tensorflow.org/datasets/catalog/ag_news_subset.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017, December 6). *Attention Is All You Need*. arXiv.org. https://arxiv.org/abs/1706.03762v5.