Assignment #3: NLP with AG News

MSDS 458 Artificial Intelligence and Deep Learning

Alison Au

August 6, 2022

Abstract

Market research Company X seeks an automated method to classify their vast collection of articles into topic categories. With a more organized system, their researchers can more easily find relevant information for their projects. The AG newswire dataset is leveraged to build a neural network for this purpose. In this phase of the project, 14 models are tested. They are primarily bidirectional LSTM given their historic effectiveness, although a simple RNN and 1-dimensional CNN are evaluated for comparison. The neural networks vary in terms of number of nodes, layers, and text preparation approaches. The impact of these various levers is investigated in order to find the most accurate text classification neural network.

Introduction

Company X is a market research firm with an archive of thousands of articles. They aim to classify the articles into topic categories using machine learning. This would help researchers more easily find information relevant to their projects. This assignment is the first stage to explore different neural networks' effectiveness in text classification. The objective is to maximize accuracy in topic classification.

The AG newswire dataset is used in this initial phase, as it is a substantial sample of easily accessible, labeled data. It contains a collection of news articles across over 2,000 news sources. There are 120,000 training and 7,600 test samples evenly divided across the 4 largest classes (Business, Science/Tech, Sports, and World).

Literature Review

The AG newsire dataset is used for a variety of purposes including text mining (clustering, classification), information retrieval (ranking, search), and data compression. In this case, the main focus is natural language processing (NLP) for classification. Historically,

language systems were built using handcrafted rules like if/then/else. This encountered many issues given the complexity of human language. In the late 1980s, machine learning approaches began to be applied such as decision trees and logistic regression. Only recently around 2013 did recurrent neural networks become popular for NLP, picking up patterns in text. Bidirectional LSTM was the go-to model, which is the primary architecture examined in this project. More recently around 2017-2018, transformers arose as the state-of-the-art approach for many NLP tasks. In this project phase, LSTM, simple RNN, and a 1-dimensional CNN will be evaluated to better understand how neural networks work for text classification.

Methods

The AG newswire dataset is already divided into 120,000 news articles for training and 7,600 for testing. There are 4 classes, Business, Science/Tech, Sports, and World with 30,000 training samples and 1,900 test samples in each class. A validation set of 6,000 articles was carved out of the training set (5%), leaving 114,000 articles for training the model. These were converted into TensorFlow datasets with a batch size of 32.

Examining the training set, the articles contain between 3 to 173 words with a normal distribution centered around 30. In total across the corpus, there are 3.7M words and 93K unique words. The most common words were “the”, “a”, and “to” (see Figure 1). Given their poor predictive power, stop words were removed. The most common words excluding these are “39s” (representing apostrophe ‘s’), “said”, and “new.”

Fourteen models were run using Tensorflow’s Keras in the Google Colab Pro environment with GPU enabled. In pre-processing, the data was vectorized and standardized. This includes lowercasing all words, stripping punctuation, splitting each sample into words, and indexing the tokens. The words were embedded with the pre-trained GloVe embeddings (Global

Vectors for Word Representation). All models utilized accuracy as the performance metric, RMSprop for the optimizer, and sparse categorical cross-entropy for the loss function since the labels were not one-hot encoded. Softmax with 4 nodes was used in the output layer since this is a single-label classification with 4 classes. All LSTM models (experiments 1-12) used bidirectional layers. Each model was set to run for 200 epochs but with early stopping if validation accuracy did not improve after 3 epochs. The experiments can be broken down as follows:

- Experiments 1-3 tested the number of words per document (30, 40, 60) with a single-layer, bidirectional LSTM
- Experiments 4-6 evaluated the vocabulary sizes 2,000, 3,000, and 4,000 as compared to 1,000 in experiments 1-3
- Experiment 7 used GloVe embeddings with 200 dimensions to compare to 100 dimensions in prior experiments
- Experiments 8-9 tested 64 and 128 nodes in the single-layer LSTM as compared to 32 nodes in prior models
- Experiments 10-12 examined 2-layer, bidirectional LSTMs
- Experiments 13 and 14 evaluated a simple RNN and a 1-dimensional CNN, respectively

Results

A performance summary table of the 14 experiments is in Figure 2, and findings are detailed below.

Text Preprocessing

In NLP, EDA and data preprocessing are important steps to understand the text and impact the model performance. One criterion to decide is how many tokens to use per document. The length of the articles is an approximate normal distribution centered around 30 words (see Figure 3). Since we do not want too few tokens (captures too little information) nor too many tokens (leads to many zeroes from padding), the number of words per document tested were 30, 40, and 60. These were extracted after removing stop words, using a vocabulary of 1,000 most frequent words, and GloVe embedding of 100 dimensions. This was then fed into a bidirectional LSTM model with one hidden layer of 32 nodes.

Varying the number of tokens did not change test accuracy significantly. Using the first 60 words per article resulted in the best accuracy of 86.2%, compared to 30 words which had 85.6% accuracy (0.6% difference). An output sequence length of 60 was optimal to extract enough information without being too long. Interestingly, its loss and accuracy curves showed the most underfitting despite having the most words. All three models had similar training times of around 6 minutes.

Vocabulary size had more of an impact than the volume of words per document. While the entire corpus had 93K unique words, a small subset of 1,000, 2,000, 3,000, and 4,000 words were evaluated for the vocabulary size. As the size increased, test accuracy improved. A vocabulary of 1,000 most common words generated 86.2% accuracy, while 4,000 words achieved 89.3% accuracy (3% difference). This makes sense since a more expansive vocabulary allows more meaning or patterns to be extracted from each article. This did also cause slightly more overfitting. Training time did increase from 6 to 7-8 minutes, although this was worth the improvement in performance.

The improvement did diminish as the vocabulary size increased. The increase from 1,000 to 2,000 words produced a 2.2% increase in accuracy compared to just 0.3% going from 3,000 to 4,000 words. Additionally, even though 4% or fewer of the corpus' total unique words was used in the models, this was sufficient to attain close to 90% classification accuracy. This indicates the strong predictive power of the most common words (excluding stop words) in extracting the topic of articles.

The final text preprocessing examined was changing the GloVe embedding from 100 to 200 dimensions. This produced minimal improvement of 0.3% in accuracy to 89.6%. However, training time did not increase despite the greater number of dimensions. Overall, changing the dimensions within the same pre-trained embedding does not seem to impact much. Given these findings, all subsequent models use 60 words per document, a vocabulary size of 4,000, and GloVe embedding with 200 dimensions.

Number of Hidden Nodes and Layers

After experimenting with text preprocessing, various network architectures were evaluated next. When testing the number of hidden neurons and layers, the former made a greater impact on performance.

LSTM models with a single bidirectional layer of 32, 64, and 128 nodes were compared. Accuracy generally improved as the number of nodes increased, as they could capture more representation of the text data. However, the improvement diminished with each successive increase, indicating a point where increasing nodes would not be beneficial. The network with 128 nodes performed best across all 14 models tested in this project, achieving a test accuracy of 90.0%. This was 0.4% points better than the model with 32 hidden nodes. It also took 1 minute and 30 seconds longer to train at 8 minutes 30 seconds total, relatively not a significant amount

of time. The main concern is the increase in overfitting. By the 2nd epoch, training accuracy started to exceed validation with a large 4% difference by the time the model stopped training. The model would be more generalizable with more regularization in addition to the 50% dropout implemented.

LSTM networks with 2 bidirectional layers were also tested, although their accuracy did not exceed that of the 1-layer, 128-node model. Having a hidden layer with 128 nodes followed by another with 64 nodes produced 89.8% training accuracy, 0.3% lower than the model with just 128 nodes in a single layer. It also took almost 4 more minutes to train and had similar issues of overfitting. Two layers with 64 and 32 nodes performed even worse, likely because fewer nodes capture less patterns in the data.

Experiment 12 tested adding two hidden dense layers with 128 and 64 nodes and sigmoid activation to the two bidirectional LSTM layers with 128 and 64 nodes. This slightly improved test accuracy by 0.2% compared to the model without the hidden dense layers. It generated an accuracy of 89.96%, slightly below the best model (128 nodes in single layer). However, it took 48% longer to train at 12:34 vs. 8:31. Consequently, a single bidirectional LSTM layer is sufficient and evidently can produce the strongest results.

Neural Network Type

In addition to the LSTM models, a simple RNN and a 1-dimensional CNN were evaluated. The RNN contained two hidden layers with 128 and 64 nodes. This produced the worst performance among all experiments with just 85.2% accuracy. The processing time was also the longest by far at 54 minutes. This supports the importance of gated cells in LSTM to control what to remember versus forget. Only the important information is retained. In contrast, simple RNNs keep all the information and use the same weights across the layers, leading to

longer training time and issues with vanishing/exploding gradients. This validates why LSTM models became the go-to industry approach over simple RNNs.

A 1-dimensional CNN architecture with two hidden layers (64 and 32 nodes) performed better than the simple RNN but subpar to the LSTM models. “Layer” here refers to the combined convolution and max pooling layers. It attained a test accuracy of 87.8%. CNN is evidently still effective despite scanning the text for nearby correlations, in this case using a kernel size of 3, rather than using a “memory” approach of RNN/LSTM. It also took the least computing power, requiring only 4 minutes and 29 seconds to train. The model also showed underfitting, suggesting that it could be refined further for even better accuracy. A 1D CNN could be suitable for NLP depending on the use case.

Conclusions

In summary, the best performing model based on test accuracy of 90.0% was a single layer LSTM with 128 hidden nodes. It leveraged a vocabulary of 4,000 words, the first 60 words in each article, and GloVe word embedding with 200 dimensions. The confusion matrix shows a clear diagonal, indicating accuracy across the classes (see Figure 4). The most misclassification occurred with Sports and World articles, which had F1-scores of 86% and 87%, respectively. Science/Tech were easiest to identify with an F-score of 96%. The main issue of this model is clear overfitting, as training and validation accuracy diverge already by the 2nd epoch. This brings into question how generalizable the network is despite achieving strong test accuracy on this data set.

Company X should move forward with this model thus far. However, further exploration would be beneficial. This includes more EDA and text feature engineering, trying different embeddings whether pre-trained or modeled from the data itself, more regularization, and

transformer models. Real articles from Company X's collection should also be utilized with labels based on the topic categories of interest.

When experimenting further, some findings from this research should be kept in mind. Vocabulary size has more of an impact than words used per document. Generally, the larger the vocabulary, the better the accuracy since more meaning/patterns can be extracted. However, the improvement diminishes as the size increases, indicating an optimal point. Less than 5% of the unique words in the corpus could be sufficient to classify the articles. In terms of word embedding, the number of dimensions chosen in a pretrained embedding like GloVe minimally affects performance. Perhaps there is more variance between different types of embedding. Having more neurons is more impactful than more layers. In fact, additional layers can significantly increase training time while having worse accuracy. Lastly, a 1D CNN can be useful in cases that require fast processing, while simple RNNs should not be used given their poor classification accuracy and extremely long run times.

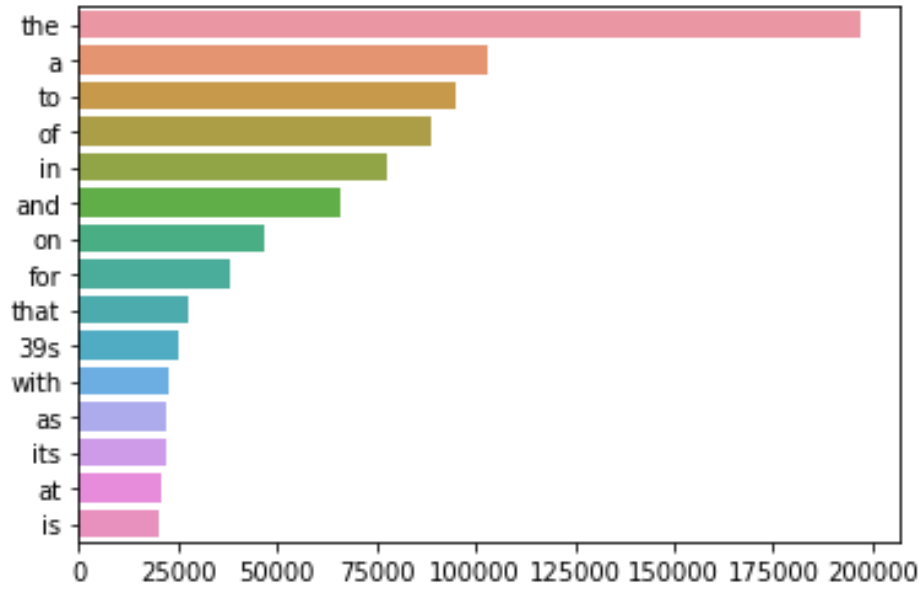
References

Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow* (2nd ed.). O'Reilly.

Appendix

Figure 1. Most Common Words.

Including Stop Words



Excluding Stop Words

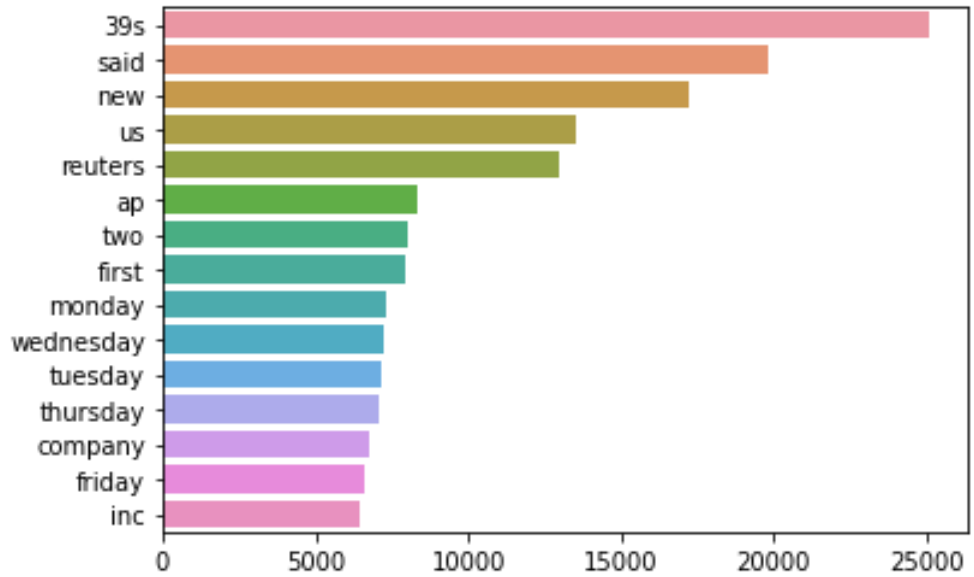


Figure 2. Summary Table of Model Results.

Evaluation	Words Per Doc			Vocabulary Size			Embedding Dims	Number of Neurons		Multiple Layers			Other Neural Networks	
Experiment	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
Words Per Doc	30	40	60	60	60	60	60	60	60	60	60	60	60	60
Vocab Size	1,000	1,000	1,000	2,000	3,000	4,000	4,000	4,000	4,000	4,000	4,000	4,000	4,000	4,000
Embedding	GloVe 6B 100d	GloVe 6B 100d	GloVe 6B 100d	GloVe 6B 100d	GloVe 6B 100d	GloVe 6B 100d	GloVe 6B 200d	GloVe 6B 200d	GloVe 6B 200d	GloVe 6B 200d	GloVe 6B 200d	GloVe 6B 200d	GloVe 6B 200d	GloVe 6B 200d
Model Type	LSTM	LSTM	LSTM	LSTM	LSTM	LSTM	LSTM	LSTM	LSTM	LSTM	LSTM	LSTM	Simple	1D CNN
# of Layers	1	1	1	1	1	1	1	1	1	2	2	2	2	2
# of Nodes	32	32	32	32	32	32	32	64	128	128, 64	64, 32	128, 64 & 128, 32 dense	128, 64	32, 64 & 32 dense
Optimizer	RMSprop	RMSprop	RMSprop	RMSprop	RMSprop	RMSprop	RMSprop	RMSprop	RMSprop	RMSprop	RMSprop	RMSprop	RMSprop	RMSprop
Regularization	50% dropout	50% dropout	50% dropout	50% dropout	50% dropout	50% dropout	50% dropout	50% dropout	50% dropout	50% dropout	50% dropout	50% dropout	50% dropout	50% dropout
Training Loss	0.3782	0.3655	0.3631	0.2882	0.2659	0.2629	0.2385	0.2080	0.1490	0.1728	0.1750	0.1940	0.4640	0.4512
Training Acc	0.8632	0.8685	0.8699	0.8995	0.9070	0.9104	0.9181	0.9276	0.9478	0.9402	0.9403	0.9409	0.8493	0.8615
Validation Loss	0.3917	0.3728	0.3728	0.3316	0.3062	0.3026	0.3095	0.3179	0.3900	0.3596	0.3667	0.4223	0.4318	0.3604
Validation Acc	0.8598	0.8638	0.8693	0.8857	0.8977	0.8942	0.9013	0.9005	0.9050	0.9003	0.8988	0.8985	0.8587	0.8855
Testing Loss	0.4023	0.3879	0.3900	0.3388	0.3244	0.3201	0.3274	0.3318	0.3977	0.3959	0.3853	0.4184	0.4554	0.3843
Testing Acc	0.8562	0.8599	0.8620	0.8841	0.8901	0.8934	0.8964	0.8987	0.9003	0.8975	0.8934	0.8996	0.8522	0.8775
Training Time	6:02	5:42	6:17	8:52	8:14	6:58	6:57	8:41	8:31	12:15	15:24	12:34	54:21	4:29

Figure 3. Distribution of Words Per Article.

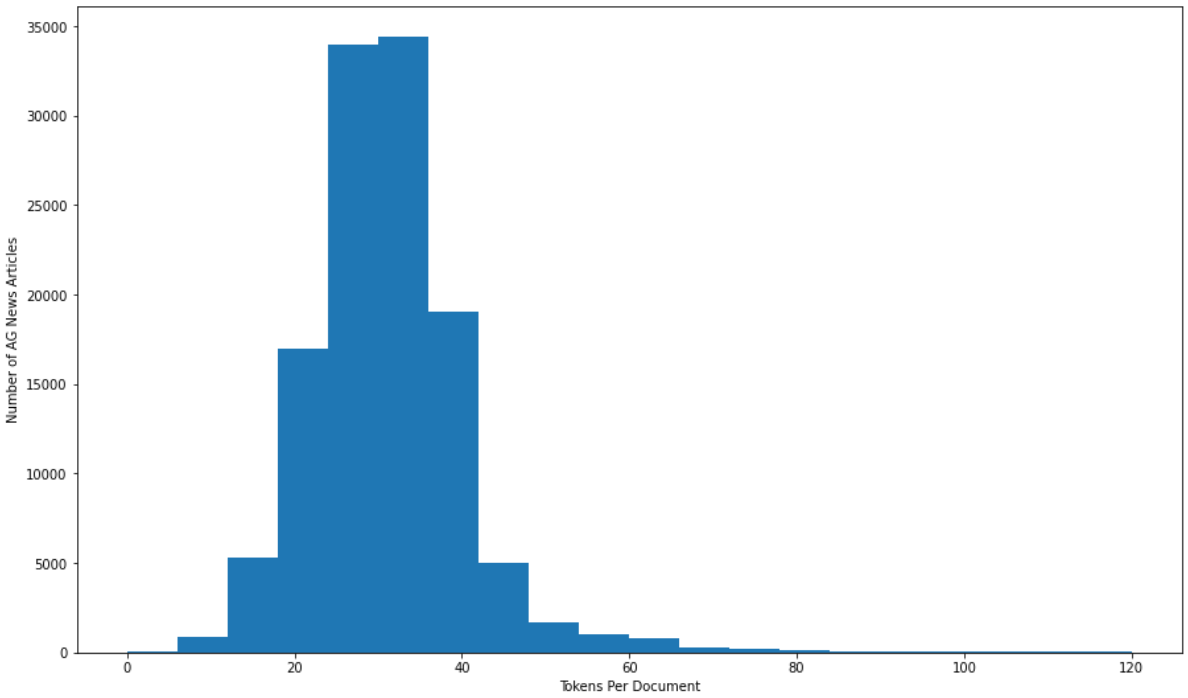
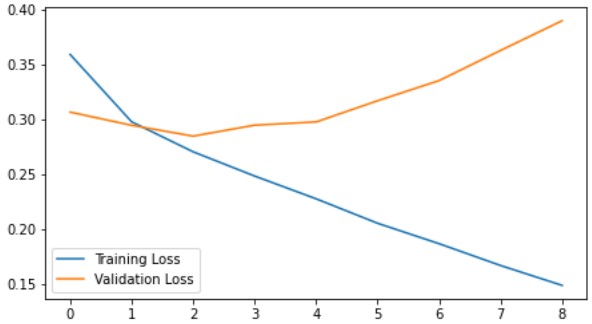
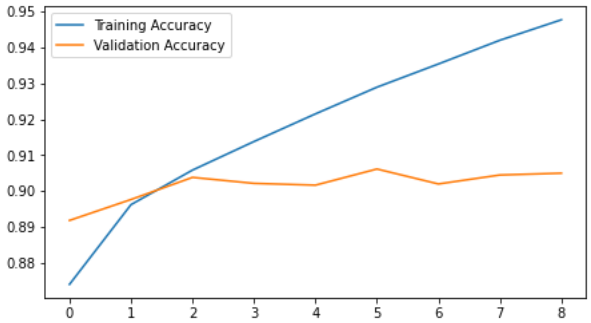


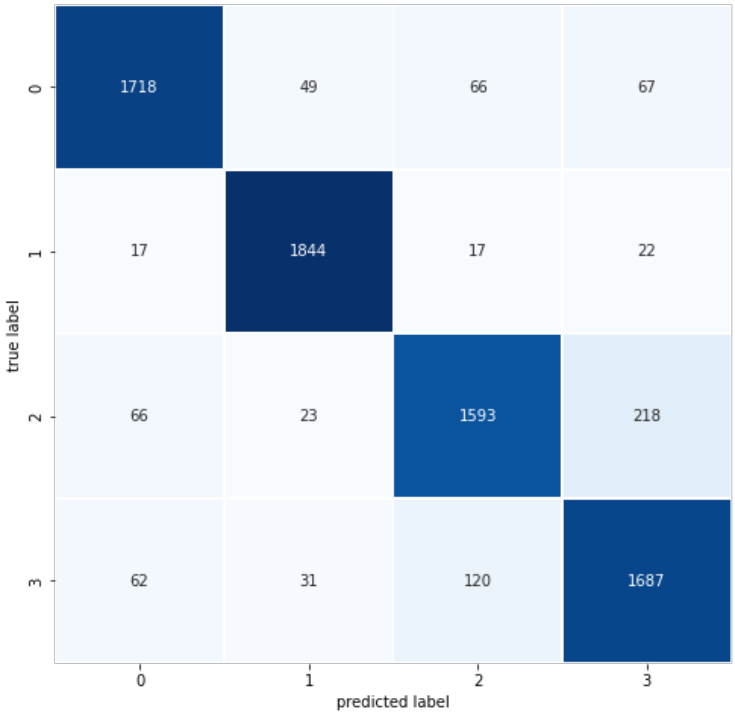
Figure 4. Performance Metrics of Best Model (Experiment 9).

Performance Curves





Confusion Matrix



Classification Report

```
Classification Report
              precision    recall  f1-score   support

           0       0.92      0.90      0.91      1900
           1       0.95      0.97      0.96      1900
           2       0.89      0.84      0.86      1900
           3       0.85      0.89      0.87      1900

    accuracy                           0.90      7600
   macro avg       0.90      0.90      0.90      7600
weighted avg       0.90      0.90      0.90      7600

Accuracy Score: 0.9002631578947369
Root Mean Square Error: 0.5554751403788898
```