Performance Evaluation of Deep Neural Networks Applied to News Classification: LSTM and GRU

BY L. QIU

School of Economics, Xiamen University, Xiamen, Fujian, China

jeffery.l.qiu@gmail.com

SUMMARY

Text mining has gained quite a significant importance during the past few years. Nowadays, data is available from a myriad of sources like electronic and digital media, often in an unstructured form. It is highly desirable to structure this data, classify it into categories. News contents are one of the most important factors that have influence on various sections. In this paper we have considered the problem of classification of news articles. This paper performs Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) for category identification of English news and compared the accuracy and loss of the two models. The experiments indicate GRU has better performance than LSTM in news classification task.

Some key words: LSTM; GRU; News Classification.

1. INTRODUCTION

There exists a vast quantity of information stored in electronic format. With such data, means for interpreting and analysing it to extract facts aiding decision-making have become necessary. Data mining, used to extract hidden information from large databases, is a powerful tool for this purpose. News information was not easily and rapidly available until the beginning of the last decade. However, news is now readily accessible via content providers such as online news services. A sizable amount of unstructured text exists across diverse domains whose analysis could benefit multiple fields. Text classification poses considerable challenges as it requires preprocessing steps to transform unstructured data into structured information. With increasing news volume, it has become difficult for users to access articles of interest, necessitating news categorization for easier access. Categorization refers to grouping facilitating navigability among articles. Internet news must be divided into categories to help users promptly access news matching their interests without wasted time. Classifying news is particularly challenging as continuously emerging, potentially novel articles require processing, which may fall into new categories. Deep learning provides a viable solution for this problem. The present study implements Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), evaluating their performance for news classification.

2. DATA SET

2.1. *Data source*

The news category dataset used in this paper was obtained from Kaggle (Misra & Rishabh, 2022). This dataset contains approximately two hundred and ten thousand news headlines pub-

lished by HuffPost between 2012 and 2022. This is one of the biggest news datasets and can serve as a benchmark for a variety of computational linguistic tasks.

The raw dataset contains 209,527 examples described by 6 features. Table 1 details the feature explanations.

Table 1. *Feature explanation*

Feature	Description
Category	Category in which the article was published.
Headline	The headline of the news article.
Authors	List of authors who contributed to the article.
Link	Link to the original news article.
Short Description	Abstract of the news article.
Date	Publication date of the article.

The 'Category' feature has 42 distinct values. The number of examples in each category ranges from 1,014 to 35,602. Figure 1 depicts the top 10 most prevalent news categories; the remaining categories are relatively evenly distributed between 1,014 and 5,400 examples. This distribution aptly facilitates model training and evaluation across diverse categories.

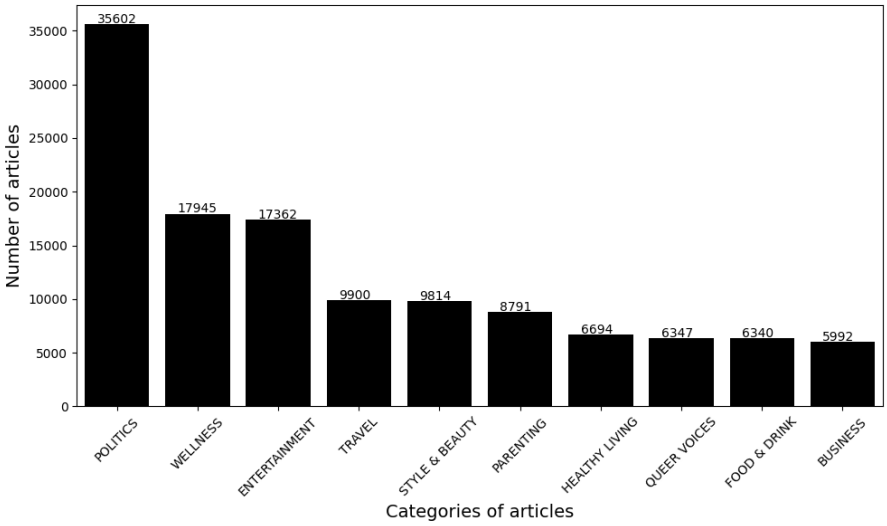

Fig. 1. Top 10 categories of news articles.

2.2. *Data processing*

Firstly, irrelevant features such as author, links, and date were dropped as they bear no relation to the classification task. The headline and short description were then concatenated to form a single raw text input feature. And statistical analysis revealed a minimum length of 3 words, first quartile of 122 words, median of 170 words, third quartile of 207 words, and maximum of 1,486 words for the concatenated text.

Considering the excessive example size of the 'politics' class, the training set was balanced by keeping only the first 16,000 examples. Several text preprocessing techniques were applied to reduce noise, including removal of usernames, lowercasing, and lemmatization of verbs, etc.

For model evaluation, the dataset was randomly partitioned into 80% for training and 20% for validation.

To transform the unstructured text into numerical data, the training set was tokenized into words to build a vocabulary of maximum 20,000 items, establishing a word-to-index mapping applied to both training and validation sets. Truncation and padding were performed based on batched training requirements, with an empirical maximum length of 50 words. This preprocessing facilitated training LSTM and GRU on the structured sequential input data.

3. Research Methodology

3.1. *Model introduction*

For sequential data learning problem, Recurrent Neural Networks (RNN) is often the fundamental model. But RNNs struggle to learn long-distance dependencies due to the vanishing gradient problem.

As a solution to the shortcoming of normal RNNs, Hochreiter and Schmidhuber (1997) came up with LSTM networks. Special memory cell architecture in LSTM make it easier to store information for long period. The cell structure has been modified by many people since then, however, the standard formulation of single LSTM cell can be given by following equations

$$\boldsymbol{i}_t = \sigma(\boldsymbol{W}_{ii}\boldsymbol{x}_t + \boldsymbol{b}_{ii} + \boldsymbol{W}_{hi}\boldsymbol{h}_{t-1} + \boldsymbol{b}_{hi}),$$

$$\boldsymbol{f}_t = \sigma(\boldsymbol{W}_{if}\boldsymbol{x}_t + \boldsymbol{b}_{if} + \boldsymbol{W}_{hf}\boldsymbol{h}_{t-1} + \boldsymbol{b}_{hf}),$$

$$\boldsymbol{g}_t = tanh(\boldsymbol{W}_{ig}\boldsymbol{x}_t + \boldsymbol{b}_{i}g + \boldsymbol{W}_{hg}\boldsymbol{h}_{t-1} + \boldsymbol{b}_{hg}),$$

$$\boldsymbol{o}_t = \sigma(\boldsymbol{W}_{io}\boldsymbol{x}_t + \boldsymbol{b}_{io} + \boldsymbol{W}_{ho}\boldsymbol{h}_{t-1} + \boldsymbol{b}_{ho}),$$

$$\boldsymbol{c}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \boldsymbol{g}_t,$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot tanh(\boldsymbol{c}_t),$$

where $\boldsymbol{h}_t$ is the hidden state at time t, $\boldsymbol{c}_t$ is the cell state at time t, $\boldsymbol{x}_t$ is the input at time t, and $\boldsymbol{i}_t$, $\boldsymbol{f}_t$, $\boldsymbol{g}_t$, $\boldsymbol{o}_t$ are the input, forget, cell, and output gates, respectively. σ is the sigmoid function, $tanh$ is the hyperbolic tangent function, and $\odot$ is the Hadamard product. The sigmoid function is used to form three gates in the memory cell, whereas the tanh function is used to scale up the output of a particular memory cell.

Introduced by Chung et al. (2014), GRUs are similar to LSTMs but they have fewer parameters. They also have gated units like LSTMs which controls the flow of information inside the unit but without having separate memory cells. Unlike LSTM, GRU does not have output gate, thus exposing its full content. GRU formulation can be given by following equations

$$\boldsymbol{r}_t = \sigma(\boldsymbol{W}_{ir}\boldsymbol{x}_t + \boldsymbol{b}_{ir} + \boldsymbol{W}_{hr}\boldsymbol{h}_{t-1} + \boldsymbol{b}_{hr}),$$

$$\boldsymbol{z}_t = \sigma(\boldsymbol{W}_{iz}\boldsymbol{x}_t + \boldsymbol{b}_{iz} + \boldsymbol{W}_{hz}\boldsymbol{h}_{t-1} + \boldsymbol{b}_{hz}),$$

$$\boldsymbol{n}_t = tanh\{\boldsymbol{W}_{in}\boldsymbol{x}_t + \boldsymbol{b}_{in} + \boldsymbol{r}_t \odot (\boldsymbol{W}_{hn}\boldsymbol{h}_{t-1} + \boldsymbol{b}_{hn})\},$$

$$\boldsymbol{h}_t = (1 - \boldsymbol{z}_t) \odot \boldsymbol{n}_t + \boldsymbol{z}_t \odot \boldsymbol{h}_{t-1},$$

where $\boldsymbol{r}_t$, $\boldsymbol{z}_t$, $\boldsymbol{n}_t$ are the reset, update, and new gates, respectively.

Both LSTM and GRU are equally capabel to handle long term dependencies, and have been experimented and compared with machine translation tasks and proved to be comparably efficient (Bahdanau et al., 2014).

3.2. *Architecture for experiments*

The model architecture takes text records as input and predicts categorical output. Consider a single input $\boldsymbol{x}$ and label y as being sampled from the training set $\mathbb{X} = \{(\boldsymbol{x}^{(1)}, y^{(1)}), \ldots, (\boldsymbol{x}^{(n)}, y^{(n)})\}$ where every $\boldsymbol{x}^{(i)}$ represents a sequence of words or tokens of length T; T is the fixed length of each input vector, and was set to be 50, as required for batch training. Batch size was set to 128 for computational efficiency. Eventually, the sequence of inputs $\boldsymbol{x}$ are converted into the sequence of numerical indices for classifying $y \in \{0, \ldots, 41\}$.

Our experiments are performed with two models, one where we use single layer LSTM, the second with one layer GRU. Both models is structured with a fully connected layer of 14 neurons as embedding layer and a fully connected layer of 42 neurons as output layer.

The output layer employs a standard softmax function to calculate predicted word probabilities for the final timestep $T = 50$ and category j. This output is given by

$$h_{j,T}^{[2]} = \hat{p}_j \equiv pr(c = j \mid \boldsymbol{x}) = \frac{exp\left(\boldsymbol{W}_j^{[2]\mathsf{T}} \boldsymbol{h}_T^{[1]} + b_j^{[2]}\right)}{\sum_{k=1}^{K} exp\left(\boldsymbol{W}_k^{[2]\mathsf{T}} \boldsymbol{h}_T^{[1]} + b_k^{[2]}\right)}, \tag{1}$$

where c represent categories, $j \in \{0, \ldots, 41\}$, $\boldsymbol{W}_k^{[2]}$ and $\boldsymbol{b}_k^{[2]}$ denote the kth column of the weight matrix and the kth bias, respectively.

After the category probabilities are calculated, the cross-entropy loss is calculated next. The cross-entropy loss is the most common loss used in classification problems. Given the output of the network, the cross-entropy loss calculates the negative log likelihood of the probability of the target. For this, the network output from (1), the probabilities of the categories, is the input of the cross-entropy loss. This loss can be derived and then be 'backpropagated' to the weights of the network. The Adam optimization algorithm (Kingma & Ba, 2014) has been used for the backpropagation training since this training algorithm is tolerant to learning rate as well as to other training parameters, and thus, requires less fine-tuning.

4. Experiments and Results

4.1. *Parameter setup*

Table 2 lists parameters were used for the runs in both models.

Table 2. *Main parameter setup*

Parameter	Value	Parameter	Value
Number of epochs	20	Neuron count in embedding layers	14
Training batch size	128	Neuron count in hidden layers	128
Validation batch size	128	Neuron count in output layers	42
Activation function	ReLu	Learning rate for Adam optimizer	0.001

4.2. *Evaluation measures*

For multi-class classification problems, common evaluation metrics are accuracy and cross-entropy loss. Accuracy is defined as

$$AC = n^{-1} \sum_{i=1}^{n} I\left(\hat{y}^{(i)} \neq y^{(i)}\right),$$

where $\hat{y}^{(i)} = \arg\max_j \hat{p}_j^{(i)}$, and n is the total number of records in the dataset. It reflects the accuracy of predictions from the trained model.

The cross-entropy loss for one prediction is given by

$$\mathcal{L}(y_j, \hat{p}_j) = \sum_{j=0}^{41} y_j \log \hat{p}_j.$$

The interpretation of loss is that lower validation loss indicates better news classification, since the model more closely approximates the true class probability distribution.

4.3. *Results*

Figure 2 plots the accuracy values graphically. GRU achieves a higher accuracy of 53.87% followed closely by LSTM at 52.74%. And GRU reaches its peak performance at the 14th epoch, whereas LSTM obtains its best accuracy at the 18th epoch.

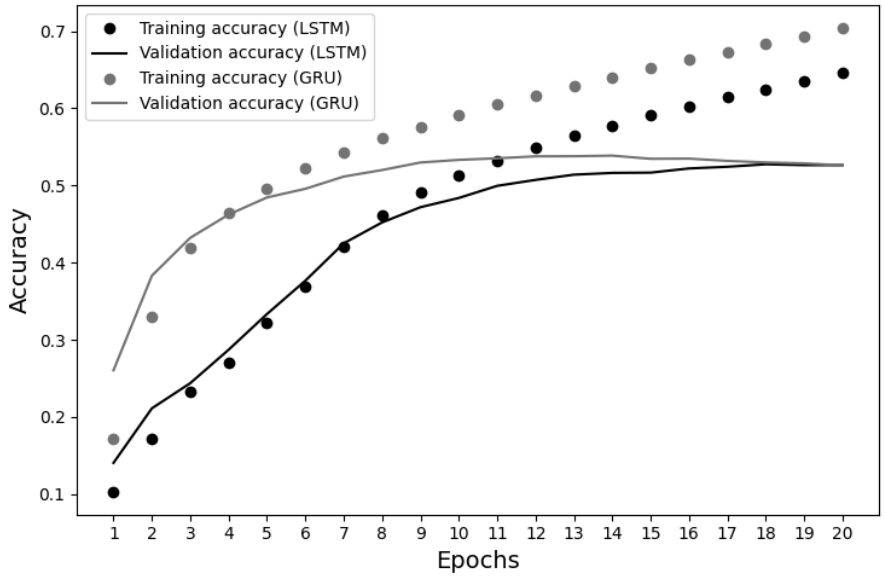

Fig. 2. Accuracy values per epoch for all models.

As shown in the loss curve in Fig. 3 for 20 training epochs, GRU demonstrates superior performance to LSTM on the validation set prior to the 16th epoch.

However, in terms of computational efficiency, LSTM outperforms with an execution time of approximately twelve minutes, while GRU requires around four times as long. Moreover, given the narrow differences observed in both accuracy and loss between LSTM and GRU, LSTM emerges as a strong contender for news classification due to its computational advantages.

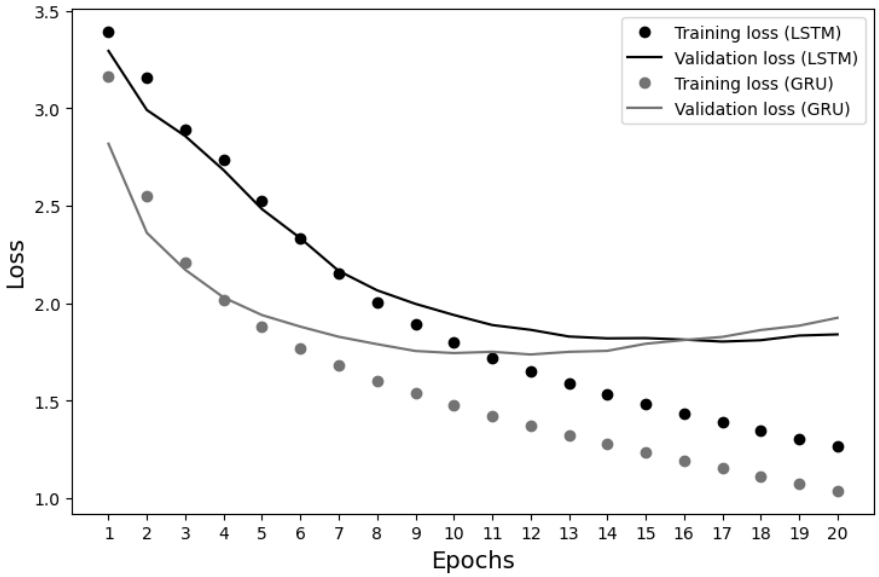

Fig. 3. Cross-entropy loss per epoch for all models.

5. Conclusion

Since standard feedforward neural networks are unable to adequately model sequential text data (due to lacking a way to feed information from a later layer back to an earlier layer), thus, RNNs have been introduced to take the temporal dependencies of text data into account. However, RNNs struggles to capture long-term dependencies due to the vanishing/exploding gradient problem. Therefore, LSTMs and a few years later GRUs were introduced to overcome the shortcomings of RNNs.

The present study evaluated LSTM and GRU and compared their performances on a news category data set. The evaluation measures used were accuracy, cross-entropy loss, and the execution time. Results show that while LSTM and GRU achieved similar accuracy (with GRU scoring marginally higher), LSTM exhibited notably faster execution times. Therefore, LSTM is recommended for this news categorization task due to its ability to achieve good predictive performance within an acceptable computational budget.

Future work will include parameter optimization in order to investigate the influence on different parameter settings. Parameters such as hidden layer count and size, learning rate, and dropout rate will be explored. This will provide further insight into improving predictive performance for news categorization.

References

Misra & Rishabh (2022). News category dataset. *arXiv*: 2209.11429.

Hochreiter, S. & Schmidhuber, J. (1997). Long short-term memory. *Neural Comput.*, 9, 8, 1735-1780.

Chung, J., Gulcehre, C., Cho, K. & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv*: 1412.3555.

Bahdanau, K., Cho, K. & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv*: 1409.0473.

Kingma, D. & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv*: 1412.6980.