Text classification using Recurrent Neural Networks

Cristiano Pacini, Yuri Tirendi

February 25, 2023

Abstract

This report goes through our work on text classification by different approaches. First, we describe the datasets used and the feature exctraction process on them. We then analyze the results with simple machine learning algorithms, i.e. Logistic Regression, Decision Tree and k-NN. Finally, we compare them with a basic RNN, a deeper one and a Bi-directional LSTM.

1 Data sets

1.1 News dataset

The "News dataset" is composed by 120'000 instances, which are descriptions of newspaper articles. The classification task consists in building a model that can distinguish articles by topics. The four categories are:

- World
- Sports
- Business
- Sci/Tech

1.2 Clickbait dataset

This dataset contains headlines from various news sites such as 'WikiNews', 'New York Times', 'The Guardian', 'The Hindu', 'BuzzFeed', 'Upworthy', 'ViralNova', 'Thatscoop', 'Scoopwhoop' and 'ViralStories'. Each headline is labeled as clickbait or non-clickbait. The dataset contains a total of 32000 instances.

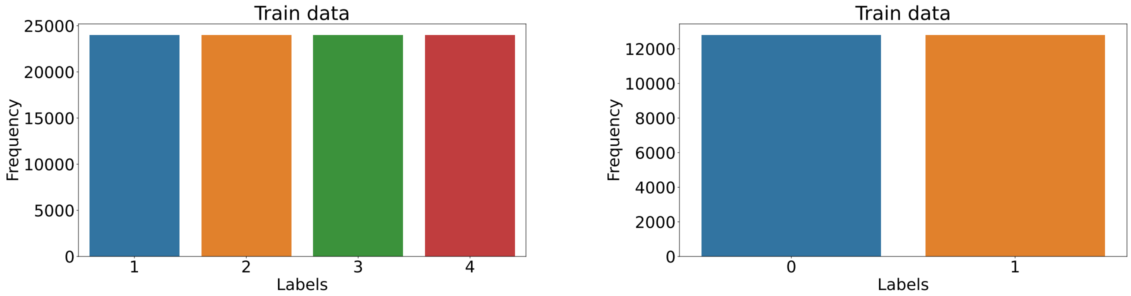

Figure 1: The two images represent the distribution of the target classes in News dataset (on the left) and Clickbait dataset (on the right) in the training data. We can see that the target classes are equally distributed in both datasets.

2 Data preprocessing and feature exctraction

As always to train and evaluate a model it is important to split the datasets. We divided both dataset keeping the 80% for the train set, 10% for the validation set and the 10% for the test set. First of all from Figure 1 we can deduce that in both datasets the target classes are perfectly balanced. Thus no undersampling or oversampling strategy is needed. Secondly we took care of counting how the number of words per sentence is distributed in the datasets. Looking at Figure 2 we noticed that a good cut is given by the first 40 words for each sample of News dataset (this will be needed for padding). Similarly in Figure 13 we decided to consider the first 20 words in Clickbait dataset's samples. After that we applied a tokenization for both data set. We map each word into a number. After that, for each instance which is shorter than 40 words we add zeros. In the end we use a BoW (Bag of Words) approach.

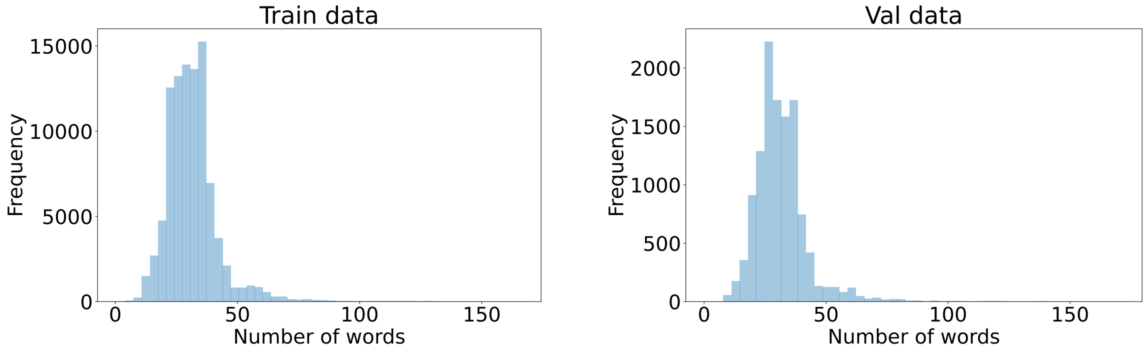

Figure 2: Distributions of number of words for each instance in News dataset for train and test data.

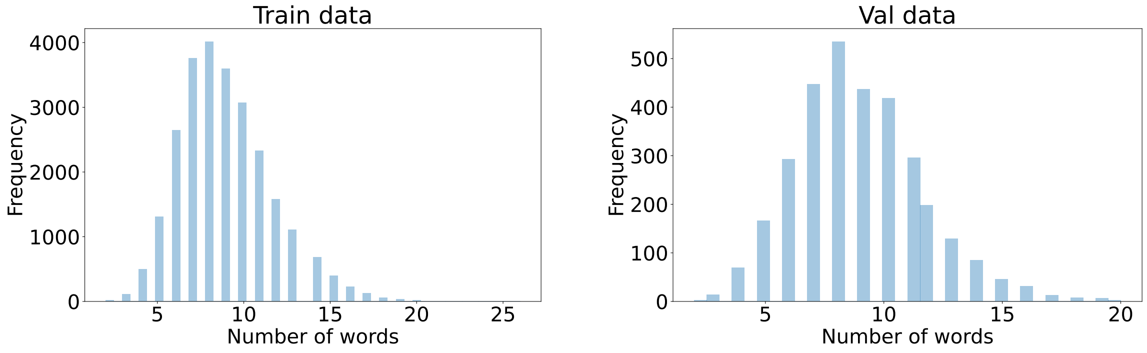

Figure 3: Distributions of number of words for each instance in Clickbait dataset for train and test data.

3 Classic Machine Learning algorithms and optimization

We apply classic machine learning techniques to the two data sets: Logistic Regression, Decision Tree Classifier and k-Nearest Neighbors. We discovered that for both data sets the Logistic Regression algorithm gives the best predictions, however we decided to apply a Grid Search also for the Decision Tree model to be sure about the result. Regarding the k-NN, it was already clear with standard hyperparameters that the performances were way lower than the other two models. Our best results are reported in Table 1 and Table 2.

3.1 Logistic Regression

The Logistic Regression algorithm has the best performance for both datasets (see Table 1 and Table 2). We applied a Grid Search to tune the regularization hyperparameter. As a consequence we chose a regularization coefficient C=0.1 for the News dataset and C=0.9 for the Clickbait Dataset.

3.2 Decision Tree

Applying a Grid Search we tuned the depth of the tree and the minimum number of samples to split a leaf. For the News and the Clickbait data sets we observed that it is better not to set a maximum lenght of the tree and that 5 is the best minimum number of sample to split a leaf.

3.3 k-Nearest Neighbors

The k-NN algorithm obtains clearly the worst results with both the datasets, which is due to the types of data considered. Therefore, we did not proceed to tune any hyperparameter.

3.4 Best results and optimization

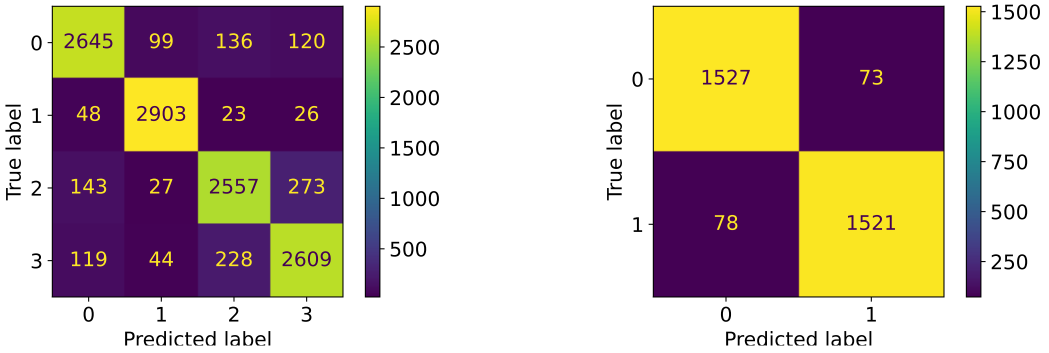

Figure 4: Confusion matrix for the News (left) and Clickbait (right) dataset with Logistic Regression.

News dataset	**Logistic Regression**	**Decision Tree**	**k-Nearest Neighbors**
Accuracy	0.893	0.814	0.488
Precision	0.892	0.814	0.682
Recall	0.893	0.815	0.488
F1 score	0.892	0.815	0.473

Table 1: Results on News dataset with Classical Machine Learning techniques

4 Recurrent Neural Networks

We used the python `keras` libray to build Recurrent Neural Networks (RNNs), which are optimal for processing sequential data such as text by preserving an internal memory state, thanks to feedback

Clickbait dataset	**Logistic Regression**	**Decision Tree**	**k-Nearest Neighbors**
Accuracy	0.953	0.862	0.605
Precision	0.953	0.873	0.727
Recall	0.953	0.857	0.605
F1 score	0.953	0.861	0.544

Table 2: Results on Clickbait dataset with Classical Machine Learning techniques

loops that allow to deal with the current input with information from the previous. We considered three different models: a simple recurrent neural network, a more complex neural network, which is composed by more layers than the previous one, and in the end a bi-directional LSTM neural network. The News data set is meant to work on a multi-calssification task. Therefore we used a Softmax function as activation function and a Sparse Categorical Crossentropy function as lossfunction. On the other hand we worked on a binary classification task for the Clickbait data set, for this reason we chose a Sigmoid and a Binary Crossentropy as activation function and loss function respectively.

Moreover, we decided to have an early stopping approach using 2 as number of epochs with no improvement after which training will be stopped and to restore model weights from the epoch with the best value of the loss function. In the next paragraphs we will see the results obtained tuning the hyperparameters using the validation set. In the Tables 3-8 we can see how the accuracy change varying the Learning Rate and the Dropouts. For the best results we reported a confusion matrix for each RNN and for both data sets.

Other important specifics of the Neural Networks, such as the number of units per layer and the order of layers, are contained in the notebooks attached.

4.1 Simple Recurrent Neural Network

Our first attempt consists of a relatively shallow RNN, with 3 layers. In Table 3-4 we can see that for LR=0.001 and Dropout=0.2 we obtain the best accuracy for both dataset. In Figure 5 we can observe the confusion matrices.

Simple RNN	Dropout = 0.1	Dropout = 0.2
LR = 0.001	0.878	0.880
LR = 0.005	0.871	0.863
LR = 0.01	0.888	0.859

Table 3: Accuracy score for the Simple RNN on the News dataset for different choices of hyperparameters.

Simple RNN	Dropout = 0.1	Dropout = 0.2
LR = 0.001	0.967	0.972
LR = 0.005	0.966	0.963
LR = 0.01	0.964	0.959

Table 4: Accuracy score for the Simple RNN on the Clickbait dataset for different choices of hyperparameters.

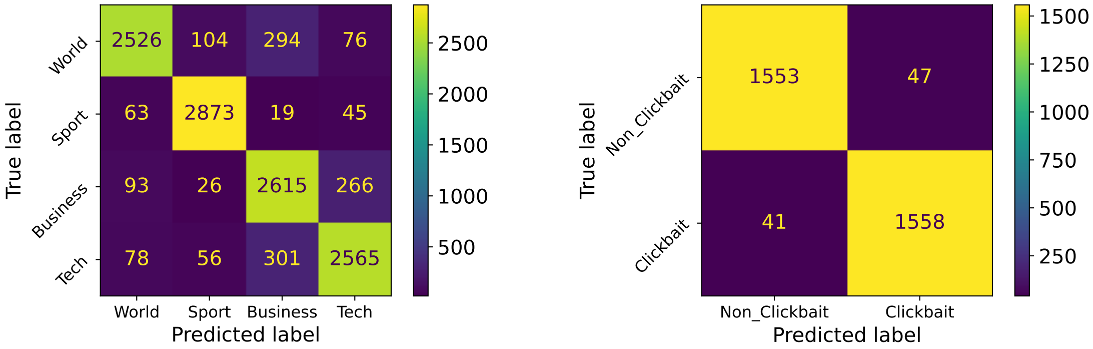

Figure 5: Confusion matrix for the News (left) and Clickbait (right) dataset with simple neural network.

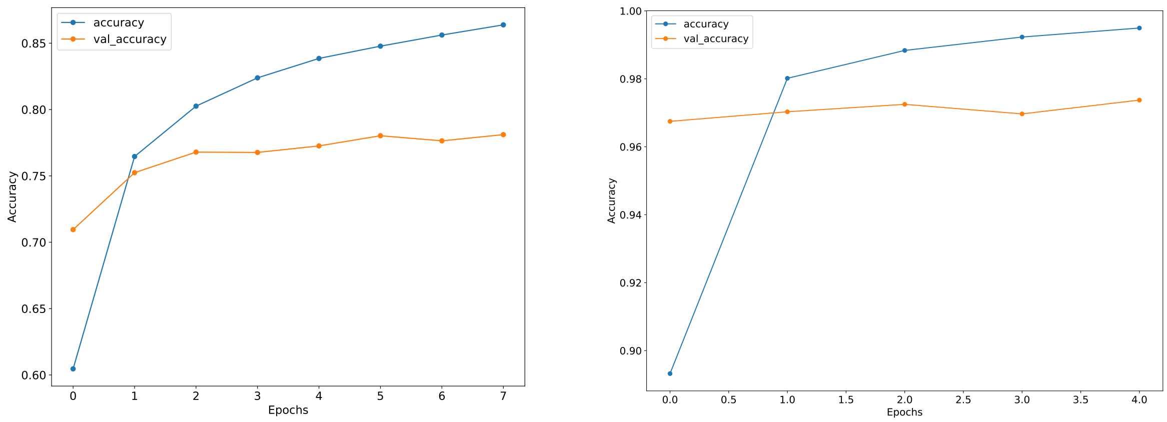

Figure 6: Accuracy and validation accuracy for the News (left) and Clickbait (right) dataset with simple neural network.

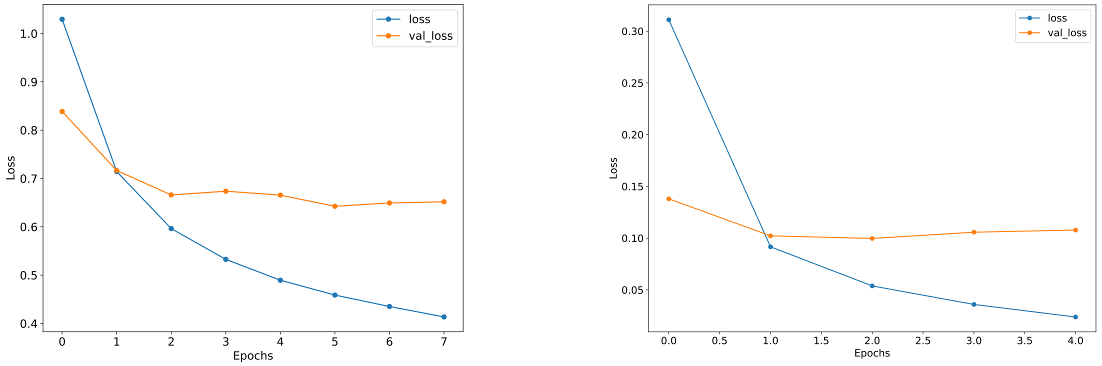

Figure 7: Loss and validation loss for the News (left) and Clickbait (right) dataset with simple neural network.

4.2 Deeper Recurrent Neural Network

We then built a deeper RNN, consisting of 5 layers. In Table 5-6 we can see that again, as in the previous case, for LR=0.001 and Dropout=0.2 we obtain the best accuracy for both dataset. In Figure 8 we can observe the confusion matrices.

Deeper RNN	Dropout = 0.1	Dropout = 0.2
LR = 0.001	0.868	0.877
LR = 0.005	0.859	0.852
LR = 0.01	0.855	0.855

Table 5: Accuracy score for the Deeper RNN on the News dataset for different choices of hyperparameters.

Deeper RNN	Dropout = 0.1	Dropout = 0.2
LR = 0.001	0.965	0.972
LR = 0.005	0.963	0.966
LR = 0.01	0.955	0.957

Table 6: Accuracy score for the Deeper RNN on the Clickbait dataset for different choices of hyperparameters.

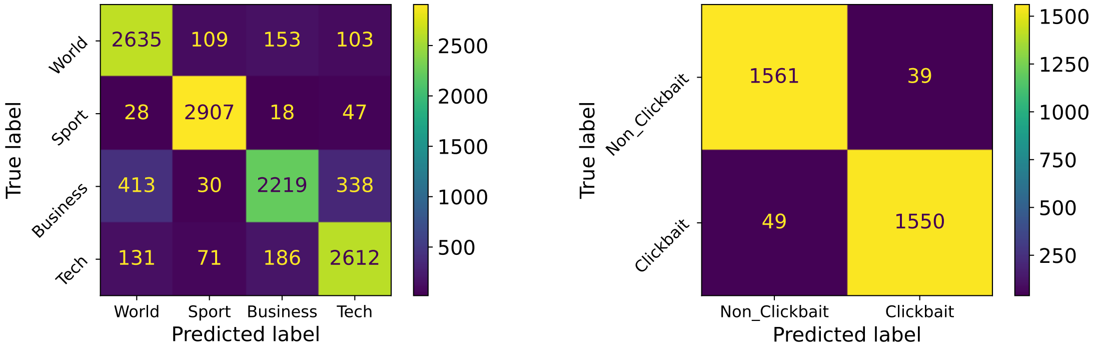

Figure 8: Confusion matrix for the News (left) and Clickbait (right) dataset with deeper neural network.

4.3 Bi-directional LSTM Neural Network

As a variant of RNN, we exploited a Long Short-Term Memory (LSTM) network, which can help to avoid the major RNN problem, i.e. the vanishing gradient. We built a neural network composed by 11 layers, including Bi-directional LSTM layers. Our results reported in Table 6 and Table 6 highlight that LR=0.001 and Dropout=0.2 are still the best options for the hyperparameters.

5 Comparison between classical Machine Learning and Deep Learning approaches

5.1 Overfitting

In the Figures 6-10 we analyze and compare for each RNNs the accuracy with respect to the validation accuracy and the loss with respect to the validation loss. In general we observe that when the number of epochs increases the loss function tends to get smaller, but this is not true for the validation loss function, which can start to increase again. This is due to the overfitting. Also it is important to notice that after a certain number of epochs the accuracy stop increasing, which is another good reason to apply an early stop to preserve computational time.

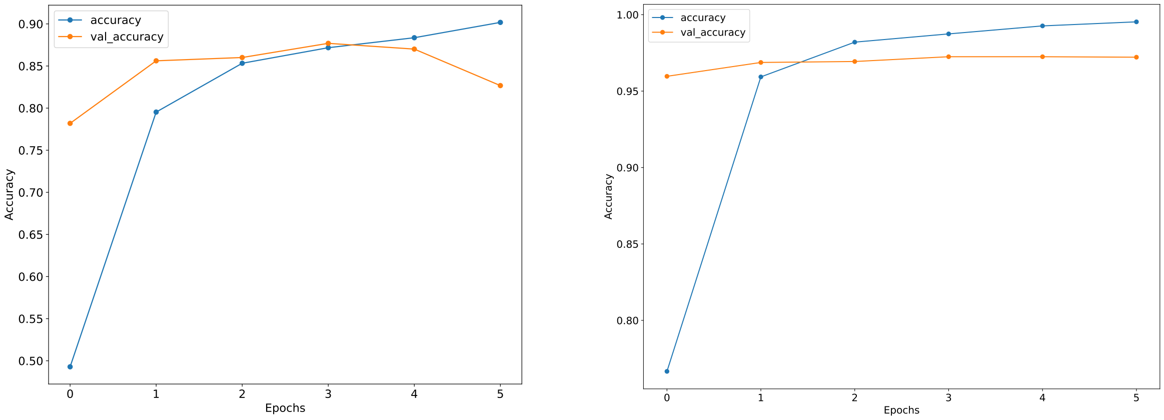

Figure 9: Accuracy and validation accuracy for the News (left) and Clickbait (right) dataset with deeper neural network.

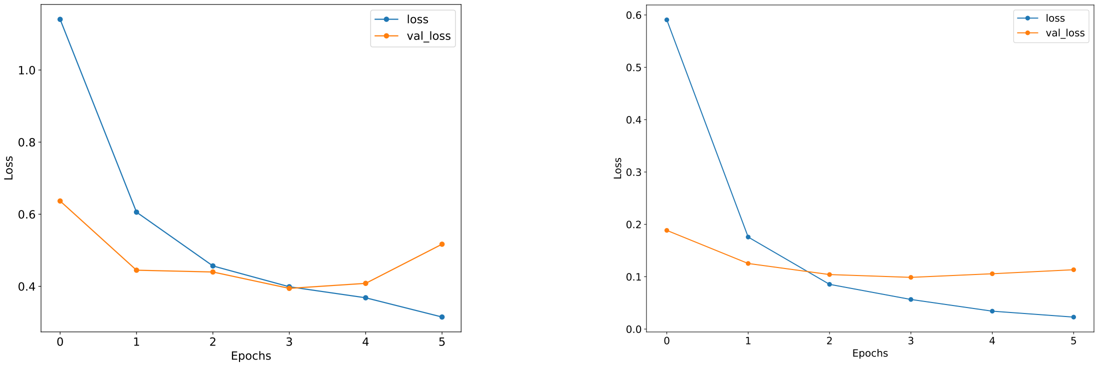

Figure 10: Loss and validation loss for the News (left) and Clickbait (right) dataset with deeper neural network.

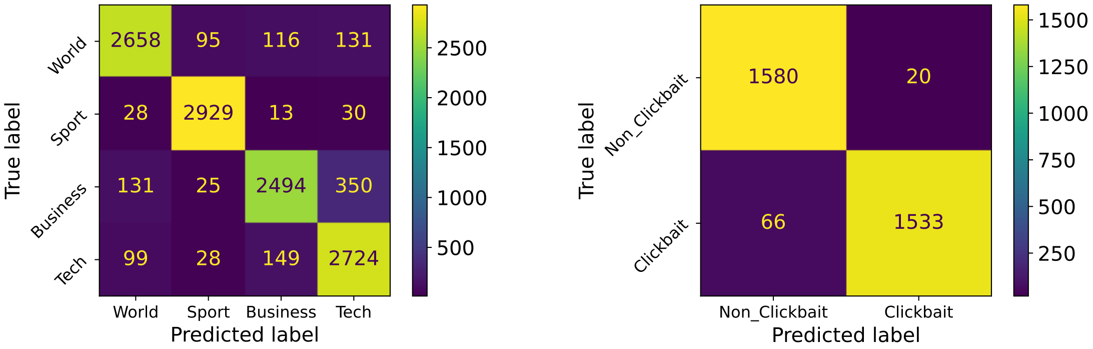

Figure 11: Confusion matrix for the News (left) and Clickbait (right) dataset with LSTM neural network.

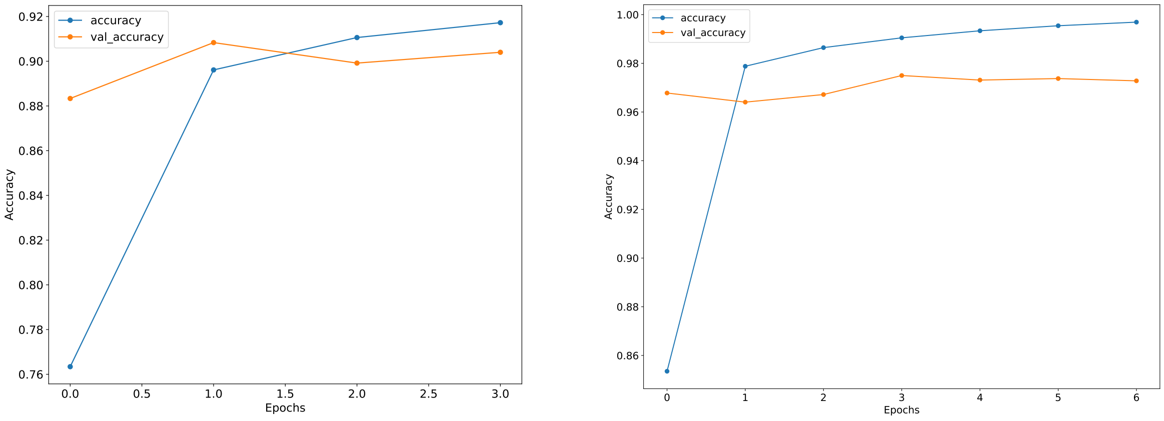

Figure 12: Accuracy and validation accuracy for the News (left) and Clickbait (right) dataset with LSTM neural network.

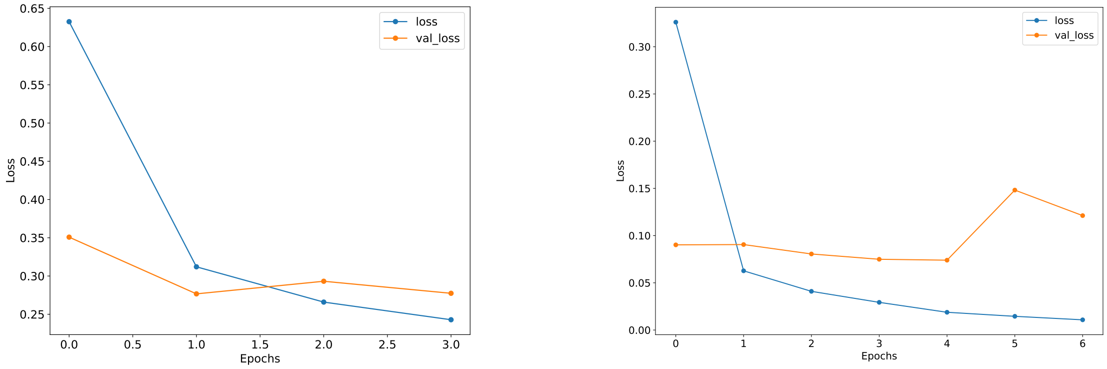

Figure 13: Loss and validation loss for the News (left) and Clickbait (right) dataset with LSTM neural network.

LSTM	Dropout = 0.1	Dropout = 0.2
LR = 0.001	0.901	0.908
LR = 0.005	0.899	0.891
LR = 0.01	0.887	0.867

Table 7: Accuracy score for the LSTM on the News dataset for different choices of hyperparameters.

LSTM	Dropout = 0.1	Dropout = 0.2
LR = 0.001	0.970	0.973
LR = 0.005	0.969	0.966
LR = 0.01	0.963	0.859

Table 8: Accuracy score for the LSTM on the Clickbait dataset for different choices of hyperparameters.

5.2 Run time

In Table 9 we can visualize the computational time required by the different RNNs that we used with the best hyperparameters. Obviously, the running times of the News data set are bigger due to the fact that this data set is larger than the Clickbait data set. Looking at computational times we can also notice, as aspected, the depth of the network has a strong impact on the running time of the fitting.

	Log. Reg.	Dec. Tree	k-NN	Simple RNN	Deeper RNN	LSTM
News dataset	9.8s	40.1s	1min 1.1 s	19.2s	1min 43.2s	11min 51s
Clickbait dataset	0.2s	3.1s	4.1s	5.0s	21.4s	1min 56.2s

Table 9: Results on Clickbait dataset with Classical Machine Learning techniques

5.3 Effectiveness

In general we can say that the RNNs are performing better than the classical machine learning algorithm, and in particular LSTM give the best results by avoiding the gradient-related problem of vanishing and exploding according to the length of sentences. Indeed the best accuracy results obtained with the Logistic Regression are 0.893 (News dataset) and 0.953 (Clickbait dataset) rispectively, instead using a LSTM RNN we are able to achieve 0.908 (News dataset) and 0.973 (Clickbait dataset). In the end we can see that we can reach a 2% of improvment with respect to our accuracy results in the Clickbait dataset. It is however true that a simple algorithm like the Logistic Regression can perform better than a too shallow Neural Network.