A.3 Third Research /Programming Assignment

Language Modeling

MSDS 458 – Artificial Intelligence and Deep Learning

Grayson Matera

November 6, 2022

ABSTRACT

Utilizing the ag_news_subset dataset from Reuters, several Natural Language Processing (NLP) algorithms were explored in a series of 10 experiments. The purpose of this research is to examine the effectiveness of an assortment of neural network structures on NLP tasks. Throughout the experimentation phase, incremental adjustments are made to the networks architecture and hyperparameter sensitivities, as well as, to the ag_news_subset dataset itself. The results of the experiments are then compared to determine the impact of each structural change in the network. The methods of this research are non-comprehensive and open-ended. However, the conclusions drawn from these experiments provide a controlled look at the performance integrity of a variety of neural network designs and architectures on NLP tasks.

The experiments in this study are broken up into four phases. Phase 1 of the experimentation is an open-ended exploratory analysis of the data relating to the impact of different preprocessing techniques. Primarily, Phase 1 experiments exist as a precursor in preparation for the subsequent phases of experimentation. Phase 2 of experimentation, Experiments 1-3, consists of a series of simple recurrent neural network (RNNs) designs with incremental structural changes to each iterative model. Phase 3 of experimentation, Experiments 4-6, build upon the “best model” from Phase 2 through implementation of long short-term memory (LSTM) into the network architecture. Additional structural changes are then incrementally assessed utilizing the LSTM component. Phase 4 of experimentation, Experiments 7-10, introduce components of one-dimensional convolutional neural network (CNN) architecture, as well as transformer encoder and positional embedding components. The Phase 4 is designed to attempt to push the model to peak performance with incremental changes to the model structure and parameters.

INTRODUCTION

In recent years, natural language processing techniques have witnessed an array of major innovations contributing to their overall performance. Mainstream applications of these techniques have become widespread and the complexity of their uses even more profound and impactful. These days, the general population interacts with some form of NLP on a regular basis. Whether it be an online retail chatbot, a search engine, or an interactive voice assistant on their phone, the uses of NLP are increasingly shaping the way individuals interact with digital and real-world spaces. The addition of neural network innovations to NLP have had an astonishingly positive effect on NLP applications. The purpose of this study is to assess the utility and design of various neural network architectures for NLP related tasks. Using the ag_news_subset data set from Reuters, a series of NLP experiments will be performed as a means to better understand the importance of key network design components on their relative performance outcome.

LITERATURE REVIEW

Over the past decade, NLP innovations have pushed it utility to the forefront. This relatively new technology is being implemented in an array of industries from business to healthcare. In the early days of NLP, its utility was much less accessible. With the introduction of deep learning through CNNs and RNNs it is now possible for these tools to easily scale to a wide range of data and use-cases. These innovations have led to major breakthrough technology such as Natural Language Generation and Sentiment Analysis.

METHODS

The following experiments were run using python via the Google CoLab environment with a standard GPU runtime setup. This was strategically chosen to help reduce network runtimes and GPU strain. Additionally, the environment comes preinstalled with all the packages necessary for the experiments including, numpy and pandas for data manipulation, matplotlib and seaborn for visualization, an array of module from sklearn and keras for modeling. Upon loading in the ag_news_subset dataset, consisting of an array of news articles by the Reuters news outlet, the data was split into 114,000 training articles, 7600 test articles, and 6000 validation articles for network evaluations. Exploratory data analysis (EDA) was then performed on the dataset as a whole to assess the outcomes of vectorization within the corpus of texts. EDA during Phase 1 consisted of various vectorized encodings to identify what different vocabulary sizes may do to a network during training. Additionally, the data preprocessing included implementation of NLTK stopwords and Porterstemmer packages on the dataset. This was done to eliminate common words such as “and”, “a”, and “the” from effecting the vectorized vocabulary. Porterstemmer was used to identify and stack equivalence terms in the data.

The network experimental portion of our research consisted of 10 individual experiments separated into three phases starting with Phase 2. In Phase 2, consisting of Experiments 1-3, we constructed, trained, and tested three different Simple Bidirectional Recurrent Neural Network models on the preprocessed dataset. Each experiment in Phase 2 builds on the previous model design by implementing embedding layers and masking techniques to the model design. For all models in Phase 2, Epochs were set to 200 with and Early Stopping callback with 3 patience. Maximum tokens was set to 1000 and RMSprop was utilized for activation.

In Phase 3, consisting of Experiments 4-6, the RNN architecture from Phase 2 was modified my implementing LSTM in the models structure. Each model in Phase 3 utilizes

embedding layers and masking. The incremental changes across Phase 3 include additional stacking of LSTM layers in the structure. All design standards from Phase 2 are carried over into Phase 3.

In Phase 4, consisting of Experiments 7-10, Convolutional Neural Networks (CNN) are assess against the dataset. In Experiment 7, a single layer one-dimensional convolutional design is trained and tested on the preprocessed data. In Experiment 8, a two layer convolution design is tested and compared. Experiment 9 consists of replacing the convolutional layer with a transformer encoder layer. Positional Embedding is also implemented in Experiment 9. Experiment 10 builds upon the previous experiment with several hyperparameter adjustments as well as an additional of descending dropout layers in the model structure. The best models of each phase are then compared based on performance and model adjustments.

RESULTS

This section will cover the results of each network experiment phase in order starting from Phase 2. Comparisons will then be made when relevant to our methodology.

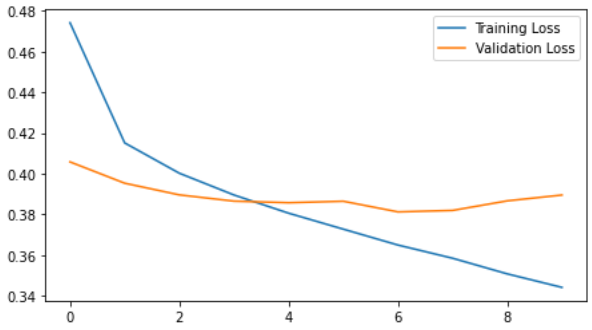
PHASE 2:

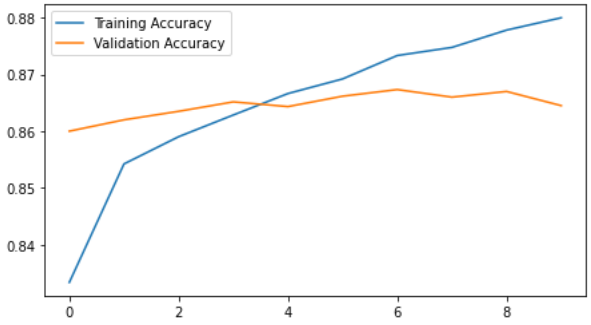
Phase 2 consisted of three simple RNN model designs. The table below shows the performance metrics of Phase 2 experiments.

Model	Type	Train Time	Train Loss	Train Acc	Val Loss	Val Acc	Test Loss	Test Acc
Model1	RNN	X	0.472	0.841	0.441	0.847	0.428	0.85
Model2	RNN	01:24:52	0.419	0.856	0.428	0.852	0.425	0.85

Model3	RNN	00:11:59	0.344	0.88	0.389	0.864	0..394	0.865

From the table, it is apparent that the RNN models are capable of performing relatively well. However, for undetermined reasons, there are relatively slow at training. The incremental changes of adding embedded layers and masking in Model2 and Model3 does appear to benefit the overall model's performance. Model3 training performance can be seen below and appears to train very well. However, without early-stopping present, the model would overfit rather quickly.





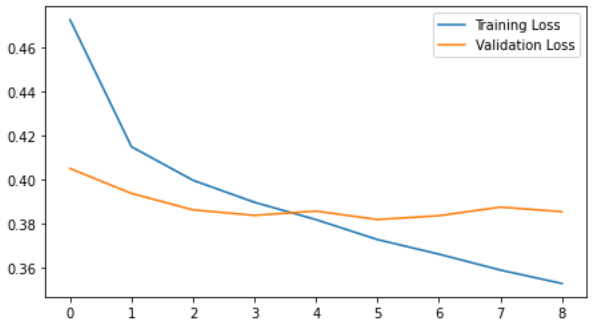
PHASE 3:

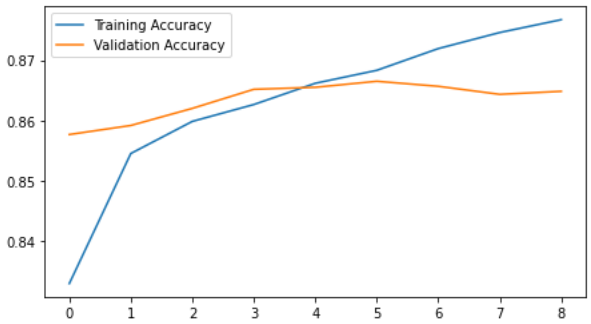
Phase 3 consists of our best RNN model from Phase 2 with embedding layers and masking. Simple RNN is replaced with LSTM and incremental adjustments are made to the architecture throughout Phase 3. Experiment 5 & 6 implements stacked LSTM layers to the original design in order to compare performance outcomes. Below is a table of the models performance.

Model	Type	Train Time	Train Loss	Train Acc	Val Loss	Val Acc	Test Loss	Test Acc
Model4	LSTM X1	00:11:06	0.352	0.876	0.385	0.864	0.395	0.861
Model5	LSTM X2	00:08:56	0.331	0.883	0.375	0.868	0.390	0.865
Model6	LSTM X3	00:13:03	0.279	0.901	0.410	0.87	0.42	0.866

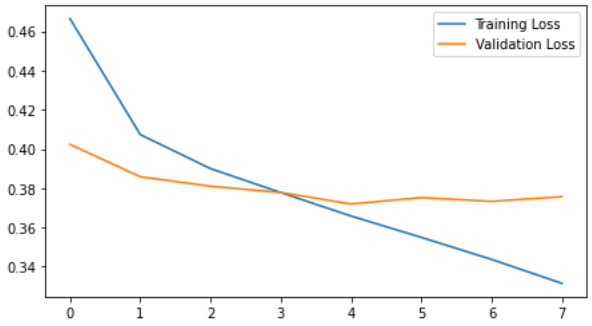
The introduction of LSTM into the model design appears to have benefitted the networks performance compared to Simple RNN layering. Additionally, the stacking of layers also appears to have a beneficial impact on performance. However, this benefit is overall minimal. Perhaps the most intriguing change too performance upon implementing LSTM and stacking is the reduction in training time. From the Phase 3 models, its apparent that duel-stacked LSTM layers both increased performance and reduced runtime compared to other Phase 3 models. Below are visualizations of the Phase 3 experiments in order.

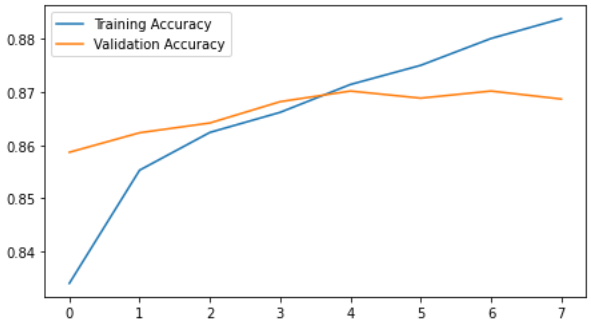
Model4



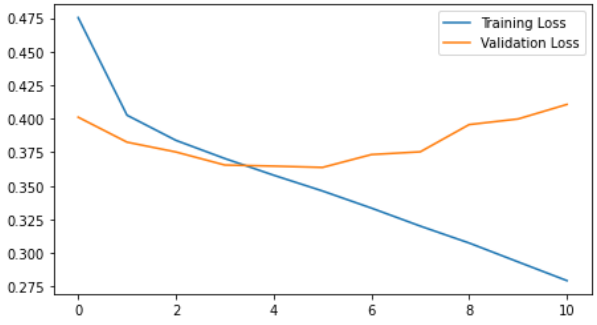


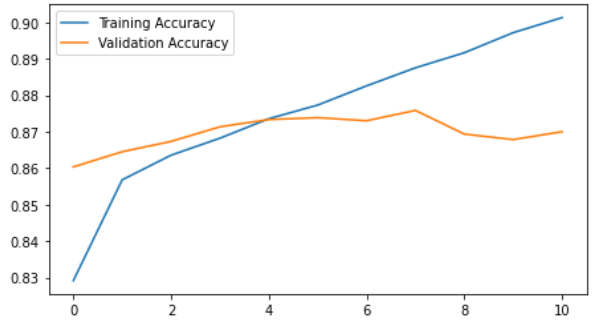
Model5





Model6





Phase 4:

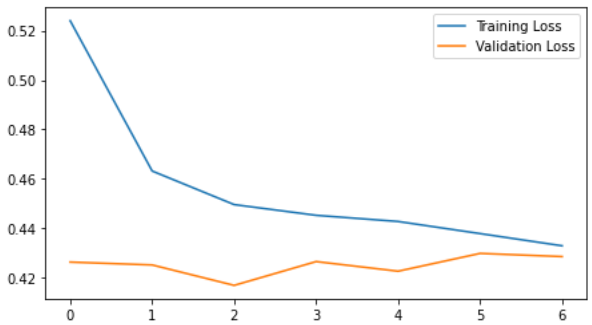
Phase 4 consists of exploration of CNN architecture on NLP tasks. Retaining all standard elements of the previous phases of experimentation, Phase 4 introduces convolutional and transformer encoder layers to the models. Additionally, global max pooling layers are introduced as necessary in CNN structure. Experiment 7 includes a single one-dimensional convolutional layer. Experiment 8, builds upon the previous experiment by stacking two convolutional layers. Experiment 9 introduces a transformer encoder layer and Experiment 10 implements positional

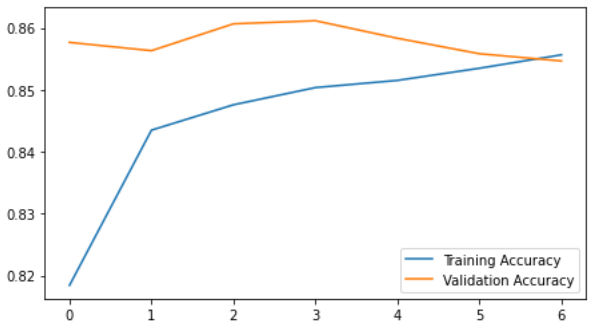
embedding techniques and descending dropout layers. Below is a table of the performance outcomes from Phase 4.

Model	Type	Train Time	Train Loss	Train Acc	Val Loss	Val Acc	Test Loss	Test Acc
Model7	CNN	00:02:18	0.432	0.855	0.428	0.854	0.45	0.851
Model8	CNN X2	00:02:01	0.443	0.857	0.503	0.849	0.548	0.84
Model9	CNN TRAN	00:08:17	0.375	0.866	0.385	0.863	0.397	0.856
Model10	CNN TRAN	00:11:36	0.395	0.861	0.554	0.855	0.572	0.84

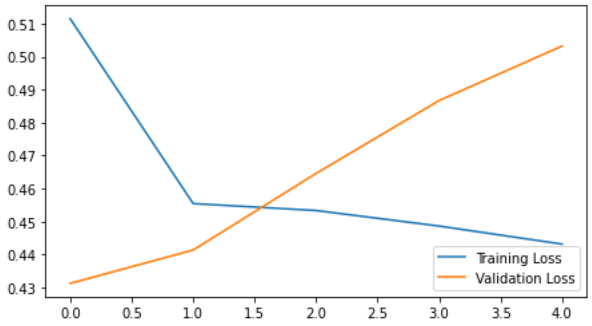
The performance metrics of Phase 4 appear to show that CNNs perform rather well against the data. The standard one-dimensional CNNs appear to drastically reduce training time. However, the doesn't appear to be any benefit in the stacking of layers in Model8. The introduction of a transformer encoder layer and positional embedding in Model9 may have had a slight impact on performance, with the overall loss on the transformer models decreasing. Model10 does not appear to out perform the previous model despite the changes in design. Below are visualizations of the performance outcomes for these 4 models.

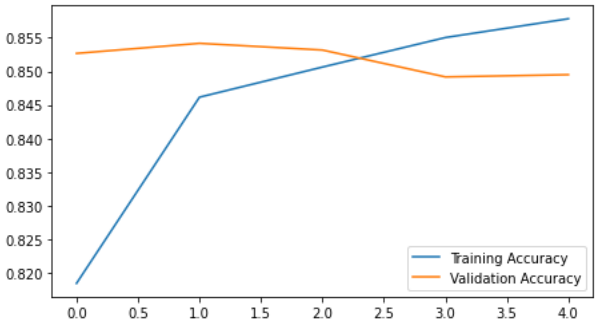
Model7



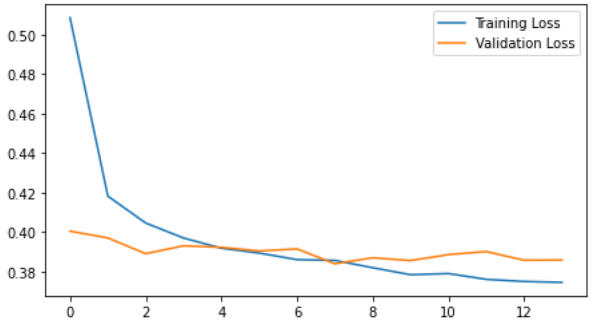


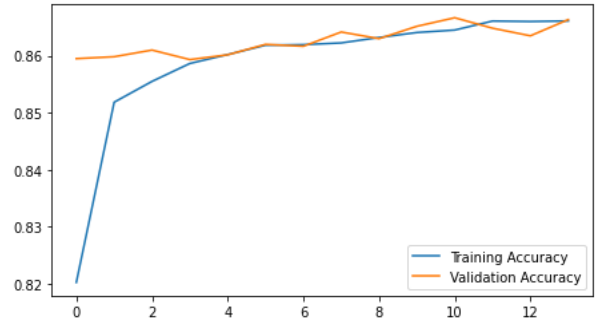
Model8



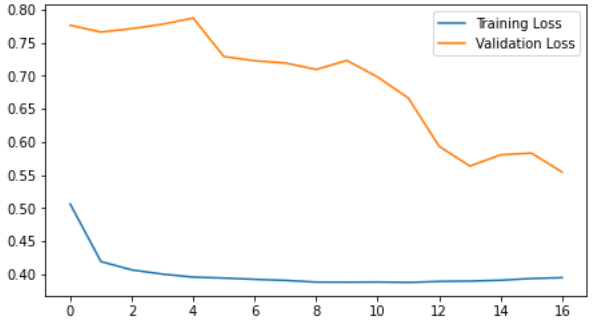


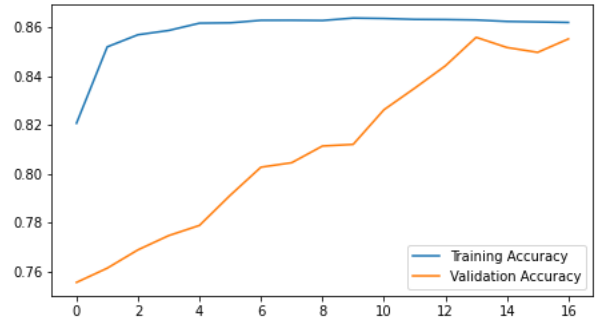
Model9





Model10





CONCLUSION

Across all four phases of experimentation, we were able to see several minor performance increases depending on the architecture of the model. The simple RNN models in Phase 2 appear to consistently perform well. However, the runtime is of primary concern during training. The implementation of embedding layers and masking did resolve this runtime issue and produce overall better performance for the RNN models.

Models with LSTM introduced additional performance improvements with increased accuracy and reduced runtime. Somewhat surprisingly, stacking LSTM layers appeared to continuously improve performance in our Phase 3 models. Further experimentation of stacked structures could prove worthwhile for increasing accuracy and reducing loss.

Phase 4 showed a slight reduction in performance compared to the LSTM models, However, the CNN models did appear to drastically reduce runtime while still maintaining an acceptable performance level. Overall, there appears to be a lot of trade offs in benefit between CNNs and LSTM. Further experimentation is likely necessary to better understand how each design can be tweaked to yield the best overall performance. Nevertheless, our data illustrates several beneficial components useful in their relative designs.

REFERENCES

Chernyavskiy, Ilvovsky, D., & Nakov, P. (2021). Transformers: "The End of History" for Natural Language Processing? In *Machine Learning and Knowledge Discovery in Databases. Research Track* (pp. 677–693). Springer International Publishing. https://doi.org/10.1007/978-3-030-86523-8_41

IBM (2022). What is Natural Language Processing? https://www.ibm.com/cloud/learn/natural-language-processing