***ASSIGNMENT 4***

***RECURRENT NEURAL NETWORK***

*MSBA 64061 Gloria Stephen*

***Objective:***

* The goal is to improve a language model's performance by examining the effects of different factors such as training data size, vocabulary size, and word embeddings. The goal is to find the model's combinational configurations and enhance its performance under various scenarios.

***Implementations:***

* RNNs are useful for text and sequence data processing because of their capacity to accept variable-length inputs and capture temporal relationships.
* To enhance RNN performance with limited data, approaches such as data augmentation are used, with the purpose of increasing the variety of the training data and decreasing overfitting.
* The best ways for enhancing RNN prediction are determined by criteria such as the type of data, the size of the dataset, and the specific job at hand. Techniques like fine-tuning and hybrid models may be useful in some cases.

|  |  |  |  |
| --- | --- | --- | --- |
| Action | Loss | Accuracy | Observation |
| Cut reviews after 150 words | 34% | 86% | The Action has a high accuracy rate but there is still potential for improvement, as seen by the loss percentage. Further training data and fine-tuning may assist in polishing the model's performance. |
| Chart, line chart  Description automatically generatedChart, line chart  Description automatically generated | | | |
| Restrict training samples to 100. | 69% | 50% | This implies that it requires major development. A detailed review of the training data and model architecture may aid in identifying methods for enhancing its performance. |
| Chart  Description automatically generatedChart  Description automatically generated | | | |
| Evaluating the performance on embedding layers on changing the sample size | | | |
| Training the embedded sample with 10K | 66% | 59% | The number of training samples changed to see when the embedding layer ran out of pre-trained word embedding. The embedding layer fared better with 10,000 training samples, obtaining a test accuracy of 59% compared to a test loss of 66%. |
| Chart, line chart  Description automatically generatedChart  Description automatically generated with medium confidence | | | |
| Training the embedded sample with 20K | 41% | 80% | Employing conv1D and embedding layers and increasing the size of the training sample can enhance the model's efficacy giving more diverse data to learn from. |
| Chart, line chart  Description automatically generatedChart, line chart  Description automatically generated | | | |
| Training the embedded sample with 30K | 44% | 81% | This aids the algorithm in better comprehending the subtleties of different reviews and generates more accurate. |
| Chart, line chart  Description automatically generatedChart, line chart  Description automatically generated | | | |

1. Which approach works better?

* The approach of evaluating the pre-trained higher sample size yields out better results clearly.

***Conclusion***:

* The use of embedding layers or pre-trained word embeddings can enhance deep learning models for text and sequence tasks such as sentiment analysis or language translation.
* The decision between the two methodologies is influenced by a number of criteria, including the amount and quality of the training data, domain-specific vocabulary, and task constraints.
* While pre-trained embeddings based on big corpora can offer a broad representation of words, embedding layers can adapt to the specific job and learn domain-specific characteristics from training data.
* Furthermore, utilizing embedding layers can assist reduce overfitting and increase the generalization capacity of the model.
* The neural network learns to discover patterns and correlations in data by modifying its parameters to minimize the gap between expected and goal outputs. Furthermore, by limiting overfitting, the model can avoid becoming overly specialized to the Pre-training data running through higher samples.

***Citations:***

* <https://scikit-learn.org/stable/tutorial/text_analytics/working_with_text_data.html>
* <https://github.com/fchollet/deep-learning-with-python-notebooks/blob/master/chapter12_part01_text-generation.ipynb>
* <https://github.com/fchollet/deep-learning-with-python-notebooks/blob/master/chapter11_part04_sequence-to-sequence-learning.ipynb>

***References:***

* Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. (2003). A neural probabilistic language model. Journal of machine learning research, 3(Feb), 1137-1155. <https://www.jmlr.org/papers/volume3/bengio03a/bengio03a.pdf>
* Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems (pp. 3111-3119). <https://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf>
* Le, Q., & Mikolov, T. (2014). Distributed representations of sentences and documents. In International conference on machine learning (pp. 1188-1196). <https://cs.stanford.edu/~quocle/paragraph_vector.pdf>