NATURAL LANGUAGE PROCESSING REPORT

cgcr45

1 Introduction

Text classification is a problem in Natural Language Processing (NLP) in which collections of texts must be sorted into one or more categories. This has wide ranging applications in domains such as information retrieval, sentiment analysis, content moderation etc. [1] where the ability to quickly categorise huge amounts of data can be invaluable. Approaches to this problem typically come in two distinct categories: rule-based methods such as Support Vector Machines (SVMs) and Hidden Markov Models (HMMs) which originated in the 1980s, and modern neural network techniques, which rose with the development of deep learning methods in the 2010s producing successful models such as BERT [2] and GPT-3 [3], both utilising modern Transformer architectures [4].

2 Problem Definition

In this paper we will be focusing on the problem of fake news detection, more specifically the task of classifying how a headline is related to an article. Our dataset includes 50,000 headlines, each attached to one of 1,700 articles. We will produce a system to classify these headlines into one of four categories: agree, disagree, discuss or unrelated to the attached article by first classifying the headlines as either related or unrelated, then classifying the related headlines as either agreeing, disagreeing or discussing the related article. There is also a testing dataset of 25,000 headlines and 900 articles to evaluate our system with.

3 Proposed Solutions

Before feeding data into models, we must convert it into a machine readable form called an embedding. We will compare two techniques: a statistical technique called TF-IDF and a neural technique using a pretrained model called BERT.

TF-IDF is a measure of how important a word is to a text. It is calculated by multiplying the Term Frequency, the frequency of a term within a document, by the Inverse Document Frequency, a measure of how rare a term is amongst all documents. A high TF-IDF value implies a term has a strong relationship with a document [5]. This allows us to represent a text as a 1D vector highlighting the words most relevant to it. We calculate these using the formulae below:

$$TF(t,d) = \frac{f_d(t)}{\sum_{t' \in d} f_d(t')} \tag{1}$$

$$IDF(t,D) = \log \frac{|D|}{|\{d \in D : t \in d\}|} \tag{2}$$

where t, d and D denote a term, document and collection of documents respectively, and $f_d(t)$ denotes the occurrences of a term t in a document d.

A statistical model like this is easy to implement, requires little computing power and gives a human-understandable way to compare texts. However, it fails to capture contextual information or semantics by focusing only on words, meaning it struggles with homonyms or synonyms. For example, two articles mentioning bird-watching and construction would be seen as similar for both mentioning 'cranes', or a user looking for articles about motorbikes may miss ones using the word 'motorcycle' [5].

The neural technique, BERT, is based on a Transformer architecture that was developed in 2018 by engineers at Google [6]. It uses an attention mechanism to learn contextual relations between 'tokens' in a text, which are components used to build up larger words. This allows us to construct any word using a vocabulary of only 30,000 tokens, for example 'snowman' would be broken down into 'snow' and '##man'. The sequence of tokens is then input, and a 768-dim vector is output for each token encoding meaningful context-aware information.

The BERT model used[1] includes 12 transformer layers and was trained on a dataset of books and wikipedia articles totalling 3.3 billion words [6]. The system can only handle inputs up to 512 tokens, shorter than many articles, so to encode as much information as possible we strip punctuation and stopwords[2] from the dataset, saving tokens for meaningful words.

The benefit of BERT over TF-IDF is that it encodes contextual information, theoretically allowing us to make more informed predictions. However, it is far more computationally expensive and since BERT is a black-box algorithm, it is very difficult to explain how it produces an embedding.

We will compare two methods for the initial classification: a simple Logistic Regression (LR) model and a type of Recurrent Neural Network (RNN) architecture called a Gated Recurrent Unit (GRU) [7].

The LR model attaches a learned weight to each entry of the embedding and uses a sigmoid function to convert the sum of the weighted vectors to a probability. While basic, it is very quick and easy to train and is effective on many simple classification tasks [8]. It only works with inputs of fixed length however, so we concatenate our headline & body embeddings and input the first tokens of our BERT embeddings as they encode information about the full text.

The GRU is a deep neural model. It was designed as a simpler version of the Long-Short Term Memory (LSTM) architecture [9], another type of RNN. While both work with sequential data, an LSTM maintains a hidden state separate from its outputs, while the GRU only has one hidden state and a simpler internal structure. This cuts down on computing and memory requirements, and testing shows it has similar performance to LSTMs in most applications [10].

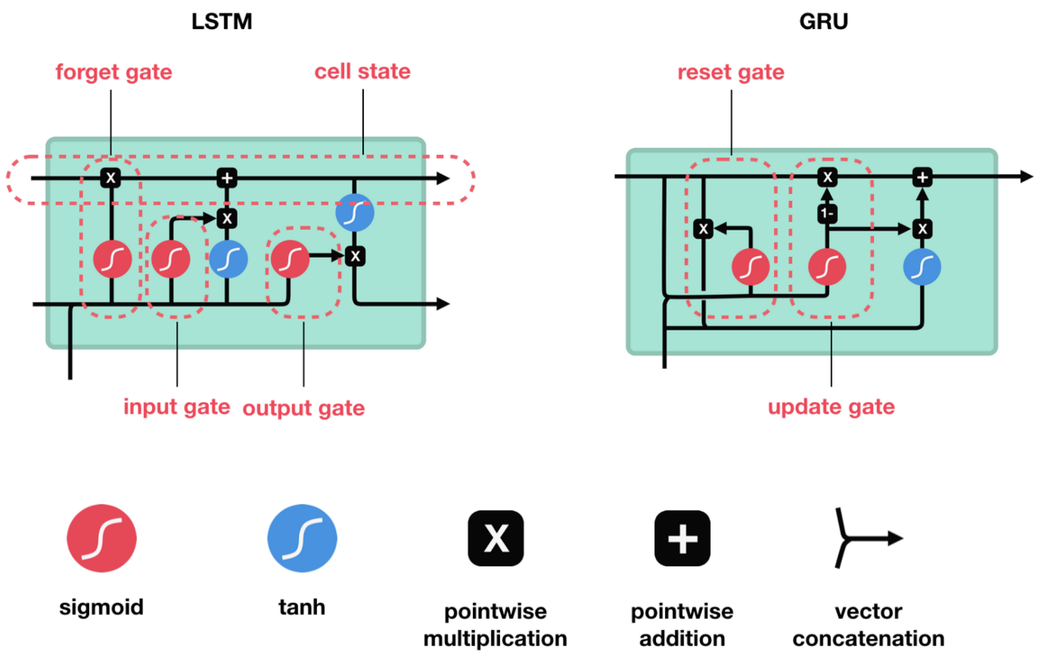


Figure 1: Schematic diagram of LSTM vs. GRU architecture. Note how LSTM has 1 more gate than GRU and outputs an extra tensor. Diagram from [11].

For BERT embeddings we train a pair of 2-layer stacked GRUs, one for headlines and one for article bodies. Stacked GRUs pass the hidden state through multiple layers at each timestep, improving feature extraction capabilities [12]. We pass the final hidden layers of the GRUs through 3 fully connected layers broken up by Rectified Linear Units (ReLU), reducing the tensor to a single value, followed by a sigmoid function to get a prediction. TF-IDF embeddings only

[1]https://huggingface.co/bert-base-uncased

[2]Stopwords are words which typically contribute little meaning to a sentence.

produce 1 tensor per text, so we train a single 2-layer stacked GRU, inputting the headline then body before passing the output through the fully connected layers.

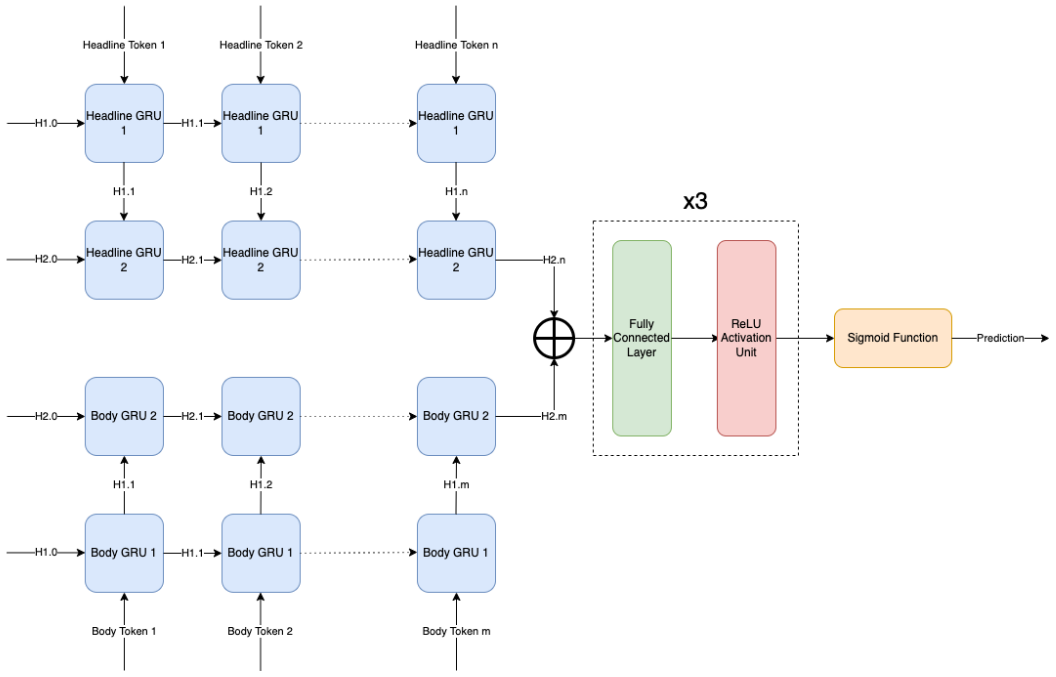


Figure 2: Schematic diagram of GRU for use with BERT embeddings.

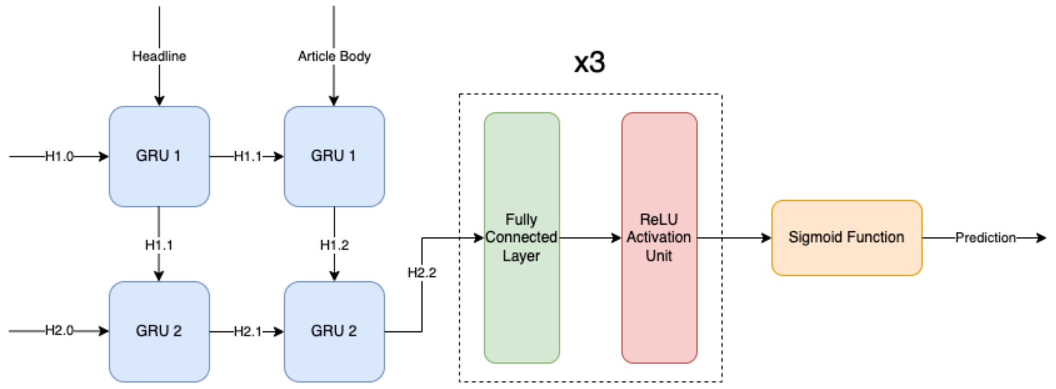


Figure 3: Schematic diagram of GRU for use with TF-IDF embeddings.

We will use the Adam optimiser [13] since it has generally good performance and Binary Cross-Entropy (BCE) loss as it is well suited to binary classification tasks. It works by calculating the log of the distance between the prediction and label, given by the formula below:

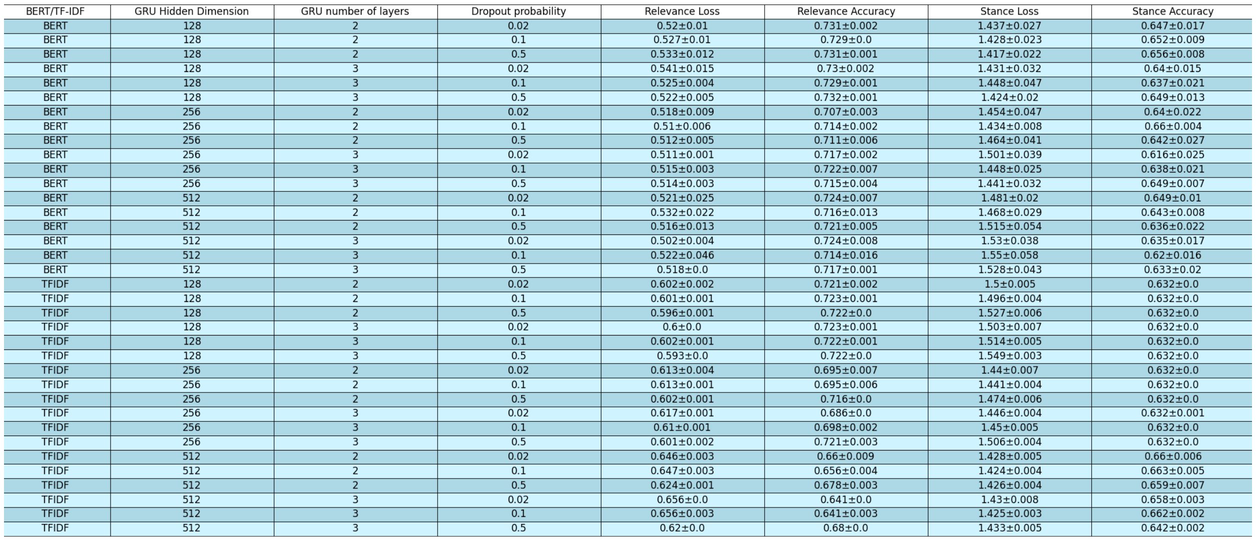
$$BCE(X, p_X) = -(X \cdot log(p_X) + (1 - X) \cdot log(1 - p_X)) \tag{3}$$

Where X denotes the class label and p_X denotes the predicted probability of belonging to class 1.

To classify related articles into the remaining categories, a similar neural network will be used except the fully connected layer will reduce the tensor to 3 values culminating in a softmax layer to produce a probability distribution over 3 values. Loss will then be calculated by summing BCE loss for the 3 classes.

4 Analysis of Results

To select suitable hyperparameters for the deep model, a range of learning rates were tried and we found 0.00001 worked best over 3 epochs for both models. We used this to test combinations of other hyperparameters, doing 5 trials and evaluating results after 1 epoch, giving the results below.

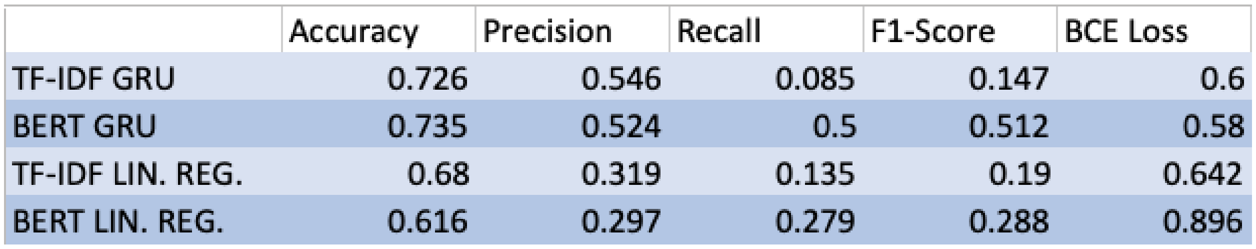


BERT/TF-IDF	GRU Hidden Dimension	GRU number of layers	Dropout probability	Relevance Loss	Relevance Accuracy	Stance Loss	Stance Accuracy
BERT	128	2	0.02	0.52±0.01	0.731±0.002	1.437±0.027	0.647±0.017
BERT	128	2	0.1	0.527±0.01	0.729±0.0	1.428±0.023	0.652±0.009
BERT	128	2	0.5	0.533±0.012	0.731±0.001	1.417±0.022	0.656±0.008
BERT	128	3	0.02	0.541±0.015	0.73±0.002	1.431±0.032	0.64±0.015
BERT	128	3	0.1	0.525±0.004	0.729±0.001	1.448±0.047	0.637±0.021
BERT	128	3	0.5	0.522±0.005	0.732±0.001	1.424±0.02	0.649±0.013
BERT	256	2	0.02	0.518±0.009	0.707±0.003	1.454±0.047	0.64±0.022
BERT	256	2	0.1	0.51±0.006	0.714±0.002	1.434±0.008	0.66±0.004
BERT	256	2	0.5	0.512±0.005	0.711±0.006	1.464±0.041	0.642±0.027
BERT	256	3	0.02	0.511±0.001	0.717±0.002	1.501±0.039	0.616±0.025
BERT	256	3	0.1	0.515±0.003	0.722±0.007	1.448±0.025	0.638±0.021
BERT	256	3	0.5	0.514±0.003	0.715±0.004	1.441±0.032	0.649±0.007
BERT	512	2	0.02	0.521±0.025	0.724±0.007	1.481±0.02	0.649±0.01
BERT	512	2	0.1	0.532±0.022	0.716±0.013	1.468±0.029	0.643±0.008
BERT	512	2	0.5	0.516±0.013	0.721±0.005	1.515±0.054	0.636±0.022
BERT	512	3	0.02	0.502±0.004	0.724±0.008	1.53±0.038	0.635±0.017
BERT	512	3	0.1	0.522±0.046	0.714±0.016	1.55±0.058	0.62±0.016
BERT	512	3	0.5	0.518±0.0	0.717±0.001	1.528±0.043	0.633±0.02
TFIDF	128	2	0.02	0.602±0.002	0.721±0.002	1.5±0.005	0.632±0.0
TFIDF	128	2	0.1	0.601±0.001	0.723±0.001	1.496±0.004	0.632±0.0
TFIDF	128	2	0.5	0.596±0.001	0.722±0.0	1.527±0.006	0.632±0.0
TFIDF	128	3	0.02	0.6±0.0	0.723±0.001	1.503±0.007	0.632±0.0
TFIDF	128	3	0.1	0.602±0.001	0.722±0.001	1.514±0.005	0.632±0.0
TFIDF	128	3	0.5	0.593±0.0	0.722±0.0	1.549±0.003	0.632±0.0
TFIDF	256	2	0.02	0.613±0.004	0.695±0.007	1.44±0.007	0.632±0.0
TFIDF	256	2	0.1	0.613±0.001	0.695±0.006	1.441±0.004	0.632±0.0
TFIDF	256	2	0.5	0.602±0.001	0.716±0.0	1.474±0.006	0.632±0.0
TFIDF	256	3	0.02	0.617±0.001	0.686±0.0	1.446±0.004	0.632±0.001
TFIDF	256	3	0.1	0.61±0.001	0.698±0.002	1.45±0.005	0.632±0.0
TFIDF	256	3	0.5	0.601±0.002	0.721±0.003	1.506±0.004	0.632±0.0
TFIDF	512	2	0.02	0.646±0.003	0.66±0.009	1.428±0.005	0.66±0.006
TFIDF	512	2	0.1	0.647±0.003	0.656±0.004	1.424±0.004	0.663±0.005
TFIDF	512	2	0.5	0.624±0.001	0.678±0.003	1.426±0.004	0.659±0.007
TFIDF	512	3	0.02	0.656±0.0	0.641±0.0	1.43±0.008	0.658±0.003
TFIDF	512	3	0.1	0.656±0.003	0.641±0.003	1.425±0.003	0.662±0.002
TFIDF	512	3	0.5	0.62±0.0	0.68±0.0	1.433±0.005	0.642±0.002

Figure 4: Results from parameter search.

For relevance classification, BERT embeddings are superior with lower loss values and slightly higher accuracy, while for classification performance is roughly equal. We therefore decide to use BERT embeddings, and after further testing we found a 2-layer BERT model with a 256-dim hidden space and dropout of 0.1 is best for relevance and a 128-dim hidden space for stance classification.

We can now compare our deep learning models against the logistic regression models, since these had no hyperparameters to select. Performance on the testing set alongside additional metrics can be seen below:

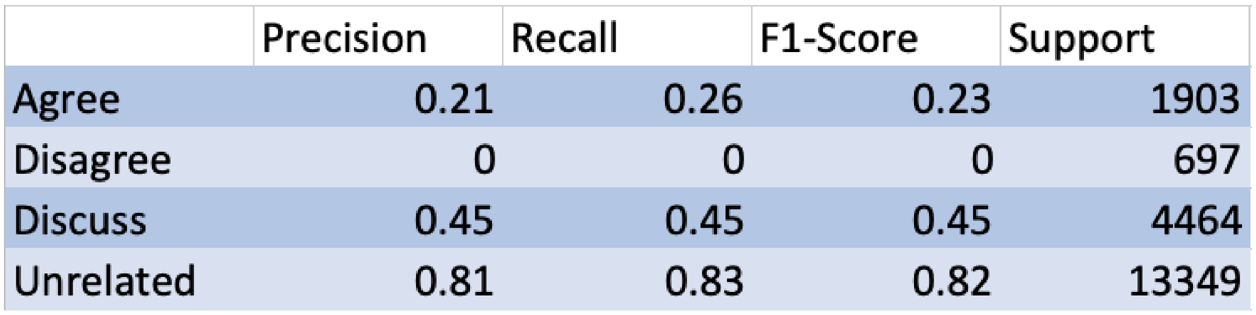


	Accuracy	Precision	Recall	F1-Score	BCE Loss
TF-IDF GRU	0.726	0.546	0.085	0.147	0.6
BERT GRU	0.735	0.524	0.5	0.512	0.58
TF-IDF LIN. REG.	0.68	0.319	0.135	0.19	0.642
BERT LIN. REG.	0.616	0.297	0.279	0.288	0.896

Figure 5: Table showing model performances on testing set.

Our deep models outperform the logistic regression models, and BERT outperforms TF-IDF in nearly all metrics. Differences in recall between TF-IDF and BERT models are strikingly high. This suggests the BERT models identify significantly more of the relevant articles, even if a slightly lower proportion of the recalled articles are relevant. Since the BERT-GRU model performs best in almost every metric, it makes sense to implement it for both stages in our final system. Our final model had an overall accuracy of 0.694, with detailed metrics below.

Interestingly, our model never predicted disagree. This could be due to it being heavily underrepresented in both sets, with only 10% of relevant headlines disagreeing. This would also explain why agree has a lower precision and recall than discuss: if a class occurs rarely in training, the model can avoid predicting it without significant penalty.



	Precision	Recall	F1-Score	Support
Agree	0.21	0.26	0.23	1903
Disagree	0	0	0	697
Discuss	0.45	0.45	0.45	4464
Unrelated	0.81	0.83	0.82	13349

Figure 6: Table showing final model performance.

5 Discussion

Performance was good when classifying relevant/irrelevant headlines but not on the more subtle classification of agree/disagree/discuss. This could be due to imbalances in the training data, with these classes having very little representation, or from our model not being complex enough to learn the training data effectively.

6 Ethical Implications

No analysis was done on the content of the dataset, so we don't know what articles and headlines are represented. There could be an over-representation of, for example, disagreeing articles surrounding a particular public figure which could cause our model to flag articles mentioning this figure as disagreeing, even if they agree with the headline. This could damage the persons reputation if the headline was positive, or vice-versa. People aiming to produce fake news could also use our model to try and create more convincing fake news to trick moderation systems. As a result, our system should not be trusted by its own, and should be used as a tool to aid human classifiers.

7 Conclusion

Our work has shown that for headline classification, deep neural networks are vastly superior to traditional methods thanks to their ability to learn complex nonlinear relationships and understand contextual information.

References

[1] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao, "Deep learning based text classification: A comprehensive review," *CoRR*, vol. abs/2004.03705, 2020.

[2] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," *CoRR*, vol. abs/1810.04805, 2018.

[3] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," *CoRR*, vol. abs/2005.14165, 2020.

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *CoRR*, vol. abs/1706.03762, 2017.

[5] J. Ramos, "Using tf-idf to determine word relevance in document queries," 01 2003.

[6] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *NAACL*, 2019.

[7] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," 2014.

[8] J. S. Cramer, "The origins of logistic regression," 2002.

[9] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Computation*, vol. 9, pp. 1735–1780, 11 1997.

[10] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Empirical evaluation of gated recurrent neural networks on sequence modeling,” 2014.

[11] M. Phi, “Illustrated guide to lstm’s and gru’s: A step by step explanation,” Jun 2020.

[12] F. Pan, J. Li, B. Tan, C. Zeng, X. Jiang, L. Liu, and J. Yang, “Stacked-gru based power system transient stability assessment method,” *Algorithms*, vol. 11, no. 8, 2018.

[13] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2014.