[Group 27] CS4248 Final Report

A0290882U, A0276528R, A0285647M, A02179062X, A0214260W, A0233573E
Group 27
Mentored by Rishabh Anand
{e1327913, e1132261, e1216292, e0543998, e0518544, e0725573}@u.nus.edu

Abstract

In this project, we aim to answer a simple question, namely: "Why is it so difficult to get good performance on the LUN dataset?". Our methodology can be broken down into two main categories: analyzing the existing LUN dataset, as well as modifying the LUN dataset and re-training models on the modified dataset. Ultimately, our group was unable to find the exact problems with the LUN dataset, but we have discovered that the low performance is not because of the wrong choice of the model. The source code to our project can be found in this footnote[1].

1 Introduction and motivation

Fake news is a relatively new but rampant problem in today's digital society. As a result, any method to reliably detect fake news is highly desirable. One way to automatically detect fake news would be to train a machine learning model to classify a piece of text as fake news or not. To this end, Rashkin et al. have constructed a dataset, known as the LUN dataset which labels a piece of text as one of four labels: *hoax*, *propaganda*, *satire*, or *reliable information*. (Rashkin et. al, 2017). If we could train a machine learning model on this LUN dataset and achieve good performance, it would help tackle fake news. However, in Rashkin's paper, they could only achieve an F1 score of 65% even with a state-of-the-art LSTM model. Our group was surprised that even SOTA models could not perform well on this dataset. Hence we decided to analyze the dataset more closely to understand why even SOTA models cannot perform well. To better understand the dataset, our group did two things: First, we augmented the original LUN dataset by adding new features, as well as adding new rows obtained from asking the Mixtral LLM. The dataset augmentation aims to diagnose whether issues lie within the dataset or the models themselves. While it may not improve model performance, it helps identify potential dataset-related issues. We trained four distinct models on both the original and augmented datasets to assess if model choice was a factor. Furthermore, we employed Captum for in-depth model analysis and interpretation.

[1]Source code for our project can be found here: https://github.com/charliemoweng/4248_project_Dissect_LUN

2 Related work and background

2.1 Labelled Unreliable News

In Rashkin's work on the LUN dataset, lexicon's were found to be useful indicators to check between reliable and unreliable news by getting a deeper understanding of the linguistic differences between the classes (Rashkin et. al, 2017). Zhao Wenbin's research uses a BERT architecture for knowledge representation for the task of fake news detection resulting which performs better than the original paper's LSTM (Zhao et. al, 2024).

2.2 Latent Dirichlet Allocation

Following a notion similar to LiJuan Sun, where incorporation of external knowledge and being topic-aware improves the model's performance (Sun & Wang, 2023), in our work we have incorporated Topic Modelling. Sun's paper uses a graph-based Neural Network modelling to make the model topic-aware. On the other hand, we have implemented Latent Dirichlet Allocation, which is a generative probabilistic Bayesian form of modelling in which each item in the collection is modeled as a finite mixture over an underlying set of topics (Blei et al, 2003).

2.3 Swear Word Analysis

Swear words are very versatile as they are able to perform different interpersonal functions according to different contexts ranging from expressing aggression to signaling group identity of the informality of the communication. (Holgate et al., 2018). As our dataset contains swear words, we believe that this is a good opportunity to analyse how swear words impacts text classification.

2.4 Valence Aware Dictionary and Sentiment Reasoner (VADER)

VADER is a lexicon and rule-based sentiment analysis tool that is specifically meant to analyse sentiments expressed in social media (Hutto & Gilbert, 2014). Social media network share similarities with traditional media such as newspapers where social media and traditional media plays a role in incidental exposure to news (Bergström & Belfrage, 2018). As such, VADER was used to perform sentiment analysis on the dataset

2.5 Bidirectional Encoder Representations from Transformers (BERT)

BERT is a language model that was designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers (Devlin et al., 2019). As BERT are adapted frequently in supervised text classification (González-Carvajal & Garrido-Merchán, 2021), we believe that this can apply to us as well.

2.6 XGBoost

XGBoost is a state-of-the-art tree boosting system that has been used in many machine learning problems (Chen & Guestrin, 2016). Even though it is not as optimized to work with NLP problems as other methods such as transformers, we decided to train it due to its integration with scikit-learn, which makes it much easier to train.

2.7 LinearSVC

LinearSVC is a linear Support Vector Machine (SVM)(Vapnik, 1995) tailored for classifying linearly separable data. With our feature space being linearly separable due to our vectorization methods (bag-of-words, TF-IDF), LinearSVC is chosen as it implements the "one-versus-one" approach for multi-class classification, suited for our 4-way classification problem.

3 Corpus Analysis and Methods

3.1 Valence Aware Dictionary and Sentiment Reasoner

VADER was used to measure the news articles' sentiments in effort to better understand why these news articles belongs to their respective classes. We measured the mean negative, mean positive, mean neutral and mean compound polarity scores per class in order to understand the average sentiment values per class.

We noticed that in Figure 1, *hoax* and *propaganda*

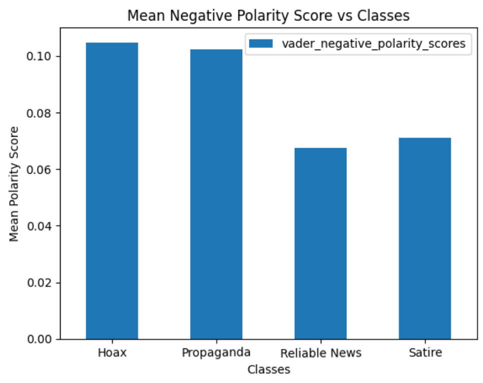

Figure 1: VADER Mean Negative Polarity Score per Class

news contain more negative sentiment compared to *reliable news* and *satire*. This tells us that we should expect news that have negative sentiment can be more towards *hoax* or *propaganda*.

In Figure 2 and 3 respectively, on average, all four

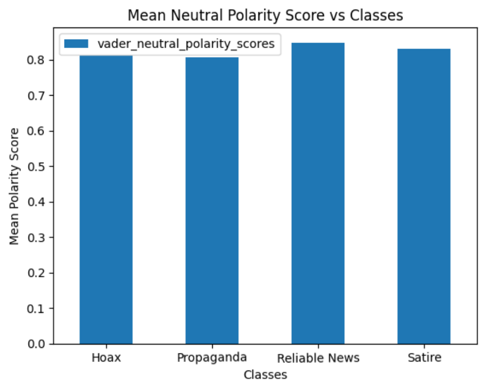

Figure 2: VADER Mean Neutral Polarity Score per Class

classes have neutral and positive sentiments which tells us that any news articles could fall under these classes.

For Figure 4, we can see that *hoax* and *propaganda* are towards the negative sentiment while *reliable news* and *satire* are towards the positive sentiment. To summarize from this analysis, we can see that *hoax* and *propaganda* tends to contain negative sentiment while *reliable news* and *satire* contain positive sentiment.

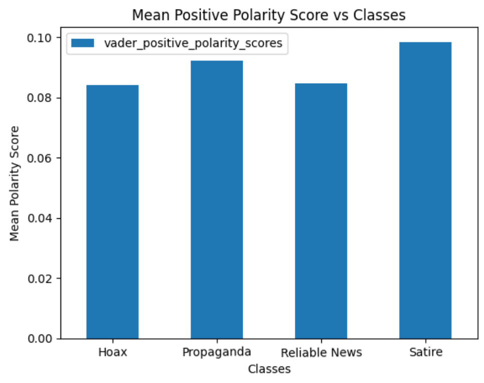

Figure 3: VADER Mean Positive Polarity Score per Class

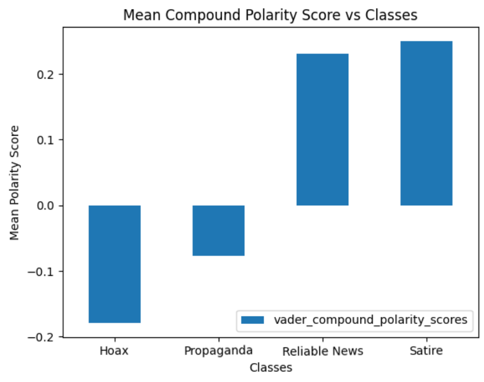

Figure 4: VADER Mean Compound Polarity Score per Class

3.2 Augmentation using LLM Script

Given the imbalance in the LUN dataset, we utilized the Mixtral-46.7B LLM for data augmentation from Hugging Face Spaces, aiming to balance the classes. While the LLM-generated paraphrases appeared promising on manual inspection, we noted that the evaluation using METEOR and ROGUE metrics yielded average or poor scores. This suggests potential limitations with the metrics themselves perhaps due to the nature and length of the data points. Nevertheless, we decided to proceed with adjusting the counts of *hoax* and *reliable news* categories to match *satire* by taking a leap of faith.

3.3 Advanced EDA and Feature Engineering on Augmented Dataset

Identification of Swear Words and Severity ’has_swear’, ’severity’ feature: For our analysis, we decided to use Surge-AI’s profanity dataset which contains over 1600 popular swear words, their colloquial usage, canonical forms and severity of each swear word. Using the swear words dataset,

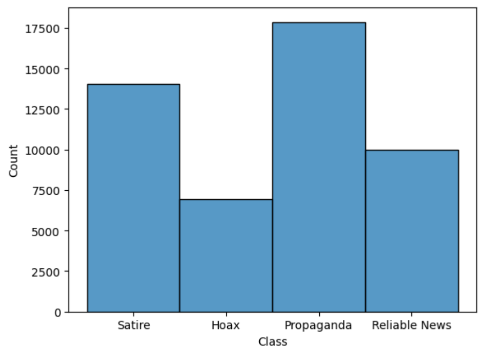

Figure 5: Total counts per class in LUN Dataset

the distribution of swear words and severity of articles were analyzed. In the dataset, a severity score is assigned for each swear word. The severity score of an article would be the sum of the severity scores of all swear words present in it. If an article does not contain even a single swear word, then the severity score would be 0. The average severity score of a class is the sum of severity scores of all articles belonging to that class divided by the number of articles in that class.

Class	% of Rows	Avg Severity Rating
1	24.98	0.72
2	9.46	0.21
3	22.10	0.49
4	13.51	0.27

Table 1: Statistics of Swear Words in Each Class. **The percentage of rows with swear words in classes 1 and 3 are almost double in comparison to classes 2 and 4**

It can be noticed that almost 25% of class-1 articles contain swear words and has the highest average severity score amongst all classes, 0.72 and ∼23% of class-3 articles contain swear words, with an average severity score of 0.49. On the other hand, articles of class-2 and class-4 do not contain as many swear words. Hence we encoded ’has_swear’ and ’severity’ as new feature-engineered columns in the augmented dataset since they would help the model in distinguishing the classes.

Latent Dirichlet Allocation (LDA) Topic Modeling: Latent Dirichlet Allocation (LDA) is a Bayesian statistical method used for modelling the automatically extracted topics in the text data. Assigning topics to the news articles in the dataset and analysing the distribution would help us distinguish between the classes in a better manner. We experimented with different numbers of topics and fixated on num_topics as '5' since having 10 or 20 topics makes the distribution of topics across classes uniform, which would not help in differentiating the classes. While LDA would assign one of the five topics to each row of the dataset, similar to clustering, it is up to us to identify what each topic/cluster means. After running the LDA model and obtaining a topic for each article, from our manual analysis of the articles in each topic and the top words of each topic, we decided to label each topic as the following: Topic 0: General Opinions and Discussions, Topic 1: Politics and Government Affairs, Topic 2: Economic Discussions, Topic 3: Health and Medical Topics, Topic 4: News Events and Incidents. The distribution of articles across topics and class labels was then analyzed.

Label	T-1	T-2	T-3	T-4	T-5
1	55.98	13.54	2.81	8.74	18.92
2	16.89	48.25	0.17	1.45	33.23
3	13.53	49.13	5.51	27.49	4.34
4	10.59	34.59	30.37	9.02	15.44

Table 2: LDA Matrix 1: Percentage of rows per topic within each class (T indicates Topic). **Each class has a 'favourite' topic, i.e it comprises mostly of one particular topic.**

LDA Matrix 1: For each class, what is the composition of its rows across the 5 topics? (each row sums up to 1) In the rows, we have the 4 classes[1-hoax, 2-satire, 3-propaganda, 4-reliable], and in the columns, we have our 5 topics[0-4]. 55% of class-1 articles belong to topic-1. 48% of class-2 articles belong to topic-2. 49% of class-3 articles belong to topic-1 and 27% belong to topic-3. 34% of class-4 articles belong to topic-1 and 30% belong to topic-2.

Each topic has a 'favourite' class and each class has a 'favourite' topic. Hence, encoding the topic as a feature would help the models in distinguishing between classes.

4 Experiments

4.1 LinearSVC

TF-IDF vectorization was carried out with stop words removed and n-gram feature (unigrams and bigrams). Both stop words removal and n-gram feature are used as input parameters to the TF-IDF vectorizer. Four hyper-parameters were examined for their suitability to be tuned further.

C (Regularization Parameter) It controls the trade-off between maximizing the decision boundary and minimizing the classification error. Its value represents the inverse of regularization strength.

penalty (Penalization Norm) It specifies the norm used in the penalization term of the loss function. Either 'l1' or 'l2' regularization is applied to the model coefficients.

loss (Loss Function) It specifies the loss function used during optimization, either 'hinge' or 'squared_hinge'. The difference is that 'hinge' loss is more sensitive to outliers and may lead to larger gradients during optimization compared to 'squared_hinge' loss

dual (Dual Formulation) It determines whether to solve the primal or dual optimization problem. It's recommended to set it to 'False' when the number of samples is greater than the number of features, based on the documentation.

For the value of dual, according to the documentation, it prefers dual=False when $n_{\text{samples}} > n_{\text{features}}$. After data exploration, $n_{\text{samples}} = 48854$ and $n_{\text{features}} = 7388647$ for the original dataset, while $n_{\text{samples}} = 59795$ and $n_{\text{features}} = 7618304$ for the augmented dataset. In both cases, $n_{\text{samples}} < n_{\text{features}}$. This could be due to TF-IDF vectorization being used. Therefore, dual is set to be True, which is the default value.
GridSearchCV was used to determine the optimal value of C. Among the values used, the best was determined to be C=10.0. LinearSVC model was run using both the default $C = 1.0$ and $C = 10.0$. There is no significant change in performance metrics, where both yield accuracy $= 0.76$ and f1_score $= 0.75$.

4.2 XGBoost

We tuned the hyperparameters using sklearn's GridSearchCV. Given our time and compute constraints, we only decided to tune three hyperparameters, namely **n_estimators**: the number of models in the ensemble, **eta**: the learning rate, and **max_depth**: the maximum depth of each tree-based model in the ensemble Tuning the hyperparameters was only done on the original dataset. We would be running the same model on the augmented dataset anyway for comparison, so there was no need to do hyperparameter tuning on both datasets.

4.3 BERT

From our experiments, we noticed that the model performance improves when it is trained on our augmented dataset. Therefore, for this experiment, we decided to just test on the augmented dataset. For our experiments, we utilized a Pretrained BERT model, "bert-base-uncased" from HuggingFace.

Model	**Epochs**	**Learning Rate**	**Token Length**	**Batch Size**
Pretrained BERT(Zero-Shot)	None	None	512	8
Finetuned BERT	3	0.00003	512	8
Finetuned BERTwith extra features	3	0.00003	512	8

Table 3: BERT Experiment Configurations. **Finetuned BERT performs the best over the other configurations**

We ran three experiments with the following models with the following configurations stated in Table 3. The hyperparameters were selected due to recommendation in (Devlin et al., 2019) for finetuning. Each models are evaluated using the F1 Macro and F1 Micro scores.

4.4 Basic Feed-Forward Neural Network

Our team has also trained a basic feed-forward neural network as a baseline model for comparison. Our rationale for the neural network model is its simplicity, which offers a clear starting point for understanding the problem's complexity and how well more sophisticated models might perform. In our first iteration, we incorporated the following features in our neural network: *1. Vectorisation using Word2Vec "word2vec-google-news-300"*, a pretrained model from Google News data using the Word2Vec algorithm. Word2Vec provides a dense and more meaningful representation of words by capturing the context in which they appear, which is more suitable for tasks involving understanding semantic relationships between words.*2. Using Adam as the optimisation algorithm.* Adam is an adaptive learning rate optimisation algorithm designed to converge faster and more efficiently. *3. L2 regularization:* Penalises large weights during training, which helps to prevent overfitting. *4. Adding dropout layers as a form of regularisation to prevent overfitting. 5. Including two hidden linear layers* to the neural network to increase its capacity for learning more complex patterns. *6. Activation Layers with Rectified Linear Unit (ReLU):* Introduces non-linearity to the model, allowing for the learning of complex patterns.

On top of the new features extracted during data augmentation, our team also performed additional feature engineering for both our original and augmented datasets, which include: 1. sentiment analysis with TextBlob, 2. finding the occurrence of each POS tag in a sentence, 3. unique words vs sentence length ratio, 4. sentence length. Our team chose these feature engineering steps as they have proven to be effective in increasing our F1 score in Assignment 2. A 80%-20% train-validation split was also performed on the training data. However, our model saw low F1 scores during testing, with 0.329 when trained on the original dataset and 0.306 when trained on the augmented dataset. Hence, our team suspected overfitting for our model and decided to remove one hidden layer from the neural network. Next, we employed Optuna to perform hyperparameter tuning. Given our constraints on time and computational power, we only decided to tune three hyperparameters, namely: 1. The dropout rate, 2. The learning rate, 3. The number of epochs.

Tuning the hyperparameters was only performed on the original dataset. We will use the same set of model hyperparameters when training the augmented dataset for direct comparison. Optuna uses a Bayesian optimisation technique called Tree-structured Parzen Estimator, which ensures efficiency and the ability to handle a large search space, allowing Optuna to quickly find good hyperparameters without exhaustively testing every combination (Lim, 2022).

5 Discussion

5.1 Results of models

5.1.1 Linear SVC

The performance metrics results are shown as follows, after running the LinearSVC model.

Model	**Accuracy**	**Macro F1 Score**	**Micro F1 Score**
LinearSVC (in-domain test)	0.97	0.97	0.97
Finetuned LinearSVC (original dataset)	0.76	0.75	0.75
Finetuned LinearSVC (augmented dataset)	0.77	0.76	0.76

Table 4: LinearSVC Model Performance Metrics with three different datasets.

Based on the above results, the LinearSVC model performs surprisingly well for in-domain tests (when training corpus is split into 80% training and 20% testing). This performance

is better than the Max-Entropy classifier used by Rashkin et al. (2017). One possible explanation is that this dataset, with TF-IDF vectorization, stop words removal, and n-gram features (including unigrams and bigrams), might be more linearly separable in its feature space, benefiting the LinearSVC model.
The augmented dataset results in better performance compared to the original, showing the significance of additional features such as having swear words.

5.1.2 XGBoost

Running the tuned XGBoost model on both datasets, we obtained the following scores.

Class	F1 Macro Score	F1 Micro Score	Accuracy
0	0.70	0.66	0.66
1	0.60	0.66	0.66
2	0.52	0.66	0.66
3	0.77	0.66	0.66
average	0.65	0.65	0.66

Table 5: XGBoost Original Dataset Evaluation Metrics

The average F1 score for the original dataset was 0.65. **This score corroborates the results obtained by Rashkin et. al, and is a sign that the training of the model was done correctly.**

Class	F1 Macro Score	F1 Micro Score	Accuracy
0	0.72	0.63	0.63
1	0.61	0.63	0.63
2	0.42	0.63	0.63
3	0.73	0.63	0.63
average	0.62	0.62	0.63

Table 6: XGBoost Augmented Dataset Evaluation Metrics

The average F1 score obtained was 0.62. **This is slightly lower than the original dataset, and it showed that for the XGBoost model the data augmentation did not help in improving the performance of the model.**

5.1.3 Basic feed-forward Neural Network

We use the macro average F1 score (rounded to 3 significant figures) to evaluate the performance of our models from the different iterations of the neural network, summarised in the tables below:

For both original and augmented datasets, the test F1 score is significantly lower than the validation F1 score. This suggests that the test dataset might contain text with different expressions or linguistic patterns compared to those in the training dataset. The model may have learnt these specific patterns or noise in the training set, making it hard to generalise to new data. Also, after training on the augmented dataset, the model's validation F1 score increased but experienced a decrease in its test F1 score. This could be due to data augmentation having captured and emphasised linguistic patterns or noise in the training data which are absent in the test data, thereby reinforcing more bias in our model, directing it away from making accurate predictions for the test data.

Model	Neural network (NN) with 2 hidden layers	
Dataset	Original	Augmented without additional features
Validation F1 Score	0.835	0.851
Test F1 Score	0.329	0.306

Table 7: Results of NN model with 2 hidden layers, before hyperparameter tuning

Model	NN with 1 hidden layer		
Dataset	Original	Augmented without additional features	Augmented with additional features
Validation F1 Score	0.851	0.864	0.865
Test F1 Score	0.381	0.345	0.346

Table 8: Results of NN model with 1 hidden layer, before hyperparameter tuning

After removing one hidden layer from the neural network, both validation and test F1 scores increased on both the original and augmented datasets. This supports the assumption that removing one hidden layer reduces overfitting and allows the model to better generalise to unseen data. Furthermore, after including the additional features when training the augmented dataset, there is an increase in both validation and test F1 scores by 0.001, suggesting that the additional features' contribution to the model's prediction is not very significant (although positive).

Model	NN with 1 hidden layer, optimised hyperparameters	
Dataset	Original	Augmented with additional features
Validation F1 Score	0.852	0.866
Test F1 Score	0.442	0.349

Table 9: Results of NN model with 1 hidden layer, after hyperparameter tuning

Hyperparameter tuning with respect to the training data increased the train F1 score for the original dataset by around 0.06. However, the increase is limited for the augmented dataset (with additional features), at only 0.003. This is likely attributed to the fact that the hyperparameters were only optimised for the original dataset and not for the augmented one.

5.1.4 BERT

After running our experiments based on Table 3, here are our results:-

Model	Accuracy (%)	Micro F1 Score	Macro F1 Score
Pretrained BERT (Zero-Shot)	0.250	0.250	0.101
Finetuned BERT	0.594	0.594	0.527
Finetuned BERT with Extra Features	0.586	0.586	0.534

Table 10: BERT Experiment Results

From our results, we observe that fine-tuning BERT models plays a huge role in having better performance. This also lead us to suspect the pretrained BERT model that we used was not trained on such corpora like the LUN Dataset or any other fake news datasets. From our two fine-tuned models, their performances have a 0.01 difference between their F1 Macro score and F1 Micro score. Although feature engineering is still an important aspect in Natural Language Processing, this tell us that crafting these linguistic features doesn't really give a huge boost in performance and might not be worth the effort as it requires someone with a deep domain knowledge to craft these features. It also tells us that how BERT can be non-reliant on these extra crafted features to perform well because of BERT inherent understanding of the word representations from unlabeled text (Devlin et al., 2019)

5.2 Interpretability of models

5.2.1 Basic Feed-Forward Neural Network with CAPTUM

Our team utilised CAPTUM to analyse how positive or negative the attribution to each feature of our neural network is. A positive attribution means the feature is directing the model's prediction towards the target class, while a negative attribution means the feature is directing against the target class. We then plotted 1 heatmap each for the original and augmented dataset (with additional features), with brighter colours representing a certain feature having a more positive attribution and darker colours representing a certain feature having a more negative attribution.

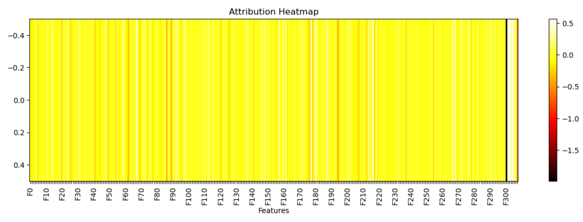

Figure 6: Heatmap for original dataset.

Top 5 features with the most positive attribution (impact): **302** (occurrence of nouns in a sentence), **301** (sentence length), **304** (occurrence of adjectives in a sentence), **67**, **177**. Top 5 features with the most negative attribution (impact): Features **300** (unique words vs sentence length ratio), **307** (sentiment subjectivity), **194** (Word2Vec feature), **86**, **89**. Features 0 to 300 are Word2Vec features.

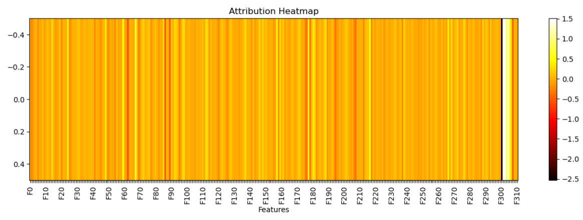

Figure 7: Heatmap for augmented dataset.

The top 5 features with the most positive attribution (impact) are: **302** (occurrence of nouns in a sentence), **301** (sentence length), **304** (occurrence of adjectives in a sentence), **303** (occurrence of verbs in a sentence), **305** (occurrence of adverbs in a sentence). Top 5 features with the most negative attribution (impact): Features **300** (unique words vs sentence length ratio), **62**, **86**, **307** (sentiment subjectivity), **89**.
From the heatmaps, we can see that the 1. occurrence of nouns in a sentence, 2. sentence length and 3. the occurrence of adjectives in a sentence are consistently the top 3 features with the most positive contribution to the model's prediction. Whereas for the features with the most negative contribution to the model's prediction, 1. unique words vs sentence length ratio and 2. sentiment subjectivity are consistently in the top 5. Most of the first 300 features (from Word2Vec) contribute positively to the model's predictions on the original dataset, compared to most of them contributing negatively on the augmented dataset.

5.2.2 BERT with CAPTUM

CAPTUM 's Layer Conductance and Layer Integrated Gradients was used to assist us in understanding our BERT models by enchancing the interpretability of neural networks. Layer Conductance quantifies the impact of input feature changes on specific layer activations within the neural network, providing a deeper understanding of feature importance and their influence on model output. Conversely, Layer Integrated Gradients analyzes the contributions of individual layers to model predictions, illuminating the significance of each layer for specific tasks and revealing where the model focuses its attention. This approach allowed us to visualize a typical forward pass, elucidating how individual inputs affect the model's outputs and providing invaluable insights into the model's decision-making process.

True Label	Reliable
Predicted Label	Reliable (1.00)
Attribution Label	Results Sunday from the Japan Open, a $1.2 million ATP event on hard courts at Ariake Colosseum (seedings in parentheses): Singles Final Jo-Wilfried Tsonga (2), France, def. Mikhail Youzhny, Russia, 6-3, 6-3. Doubles Final Julian Knowle and Jurgen Melzer, Austria, def. Ross Hutchins, Britain, and Jordan Kerr, Australia, 6-2, 5-7, 10-8.
Attribution Score	2.27
Word Importance	results sunday from the japan open, a $ 1 . 2 million atp event on hard courts at ariake colosseum (seedings in parentheses): singles final jo-wilfried tsonga (2), france, def. mikhail youzhny, russia, 6-3, 6-3. doubles final julian knowle and jurgen melzer, austria, def. ross hutchins, britain, and jordan kerr, australia, 6-2, 5-7, 10-8.

Table 11: Replication of the Layer Integrated Gradients Example 1. **This tell us that the green tokens supports the predictions for Reliable while the red tokens does not support the predictions for Reliable**

Green color represents the positive attribution score (how much impact did this word have on predicting the output class as hoax with regards to the BERT embedding layer only). Red color refers to negative attribution score along similar lines. We analyzed this example from the test dataset where the original query belongs to "Reliable News". From this, one might expect that the class is reliable because of the usage specific details such as scores, player names, and the prize money. Therefore, a good model would have given high attribution scores to those aspects of the query. We can see that some of the player names and the details are given positive attributions which again is expected from our trained BERT models due to their average performance.

In Figure 13, lighter shades refers to more positive attributions while darker shades refers to more negative attributions. We can see that the comma after the phrase "results sunday from the japan open" seems to have a high positive attribution in predicting this class as reliable. We suspect that comma was used across all attention layers in BERT when it is training for the Reliable News class.

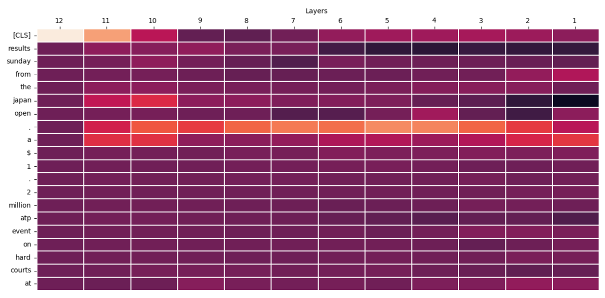

Figure 8: Layer Conductance Example 1 Snippet. **The figure shows the attention's activation in BERT across all layers. BERT was activated by the Comma token across multiple layers which tells us that Comma is important in Reliable News classification**.

5.3 LDA Analysis of Model Prediction Performance

After training the models and obtaining predictions, the results were analyzed from LDA topic modelling's perspective. The tables of these experiments are attached in the appendix [7]. The key observations have been discussed in this section.

Topic-2 Economic Discussions had only 5 rows in total in test set. So if just the most misclassified category was picked, it might not portray the actual bigger picture, hence all topics were ranked from most misclassified to least misclassified. From the topic-wise accuracy and ranking metrics of the model, it can inferred that Linear SVC does well with topic-3 which is Health and Medical related topics and it consistently does well with class-1 articles i.e Satire. The feed-forward Neural Network does well on both topic-2 and topic-3, where topic-2 is Economic Discussions. BERT on the other hand struggles with topic-3 while it does generally well on topic-1 which is Politics and Government Affairs. It is interesting to observe that both machine models - Linear SVC misclassified topic-2 articles the most, while on the other hand both deep learning models - Basic NN and BERT did well on topic-2 articles belonging to classes 'propaganda' and 'reliable' news. It is clear that all models are not able to equally well across multiple topics. Linear SVC classifies topics such as Health really well since the main predictors in Health related sentences are scientific and medical jargons while it fails on topics such as Economic Discussions, which require the model to provide more attention to the semantic context to distinguish between classes - which BERT was able to do well.

6 Conclusion

The identification of fake news is a challenging task as evident from existing research work and our exploratory project. Feed-forward Neural Networks fail to capture the nuances in the sentences that push the model towards or away from a particular class. While BERT is able to do a better job, the results still lean towards our initial thought-process that just text data would not suffice the task of identifying fake news. Tonal intensities, usage of sarcasm, external knowledge in the form of article topic, author, subject of the article are hard to infer from just text data. From LDA, it was clear that different algorithms perform better on some topics, while terribly on others. Since stand-alone single model approaches are not that effective, in order to improve model performance, based on the topic of the article, a multi-model ensemble setup should be incorporated. Our group was not able to improve on the performance by Rashkin et. al, but we did manage to produce different models with similar F1 scores. This result suggests that the LUN dataset is a difficult one to train on regardless of the models used. In addition, our efforts to augment the dataset together with our own feature engineering failed to produce a better dataset to train on. Ultimately, we were able to validate the initial assumptions about the dataset's behaviour.

References

Annika Bergström and Maria Jervelycke Belfrage. 2018. News in social media. *Digital Journalism*, 6(5):583–598.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3(null):993–1022.

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*.

Eduardo C. Garrido-Merchan, Roberto Gozalo-Brizuela, and Santiago Gonzalez-Carvajal. 2023. Comparing bert against traditional machine learning models in text classification. *Journal of Computational and Cognitive Engineering*, 2(4):352–356.

Eric Holgate, Isabel Cachola, Daniel Preoţiuc-Pietro, and Junyi Jessy Li. 2018. Why swear? analyzing and inferring the intentions of vulgar expressions. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, page 4405–4414.

C. Hutto and Eric Gilbert. 2014. Vader: A parsimonious rule-based model for sentiment analysis of social media text. *Proceedings of the International AAAI Conference on Web and Social Media*, 8(1):216–225.

Yenwee Lim. 2022. State-of-the-art machine learning hyperparameter optimization with optuna.

Hannah Rashkin, Eunsol Choi, Jin Yea Jang, Svitlana Volkova, and Yejin Choi. 2017. Truth of varying shades: Analyzing language in fake news and political fact-checking. *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*.

Lijuan Sun and Hongbin Wang. 2023. Topic-aware fake news detection based on heterogeneous graph. *IEEE Access*, 11:103743–103752.

Vladimir N. Vapnik. 1995. *The nature of statistical learning theory*.

Wenbin Zhao, Peisong He, Zhixin Zeng, and Xiong Xu. 2024. Fake news detection based on knowledge-guided semantic analysis. *Electronics*, 13(2):259.

7 Appendix

LDA Experiments The below numbers indicate how well each model classified the topics, i.e accuracy % for articles of each topic. Each cell indicates classification accuracy for articles belonging to topic(row) and class(column), hence each cell is a standalone value and the row/col need not sum up to one. Since XGBoost's performance was similar Linear SVC, XGBoost was not taken into account for analysis in report.

Note: The takeaway for the following tables are consolidated together after the analysis of accuracy ranks since the tables have to be looked at collectively.

Table 12: Linear SVC LDA Analysis (T indicates Topic)

Label	**T-1**	**T-2**	**T-3**	**T-4**	**T-5**
1	0.77	0.86	0.00	0.72	0.79
2	0.48	0.33	0.50	0.80	0.55
3	0.83	0.94	0.00	0.91	0.79
4	0.94	0.82	1.00	1.00	0.90

Table 13: Feed Forward Neural Network LDA Analysis (T indicates Topic)

Label	**T-1**	**T-2**	**T-3**	**T-4**	**T-5**
1	0.31	0.29	0.00	0.45	0.29
2	0.03	0.22	0.00	0.27	0.06
3	0.99	0.94	0.00	0.97	0.98
4	0.35	0.82	1.00	0.85	0.50

Table 14: BERT LDA Analysis (T indicates Topic)

Label	**T-0**	**T-1**	**T-2**	**T-3**	**T-4**
1	0.70	0.70	0.55	0.68	0.65
2	0.76	0.88	0.67	0.73	0.80
3	0.03	0.00	0.00	0.00	0.05
4	0.95	0.89	1.00	0.94	0.95