ADVANCED NATURAL LANGUAGE PROCESSING

NAME: HARIS ROY PULINILKUNNATHIL

CANDIDATE NUMBER: 261608

1. Introduction

Propaganda, a strategic tool often deployed in media, politics and education sectors, works by influencing public attitudes and actions subtly yet substantially. The study of propaganda becomes crucial and difficult because its effects can significantly change society's attitudes and public conversation. This report explores two different but related problems related to the computational analysis of propaganda in texts.

Designing and evaluating techniques to determine whether a text contains propaganda is the first task, inspired by the growing concern regarding the spread and impact of propaganda in the digital era. The complexity of propaganda makes this endeavour fundamentally tricky. Propaganda techniques frequently use emotional language, exaggerated claims, and appeals to authority to mislead the audience, making their detection difficult. This intricacy is further increased by the wide range of propaganda kinds, demanding practical approaches to distinguish between propaganda and non-propagandistic literature.

The second task, which builds on the first, aims to classify the propaganda technique used within a known propagandistic text fragment or passage. This challenge emphasises the significance of comprehending the propagandist's tactical decisions and aims. It shifts the focus from the entire sentence to specific annotated text snippets marked by the tokens <BOS> (beginning of span) and <EOS> (end of span). These tokens help pinpoint the text's actual segment embodying the propaganda technique, guiding our attention to the core propagandistic elements within the broader sentence context.

Unravelling these tasks' complexities requires exploring advanced approaches in computational linguistics and machine learning. This report will investigate various methods, such as n-gram language models and uncontextualised word embeddings(word2vec), neural language models (LSTM) and pre-trained large language models (BERT).

2. Dataset

The dataset supplied for this project is organised into two distinct tab-separated value (tsv) files intended for model training and testing. Each file has two key columns: 'label' and 'sentence(tagged_in_context)'. The 'label' column is populated with one of nine potential value subsets of the tactics identified in the Propaganda Techniques Corpus. Eight labels represent different types of propaganda, while the ninth label signifies that the text does not contain any propaganda.

The 'sentence(tagged_in_context)' column provides sentences or text fragments that display the corresponding propaganda technique indicated by the label. Each sentence is marked with <BOS> and <EOS> tokens, signifying the commencement and conclusion of the annotated section within the text where the propaganda technique is evident. The primary purpose of this dataset is to facilitate the development and assessment of models capable of recognising various propaganda techniques within text content.

3. Methodology

3.1 Task 1

The primary objective of Task 1 in this study is to discern the presence of propaganda within a given text. Because of this, we approach this task as a binary classification problem.

3.1.1 – Approach 1: Multinomial Naive Bayes classifier (unigrams, bigrams, and trigrams)

In approach 1, we intend to employ a probability-based approach using n-gram models in conjunction with a Multinomial Naive Bayes (MNB) classifier. The simplicity, computational effectiveness, and aptitude for handling high-dimensional and sizable datasets of the MNB classifier feature that are frequently crucial in text classification tasks -are the reasons behind the selection of this classifier.

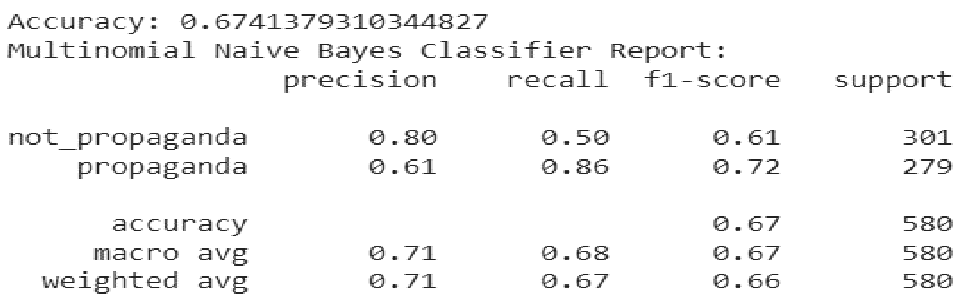
This task-specific methodology comprises several sequential steps, including Data Preparation, Text Vectorization, Model Training, Evaluation, Hyperparameter Tuning, and Result Analysis.

During the Data Preparation phase, the dataset is organised into two separate tab-separated value files for training and testing the MNB model. Each file includes two key columns: 'label' and 'processed_text'. The 'label' column serves as the binary target variable, indicating the presence or absence of propaganda, while the 'processed_text' column holds the corresponding pre-processed text data.

The Text Vectorization phase transforms the text data into numerical feature vectors using TfidfVectorizer. It involves converting training and testing data into vector representations with term frequency-inverse document frequency (TF-IDF) weighting. The ngram_range parameter, set to capture unigrams, bigrams, and trigrams, allows the model to consider different word sequence lengths in the text data.

The Model Training phase involves training the MNB model on the vectorised data. Here, the smoothing parameter alpha is adjusted to control the model's approach to terms not found in the learning samples, thus mitigating the impact of unseen features on model predictions.

Post-training, the model is evaluated. The trained model predicts labels for the testing data, and its performance is assessed via the accuracy metric. This evaluation process is conducted iteratively across different alpha and n-gram range values, systematically exploring the impact of these parameters on performance.



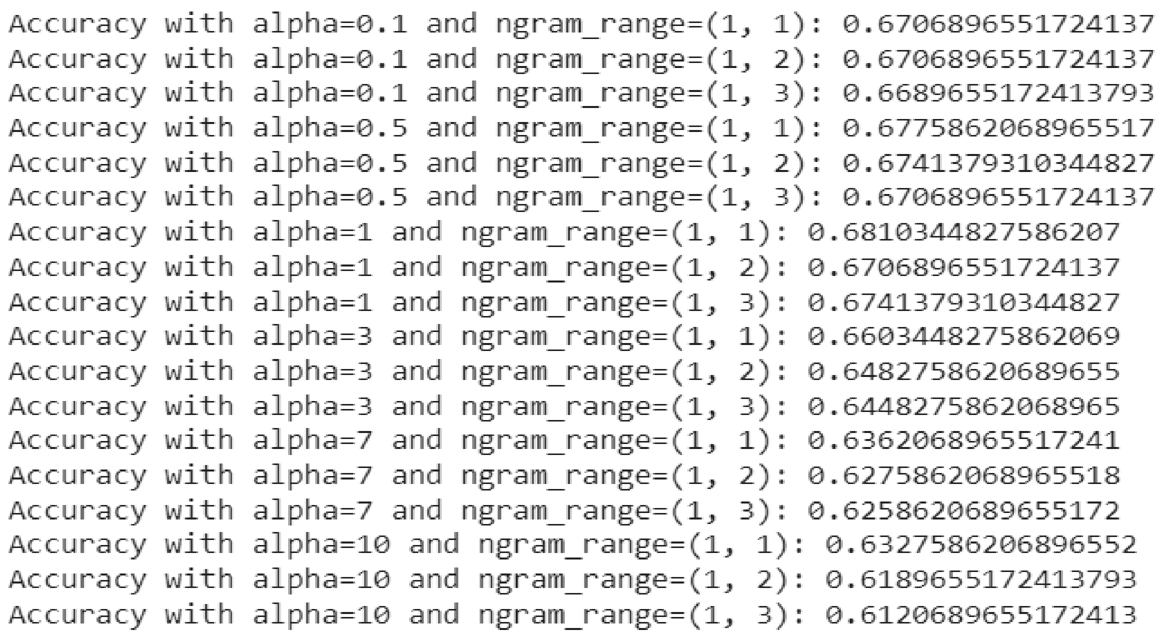
```
Accuracy: 0.6741379310344827
Multinomial Naive Bayes Classifier Report:
                precision    recall  f1-score   support

not_propaganda       0.80      0.50      0.61       301
    propaganda       0.61      0.86      0.72       279

      accuracy                           0.67       580
     macro avg       0.71      0.68      0.67       580
  weighted avg       0.71      0.67      0.66       580
```

Fig 1: Classification report of MNB Classifier

As the classification report shown in the fig1; an overall accuracy of 67.41%, denoting correct predictions made by the model. Breaking it down by classes, for 'not_propaganda', the model demonstrated a high precision of 0.80 but a lower recall of 0.50, leading to an F1-score of 0.61. On the contrary, 'propaganda' had lower precision (0.61) but higher recall (0.86), yielding an F1-score of 0.72; this showcases the model's stronger recall for detecting propaganda over its precision.

According to error analysis, the model frequently misclassifies instances of "not_propaganda" as "propaganda" (producing more false positives for "propaganda") and ignores genuine "not_propaganda" cases (producing more false negatives for "not_propaganda"). These results point out the model's flaws and indicate additional research into the instances incorrectly classified to identify the underlying patterns or traits that caused these mistakes.

For Hyperparameter Tuning, a grid search is conducted across a predefined range of alpha values and n-gram ranges to find the optimal parameter combination for the best performance.



```
Accuracy with alpha=0.1 and ngram_range=(1, 1): 0.6706896551724137
Accuracy with alpha=0.1 and ngram_range=(1, 2): 0.6706896551724137
Accuracy with alpha=0.1 and ngram_range=(1, 3): 0.6689655172413793
Accuracy with alpha=0.5 and ngram_range=(1, 1): 0.6775862068965517
Accuracy with alpha=0.5 and ngram_range=(1, 2): 0.6741379310344827
Accuracy with alpha=0.5 and ngram_range=(1, 3): 0.6706896551724137
Accuracy with alpha=1 and ngram_range=(1, 1): 0.6810344827586207
Accuracy with alpha=1 and ngram_range=(1, 2): 0.6706896551724137
Accuracy with alpha=1 and ngram_range=(1, 3): 0.6741379310344827
Accuracy with alpha=3 and ngram_range=(1, 1): 0.6603448275862069
Accuracy with alpha=3 and ngram_range=(1, 2): 0.6482758620689655
Accuracy with alpha=3 and ngram_range=(1, 3): 0.6448275862068965
Accuracy with alpha=7 and ngram_range=(1, 1): 0.6362068965517241
Accuracy with alpha=7 and ngram_range=(1, 2): 0.6275862068965518
Accuracy with alpha=7 and ngram_range=(1, 3): 0.6258620689655172
Accuracy with alpha=10 and ngram_range=(1, 1): 0.6327586206896552
Accuracy with alpha=10 and ngram_range=(1, 2): 0.6189655172413793
Accuracy with alpha=10 and ngram_range=(1, 3): 0.6120689655172413
```

Fig 2: Accuracy scores for different combinations of alpha and n-gram range values for MNB Classifier.

From the results of hyperparameter tuning as in Fig 2, various alpha values (0.1,0.5,1,3,7,10) were tested alongside different n-gram ranges (1,1)-unigrams, (1,2)-bigrams, (1,3)-trigrams). The performance, measured by accuracy, was recorded for each combination of parameters. Results reveal that the highest accuracy, 68.10, was achieved with alpha set at 1 and n-gram range as (1,1). The accuracy gradually declined as the alpha values increased, indicating an over-smoothing effect.

3.1.2 Approach 2: Uncontextualised Word Embeddings (Word2Vec) and Logistic Regression

The second approach employs a unique combination of uncontextualised word embeddings and a statistical binary classification model. Word embeddings, particularly Word2vec, offer a dense vector representation that effectively captures semantic information and relationship between words. Specifically, Word2vec(word2vec-google-news-300), a group of shallow, two-layer neural network models, was employed to generate these embeddings, while the classification task was performed using Logistic Regression (LR) due to its robustness in handling high-dimensional data and probabilistic outcomes, offering not just class predictions but also the confidence associated with those predictions.

During the initial phase of pre-processing with word embeddings, the pre-trained Google News Word2Vec model was used. The pre-processed text was subsequently transformed into vector representation by calculating the mean of the Word2Vec vectors corresponding to each constituent word. A classification report was generated to evaluate the model, presenting essential metrics such as accuracy, precision, recall and the F1 score. The results of this classification process are shown in Figure 3.

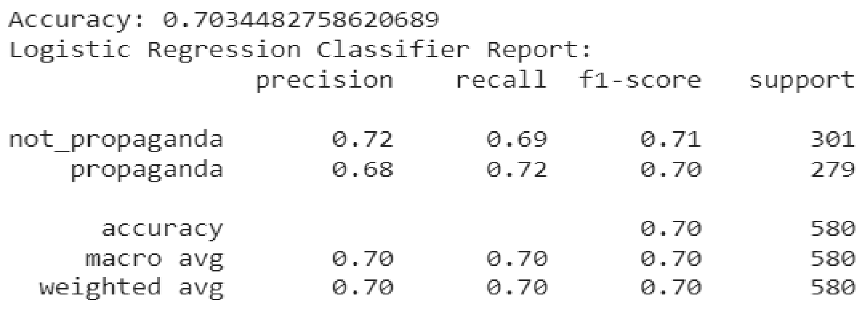
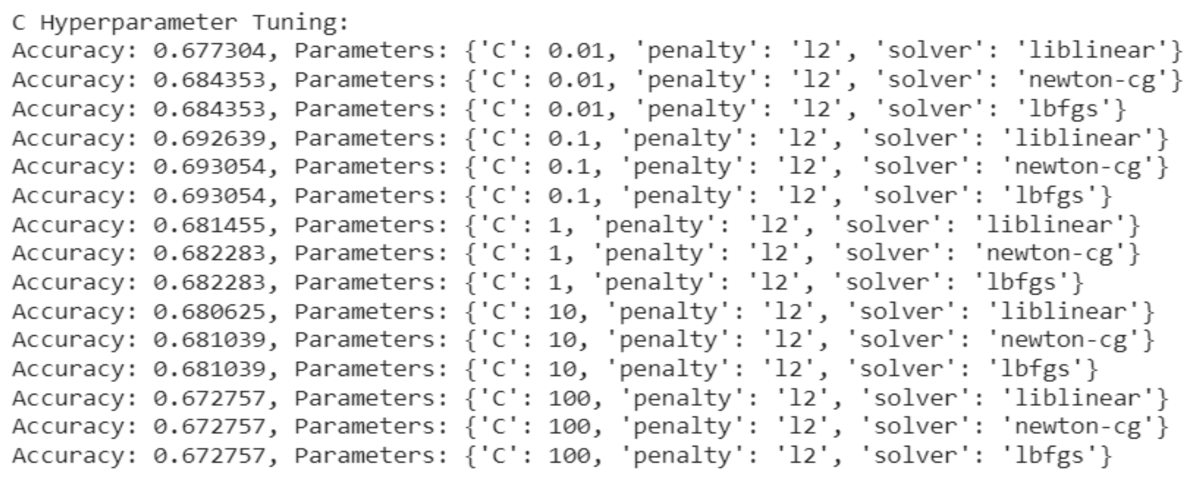
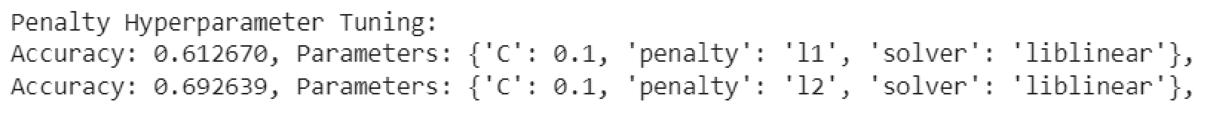

```
Accuracy: 0.7034482758620689
Logistic Regression Classifier Report:
                precision    recall  f1-score   support

not_propaganda       0.72      0.69      0.71       301
    propaganda       0.68      0.72      0.70       279

      accuracy                           0.70       580
     macro avg       0.70      0.70      0.70       580
  weighted avg       0.70      0.70      0.70       580
```

Fig 3: Classification report of Logistic Regression model with word2vec embeddings

As indicated in Figure 3, the LR model achieves an overall accuracy of roughly 70.34% on the test set. Precision, recall, and F1-scores for the 'propaganda' class were calculated as 0.68, 0.72, and 0.70, respectively, and for the 'non-propaganda' class, they were 0.72, 0.69, and 0.71, respectively. This value suggests a slightly higher accuracy of the model in classifying 'non-propaganda' instances, although the model exhibits relatively balanced performance across both categories. Following the initial evaluation of the model's performance, the subsequent phase involved fine-tuning its hyperparameters to optimise its functioning. The outcomes of hyperparameter adjustment for the LR model are shown in Figure 4.



```
C Hyperparameter Tuning:
Accuracy: 0.677304, Parameters: {'C': 0.01, 'penalty': 'l2', 'solver': 'liblinear'}
Accuracy: 0.684353, Parameters: {'C': 0.01, 'penalty': 'l2', 'solver': 'newton-cg'}
Accuracy: 0.684353, Parameters: {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
Accuracy: 0.692639, Parameters: {'C': 0.1, 'penalty': 'l2', 'solver': 'liblinear'}
Accuracy: 0.693054, Parameters: {'C': 0.1, 'penalty': 'l2', 'solver': 'newton-cg'}
Accuracy: 0.693054, Parameters: {'C': 0.1, 'penalty': 'l2', 'solver': 'lbfgs'}
Accuracy: 0.681455, Parameters: {'C': 1, 'penalty': 'l2', 'solver': 'liblinear'}
Accuracy: 0.682283, Parameters: {'C': 1, 'penalty': 'l2', 'solver': 'newton-cg'}
Accuracy: 0.682283, Parameters: {'C': 1, 'penalty': 'l2', 'solver': 'lbfgs'}
Accuracy: 0.680625, Parameters: {'C': 10, 'penalty': 'l2', 'solver': 'liblinear'}
Accuracy: 0.681039, Parameters: {'C': 10, 'penalty': 'l2', 'solver': 'newton-cg'}
Accuracy: 0.681039, Parameters: {'C': 10, 'penalty': 'l2', 'solver': 'lbfgs'}
Accuracy: 0.672757, Parameters: {'C': 100, 'penalty': 'l2', 'solver': 'liblinear'}
Accuracy: 0.672757, Parameters: {'C': 100, 'penalty': 'l2', 'solver': 'newton-cg'}
Accuracy: 0.672757, Parameters: {'C': 100, 'penalty': 'l2', 'solver': 'lbfgs'}
```



```
Penalty Hyperparameter Tuning:
Accuracy: 0.612670, Parameters: {'C': 0.1, 'penalty': 'l1', 'solver': 'liblinear'},
Accuracy: 0.692639, Parameters: {'C': 0.1, 'penalty': 'l2', 'solver': 'liblinear'},
```

Fig 4: Hyperparameter Tuning results of LR model with parameters C, penalty and solver

As illustrated in Figure 4, the LR model exhibits optimal performance when the ‘C’ hyperparameter is set to 0.1 with ‘l2’ penalty and ‘newton-cg’ and ‘lbfgs’ solver, yielding an accuracy of 0.6930. Any deviation from this optimal ‘C’ value resulted in a decrement in model performance, thereby highlighting that a ‘C’ value of 0.1 is most suitable for this specific task.

Despite these results, there needs to be a thorough error analysis that might have shown the model's possible flaws and areas needing development. Further iterations could profit from thoroughly investigating the underperforming inputs or categories to improve the model's performance.

3.1.3 Comparison

No	Models	Accuracy	Precision	Recall	f1 score
1	MNB Classifier	67.41%	71%	68%	67%
2	Logistic Regression with word2vec	70.34%	70%	70%	70%

The first and second approaches - the MNB model utilising n-grams, and the LR model paired with Word2Vec embeddings, respectively - provide substantial solutions for detecting propaganda. However, when we compare their overall performance, the second approach gains a slight upper hand with an accuracy score of around 70.34%, which stands against the first approach's score of 67.41%. The second approach also showcases a well-balanced precision and recall for propaganda and non-propaganda categories. In contrast, the first approach has high precision but comparatively lower recall for the 'non-propaganda' category. When we delve into the specifics of hyperparameter tuning, it is intriguing to find that both models establish their unique optimum parameters. The first approach exhibits computational efficiency and simplicity. In contrast, despite requiring more computational resources, the second approach captures the more nuanced and intricate patterns in the data, resulting in a better overall performance. Given that both models possess unique tendencies for errors, the selection between the two would need to be informed by the specifics of the task.

3.2 Task- 2

Task 2 identifies and classifies the propaganda technique used within an annotated text fragment, signified by the tokens <BOS> and <EOS>. Fundamentally, this is a multi-class text classification problem, where the goal is to assign one out of several potential propaganda technique labels to each text segment.

3.2.1 Approach 1 – Neural Networks (GRU Ensemble)

For multi-class text categorisation in approach 1, we implemented an ensemble of GRU (Gated Recurrent Unit) neural network models, which enhance efficiency, decrease overfitting, and boost robustness; this method mixes numerous GRU models. The ensemble use of the model variety captures many patterns in the text data and makes more accurate predictions. By combining the advantages of separate models, the ensemble provides a comprehensive and accurate solution.

Model 1: This model consisted of an Embedding layer followed by two GRU layers with 128 and 64 units, respectively. The first GRU layer returned sequences for the second layer. A dropout layer with a rate of 0.5 was added to prevent overfitting, and a Dense layer with 64 units and a ReLU activation function was included. The output Dense layer used SoftMax activation for multi-class classification.

Model 2: Like Model 1, this configuration included an Embedding layer followed by two GRU layers with 64 and 32 units. The first GRU layer also returned sequences for the second layer. A dropout layer with a rate of 0.4 was introduced to control overfitting, and a Dense layer with 32 units and ReLU activation was used before the output Dense layer with SoftMax activation.

Model 3: This configuration featured an Embedding layer followed by two GRU layers with 256 and 128 units. The first GRU layer returned sequences for the second layer. A dropout layer with a rate of 0.6 was applied to mitigate overfitting. A Dense layer with 128 units and a ReLU activation function was added before the output Dense layer with SoftMax activation.

The models were trained using categorical cross-entropy loss and the Adam optimiser. Early stopping was implemented during training to prevent overfitting.

The classification report below offers a detailed assessment of the models' performance, presenting precision, recall, and F1-score for each propaganda technique class, alongside overall accuracy. This report provides valuable insights into the effectiveness of our chosen approaches for identifying propaganda techniques and their capability to classify specific propaganda techniques within given text snippets accurately.

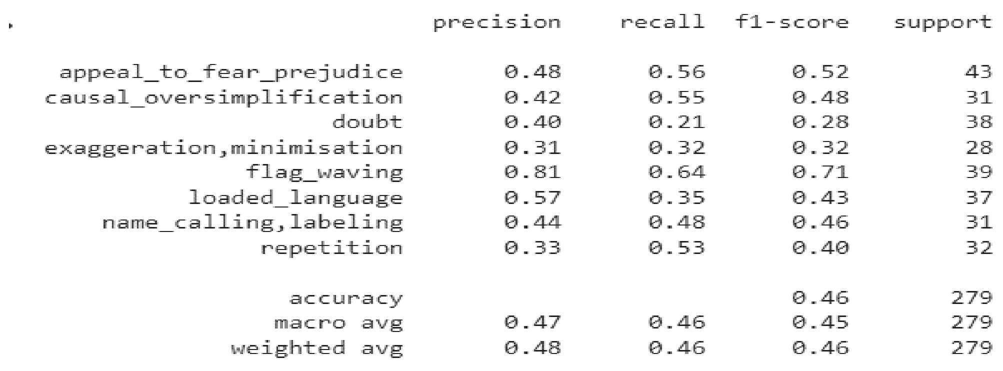

```
                              precision    recall  f1-score   support

   appeal_to_fear_prejudice       0.48      0.56      0.52        43
  causal_oversimplification       0.42      0.55      0.48        31
                      doubt       0.40      0.21      0.28        38
  exaggeration,minimisation       0.31      0.32      0.32        28
               flag_waving       0.81      0.64      0.71        39
            loaded_language       0.57      0.35      0.43        37
      name_calling,labeling       0.44      0.48      0.46        31
                 repetition       0.33      0.53      0.40        32

                   accuracy                           0.46       279
                  macro avg       0.47      0.46      0.45       279
               weighted avg       0.48      0.46      0.46       279
```

Fig 5: Classification report for the GRU ensemble

The precision, representing the accuracy of positive predictions, varies across classes, with the highest at 0.81 for "flag_waving" recall measuring the ability to identify instances correctly ranging from 0.21 for "doubt" to 0.64 for "flag_waving." The f1-score, a balanced measure, spans from 0.28 for "doubt" to 0.71 for "flag_waving." Overall accuracy is 0.46, indicating correct predictions for 46% of the sentences. The macro avg and weighted avg give average performance across classes. The report helps understand model strengths and areas for improvement in identifying propaganda techniques.

The error analysis reveals challenges in correctly classifying certain propaganda technique classes, particularly in the "doubt" and "exaggeration,minimisation" categories. These classes exhibit lower precision, recall, and f1-scores, suggesting difficulty accurately predicting instances. The errors may stem from the complexity and ambiguity of language in different propaganda instances. Due to the large parameter space and limited resources, we encountered computational challenges during hyperparameter tuning using GridSearchCV.

In conclusion, our text classification model, employing an ensemble of GRU (Gated Recurrent Unit) neural networks, exhibited promising results in detecting propaganda techniques within given texts. The model's architecture, consisting of multiple GRU layers, dropout regularisation, and dense layers with ReLU activation, significantly contributed to its effectiveness. Although we encountered computational challenges during hyperparameter tuning, we optimised the model's performance using a limited combination approach.

3.2.2 Approach 2: Pretrained Large Language Model (BERT)

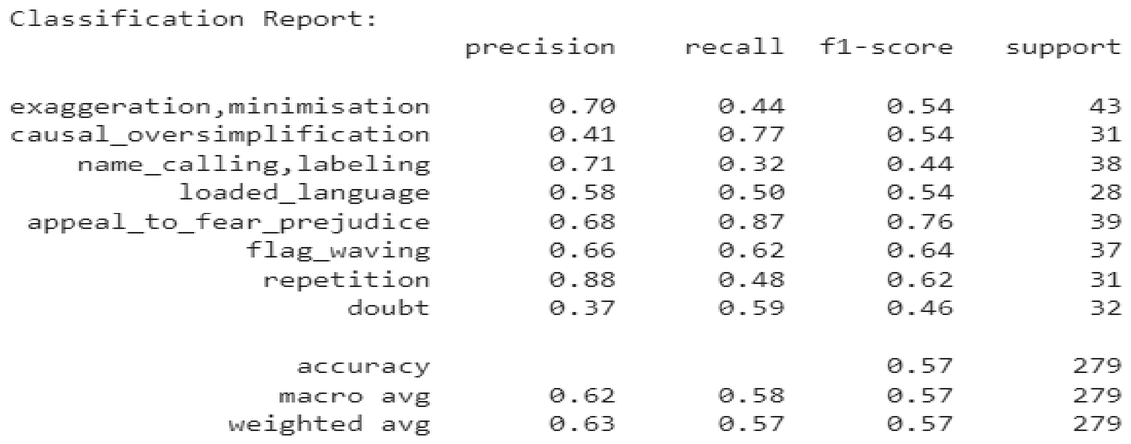
By using BERT (Bidirectional Encoder Representations from Transformers) for multi-class text categorisation, we could recognise the methods used in this method. BERT is a powerful language model that captures contextualised word representations and has produced encouraging outcomes in several natural language processing tasks.

We first installed the transformers library and imported the necessary modules. Next, we loaded the pre-trained BERT tokeniser and model (bert-base-cased) using the AutoTokenizer and TFBertModel from transformers, respectively. The text data was tokenised, and the BERT embeddings were used as input for the neural network.

A neural network architecture was incorporated to make the BERT model more suitable for our propaganda technique identification task. The model comprised dense layers, a global max-pooling layer, and an input layer for BERT embeddings. The output layer employed a sigmoid activation function to conduct multi-class classification.

The maximum length of input sequences (max_len=70), the learning rate schedule, and early halting to avoid overfitting are some of the hyperparameters examined in this methodology. After every 100000 decay steps, the learning rate schedule (ExponentialDecay) degraded at a rate of 0.96 from its starting value of 5e-5. With the categorical cross-entropy loss function, we utilised an Adam optimiser. The model was trained for 15 epochs with a batch size of 36.

The evaluation was performed using a classification report, which provided precision, recall, and F1-score for each propaganda technique class and overall accuracy.



```
Classification Report:
                            precision    recall  f1-score   support

exaggeration,minimisation       0.70      0.44      0.54        43
causal_oversimplification       0.41      0.77      0.54        31
    name_calling,labeling       0.71      0.32      0.44        38
          loaded_language       0.58      0.50      0.54        28
  appeal_to_fear_prejudice      0.68      0.87      0.76        39
              flag_waving       0.66      0.62      0.64        37
               repetition       0.88      0.48      0.62        31
                    doubt       0.37      0.59      0.46        32

                 accuracy                           0.57       279
                macro avg       0.62      0.58      0.57       279
             weighted avg       0.63      0.57      0.57       279
```

Fig 6: Classification report for the BERT

The classification report reveals the model’s performance in different propaganda technique classes. The class “repetition” achieved the highest precision of 0.88, while the class “doubt” had the lowest precision of 0.37. On the other hand, the class “appeal_to_fear_prejudice" achieved the highest recall of 0.87, and the class” name_calling,labeling” had the lowest recall of 0.32. The F1-score ranges from 0.44 for the class “name_calling,labeling” to 0.76 for the class "appeal_to_fear_prejudice." The model's overall accuracy was 0.57, with a macro-average F1-score of 0.57, indicating varying performance across different propaganda technique classes. The error analysis highlighted difficulties in classifying "doubt" and "exaggeration,minimisation" propaganda techniques, possibly due to language complexity and ambiguity. Overall, the BERT-based model showed promising results in detecting propaganda techniques within texts. Combining BERT's contextualised embeddings and neural network architecture effectively achieved accurate classification.

To improve the model's performance, further work could explore hyperparameter tuning with a more comprehensive search strategy, such as Bayesian optimisation, to address computational challenges.

3.2.3 Comparison

No	Model	Accuracy	Precision	Recall	f1 score
1	GRU Ensemble	46%	47%	46%	45%
2	BERT	57%	62%	58%	57%

In comparing the two approaches, both utilise deep learning methodologies to tackle the challenge of propaganda technique identification in text. The first approach adopts an ensemble of Gated Recurrent Unit (GRU) neural networks, while the second approach leverages the Bidirectional Encoder Representations from Transformers (BERT) model.

Regarding model complexity, BERT is more computationally intensive given its vast number of parameters and the depth of the architecture, which also translates to potentially higher accuracy given the proper training and fine-tuning procedures. On the other hand, while less complex, the GRU ensemble provides robustness through model diversity and might be more computationally efficient.

Comparing the results, the BERT model has achieved an overall accuracy of 0.57, which is superior to the 0.46 achieved by the GRU ensemble. Regarding class-specific performance, the BERT model also appears to have higher precision, recall, and f1-scores across most classes. However, both models have shown difficulties correctly classifying specific propaganda techniques, such as "doubt" and "exaggeration, minimisation."

Despite the better performance of BERT, both models have their merits. The GRU ensemble may be more suitable for constrained computational resources. In contrast, despite its computational demands, the BERT model provides a more effective solution given sufficient resources and proper tuning.

On another note, both models showed significant improvements when early stopping and dropout were implemented to avoid overfitting, indicating that these techniques are beneficial in training deep-learning models.

4. Conclusion

In conclusion, this report explores the computational analysis of propaganda in texts through two tasks: detecting propaganda presence and classifying propaganda techniques. Two approaches were used for each task. For Task 1, Approach 2, employing Word2Vec and Logistic Regression, outperformed Approach 1, achieving an accuracy of 70.34%. For Task 2, BERT demonstrated superior performance with an accuracy of 0.57, outperforming the GRU ensemble. Both models benefited from early stopping and dropout to prevent overfitting. Overall, the study emphasises the significance of complex models like BERT being accurate in propaganda detection and classification. Computational linguistics and machine learning are crucial in understanding and countering propaganda's impact. Further research and development in this field hold the potential to refine and enhance these models, enabling us to address propaganda's influence and promote informed decision-making.

5. Future Work

The computational study of propaganda in texts can be improved in the future by investigating more complex model designs outside of BERT and transformers. Bayesian optimisation can be used to tune hyperparameters, improving model performance systematically. Pre-trained models can be fine-tuned using data unique to propaganda, increasing classification precision and the models' comprehension of propaganda-related patterns.

REFERENCES

1. Analytics Vidhya. (2021). Multiclass Classification Using Transformers for Beginners. Available at: https://www.analyticsvidhya.com/blog/2021/12/multiclass-classification-using-transformers/ [Accessed 3 August 2023].

2. Brownlee, J. (2018). How to Develop a Stacking Ensemble for Deep Learning Neural Networks in Python With Keras. Available at: https://machinelearningmastery.com/stacking-ensemble-for-deep-learning-neural-networks/ [Accessed 3 August 2023].

3. Galarnyk, M. (2020). Logistic Regression using Python (scikit-learn). Available at: https://towardsdatascience.com/logistic-regression-using-python-sklearn-numpy-mnist-handwriting-recognition-matplotlib-a6b31e2b166a [Accessed 3 August 2023].

4. Ahmed, J. and Ahmed, M. (2023). Classification, detection and sentiment analysis using machine learning over next generation communication platforms. Microprocessors and Microsystems, 98, p.104795. Available at: https://doi.org/10.1016/j.micpro.2023.104795 [Accessed 1 August 2023].

5. Grzywalski, T., Piecuch, M., Szajek, M., Bręborowicz, A., Hafke-Dys, H., Kociński, J., Pastusiak, A. and Belluzzo, R. (2019). Practical implementation of artificial intelligence algorithms in pulmonary auscultation examination. European Journal of Pediatrics, 178(6), pp.883–890. Available at: https://doi.org/10.1007/s00431-019-03363-2 [Accessed 3 August 2023].

6. Lim, X.R., Lee, C.P., Lim, K.M. and Ong, T.S. (2023). Enhanced Traffic Sign Recognition with Ensemble Learning. Journal of Sensor and Actuator Networks, 12(2), p.33. Available at: https://doi.org/10.3390/jsan12020033 [Accessed 3 August 2023].

7. IJRASET (n.d.). Threat Prediction using Honeypot and Machine Learning. Available at: https://www.ijraset.com/research-paper/threat-prediction-using-honeypot-and-ml [Accessed 25 July 2023].

8. Machine Learning and Metaheuristics Algorithms, and Applications. (2021). Communications in computer and information science. Springer Science+Business Media. Available at: https://doi.org/10.1007/978-981-16-0419-5 [Accessed 3 August 2023].

9. Tucker, A., Vinciotti, V., Liu, X. and Garway-Heath, D. (2005). A spatio-temporal Bayesian network classifier for understanding visual field deterioration. Artificial Intelligence in Medicine, 34(2), pp.163–177. Available at: https://doi.org/10.1016/j.artmed.2004.07.004 [Accessed 3 August 2023].

10. Scikit-learn.org. (2019). sklearn.naive_bayes.MultinomialNB — scikit-learn 0.22 documentation. Available at: **https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html** [Accessed 3 August 2023].

ADVANCEMENTS IN MACHINE TRANSLATION: A COMPARATIVE ANALYSIS OF NEURAL AND HYBRID APPROACHES

Abstract

Machine Translation (MT) is a sophisticated technique to translate text from one language to another. Machine translation is essential for removing language barriers and promoting intercultural dialogue. This essay explores two prominent approaches to machine translation: Neural Machine Translation (NMT) and Hybrid Machine Translation (HMT). For fluency, MT uses neural networks, whereas, for accuracy, HMT combines rule-based and neural components. The outcomes demonstrate HMT's superiority and the state-of-the-art performance of NMT.

1. Introduction

Machine Translation (MT) represents the automated text translation between languages, a complex task driven by intricate algorithms. The core objective of MT is to deliver accurate translations that mirror the target language's natural flow. However, this is far from straightforward.

Languages are inherently complex, each with idioms, grammar, and syntax, which adds to the difficulty. For instance, when directly translated into another language, idioms like "break a leg" in English lose their intended meaning. Additionally, accurate real-time translation necessitates thoroughly comprehending the words, cultural context, and nuances at play.

Considering these challenges, two promising approaches have emerged in the field of MT: Neural Machine Translation (NMT) and Hybrid Machine Translation (HMT). These approaches use various methodologies, which will be examined and compared throughout this essay to address the complex nature of translation.

2. Description of Approaches

2.1 Neural Machine Translation (NMT)

Neural Machine Translation (NMT) is a modern machine translation method that has attained state-of-the-art in recent years. Utilising artificial neural networks to learn language mapping is the fundamental principle behind NMT. We will discuss the issues with machine translation in this essay, review the NMT method then assess how well it works considering the existing research. Due to the complexity and diversity of human languages, this is a complex problem to encounter. Traditional methods, such as rule-based and statistical machine translation, have trouble dealing with colloquial expressions and complicated linguistic structures (Kohen,2017). To learn the mapping across languages, researchers have turned to neural networks.

In NMT systems, the encoder reads the source sentence, turning it into a series of hidden states. The target sentence is then generated by the decoder using the hidden states. The

NMT's basic encoder-decoder is distinguished because it avoids attempting to compress the entire input sentence into a single fixed-length vector. As an alternative, it adaptively selects a selection of these vectors while decoding the translation by encoding the input sentence into a series of vectors (Bahdanau et al., 2015). In several benchmarks and evaluation measures, NMT has attained cutting-edge performance. For instance, the Workshop on Machine Translation 2020 News Translation Task demonstrated that NMT outperformed Rule-Based Machine Translation (RBMT) across all language pairs and parameters. Modelling, inference, and learning are the main factors when creating an NMT model.

2.1.1 Modelling

Modelling creates neural network architecture that can precisely represent the conditional distribution of translations. This requires selecting the correct number of layers, the kind of layers to utilise (such as convolutional, recurrent, or transformer layers), the activation functions, and other hyperparameters. The objective is to develop a model that accurately represents the intricate interactions between the input and output sequences. Figure 1 illustrates an example of an NMT architecture, reflecting these complexities. A deep understanding of these components is vital for constructing an effective translation system (Bahdanau et al., 2015)

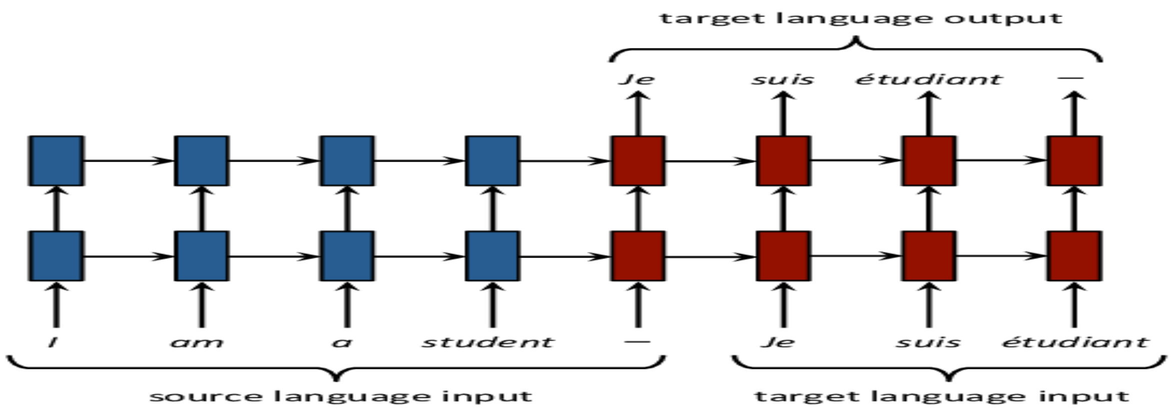


Fig1: Diagram of Neural Machine Translation Model (Luong and Manning, 2016)

1. Types of Layers and Their Importance

The construction of an NMT system requires a comprehensive understanding of various kinds of layers and their underlying functions. Specifically:

- Recurrent Layers such as LSTM and GRU are adept at managing sequential data, rendering them suitable for sequence learning tasks.
- Convolutional Layers specialise in spatial data processing, thereby effectively capturing spatial relationships.
- Transformer Layers adapt to both sequence-to-sequence and sequence-classification functions and have been shown to grasp global dependencies between words successfully (Vaswani et al., 2017).

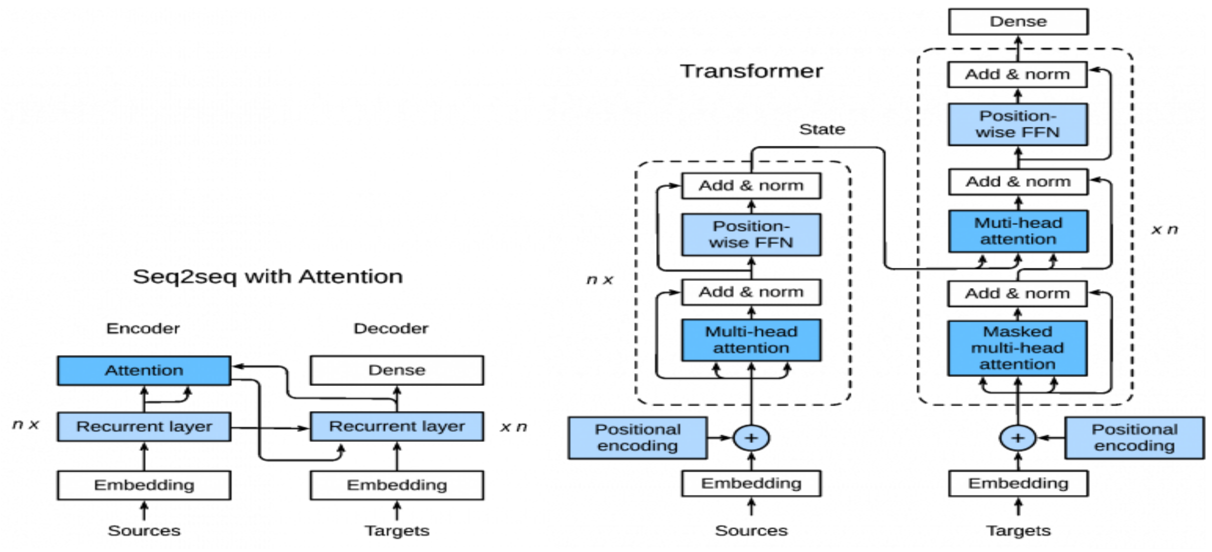


Fig 2: Transformer architecture (Vaswani et al.,2017)

2. Considerations in Layer selection

When modelling an NMT, the selection of appropriate layers and activation functions is important:

- Number of Layers: It's crucial to strike a balance between a deep network, which can model complicated relationships but is vulnerable to overfitting, and a shallow network, which is less likely to pick up on complex patterns.
- Activation Functions: Adding non-linearity using activation functions such as ReLU, sigmoid, and tanh enhances the network's ability to recognise complicated patterns.

3. Training Process

After being defined, the NMT model is trained using a sizable corpus of multilingual data. Minimising a loss function that measures the deviation between expected and actual translations, such as entropy, is crucial. The model's parameters are adjusted currently to reduce this loss and improve translation accuracy (Bahdanau et al., 2015).

2.1.2 Inference

The process of utilising a trained model to produce translations for new input sentences is called inference in neural machine translation. Passing the source sentence through the model generates a probability distribution of potential translations. The model's decoder network creates this probability distribution. The decoder network generates the conditional probability of each target word given the source sentence and previously created words using the encoder network output.

Probability Distribution Generation: The model's decoder network utilises the source sentence and previously generated words to create a probability distribution of potential translations, outlining the conditional likelihood of each target word.

Translation Selection Techniques: Several methods pinpoint the most likely translation. These include:

- Beam Search: Evaluates multiple potential translations to select the one with the highest aggregate probability.
- Greedy Decoding: Chooses the most probable word at each stage while examining numerous potential words.

Challenges and Limitations: These methodologies, while effective, may not always result in the best translation. Issues might arise concerning fluidity or repetition of words or phrases.

2.1.3 Learning

The development and improvement of translation systems rely heavily on learning, a fundamental aspect of machine translation. Learning in this context refers to how algorithms and models gain knowledge and patterns from data to enhance their translation capabilities. Advancements in neural networks and deep learning techniques have led to significant progress in learning algorithms in machine translation.

In machine translation, the training process is a crucial component of learning, where models are exposed to vast amounts of bilingual data to learn the associations between source and target languages. For example, the neural Machine Translation (NMT) models use parallel corpora, which comprise aligned sentences in both languages. The models learn to map the source language to the target language by adjusting their parameters based on the training data.

During the training phase, the primary aim is to minimise a loss function, such as cross-entropy, which measures the difference between predicted and actual translations. The model gradually enhances its translation performance by iteratively updating the model's parameters using gradient-based optimisation techniques. This process involves learning statistical patterns and dependencies between both languages' words, phrases, and sentences. Continuous learning and adaptation are crucial for keeping translation models up to date and enhancing performance. As new data becomes available or errors are identified, models can be fine-tuned or retrained to incorporate new knowledge and improve translation accuracy. This ongoing learning process enables translation systems to adapt to changes in language usage, domain-specific terminology, and linguistic variations.

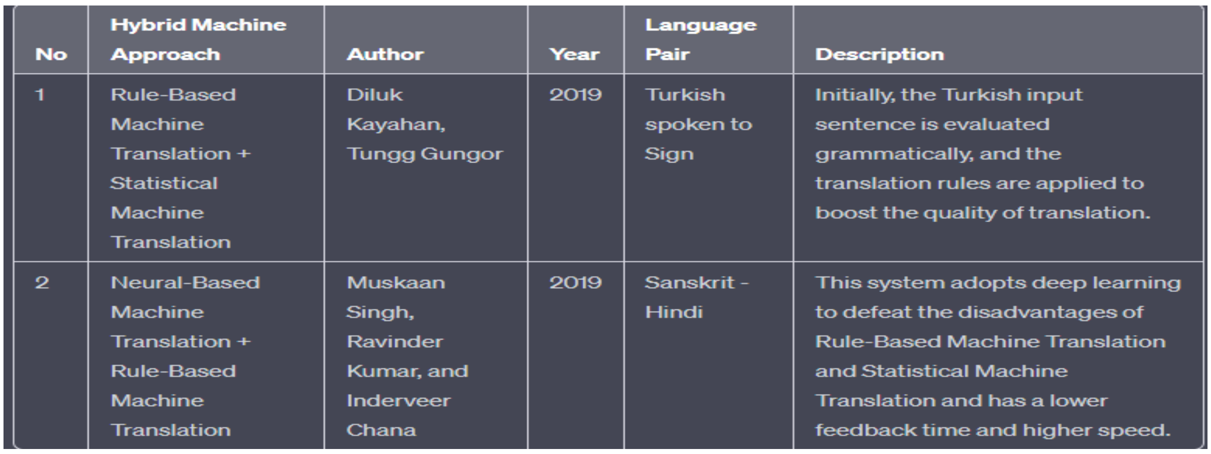
2.2 Hybrid Machine Translation (HMT)

Hybrid Machine Translation (HMT) is an approach to machine translation that blends rule-based and neural techniques to leverage the strengths and mitigate the weaknesses of each approach. HMT aims to harmonise the precision and grammatical correctness offered by Rule-Based Machine Translation (RBMT) with the fluency and contextual understanding provided by Neural Machine Translation (NMT).

RBMT in HMT: RBMT offers a set of linguistic guidelines and predetermined translation patterns. Created by linguists and language specialists, these guidelines superbly preserve the accuracy of translations and grammatical correctness by addressing linguistic issues like grammatical conventions, syntax, and idiomatic expressions. However, RBMT systems often struggle to capture the subtleties and variances of natural language, particularly with specialised or informal terms.

NMT in HMT: On the other hand, NMT utilises deep learning techniques to uncover patterns in translation from a vast amount of bilingual data (H. W. Xuan et al., 2011). NMT models excel at producing fluent translations and handling complex linguistic structures but may sometimes translate specialised terminology imprecisely.

Integration in HMT: HMT combines RBMT's clear linguistic rules with NMT's fluency, context understanding, and handling of idiomatic expressions. The synergy between these two approaches allows HMT to maximise their respective benefits while minimising their limitations (Anugu & Ramesh, 2020). Specific examples of such hybrid systems can be found in Table 1 below.



No	Hybrid Machine Approach	Author	Year	Language Pair	Description
1	Rule-Based Machine Translation + Statistical Machine Translation	Diluk Kayahan, Tungg Gungor	2019	Turkish spoken to Sign	Initially, the Turkish input sentence is evaluated grammatically, and the translation rules are applied to boost the quality of translation.
2	Neural-Based Machine Translation + Rule-Based Machine Translation	Muskaan Singh, Ravinder Kumar, and Inderveer Chana	2019	Sanskrit - Hindi	This system adopts deep learning to defeat the disadvantages of Rule-Based Machine Translation and Statistical Machine Translation and has a lower feedback time and higher speed.

Table 1: *Comparison of HMT systems (Anugu & Ramesh, 2020)*

Comparison and Practical Considerations: HMT has performed better than RBMT and NMT in several evaluation measures, including BLEU scores and human evaluations (H. W. Xuan et al., 2011). In a study conducted by (Akhand et al., 2021), the HMT system achieved higher translation quality in domain-specific translation tasks. By balancing language integrity and fluency, HMT systems can produce translations that are both correct and naturally sounding.

Advantages and Theoretical Considerations: Using rule-based and neural components has significant theoretical and practical advantages. The theoretical framework behind HMT allows for a versatile system that adapts to various translation needs. Further, more modules or elements like post-editing tools or external language resources enhance translation quality.

HMT represents a promising avenue in machine translation, bringing together the best aspects of RBMT and NMT. By combining linguistic expertise, clear guidelines, and machine learning, HMT stands as a flexible and robust solution for diverse translation challenges.

3. Conclusion

In conclusion, the study of neural machine translation (NMT) and hybrid machine translation (HMT) has resulted in substantial improvements in machine translation. NMT, a neural network-based technology, excels at interpreting idiomatic expressions, recognising context, and fluency. However, it could need help with using precise grammar and specialised language. HMT balances linguistic precision and fluency by combining rule-based and neural components. In several evaluation parameters, it performs better than standalone RBMT and NMT systems. While only one solution works for some, HMT is a viable option because of its well-rounded capabilities.

Machine translation benefits significantly from both NMT and HMT. The use of rule-based systems by HMT improves translation accuracy. Language barriers will continue to be overcome, and developments in this field will promote international communication.

4. Future Work

The evolution of Neural and Hybrid Machine Translation (MT) opens several avenues for future exploration. There is a need to enhance linguistic understanding to adapt to cultural nuances, idiomatic expressions, and evolving languages. Integrating multimodal inputs, such as images and voice, promises richer contextual translations. Special attention must be directed towards developing domain-specific models for fields like medicine, law, or technology and optimising models for real-time applications.

References

1. Koehn, P., 2017. 'Statistical Machine Translation: Draft of Chapter 13: Neural Machine Translation'. Center for Speech and Language Processing, Department of Computer Science, Johns Hopkins University, arXiv:1709.07809v1 [cs.CL], 22 Sep 2017.
2. Bahdanau, D., Cho, K.H., & Bengio, Y., 2015. 'Neural Machine Translation by Jointly Learning to Align and Translate'. Université de Montréal.
3. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., & Polosukhin, I., 2017. 'Attention Is All You Need'. arXiv: [cs.CL] (v7), Revised 2 Aug 2023.
4. Xuan, H.W., Li, W., & Tang, G.Y., 2012. 'An Advanced Review of Hybrid Machine Translation (HMT)'. Procedia Engineering, 29, pp.3017-3022. doi:**https://doi.org/10.1016/j.proeng.2012.01.432**.
5. Anugu, A. & Ramesh, G., 2020. 'A Survey on Hybrid Machine Translation'. E3S Web of Conferences, 184, p.01061. doi:**https://doi.org/10.1051/e3sconf/202018401061**.
6. Akhand, M.A.H., Roy, A., Dhar, A.C., & Kamal, M.A.S., 2021. 'Recent Progress, Emerging Techniques, and Future Research Prospects of Bangla Machine Translation: A Systematic Review'. Dept. of Computer Science and Engineering, Khulna University of Engineering & Technology; Graduate School of Science and Technology, Gunma University.
7. Kumar, K.M.C., Aswale, S., Shetgaonkar, P., Pawar, V., Kale, D., & Kamat, S., 2020. 'A Survey of Machine Translation Approaches for Konkani to English'. [online] IEEE Xplore. doi:**https://doi.org/10.1109/ic-ETITE47903.2020.110**.
8. Forcada, M.L., 2017. 'Making sense of neural machine translation'. Translation Spaces. A multidisciplinary, multimedia, and multilingual journal of translation, 6(2), pp.291-309. doi:**https://doi.org/10.1075/ts.6.2.06for**.
9. Liu, D. & Gildea, D., 2005. 'Syntactic Features for Evaluation of Machine Translation'. [online] ACLWeb. Available at: **https://aclanthology.org/W05-0904** [Accessed 14 May 2023].
10. Bahdanau, D., Cho, K., & Bengio, Y., 2014. 'Neural Machine Translation by Jointly Learning to Align and Translate'. [online] arXiv.org. Available at: **https://arxiv.org/abs/1409.0473**.
11. Luong, M.-T. & Manning, C.D., 2016. 'Achieving Open Vocabulary Neural Machine Translation with Hybrid Word-Character Models'. [online] arXiv.org. doi:**https://doi.org/10.48550/arXiv.1604.00788**.