Deception and Lie Detection Using Reduced Linguistic Features, Deep Models and Large Language Models for Transcribed Data

Tien Nguyen, Faranak Abri
San Jose State University, USA
{tien.t.nguyen04 | faranak.abri}@sjsu.edu

Akbar Siami Namin, Keith S. Jones
Texas Tech University, USA
{akbar.namin | keith.s.jones}@ttu.edu

***Abstract*—In recent years, there has been a growing interest in and focus on the automatic detection of deceptive behavior. This attention is justified by the wide range of applications that deception detection can have, especially in fields such as criminology. This study specifically aims to contribute to the field of deception detection by capturing transcribed data, analyzing textual data using Natural Language Processing (NLP) techniques, and comparing the performance of conventional models using linguistic features with the performance of Large Language Models (LLMs). In addition, the significance of applied linguistic features has been examined using different feature selection techniques. Through extensive experiments, we evaluated the effectiveness of both conventional and deep NLP models in detecting deception from speech. Applying different models to the Real-Life Trial dataset, a single layer of Bidirectional Long Short-Term Memory (BiLSTM) tuned by early stopping outperformed the other models. This model achieved an accuracy of 93.57% and an F1 score of 94.48%.**

***Index Terms*—Deception Detection; Natural Language Processing (NLP); Linguistic Features; Large Language Models (LLM)**

I. Introduction

Deception is manipulative human behavior that exploits other people's trust to gain personal advantages. This type of behavior can have significant and severe consequences in various aspects, ranging from damaging personal relationships and fraudulent business transactions to falsified testimonies in criminal investigations and court proceedings. Dishonesty can result in harmful outcomes, such as wrongful convictions or financial losses. As a result, detecting deception can potentially prevent harm in personal and professional relationships. More importantly, identifying deception can also aid in uncovering the truth and promoting justice in legal proceedings. Therefore, the ability to accurately detect deception in speech is of great interest to society.

The recent advancement in machine learning and deep learning algorithms enables the creation of classification schemes trained on textual features to accurately classify people's truthfulness in a given case or scenario. The use of machine learning in lie detection has gained significant attention in recent years, with many researchers achieving promising results in accurately detecting deception from speech by combining these features and using machine learning algorithms [1], [2].

978-1-7281-6251-5/20/$31.00 ©2020 IEEE

In this paper, we aim to use NLP for text analysis to detect deceptive contexts, contributing to the development of accurate lie detection methods. These methods, requiring minimal linguistic features, have potential applications across various fields, so we can protect individuals from deception's harmful consequences and maintain legal proceedings' integrity. The key contributions of this paper are as follows:

1) Development of a Minimalistic Textual Feature Set. We identify and utilize relevant linguistic features for deception detection. This avoids irrelevant data, leading to more efficient and accurate detection.
2) Comparative Analysis of Conventional and LLM-Based Models. Our study provides a comparative performance analysis between traditional NLP models and LLMs.
3) Experimentation on the "Real-Life" dataset. We apply our models to the Real-Life Trial dataset to ensure our findings are based on practical, real-world scenarios.

The structure of this paper is as follows: Section II reviews existing research in the deception detection field. Section III outlines our methodology, including feature extraction and selection processes. Section IV lays out the experimental setup, giving specifics on the dataset, preprocessing, and feature selection. Section V details the detection models created. Section VI presents the findings, emphasizing the comparative performance of these models in deception detection. Section VII discusses these results, providing insights into the effectiveness of various approaches and their alignment with existing literature. Lastly, Section VIII concludes the paper and suggests future work.

II. Related Work

This section reviews deception and lie detection approaches with respect to the verbal and non-verbal features leveraged in developing the detection models.

A. Textual and Audio Features Only

Researchers have taken different approaches to lie detection. One possible approach is to focus only on speech features, which can include textual and audio data because they both contain information about an individual's spoken content.

Mendels et al. [3] used different types of features in their deception detection models using the Columbia X-Cultural

Deception (CXD) Corpus. The study used spectral, acoustic-prosodic, such as pitch, duration, etc., and n-gram sets and concluded that the hybrid deep learning model that combines DNN and LSTM using both acoustic and lexical features gave the best F-1 scores, at 63.9%.

Kopev et al. [4] used a real-world political debate dataset instead of staged setups, which could provide more realistic scenarios for lie detection. Unlike other datasets, the political debate dataset studied had three labels: true, half-true, or false. The authors implemented a deep-learning model that incorporated audio and text using Linguistic Inquiry and Word Count (LIWC) [5] and Term Frequency - Inverse Document Frequency (TF-IDF), and metadata (i.e., speakers) features for deception detection. LIWC is a software tool that examines text to uncover linguistic features. TF-IDF is a method to determine how important a word is in a specific document compared to a larger group of documents while accounting for the fact that some words are common in most texts [6]. Their multi-input feed-forward neural network model using audio features consistently improved performance compared to only using textual and metadata features. Their model achieved an accuracy of 67%, a macro-average F1 (MA F1) of 45.07%, and a macro-average recall (MAR).

B. Textual, Audio, and Video Features

Analyzing text and audio has been effective for lie detection, but adding video cues like expressions, body language, and tone can enhance accuracy. This combination allows a more comprehensive and accurate analysis of speech.

Hsiao and Sun [7] proposed an attention-aware neural network that could identify which parts of audio, video, and transcription are most critical for deception detection. The authors extracted visual features represented as a 136-dimensional vector, audio features in MFCC format, and text features represented by 64-dimensional vectors per transcription. Once the features were extracted, the authors created separate visual, audio, and transcription models using BiLSTM. They then combined the three models as an ensemble model, which achieved 96% accuracy.

Sen et al. [8] collected videos from the actual trial and built models that used verbal, acoustic, and visual modalities to detect deception. Initially, they performed experiments with each set of features separately using Support Vector Machine (SVM), Random Forest (RF), and Neural Network (NN) classifiers. Then, they tried various combinations of features using early fusion and late fusion. The best accuracy (84.18%) was achieved through late fusion, combining visual and acoustic features using the NN classifier.

Hu et al. [9] collected a unique corpus of professionals practicing for oral exams while concealing information and identified signs of concealed information in speech and text. They conducted experiments to detect concealed information using different machine learning models automatically and developed a multi-task learning framework. The experiments show that a hybrid multi-task learning model (MLP, BiLSTM, and multi-task) gives the highest F1 score, 71.51%.

Zhang et al. [10] created a Graph-based Cross-modal Fusion Model (GCFM) along with a Cross-modal Attention Mechanism to detect deception in two datasets, the Real-Life Trial dataset [8]. They extracted visual, textual, and audio features by using a pre-trained ResNet and LSTM neural network with attention mechanisms. The proposed GCFM method achieved an accuracy of 88.14% and an F1-score of 78.50%. Additionally, using association learning increased the accuracy by 1.87% while the cross-modal attention mechanism improved the accuracy by 2.44%.

Sehrawa et al. [11] extracted text, audio, and video features from the Real Life Court Trial dataset, the Miami University Deception Detection dataset [12], and the Bag of Lies dataset [13]. The study achieves an accuracy of around 80% and up to 96% when using video transcriptions. When tested on the Bag-of-Lies dataset, the model achieves 85% accuracy. In the Real-Life Trial dataset, the model reaches an accuracy of 98.1%. The research uses a combination of deep learning techniques such as LSTM, BiLSTM, CNN, and RestNet50.

III. Methodology

Initially, we extracted 16 features from the texts. The 16 features are shown in Table I. Subsequently, we conducted a feature significance analysis using the Overlapping Coefficient (OVL) to assess the significance of specific features in distinguishing between different categories [14]. Furthermore, we employed the stepwise method for feature selection, a widely used technique in machine learning, to identify the most relevant subset of features. By combining stepwise with an SVM, we iteratively added/removed features while evaluating their impact on the model's performance [15]. This iterative process allowed for a comprehensive assessment of the significance of each feature, ensuring that only the most informative ones were retained. The procedure commenced with either an empty set or an initial feature selection, subsequently evaluating the importance of each feature using statistical criteria. This evaluation involved training the model with the selected features and measuring its performance using accuracy and F1 score metrics. Based on these performance metrics, a decision was made to include or exclude a feature, and the process continued until a predefined stopping condition was met.

Considering the performance of deep learning and large language models in similar research [16]–[18], we also utilized BiLSTM models to classify textual data from the dataset used in this research paper. The BiLSTM model leverages complete sequential information by considering past and future data points for each position in the sequence, thereby enhancing the original LSTM designed for sequence learning [19].

IV. Experimental Setup

A. Dataset Description

The Real Life Trial dataset [8] is a set of videos from public court trials collected by Dr. Mihalcea's group at the University of Michigan. It contains a total of 121 brief videos. These videos are divided into 61 videos that depict real instances of deception and 60 videos that depict real instances of

truthfulness. In addition to the videos, the dataset also includes transcriptions of each video and annotations of the gestures, such as smile, laugh, scowl, etc., made by the individuals in the videos.

B. Preprocessing and Cleaning

Our text processing pipeline involves several steps. First, we removed non-alphabetic characters from the text to ensure only alphabetic letters remained. This step is crucial to avoid any noise in the data that could affect the subsequent processing steps. For conventional models, we used these texts to extract new features. For deep models, we further implemented stemming for each word in the text Stemming is the process of reducing each word to its root form by removing any suffixes. This step helps to reduce the number of unique words in the text, which is useful for subsequent analysis. After the text had been preprocessed, we performed one-hot encoding. This process involves representing each word in the text as a unique integer index, where the vocabulary size is 5000. Machine learning algorithms allow us to represent the text in an easily digestible format. Finally, we padded the encoded sequences with zeros to ensure they were all the same length (i.e., 221). This step is important because machine learning algorithms require input data to have a fixed shape. Padding ensured that the sequences contained the same number of features.

C. Linguistic Features Selection

After preprocessing, we extracted 16 textual features, or lexicons, that might be considered for lie detection from textual data. For instance, the sentiment score in the text was also extracted as a compound score, a measurement that sums up the scores assigned to words in a lexicon, ranging from -1 to 1. A score close to 1 indicates positive emotions, while a score close to -1 indicates negative emotions. A score of 0 represents neutral sentiment. Part-of-speech tagging was used to determine the frequency of adjectives and adverbs, providing insights into the descriptive language used. Additionally, the number of pronouns, conjunctions, and verb tenses (i.e., past, present, and future) is computed to understand the speaker's perspective, discourse structure, and temporal references. The count of filler words such as 'um', 'uh', 'hmm' or 'like', repetition of words, negations, and self-references provide further insights into the speaker's fluency, rhetorical style, emphasis, and attempts to persuade the audience. The rest of the features are described in Table I. Together, these features enable a comprehensive text analysis, contributing to lie detection based on linguistic patterns.

Feature Name	**Description**
Word Count	The total number of words in the text.
Sentence Count	The total number of sentences in the text.
Sentiment Score	A numerical score indicating the overall sentiment of the text.
Average Word Length	The average length of words in the text.
Vocabulary Diversity	The ratio of unique words to the total number of words in the text.
Adjective Frequency	The proportion of adjectives in the text.
Adverb Frequency	The proportion of adverbs in the text.
Pronoun Frequency	The proportion of pronouns in the text.
Conjunction Frequency	The proportion of conjunctions in the text.
Past Tense Frequency	The proportion of verbs in the past tense in the text.
Present Tense Frequency	The proportion of verbs in the present tense in the text.
Future Tense Frequency	The proportion of verbs in the future tense in the text.
Filler Word Count	The number of common filler words in the text.
Repetition Count	The proportion of words that appear more than once in the text.
Negation Count	The number of negations in the text.
Self-Reference Count	The number of self-referential words in the text (e.g., "I," "me," "myself").

TABLE I: Description for Extracted Linguistic Features

V. Detection Models

A. Conventional Models

To train our deception detection models, we explored various conventional algorithms such as:

1) Support Vector Machines (SVM) (called Model 1)
2) k-Nearest Neighbors (KNN) (called Model 2)
3) Logistic Regression (LG) (called Model 3)

To optimize their performance, we conducted a grid search to fine-tune each model's hyperparameters. This thorough parameter tuning significantly improved our models' predictive power.

B. Deep Models and Pre-trained Models

1) Model 4: 1 BiLSTM: We created a sequential Bidirectional neural network model. The model had three layers: an embedding layer, a BiLSTM layer, and a dense layer with a sigmoid activation function. The embedding layer mapped the integer-encoded words to dense vectors of fixed size. The BiLSTM layer takes the embedded sequence as input and produces a sequence of output vectors. The dense layer outputs a single sigmoid value that represents the probability of the input text being deceptive or truthful. The model was compiled with the binary cross-entropy loss function, the Adam optimizer, and the accuracy metric.

2) Model 5: 1 BiLSTM + Dropout Layer: We modified model 5. A dropout layer was added to prevent overfitting, followed by a BiLSTM layer. The output of the LSTM layer was then passed through a GlobalMaxPool1D layer, which selected the maximum value from each feature map, producing a 1D vector. This vector was then passed through two fully connected layers with 64 and 1 neuron(s), respectively, using ReLU and sigmoid activation functions. Finally, the binary cross-entropy loss function optimized the model parameters with the Adam optimizer.

3) Model 6: 1 BiLSTM + Early Stopping: Early stopping was used to prevent overfitting. The training process stopped if there was no improvement in validation loss for five consecutive epochs.

4) Model 7: Bert + Early Stopping + Dropout: BERT (Bidirectional Encoder Representations from Transformers) is a language model developed by Google that understands

the context of words in a sentence by considering both the words before and after [20]. It uses a bidirectional approach and a transformer architecture to create context-aware word representations, enabling it to perform well in a variety of natural language processing tasks [20]. TFBertForSequence-Classification, a TensorFlow 2.0 compatible implementation of the BERT model for sequence classification, was used. It takes a sequence of tokens as input and outputs a probability distribution over a set of labels for that sequence. It also loaded the pre-trained weights for the specified model, 'bert-base-uncased', which was a pre-trained BERT model with uncased English text. The loss function used was Sparse Categorical Cross-entropy, which was suitable for multi-class classification tasks. The optimizer used was Adam, with a learning rate of 2e-5 and an epsilon value of 1e-08. The model was compiled with the specified loss function, optimizer, and metrics.

5) Model 8: Pretrained GPT-2 model: . The pretrained GPT-2 model is an advanced language model created by OpenAI that has already learned from a large amount of text data, enabling it to generate coherent and contextually relevant text based on a given prompt [21]. We built a GPT2 model that was initialized with pre-trained weights using the $GPT2Model.from_pretrained()$ method. A linear layer (i.e., self.fc1) was added to the model that took the hidden states and performed a linear transformation to perform sequence classification. In the $forward()$ method, the input ID and mask were passed to the GPT-2 model, and the output was flattened using $gpt_out.view(batchsize, -1)$ and passed through the linear layer to generate the final output. This architecture uses the GPT-2 model as a feature extractor and transforms the extracted features into class predictions.

6) Model 9: Pretrained Roberta model: The pretrained RoBERTa (Robustly optimized BERT approach) model, developed by Facebook AI Research, is an improved version of BERT. It enhances performance by optimizing the pretraining process through hyperparameter adjustment, training data modification, and the removal of the next sentence prediction task [22]. We built a RoBERTa model for sequence classification using the PyTorch framework and the Hugging Face Transformers library. The 'roberta-base' pre-trained model and tokenizer were used. The input corpus was tokenized, and the resulting tokenized sequences were then padded to a fixed length to ensure consistent input dimensions. Both the tokenized sequences and corresponding labels were converted into tensors for efficient processing. An Adam optimizer with a learning rate of 2e-5 was used to train the model, while the loss function employed was the cross-entropy loss.

C. Model Evaluation

We evaluated the models using 5-fold cross-validation. This ensures robustness, reduce the variance in models caused by randomness and parameter tuning. The model's performance was evaluated using accuracy and the F1 score.

Features	OVL Score
filler_word_count	0.5471
future_tense_frequency	0.5517
negation_count	0.6097
adverb_frequency	0.7367
present_tense_frequency	0.7512
sentence_count	0.7811
self_reference_count	0.8106
sentiment_score	0.8159
adjective_frequency	0.8182
word_count	0.8214
pronoun_frequency	0.8299
past_tense_frequency	0.8345
avg_word_length	0.8479
repetition_count	0.8497
conjunction_frequency	0.9001
vocabulary_diversity	0.9119

TABLE II: Feature Significance Analysis using OVL

VI. Results

A. Overlapping Probability Density Functions

Feature significance analysis using the Overlapping Coefficient (OVL) is a method that assesses the importance of specific features in distinguishing between different categories, such as lies and truths. This method involves comparing the probability density functions (PDFs) of these features for each category, enabling the measurement of their overlap by calculating the OVL value [14]. By quantifying the degree of overlap, the OVL value provides valuable insights into the effectiveness of a feature in differentiating between categories. When the OVL value is low, it indicates that a particular feature is highly effective at distinguishing between the categories, as their PDFs demonstrate minimal similarity. Conversely, when the OVL value is high, it suggests that the feature might not be as effective in differentiation, as the overlap between the PDFs is more substantial.

Table II presents the quantitative results of the Overlapping Probability Density Functions analysis. This analysis provides a more precise measure of the discriminatory power of individual features in distinguishing between the "Lie" and "Truth" categories. By calculating the OVL scores, we can determine how much the probability density functions of different features overlap between the two categories.

Features such as "vocabulary diversity" and "conjunction frequency" exhibit high OVL scores. This indicates a substantial overlap in their probability density functions between the "Lie" and "Truth" categories. This suggests that these features may not be strong indicators on their own when it comes to distinguishing between lies and truth. On the other hand, features like "filler word count" and "negation count" display lower OVL scores, implying less overlap in their probability density functions. This indicates a higher potential for effectively distinguishing between instances of "Lie" and "Truth" using these features. However, it is important to note that feature interactions and the analysis context can significantly influence their discriminatory power.

B. Detection Models

1) Conventional Models: From the initial set of 16 features shown in Table I, the stepwise approach carefully chose five

Model	Train Acc	Test Acc	F1
1: SVM	64.46	63.77	69.8
2: KNN	71.69	62.83	63.07
3: LR	66.11	68.53	71.69
4: 1 BiLSTM	100	67.73	69.83
5: 1 BiLSTM + Dropout	100	66.9	66.18
6: 1 BiLSTM + Early Stopping	100	93.57	**94.48**
7: Bert + Early Stopping + Dropout	83.54	68.73	64.63
8: Pretrained GPT2 model	99.79	58.73	60.12
9: Pretrained Roberta model	88.18	71.2	73.71

TABLE III: Accuracy and F1 scores obtained by the models.

features that showed the strongest discriminatory potential: 1) average word length; 2) vocabulary diversity; 3) frequency of adjectives; 4) frequency of adverbs; and 5) the count of filler words. These features played a crucial role in our efforts to detect deception.

Table III reports an evaluation of conventional models in terms of accuracy and F1 scores. SVM achieves relatively lower test accuracy and F1 score. KNN shows reasonable training accuracy but faces challenges in generalization, with a lower test accuracy and F1 score. LR stands out with its test accuracy of 68.53% and F1 score of 71.69. These results highlight the strong potential of this model for distinguishing deceptive actions.

2) Deep Models: Table III summarizes the performance of different deep models with various architectures and techniques. Model 4, with only one BiLSTM layer, shows improvements in accuracy (67.73%) and F1 score (69.83%), indicating the importance of simplifying the model structure. Model 5 incorporates a Dropout layer alongside a single BiLSTM layer, demonstrating the impact of regularization techniques. However, its accuracy (66.9%) and F1 score (66.18%) are slightly lower than Model 8. Model 6 introduces Early Stopping and significantly enhances predictive performance. Model 6 achieves an impressive accuracy of 93.57% and an F1 score of 94.48%. This finding highlights the importance of monitoring the validation loss during training to prevent overfitting. Among the three pre-trained models, Model 9 applied pretrained Roberta, giving the highest scores. Table III reveals the performance variations among different models and emphasizes the importance of carefully selecting architecture and techniques. The findings further show that regularization techniques, such as Early Stopping, can help prevent overfitting and improve generalization capabilities.

VII. Discussion

Average word length, vocabulary diversity, frequency of adjectives, frequency of adverbs, and count of filler words are selected by stepwise method. On the other hand, the OVL method prioritized a different set of features: the count of filler words, frequency of adverbs, future tense frequency, negation count, and present tense frequency. The OVL approach emphasizes features related to grammatical and syntactical aspects, such as tense and negation. These features can provide valuable insights into deceptive language patterns. The count of filler words appears in both sets of selected features. These words are often used as hesitations or distractions and may serve as markers of deceptive speech. The frequency of the adverb feature, which is also included in both selections, suggests that the intensity or manner of expression in speech could play a role in deception detection.

LR (model 3) shows signs of underfitting in our analysis. The relatively low train accuracy of 66.11% implies that the model is not complex enough and struggles to fit the training data adequately. However, the test accuracy is higher at 68.53%. This difference between train and test accuracy is a classic indicator of underfitting. This underfitting issue may be attributed to the simplicity of the LR model, which may not be able to capture complex, nonlinear relationships within the data. Consequently, the LR model's limited capacity to capture these complex patterns ultimately compromises its overall performance and prevents it from achieving higher accuracy on both the train and test sets. By exploring more sophisticated models, such as deep models, we can strive to improve our models' accuracy and generalization capabilities, ultimately enhancing our analysis's overall performance.

For the deep models, we examined regularization techniques and abilities to manage overfitting. Model 4, with just 1 BiLSTM layer, shows signs of overfitting. This means it may not perform well on new data. In the context of deep learning model optimization, Dropout and Early Stopping are crucial techniques that play a vital role in addressing the challenges of overfitting and enhancing the overall performance of a model. Dropout is a probabilistic method that helps trim the neural network by randomly dropping out certain units during training [23]. However, in our case, adding Dropout to Model 5 did not lead to better performance compared to Model 4.

Model 5 had similar training accuracy to Model 4 but slightly lower test accuracy and F1 score. This implies that introducing Dropout in Model 5 did not effectively address overfitting or enhance the model's ability to generalize to the test data. Model 6, an extension of Model 4 with the addition of early stopping, shows a significant improvement in performance compared to both Models 4 and 5. With a test accuracy of 93.57% and an impressive F1 score of 94.48%, this model excels in balancing precision and recall. It is particularly suitable for tasks where minimizing false positives and false negatives is crucial. While maintaining the high training accuracy observed in Model 4, Model 6 achieves significantly higher test accuracy and F1 score due to the inclusion of early stopping. This indicates that early stopping effectively addresses the overfitting issue present in Model 4. Early Stopping is a technique that controls the number of epochs used in both the backpropagation and forward propagation operations, aiming to prevent overfitting and find the optimal point of model performance [23]. These findings emphasize the significance of selecting appropriate regularization techniques to address overfitting and achieve optimal model performance in specific tasks.

The success of Model 6 highlights the importance of early stopping as a powerful tool in deep learning and makes it a great choice for applications that value simplicity and high performance. When compared to the BiLSTM-based models,

the pretrained models have not proven to be the best choice for the specific task at hand. The effectiveness of a model, whether it is pretrained or not, depends heavily on the dataset's nature and the task's characteristics.

Sehrawat et al.'s model stands out with an exceptional accuracy of 98.1% on the Real-life trial dataset [11]. This achievement notably outperforms our best model, Model 6. It's worth noting that Sehrawat et al.'s approach, which incorporates Mel Spectrograms features, word dimension vector features, and video frame dimension features from audio, transcription, and video sources, demonstrates the effectiveness of a multi-modal approach. Additionally, when compared to other researchers such as Hsiao and Sun [7], and Chebbi et al. [24], our best model, Model 6, falls short in terms of accuracy. These studies also incorporated multi-modal data sources, combining transcription, audio, and video features to train their models. This highlights the value of cross-modal fusion and the richer information available in multi-modal datasets. While Model 6 demonstrates the potential of a single-modal approach by achieving a high F1 score and balancing precision and recall, the comparison emphasizes the advantages of multi-modal models in deception detection. These models benefit from a more comprehensive representation of deceptive behavior, capitalizing on audio, video, and textual cues. Given that our models were exclusively applied to textual data, achieving an F1 performance of 94.48% is a noteworthy result.

VIII. Conclusion and Future Work

In this work, we extracted a total of 16 textual features. Using both the stepwise method and the OVL method, we identified 5 highly significant features from each approach. We conducted experiments using both conventional models and deep models. Our findings show that among the conventional models, LR achieved the highest accuracy with an impressive 68.53% and an F1 score of 71.69%. However, the deep model consisting of a single layer of BiLSTM with Early Stopping outperformed all other models. This deep model achieved an outstanding accuracy of 93.57% and an F1 score of 94.48%.

In future work, we aim to extract new audio features and improve detection models by integrating them with textual features. This multi-modal approach can enhance understanding. Although the Real Life Trial dataset's results are promising, its small size limits robustness and generalizability. Testing larger datasets will validate our model across various contexts and populations.

Acknowledgment

This research was supported by the U.S. National Science Foundation (Awards#: 2319802 and 2319803) and by the U.S. Office of Naval Research (Award#: N00014-21-1-2007).

References

[1] F. M. Talaat, "Explainable enhanced recurrent neural network for lie detection using voice stress analysis," *Multimedia Tools and Applications*, pp. 1–23, 2023.

[2] M. Aslan, M. Baykara, and T. B. Alakuş, "Lstmncp: lie detection from eeg signals with novel hybrid deep learning method," *Multimedia Tools and Applications*, pp. 1–17, 2023.

[3] G. Mendels, S. I. Levitan, K.-Z. Lee, and J. Hirschberg, "Hybrid acoustic-lexical deep learning approach for deception detection." in *Interspeech*, 2017, pp. 1472–1476.

[4] D. Kopev, A. Ali, I. Koychev, and P. Nakov, "Detecting deception in political debates using acoustic and textual features," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2019, pp. 652–659.

[5] J. W. Pennebaker, M. E. Francis, and R. J. Booth, "Linguistic inquiry and word count: Liwc 2001," *Mahway: Lawrence Erlbaum Associates*, vol. 71, no. 2001, p. 2001, 2001.

[6] J. D. Ullman, J. Leskovec *et al.*, "Mining of massive datasets," 2014.

[7] S.-W. Hsiao and C.-Y. Sun, "Attention-aware multi-modal rnn for deception detection," in *2022 IEEE International Conference on Big Data (Big Data)*. IEEE, 2022, pp. 3593–3596.

[8] M. U. Şen, V. Perez-Rosas, B. Yanikoglu, M. Abouelenien, M. Burzo, and R. Mihalcea, "Multimodal deception detection using real-life trial data," *IEEE Transactions on Affective Computing*, vol. 13, no. 1, pp. 306–319, 2020.

[9] S. Hu, "Detecting concealed information in text and speech," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 402–412.

[10] H. Zhang, Y. Ding, L. Cao, X. Wang, and L. Feng, "Fine-grained question-level deception detection via graph-based learning and cross-modal fusion," *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 2452–2467, 2022.

[11] P. K. Sehrawat, R. Kumar, N. Kumar, and D. K. Vishwakarma, "Deception detection using a multimodal stacked bi-lstm model," in *2023 International Conference on Innovative Data Communication Technologies and Application (ICIDCA)*. IEEE, 2023, pp. 318–326.

[12] E. P. Lloyd, J. C. Deska, K. Hugenberg, A. R. McConnell, B. T. Humphrey, and J. W. Kunstman, "Miami university deception detection database," *Behavior research methods*, vol. 51, pp. 429–439, 2019.

[13] V. Gupta, M. Agarwal, M. Arora, T. Chakraborty, R. Singh, and M. Vatsa, "Bag-of-lies: A multimodal dataset for deception detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0.

[14] A. S. Namin, F. Abri, L. F. Gutierrez, K. Jones, and D. Sears, "Linguistic features for detecting fake reviews," 2020.

[15] M. Arshad, D. Zhao, E. Zare, M. Sefton, and J. Triantafilis, "Proximally sensed digital data library to predict topsoil clay across multiple sugarcane fields of australia: Applicability of local and universal support vector machine," *Catena*, vol. 196, p. 104934, 2021.

[16] B. Sachithanandam, A. S. Namin, and F. Abri, "The performance of machine and deep learning algorithms in detecting fake reviews," in *2023 IEEE International Conference on Big Data (BigData)*, 2023, pp. 2499–2507.

[17] D. O. Otieno, F. Abri, A. S. Namin, and K. S. Jones, "Detecting phishing urls using the bert transformer model," in *2023 IEEE International Conference on Big Data (BigData)*, 2023, pp. 2483–2492.

[18] B. Karki, F. Abri, A. S. Namin, and K. S. Jones, "Using transformers for identification of persuasion principles in phishing emails," in *2022 IEEE International Conference on Big Data (Big Data)*, 2022, pp. 2841–2848.

[19] F. Zhang, C. Hu, Q. Yin, W. Li, H.-C. Li, and W. Hong, "Multi-aspect-aware bidirectional lstm networks for synthetic aperture radar target recognition," *Ieee Access*, vol. 5, pp. 26 880–26 891, 2017.

[20] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[21] I. Solaiman, M. Brundage, J. Clark, A. Askell, A. Herbert-Voss, J. Wu, A. Radford, G. Krueger, J. W. Kim, S. Kreps, M. McCain, A. Newhouse, J. Blazakis, K. McGuffie, and J. Wang, "Release strategies and the social impacts of language models," 2019.

[22] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," 2019.

[23] C. Sitaula and N. Ghimire, "An analysis of early stopping and dropout regularization in deep learning," *International Journal of Conceptions on Computing and Information Technology*, vol. 5, no. 1, pp. 17–20, 2017.

[24] S. Chebbi and S. B. Jebara, "Deception detection using multimodal fusion approaches," *Multimedia Tools and Applications*, pp. 1–30, 2021.