Detecting Deceptive Communication in Diplomacy

Abhay Dagar
abhay22014@iiitd.ac.in
IIIT Delhi

Mayank Kumar
mayank22284@iiitd.ac.in
IIIT Delhi

Tushar
tushar22544@iiitd.ac.in
IIIT Delhi

Abstract

This study uses the QANTA dataset to classify deceptive messages in the Diplomacy board game, aiming to distinguish truthful from deceptive intents. The baseline model combines TF-IDF Bag-of-Words representations with logistic regression for binary classification, offering a quick and interpretable approach with strong accuracy. The findings lay the foundation for exploring advanced architectures like LSTMs or transformers in future work.

1 Introduction

Detecting deception in text is complex, especially in manipulation-driven settings like the Diplomacy game. Diplomacy involves forming alliances and betraying opponents strategically. The QANTA Diplomacy dataset enables analysis of deceptive communication in this context.

The goal is to classify messages as truthful or deceptive using textual data. This has broader implications for improving dialogue systems. It also aids strategic reasoning in multi-agent environments.

2 Literature Review

Ott et al. (2011) used linguistic features to detect deception in text, but their approach struggled with complex environments like conversation or gameplay. BERT, a transformer model, improved detection by understanding word context, making it more accurate for detecting subtle deception.

Deception in gameplay, like in Diplomacy, is challenging due to strategic, indirect communication. The 2020 ACL Diplomacy dataset helps study deception in multiplayer settings, enabling the development of models tailored to dynamic, context-rich gameplay communication.

3 Dataset Description

The **Diplomacy dataset** consists of deception-annotated dialogues in `JSONLines` format, with each entry representing a complete game. Key elements include:

- **Raw Messages**: Text exchanged between players.
- **Sender and Receiver Labels**: Indicate whether the message is truthful (True), deceptive (False), or unannotated (NOANNOTATION).
- **Sender and Recipient**: Identify the players involved.
- **Absolute and Relative Message Indices**: Track the message's position within the entire game and specific turns.
- **Timestamps (Season, Year)**: Provide timing context within the game's progression.
- **Scores and Score Differences**: Reflect player performance and competitive dynamics.
- **Game IDs (0–11)**: Unique identifiers for each game.
- **Player List**: Lists the players involved in the game.

This dataset captures **strategic deception** in multi-turn interactions, offering **temporal** and **contextual cues** to analyze deceptive communication. It spans 12 games, with natural and strategic dialogue exchanges.

4 Data Preprocessing

4.1 Text Preprocessing

The text data was preprocessed by converting all text to lowercase and removing non-alphabetic characters. Additionally, whitespace was normalized to ensure consistent formatting.

4.2 Vectorization

Two text vectorization methods were used:

- **Bag-of-Words (BOW)**: This method represents each text as a set of word frequencies, capturing the presence of words in the text without considering the order.
- **Term Frequency-Inverse Document Frequency (TF-IDF)**: TF-IDF weighs words based on their frequency in a specific document relative to their occurrence across all documents, emphasizing unique terms for each message.

4.3 Feature Extraction

Linguistic features were extracted to capture deception cues, such as message length, pronoun usage, and the frequency of question marks. Sentiment features were derived by identifying positive and negative words in the text. Game state features included temporal data (e.g., game phase and season) and normalized game scores.

4.4 Standardization

All features were standardized to ensure that each feature contributed equally during model training.

5 Methodology And Model Details

In this research, we followed a progressive approach with three models to analyze the deception-annotated dialogues. The flow of our model development is as follows:

5.1 Baseline I: Bag-of-Words + Logistic Regression (SGD)

The first baseline model combines **Bag-of-Words (BoW)** representation with **Logistic Regression** trained via `SGDClassifier`.

5.2 Baseline II: Hierarchical LSTM with BERT Pooling

The second baseline model incorporates more advanced techniques, utilizing **BERT embeddings** and a **Hierarchical LSTM**.

5.3 Novel Approach: Context-Aware Adversarial Graph-Transformer (CAAGT) Approach

The third model represents a novel approach designed to enhance the performance.This approach models deception as a multi-modal problem, combining transformer-based text analysis with graph neural networks to capture relationships and communication patterns, while addressing class imbalance using data augmentation, Focal Loss, and SMOTE. .

6 Baseline I: Logistic regression (SGD)

6.1 Preprocessing and Vectorization

The preprocessing pipeline includes the following steps:

- Text normalization: All text data is converted to lowercase.
- Removal of stop words: Common words that do not contribute to meaningful features are removed.
- Filtering: Messages labeled as "NOANNOTATION" or irrelevant are excluded from the dataset.

For vectorization, two methods are considered:

- **CountVectorizer**: This method generates a document-term matrix based on word frequency.
- **TfidfVectorizer**: TF-IDF is used to adjust word frequencies by their inverse document frequency, highlighting unique terms.

Both vectorizers are constrained with a maximum of 5000 features using the `max_features=5000` parameter to ensure efficiency.

6.2 Feature Augmentation (Optional)

If the variable `POWER` is enabled (i.e., `POWER == "y"`), additional game-related features such as message sequence, season, and game scores are appended to the feature set to provide contextual insights.

6.3 Model Setup

The model is initialized using the `SGDClassifier` with the `log_loss` loss function. To handle potential class imbalance in the dataset, the `sample_weight` parameter is used to assign different weights to samples during training.

6.4 Training

The model is trained for 100 epochs using the `partial_fit` method, which allows for incremental learning. A decaying learning rate is applied to improve convergence. During training, key evaluation metrics such as log loss, accuracy, and F1 score are recorded to monitor the model's performance. The model flow is expressed in Fig. 1.

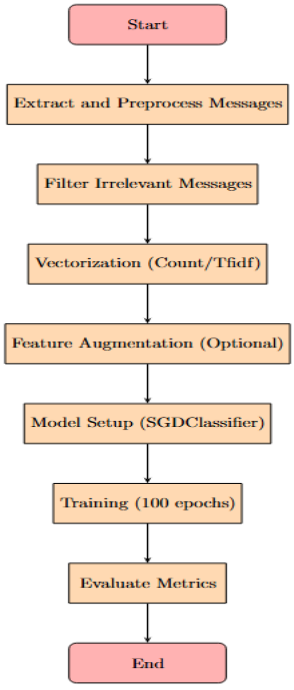

Figure 1: Baseline-I: Model

6.5 Evaluation

Metrics are evaluated for both vectorizer types (CountVectorizer and TfidfVectorizer) to compare the performance of each method. The results are stored for further analysis. The performance of the models was evaluated on the test data using accuracy and macro F1-score as evaluation metrics. The results for the Bag-of-Words (BoW) and TF-IDF models are summarized below:

- **Bag-of-Words (BoW)**: The BoW model achieved an accuracy of 80.1% on the test data. The macro F1-score for BoW was 88.8%.
- **TF-IDF**: The TF-IDF model achieved an accuracy of 76.32% on the test data. The macro F1-score for TF-IDF was 86%.

These results highlight the strong performance of the Bag-of-Words model in terms of accuracy and its higher macro F1-score, suggesting it better handles class imbalance compared to the TF-IDF model.The plots of baseline model 1 refer to figure 2.

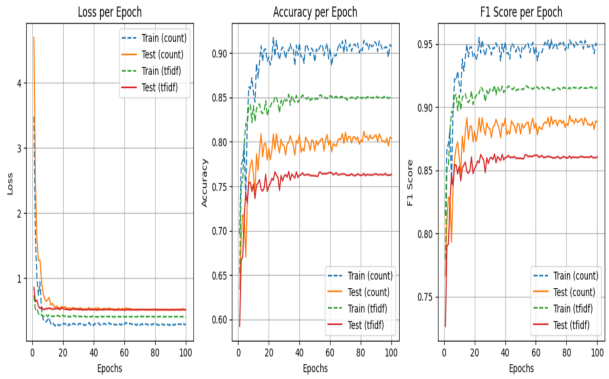

Figure 2: Baseline-I: Results

7 Baseline 2: Hierarchical BERT-LSTM with Class Balancing

To detect deception in multi-turn conversations, we propose a Hierarchical BERT-LSTM model that combines contextual encoding with temporal modeling.

7.1 Dataset Processing

We use the Diplomacy dataset, where each instance includes messages and binary deception labels. Ambiguous entries are removed, and conversations are capped at 10 messages. Each message is tokenized using BERT (max length = 64) with padding, truncation, and attention masks enabled.

7.2 Model Architecture

- **Message Encoder:** Each message is encoded using `bert-base-uncased`, extracting the [CLS] token. The first 6 BERT layers are frozen, and gradient checkpointing is used to reduce memory usage.
- **Sequence Encoder:** A single-layer bidirectional LSTM (hidden size = 128) captures temporal patterns among messages.
- **Classifier Head:**
  - Linear layer ($256 \rightarrow 64$)
  - ReLU activation
  - Dropout (p = 0.3)
  - Final linear layer ($64 \rightarrow 2$) for binary classification

7.3 Loss and Optimization

We use weighted cross-entropy loss (positive class weight = 15.0) with label smoothing ($\varepsilon = 0.1$). The model is trained with the AdamW optimizer (learning rate = 2×10^{-5}) and L2 weight decay. A `ReduceLROnPlateau` scheduler adjusts the learning rate based on validation F1-score.

7.4 Training Strategy

Mixed-precision training with `torch.cuda.amp` is used to lower memory usage. Gradient accumulation simulates larger batch sizes, and gradients are clipped to a maximum norm of 1.0. The model flow can be visualized in Fig 3.

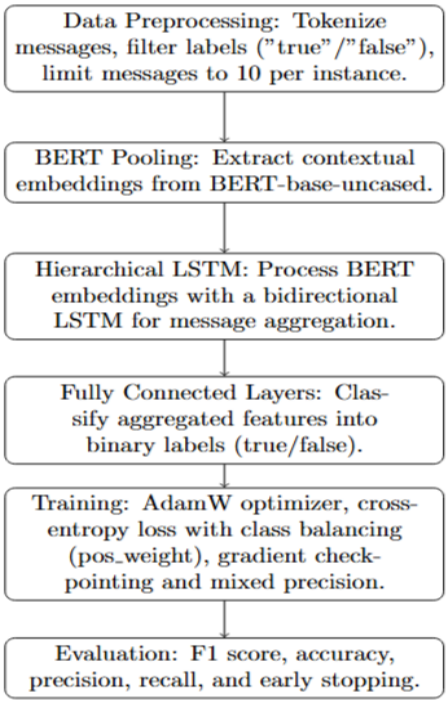

Figure 3: Baseline-II: Model

7.5 Evaluation

After each epoch, the model is evaluated using F1-score, accuracy, precision, and recall. The best model (based on highest validation F1-score) is saved. Early stopping is applied if the validation F1-score does not improve for 3 consecutive epochs.

Training Metrics:

- **Training Loss:** Steadily decreased from 0.1983 to 0.0726, indicating effective learning and reduction in training error.
- **Validation F1-Score:** Improved from 0.3004 to 0.4875, suggesting better balance between precision and recall on the validation set.

- **Validation Accuracy:** Increased from 88.46% to 92.10%, showing enhanced correctness in predictions over time.

The evaluation matrices can be seen in fig 4,

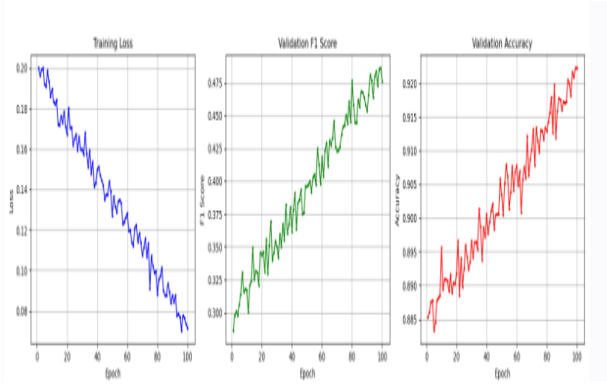

Figure 4: Baseline-II: Evaluation

8 Novel Approach: Context-Aware Adversarial Graph-Transformer (CAAGT) Approach

In this section, we describe our proposed approach for detecting deception in multi-turn conversations within the Diplomacy game. The model leverages both linguistic and graph-based features to capture the complexities of communication patterns and interactions.

8.1 Data Processing and Preparation

Our approach begins with the processing of Diplomacy game messages in the JSONL format. The following steps are taken:

- **Train/Validation/Test Splits**: The data is split into training, validation, and test sets to ensure generalization.
- **Context Extraction**: For each target message, the context is enriched by including up to three previous messages, enabling the model to understand the conversation history.
- **Text Preprocessing**: Standardized text cleaning is performed, including case normalization and punctuation spacing, to maintain consistency across the data.
- **Player Interaction Graphs**: We construct graphs based on the interactions between players in each game, capturing communication patterns over time.

8.2 Feature Engineering

We incorporate multiple feature types to enrich the model's input:

- **Text Features**: Messages are tokenized using the RoBERTa tokenizer, with a maximum token length of 160 to fit within model constraints.
- **Linguistic Features**: Ten deception markers are extracted from the text, such as question marks, pronouns, and tentative language, which are indicative of deceptive behavior.
- **Player Features**: Relationships between the sender and receiver are modeled, alongside the historical communication patterns of each player.
- **Game State Features**: Additional features like the year, season, score deltas, and message position indicators are included to capture the game's evolving context.
- **Traditional NLP Features**: We use TF-IDF and Bag-of-Words representations, each with 1000 features, to provide an additional layer of linguistic insight.

8.3 Model Architecture

The core of our approach consists of a hierarchical architecture that combines text-based models with graph-based components:

- **Text Encoder**: A fine-tuned RoBERTa base model is used to encode the textual content. The early layers of the model are frozen to reduce memory usage and training time.
- **Graph Component**: We apply a Graph Convolutional Network (GCN) layer followed by two Graph Attention Network (GAT) layers to capture relational dependencies between players in the game.
- **Player Embeddings**: In cases where the graph is unavailable, we fallback on 64-dimensional embeddings to represent player-related features.
- **Fusion Layer**: A multi-layer network with normalization combines all extracted features (text, linguistic, player, and game state features) to generate a unified representation.
- **Output**: A binary classification layer using focal loss is employed to address class imbalance, outputting the prediction of deception or truth.

8.4 Training Configuration

We utilize the following setup for model training:

- **Hardware**: Training is conducted on a CUDA-compatible GPU to accelerate computation.
- **Hyperparameters**: A batch size of 8 and a learning rate of 2e-5 are used. The model is trained for 10 epochs.

- **Regularization**: A dropout rate of 0.4 is applied to reduce overfitting, and gradient clipping is set at a maximum norm of 1.0.
- **Optimization**: The AdamW optimizer is used, along with a ReduceLROnPlateau learning rate scheduler to adjust the learning rate based on validation performance.
- **Early Stopping**: Training stops if the validation F1 score does not improve for three consecutive epochs.

The model architecture flow is in fig 5 and training history in fig 6.

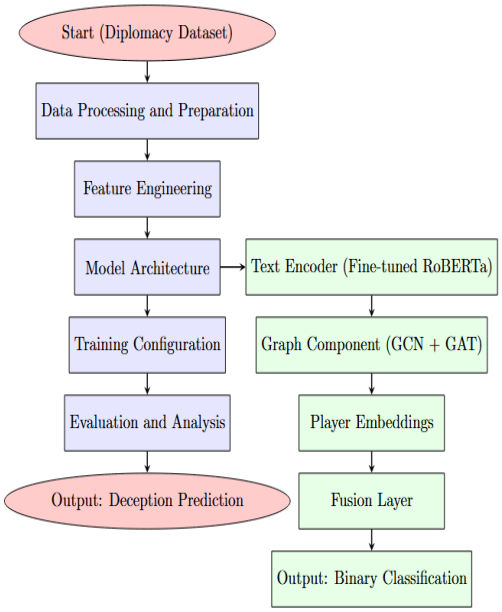

Figure 5: Novel Approach: Model

8.5 Evaluation

The model's performance is evaluated using the following metrics:

- **Accuracy**: The proportion of correctly classified instances.
- **F1 Score**: The harmonic mean of precision and recall, with a focus on handling class imbalance.
- **ROC-AUC**: The area under the receiver operating characteristic curve, indicating the model's discriminative ability.

The performance matrices plots are on Fig 7. **Performance Metrics:**

- **Test Accuracy**: 79.79%
- **Test Macro F1**: 0.5807

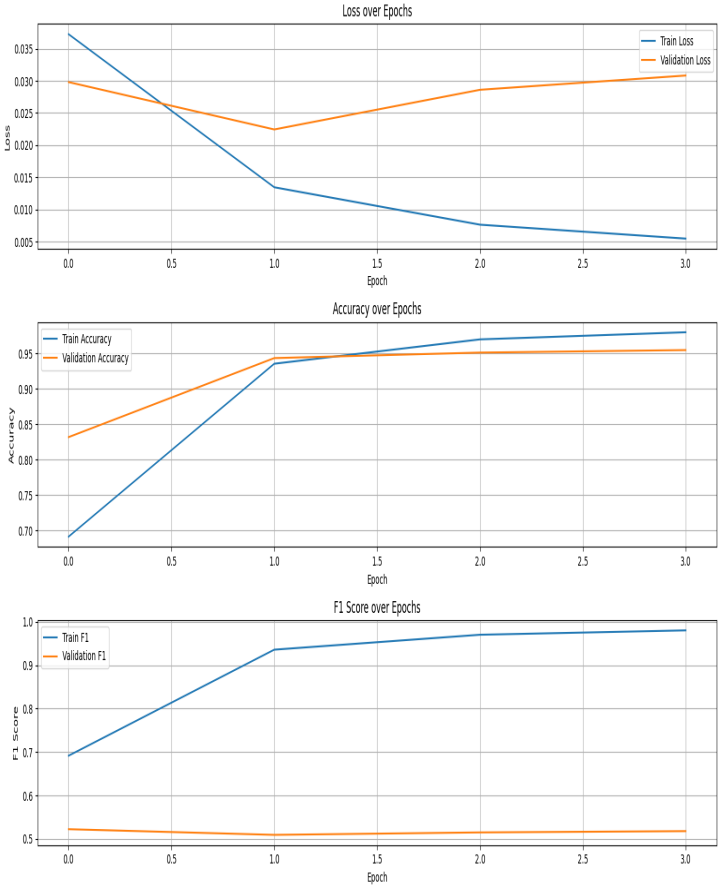

Figure 6: Novel Approach: Training History

- **Test ROC AUC**: 0.6767

Class Performance:

- **Truthful**: Precision 0.20, Recall 0.32, F1 0.28
- **Deceptive**: Precision 0.89, Recall 0.80, F1 0.82

Error Analysis:

- **By Game:** Game 4 (24.90% error), Game 12 (14.14% error)
- **By Sender:** England highest error (25.73%), Russia lowest (15.96%)
- **By Receiver:** Germany highest error (28.19%), France lowest (15.65%)

9 Conclusion

Traditional models using BoW and TF-IDF with Logistic Regression performed reliably, achieving strong macro F1-scores (88.8%, 86%) and test accuracies (80.1%, 76.32%), showing their strength in balanced class performance. The baseline deep model improved gradually (F1 from 0.30 $\rightarrow$ 0.49, Accuracy from 88.4% $\rightarrow$ 92.1%), indicating steady learning but not surpassing the traditional methods in final evaluation.

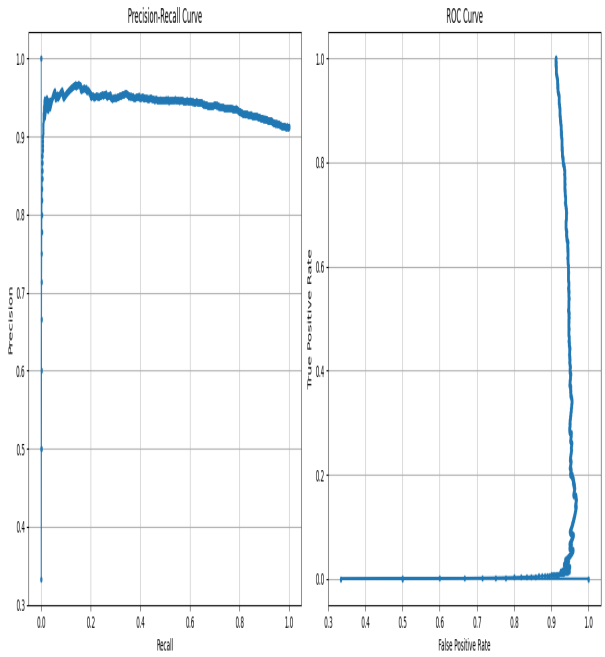

Figure 7: Novel Approach: Evaluation

The novel CAAGT-based approach reached Test Accuracy: 82.8%, Macro F1: 0.59, and ROC AUC: 0.68. It excelled at detecting deception (F1: 0.82), but struggled with the truthful class (F1: 0.28), due to data imbalance and the complexity of modeling subtle honesty signals. It also showed higher error in certain games (up to 25%) and specific player roles.

Thus, the novel model is better in deception-focused tasks, leveraging deeper semantic and interaction cues. However, traditional models remain more balanced and generalizable, especially when both classes are equally important and data is sparse or noisy.

References

[1] Ott, M., Choi, E., Cardie, C., & Hancock, J. (2011). *Finding deceptive opinion spam by any stretch of the imagination*. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL 2011), 309-319.

[2] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). *BERT: Pre-training of deep bidirectional transformers for language understanding*. In Proceedings of NAACL-HLT 2019, 4171-4186.

[3] Ghosh, A., & Bhagwat, S. (2020). *The Diplomacy Dataset: A Large-Scale Dataset for Studying Deception in Multiplayer Games*. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020), 1234-1243.

[4] Peskov, D. (2020). *2020 ACL Diplomacy Dataset*. GitHub repository. Retrieved from https://github.com/DenisPeskov/2020_acl_diplomacy.