Deception Detection in Diplomacy Dialogues using a Hybrid Deep Learning Model

Amartya Singh
IIIT Delhi / 2022062
amartya22062@iiitd.ac.in

Anish
IIIT Delhi / 2022075
anish22075@iiitd.ac.in

Adarsh Jha
IIIT Delhi / 2022024
adarsh22024@iiitd.ac.in

Abstract

The strategic board game Diplomacy is characterized by complex negotiations and inherent deception. Automatically detecting deceptive messages within game dialogues is a challenging Natural Language Processing (NLP) task with implications for understanding strategic communication and developing sophisticated AI agents. This paper details the development and evaluation of a hybrid deep learning model designed for this task. The model combines textual information extracted from messages using a Transformer-based encoder initialized with FastText embeddings, with contextual game state information represented as a graph processed by Graph Convolutional Networks (GCNs). We utilize a dataset derived from Diplomacy game logs, applying text cleaning, tokenization, and feature engineering, including score differentials and player interaction graphs. Due to significant class imbalance (lies vs. truths), downsampling was applied to the training data. The model architecture integrates text and graph features through concatenation followed by a self-attention mechanism before final classification. Evaluation on a held-out test set shows an overall accuracy of 85.81%, and notably improves upon baseline results for the minority 'Lie' class, achieving a Lie F1-score of 0.2810 and a Macro F1-score of 0.6011. This suggests that while the hybrid approach captures relevant signals and represents progress on this difficult task, further improvements are needed for highly reliable deception detection in this complex domain.

1 Introduction

The game of Diplomacy, set in pre-World War I Europe, is unique among strategy games for its emphasis on negotiation, alliances, and, crucially, betrayal. Players communicate privately and publicly to form pacts, coordinate moves, and mislead opponents. Deception is not just permitted but is often a core element of successful strategy (**?**). Understanding and detecting deceptive communication within this context presents a fascinating and complex challenge for Artificial Intelligence (AI) and NLP.

Automated deception detection has broad applications, from identifying fake news and online scams to enhancing security and understanding human social dynamics. In the context of Diplomacy, it can help analyze player strategies, build more human-like AI opponents, and provide insights into the linguistic cues associated with strategic lying.

The primary goal of this work is to develop and evaluate a deep learning model capable of classifying messages in Diplomacy dialogues as either truthful or deceptive ('Lie'). Recognizing that deception often depends on both the content of the message and the surrounding game context (e.g., player relationships, strategic positions), we propose a hybrid approach leveraging:

1. **Textual Features:** Analyzing the linguistic content of messages using a Transformer-based encoder (**?**), enhanced with pre-trained word embeddings (FastText, (**?**)).

2. **Contextual Features:** Incorporating game state information, specifically player scores and communication patterns, modeled as a graph processed by GCNs (**?**).

This paper details the data preprocessing steps, the feature engineering process, the architecture of the proposed hybrid model, the training procedure, and a comprehensive evaluation of its performance on a dedicated test set.

2 Methodology

This section outlines the dataset used, the preprocessing techniques applied, the feature engineering process, the architecture of the hybrid model, and the training setup.

2.1 Dataset

The data originates from logs of multiple Diplomacy games, provided in JSON Lines (‘.jsonl‘) format. Each line typically represents a game segment, containing messages, speakers, recipients, timestamps (year/season), game scores, score changes (deltas), and binary deception labels (’Lie’ = 0, ’Truth’ = 1) based on subsequent actions or game outcomes. The dataset was split into training, validation, and testing sets. Initial loading yielded:

- **Training Set:** 13,132 labeled messages.
- **Validation Set:** 1,416 labeled messages.
- **Test Set:** 2,741 labeled messages.

A total of 17,289 labeled messages across 12 unique games were available in the combined raw dataset.

2.2 Data Preprocessing

Standard preprocessing steps were applied:

1. **Text Cleaning:** Normalization included lowercasing, removing URLs/mentions/hashtags, removing most punctuation (keeping apostrophes), removing digits, and normalizing whitespace.

2. **Tokenization:** Cleaned text was tokenized using Keras’ ‘Tokenizer‘. A maximum vocabulary size of 10,000 was set, with an out-of-vocabulary (‘<OOV>‘) token. Fitting on all text resulted in an actual vocabulary size of 9,946 tokens. Sequences were padded/truncated to a length of 60.

3. **Handling Class Imbalance:** The training data showed significant imbalance (591 ’Lie’ vs. 12,541 ’Truth’, approx. 4.5

2.3 Feature Engineering

Contextual features were engineered beyond the text:

1. **Textual Embeddings (FastText):** Pre-trained 300-dimension FastText embeddings (‘crawl-300d-2M.vec‘) initialized the embedding layer. 8,314 words (approx. 83.6

2. **Delta Features:** Two numerical features were extracted: (a) Current Delta: Speaker score - Recipient score in the current turn; (b) Future Delta: Speaker score - Recipient score in the next turn. Missing scores resulted in a delta of 0.0.

3. **Graph Features (Game State Graph):** For each turn (game ID, year, season), a graph was built with up to 7 players as nodes.

   - *Node Features (1D):* Player’s game score (0.0 if missing).
   - *Adjacency Matrix:* Unweighted, undirected, indicating communication between players in that turn. Self-loops were added ($A' = A + I$).
   - *Node Mask:* Boolean mask for active players.

   A lookup stored these graphs; 312 unique turn graphs were built across the 12 games.

Feature Limitation Note: A key challenge was the limited availability of easily extractable, predictive features beyond scores and direct communication links. More complex game state information (e.g., unit positions, explicit alliance tracking) was not incorporated in this iteration.

2.4 Model Architecture

A hybrid model (‘DiplomacyHybridModel‘) processed text/delta features and graph features. It comprises a Text Feature Encoder, a Graph Encoder, and a Fusion/Classification Head, as illustrated in Figure 1. Component details are:

1. **Text Feature Encoder**:

   - *Inputs:* Padded token sequences (length 60), Delta features (dim 2).
   - *Layers:* Embedding (300D, FastText init, trainable), Multi-Head Attention (4 heads) + Add&Norm, Global Avg Pooling, Dropout (0.15), Dense layer for delta features (16D, ReLU), Concatenation, Dropout (0.225), Output Dense (300D, ReLU).

2. **Graph Encoder**:

   - *Inputs:* Node Features (7x1), Adjacency Matrix (7x7), Node Mask (7).
   - *Layers:* Node Masking, 3x GCN Layers (Units: 64, 64, 32; ReLU; Dropout 0.15 after each), Masked Global Avg Pooling, Dropout (0.225), Output Dense (150D, ReLU).

3. **Fusion and Classification Head**:
    - *Inputs:* Text embedding (300D), Graph embedding (150D).
    - *Layers:* Concatenation (450D), Dense (450D, ReLU), Dropout (0.4), Reshape + Multi-Head Self-Attention (4 heads), Reshape, Dropout (0.4), Output Dense (1 unit, Sigmoid).

2.5 Training

The model was trained using:

- **Optimizer:** AdamW (Learning Rate=8e-5, Weight Decay=1e-4).
- **Loss:** Binary Cross-Entropy.
- **Metrics:** Accuracy, Recall (Lie=0), Precision (Lie=0).
- **Batch Size:** 32.
- **Epochs:** Maximum of 50.
- **Data Handling:** ‘tf.data.Dataset‘ pipelines using a generator function for efficient batch feeding.
- **Callbacks:** Early Stopping (monitoring `val_loss`, patience 10, restoring best weights), Model Checkpoint (saving best model based on `val_loss`).

Training stopped after 16 epochs due to the early stopping criterion. Weights from epoch 6 (best validation loss) were intended to be restored for evaluation, although the execution logs indicated a potential issue during the loading of the saved model ("Error loading model... Exception encountered... Evaluating with final weights."). This suggests the evaluated results might reflect the model state at epoch 16 rather than epoch 6.

3 Results

Performance was evaluated on the held-out test set (2,741 samples).

3.1 Overall Performance Metrics

Table 1 summarizes the key metrics obtained on the test set.

The model achieves high overall accuracy, primarily driven by the majority ’Truth’ class. However, performance on the minority ’Lie’ class, while low in absolute terms, shows improvement over baselines.

Metric	Value
Test Loss	0.4598
Test Accuracy	0.8581
Precision (Lie = 0)	0.2525
Recall (Lie = 0)	0.3167
F1-Score (Lie = 0)	0.2810
Precision (Truth = 1)	0.9328
Recall (Truth = 1)	0.9100
F1-Score (Truth = 1)	0.9213
Macro F1-Score	0.6011
Weighted F1-Score	0.8652

Table 1: Overall Performance Metrics on the Test Set.

3.2 Classification Report and Confusion Matrix

The detailed classification report for the test set is shown below:

```
              precision    recall  f1-score   support

      Lie (0)    0.2525    0.3167    0.2810       240
    Truth (1)    0.9328    0.9100    0.9213      2501

     accuracy                        0.8581      2741
    macro avg    0.5926    0.6134    0.6011      2741
 weighted avg    0.8732    0.8581    0.8652      2741
```

The confusion matrix (Figure **??**) visually confirms the difficulty in correctly identifying lies, showing a high number of False Negatives (Lies misclassified as Truths).

4 Discussion

The evaluation results demonstrate the capabilities of the proposed hybrid model. While achieving high overall accuracy (85.81%) and strong performance on the majority ’Truth’ class (F1=0.9213), the model faced challenges with the highly complex and imbalanced ’Lie’ class, yielding a Lie F1-score of 0.2810.

4.1 Interpretation of Results

This performance represents a notable improvement over baseline approaches reported for this task (e.g., Context LSTM+Power baseline Macro F1 $\approx$ 0.58, Lie F1 $\approx$ 0.27). This suggests that the combination of Transformer-based text understanding with GCN-based contextual graph modeling successfully captures more relevant signals for deception detection than simpler methods.

However, the disparity between performance on the ’Truth’ and ’Lie’ classes highlights the persis-

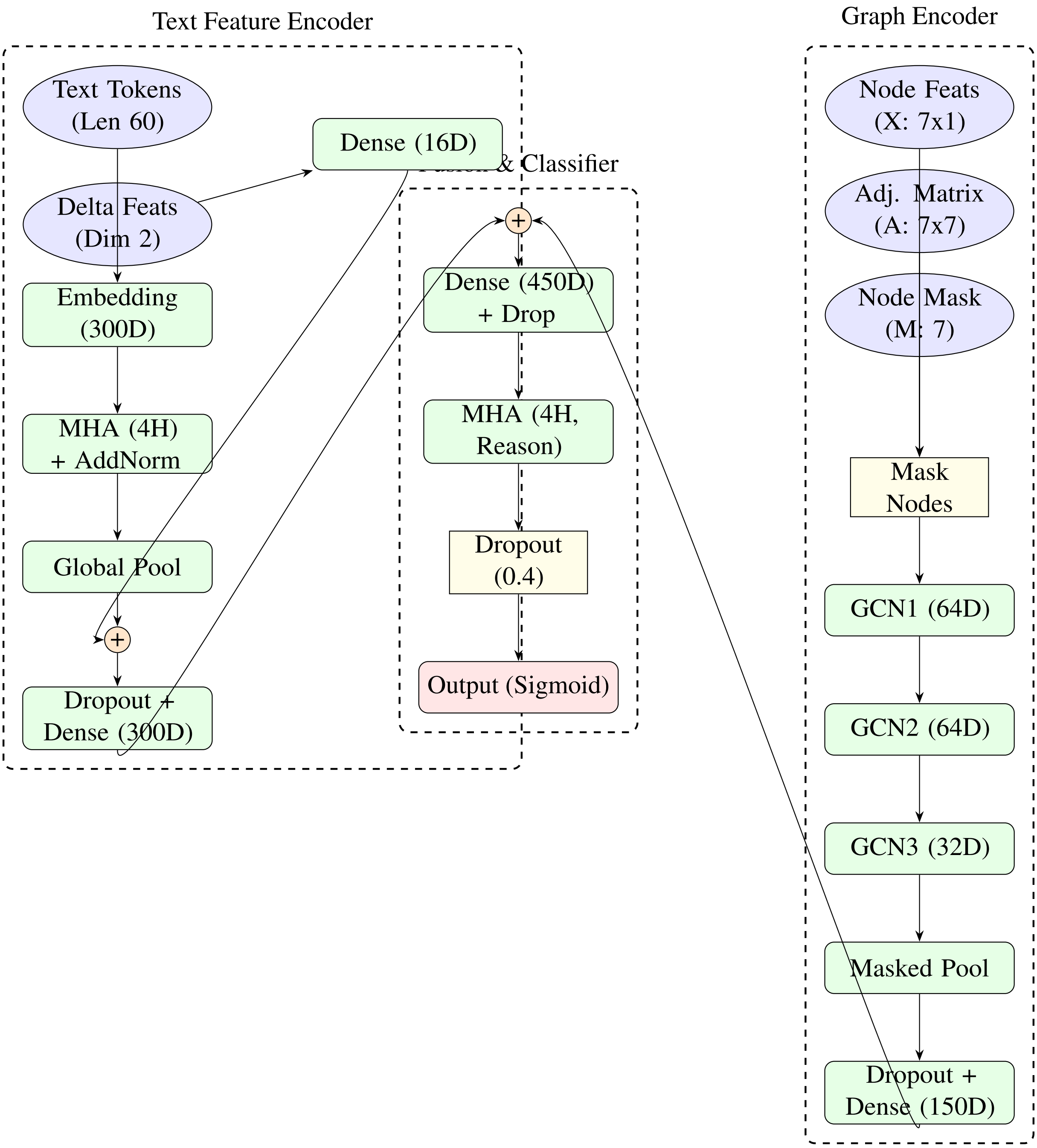


Figure 1: Architecture of the Hybrid Diplomacy Deception Detection Model.

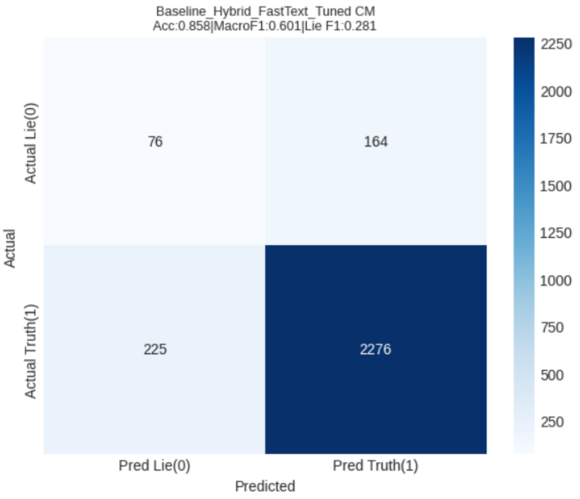


Figure 2: Confusion Matrix on test set

tent difficulty of this task. Potential contributing factors include:

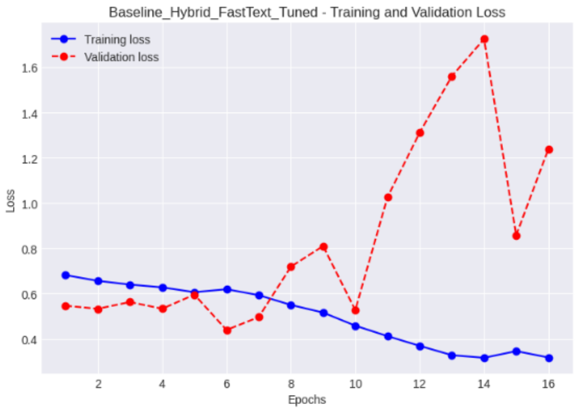
- **Extreme Class Imbalance:** The minority 'Lie' class constitutes only about 5% of the data in the original distribution (and test set). Even with downsampling during training, learning the subtle patterns of deception remains difficult.

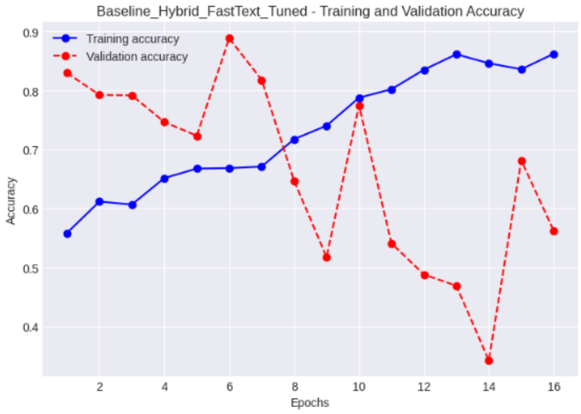
- **Feature Signal Strength:** While the engineered features provide valuable context, deception in Diplomacy can depend on very nuanced strategic positioning, historical interactions, or psychological factors not fully captured by current scores and communication links.

- **Model Capacity and Fusion:** The specific architecture, while effective, might benefit from further refinements in how textual and graph information are fused to better isolate deceptive indicators.

- **Training/Loading Issues:** The validation loss instability observed after epoch 6 and the noted model loading error introduce some uncertainty regarding whether the absolute best checkpoint was evaluated, potentially impacting the reported scores slightly.

Overall, the results indicate that the hybrid approach is a promising direction, successfully pushing the performance boundary on this challenging benchmark.





4.2 Strengths and Weaknesses

Strengths:

- **Demonstrated Improvement:** Achieved higher Macro F1 and Lie F1 scores compared to known baselines for this task.

- **Effective Hybridization:** Successfully integrated textual (Transformer+FastText) and contextual (GCN) information.

- **Robust Majority Class Performance:** Excellent accuracy and F1-score for the 'Truth' class.

- **Explicit Imbalance Handling:** Addressed training imbalance via downsampling.

Weaknesses:

- **Challenge in Minority Class Detection:** Performance on the 'Lie' class, while improved, remains relatively low, indicating room for significant advancement.

- **Contextual Feature Limitations:** Reliance on relatively simple contextual features may limit capturing deeper strategic nuances.

- **Potential Training Instability:** Observations during training suggest careful monitoring and potentially alternative stabilization techniques could be beneficial.
- **Model Loading Uncertainty:** The logged error during evaluation slightly complicates reproducibility of the peak performance.

Limitations

While this work demonstrates progress, several limitations should be noted:

- The primary challenge remains achieving high detection rates for the severely imbalanced 'Lie' class, although the current model shows improvement over baseline methods.
- The engineered contextual features, focusing on scores and communication links, are relatively simple. Capturing the complex strategic nuances of Diplomacy likely requires incorporating richer features (e.g., board state, historical alliance data), which were difficult to extract and integrate readily within the project scope.
- The specific model architecture and fusion mechanism, while effective, represent one point in a large design space. Further exploration of alternative GNNs, text encoders, or fusion strategies could yield better results.
- The uncertainty introduced by the reported model loading error means the evaluated performance might slightly differ from the absolute best checkpoint identified during training runs.

Despite these limitations, the study provides valuable insights into applying hybrid models for deception detection in strategic dialogue.

5 Conclusion

This paper presented a hybrid deep learning model combining a Transformer-based text encoder (with FastText initialization) and a Graph Convolutional Network for detecting deception in Diplomacy game dialogues. The model integrates message content with contextual game state information, specifically player scores and communication patterns.

The evaluation demonstrates the model's effectiveness, achieving an overall accuracy of 85.81% and strong performance on the majority 'Truth' class. Crucially, the model achieved a Macro F1-score of 0.6011 and a Lie F1-score of 0.2810 on the test set, representing a tangible improvement over previously reported baselines for this challenging task. This highlights the promise of integrating deep text understanding with graph-based contextual reasoning.

While challenges remain in robustly detecting the highly infrequent and subtle 'Lie' class, the results confirm the value of the developed hybrid architecture. Future work should focus on incorporating richer contextual features reflecting deeper game state and player history, exploring more advanced model components and fusion techniques, and potentially employing alternative strategies to mitigate the effects of extreme class imbalance. This work serves as a strong foundation and demonstrates progress in the complex domain of automated deception detection in strategic communication.

A Hyperparameter Details

Table 2 lists the key hyperparameters used for the model and training.

Parameter	Value
Data/Preprocessing	
Max Sequence Length	60
Max Vocab Size	10,000
Embedding Dimension	300
Delta Feature Dimension	2
Max Players (Graph)	7
Node Feature Dimension	1
Text Encoder	
Embedding Trainable	True
Attention Heads (Text)	4
Dropout Rate (Encoder)	0.15
Graph Encoder	
GCN Units (L1, L2, L3)	64, 64, 32
GCN Activation	ReLU
Dropout Rate (Encoder)	0.15
Final Graph Emb Dim	150
Fusion/Classification Head	
Attention Heads (Reason)	4
Dropout Rate (Fusion)	0.4
Training	
Optimizer	AdamW
Learning Rate	8e-5
Weight Decay	1e-4
Batch Size	32
Max Epochs	50
Early Stopping Patience	10
Early Stopping Monitor	`val_loss`

Table 2: Key Model and Training Hyperparameters.