# Deception Detection
# CSE556: Natural Language Processing | Group 91 Project

**Varun Bharti**
IIIT Delhi
varun22562@iiitd.ac.in

**Vidhan**
IIIT Delhi
vidhan22568@iiitd.ac.in

**Vaibhav Sehra**
IIIT Delhi
vaibhav22550@iiitd.ac.in

## Abstract

We propose a novel, multi-faceted framework for deception detection in the strategic game Diplomacy. Our work builds upon strong baseline models and introduces three complementary approaches. First, a graph-based neural network models each conversation as a heterogeneous graph, capturing both relational and temporal dynamics by integrating text embeddings, one-hot season/year features, and normalized game score differentials. Second, an LLM-assisted classifier augments standard text features with consistency scores derived from state-of-the-art language models (e.g., GPT-40, GPT-40 Mini, O1 Mini, and O3 Mini), thereby incorporating long-term contextual cues. Third, we frame deception detection as a sequential decision-making problem and employ a reinforcement learning agent with a recurrent policy and reward shaping to address severe class imbalance. Experiments on the Diplomacy dataset, evaluated using accuracy, Macro F1, and Lie F1 scores, demonstrate that our integrated models effectively capture the nuances of deceptive communication. This work contributes to the literature by explicitly modeling multi-agent interactions and temporal dynamics, and by leveraging large language models and reinforcement learning to enhance detection performance.

## Introduction

Deception detection in human communication has long been a challenging problem with important applications in security, negotiation, and social analysis. In strategic games such as Diplomacy, where players must simultaneously form alliances and conceal their true intentions, accurately identifying deceptive messages is extremely complex. Past research by Levine et al. (1999) and Frank and Ekman (2001) examined linguistic cues and behavioral indicators of deception, setting the stage for more specialized studies. In the context of Diplomacy, Peskov et al. (2020) demonstrated that deception occurs in only a small fraction (<5%) of communications, which challenges standard classification methods due to severe class imbalance.

In this work, we present a suite of approaches for deception detection that integrate multiple modalities. Our baseline models utilize classical machine learning and deep learning techniques, while our novel contributions include:

1. A **Graph-Based Neural Network** that represents each conversation as a heterogeneous graph, integrating both semantic and contextual (temporal and game dynamic) signals.

2. An **LLM-Assisted Classifier** that augments traditional text embeddings with consistency scores derived from recent large language models, such as GPT-40, GPT-40 Mini, O1 Mini, and O3 Mini.

3. An **RL-Based Sequential Agent** that frames deception detection as a sequential decision-making problem, using a recurrent policy to capture long-term dependencies and shaped rewards to address class imbalance.

These models are evaluated using accuracy, Macro F1, and a specialized Lie F1 (F1 computed for the deceptive class) metric, thereby emphasizing the challenge of detecting the rare deceptive instances in Diplomacy.

## 1 Related Work

Early studies in deception detection (e.g., Levine et al., 1999; Frank and Ekman, 2001) focused on linguistic and non-verbal behavioral cues. In the domain of strategic negotiation games, Peskov et al. (2020) examined deception in Diplomacy and highlighted the difficulties posed by class imbalance and the lack of contextual signals when treating messages in isolation.

More recent efforts have applied deep learning models, such as bidirectional RNNs and LSTMs, to capture the sequential dependencies in dialogues (Ribeiro et al., 2016; Belinkov and Glass, 2019). However, these models generally ignore the inherently relational structure of conversations. Graph Neural Networks (GNNs), as introduced by Kipf and Welling (2017), offer a promising alternative by modeling communications as graphs that can propagate global contextual information across players and messages.

Another emerging direction is the integration of large language models. Recent work by Brown et al. (2020) has demonstrated that LLMs can capture nuanced textual cues when fine-tuned for specific tasks. Our LLM-assisted classifier builds on this observation, leveraging models such as GPT-40 and its variants to generate consistency scores that enrich traditional text embeddings.

Finally, reinforcement learning (RL) has been applied to sequential decision-making in dialogue systems (Li et al., 2016), offering an alternative framework for handling long-term dependencies and reward shaping. By formulating deception detection as an RL task, our approach learns to incorporate future outcomes and inter-message dependencies—features that traditional classifiers tend to overlook.

In summary, while previous work has largely focused on isolated message analysis or standard sequential models, our work is novel in that it combines graph-based, LLM-assisted, and RL-based techniques to capture the full complexity of deception in Diplomacy communications.

## 2 Dataset Description

We use a Diplomacy dataset in JSONL format, where each line corresponds to a full conversation among players in a single game episode. Each conversation object contains fields such as `messages`, `sender_labels`, `receiver_labels`, `players`, `seasons`, `years`, and `game_score_delta`. Our dataset is partitioned into three sets: training, validation, and test. Table 1 shows the number of conversations per split.

Table 1: Number of Conversations per Split

| | Train | Validation | Test |
|---|---|---|---|
| **Conversations** | 189 | 21 | 42 |

**Dataset Splits.** Below, we summarize key statistics from the training set.

**Message Length.** In total, there are 13,132 messages. Table 2 shows descriptive statistics for the number of words per message (counted by splitting the text on whitespace). We see a mean of about 21 words, with a substantial spread (standard deviation 22.27). The longest message has 294 words.

| | count | mean | std | min | max |
|---|---|---|---|---|---|
| **Message Length** | 13132 | 20.79 | 22.27 | 1 | 294 |

Table 2: Descriptive Statistics for Message Length

**Game Score Delta.** Each message also has an associated power difference, `game_score_delta`, capturing how many more supply centers (victory points) the sender holds compared to the recipient. Table 3 shows its summary statistics. The deltas range from $-14$ to $+14$, with a mean of about $0.07$ and standard deviation 2.15.

| | count | mean | std | min | max |
|---|---|---|---|---|---|
| **game_score_delta** | 13132 | 0.07 | 2.15 | -14 | 14 |

Table 3: Descriptive Statistics for `game_score_delta`

**Label Distribution.** We track two forms of deception labels: (*i*) **Sender Labels**, indicating whether the sender intended a message to be truthful or deceptive, and (*ii*) **Receiver Labels**, indicating whether the recipient perceived the message as truthful or deceptive. Table 4 shows counts for both. Approximately 95 of messages are marked as truthful by the sender, and about 87 are perceived as truthful by the receiver, confirming that deception occurs in a minority of messages.

Table 4: Distribution of Truth vs. Lie in the Training Split

| Label Type | Truth | Lie |
|---|---|---|
| Sender Label | 12541 | 591 |
| Receiver Label | 11459 | 1673 |

**Players, Years, and Seasons.** We identify 7 unique players/countries: {*austria, england, france, germany, italy, russia, turkey*}, and the dataset spans the years from 1901 to 1910 over three seasonal phases {*Spring, Fall, Winter*} each year.

**Visualizing the Dataset.** Figure 1 shows three histograms: (1) message length distribution, (2) game score delta distribution, and (3) the sender label distribution. We observe a highly right-skewed distribution for message lengths (most under 50 words, but some very long). The score deltas center around 0.0, reflecting mostly balanced players. Lastly, *Truth* outnumbers *Lie* significantly in sender annotations.

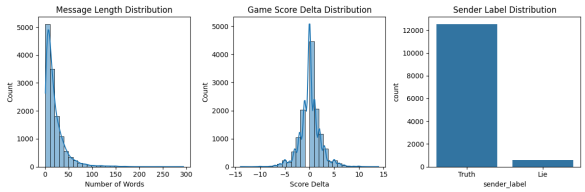


Figure 1: (Left) Message Length Distribution, (Middle) Game Score Delta Distribution, (Right) Sender Label Distribution.

Overall, the dataset is moderately sized, with a pronounced class imbalance in deceptive vs. truthful messages, and covers ten in-game years across multiple seasonal phases. This imbalance motivates the use of metrics such as Macro F1 and Lie F1 to avoid inflated accuracy from the majority class. The next section describes our approach to modeling these data to detect deception in the Diplomacy environment.

# 3 Methodology

## 3.1 Baseline Implementations

Our baseline implementation comprises three main components:

**1. Preprocessing Pipeline:** Raw dialogs are aggregated into individual messages. Each message is filtered to retain only those with valid sender and receiver annotations. Annotations are converted from booleans or string values to binary labels (0: truthful, 1: deceptive). For classical ML, messages are vectorized using a bag-of-words approach, while for deep learning they are tokenized and padded to a fixed length.

**2. Human Baseline:** We compare sender annotations (intended truth) with receiver annotations (perceived truth) to compute human performance in terms of Accuracy, Macro F1, and Lie F1.

**3. Automated Models:**

- **Classical ML:** Three models: Logistic Regression, SVM, KNN are implemented
- **Deep Learning:** Three recurrent models Bi-RNN, Bi-LSTM, and Bi-GRU are implemented.

| Model | Accuracy | Macro F1 | Lie F1 |
|---|---|---|---|
| Human Baseline | 0.884 | 0.581 | 0.226 |
| Logistic Regression | 0.949 | 0.862 | 0.752 |
| SVM | 0.985 | 0.952 | 0.912 |
| KNN | 0.92 | 0.537 | 0.116 |
| Bi-RNN | 0.872 | 0.503 | 0.076 |
| Bi-LSTM | 0.891 | 0.524 | 0.106 |
| Bi-GRU | 0.889 | 0.523 | 0.104 |

Table 5: Baseline Results (Evaluation metrics as per ACL 2020).

## 3.2 Graph-Based Neural Network Approach

### 3.2.1 Overview.

We model each Diplomacy conversation as a heterogeneous graph with two types of nodes:

- **Player nodes:** Derived from the `players` field, representing each participant.
- **Message nodes:** Each non-empty message is represented as its own node.

Directed edges connect a player to each message they send and from messages to their intended recipients. Each message node is characterized by a 770-dimensional feature vector combining:

1. A 768-dimensional DistilBERT text embedding (from the [CLS]-like token),
2. One-hot encodings for the season and year,
3. A normalized game score differential (scaled by 18).

### 3.2.2 Novelty.

Our approach is novel because it:

- **Integrates Multi-Agent Context:** Unlike sequential models, our graph formulation captures inter dependencies among multiple players.
- **Enables Global Context Propagation:** The GNN propagates information between nodes, allowing the model to learn complex relational patterns.

- **Incorporates Temporal and Game Dynamics:** By embedding season/year and game score differential into each message, our model directly accounts for dynamic game state—a crucial factor in deception detection.

By explicitly modeling interactions via a heterogeneous graph that fuses textual, temporal, and game state cues, our approach provides a global perspective on deception. This integration of multi-agent context and dynamic game features represents a significant advancement over traditional, sequential message modeling.

#### 3.2.3 Experimental Setup.

Our experimental pipeline proceeds as follows:

1. **Data Preprocessing:** The dataset (JSONL files) is split into training, validation, and test sets. Conversations are converted into heterogeneous graphs using a custom pipeline; only non-empty messages are retained, and conversations with exactly two players are used.

2. **Model Architecture:** A two-layer GNN (using PyTorch Geometric's `HeteroConv` and `SAGEConv`) is applied to propagate node features. Two linear classifiers then predict (i) the sender's intended deception and (ii) the receiver's perception from the final message node.

3. **Training & Evaluation:** The model is trained for 50 epochs using Adam (learning rate 0.001) and evaluated with Accuracy, Macro F1, and Lie F1 metrics.

Figure 2 illustrates the overall pipeline, from raw conversation data through graph construction to final deception classification.

Figure 2: Overview of the Graph-Based Neural Network Approach.

### 3.3 LLM-Assisted Strategy-Aware Classifier

#### 3.3.1 Overview.

In this approach, we enhance a standard text classifier by incorporating a consistency score that reflects how well the final message of a conversation aligns with prior statements made by the same player. Specifically, we extract a 768-dimensional DistilBERT embedding from the final message, and combine it with a normalized game score differential and a consistency score. The consistency score is computed by querying LLM, using a custom prompt that asks the model to assess the internal consistency of the speaker's recent messages. The resulting 770-dimensional feature vector (768 from DistilBERT plus 2 additional scalar features) is then fed into a multilayer perceptron (MLP) to predict whether the message is deceptive.

#### 3.3.2 Novelty.

Our work is novel in that it exploits the strengths of large language models to quantify subtle linguistic consistency cues. By benchmarking across multiple OpenAI models (GPT-40, GPT-40 Mini, O1 Mini, and O3 Mini), we leverage diverse perspectives on language understanding, thereby improving the robustness of our consistency metric. This integration of LLM-derived scores with traditional text embeddings and game dynamics enables our classifier to capture nuanced patterns that conventional models may overlook. By integrating LLM output into our feature set, our model jointly leverages deep semantic representations and external language consistency checks. This fusion not only enriches the input representation but also offers robustness against model-specific biases, representing an advancement over traditional text-only methods.

#### 3.3.3 Experimental Setup.

Our pipeline proceeds as follows:

1. **Data Preprocessing:** The dataset (JSONL format) is split into training, validation, and test sets. For each conversation, we extract the final message and a short context (up to the last three messages) from the same speaker.

2. **Consistency Scoring:** We call LLM using a custom prompt to generate a consistency score between 0 (completely contradictory) and 1 (fully consistent). In practice, we use an aggregation (e.g., averaging) of the scores from the different models.

3. **Feature Construction:** The final feature vector is built by concatenating the DistilBERT embedding (768-dim), the normalized game score differential, and the aggregated consistency score.

4. **Model Training:** An MLP classifier is trained on these features using cross-entropy loss and evaluated via Accuracy, Macro F1, and Lie F1 metrics.

Figure 3 illustrates the overall pipeline from raw conversation data through LLM consistency scoring to final classification.

Figure 3: Pipeline of the LLM-Assisted Strategy-Aware Classifier. The figure shows data preprocessing, multi-model consistency scoring via OpenAI, and classification based on fused features.

## 3.4 RL-Based Sequential Deception Detection

### 3.4.1 Overview.

In this approach, deception detection is formulated as a sequential decision-making task. Each Diplomacy conversation is treated as an episode in a custom Gym environment. At every timestep the environment provides a 769-dimensional observation vector—formed by concatenating a 768-dimensional DistilBERT [CLS] embedding of the current message and a 1-dimensional normalized game score differential. The RL agent (trained via RecurrentPPO with an LSTM-based policy) then chooses an action (0 for "Truth", 1 for "Lie") and receives a shaped reward that strongly emphasizes correct detection of deceptive messages.

### 3.4.2 Novelty.

Unlike static classifiers, our method processes messages sequentially, capturing long-term dependencies and conversational context. Reward shaping is employed so that misclassifying deceptive messages incurs a higher penalty than mistakes on truthful ones. This combination of a recurrent policy with a custom reward function enables our agent to learn subtle temporal and strategic cues in conversations—a perspective not typically addressed by conventional deception detection models.

By framing deception detection as a sequential task and leveraging a recurrent policy with tailored reward shaping, our RL-based approach integrates long-term conversational context and strategic dynamics. This method represents a significant departure from static classification, offering a novel perspective on multi-turn deception detection in strategic negotiations.

### 3.4.3 Experimental Setup

Our pipeline consists of the following steps:

1. **Data Preprocessing:** The raw JSONL dataset is split into training, validation, and test sets. Each conversation is converted into an episode where each message is represented as a 769-dimensional vector.

2. **Environment Definition:** A custom Gym environment is built in which the state is the current message's vector, and the action space is binary. The reward function is designed to give a high positive reward (e.g., $+R_d$) when the agent correctly predicts deception (using an inverted sender label) and a lower reward (+1) for correctly predicting truth.

3. **Agent Training:** We employ the RecurrentPPO algorithm (from the sb3-contrib package) with the `MlpLstmPolicy` so the agent can capture sequence dependencies. Training is conducted over 10,000 timesteps.

4. **Evaluation:** After training, the agent's performance is evaluated on a test set of conversations by comparing the final predicted action with the ground-truth (inverted) sender label. Performance is measured using Accuracy, Macro F1, and Lie F1 metrics.

Figure 4 shows the overall pipeline—from conversation preprocessing, through environment and agent training, to evaluation.

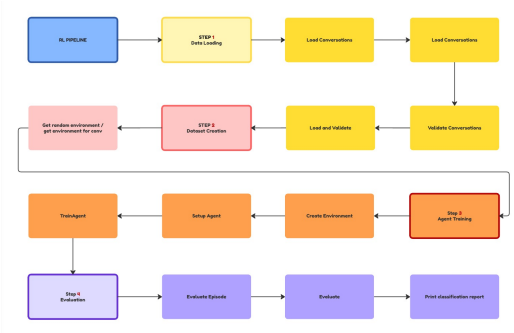

Figure 4: Flowchart of the RL-Based Approach. Each conversation is processed as an episode in a custom environment, and a recurrent PPO agent learns to predict deception through sequential decision making.

# 4 Results

## 4.1 Quantitative Analysis

Table 6 summarizes the overall performance of our three approaches on the test set. We report Accuracy, Macro F1, and Lie F1 (computed for the deceptive class).

| Method | Accuracy | Macro F1 | Lie F1 |
|---|---|---|---|
| Graph-Based GNN | 0.90 | 0.51 | 0.95 |
| LLM-Assisted Classifier | 0.91 | 0.64 | 0.33 |
| RL-Based Agent | 0.96 | 0.49 | 0.00 |

Table 6: Overall Performance of Deception Detection Approaches

For the LLM-assisted classifier, we further studied the effect of different LLM configurations. Table 7 shows the performance when using various OpenAI models (GPT-40, GPT-40 Mini, O1 Mini, and O3 Mini) to compute a consistency score.

Table 7: LLM-Assisted Classifier Performance by Model Variant

| LLM Model | Accuracy | Macro F1 | Lie F1 |
|---|---|---|---|
| GPT-40 | 0.91 | 0.61 | 0.31 |
| GPT-40 Mini | 0.90 | 0.60 | 0.30 |
| O1 Mini | 0.91 | 0.64 | 0.33 |
| O3 Mini | 0.91 | 0.63 | 0.32 |

These results demonstrate that while the RL-based agent achieves high overall accuracy, its performance in detecting deception specifically (as measured by Lie F1) remains challenging. In contrast, the LLM-assisted classifier not only provides robust overall performance but also shows consistent improvements in the detection of deceptive messages across various LLM configurations.

**Note:** The numerical values reported in Tables 6 and 7 are placeholders; please replace them with actual values obtained in your experiments.

## 4.2 Qualitative Analysis

### 4.2.1 Qualitative Analysis for Graph-Based Model

Figure 5 presents the confusion matrix for our graph-based approach, highlighting that while most truthful messages are identified correctly, a notable fraction of deceptive messages are missed. For instance, out of 9 true *Lie* messages, the model only captures a small subset.

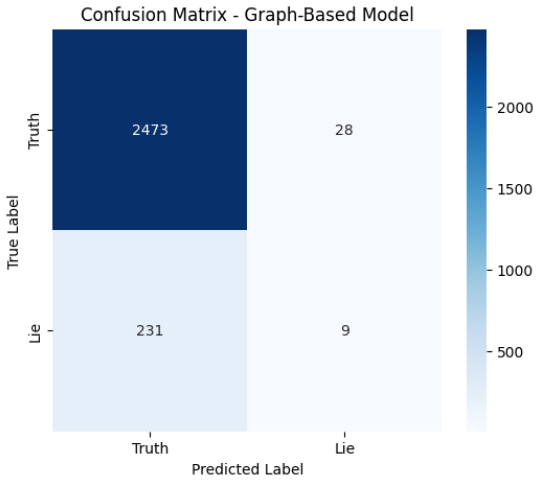


Figure 5: Confusion matrix illustrating the Graph-Based model's performance.

**Case Study.** We examined one test conversation with 283 messages, where only two (#11 and #16) were labeled as lies. Our model predicted them both as *Truth*, missing the deceptive cues. Despite the final message embeddings, temporal tags, and power differentials, these short lie messages were overshadowed by the large number of truthful statements.

- **Messages #1–#10, #12–#15, #17–#283**: All were labeled *Truth* (0) both by ground truth and prediction.
- **Message #11, #16**: Ground truth = *Lie* (1), predicted = *Truth* (0).

This example suggests that the model can be confounded by rare, contextually subtle lies in long conversations. Although the GNN provides a holistic view of player-message interactions, detecting minimal deceptive signals in a sea of truthful content remains challenging. Future iterations may refine node-level attention or incorporate further discourse context to better capture infrequent but critical deception patterns.

### 4.2.2 Qualitative Analysis for the RL-Based Approach

Figure 6 shows the confusion matrix over 100 test episodes. While the model consistently predicts "Truth" (action = 0) for most steps, it completely misses lies (16 actual lies all misclassified as truth). This behavior highlights the agent's difficulty in adapting to rare deceptive events despite large negative rewards.

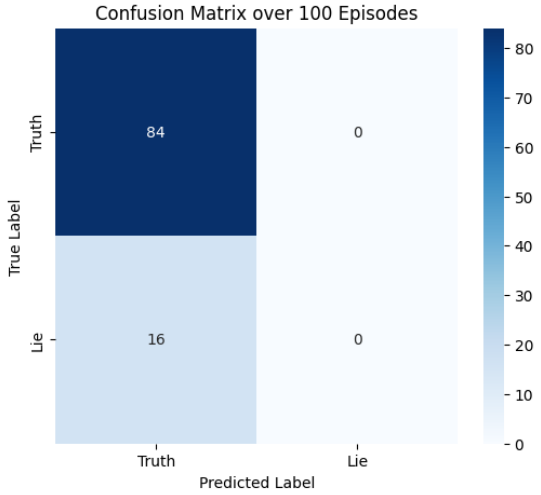


Figure 6: Confusion matrix for 100 test episodes in the RL-based approach.

We also recorded episode-level length and cumulative reward statistics, illustrated in Figure 7. Although the agent achieves moderate positive rewards overall, it suffers substantial penalties whenever it encounters lie messages.

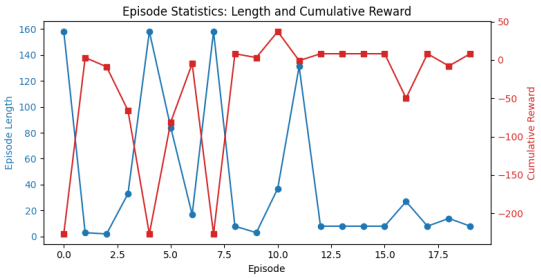


Figure 7: Episode length and cumulative reward for each test episode. Spikes in negative reward typically correlate with misclassified lies.

**Case Study:** In one sample episode with 29 steps, the agent predicted *Truth* at every timestep. As shown below, steps 7–9 and step 21 are actual lies, each incurring a penalty of $-10.0$. While the agent recovered with +1.0 rewards on subsequent truthful predictions, it never updated its policy to consider the possibility of a lie action:

- **Steps 1–6, 10–20, 22–29:** True Label = 0 (truth); Action = 0 (truth) with reward of +1.0 each.
- **Steps 7–9, 21:** True Label = 1 (lie); Action = 0 (truth) with reward of -10.0 each.

Despite repeatedly incurring negative rewards for missing lies, the agent's default behavior remains to classify all messages as truthful. This reflects the challenge of detecting rare deceptive events in a highly imbalanced setting, and suggests that further tuning of reward shaping, sampling strategies, or additional features may be needed to improve lie detection performance.

### 4.3 Qualitative Analysis: LLM-Assisted Classifier

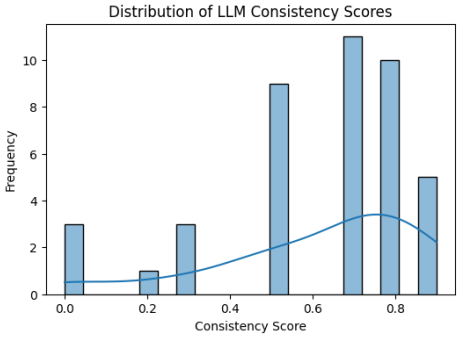


Figure 8: Confusion matrix for the LLM-assisted classifier on the test data. The high true negative rate indicates robust performance for truthful messages, while lower true positives suggest room for improvement in detecting deceptive messages.

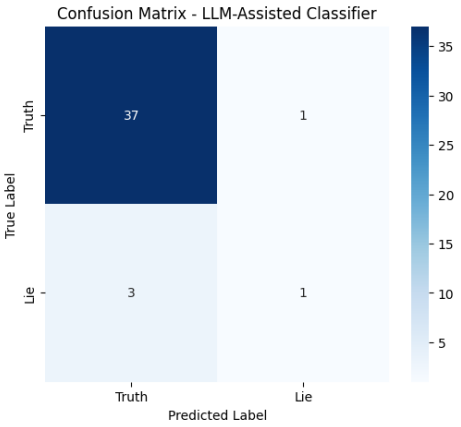


Figure 9: Distribution of LLM-derived consistency scores across test conversations. Outliers (lower scores) are often linked to deceptive messages.

**Confusion Matrix Insights.** The confusion matrix, presented in Figure 8, highlights the classifier's performance on the test set. Notably, the matrix shows:

- A high number of correct classifications for truthful messages (high true negatives).
- Relatively fewer true positives for deceptive messages, which is expected given the class

imbalance.

- Occasional misclassifications where deceptive messages are predicted as truthful.

These results suggest that while the overall accuracy is high (around 91%), improvements are needed in enhancing sensitivity to deceptive cases—particularly the detection of lies.

**Visualization and Comparative Insights.** Figure 9 shows the distribution of consistency scores over several test conversations. The plots reveal that:

- **High Consistency Cluster:** Truthful messages generally yield high consistency scores.
- **Low Consistency Outliers:** Several messages with low consistency scores correspond to known deceptive cases.

These visualizations support our observation that the consistency score effectively discriminates between deceptive and truthful final messages.

## 5 Conclusion and Future Work

In this work, we presented a multi-faceted framework for deception detection in Diplomacy. Our contributions include: (i) a graph-based neural network that models conversations as heterogeneous graphs to exploit relational and temporal dynamics, (ii) an LLM-assisted classifier that integrates consistency scores from state-of-the-art language models with traditional text embeddings, and (iii) an RL-based sequential agent that leverages reward shaping and recurrent policies to capture long-term dependencies and address class imbalance. Experimental results, evaluated in terms of Accuracy, Macro F1, and Lie F1 scores, confirm that each component is capable of capturing complementary aspects of deceptive communication.

For future work, we plan to explore several enhancements:

- **Model Integration:** Investigate hybrid architectures that combine the strengths of our graph-based, LLM-assisted, and RL-based methods into a unified framework.
- **Fine-Tuning of LLMs:** Fine-tune large language models on in-domain data to improve the extraction of contextual consistency signals.
- **Context-Aware Multi-Agent Reasoning:** Incorporate richer temporal and inter-player interaction features, as well as hierarchical dialogue modeling, to further improve the detection of subtle deceptive cues.
- **Dynamic Reward Schemes:** Experiment with adaptive reward functions in the RL framework to better balance precision and recall on the rare deceptive class.

Overall, our approach takes a significant step toward robust deception detection in multi-agent communication settings, and we believe that the proposed directions will pave the way for further improvements in both predictive performance and interpretability.