Technical Report: Profanity and Privacy Detection in Call Conversations

1. Introduction

This report details two approaches for detecting **profanity** and **privacy violations** in call center conversations:

1. **Pattern Matching** (static, regex-based)
2. **LLM-based Analysis** (dynamic, model-driven)

Both approaches aim to identify when agents or customers use profane language and when agents share sensitive information without proper verification.

---

2. Methodologies

2.1 Pattern Matching

Description:
The pattern matching approach uses **regular expressions** to detect profanity and sensitive information in conversation transcripts. It scans the text for pre-defined patterns including:

- Profanity words and censored forms (e.g., f***, s***)
- Sensitive information (account numbers, balances, SSN, dollar amounts)
- Verification attempts (DOB, address, security questions)

Workflow:

1. Load conversation data from JSON files inside a zip archive.
2. Apply compiled regex patterns to each utterance.
3. Record results per conversation and speaker.
4. Output structured results in **dataframes** for analysis.

Pros:

- Extremely **fast** and computationally inexpensive.
- Easy to **scale** across large datasets.
- No inference cost.

Cons:

- **Static:** Cannot detect nuanced or unexpected expressions.

- May **miss profanities or violations** not covered by regex.
- Less adaptable to new conversational styles or context.

2.2 LLM-based Analysis

Description:
The LLM-based approach uses **ChatGroq (LLaMA 3.1-8B instant)** to analyze conversations dynamically. The model is prompted to detect profanity and privacy violations and return structured JSON output using **Pydantic models** (ProfanityAnalysis and PrivacyAnalysis).

Workflow:

1. Clean and format conversation data.
2. Generate prompts describing what to look for (profanity, sensitive info, verification attempts).
3. Batch conversations and send them to the LLM for analysis.
4. Collect structured responses and convert them into **dataframes**.

Pros:

- **High accuracy** due to contextual understanding.
- Can detect subtle or **uncommon patterns** missed by static regex.
- Easily extensible to handle new types of violations without rewriting code.

Cons:

- **Slower** and more resource-intensive due to model inference.
- **Costly** for large volumes of conversations.
- Requires careful prompt engineering for consistent results.

---

3. Performance and Use Case Considerations

Metric	Pattern Matching	LLM-based Analysis
Speed	Very fast	Slower due to inference
Scalability	Easy	Limited by compute resources
Accuracy	Moderate, static	High, context-aware

Cost	Minimal	Higher (API/compute costs)
Maintenance	Regex updates required	Prompt refinement possible
Coverage of edge cases	Low	High

Recommendation:

- If the goal is **speed and large-scale analysis**, **pattern matching** is preferred.
- If **accuracy and comprehensive coverage** are critical, **LLM-based detection** is the better choice.

---

5. Conclusion

Both approaches provide viable solutions for detecting profanity and privacy violations in call center conversations:

- **Pattern Matching:** Best for high-throughput, low-cost monitoring.
- **LLM-based Analysis:** Best for deep, context-aware inspection with fewer missed violations.