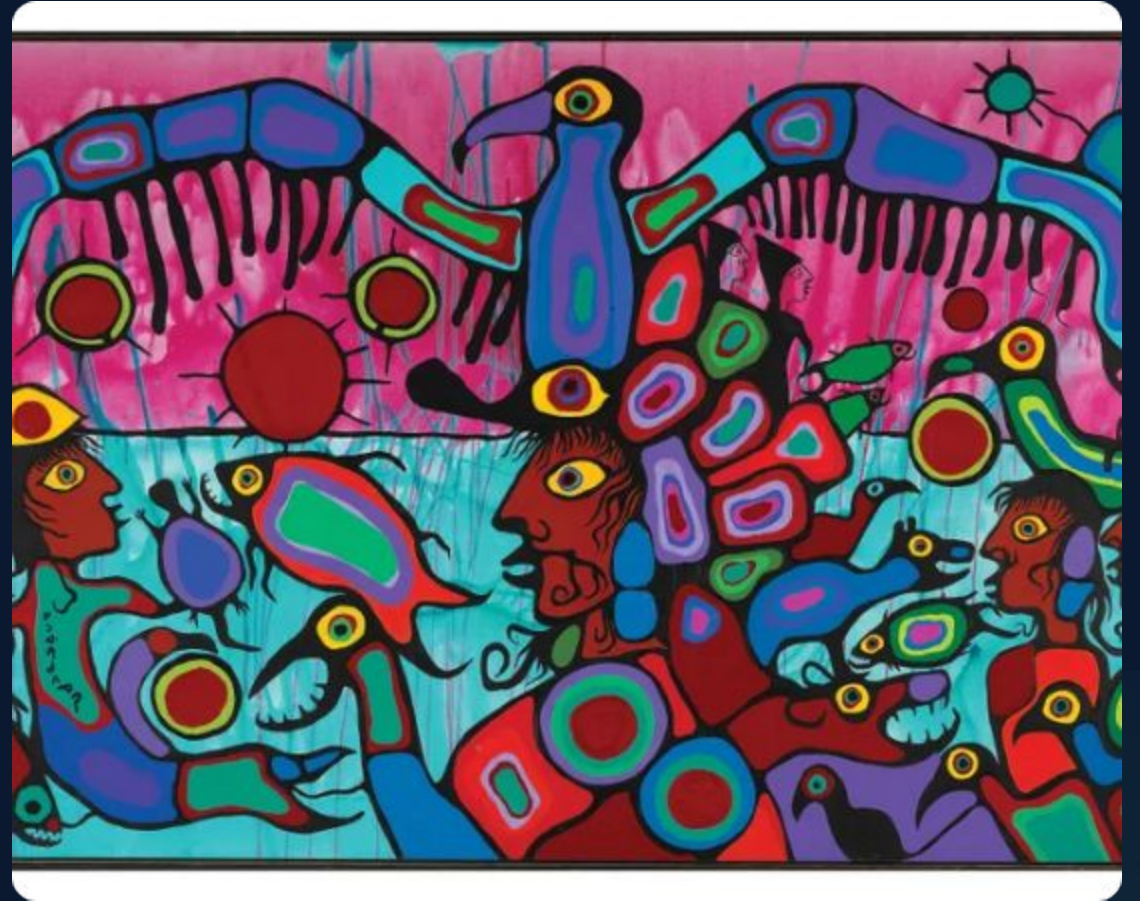


Ensemble Model for Contextual Aggression Detection

A Stacking Architecture for Hinglish (TRAC 2018)

Introduction

Online aggression poses a significant threat to the health of digital communities. While overt aggression is easily filtered, 'covert' hostility, such as sarcasm and passive-aggression, often evades detection. This challenge is amplified in code-mixed Hinglish, a language prevalent on social media. This project implements a deep contextual analysis to distinguish covert aggression from overt and non-aggressive text.



The Problem Statement



Nuance of Language

Current systems fail to identify aggression that lacks obvious slurs. Covert aggression relies on context and subtext, allowing toxicity to persist.



Code-Mixing (Hinglish)

The informal mixing of Hindi and English breaks traditional NLP pipelines that are built for monolingual text, making analysis extremely difficult.

The Core Challenge: 3-Class Classification

The goal is a three-class classification that requires true contextual understanding, not just keyword flagging.

- ⚠ **OAG (Overtly Aggressive):** Obvious insults, slurs, and direct threats.
- ❓ **CAG (Covertly Aggressive):** Sarcasm, passive-aggression, and micro-aggressions.
This is the primary challenge.
- ✅ **NAG (Non-Aggressive):** Normal, neutral, or positive conversation.

Dataset: TRAC 2018

We used the **TRAC 2018 (Trolling, Aggression and Cyberbullying)** dataset, a key resource from a COLING 2018 shared task.

- ▶ **Language:** Hinglish (code-mixed Hindi-English), perfect for our problem.
- ▶ **Source:** Real-world social media posts and comments.
- ▶ **Task:** Labeled for our specific 3-way classification: OAG, CAG, and NAG.



Pre-processing & Feature Engineering

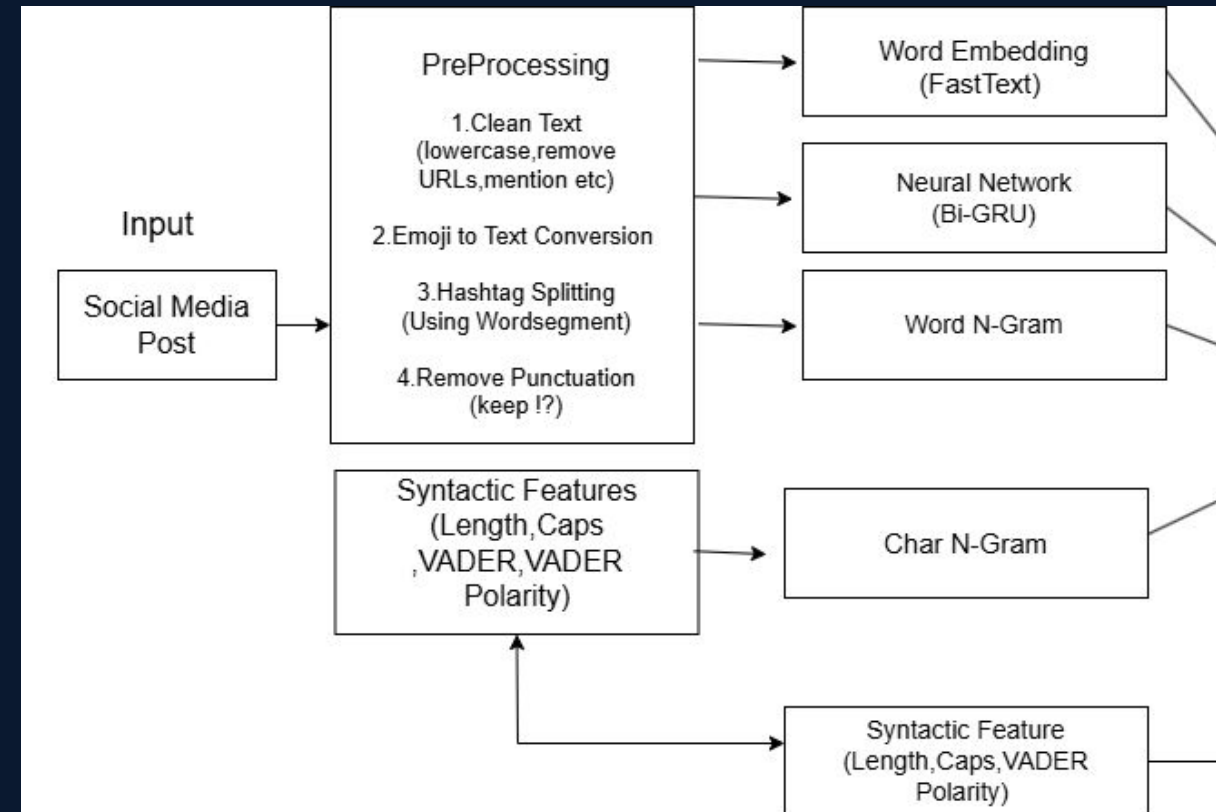
Raw social media text is messy. We created a pipeline to clean and prepare it for our models.

- ✂ **Text Cleaning:** Lowercasing, removing all URLs, and stripping user mentions (@username).
- 😊 **Normalization:** Converting Emojis to their text meaning (e.g., 😊 becomes ":smiling_face:").
- # **Hinglish Handling:** Splitting joined hashtags (e.g., #deshbhakt → "desh bhakt") using ``wordsegment``.
- ! **Punctuation:** Removing most punctuation but strategically keeping `!` and `?` as they indicate sentiment.
- 1/2 **Label Encoding:** Converting text labels (NAG, CAG, OAG) to a numerical format (0, 1, 2) for the models.

Base Model 1: Bi-GRU + FastText

This is our deep learning model, designed to understand **context and word order**.

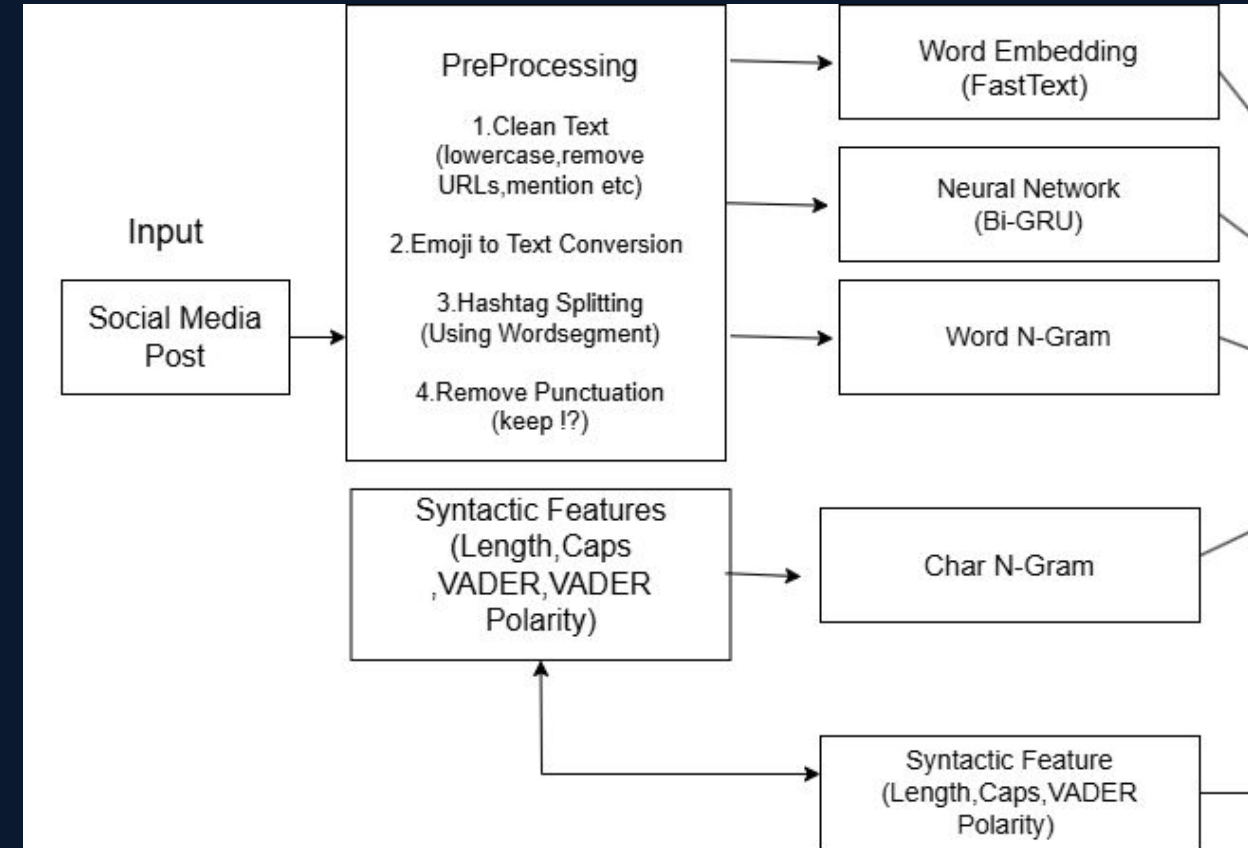
- **FastText Embeddings:** Uses pre-trained Hindi word vectors (300-dim) to represent words. This helps understand misspellings and sub-word information.
- **Bidirectional GRU:** A recurrent neural network (RNN) that reads the text in both directions (forward and backward) to capture the full context.
- **Strength:** Captures nuanced meaning that simple keywords miss.



Base Model 2: Word N-Gram

This is a traditional machine learning model focused on **common word patterns**.

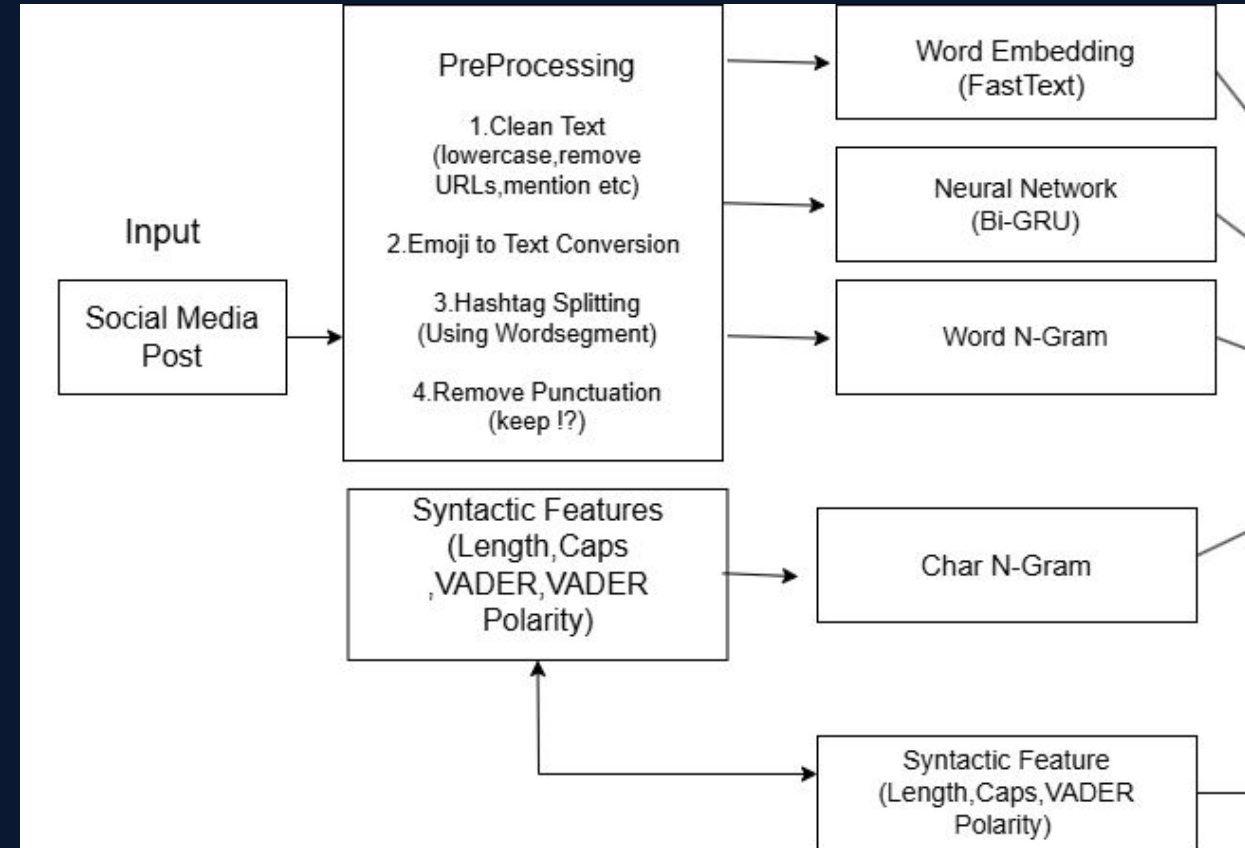
- ▶ **TF-IDF Vectorizer:** Analyzes text based on the frequency of words (1-grams) and word pairs (2-grams).
- ▶ **How it works:** It finds which word pairs (e.g., "very good", "go away") are statistically important for predicting aggression.
- ▶ **Classifier:** Uses Logistic Regression to classify the text based on these features.
- ▶ **Strength:** Very effective at catching common, overt phrases.



Base Model 3: Character N-Gram

This model is similar to Word N-Grams but operates on **characters, not words**.

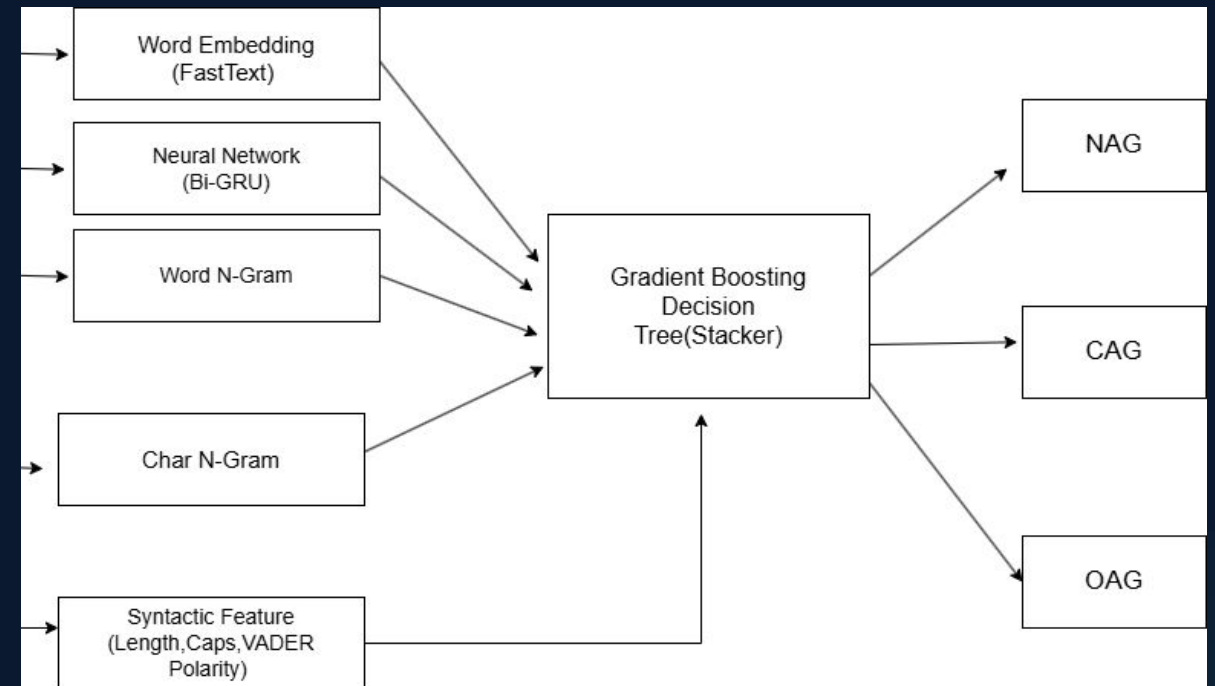
- ▶ **TF-IDF Vectorizer:** Analyzes text using sequences of characters (from 2 to 6 characters long).
- ▶ **Example:** It sees patterns like "bc", "stup", even inside misspelled words.
- ▶ **Classifier:** Also uses Logistic Regression.
- ▶ **Strength:** Extremely robust against misspellings, abbreviations, and attempts to evade filters (e.g., "a\$\$hole").



Base Model 4: Hand Picked Feature Model

This model ignores the words themselves and focuses only on the **style and structure** of the text.

- ▶ **VADER Sentiment:** Extracts polarity scores (positive, negative, neutral, compound).
- ▶ **Syntactic Features:**
 1. Number of words
 2. Proportion of UPPERCASE letters
 3. Number of exclamation marks (!)
 4. Number of question marks (?)
- ▶ **Classifier:** A Logistic Regression model predicts aggression based only on this metadata.



Our Proposed Solution: The Stacking Ensemble

No single model captures all features. We propose a **Stacking Ensemble** to combine the strengths of all 4 base models.



Bi-GRU (Deep Learning): Captures semantic meaning and context.



Word/Char N-Grams (ML): Captures statistical text patterns and misspellings.



Syntactic Features (ML): Analyzes sentiment and writing style.

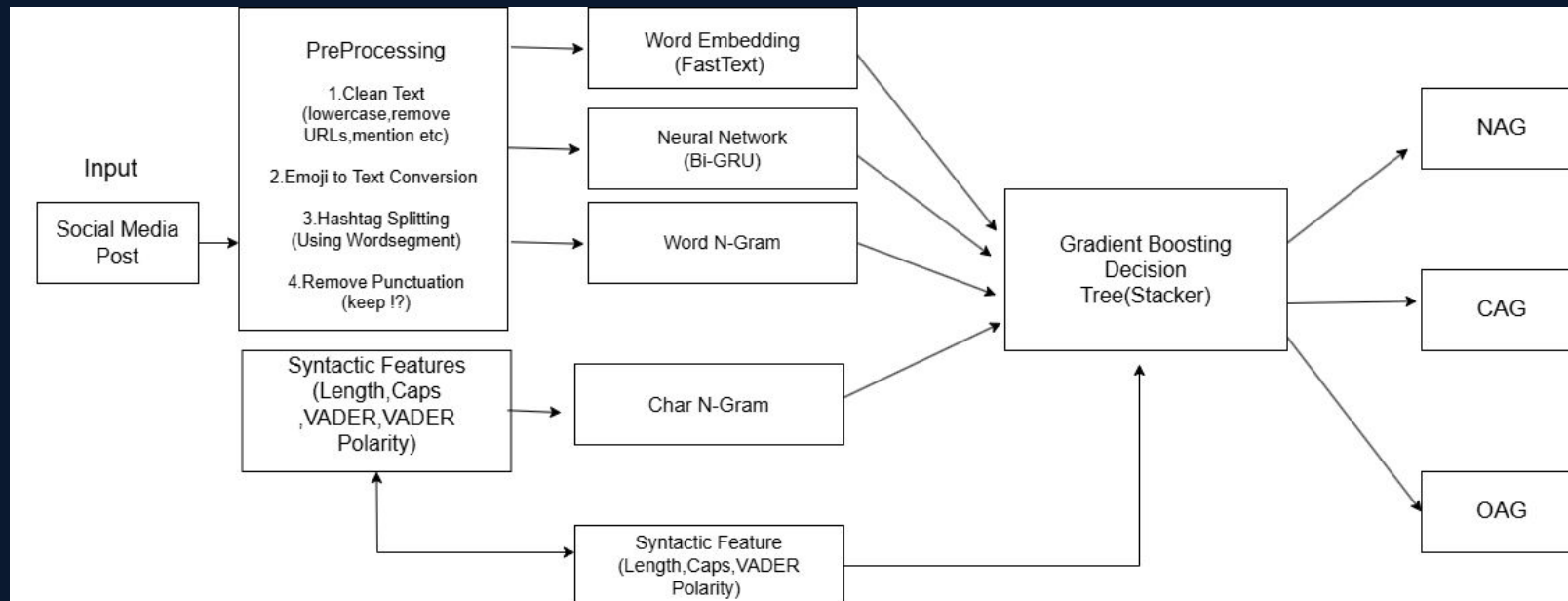


Stacker (Gradient Boosting): A 'meta-model' that learns how to best combine the predictions from all base models to make a final, more accurate decision.

Model Architecture

This diagram shows the flow from a raw social media post to the final classification.

Multiple "base learners" (Bi-GRU, N-Grams, etc.) analyze the text in parallel. Their outputs, plus raw syntactic features, are then fed into the "Stacker" model.



Architecture Explained



1. Feature Engineering

Raw input is cleaned. Syntactic features (Length, VADER) are extracted. Text is tokenized for N-Grams and the Bi-GRU.



2. Base Learners

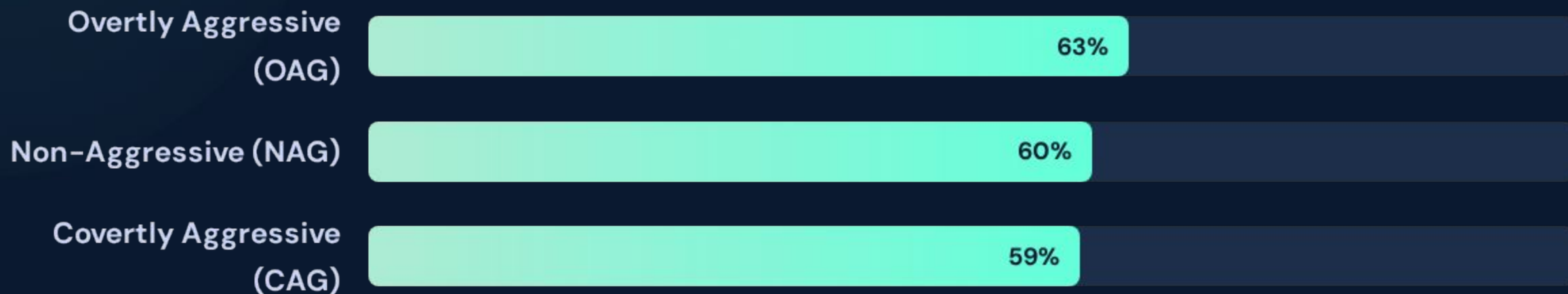
The four base models (Bi-GRU, Word N-Gram, Char N-Gram, Syntactic) all make their own predictions on the validation set.



3. Stacking & Classification

The predictions from Step 2 become the **new input** for the Gradient Boosting Stacker, which learns from them to make the final call.

Results: Final Ensemble F1-Score (Test Set)



The model achieves a weighted F1-score of 61%. As hypothesized, Covert Aggression (CAG) is the most challenging class to detect, which highlights the difficulty of contextual, nuanced language vs. overt aggression.

Conclusion & Future Work

Conclusion

Contextual aggression in Hinglish is a solvable problem. Our ensemble architecture, combining deep learning (Bi-GRU) with traditional ML features (N-Grams) and a Gradient Boosting stacker, proves effective at establishing a strong baseline for this complex task.

Future Work

- ▶ Integrate larger transformer models (e.g., mBERT).
- ▶ Expand the dataset with more verified examples of sarcasm.
- ▶ Move towards a real-time deployment.

Thank You