Hierarchical Opinion Classification using Large Language Models

FIRE 2025 — Forum for Information Retrieval Evaluation

Shuvam Banerji Seal
sbs22ms076@iiserkol.ac.in
Department of Chemical Sciences,
Indian Institute of Science Education & Research
Kolkata

Alok Mishra
maa24ms215@iiserkol.ac.in
Indian Institute of Science Education & Research
Kolkata

Utkarsha Ghosh
utkarsha.ghosh2023@iem.edu.in
Department of Information Technology,
Institute of Engineering and Management, Kolkata

December 17–20, 2025

1. Outline

Presentation Outline

2. Introduction

Motivation & Dataset Overview

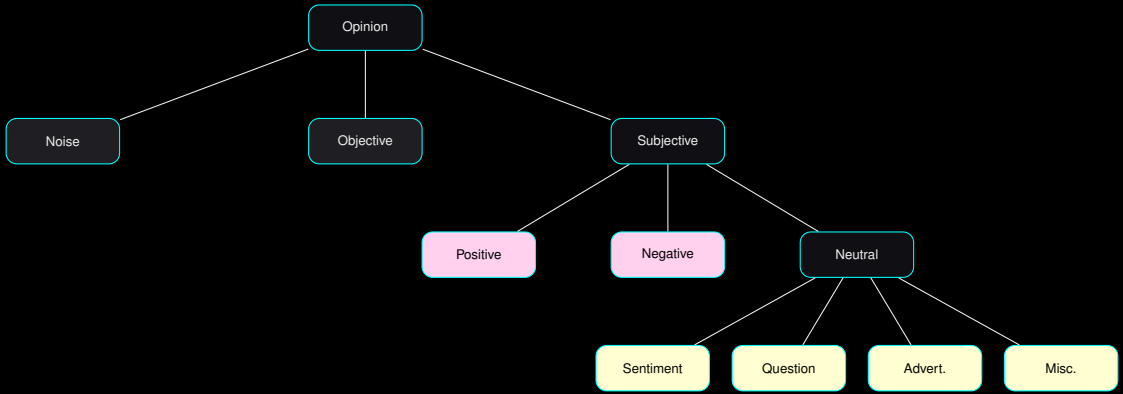
Context: FIRE 2025 Shared Task

- The datasets (Reddit, Twitter, YouTube) and the **3-level hierarchy** were provided as part of the shared task.
- **Why Hierarchy Matters:** It disambiguates *intent* from *sentiment*.
  - → Differentiates Complaints (Negative) from Inquiries (Questions).
  - → *Where it doesn't matter:* **Noise** is flat and requires no depth.

Real Examples from Dataset:

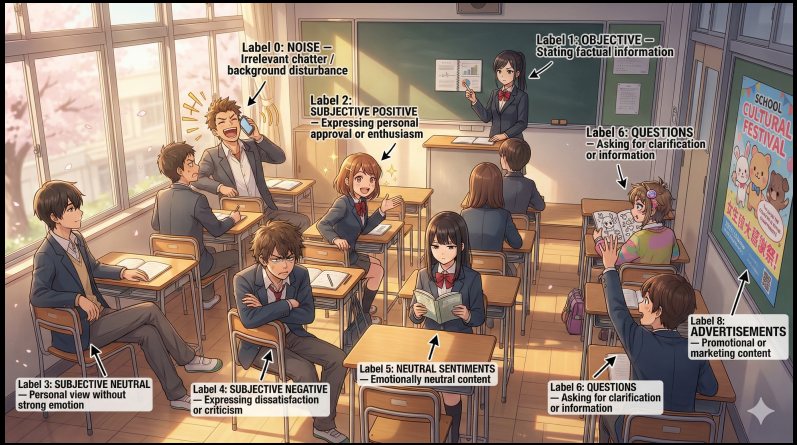
1. **Objective (Level 1):**
   "Binance is reportedly building a $1 billion insurance fund to counter crypto hacks."
2. **Subjective → Negative (complaint):**
   "Do yourself a favor and don't use Kucoin... They are not reliable... my coins are now lost..."
3. **Subjective → Neutral → Question (intent):**
   "How does TDS work? ... they give our 1% TAX with our PAN... can someone explain?"

Hierarchical Structure of Opinions



- **Level 1:** Coarse-grained (Noise, Objective, Subjective)
- **Level 2:** Sentiment refinement (Neutral, Negative, Positive)
- **Level 3:** Fine-grained neutral categories

Label Visualization



Our Approach — Overview

Approach 1: Classification Fine-tuning

- Custom classification head
- Parameter-efficient fine-tuning
- Weighted cross-entropy loss

Approach 2: Instruction Fine-tuning

- Prompt-response format
- Next-token prediction
- Masked loss on answer tokens

Base Model: Gemma3-1B — Compact 1-billion parameter LLM

3. Problem Statement

The 8-Class Classification Task

Hierarchical → Flat Mapping:

ID	Category
0	NOISE
1	OBJECTIVE
2	SUBJECTIVE_POSITIVE
3	SUBJECTIVE_NEGATIVE
4	NEUTRAL_SENTIMENTS
5	QUESTIONS
6	MISCELLANEOUS
7	ADVERTISEMENTS

Key Challenges

- Severe **class imbalance**
- **Semantic overlap** among categories
- **Noisy** social media text
- **Hierarchical dependencies**

4. Data Preparation

4-Phase Data Cleaning Workflow

Phase 1: Structural Integrity

- Schema alignment across sources
- Null entry elimination
- Token-based duplicate detection
- Corpus integrity verification

Phase 2: Content Cleaning

- URL/hyperlink removal
- Mention/hashtag filtering
- Emoji normalization
- Whitespace compacting

Phase 3: Text Normalization

- Case normalization (lowercase)
- Domain-specific token preservation
- Final text validation

Phase 4: Label Formatting

- Hierarchical label consolidation
- Numeric label encoding (0–8)
- Human-readable mapping

Dataset Statistics

Reddit Dataset

- 5,000 samples
- Mean: 186 tokens
- Max: 15,535 tokens

Top classes:

- Noise: 645
- Objective: 503
- Neutral: 476

Twitter Dataset

- 4,987 samples
- Mean: 46 tokens
- Max: 151 tokens

Top classes:

- Objective: 1,700
- Noise: 1,338
- Positive: 268

YouTube Dataset

- 5,000 samples
- Mean: 38 tokens
- Max: 1,128 tokens

Top classes:

- Negative: 1,574
- Neutral: 1,391
- Question: 1,000

Class Imbalance

Significant variation across platforms — *Advertisements* has only 1 sample in YouTube!

5. Model Architecture

Gemma-1B with Custom Classification Head

Architecture Pipeline:

1. **Token Embedding**
   $H_0 \leftarrow \text{EmbeddingLookup}(X_{\text{tokens}})$
2. **Transformer Backbone**
   L transformer layers with `Gemma3RMSNorm`
3. **Final Normalization**
   $H_{\text{final}} \leftarrow \text{RMSNorm}(H_L)$
4. **Last Token Extraction**
   $h_{\text{last}} \leftarrow H_{\text{final}}[:, -1, :]$
5. **Classification Head**
   $\text{LayerNorm} \rightarrow \text{Dropout} \rightarrow \text{Linear}(8)$

Custom Classification Head

```
Sequential(
LayerNorm(1152)
Dropout(p=0.4)
Linear(1152 → 8)
)
```

Key Design

Replace original LM head
$(1152 \rightarrow 262144)$
with classification head
$(1152 \rightarrow 8)$

Parameter-Efficient Fine-Tuning (PEFT)

Trainable Components

- Last transformer block
- Final LayerNorm
- LM head
- Custom classification head

Parameter-Efficient Fine-Tuning (PEFT)

Trainable Components

- Last transformer block
- Final LayerNorm
- LM head
- Custom classification head

Frozen Components

- All earlier transformer blocks
- Preserves pretrained linguistic knowledge

Parameter-Efficient Fine-Tuning (PEFT)

Trainable Components

- Last transformer block
- Final LayerNorm
- LM head
- Custom classification head

Frozen Components

- All earlier transformer blocks
- Preserves pretrained linguistic knowledge

Benefits:

- Reduces trainable parameters significantly
- Prevents overfitting on small datasets
- Enables training on single 24GB GPU

6. Training Methodology

Efficiency Techniques

Memory Optimization

- **4-bit Quantization** (NF4)
- Reduces memory by $\sim$75%
- Uses `bitsandbytes` backend

Gradient Stability

- Gradient clipping (max norm = 1.0)
- `bfloat16` precision
- Prevents NaN values

Optimizer: AdamW

Learning rate	5×10^{-5}
Weight decay	0.1
Scheduler	Cosine + warmup
Warmup ratio	15%

Training Config

- 3 epochs
- 70/20/10 train/val/test split
- Batch size: 4

Handling Class Imbalance

Weighted Cross-Entropy Loss:

Class weights based on inverse frequency:

$$\text{weights} = [225, 175, 80, 130, 175, 900, 70, 30]$$

$$\text{normalized} = \frac{\text{weights}}{\sum \text{weights}} \times 8$$

Effect:

- Higher penalty for minority classes
- Forces model to learn rare categories

Without Weighting

- Model overfits to *Noise* and *Objective*
- Near-zero F1 for minorities
- Accuracy appears high but misleading

7. Instruction Fine-Tuning

Instruction Fine-Tuning Approach

Key Steps:

1. **Tokenization of Prompt-Response Pairs**
   Concatenate prompt + input text
2. **Sequence Shifting**
   $Y = [x_1, x_2, \ldots, x_N]$ from $X = [x_0, x_1, \ldots, x_{N-1}]$
3. **Masking Non-Label Tokens**
   Set prompt tokens to `-100` (ignored)
4. **Label Alignment Verification**
   Ensure exact match of label tokens

Model Used: Gemma3:27B

Objective

- Next-token prediction
- Loss only on answer tokens
- Preserves autoregressive nature

Benefit

Better alignment with instruction-based reasoning

8. Results

Experimental Comparison: Classification vs. Instruction Tuning

Run 1: Classification Fine-tuning

Method: Standard supervised learning (Flat 8-class head).

Reddit (500 samples):

- **Subjective:** 41.4% (Dominated by opinion)
- **Objective:** 31.6%
- **Noise:** 27.0%

Twitter (500 samples):

- **Noise: 64.4%** (Model collapsed to majority)
- **Objective:** 18.2%
- **Subjective:** 17.4%

Run 2: Instruction Fine-tuning

Method: Prompt-response (Next-token prediction).

Reddit (500 samples):

- **Objective:** 44.6% (Better factual detection)
- **Noise:** 28.2%
- **Subjective:** 27.2%

Twitter (500 samples):

- **Noise: 46.6%** (Significant reduction)
- **Subjective:** 35.6% (Better opinion recovery)
- **Objective:** 17.8%

Impact on Minority Classes (QnA Dataset)

The Challenge

In the QnA dataset, **Class 1 (Relevant/Question)** is a severe minority compared to **Class 0 (Noise/Irrelevant)**.

Category	Run 1 (Class. Head)	Run 2 (Instruct.)
Class 0 (Majority)	99.75%	91.11%
Class 1 (Minority)	0.25%	**8.89%**

Improvement Factor

35×

Improvement in minority class recognition.

Key Findings & Analysis

Why Instruction Tuning Won?

- **Reduced Noise Dominance:** Run 2 successfully shifted predictions away from the "safe" majority class (Noise).
- **Minority Coverage:** Significant gains in *Questions* and *Advertisements* categories.
- **Alignment:** The prompt-response format aligns better with the model's pre-trained reasoning capabilities.

Remaining Challenges

- **Noise Persistence:** Even in Run 2, noise remains a high percentage for Twitter (46.6%), due to the messy nature of the platform's data.
- **Subjective/Objective Flip:** In Reddit, Run 2 favored Objective labels more than Run 1, altering the distribution profile.

Conclusion: Instruction fine-tuning offers superior robustness for imbalanced, hierarchical datasets compared to standard classification heads.

9. Challenges & Solutions

Challenges I: Training Stability & Resources

Gradient Explosion

The Problem:

- Training destabilized early on.
- Large gradients caused weight updates to spiral.
- Resulted in **NaN values** in output tensors.

The Solution:

1. **Gradient Clipping:** Capped max norm at `1.0` to constrain updates.
2. **Precision Shift:** Switched from `float32` to **`bfloat16`**.
3. *Result:* Prevents numerical overflow while maintaining dynamic range.

Memory Overflow (OOM)

The Problem:

- 24GB VRAM limit on single GPU.
- Quadratic complexity of attention mechanism.
- `float32` weights exceeded capacity immediately.

The Solution:

1. **4-bit Quantization:** Used NF4 (NormalFloat4) via `bitsandbytes` ($\sim$75% reduction).
2. **Batch Size Reduction:** Reduced training batch size from 8 $\rightarrow$ 4.
3. *Result:* Enabled fine-tuning of 1B parameters on consumer hardware.

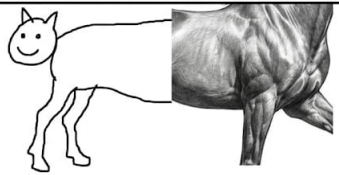
Visualization

Gemma-1B Backbone (Frozen Layers)

A pre-trained linguistic powerhouse. Understands English perfectly. Ready for action.

Our New Classification Head & Fine-Tuning Process

Attaching a simple MLP. But wait... Gradient Explosions! Memory Overflow! Catastrophic Forgetting! The struggle is real.



The Challenge: Performing '**Brain Surgery**' without Breaking the Patient. We had to use **PEFT, bfloat16**, and gradient **clipping** just to get the new head to talk to the old body!

Challenges II: Optimization & Generalization

Overfitting to Majority Classes

The Problem:

- Model achieved high surface-level accuracy.
- *Reality:* It was predicting "Noise" or "Objective" for everything.
- **F1-Score $\approx$ 0** for minority classes (Questions, Ads).

The Solution:

- **Weighted Cross-Entropy Loss:**
- Calculated inverse frequency weights:

$$w_c = \frac{N}{N_c}$$

- Penalized misclassification of rare classes heavily.

Stagnant Convergence

The Problem:

- Accuracy plateaued at $\sim$50% during mid-training.
- Model unable to escape local minima.

The Solution:

- **Architecture Tuning:** Added extra `LayerNorm` before the head.
- **Regularization:** Introduced `Dropout (p=0.4)` to prevent neuron co-adaptation.
- **Activation:** Switched to GELU for smoother gradients.

10. Conclusion

Summary

- Adapted Gemma-1B for **8-class hierarchical opinion classification**
- Implemented **parameter-efficient fine-tuning**:
  - → Custom classification head
  - → Selective layer unfreezing
  - → 4-bit quantization for memory efficiency
- Compared two approaches:
  - → **Classification fine-tuning:** Direct supervised learning
  - → **Instruction fine-tuning:** Better minority class coverage
- Addressed **class imbalance** via weighted loss functions
- Demonstrated effectiveness on **Reddit, Twitter, YouTube, QnA** datasets

Future Directions

Extensions

- Cross-domain generalization
- Cross-lingual adaptation
- Hierarchical label dependency modeling

Improvements

- Robustness to noisy data
- Temporal pattern analysis
- Refined evaluation metrics

Code Available:

https://github.com/Shuvam-Banerji-Seal/FIRE_2025_Hierarchical_Opinion_Classification_using_Large_Language_Models

11. Acknowledgments

Acknowledgments

We thank Dr. Dwaipayan Roy

Department of Computational and Data Sciences (CDS)
Indian Institute of Science Education and Research, Kolkata

for providing the computing resources for our fine-tuning runs.

Thank You!

Questions?