AI-Based Classroom Observation Evaluation: A Multimodal Research Pipeline

Matt Krasnow
Swiftscore | *Harvard*
matt@swiftscore.org

June 4, 2025

1 Abstract

This document outlines a comprehensive validation and evaluation pipeline for comparing AI-based classroom observation evaluators against human expert evaluations. The pipeline incorporates multimodal AI models, defines robust distance metrics for mixed scoring systems, computes inter-rater reliability measures, and provides visualization tools for performance assessment.

2 Pipeline Overview

The evaluation pipeline consists of ten core components designed to systematically validate AI evaluators against human ground truth across multiple domains of classroom observation. The system supports both text-based and multimodal (audio + text) AI evaluators.

2.1 Core Components

1. Data loading and preprocessing framework
2. Distance metric computation for heterogeneous scoring systems
3. Inter-rater reliability analysis (Cohen's κ)
4. Multi-model evaluation orchestration
5. Comprehensive visualization suite
6. Test set evaluation controls with confirmation gates

3 Mathematical Framework

3.1 Score Normalization

The pipeline handles three primary scoring types through the normalization function $\alpha : S \to \mathbb{R} \cup \{\text{NaN}\}$:

$$\alpha(s) = \begin{cases} 1.0 & \text{if } s \in \{\text{Y, y, Yes, 1}\} \\ 0.0 & \text{if } s \in \{\text{N, n, No, 0}\} \\ \text{NaN} & \text{if } s \in \{\text{N/A, NA, empty}\} \\ 1.0, 2.0, 3.0 & \text{if } s \in \{\text{L, M, H}\} \text{ respectively} \\ \text{float}(s) & \text{if } s \text{ is numeric (1-5 scale)} \\ \text{NaN} & \text{otherwise} \end{cases} \tag{1}$$

3.2 Component Distance Metric

For each component c, the distance between human score h_c and AI score a_c is computed as:

$$d_c(h_c, a_c) = \begin{cases} 0.0 & \text{if } \alpha(h_c) = \alpha(a_c) = \text{NaN} \\ 1.0 & \text{if } \alpha(h_c) = \text{NaN} \oplus \alpha(a_c) = \text{NaN} \\ \frac{|\alpha(h_c) - \alpha(a_c)|}{d_{\max}} & \text{otherwise} \end{cases} \tag{2}$$

where $d_{\max}$ is the maximum possible difference for the scoring type:

$$d_{\max} = \begin{cases} 1.0 & \text{for Y/N scoring} \\ 2.0 & \text{for L/M/H scoring} \\ 4.0 & \text{for 1-5 numeric scoring} \end{cases} \tag{3}$$

3.3 Domain and Overall Distance Aggregation

For domain D containing components $\{c_1, c_2, \ldots, c_n\}$ with weights $\{w_1, w_2, \ldots, w_n\}$:

$$D_{\text{domain}} = \frac{\sum_{i=1}^{n} w_i \cdot d_{c_i}}{\sum_{i=1}^{n} w_i} \tag{4}$$

The overall distance across all domains $\{D_1, D_2, \ldots, D_m\}$ with domain weights $\{W_1, W_2, \ldots, W_m\}$:

$$D_{\text{overall}} = \frac{\sum_{j=1}^{m} W_j \cdot D_j}{\sum_{j=1}^{m} W_j} \tag{5}$$

3.4 Inter-Rater Reliability

Cohen's Kappa coefficient is computed for each component using encoded categorical scores. For ordinal scales (L/M/H, 1-5), weighted kappa is employed:

$$\kappa = \frac{p_o - p_e}{1 - p_e} \tag{6}$$

where p_o is the observed agreement and p_e is the expected agreement by chance. For weighted kappa with quadratic weights:

$$\kappa_w = \frac{\sum_{i,j} w_{ij} p_{ij} - \sum_{i,j} w_{ij} p_{i\cdot} p_{\cdot j}}{1 - \sum_{i,j} w_{ij} p_{i\cdot} p_{\cdot j}} \tag{7}$$

where $w_{ij} = 1 - \frac{(i-j)^2}{(k-1)^2}$ for k categories.

4 Implementation Architecture

4.1 Data Pipeline

The system loads evaluation data from cleaned transcript CSVs and framework JSON specifications. Base IDs are extracted using regex pattern matching on school clip identifiers: base_id $\leftarrow$ extract(School_Clip, pattern = $\backslash d\{6,7\}$).

4.2 Model Integration

The pipeline supports multiple evaluator architectures:

- `SimpleModelEvaluator`: Text-only evaluation
- `MultiModalModelEvaluator`: Audio + text evaluation
- `AdvancedModelEvaluatorX`: Extended feature evaluation

Each evaluator implements a standardized interface: `evaluate(transcript_text, audio_file_path=N` $\rightarrow$ evaluation dictionary.

4.3 Evaluation Orchestration

For each selected model M and dataset split S:

1. Load human evaluations $H = \{h_1, h_2, \ldots, h_n\}$
2. Generate AI evaluations $A_M = \{a_1^M, a_2^M, \ldots, a_n^M\}$
3. Compute distance matrix $D_M = [d_c(h_i, a_i^M)]_{i,c}$
4. Calculate reliability metrics $\kappa_M = [\kappa_c^M]_c$

5 Visualization and Analysis

The pipeline generates comprehensive performance visualizations:

- **Distance Analysis**: Boxplots and histograms of component and overall distances
- **Reliability Assessment**: Bar charts of Cohen's κ values per model and component
- **Performance Correlation**: Scatter plots of transcript length vs. evaluation distance
- **Model Comparison**: Comparative analysis across multiple AI evaluators

6 Quality Assurance

6.1 Test Set Protection

The pipeline implements mandatory confirmation controls for test set evaluation to prevent accidental model validation leakage. Test set access requires explicit user confirmation via checkbox validation.

6.2 Error Handling

Robust exception handling ensures graceful degradation when model calls fail, with standardized error reporting and result aggregation.

7 Future Extensions

- **LLM-as-Judge Integration**: Third-party LLM comparison of human vs. AI evaluation summaries
- **Parallel Processing**: Multi-threaded model evaluation for large-scale deployment
- **Advanced Metrics**: Implementation of weighted F1-scores and custom domain-specific metrics
- **Real-time Evaluation**: Streaming evaluation pipeline for live classroom observation

8 Conclusion

This pipeline provides a systematic framework for validating AI-based classroom observation tools against expert human evaluation. The mathematical rigor of the distance metrics, combined with established inter-rater reliability measures, enables robust assessment of AI evaluator performance across diverse classroom contexts and observation frameworks.