# Music Piece Classification

Selina Zou, Phillip Duarte, Ethan Fan

STAT 4830

Tuesday, March 3, 2026

# Here's a Question:

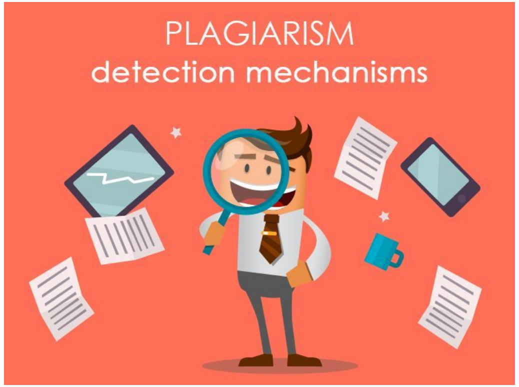
- If I played you five seconds of a Bach chorale – no context, no title – could you tell me which one it was? Out of 430?

# But Why?

## Real Applications:

### Recommendation


### Plagiarism (Sample) Detection




### Archival Search

# Short History

- Hand-crafted → CNN baselines
  - Mel-frequency cepstral coefficients + Support Vector Machines
    - Interpretable but brittle, no long-range harmonic structure
  - CNNs on mel-spectrograms (Dong 2018; Costa et al.)
    - Substantially outperform hand-crafted features, but require large labeled datasets and don't transfer well
- Transfer learning → music-specific SSL
  - VGGish / PANNs
    - General-purpose pretrained audio encoders
    - Reduce labeled data needs, but aren't organized around music-specific tonal/harmonic structure
  - MERT (Li et al., 2023) (SoTA)
    - BERT-style transformer trained on music with a dual-teacher setup
    - → **the setup we directly adopt**

# Initial Approach

- Use a small set of data from music21 with 5 extracted music features of pieces
  - key, time signature, average pitch, pitch range, note density
  - develop a ‘similarity function’ for each of these
- Systematically split music pieces into pages and create pairs of matching and non-matching pages
  - Goal: classify whether matching or not → **Logistic Regression**

# Initial Approach: Problem Formulation

- **Features:** similarity of aforementioned 5 metrics in a vector $s = [s_1, s_2, s_3, s_4, s_5]$, $s_i$ in range -1 to 1
- **Labels:** 1 for match (2 pages from same piece), 0 for non-match (2 pages from different pieces)
- **Loss**: Binary Cross-Entropy Loss
- **Optimization**
  - Projected Gradient Descent - with PyTorch
  - Compared to: convex solver (CVXPY) and Sequential Least Squares Programming (SciPy)

# Initial Approach: Problem Formulation (cont.)

## Logistic Regression

## For similarity vector $s = [s_1, s_2, s_3, s_4, s_5]$

Minimize

$$L(\mathbf{w}) = -\sum_{(i,j)\in\text{training pairs}} \text{label}_{ij} \cdot \log\left(\sigma(\mathbf{w}^T \mathbf{s}_{ij})\right) + (1 - \text{label}_{ij}) \cdot \log\left(1 - \sigma(\mathbf{w}^T \mathbf{s}_{ij})\right)$$

Subject to

1. $w_k \geq 0$ for all $k \in \{1, 2, 3, 4, 5\}$ (non-negative weights)

For reference, the gradient with respect to $w$ is as follows:

$$\nabla L(\mathbf{w}) = \sum_{(i,j)} (\sigma(\mathbf{w}^T s_{ij}) - \text{label}_{ij}) \cdot \mathbf{s}_{ij}$$

**Goal:**

$$\mathbf{1}(\mathbf{w}^T \mathbf{s}_{ij} \geq 0) = \begin{cases} 1, & \mathbf{w}^T \mathbf{s}_{ij} \geq 0 \\ 0, & o.w. \end{cases}$$

Learn a weight vector of

$$\mathbf{w} = [w_1, w_2, w_3, w_4, w_5]$$

# Initial Approach: Results

**Training set:** 35 examples (14 matching, 21 non-matching)

**Test set:** 15 examples (9 matching, 6 non-matching)

Hyperparameters: learning rate, max iterations, tolerance (stopping condition is when gradient norm is < this)

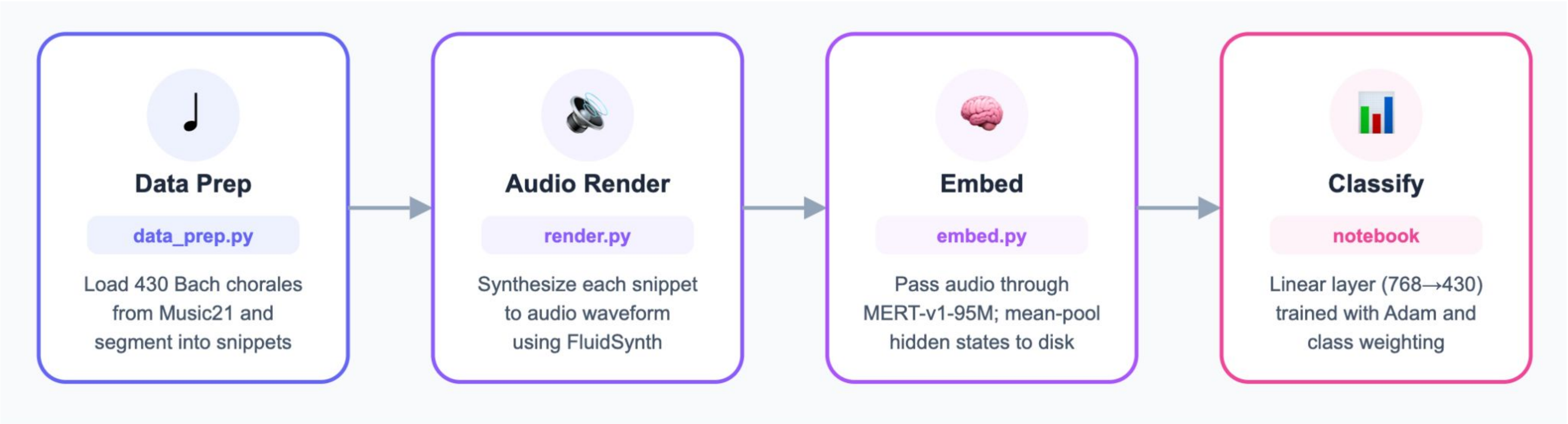
| | Learned weights | Loss | Train Accuracy | Test Accuracy | Iters | Time (s) |
|---|---|---|---|---|---|---|
| Projected Gradient Descent | [0.0582, 0, 0, 0.2407, 0] | 0.6902 | 0.5429 | 0.5333 | 1000* | 0.2327 |
| Sequential Least Squares Programming | [0.0582, 0, 0, 0.2457, 0] | 0.6902 | 0.5429 | 0.5333 | 7 | 0.0489 |
| Convex Optimization | [0.0575, 0, 0, 0.2446, 0] | 0.6902 | 0.5429 | 0.5333 | N/A | 0.076 |

* learning rate = 0.01, max_iterations = 1000, tolerance = 1e-4

# Initial Approach: Analysis

- The learned weight vector was ~[0.0582, 0, 0, 0.2407, 0]
  - Only key signature and pitch range where important
- Training (0.54) and test (0.53) accuracies not high
- When separating based on positives (match), hard negatives (different pieces by same composer), easy negatives (different pieces by different composers), accuracies were 0.67, 0.25, and 0.60 respectively
- Concerns
  - Validity of similarity functions from music theory perspective
  - Lack of independence between features
  - Overall simplicity of approach that limits what can be learned

# Our Pipeline



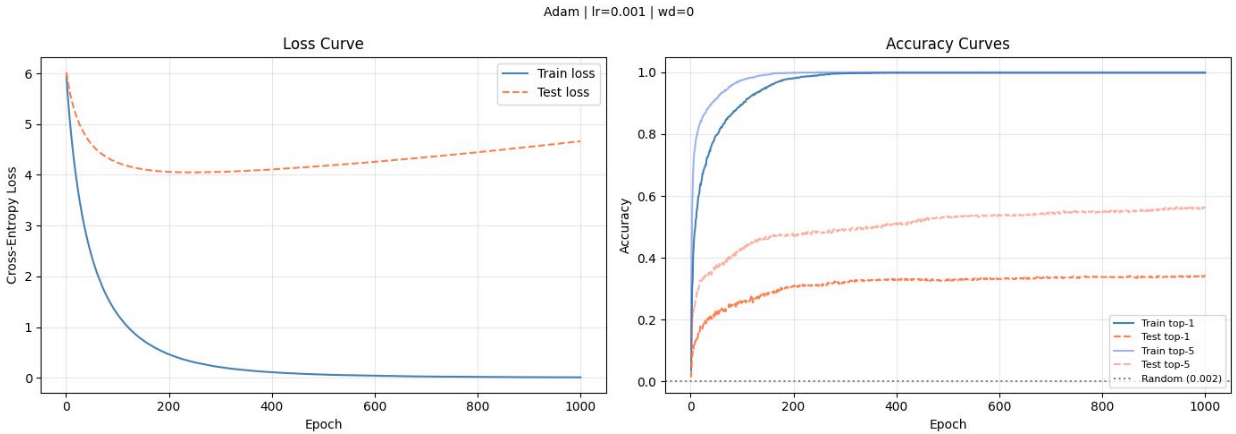
# Results

| | | | |
|---|---|---|---|
| **2,487** | **521** | **34.4%** | **56.4%** |
| **Training Snippets** | **Test Snippets** | **Top-1 Accuracy** | **Top-5 Accuracy** |
| from 430 chorales | 768-dim MERT embeddings | vs. 0.23% baseline | ~150× over chance |

# Analysis

## The bottleneck is linearity

- Training accuracy reaches 99.8% while test accuracy plateaus at 34.4% with no upward trend across 1,000 epochs.
- Sweeping regularization across $10^{-4} \rightarrow 0$ produced no meaningful change in test accuracy, ruling out overfitting as the cause.
- **Root Cause:** A linear decision boundary in 768-dimensional space is insufficient to separate 430 classes whose embeddings are not linearly arranged.



# Analysis (cont.)

## Per-Class Breakdown

- Median per-class accuracy: 0%
- 8 of 430 pieces identified at 100%
- 410 of 430 pieces identified at 0%
- Aggregate 34.4% driven by ~8 'easy' pieces with distinctive harmonic content
- bwv248.64-6 and bwv79.3 act as prediction attractors, absorbing misclassifications

## Implementation Insight

- L2 normalization before the linear layer dropped top-1 accuracy from 34.4% → ~4.4%
- MERT embedding magnitudes carry piece-identity information
- Normalizing to unit length discards that signal
- Suggests magnitude encodes harmonic energy relevant to piece identity

Penn Engineering

# Future Work

- **MLP Classifier**
  - 768 → 512 → 430
  - ReLU / Dropout for regularization
  - Non-convex  optimizer choice matters
- **Harder Eval Split**
  - Switch to by-piece split
  - Tests generalization to unseen chorales
  - Required for metric learning
- **Probe Embedding Space**
  - Compute same- vs. different-piece distances
  - Diagnose whether bottleneck is classifier or geometry
  - Informs whether to fine-tune MERT