Methods for Temporal AI Classification of Pianist Fine-Motor Skill

Stanford CS229 Project

Max Rodriguez
Department of Computer Science
Stanford University
maxrod@stanford.edu

Methods for Dimensionality Reduction

Principal Component Analysis (PCA)

- Applied to both **convoluted** and **flattened** datasets (training and holdout).
- Reduced computational cost and noise while maintaining essential features.

Linear Discriminant Analysis (LDA)

- Applied to maximize **between-class variance** relative to **within-class variance**.
- Solves the generalized eigenvalue problem:

$$S_W^{-1} S_B \, \mathbf{w} = \lambda \, \mathbf{w}$$

- Select the eigenvector corresponding to the largest eigenvalue and project the data onto this direction.

Classification Methods

Gaussian Discriminant Analysis (GDA)

I evaluated GDA both with and without L2 regularization. Under the assumption of Gaussianity, GDA is asymptotically efficient; however, its performance may suffer if the assumption is violated [2]. To mitigate overfitting, L2 regularization was added to the covariance via:

$$\Sigma_{\text{reg}} = \Sigma + \lambda I.$$

Multi-Layer Perceptron (MLP)

The **MLP** is trained using cross-entropy loss with the architecture

[num_features, 64, 32, 2]

(for binary classification). Hyperparameters—including learning rate, number of epochs, and layer sizes—were determined through grid search, and the model was optimized using mini-batch stochastic gradient descent. No regularization was necessary, as overfitting was not observed.

Figure 1: Effect of layer depth on MLP accuracy.

Decision Tree

A conventional **Decision Tree** serves as the baseline. Its maximum depth was optimized via grid search.

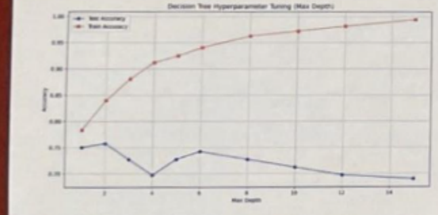

Figure 2: Variance-bias tradeoff for decision tree depth selection.

Activity Classifier with AdaBoost

For a two-stage approach, the training data were first segmented into 6 clusters using k-means. Holdout samples were assigned to clusters based on their proximity to centroids. An **AdaBoost classifier** was then trained separately on each cluster, using a decision tree (with a maximum depth of 4) as the base estimator and 50 estimators (configured with `random_state=42`). Hyperparameters (iterations, maximum depth, and number of clusters) were tuned via grid search.

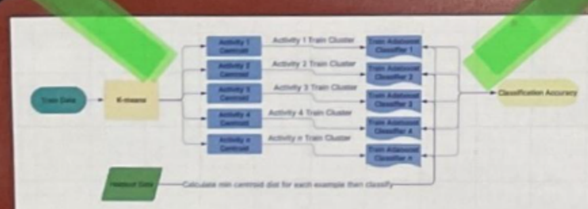

Figure 3: Activity classification to proficiency level classification pipeline architecture. Training data is first sorted into activity clusters using k-means. Next, separate AdaBoost decision tree classifiers are trained on the unique clusters. Finally, holdout data is matched to a classifier using a minimum distance calculation between all training data k-means centroids, and a total classification accuracy is computed from all holdout data predictions.

Project Overview

Research Question

My project explores whether **temporal featurization techniques** and **ML classifiers** can effectively classify **proficiency levels** for **fine motor tasks**—specifically, **piano playing**.

Motivation

I wanted to Investigate if **AI** can support **pedagogy** in **fine motor tasks** such as piano playing, where traditional instruction is often prohibitively expensive.

Input

- Videos of **advanced** and **beginner** pianists
- Featurized using **Hidden Markov Models (HMMs)**, **convolution**, and **flattening**

Key Inquiry

How does temporal data extracted with **HMMs** compare to **convolution** and **flattening** methods?

Activity Classification

Training and holdout data are grouped by **motion trends** to mitigate **classification accuracy** issues arising from contrasting motion patterns.

Output

Models classify holdout examples as either **beginner** or **advanced**, with **accuracy** computed against **ground truth labels**.

Data

Video Collection

- **Hand movement sequences** captured from a **bird's-eye view**, recording both **hands** and **wrists**.
- Data gathered from **in-person performances** and **YouTube videos**, ensuring diversity in **video quality** and **pianist proficiency**.

Metrics

As a part of preprocessing, **videos were converted into multiple 20-image sequences** (each sequence being one data point). Each sequence was labeled with **1** (corresponding to advanced) or **0** (corresponding to beginner)

Training Data

- **Advanced**: 145 unique pianists → 434 sequences
- **Beginner**: 70 unique pianists → 188 sequences

Holdout Data

- **Advanced**: 10 unique pianists → 60 sequences
- **Beginner**: 10 unique pianists → 60 sequences

Featurization & Preprocessing

Landmark Extraction

Utilizes **Google's MediaPipe Hands** to infer **21 3D landmarks** per frame.

Hand Isolation and Orientation Normalization

Left hand sequences were isolated from right hand sequences. The hand orientation was normalized by rotating using the angle between a unit y-axis vector and the wrist-to-middle finger MCP vector. Specifically, compute:

$$v = \text{MVP\_kp} - \text{Wrist\_kp}$$

and

$$\theta = \text{degrees}(\arctan 2(v_y, v_x)).$$

After rotation, the left hand is mirrored horizontally to correspond to the right hand orientation.

Feature Processing Methods

- **Baseline 1: Flattening**: Concatenates landmark coordinates across **20 frames** into a **1260-feature 1D vector**.
- **Baseline 2: Convolution**: Applies a **3×3 low-pass filter** to each sequence, producing a **270-feature 1D vector**.

Temporal Modeling with HMMs

Trains **21 separate HMMs** (one per landmark). Each HMM outputs a sequence probability, resulting in a **21-feature 1D vector** per sequence.

Results

The LDA data table was excluded because it did not yield any improvement in accuracy across data featurization and classification methods. PCA did not significantly improve results; its primary benefit was a reduction in computational load, so PCA results are thinned across tables.

For probability data generated by HMMs, accuracy slightly increased as the number of hidden states increased from 2 to 4 to 6. Consequently, HMM data with 2 and 4 components were excluded from the tables.

GDA performed worse overall; however, regularization improved its accuracy by roughly 10%, achieving up to approximately 62% on non-compressed data. This outcome is likely due to a mismatch between Gaussian assumptions and the complex variability in piano motion data. Notably, GDA performed best on data that had been compressed via PCA.

The **Decision Tree model achieved the highest test accuracy of 75.76%** when training and evaluating on holdout uncompressed convolution data.

The **MLP achieved its highest accuracy at 72.73%** with PCA-compressed convolution data.

The **Activity Classifier Pipeline**, which segments data into activity clusters followed by **AdaBoost** classification, yielded a promising accuracy of **71.97%** on uncompressed convolution data, despite cluster sizes ranging from 10 to over 100 examples.

Table 2: Filtered Confusion Matrices for Classification Methods

Method	Accuracy	Confusion Matrix
6comp HMM MLP	66.67%	[[43 25] [19 45]]
6comp HMM Decision Tree	65.15%	[[38 30] [16 48]]
Baseline Flatten MLP	63.64%	[[31 37] [11 53]]
Baseline Flatten Decision Tree	74.24%	[[37 31] [3 61]]
Baseline Convolve MLP	64.39%	[[30 38] [9 55]]
Baseline Convolve Decision Tree	75.76%	[[37 31] [1 63]]

Table 3: Simplified Performance on Raw Features (Flattened, HMM, Convolution)

Model	Accuracy	Precision	Recall	F1-score
Baseline Flatten Decision Tree	74.24%	80.00%	74.00%	73.00%
Baseline Convolve MLP	64.39%	68.00%	64.00%	63.00%
Baseline Convolve Decision Tree	75.76%	83.00%	76.00%	75.00%
6comp HMM MLP	66.67%	67.00%	67.00%	67.00%
6comp HMM Decision Tree	65.15%	66.00%	65.00%	65.00%
Baseline Convolve GDA (Test, reg)	60.61%	56.00%	87.50%	68.29%
6comp HMM GDA (Test, reg)	62.88%	60.87%	65.62%	63.16%
Baseline Flatten GDA (Test, reg)	61.32%	69.15%	40.88%	51.38%

Table 4: Simplified Performance on PCA-Based Features

Model	Accuracy	Precision	Recall	F1-score
PCA Conv GDA (Test)	64.39%	57.94%	96.88%	72.51%
PCA Conv MLP	72.73%	74.00%	73.00%	72.00%
PCA Conv Decision Tree	68.18%	68.00%	68.00%	68.00%
PCA 6comp Decision Tree	63.64%	64.00%	64.00%	64.00%
PCA Flattened MLP	68.94%	77.00%	69.00%	67.00%

Discussion

The decision tree likely obtained the highest accuracy because it does not make any assumptions about the dataset and was provided with a rich set of features from both convolution and flattened data to determine the optimal separation threshold. In contrast, lower accuracy was expected with HMMs because the multiple differing motion sequences—such as scales, arpeggios, and chord progressions—likely did not yield highly generalizable hidden states for holdout data probabilities.

It is noteworthy that **HMMs achieved nearly 70% accuracy without any prior activity classification.** Equally surprising was the high accuracy obtained when performing activity classification before proficiency level classification, despite each activity classifier being trained on much smaller subsets of data after segmentation.

Overall, these findings confirm that temporal feature extrac- tion from pianist hand movements preserves the fine-grained details necessary for proficiency level classification. However, further improvements in data quality, feature representation, and model refinement are needed before this methodology can be deployed in real-world pedagogical applications.

HMM Forward Algorithm for Pianist Hand Movement Analysis

Abstract

To model the complexity of pianist hand movements, **Hidden Markov Models (HMMs)** were trained separately for each of the **21 key hand landmarks**. Each HMM estimates transition probabilities between states corresponding to distinct hand positions over time. The **final probability output** of the Forward Algorithm quantifies the likelihood that a given landmark sequence was generated by an advanced pianist.

Initialization

$$\alpha_1(j) = \pi_j \, b_j(o_1)$$

Each state j is initialized by multiplying its prior probability π_j with the likelihood of emitting the first observed hand position o_1 [1].

Recursion

$$\alpha_t(j) = \left(\sum_{i=1}^{N} \alpha_{t-1}(i) \, a_{ij} \right) b_j(o_t)$$

At time t, the forward probability $\alpha_t(j)$ aggregates weighted contributions from all previous states i, determined by transition probabilities a_{ij}, and is scaled by the emission probability $b_j(o_t)$ [1].

Termination

$$P(O \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

The total likelihood of an observation sequence O under the model λ is obtained by summing the final forward probabilities across all states [1].

HMM Gaussian Emission Parameter Updates

In my Hidden Markov Models (HMMs), each hidden state j generates observations that follow a Gaussian (normal) distribution. The parameters that define this distribution—its mean and covariance—are updated as follows:

$$\mu_j = \frac{\sum_{t=1}^{T} \gamma_t(j) \, O_t}{\sum_{t=1}^{T} \gamma_t(j)} \quad \text{and} \quad \Sigma_j = \frac{\sum_{t=1}^{T} \gamma_t(j) \, (O_t - \mu_j)(O_t - \mu_j)^T}{\sum_{t=1}^{T} \gamma_t(j)}$$

where:

- $\gamma_t(i) = P(S_t = i \mid O, \lambda)$: the probability that the hidden state at time t is i given the observations O and model parameters λ.
- O_t: the observation at time t.
- T: the total number of observations.
- μ_j: the mean of the Gaussian distribution for state j.
- Σ_j: the covariance matrix of the Gaussian distribution for state j.
- λ: the set of all model parameters, including transition probabilities and emission parameters.
- S_t: the hidden state at time t.

Interpretation of Hidden States

- **HMM Training:** HMMs were trained using **2**, **4**, and **6 hidden states** across separate experiments.
- **Output:** After training, the HMMs have discovered the best states to use to evaluate sequence probabilities for holdout data. The state means provide interpretable representations of hand positions.

State	Mean Coordinates (x, y, z)	Description
0	(0.1011, −0.9907, −0.2943)	High negative y-value (wrist farther from the black keys).
1	(1.0340, 0.4431, 0.1310)	High positive x-value (wrist toward higher registers).
2	(−0.3389, 0.4405, −0.1697)	Relatively central wrist position.
3	(−0.2379, −0.0757, 1.0290)	High z-value (raised wrist position).

Table 1: State Means and Descriptions for Wrist (Landmark 0) HMM with Four Hidden States.

Future Work

Future work should focus on:

- **Expanded Activity-Specific Datasets:** Incorporate manually labeled performance sequences for distinct technical exercises (e.g., scales, arpeggios, chord progressions) rather than relying solely on k-means clustering.
- **Deep Learning-Based Feature Extraction:** Employ advanced architectures such as CNNs or stacked hourglass networks to improve keypoint extraction, potentially expanding beyond the current 21 landmarks.
- **Multiclass Proficiency Classification:** Transition from a binary system to a multiclass framework (e.g., beginner, intermediate, advanced, expert) to better capture the spectrum of pianist skills.

References

[1] Jurafsky, D., & Martin, J. H. (2024). *Speech and Language Processing* (3rd ed. draft), Appendix A. Available at: https://web.stanford.edu/~jurafsky/slp3/

[2] Stanford University. (2024). CS229 Lecture Notes: Generative Learning Algorithms (Section 2.3: Gaussian Discriminant Analysis). [Online]. Available: https://cs229.stanford.edu/main_notes.pdf