AI-Powered Dance Coaching via Pose Estimation, Vision Transformers and Dynamic Time Warping

Roshen Nair
Stanford University
roshen03@stanford.edu

Arnold Yang
Stanford University
arnoldya@stanford.edu

Henry Zhou
Stanford University
henryjz@stanford.edu

Abstract

Learning dance without expert guidance often leads to ingrained errors and limited progress due to lack of real-time, personalized feedback. We introduce an end-to-end AI dance feedback pipeline that segments a user's performance into 19 primitive figures using MoveNet and a Vision Transformer, aligns each segment to professional reference videos via Dynamic Time Warping (DTW), and generates real-time corrective feedback through a Gemini-based recommender module. Unlike prior systems that enforce exact choreography matching and offer no actionable coaching, our method tolerates stylistic variations and changes in move order while delivering personalized guidance.

Our dataset consists of 32 selected ballet videos annotated into 145 figure segments. We preprocess by extracting 17 keypoints per frame with MoveNet, translating to a hip-centered coordinate system scaled by torso length, and applying random rotations, dropout, temporal scaling, and Gaussian noise augmentations to improve robustness. Although we applied our model to ballet (leveraging our specific dataset and preprocessing pipeline), it can be readily extended to other styles of dance. On this dataset, our Vision Transformer achieves 92.5% *training accuracy and* 82.2% *validation accuracy. In a 10-user study, feedback from our system boosts mean user accuracy from* 55.9% *to* 66.8%*, with* 85% *of feedback judged coherent. We complement these results with quantitative analyses (confusion matrices, accuracy scores, training vs. validation graphs) and qualitative examples of feedback and failure cases, and we discuss extensions to possible future work in the field.*

1. Introduction

Learning complex dance movements without expert guidance often leads to frustration and ingrained bad habits. Instructors and live classes can be expensive and/or geographically inaccessible, creating a barrier for learners seeking real-time, personalized feedback. An AI-powered coach that provides move-level guidance could make dance education more accessible by delivering instant corrections, preventing user errors from becoming bad habits, and reducing reliance on face-to-face instruction.

Our motivation stems from bridging the gap between passive video tutorials and active coaching. Prior systems typically perform rigid, frame-by-frame pose comparisons (often with CNN or RNN based underlying structures) and offer limited insight into user mistakes. They struggle with variations in timing, style, and move order. In contrast, modern transformers excel at capturing long-range dependencies in video, and sequence-alignment techniques like Dynamic Time Warping (DTW) can accommodate temporal shifts. We leverage these advances to build a robust, flexible feedback loop.

The input to our algorithm is a user-recorded RGB video of a dance performance (30 frames per second, variable length) and a reference database of professional ballet videos annotated with primitive figure segments. We then use MoveNet for 2D pose extraction, a Vision Transformer (ViT) trained on our reference dataset for sliding-window figure classification, and FastDTW to align each predicted segment to its closest exemplar. Our system outputs an overall accuracy score for the user input video and generated feedback for each of the user's moves that stray too far away from the reference videos for that figure segment (also compiled into a finalized video with feedback format). The User Interface presents this in a side-by-side visualization comparing user and reference performances, with an overlaid concise caption describing the deviation from the reference and offering concrete steps for the user to improve. Figure 1 provides a visualization of the output of our model.

2. Related Work

Research on dance-focused pose estimation and recognition has advanced significantly in recent years. Hu and

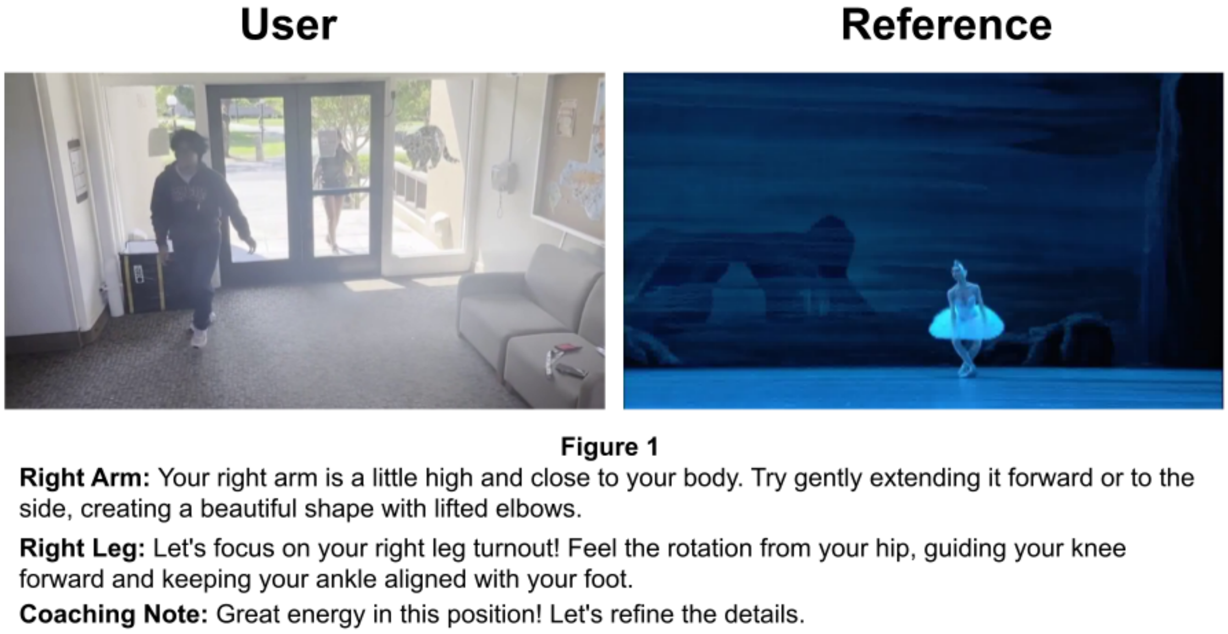

Figure 1

Right Arm: Your right arm is a little high and close to your body. Try gently extending it forward or to the side, creating a beautiful shape with lifted elbows.

Right Leg: Let's focus on your right leg turnout! Feel the rotation from your hip, guiding your knee forward and keeping your ankle aligned with your foot.

Coaching Note: Great energy in this position! Let's refine the details.

Figure 1. Sample of comparison image for the model, expected output will be annotated video.

Ahuja proposed an unsupervised 3D pose estimation framework that can track dancers, estimate 3D poses, and infer camera parameters without necessarily required ground-truth 3D data, enabling hierarchical dance video recognition under what normally would be considered very difficult settings [12]. Venkatrayappa et al. discuss recent 3D human body pose and shape estimation methods applied to dance, finding that multi-frame approaches like PHALP outperform single-frame models by using temporal context to handle rapid and more complex movements [22]. In addition, Zhang et al. introduce YeLan, a high-frequency 3D pose estimation system using event cameras, demonstrating robust performance in low-light and high-dynamic scenarios (which can be especially helpful as these features are quite common to settings where dance performances typically take place) [25]. While these works push the boundaries of 3D motion capture, they require specialized sensors or multi-view setups, limiting the accessibility of such a model to more casual learners worldwide.

Also, recently, interactive systems designed for dance practice provide insights into visualizing feedback for dance learners. Zhou et al.'s SyncUp offers interactive visualizations of pose similarity and temporal alignment to support dance practice, highlighting segments and body parts needing improvement in terms of synchronization across dancers [26]. Bera et al.'s SYD-Net benchmarks several sports, yoga, and dance postures by integrating patch-based attention on CNN structures, achieving great accuracy on newly created dance image datasets [6]. Both systems emphasize visual analysis but do not generate explicit, move-level coaching feedback or handle arbitrary move ordering beyond a set choreography.

Beyond dance, pose-based coaching applications in other domains illustrate the potential of AI for coaching related tasks. Cheng et al. present MatchPoint, which matches beginner tennis strokes to professional reference ones using keypoint extraction and k-NN, enabling players to discover how to become more like certain pro tennis players [9]. Alluri et al. automate powerlifting judging by detecting key joint angles to check how valid a lift is, showing high agreement with human referees [4]. Qu applies pose estimation to rock climbing, extracting movement patterns to identify technical errors and recommend adjustments [18]. These works show how pose comparisons for feedback can have widespread applications. However, they focus on more largely rigid, single-action sports and activities, while dance typically should allow for more stylistic variation and sequential choices.

In addition, advances in temporal feature modeling have more recently allowed for better sequence comparison. Liu et al. develop a multi-frame human pose estimation method that matches temporal features, improving accuracy on video frames even with faster motions [14]. Müeller discusses Dynamic Time Warping (DTW) to match different time series under variable speed, and this work under-

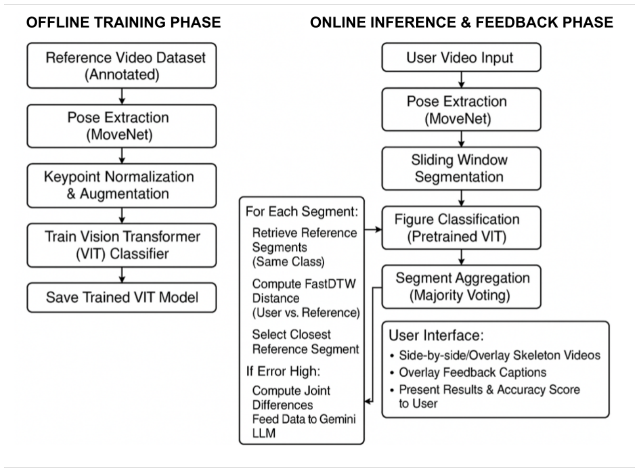
OFFLINE TRAINING PHASE
Reference Video Dataset (Annotated)
Pose Extraction (MoveNet)
Keypoint Normalization & Augmentation
Train Vision Transformer (VIT) Classifier
Save Trained VIT Model
ONLINE INFERENCE & FEEDBACK PHASE
User Video Input
Pose Extraction (MoveNet)
Sliding Window Segmentation
Figure Classification (Pretrained VIT)
Segment Aggregation (Majority Voting)
User Interface:
• Side-by-side/Overlay Skeleton Videos
• Overlay Feedback Captions
• Present Results & Accuracy Score to User
For Each Segment:
Retrieve Reference Segments (Same Class)
Compute FastDTW Distance (User vs. Reference)
Select Closest Reference Segment
If Error High:
Compute Joint Differences
Feed Data to Gemini LLM

Figure 2. Complete system workflow: (1) Offline training of ViT classifier on labeled reference dataset, (2) Online processing of user video through pose extraction, windowing, and classification, (3) DTW alignment with reference examples, (4) Loss computation and feedback generation via Gemini recommender system.

lies many efficient, windowed DTW implementations used in motion analysis [15]. These techniques influenced our choice of FastDTW for matching segments.

Our work builds on these foundations by combining real-time 2D pose extraction with transformer-based sliding-window classification and DTW-based segment matching to deliver effective, move-level feedback. Unlike 3D or multi-view systems, we rely on a single RGB camera and light preprocessing. Compared to platforms that only support visualizations, we also add the coaching element to the model for the segments / moves that are most misaligned or different from the reference dataset. In addition, by tolerating stylistic and speed variations, our system supports flexible learning, advancing from existing models that do rigid sequence matching to more personalized dance coaching.

3. Methods

3.1. System Overview and Workflow

Figure 2 illustrates our end-to-end pipeline for AI-powered dance feedback. Our system operates in two distinct phases: an offline training phase and an online inference phase. During training, we use our annotated reference dataset of 32 ballet videos (145 labeled figure segments) to train a Vision Transformer classifier that learns to identify 19 distinct most common ballet dance moves / figures. During inference, we process user-uploaded videos through pose extraction, figure segmentation and classification, DTW-based alignment with reference dataset videos, and finally generate personalized feedback through our recommender module.

The offline training phase begins with our pre-labeled reference dataset, where each video segment is annotated with its corresponding figure class (1-19). We extract pose keypoints using MoveNet [3, 23], normalize coordinates, and train a Vision Transformer to classify minimum 24-frame windows into figure categories. This trained model is then saved for later use.

During online inference, user-uploaded videos follow a similar preprocessing pipeline: pose extraction via MoveNet, coordinate normalization, and sliding-window segmentation. Each window is classified using our trained ViT model, with predictions chosen through majority voting to produce figure segments (allowing for the model to also classify the movement with a value of -1 if its confidence is not above a certain threshold value of 0.6 to identify the figure as none-of-the-above: not a standard ballet dance move but instead some other action like sitting or standing). For each identified segment, we compute DTW distances to all reference examples of the same figure class, selecting the closest match. The resulting DTW loss feeds into both our accuracy scoring function and our Gemini-based recommender system, which generates targeted feedback captions and visualizations for the user interface.

3.2. Vision Transformer for Figure Classification

We use a Vision Transformer (ViT) architecture [10] adapted for pose sequence classification. ViTs leverage self-attention mechanisms to capture long-range dependencies across input tokens, making them well-suited for understanding complex movement patterns that span multiple frames (like those common to dance). Unlike convolutional approaches that process local neighborhoods, transformers can directly model relationships between distant time steps, which is especially useful for this task of recognizing dance figures that involve coordination of full-body movements over extended periods.

Our ViT implementation is built upon the standard transformer encoder architecture from Vaswani et al. [21]. In use, it is given a pose sequence window $X \in \mathbb{R}^{T\times D}$ of $T = 24$ frames and $D = 35$ features (17 keypoints times 2 coordinates + 1 time feature) to start with, we first apply a linear embedding layer to project each frame to a $d_{model} = 128$ dimensional space:

$$h_0 = XW_e \tag{1}$$

where $W_e \in \mathbb{R}^{35\times 128}$ is the embedding matrix.

The embedded sequence passes through $L = 3$ transformer encoder layers, each containing multi-head self-attention (with $h = 8$ attention heads) followed by a position-wise feed-forward network:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (2)$$

$$\text{MultiHead}(X) = \text{Concat}(\text{head}_1, ..., \text{head}_h)W^O \quad (3)$$

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \quad (4)$$

The final transformer output is passed through a single linear classification layer that produces per-frame logits $Z \in \mathbb{R}^{24\times19}$ for our 19 figure classes. We train using cross-entropy loss averaged over all frames in each window:

$$\mathcal{L}_{\text{cls}} = -\frac{1}{T}\sum_{t=1}^{T} \log \text{softmax}(Z_t)[y_t] \quad (5)$$

where $Z_t \in \mathbb{R}^{19}$ represents the logit vector for frame t, y_t is the ground truth class label for frame t, and $T = 24$ is the window size.

Our ViT implementation extends existing transformer libraries [24] with custom preprocessing for pose sequences. We built the training pipeline, data loading, and model architecture changes, while using existing core transformer implementation and PyTorch [17].

3.3. User Video Processing Pipeline

When a user uploads a dance video, we extract pose keypoints using Google's MoveNet model [3], a CNN-based pose estimator designed for real-time applications. MoveNet uses a MobileNetV2 backbone [23] with specialized output heads to predict 17 body keypoints with precision. Each keypoint includes (x, y) coordinates and a confidence score, robust to varied lighting conditions and camera angles commonly encountered in user content.

Then, we normalize extracted poses using our torso-length scaling approach: coordinates are translated to place the hip midpoint at the origin, then scaled by the distance between hip and shoulder midpoints. This normalization removes dependencies on subject size, camera distance, and frame positioning:

$$\text{hip\_center} = \frac{\text{left\_hip} + \text{right\_hip}}{2} \quad (6)$$

$$\text{shoulder\_center} = \frac{\text{left\_shoulder} + \text{right\_shoulder}}{2} \quad (7)$$

$$\text{torso\_length} = \|\text{shoulder\_center} - \text{hip\_center}\|_2 \quad (8)$$

$$\text{keypoint\_norm} = \frac{\text{keypoint} - \text{hip\_center}}{\text{torso\_length}} \quad (9)$$

For temporal segmentation, we apply a sliding window approach with minimum window size $w = 24$ frames and stride $s = 5$ frames. Each window captures at least 1 second of motion at 30 fps, thus it is able to judge figures across a longer video rather than jumping between predictions. We append a normalized time feature $t \in [0, 1]$ to each frame within a window to help the model distinguish between positions across time. Windows are classified independently using our trained ViT, then aggregated via majority voting to assign figure labels to individual frames. Consecutive frames with identical labels are merged into coherent segments representing complete dance figures.

Figure classification uses a confidence threshold $\tau = 0.6$: segments with maximum prediction confidence below this threshold are given label -1, representing transitional movement or non-dance content (like just sitting/standing). This prevents false positive classifications during periods when users are preparing for the next move or making adjustments.

3.4. Dynamic Time Warping Alignment

For each classified user segment, we employ Dynamic Time Warping (DTW) [7] to find the best-matching reference example from our annotated dataset. DTW computes an optimal alignment between two temporal sequences by allowing flexible stretching and compression along the time axis, accommodating natural variations in movement speed and timing that occur between different dancers.

Given a user segment $S = \{s_1, s_2, ..., s_m\}$ and reference example $R = \{r_1, r_2, ..., r_n\}$, DTW constructs a cost matrix $C \in \mathbb{R}^{m\times n}$ where $C_{i,j} = ||s_i - r_j||_2$ represents the Euclidean distance between normalized pose vectors. The algorithm finds an optimal warping path $W = \{w_1, w_2, ..., w_K\}$ where each $w_k = (i_k, j_k)$ satisfies some constraints related to monotonicity and continuity (and DTW(S,R) is calculated by taking the sum of the costs across the warping path minimized across all possible candidates for the warping path):

$$\text{DTW}(S, R) = \min_W \sum_{k=1}^{K} C_{w_k} \quad (10)$$

$$\text{subject to:} \quad w_1 = (1, 1), \quad w_K = (m, n) \quad (11)$$

$$w_{k+1} - w_k \in \{(1, 0), (0, 1), (1, 1)\} \quad (12)$$

We use the FastDTW implementation [2, 19] which reduces computational complexity from $O(mn)$ to approximately $O(m + n)$ through multilevel dynamic programming and radius-constrained warping. This optimization enables real-time processing of user videos without sacrificing alignment quality.

For each user segment of predicted class c, we compute

DTW distances to all reference segments labeled with class c, selecting the reference example with minimum cost as the best match. This distance serves as our primary error metric, with lower values indicating closer alignment to professional performance.

3.5. Loss Computation and Accuracy Scoring

Our system converts raw DTW distances into interpretable accuracy scores using an exponential decay function designed to emphasize the practical difference between good and poor performance. Given a DTW distance d between user segment and reference exemplar, we compute:

$$\text{Accuracy} = e^{-d/100} \tag{13}$$

This formulation maps DTW distances to the range $(0, 1]$, where distances near 0 yield accuracies approaching 100%, while larger distances asymptotically approach 0%. The scaling factor of 100 were empirically tuned based on the distribution of DTW distances in our reference dataset, where typical distances between reference videos range from 50-150 and problematic user performances often exceed 300.

For videos containing multiple figure segments, we compute overall accuracy as the mean of individual segment accuracies, weighted by segment duration:

$$\text{Video\_Accuracy} = \frac{\sum_{i=1}^{N} \text{duration}_i \times \text{accuracy}_i}{\sum_{i=1}^{N} \text{duration}_i} \tag{14}$$

This weighting ensures that longer, more complex figures contribute proportionally to the overall assessment, preventing brief transitions from dominating the score.

We also track per-figure accuracy statistics and identify the segments / moves with highest DTW distances as the primary mistakes of the user and therefore focus on that for feedback generation. These segments represent the user's most significant deviations from professional technique, so our model then provides detailed guidance through our recommender system.

3.6. Feedback Generation via Gemini

Our recommender system uses Google's Gemini large language model [5] to translate quantitative pose differences into qualitative coaching feedback. For figures identified as requiring correction through our DTW analysis pipeline, we extract key representative frames from both user and reference sequences corresponding to the moments of maximum deviation.

Our feedback generation process operates on pre-computed DTW alignments stored with unique identifiers. When a user segment receives a DTW distance indicating significant error (above a certain threshold of 10), the system identifies the specific frame within that segment where the deviation is most pronounced. We then locate the corresponding reference frame from the best-matching reference sequence using the DTW alignment path.

The Gemini model processes structured pose difference data organized by body regions (with arms, legs, torso, and head positions) along with contextual information about the specific ballet figure being performed. Our prompt engineering framework provides detailed templates that describe the context of the dance move and highlight specific patterns in terms of joint errors and how to fix them.

Gemini generates personalized feedback structured as JSON structured objects containing figure-specific guidance for each body region. Our visualization pipeline combines this textual feedback with side-by-side frame comparisons, displaying the user's performance alongside the reference performance at the moment of maximum error. The system extracts frames using computed indices from the DTW alignment process, ensuring that users see exactly where their technique diverges from what professional did. Feedback text is overlaid beneath the visual comparison, creating an integrated coaching experience that combines visual and textual guidance.

3.7. User Interface and Visualization

Our user / command-line interface [16] presents feedback through synchronized video playback and overlay visualizations. Users view their original performance alongside the best-matching reference segment, with pose skeletons rendered in real-time using OpenCV graphics. Feedback captions appear as timed overlays synchronized to specific movement phases, allowing users to associate verbal guidance with visual cues.

Our codebase builds extensively on existing libraries: MoveNet pose extraction [3, 23], PyTorch transformers [11, 17, 24], FastDTW alignment [2, 19], OpenCV visualization [16], and Gemini API integration [5]. Our core contributions include the ViT training pipeline, DTW-based matching system, accuracy scoring functions, and integrated user interface, while using these established tools for a new application of computer vision and machine learning operations.

4. Dataset & Features

4.1. Dataset Sources

- **Let's Dance Dataset** [13]: 1000 dance videos from 10 different dance forms, including ballet and waltz, collected as part of a research study from GeorgiaTech. Includes pre-extracted pose estimation keypoints.

- **Steezy Online Platform** [20]: Additional curated professional dance videos. The data is obtained under educational fair use and was preprocessed with video editing and MoveNet for more high quality training purposes.

4.2. Reference Dataset Composition

We found 32 high-resolution ballet videos: 27 from the Let's Dance Dataset [1] and 5 from Steezy Studio [20], and then annotated the figures / moves within each section by hand, totaling 145 annotated figure segments. Videos are 1080×1920 resolution at 30 fps, with individual figure segments / moves averaging 1.83 seconds (55 frames). Our 19 classes of figures includes 7 directional variants to capture nuanced movement differences:

Class	Name	Description
1L,1R	Spin	Clockwise (L) / Counter-clockwise (R) whole-body rotation
2L,2R	LegLift	Raise left/right leg with arm extension
3L,3R	JumpDir	Elegant jump moving left/right
4L,4R	ReachBendLeg	Reach forward, bend torso, extend leg left/right
5	FloorMove	Ground-based movements (rolls, floor spins)
6L,6R	Flap	Full-body undulation left/right
7	ReachSeq	Sequential right-left arm reaches
8	StraightJump	Vertical jump arms overhead
9L,9R	JumpOut	Directional jump with outstretched limbs
10	ArmSpread	Symmetrical arm extension
11L,11R	KneeBend	Single-knee bend reaching left/right
12	StandUp	Transition from ground to standing

Table 1. 19 figure classes with 7 directional variants (L/R). See CSV attachment for full segment annotations.

4.3. Data Splits & Preprocessing

We split reference data into 116 training and 29 validation segments (80/20 dataset split proportions). Test set comprises 20 user-recorded videos performing ballet routines. All data undergoes the following:

Normalization: Pose keypoints translated to hip midpoint origin and scaled by torso length (hip-to-shoulder distance) using:

$$\hat{\mathbf{p}}_i = \frac{\mathbf{p}_i - \mathbf{c}_{\text{hip}}}{\|\mathbf{c}_{\text{shoulder}} - \mathbf{c}_{\text{hip}}\|_2}, \tag{15}$$

where $\mathbf{p}_i$ is raw keypoint and $\mathbf{c}$ are joint midpoints.

Augmentation: Randomly applied (with independent probability 0.7 for each type of augmentation) to training segments from the reference dataset: We used data augmentation to account for noise and increase the amount of data we have, also helping prevent overfitting [8].

- Random rotations of reference video (with angle of rotation about origin in range [-0.2, 0.2]). We selected a standard rotation range to accommodate various camera angles and found this range worked best.

- Joint dropout with dropout probability of 0.05. We dropped joints to account for occlusions.

- Temporal scaling ($\pm 20\%$ via linear interpolation). We used this to account for faster and slower video inputs, and found this range to work best.

- Gaussian noise ($\sigma = 0.01$) on keypoint coordinates. We used this to account for keypoint noise from Movenet, video quality, and the user.

4.4. Feature Representation

We use MoveNet's 17-keypoint outputs transformed into 35 dimensional vectors (x,y per joint plus one time feature). Each 24-frame window concatenates pose vectors with normalized timestamps, forming 24×35 input tensors.

5. Experiments, Results, & Discussion

The metrics reported in this section are based on the calculations detailed in the Methods section above. For hyperparameter selection, we validated across multiple configurations for our Vision Transformer (ViT) model and ultimately selected the model that achieved the highest validation accuracy while best preventing overfitting. After performing validation across 512 combinations of hyperparameter choices, the final hyperparameters we selected for our model are as follows: embedding dimension (embed) $= 128$, number of attention heads $= 8$, depth $= 3$, learning rate (lr) $= 0.0001$, optimizer = Adam, batch size (bs) $= 16$, number of augmentations (aug) $= 200$, window size (win) $= 24$, stride $= 5$, and tolerance threshold (tol) $= 0.6$.

As noted earlier, augmentations were crucial in reducing overfitting, a significant issue in initial versions of our model. The final model showed an acceptable overfitting level, balancing high training accuracy with good validation generalization. These hyperparameter choices were guided both by standard ViT training best practices and extensive testing across hundreds of configurations.

Figure 3 presents the overlaid training and validation loss and accuracy curves for our best-performing ViT model.

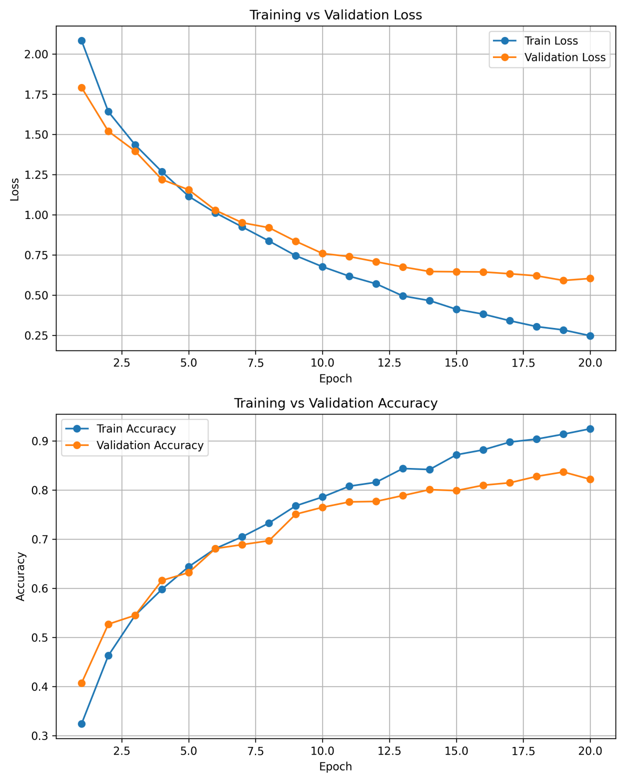

Figure 3. ViT Training/Validation Loss and Accuracy Graphs

As shown in Figure 3, after 20 epochs, the training accuracy exceeded 90%, and the validation accuracy reached 82.2%, indicating strong generalization. The global (frame-weighted) accuracy across all videos was 67.97%, a solid performance given the complexity of figure classification.

Figure 4 shows the confusion matrix for our trained ViT model. It demonstrates that the model can classify ballet figures with reasonably high accuracy, closely aligning predictions with ground truth labels.

These results indicate that our ViT model is sufficiently trained and can be reliably used in downstream tasks of the project.

After training the ViT model, we evaluated the complete project workflow, including the online system that processes user-uploaded videos using MoveNet, the trained ViT classifier, fastDTW, and the Gemini recommender module.

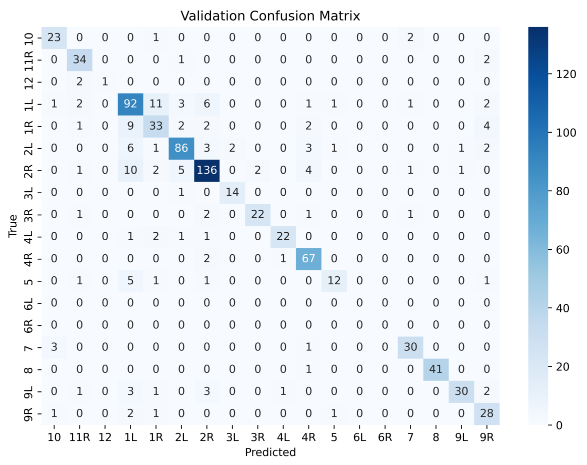

Figure 4. ViT (Transformer) Confusion Matrix

We conducted both quantitative and qualitative evaluations. For the quantitative analysis, we ran a user study measuring mean accuracy improvements before and after using the AI coach.

User	Without Coach Acc.	Post-Coach Acc.
U1	0.366	0.728
U2	0.130	0.379
U3	0.465	0.699
U4	0.769	0.670
U5	0.714	0.733
U6	0.826	0.863
U7	0.457	0.502
U8	0.863	0.815
U9	0.584	0.699
U10	0.415	0.595
Average	**0.559**	**0.668**

Table 2. User accuracy (as $e^{-\text{error\_score}/100}$) rounded to thousandths place before and after using the AI coach.

As shown in Table 2, the average improvement across 10 users was **10.9%**. This reflects a significant increase compared to the 5.1% improvement seen with our baseline model (which did not use a Vision Transformer).

We also attached a video demonstration in the project submission, further supporting the model's capability in detecting performance issues and delivering actionable, personalized feedback.

Qualitatively, the captioning system's generated feedback was evaluated for reasonableness. Out of 20 test videos, the feedback was deemed accurate (compared to expert ground truth) in 17/20 cases (with some hallucinations naturally present due to the commonalities between dance moves / figures). This was measured by checking for hallucinations and consistency with known user mistakes.

Overall, our results confirm both the effectiveness of the trained ViT model in accurately classifying ballet figures, and the complete project pipeline's success in delivering useful, personalized feedback that enhances user performance, the primary goal of our project.

6. Conclusion / Future Work

In this work, we presented an effective pipeline for dance figure recognition and alignment using a supervised Transformer-based classifier, MoveNet, and DTW-based comparison within identified figure classes. Our current approach streamlines and significantly improves upon our earlier attempts (particularly compared to our initial sliding-window DTW strategy and various unsupervised figure discovery methods), which proved either computationally too expensive or unreliable due to the complexity and amount of variation that is present across dance. The results are promising for the framework, the feedback seems to be genuinely useful and accurate.

Looking ahead, we envision some major extensions to enhance both the usability and accuracy of the system. First, upgrading from MoveNet pose keypoint detection to get more fine-grained user movement information like foot orientation, etc. Thus, creating more specific feedback for the user. Second, integrating real-time feedback via live video would allow frame-by-frame analysis and near-instantaneous figure classification, enabling actionable guidance during rehearsals. Third, moving from 2D to 3D pose estimation would allow the model to better account for depth, capturing subtle but important aspects of dancer posture. Fourth, expanding to multi-person detection and tracking would broaden the system's applicability to group rehearsals and partner dances, enabling individualized feedback even in shared scenes. Fifth, expanding our professional video dataset to allow for more accurate pose/figure recognition and processing.

All together, these future directions reflect our commitment to building a system that is not only technically accurate but also practical and enriching for dancers seeking meaningful, real-time improvement.

7. Contributions & Acknowledgements

Special thanks to the CS231N class and specifically our mentor Jiaman Li for the support and guidance throughout the project. For this project, we all worked together on the actual code and write-up, and discussed all key sections / methods we used. Henry helped with setting up the OpenCV and MoveNet, processing the dataset, creating and training all four iterations of the Vision Transformer model, user classification and segmentation, and DTW comparisons. Arnold helped with training the Vision Transformer model, tuning figure comparisons, post-processing, the recommender module, and extracting dataset. Roshen helped with extracting dataset, training Vision Transformer model, and the research behind the fast-DTW, Transformer model, and MoveNet.

In addition, here is a list of Github repos which we found useful to our project pipeline and used along the way:

- Albumentations Official repo: https://github.com/albumentations-team/albumentations
- Gym (OpenAI Gym) Official repo: https://github.com/openai/gym
- Matplotlib Official repo: https://github.com/ matplotlib/matplotlib
- NumPy Official repo: https://github.com/numpy/numpy
- OpenCV (cv2) Official repo: https://github.com/opencv/opencv
- Pandas Official repo: https://github.com/pandas-dev/pandas
- Pillow (PIL) Official repo: https://github.com/python-pillow/ Pillow
- PyTorch Official repo: https://github.com/pytorch/ pytorch
- PyTorch Lightning Official repo: https://github.com/Lightning-AI/pytorch-lightning
- PyYAML (yaml) Official repo: https://github.com/yaml/ pyyaml
- Requests Official repo: https://github.com/psf/ requests
- scikit-image Official repo: https://github.com/scikit-image/ scikit-image
- scikit-learn Official repo: https://github.com/scikit-learn/ scikit-learn
- Seaborn Official repo: https://github.com/mwaskom/seaborn
- Segmentation Models PyTorch Official repo: https://github.com/qubvel-org/segmentation_models .pytorch
- TensorFlow Official repo: https://github.com/ tensorflow/tensorflow
- TorchMetrics Official repo: https://github.com/Lightning-AI/torchmetrics

- TorchVision Official repo: https://github.com/pytorch/vision
- tqdm Official repo: https://github.com/tqdm/tqdm
- Transformers (Hugging Face) Official repo: https://github.com/huggingface/transformers
- Weights & Biases (wandb) Official repo: https://github.com/wandb/wandb

References

[1] Let's dance dataset. `https://sites.cc.gatech.edu/cpl/projects/dance/`. Georgia Tech College of Computing.

[2] Fastdtw- dynamic time warping (dtw) with a linear time and memory complexity. `https://github.com/rmaestre/FastDTW`, 2019.

[3] Movenet: Ultra fast and accurate pose detection model. `https://www.tensorflow.org/hub/tutorials/movenet`, 2024.

[4] R. Alluri, T. K. Martheswaran, and R. K. Thomas. Automating powerlifting judging through keypoint detection. CS231n Project Report, 2024.

[5] R. Anil, S. Borgeaud, Y. Assogba, J.-B. Alayrac, J. Dean, O. Vinyals, K. Kavukcuoglu, D. Hassabis, et al. Gemini: A family of highly capable multimodal models, 2023. Accessed: 2025-06-04.

[6] A. Bera, M. Nasipuri, O. Krejcar, and D. Bhattacharjee. Fine-grained sports, yoga, and dance postures recognition: A benchmark analysis. *IEEE Transactions on Instrumentation and Measurement*, 72:1–13, 2023.

[7] D. J. Berndt and J. Clifford. Using dynamic time warping to find patterns in time series. In *Proceedings of the AAAI Workshop on Knowledge Discovery in Databases (KDD-94)*, pages 359–370. AAAI Press, 1994.

[8] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, and A. Kalinin. Albumentations: Fast and flexible image augmentations. `https://github.com/albumentations-team/albumentations`, 2020.

[9] E. Cheng, R. Dange, and I. Gorelik. Matchpoint: Your computer vision tennis coach. CS231n Project Report, 2024.

[10] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021.

[11] W. Falcon et al. Pytorch lightning. *GitHub repository*, 2019.

[12] X. Hu and N. Ahuja. Unsupervised 3d pose estimation for hierarchical dance video recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 13970–13979, 2021.

[13] G. T. C. P. Lab. Dance dataset project, n.d. Georgia Tech, `https://sites.cc.gatech.edu/cpl/projects/dance/`.

[14] Z. Liu, R. Feng, H. Chen, S. Wu, Y. Gao, Y. Gao, and X. Wang. Temporal feature alignment and mutual information maximization for video-based human pose estimation. In *arXiv preprint arXiv:2203.15227*, 2022.

[15] M. Müller. Dynamic time warping. In *Information Retrieval for Music and Motion*, pages 69–84. Springer, 2007.

[16] OpenCV team. Opencv: Open source computer vision library. `https://github.com/opencv/opencv`, 2000. Version X.X, accessed 2025-06-04.

[17] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019.

[18] J. Qu. Using pose estimation to analyze rock climbing technique. CS231n Project Report, 2024.

[19] S. Salvador and P. Chan. Toward accurate dynamic time warping in linear time and space. In *KDD Workshop on Mining Temporal and Sequential Data*, pages 70–80, 2004.

[20] S. Studio. Steezy studio - online dance classes with world-class instructors, n.d. `https://www.steezy.co/`.

[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need, 2023.

[22] D. Venkatrayappa, A. Tremeau, D. Muselet, and P. Colantoni. Survey of 3d human body pose and shape estimation methods for contemporary dance applications. *arXiv preprint arXiv:2401.02383*, 2024.

[23] R. Votel and N. Li. Next-generation pose detection with movenet and tensorflow.js. https://blog.tensorflow.org/2021/05/next-generation-pose-detection-with-movenet-and-tensorflowjs.html, 2021.

[24] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush. Transformers: State-of-the-art natural language processing. `https://github.com/huggingface/transformers`, 2020. Accessed: 2025-06-04.

[25] Z. Zhang, K. Chai, H. Yu, R. Majaj, F. Walsh, E. Wang, U. Mahbub, H. Siegelmann, D. Kim, and T. Rahman. Neuromorphic high-frequency 3d dancing pose estimation in dynamic environment. *Neurocomputing*, 547:126388, August 2023.

[26] Z. Zhou, A. Xu, and K. Yatani. Syncup: Vision-based practice support for synchronized dancing. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 5(3):1–25, 2021.