Extending nano4M for Video Generation for human poses.

Ghali Berbich (361423), Amine Benjelloun Touimi (355929), Yahya Skalli (341989), Nada Ezziti (355731)
COM-304 Final Project Report

***Abstract*—We adapt the lightweight *Nano4M* transformer to generate short human-action videos on resource-limited capabilities. Our model conditions each frame on the previous one, jointly encoding RGB patches, 2-D poses, and captions from the Penn Action dataset. With only $\sim 2{,}000$ training frame pairs, it produces six temporally coherent future frames while running over four times faster than diffusion baselines. This shows that compact, multimodal tokenization can deliver realistic video synthesis without heavy compute.**

I. Introduction

The generation of video content has become a central challenge in the field of generative AI. Although recent advances have achieved impressive results using large-scale diffusion models, their computational cost remains high for resource-constrained environments. This limits their applicability in contexts such as mobile devices, embedded systems, or interactive media creation.

In this project, we address this limitation by extending Nano4M. Most existing models generate video sequences by processing multiple frames jointly, which requires large architectures and significant compute. In contrast, we adopt a lightweight, iterative generation strategy that predicts one frame at a time, conditioning only on the immediate past. This method ensures temporal coherence while significantly reducing memory and computation requirements.

We specifically target the task of human-centric video generation, focusing on sports actions and poses. By conditioning generation on rich semantic modalities such as RGB images, pose annotations, and textual captions, we aim to use Nano4M to generate temporally consistent and visually plausible frame sequences. To this end, we introduce architectural adaptations and training strategies that enable temporally aware multimodal decoding.

Throughout the project, we implemented and validated key extensions to the Nano4M model:

- Integrated pose and caption modalities into the training loop.
- Adapted the dataset loader for sequential RGB–pose–caption input.
- Used the Penn Action dataset for real-world, annotated human motion.
- Devised an iterative generation strategy capable of producing up to 6 future frames from a single input.

This work demonstrates the feasibility of adapting a lightweight foundation model for structured, multimodal video generation. Our results highlight how modest models—when properly conditioned and iteratively deployed—can achieve strong performance in generating temporally coherent video content.

II. Related Work

Masked modeling approaches such as MAE and BEiT have shown promise in learning visual representations without labels. In the multimodal space, models like Flamingo and OpenFlamingo integrate text and vision with impressive results. Nano4M extends this concept to multiple modalities, for example coordinates.

Prior work on human-pose generation has largely focused on keypoint prediction or image synthesis using GANs or VAEs. Unlike these, our method uses a unified tokenized representation, enabling cross-modal learning. We build on nano4M's architecture but tailor it for pose-video prediction by modifying data sampling, introducing custom masking strategies, and evaluating on sequence generation quality.

III. Methods

This section summarizes the M2 variant (our best performing model) of the fourM Transformer on the Penn Action dataset, covering (1) Overall Architecture, (2) Data & Preprocessing, (3) Tokenization, (4) Enhancements to Nano4M, (5) Training Configuration & Sampling, and (6) Evaluation.

A. 1. Overall Architecture

Our model follows the fourM (Massively Multimodal Masked Modeling) paradigm, extended to three modalities: RGB patches, 2D human poses, and frame captions. A high-level diagram is shown in Figure 1.

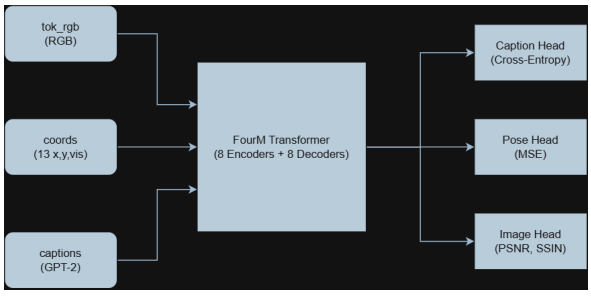

Figure 1. M2 three-modality fourM pipeline.

Input modalities:

- **tok_rgb**: COSMOS-tokenized 256×256 frame patches (truncated to 256 tokens/frame).
- **coords**: 13 joints per frame, each quantized into $(x, y, \text{vis}) \rightarrow 39$ tokens.
- **captions**: BLIP-generated sentence/tokenized via GPT-2 (truncated/padded to 1536 tokens).

The Transformer backbone comprises:

- **Encoder**: 8 layers of multi-head self-attention (hidden size $D = 512$, head dim $= 64$, MLP ratio = 4).
- **Decoder**: 8 layers of self-attention + encoder–decoder cross-attention (same dimensions).
- **Modality & Positional Embeddings**: Each token ID $\in [0, 65535]$ is embedded into R^{512}; add learned modality embeddings (one per modality) and fixed 1D sinusoidal positional embeddings.

Output heads:

- **Caption Head**: Cross-entropy over GPT-2 vocabulary ($\approx 50{,}260$ tokens).
- **Pose Head**: MSE on quantized pose tokens (8192 levels).
- **Image Head**: MSE (and PSNR at evaluation) on COSMOS tokens (65,536 levels).

B. 2. Data & Preprocessing

We use Penn Action (2 326 clips, 15 classes). Clips contain 18–663 frames (avg. ≈ 150). We split 80%/10%/10% (train/val/test), stratified by action.

Frame Extraction: decode videos to JPEGs, resize to 256×256 (bilinear), normalize pixels to $[0, 1]$.

Pose Quantization: 2D joint coordinates $(x, y) \in [0, 1]$ and visibility $\{0, 1\}$ stacked into a (3×13) array, flatten to length 39.

Caption Generation: BLIP model creates one English sentence per key frame. Each caption tokenized by GPT-2 (vocab $= 50{,}257 + 3$ special tokens), truncated/padded to 1536 tokens.

C. 3. Tokenization Schemes

All token IDs share a single learnable embedding matrix $(N_{vocab} = 65{,}536) \times 512$.

- **RGB**: COSMOS quantizes each 16×16 patch to $[0, 65535]$, stored as 40×40 grid, flattened to 1600 tokens, truncated to 256.
- **Pose**: Quantized (x, y, vis) each $\in [0, 8191]$ combine into 39 tokens per frame.
- **Text**: GPT-2 (50260 tokens total); each caption truncated/padded to 1536 tokens.

D. 4. Nano4M-Specific Enhancements

In `fourm.py` and `run_training.py`, we implemented:

1) **Decoder Masking & Stability:** ensure every query attends to at least one valid key mask (avoids silent NaNs).
2) **Zeroing Padded Embeddings:** Before each residual block in decoder, masked token embeddings are zeroed to prevent gradient leaks.
3) **Smoke Test on Startup:** A single-batch forward-backward check (with `torch.autograd.detect_anomaly`) to catch data misalignment, invalid token IDs, or shape mismatches immediately.
4) **Future-Frame Roll-Out:** Autoregressively generate up to 6-frame sequences (captions + skeleton + frame) from a single reference to verify temporal coherence.

E. 5. Training Configuration & Sampling

Parameters are defined in M2-config.yaml (there are also configs for less performant models like 1,3,4):

Frame-Sampling Pipelines:

1) *Full-Frame*: Use all frames (padding/trunc. to 150). $\rightarrow$ up to 1600 RGB tokens, 39 pose tokens, 1536 text tokens per sample.
2) *Stride-5 (Key-Frames)*: Keep frames {0,5,10,...}. $\rightarrow$ $\approx$30 frames $\rightarrow$ 256 RGB tokens, 39 pose tokens, 1536 text tokens.
3) *Ref + Next-6*: Choose reference $\lfloor n/2 \rfloor$ and next 6 frames (pad if needed). $\rightarrow$ 7 frames $\rightarrow$ 256 RGB tokens, 39 pose tokens, 1536 text tokens.
4) *(Ref,Next) Pairs*: All adjacent (frame$_i$, frame$_{i+1}$) pairs. $\rightarrow$ 2 frames $\rightarrow$ 256 RGB tokens, 39 pose tokens, 1536 text tokens.

All modalities aligned on the same frame indices; missing frames masked/padded.

F. 6. Evaluation Metrics

- **Caption BLEU-4**: NLTK smoothing 1 on generated vs. ground-truth tokens; averaged across test frames.
- **Pose MSE**: Convert $x_{\text{tok}} \rightarrow \hat{x} = x_{\text{tok}}/8191$; compute

$$\text{MSE} = \frac{1}{N_{\text{frames}} \times 13} \sum_{f=1}^{N_{\text{frames}}} \sum_{j=1}^{13} \big((x_j^{(f)} - \hat{x}_j^{(f)})^2 + (y_j^{(f)} - \hat{y}_j^{(f)})^2\big).$$

- **Frame PSNR & SSIM**: Reconstruct 256×256 image from COSMOS tokens; compute PSNR and SSIM (scikit-image default) vs. ground truth, averaged over test frames.
- **Future-Frame Qualitative**: Autoregressive 7-frame GIFs compared to ground-truth ("pose_future_predicted.gif" Vs "pose_future_original.gif").

See apendix for Losses.

IV. Experiments

To validate our approach and evaluate the performance of the extended nano4M model on human pose video generation, we designed a comprehensive set of experiments.

These experiments were structured around several data configurations and training strategies, allowing us to assess both qualitative and quantitative aspects of the model's behavior.

A. Setup

We used the Penn Action dataset, which contains over 2000 video sequences of human actions, annotated with pose coordinates. We processed the dataset into three input modalities: tokenized RGB frames, pose coordinates, and textual captions.

The model architecture was based on the FourM design, with encoder-decoder attention mechanisms and modality-specific masking strategies.

B. Baselines and Variants

We evaluated three primary data processing strategies:

1. Full-Frame Tokenization: Each video was tokenized using all available frames (150 per video). This setup served as our baseline. We observed solid performance on textual and pose reconstruction, but the RGB modality showed high reconstruction loss.

2. Stride-5 Frame Sampling: Frames were sampled at intervals of 5. This significantly reduced the sequence length and training time, while preserving temporal diversity. The model retained competitive performance, though RGB loss remained high, indicating persistent difficulty in modeling visual data.

3. Reference + Next-6 Frames: For each sample, we included a fixed reference frame followed by six consecutive frames. This configuration achieved the best results across modalities, suggesting that temporal locality aids generation. However, signs of overfitting emerged, particularly in the form of repeated captions.

C. Evaluation Metrics

We used cross-entropy loss averaged over each modality to track training performance. Additionally, qualitative inspection of generated sequences was used to assess generation quality and diversity.

D. Findings

Pose and caption modalities consistently exhibited low loss, even under reduced frame sampling.

RGB reconstruction was the most challenging, likely due to the high dimensionality and tokenization resolution.

Overfitting occurred more rapidly when using captions in small data regimes, motivating future work in data augmentation and regularization.

E. Ablation Studies

We conducted ablation studies by removing individual modalities and varying frame sampling parameters. Results confirmed that captions significantly influence the decoder and may bias generation when the dataset is small. Excluding captions improved visual diversity in generated outputs.

Overall, the experiments validate our hypothesis that carefully selected frame sequences and modality balancing are critical for high-quality video generation in low-resource settings.

V. Conclusion and Limitations

In this project, we built and tested lightweight models that generate human action videos one frame at a time using RGB images and pose data. By extending Nano4M and trying different training setups, we aimed to create realistic and consistent video sequences without the heavy cost of large models.

Our most successful model—M2—achieved the best results despite being trained on only 2,000 frame pairs. By carefully selecting visually distinct frames using SSIM thresholds and ensuring high-quality pose tokenization, M2 was able to generate up to 6 coherent future frames that preserved key scene elements and initiated realistic motion patterns. This contrasted sharply with M3, which used over 40,000 frame pairs but suffered from severe overfitting and poor generalization on evaluation data.

These results show the importance of meaningful data processing and model architecture over massive data volume. However, several limitations remain:

- Overfitting on limited data: Even if we achieved low training loss, the model generalized poorly due to the small and homogeneous training set. This shows the need for more diverse and representative datasets in future work.
- High loss on RGB token prediction: Despite architectural changes, the RGB modalities losses stayed high, reflecting the complexity of modeling rich visual content with compact representations.
- Caption modality impact on training: While captions helped for providing context in early stages, they were ultimately dropped in later versions due to redundancy and lack of diversity in the dataset. A richer textual description or fine-tuned captioning system could better exploit this modality.
- Single-frame iterative prediction: Apart from its efficiency, our step-by-step frame prediction pipeline may propagate small errors over time. Extending the architecture to model short sequences jointly (e.g., 2–4 frames) could improve temporal stability.

Future Extensions:

- Extend M2-style strategy with more data: Scaling the multi-frame + caption approach of M2 to a broader dataset is a key next step.
- Hybrid decoding schemes to improve speed: Combining auto-regressive and parallel decoding could reduce generation latency while preserving frame quality.
- Expanded dataset and pretraining: Using larger and more heterogeneous video datasets could improve gen-

eralization, especially when combined with weakly supervised or self-supervised pretraining strategies.

This work demonstrates the viability of adapting lightweight multimodal models to structured video generation and lays the foundation for future low-resource, human-aware generative systems.

VI. Individual contributions

This project was carried out as a fully collaborative effort. All team members participated equally in every phase—brainstorming ideas, researching and selecting the Penn Action dataset, designing and implementing the multimodal DataLoader, extending the fourM model, and interpreting results. We deliberately structured our workflow so that each person touched on all components (data preprocessing, tokenization, model enhancements, and evaluation) in order to maintain shared context and mutual understanding.

The only distinctly individual task was training our respective model instances: each member launched and managed a separate training run (varying certain hyperparameters or sampling strategies) to explore convergence behavior and performance trade-offs. Beyond these parallel training experiments, however, every other aspect—including writing code, debugging, and devising improvements—was done together, with ownership and responsibility shared equally among group members.

References

VII. Appendix

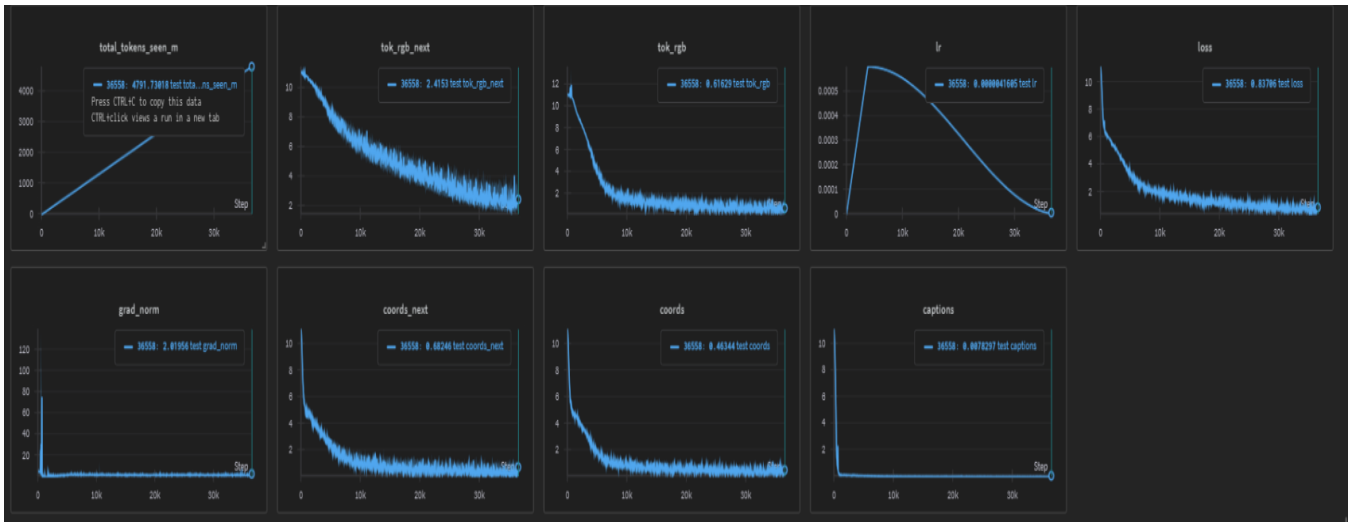

Figure 2. Training curves for M2 on Penn Action.
From left to right, top row: total tokens seen ($\times 10^6$), next-frame RGB loss (`tok_rgb_next`), current-frame RGB loss (`tok_rgb`), learning rate schedule, overall loss.
Bottom row: gradient norm, next-frame pose loss (`coords_next`), current-frame pose loss (`coords`), caption loss. Each plot is taken on the validation split and shows stable convergence toward the end of training.