CascadeFormer: A Family of Two-stage Cascading Transformers for Skeleton-based Human Action Recognition

Anonymous submission

Abstract

*Skeleton-based human action recognition leverages sequences of human joint coordinates to identify actions performed in videos. Owing to the intrinsic spatiotemporal structure of skeleton data, Graph Convolutional Networks (GCNs) have been the dominant architecture in this field. However, recent advances in transformer models and masked pretraining frameworks open new avenues for representation learning. In this work, we propose **CascadeFormer**, a family of two-stage cascading transformers for skeleton-based human action recognition. Our framework consists of a masked pretraining stage to learn generalizable skeleton representations, followed by a cascading fine-tuning stage tailored for discriminative action classification. We evaluate CascadeFormer across three benchmark datasets—Penn Action, N-UCLA, and NTU RGB+D 60—achieving competitive performance on all tasks. To promote reproducibility, we will release our code and model checkpoints.*

Introduction

Human action recognition is a fundamental computer vision task that aims to identify the action performed by a target person in a given video, represented as a sequence of frames. Human actions can be interpreted through various modalities, such as raw RGB images (Zhang, Gupta, and Zisserman 2021), skeletal joint coordinates (Zhao et al. 2019), or a fusion of both (Kim, Ahn, and Ko 2023). Compared to RGB-based approaches, skeleton-based action recognition offers distinct advantages (Zhou et al. 2023): it is computationally efficient due to relying solely on human joint coordinate data, and it is more robust to environmental noise and variations in camera viewpoints. Leveraging these strengths, skeleton-based action recognition has been explored through diverse methods, including probabilistic models (Zhao et al. 2019), recurrent neural networks (Shahroudy et al. 2016; Liu et al. 2016), and graph convolutional networks (GCNs) (Yan, Xiong, and Lin 2018).

Nevertheless, the rapid advancement of attention mechanisms (Vaswani et al. 2023) in natural language processing has driven the widespread adoption of transformer architectures in computer vision. The success of vision transformers (Dosovitskiy et al. 2021; Rao et al. 2021) across tasks such as image classification (Ridnik et al. 2021), multimodal learning (Radford et al. 2021), and visual instruction tuning (Liu et al. 2023) underscores the potential of transformers for skeleton-based action recognition. Recent approaches incorporate novel attention designs tailored to spatiotemporal data (Bertasius, Wang, and Torresani 2021; Zhou et al. 2023; Wang et al. 2023); however, these transformer-based models are predominantly trained in an **end-to-end** manner.

However, instead of relying on end-to-end training—which may risk overfitting on downstream tasks—both language modeling (Devlin et al. 2019) and multimodal learning (Srivastava and Sharma 2023, 2024) have widely embraced masked pretraining as a precursor to supervised adaptation. ***Motivated by the goal of equipping simple transformers with masked pretraining***, we propose **CascadeFormer**, a family of two-stage cascading transformers for skeleton-based human action recognition. CascadeFormer comprises a lightweight transformer for masked pretraining and an additional transformer for fine-tuning on action labels. In this paper, we begin with a review of prior work in masked pretraining with transformers and skeleton-based action recognition, followed by a brief overview of the input skeleton data format. We then present our key contributions:

1. We introduce **CascadeFormer**, a framework comprising three model variants that share a unified pipeline involving masked pretraining followed by cascading fine-tuning.
2. We conduct extensive evaluations of CascadeFormer on three widely used and diverse datasets: Penn Action, N-UCLA, and NTU RGB+D 60.
3. We conduct extensive ablation studies—summarized in the main paper and further expanded in the supplementary material—to analyze the impact of architectural choices and training configurations.

Finally, we conclude with a discussion of the implications of our work.

Related Work

In this section, we first provide a brief overview of the development of masked pretraining frameworks across various fields. We then review widely adopted approaches in skeleton-based human action recognition.

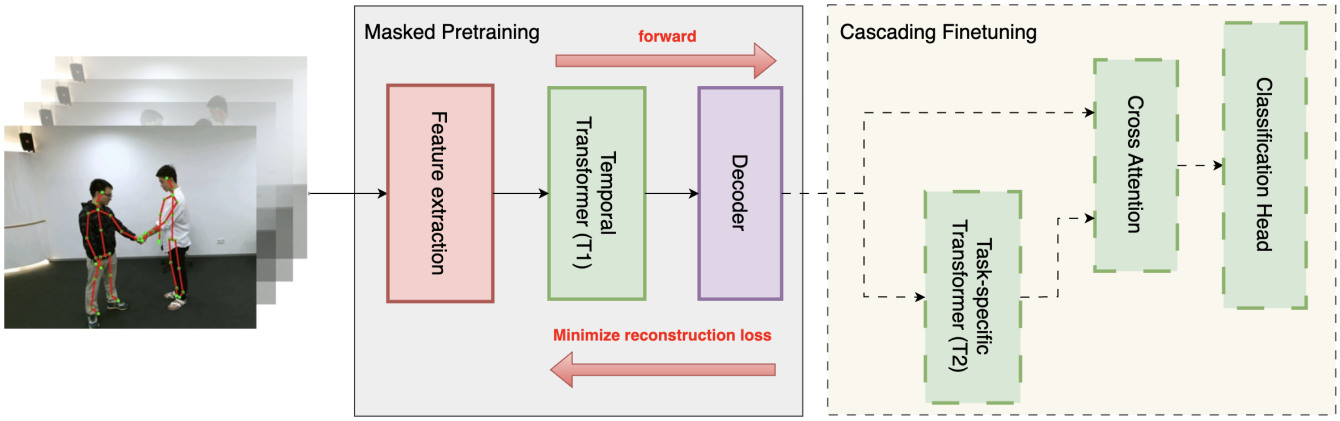

Figure 1: Overview of the masked pretraining component in CascadeFormer. A fixed percentage of joints are randomly masked across all frames in each video. The partially masked skeleton sequence is passed through a feature extraction module to produce frame-level embeddings, which are then input into a temporal transformer (T1). A lightweight linear decoder is applied to reconstruct the masked joints, and the model is optimized using mean squared error over the masked positions. This stage enables the model to learn generalizable spatiotemporal representations prior to supervised finetuning.

Masked Pretraining Framework Masked pretraining was first introduced in the domain of language modeling, most notably with BERT (Devlin et al. 2019). By randomly masking 15% of the input tokens and training the model to predict them in a self-supervised manner, BERT is able to learn deep bidirectional representations (Devlin et al. 2019). A similar paradigm has been extended to computer vision with masked autoencoders (MAEs) (He et al. 2021), which apply a significantly higher masking ratio (up to 80%), enabling effective self-supervised pretraining for vanilla vision transformers (Dosovitskiy et al. 2021). More recently, the masked pretraining framework has been rapidly adopted in multimodal learning. Models such as OmniVec (Srivastava and Sharma 2023) and OmniVec2 (Srivastava and Sharma 2024) leverage masked autoencoding techniques to learn unified representations across diverse modalities—including image, text, video, and audio.

Skeleton-based Action Recognition Human action recognition can be explored through various modalities, including raw RGB images (Zhang, Gupta, and Zisserman 2021), skeletal joint coordinates (Zhao et al. 2019), or a fusion of both (Kim, Ahn, and Ko 2023). In contrast to RGB-based methods, **skeleton**-based action recognition is characterized by its computational efficiency—owing to its reliance solely on human joint coordinate data—and its robustness to environmental noise and camera viewpoint variations, as it preserves only the spatiotemporal information of keypoints (Zhou et al. 2023). While probabilistic models (Zhao et al. 2019) demonstrated promising performance, deep learning approaches have become dominant in this domain. Given the temporal nature of video sequences, recurrent neural networks (RNNs) (Shahroudy et al. 2016; Liu et al. 2016) have proven effective in modeling temporal dynamics. Additionally, convolutional neural networks (CNNs) (Cai et al. 2023; Duan et al. 2022) have been adapted to this task by transforming skeleton data into pseudo-images, allowing convolutional operations to be applied. Furthermore, graph-based neural networks—including graph neural networks (GNNs) (Shi et al. 2019) and graph convolutional networks (GCNs) (Yan, Xiong, and Lin 2018)—have gained popularity due to the natural graph structure of human skeletons, where joints and bones are represented as vertices and edges, respectively. More recently, transformer architectures (Vaswani et al. 2023) have attracted increasing attention in skeleton-based action recognition. These models are often enhanced with efficient spatiotemporal attention mechanisms (Zhou et al. 2023; Do and Kim 2024; Bertasius, Wang, and Torresani 2021; Wang et al. 2023), and are typically trained in an end-to-end fashion.

Preliminaries

In this section, we formally define the typical format of input skeleton data used in skeleton-based action recognition.

Skeleton Data Skeletons, or pose maps, refer to sets of Cartesian coordinates representing the key joints of the human body. Formally, given a batch of B video sequences, each consisting of T frames, with J joints per frame in a C-dimensional space, the the input skeleton data X is typically structured as:

$$X \in \mathbb{R}^{B \times C \times T \times J}$$

While certain datasets such as Penn Action define skeletons in a two-dimensional space ($C = 2$), most recent datasets—including N-UCLA and NTU RGB+D 60—represent joint coordinates in three-dimensional space ($C = 3$) to more accurately capture 3D motion. Since video lengths may vary across samples, it is common to either sample a fixed number of frames or apply dynamic padding based on the longest sequence in the batch.

Multi-person Skeleton Data Some datasets, such as NTU RGB+D 60, include actions involving multiple individuals.

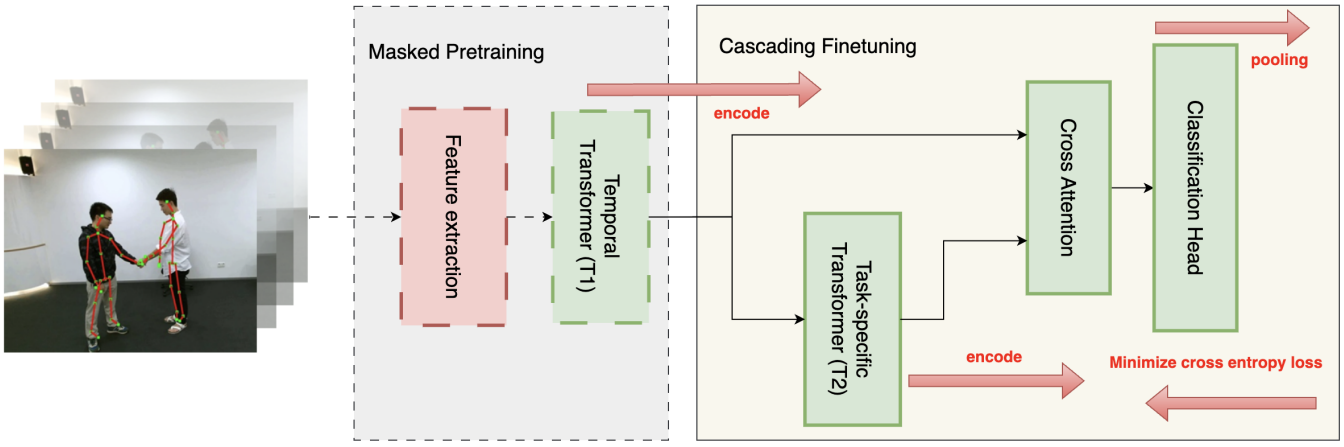

Figure 2: Overview of the cascading finetuning component in CascadeFormer. The frame embeddings produced by the pretrained temporal transformer backbone (T1) are passed into a task-specific transformer (T2) for hierarchical refinement. The output of T2 is fused with the original embeddings via a cross-attention module. The resulting fused representations are aggregated through frame-level average pooling and passed to a lightweight classification head. The entire model—including T1, T2, and the classification head—is optimized using cross-entropy loss on action labels during finetuning.

In these cases, assuming up to M persons appear in each video, the input skeleton data X can be extended to:

$$X \in \mathbb{R}^{B\times C\times T\times\ J\times\ M}$$

When multiple persons are present, many approaches (Do and Kim 2024; Zhou et al. 2023) encode all detected individuals. However, some methods (Shahroudy et al. 2016) opt to retain only the most active person in the scene to reduce computational complexity or avoid noise from irrelevant actors.

CascadeFormer

To equip vanilla transformers with masked pretraining—a technique widely adopted in language modeling and multimodal learning—we propose **CascadeFormer**, a family of two-stage cascading transformers for skeleton-based human action recognition. As illustrated in Figure 1, CascadeFormer employs a lightweight transformer for masked pretraining, which is then followed by an additional transformer for finetuning on action labels (Figure 2). We design a family of three model variants that differ in their feature extraction modules, as shown in Figure 3. The architecture is composed of three key components: masked pretraining, cascading fine-tuning, and feature extraction—each of which is described in detail in the following subsections.

Masked Pretraining Component

The masked pretraining component, illustrated in Figure 1, enables the model to learn generalizable spatiotemporal dependencies in a self-supervised manner before any supervised training on action labels. Inspired by masked modeling strategies in other domains, we apply a joint-level masking scheme: specifically, 30% of the joints across all frames are randomly selected and masked by setting their coordinates to zero, while the remaining 70% retain their original Cartesian values. The partially masked skeleton sequence is then passed through a feature extraction module to generate frame-level embeddings. Although the three CascadeFormer variants differ in their feature extraction designs, they all follow a common principle: each frame is treated as a token and projected into a high-dimensional embedding space. Formally, given a batch of B video sequences, each containing T frames with J joints per frame in a C-dimensional space, the resulting embedding tensor is denoted as:

$$E \in \mathbb{R}^{B\times T\times \text{embed\_dim}}$$

where embed_dim denotes the transformer embedding dimension. These frame embeddings are then input into a vanilla temporal transformer backbone (T1), as shown in Figure 1. The output of T1, denoted as E_{pretrain}, preserves the same shape:

$$E_{\text{pretrain}} \in \mathbb{R}^{B\times T\times \text{embed\_dim}}$$

To reconstruct the missing joint information, a lightweight linear decoder is applied to the output of T1. This decoder is used exclusively during pretraining and is discarded during downstream finetuning. The model is trained to minimize reconstruction error over only the masked joints. Let $masked\_X \in \mathbb{R}^{B\times C\times T\times J}$ represent the masked input skeletons and $masked\_X' \in \mathbb{R}^{B\times C\times T\times J}$ be the reconstructed output. The loss function is defined as the mean squared error (MSE) over the masked joints:

$$\mathcal{L}_{\text{MSE}} = \|masked\_X - masked\_X'\|^2$$

Cascading Finetuning Component

After masked pretraining, we introduce a ***cascading*** finetuning stage, as illustrated in Figure 2. This design is inspired by the hierarchical adaptation strategy proposed in OmniVec2 (Srivastava and Sharma 2024) for multimodal learning. Unlike conventional finetuning—where a lightweight

classification head is appended directly to a pretrained transformer—CascadeFormer incorporates an additional task-specific transformer (T2) to refine features in a hierarchical manner.

Specifically, the frame embeddings $E \in \mathbb{R}^{B\times T\times \text{embed\_dim}}$ obtained from the pretrained backbone (T1) are passed into the task-specific transformer (T2), which maps them into a new embedding space of the same dimensionality:

$$E_{\text{finetune}} \in \mathbb{R}^{B\times T\times \text{embed\_dim}}$$

These refined embeddings E_{finetune} are then fused with the original pretrained embeddings E_{pretrain} via a cross-attention module, as shown in Figure 2. Following the standard attention mechanism (Vaswani et al. 2023), cross-attention is computed as:

$$\text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d}}\right)V$$

$$E_{\text{cross}} = \text{Attention}(E_{\text{pretrain}}, E_{\text{finetune}}, E_{\text{finetune}})$$

To obtain a fixed-size video-level representation, we apply frame-level average pooling over the cross-attended embeddings:

$$E_{\text{avg}} = \frac{1}{T}\sum_{t=1}^{T} E_{\text{cross}}[:,t,:] \in \mathbb{R}^{B\times \text{embed\_dim}}$$

Finally, a lightweight classification head is applied to the pooled embeddings, and the model is optimized using the standard cross-entropy loss with respect to the ground-truth action labels. All parameters of the pretrained transformer backbone (T1), as well as the newly introduced T2 module and classification head, are trainable during the cascading finetuning stage.

Feature Extraction Module

As illustrated in Figure 3, input skeleton sequences are first passed through a feature extraction module to obtain preliminary frame-level embeddings E, which are then processed by the transformer backbone (T1). We design a family of three CascadeFormer variants, each differing in how frame embeddings are extracted:

CascadeFormer 1.0 This baseline variant employs a simple linear projection to generate frame embeddings directly from the input skeleton. Formally, given the input skeleton tensor $X \in \mathbb{R}^{B\times C\times T\times J}$, where B is the batch size, C the coordinate dimension, T the number of frames, and J the number of joints, the linear projection outputs:

$$E \in \mathbb{R}^{B\times T\times \text{embed\_dim}}$$

CascadeFormer 1.1 To better capture spatial locality among joints, this variant prepends a lightweight convolutional module before the linear projection. The input skeletons $X \in \mathbb{R}^{B\times C\times T\times J}$ are reshaped across the batch and temporal dimensions into $X \in \mathbb{R}^{(B\cdot T)\times C\times J}$, and then passed through a 1D convolutional layer:

$$X_{\text{convoluted}} \in \mathbb{R}^{(B\cdot T)\times C\times J}$$

The resulting activations are passed into the same linear projection as in CascadeFormer 1.0 to yield frame-level embeddings: $E \in \mathbb{R}^{B\times T\times \text{embed\_dim}}$.

CascadeFormer 1.2 In this variant, we first construct embeddings for each individual joint using a linear layer, producing:

$$E_{\text{joint}} \in \mathbb{R}^{(B\cdot T)\times J\times (\text{embed\_dim}/J)}$$

A single-layer spatial transformer (denoted as ST) is then applied to the joint embeddings:

$$E_{\text{joint\_ST}} \in \mathbb{R}^{(B\cdot T)\times J\times (\text{embed\_dim}/J)}$$

Finally, we aggregate the joint-level outputs into full frame-level embeddings $E \in \mathbb{R}^{B\times T\times \text{embed\_dim}}$. Note that for CascadeFormer 1.2, the total embedding dimension embed_dim must be divisible by the number of joints J to allow even allocation across joints. All three variants differ only in their feature extraction mechanisms; the remainder of the architecture—including masked pretraining (Figure 1) and cascading finetuning (Figure 2)—remains shared across all models.

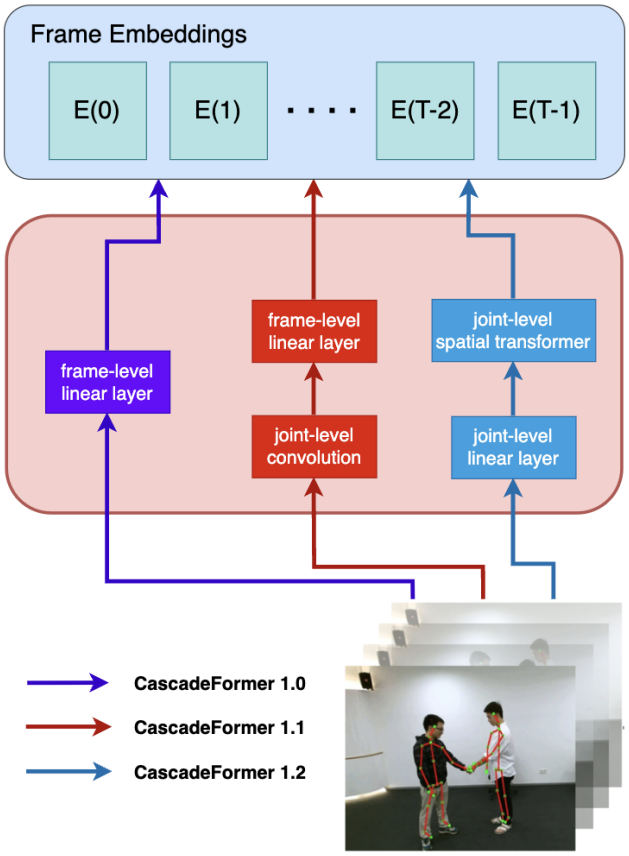

Figure 3: Feature extraction module in CascadeFormer. All variants convert input skeleton sequences into frame-level embeddings for downstream temporal modeling. CascadeFormer 1.0 (purple) applies a simple frame-level linear projection. CascadeFormer 1.1 (red) enhances this by first applying a joint-level 1D convolution to capture spatial locality before linear projection. CascadeFormer 1.2 (blue) constructs joint-level embeddings via a linear layer, refines them using a joint-level spatial transformer, and aggregates the outputs into frame-level embeddings.

Model Variant	**Penn Action** accuracy	**N-UCLA** accuracy	**NTU60/CS** accuracy	**NTU60/CV** accuracy
CascadeFormer 1.0	**94.66%**	89.66%	**81.01%**	**88.17%**
CascadeFormer 1.1	94.10%	**91.16%**	79.62%	86.86%
CascadeFormer 1.2	94.10%	90.73%	80.48%	87.24%

Table 1: **Overall accuracy evaluation results of CascadeFormer variants on three datasets.** CascadeFormer 1.0 consistently achieves the highest accuracy on Penn Action and both NTU60 splits, while 1.1 excels on N-UCLA. All checkpoints are open-sourced for reproducibility.

Experiment Setup

In this section, we introduce the datasets used for evaluation, followed by dataset-specific preprocessing steps and training configurations.

Datasets

We conduct a comprehensive evaluation of CascadeFormer on three widely-used benchmark datasets for human action recognition:

Penn Action (Zhang, Zhu, and Derpanis 2013) This dataset contains 2,326 video clips spanning 15 human action classes. Each frame is annotated with 13 human joints in a 2D space. The dataset provides a standard 50/50 train/test split, ensuring mutually exclusive samples in both sets.

N-UCLA (wang et al. 2014) N-UCLA comprises 1,494 video clips categorized into 12 action classes. Skeletons are annotated with 20 joints in 3D space. The dataset is collected from three camera viewpoints: two are used for training, and the third is reserved for evaluation, following the standard cross-view evaluation protocol.

NTU-RGB+D 60 (Shahroudy et al. 2016) This large-scale dataset includes 56,880 video samples covering 60 action classes, with each skeleton containing 25 joints in 3D space. NTU-RGB+D 60 offers two evaluation protocols: cross-subject (where subjects are split between training and testing) and cross-view (where specific camera angles are held out for testing). We adopt both protocols in our experiments.

Data Preprocessing

Due to differences in scale, format, and split protocols across datasets, we apply dataset-specific preprocessing strategies:

Penn Action We leverage the visibility flags provided with the annotations and remove occluded skeletons. Given the relatively small dataset size, we pad each video sequence within a batch to match the longest sequence in that batch.

N-UCLA Following the data preprocessing strategy in SkateFormer (Do and Kim 2024), we virtually repeat the dataset multiple times to increase training diversity. Each skeleton is normalized by centering it at the hip joint and applying scale normalization. Data augmentation techniques such as random rotation, scaling, and joint/axis dropping are applied. To avoid computational overhead from batch-level padding, we randomly sample a fixed length of 64 frames per sequence.

NTU-RGB+D 60 We adopt the preprocessing pipeline from CTR-GCN (Chen et al. 2021), which includes random rotation augmentation. To handle variable-length sequences, we uniformly sample 64 frames around the temporal center of each clip.

Training Setup

All models are implemented in PyTorch (Paszke et al. 2019) and trained on a single NVIDIA GeForce RTX 3090 GPU. We adopt dataset-specific hyperparameters to ensure optimal performance across different datasets.

Penn Action and N-UCLA We use AdamW (Loshchilov and Hutter 2019) as the optimizer. For masked pretraining, the learning rate is fixed at 1.0×10^{-4}. During the cascading finetuning stage, we reduce the learning rate to 1.0×10^{-5} and apply cosine annealing scheduling (Loshchilov and Hutter 2017).

NTU-RGB+D 60 Masked pretraining is performed with a constant learning rate of 1.0×10^{-4} and cosine annealing. During cascading finetuning, we switch to SGD (Ruder 2017) as the optimizer. For cross-subject evaluation, we use a base learning rate of 3.0×10^{-5}, while for cross-view evaluation, we set it to 1.0×10^{-4}. The number of epochs for both pretraining and finetuning varies across the three CascadeFormer variants.

Results

We present the performance of the CascadeFormer variants across three benchmark datasets in Table 1. Overall, each variant demonstrates competitive performance, with certain variants better suited to specific datasets.

Dataset-Wise Evaluation

On the Penn Action dataset, **CascadeFormer 1.0** achieves the highest accuracy at 94.66%, while both **1.1** and **1.2** closely follow at 94.10%. On the **N-UCLA** dataset, **CascadeFormer 1.1** leads with an accuracy of 91.16%, while variants **1.2** and **1.0** reach 90.73% and 89.66%, respectively. For the large-scale NTU RGB+D 60 dataset , which includes two evaluation protocols—cross-subject (CS) and cross-view (CV)—**CascadeFormer 1.0** again outperforms the others with **81.01%** (CS) and **88.17%** (CV). Variant 1.2 closely follows, while 1.1 lags slightly behind on both splits. These results suggest that, perhaps surprisingly, the simplest feature extraction design (CascadeFormer 1.0 with a linear frame encoder) can be as effective—if not more—than more

Action Type	NTU60/CS accuracy	NTU60/CV accuracy
Single-person	80.82%	**88.92%**
Two-persons	**81.81%**	84.86%
Overall	81.01%	88.17%

Table 2: **Performance of CascadeFormer 1.0 on multi-person actions in NTU RGB+D 60.** Accuracy is broken down into single-person actions and two-person interactions under both cross-subject and cross-view splits.

complex designs involving convolutions or spatial transformers. All model checkpoints used in Table 1 are publicly released via HuggingFace for further analysis and reproducibility.

Multi-Person Action Analysis

Unlike Penn Action and N-UCLA, which feature only single-person actions, NTU RGB+D 60 includes 11 classes involving interactions between two individuals (e.g., hugging, handshaking, pushing). Following the NTU authors' recommendation (Shahroudy et al. 2016), we retain the person with greater motion variance for all actions. As shown in Table 2, **CascadeFormer 1.0** achieves 81.81% accuracy on two-person actions (CS split), slightly higher than the 80.82% obtained on single-person actions. This suggests strong generalization across subjects for interaction-based actions, despite a smaller number of classes. However, on the CV split, the model performs better on single-person actions (88.92%) than on two-person actions (84.86%), indicating that generalizing interactions across varying camera viewpoints remains more challenging than across different individuals. These findings reveal that cross-view generalization of social interactions may benefit from additional structural modeling of inter-person dynamics, which could be explored in future work.

Performance Comparison

In this section, we compare the performance of all three CascadeFormer variants across different datasets with other representative models in skeleton-based action recognition.

Model	**Penn Action** accuracy
AOG	85.5%
HDM-BG	93.4%
CascadeFormer 1.0	**94.66%**
CascadeFormer 1.1	94.10%
CascadeFormer 1.2	94.10%

Table 3: **Comparison on Penn Action (Zhang, Zhu, and Derpanis 2013).** AOG (Nie, Xiong, and Zhu 2015) is a hierarchical graph-based model. HDM-BG (Zhao et al. 2019) is a probabilistic Bayesian approach. CascadeFormer variants achieve top accuracy, with 1.0 reaching the highest.

Penn Action Table 3 compares model accuracy on the Penn Action dataset (Zhang, Zhu, and Derpanis 2013). AOG (Nie, Xiong, and Zhu 2015), a hierarchical graph model, reaches 85.5% accuracy. HDM-BG (Zhao et al. 2019), a probabilistic model within a Bayesian framework, achieves 93.4%. All three CascadeFormer variants exceed 94% accuracy, with **CascadeFormer 1.0** achieving the best performance at **94.66%**. These results demonstrate the effectiveness of CascadeFormer on compact 2D skeleton datasets like Penn Action.

N-UCLA Table 4 shows the results on the N-UCLA dataset (wang et al. 2014). ESV (Liu, Liu, and Chen 2017), a CNN-based visualization method, obtains 86.09%, while Ensemble TS-LSTM (Lee et al. 2017), an RNN-based model, reaches 89.22%. CascadeFormer 1.0 already surpasses this with 89.66%, and 1.1 and 1.2 further improve to **91.16%** and 90.73%, respectively. These results confirm that CascadeFormer performs robustly on small-scale 3D skeleton datasets as well.

Model	**N-UCLA** accuracy
ESV	86.09%
Ensemble TS-LSTM	89.22%
CascadeFormer 1.0	89.66%
CascadeFormer 1.1	**91.16%**
CascadeFormer 1.2	90.73%

Table 4: **Comparison on N-UCLA (wang et al. 2014).** ESV (Liu, Liu, and Chen 2017) is a CNN-based visualization method. TS-LSTM (Lee et al. 2017) is a recurrent model ensemble. CascadeFormer 1.1 achieves the highest accuracy.

Model	**NTU60/CS** accuracy
ST-LSTM	69.2%
ST-GCN	**81.5%**
CascadeFormer 1.0	**81.01%**
CascadeFormer 1.1	79.62%
CascadeFormer 1.2	80.48%

Table 5: **Comparison on NTU RGB+D 60 (Cross-Subject)(Shahroudy et al. 2016).** ST-LSTM(Liu et al. 2016) uses recurrent modeling. ST-GCN (Yan, Xiong, and Lin 2018) adopts graph convolutions. CascadeFormer 1.0 achieves performance comparable to ST-GCN without graph structures.

Model	**NTU60/CV** accuracy
ST-LSTM	77.7%
ST-GCN	**88.3%**
CascadeFormer 1.0	**88.17%**
CascadeFormer 1.1	86.86%
CascadeFormer 1.2	87.24%

Table 6: **Comparison on NTU RGB+D 60 (Cross-View) (Shahroudy et al. 2016).** CascadeFormer 1.0 nearly matches the graph-based ST-GCN despite not explicitly modeling spatial graphs.

NTU RGB+D 60 On the NTU RGB+D 60 dataset, we evaluate on both cross-subject (CS) and cross-view (CV) splits. As shown in Tables 5 and 6, ST-LSTM (Liu et al. 2016) achieves 69.2% (CS) and 77.7% (CV), while ST-GCN (Yan, Xiong, and Lin 2018), a graph-based method, improves performance to 81.5% and 88.3%, respectively. CascadeFormer 1.0 delivers comparable performance—**81.01%** (CS) and **88.17%** (CV)—despite not using any explicit graph structure. CascadeFormer 1.2 follows closely behind, achieving 80.48% and 87.24%. These results validate CascadeFormer scalability and robustness on large-scale, complex 3D datasets. Even without spatial graphs, our model competes with state-of-the-art graph convolutional approaches.

Insights The empirical results across all three datasets provide a couple insights into the design and performance of CascadeFormer: First, the effectiveness of the cascading finetuning strategy is consistent across model variants and datasets. CascadeFormer variants demonstrate clear improvements over strong baselines in both 2D and 3D skeleton-based action recognition tasks. Second, CascadeFormer demonstrates competitive scalability to large-scale and complex datasets. On NTU RGB+D 60, despite the absence of an explicit graph structure, CascadeFormer achieves competitive accuracy performance. This suggests that transformer-based architectures, when paired with effective pretraining strategies, can be comparable to graph-based methods in modeling human motion dynamics.

Ablation Highlights

This section presents two key ablation studies that highlight the effectiveness of our proposed design choices. Additional ablation results and comprehensive analyses can be found in the supplementary material.

Necessity of Strong Pretraining

# Epochs of Pretraining	**NTU60/CS** accuracy
1 epoch	76.38%
100 epochs	**81.01%**
200 epochs	81.09%

Table 7: **Effect of pretraining duration on NTU RGB+D 60 (cross-subject) performance.** We report accuracy on the cross-subject split after pretraining **CascadeFormer 1.0** for 1, 100, and 200 epochs. Performance improves significantly with longer pretraining, but plateaus after 100 epochs, indicating diminishing returns beyond this point.

In this ablation study, we examine the necessity of employing a strong transformer backbone (T1) through masked pretraining. Specifically, we conduct a case study on **CascadeFormer 1.0** using the cross-subject split of the NTU RGB+D 60 dataset, a large-scale and complex benchmark well-suited for revealing performance differences under varying pretraining strengths. We pretrain the T1 backbone for 1, 100, and 200 epochs, followed by 100 epochs of finetuning in all settings to ensure fair comparison. A weak backbone pretrained for only 1 epoch yields an accuracy of 76.38%, while extending pretraining to 100 epochs results in a substantial improvement to 81.01%. Although training for 200 epochs yields a marginal further increase to 81.09%, the additional computational cost outweighs the negligible performance gain. Therefore, we adopt 100 epochs of masked pretraining for all subsequent experiments. This study confirms that a strong backbone—achieved via sufficient masked pretraining—is both necessary and beneficial for robust performance on challenging action recognition tasks.

Pretraining Strategy

Pretraining Strategy	**Penn Action** accuracy
random joint masking	**94.66%**
random frame masking	89.98%
regular reconstruction	91.10%

Table 8: **Effect of pretraining strategy on Penn Action performance.** We compare three pretraining objectives for **CascadeFormer 1.0**: random joint masking, random frame masking, and regular (unmasked) reconstruction. Random joint masking yields the best downstream performance, highlighting the benefit of fine-grained spatial masking.

In this ablation study, we further explore two alternative pretraining strategies using **CascadeFormer 1.0** on the Penn Action dataset (Zhang, Zhu, and Derpanis 2013), with results summarized in Table 8. Our proposed strategy applies random masking to 30% of joints across the entire video sequence, and optimizes a reconstruction loss. This method achieves a high accuracy of 94.66% on Penn Action. As a comparison, we evaluate a frame-level masking strategy, where 30% of entire frames are randomly masked. This results in a lower accuracy of 89.98%, indicating that reconstructing full frames is significantly more challenging for the transformer backbone than reconstructing scattered joint positions. We also investigate a **non-masked** variant in which the full skeleton sequence is reconstructed without any masking. This strategy achieves 91.10% accuracy, suggesting that while full reconstruction can be effective, it may lead to overfitting on smaller datasets such as Penn Action. Based on these observations, we adopt the random joint masking strategy as our default pretraining approach for all subsequent experiments.

Conclusion

We introduced CascadeFormer, a transformer-based model for skeleton-based action recognition with masked pretraining and cascading finetuning. Evaluated on Penn Action, N-UCLA, and NTU RGB+D 60, CascadeFormer consistently achieved strong performance across spatial modalities and dataset scales. Our results demonstrate the effectiveness of unified transformer pipelines for generalizable action recognition. Future directions include scaling to longer sequences, incorporating graph-aware modules, and extending to multimodal settings with RGB or depth inputs.

References

Bertasius, G.; Wang, H.; and Torresani, L. 2021. Is Space-Time Attention All You Need for Video Understanding? arXiv:2102.05095.

Cai, D.; Kang, Y.; Yao, A.; and Chen, Y. 2023. Ske2Grid: Skeleton-to-Grid Representation Learning for Action Recognition. arXiv:2308.07571.

Chen, Y.; Zhang, Z.; Yuan, C.; Li, B.; Deng, Y.; and Hu, W. 2021. Channel-wise Topology Refinement Graph Convolution for Skeleton-Based Action Recognition. arXiv:2107.12213.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805.

Do, J.; and Kim, M. 2024. SkateFormer: Skeletal-Temporal Transformer for Human Action Recognition. arXiv:2403.09508.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; Uszkoreit, J.; and Houlsby, N. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. arXiv:2010.11929.

Duan, H.; Zhao, Y.; Chen, K.; Lin, D.; and Dai, B. 2022. Revisiting Skeleton-based Action Recognition. arXiv:2104.13586.

He, K.; Chen, X.; Xie, S.; Li, Y.; Dollár, P.; and Girshick, R. 2021. Masked Autoencoders Are Scalable Vision Learners. arXiv:2111.06377.

Kim, S.; Ahn, D.; and Ko, B. C. 2023. Cross-Modal Learning with 3D Deformable Attention for Action Recognition. arXiv:2212.05638.

Lee, I.; Kim, D.; Kang, S.; and Lee, S. 2017. Ensemble Deep Learning for Skeleton-Based Action Recognition Using Temporal Sliding LSTM Networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*, 1012–1020.

Liu, H.; Li, C.; Wu, Q.; and Lee, Y. J. 2023. Visual Instruction Tuning. arXiv:2304.08485.

Liu, J.; Shahroudy, A.; Xu, D.; and Wang, G. 2016. Spatio-Temporal LSTM with Trust Gates for 3D Human Action Recognition. arXiv:1607.07043.

Liu, M.; Liu, H.; and Chen, C. 2017. Enhanced Skeleton Visualization for View Invariant Human Action Recognition. *Pattern Recognition*, 68: 346–362.

Loshchilov, I.; and Hutter, F. 2017. SGDR: Stochastic Gradient Descent with Warm Restarts. arXiv:1608.03983.

Loshchilov, I.; and Hutter, F. 2019. Decoupled Weight Decay Regularization. arXiv:1711.05101.

Nie, B. X.; Xiong, C.; and Zhu, S.-C. 2015. Joint action recognition and pose estimation from video. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1293–1301.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Kopf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; and Chintala, S. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems (NeurIPS)*, 8024–8035.

Radford, A.; Kim, J. W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; Krueger, G.; and Sutskever, I. 2021. Learning Transferable Visual Models From Natural Language Supervision. arXiv:2103.00020.

Rao, Y.; Zhao, W.; Liu, B.; Lu, J.; Zhou, J.; and Hsieh, C.-J. 2021. DynamicViT: Efficient Vision Transformers with Dynamic Token Sparsification. arXiv:2106.02034.

Ridnik, T.; Ben-Baruch, E.; Noy, A.; and Zelnik-Manor, L. 2021. ImageNet-21K Pretraining for the Masses. arXiv:2104.10972.

Ruder, S. 2017. An overview of gradient descent optimization algorithms. arXiv:1609.04747.

Shahroudy, A.; Liu, J.; Ng, T.-T.; and Wang, G. 2016. NTU RGB+D: A Large Scale Dataset for 3D Human Activity Analysis. arXiv:1604.02808.

Shi, L.; Zhang, Y.; Cheng, J.; and Lu, H. 2019. Skeleton-Based Action Recognition with Directed Graph Neural Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 7912–7921. Long Beach, CA.

Srivastava, S.; and Sharma, G. 2023. OmniVec: Learning robust representations with cross modal sharing. arXiv:2311.05709.

Srivastava, S.; and Sharma, G. 2024. OmniVec2 - A Novel Transformer Based Network for Large Scale Multimodal and Multitask Learning. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 27402–27414.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2023. Attention Is All You Need. arXiv:1706.03762.

wang, J.; Nie, X.; Xia, Y.; Wu, Y.; and Zhu, S.-C. 2014. Cross-view Action Modeling, Learning and Recognition. arXiv:1405.2941.

Wang, Q.; Shi, S.; He, J.; Peng, J.; Liu, T.; and Weng, R. 2023. IIP-Transformer: Intra-Inter-Part Transformer for Skeleton-Based Action Recognition. In *2023 IEEE International Conference on Big Data (BigData)*, 936–945. IEEE.

Yan, S.; Xiong, Y.; and Lin, D. 2018. Spatial Temporal Graph Convolutional Networks for Skeleton-Based Action Recognition. arXiv:1801.07455.

Zhang, C.; Gupta, A.; and Zisserman, A. 2021. Temporal Query Networks for Fine-grained Video Understanding. arXiv:2104.09496.

Zhang, W.; Zhu, M.; and Derpanis, K. G. 2013. From Actemes to Action: A Strongly-Supervised Representation for Detailed Action Understanding. In *2013 IEEE International Conference on Computer Vision*, 2248–2255.

Zhao, R.; Xu, W.; Su, H.; and Ji, Q. 2019. Bayesian Hierarchical Dynamic Model for Human Action Recognition. In

2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 7725–7734.

Zhou, Y.; Cheng, Z.-Q.; Li, C.; Fang, Y.; Geng, Y.; Xie, X.; and Keuper, M. 2023. Hypergraph Transformer for Skeleton-based Action Recognition. arXiv:2211.09590.