Skeleton-Based Real-Time Gesture Recognition: A Review of Methods, Datasets, and Deployment Challenges

Anurag Sharma[1*] and Shyamol Banerjee[2]

[1*]Department of Computer Science and Engineering,
Shriram College of Engineering and Management, Banmore, Morena, Madhya Pradesh, India.

[2*]Assistant Professor, Department of Computer Science and Engineering,
Shriram College of Engineering and Management, Banmore, Morena, Madhya Pradesh, India.
*Corresponding author(s). E-mail(s): anuragparashar111@gmail.com; Contributing authors: shyamolgwl@gmail.com;

Abstract

Recognising gestures from human body motion has moved, over roughly a decade, from instrumented gloves and hand-engineered video descriptors to learned mod-els that operate on skeletal keypoints produced by real-time pose estimators. Several surveys already catalogue this progress by architecture or by benchmark. This review organises it around a different question: which methods remain usable when a commodity camera is the only sensor and a CPU or edge device is the only compute? We trace the pipeline from pose-estimation front-ends through recurrent, convolutional, graph-convolutional, and attention-based recognition models, and extend the account to self-supervised representation learning and pose-based sign language recognition. Published results on NTU RGB+D 60 and 120, Kinetics-Skeleton, FineGYM, SHREC'17, WLASL, and video gesture benchmarks are consolidated from the original papers into comparison tables covering work from 2016 to 2025. Two observations recur across these tables. First, accuracy on the most-used benchmark has compressed into a narrow band near 93–94% cross-subject, and the leading figures are usually four- to six-stream ensembles whose inference cost is a multiple of the single-model cost. Second, results depend strongly on the pose estimator that produced the skeletons, so gains attributed to recognition models are partly gains in input quality. We close with design considerations for real-time systems and seven open problems:

robustness to keypoint noise, mixed hand-and-body vocabularies, population and viewpoint generalisation, consistent runtime reporting, continuous-stream spotting, data scarcity for new vocabularies, and privacy.

Keywords: Gesture recognition, Skeleton-based action recognition, Pose estimation, Graph convolutional networks, Self-supervised learning, Sign language recognition, Real-time systems

1 Introduction

Gesture is a channel of communication that predates every input device a computer has ever had. Building systems that read it reliably has been a goal of human–computer interaction (HCI) research for decades, and the motivations have multiplied: touch-free control of medical imaging during surgery, home-based rehabilitation with automatic form feedback, sign language interpretation, accessible interfaces for users with limited mobility, in-vehicle controls that leave the driver's eyes on the road, and interaction in virtual and augmented reality where no keyboard exists.

The technical landscape has changed sharply since the early surveys. Mitra and Acharya [1] catalogued the pre-deep-learning toolkit of hidden Markov models, particle filters and finite-state machines. Rautaray and Agrawal [2] reviewed vision-based hand gesture recognition for HCI at a time when systems still depended on segmentation and hand-crafted features. What followed was a change in kind rather than degree: deep models trained end-to-end, large skeleton benchmarks, graph neural networks defined over the human body, and pose estimators accurate and fast enough to run on a phone.

This review concerns the *skeleton-based* pipeline—camera, pose estimator, keypoint sequence, recognition model—and evaluates it through the lens of deployment. Our contributions are:

- a positioning of this review against recent surveys of skeleton-based action recogni-tion and hand gesture recognition (Section 2);
- a structured account of pose-estimation front-ends and the failure modes they pass downstream (Section 4);
- a taxonomy of supervised recognition models, extended with self-supervised repre-sentation learning and pose-based sign language recognition (Sections 5–7);
- comparison tables compiled from figures reported in the original publications, span-ning 2016 to 2025, together with two derived analyses: accuracy over time and accuracy against effective inference size (Section 9);
- an analysis of efficiency evidence, design considerations for real-time systems, and seven open problems (Sections 11–13).

Scope and selection. This is a narrative review. Works were identified in IEEE Xplore, the CVF open-access archive, the ACM Digital Library, SpringerLink, and arXiv using combinations of the terms *gesture recognition*, *skeleton-based action recog-nition*, *hand gesture*, *sign language recognition*, *pose estimation*, *self-supervised*, and

Table 1 Positioning of this review relative to existing surveys.

Survey	Primary subject	Organising axis	Deployment focus
Mitra & Acharya, 2007 [1]	Gestures (hand, arm, head, body)	Classical models (HMMs, filters, FSMs)	No
Rautaray & Agrawal, 2015 [2]	Vision-based hand gestures for HCI	Pipeline stages, pre-deep-learning	No
Sun et al., 2023 [5]	Human action, many modalities	Data modality	No
Ren et al., 2024 [3]	3D skeleton action	RNN / CNN / GCN / Transformer	No
Meng et al., 2025 [4]	Skeleton action	Four model families; mul-timodal role	No
Foteinos et al., 2025 [6]	Visual hand gestures	Static / isolated / contin-uous tasks	No
This review	Skeleton-based ges-tures (body, hands, sign)	Pipeline stage × deploy-ment cost	**Yes**

real-time. Inclusion favoured (i) methods that introduced a design idea later adopted by others, (ii) methods reporting results on the shared benchmarks of Section 8, and
(iii) works reporting runtime or model-size evidence. Accuracy values are reproduced as published; where a method reports several configurations, the configuration pre-sented by its authors as the main result is used, and the number of fused input streams is stated where the original makes it explicit.

2 Relation to Existing Surveys

Table 1 positions this review among existing surveys. Ren et al. [3] provide a broad account of 3D skeleton-based action recognition organised by four deep architectures—recurrent, convolutional, graph-convolutional, and Transformer-based—and present methods in a data-driven manner. Meng et al. [4] similarly organise skeletal action recognition into these four families and add a discussion of the skeleton's role in mul-timodal fusion. Sun et al. [5] review human action recognition across many modalities, of which skeletons are one. On the gesture side, Foteinos et al. [6] review deep visual hand gesture recognition using a systematic methodology, grouping methods by static, isolated dynamic, and continuous recognition tasks.
These surveys are complementary to ours. The skeleton-action surveys centre on recognition architectures evaluated on body-action benchmarks, and the hand-gesture survey centres on visual hand gestures across input

types. Neither makes deployment cost the primary axis of comparison. The present review differs in three respects: it treats the pose estimator as an integral, error-introducing component rather than as given; it brings body-action, hand-gesture and sign-language results into one account because practical HCI vocabularies mix all three; and it analyses reported accuracy jointly with ensemble size and model footprint, which determine whether a method can run interactively.

3 Background

3.1 What counts as a gesture

The literature uses *gesture* and *action* with overlapping meanings. Here a gesture is a deliberate movement intended to communicate or command; an action is any purposeful movement. Waving at a camera is both; drinking water is an action but not, in an HCI sense, a gesture. Many skeleton methods are developed on action datasets and then applied to gestures, so both literatures are reviewed.

Three further distinctions shape method design. *Static* gestures are defined by a held configuration and can in principle be read from one frame; *dynamic* gestures are defined by a trajectory and require temporal modelling. *Isolated* recognition assumes each clip contains one gesture; *continuous* recognition must also find where gestures begin and end. Finally, gestures differ in their *articulators*: some are carried by the fingers alone, some by the arm and torso, and some by both hands in coordination with the body.

3.2 Sensing options

Wearable sensing—data gloves, magnetic trackers, inertial measurement units—gives clean, high-rate motion signals that are indifferent to lighting, at the price of requiring the user to put equipment on. Depth cameras such as the Kinect family supplied both depth maps and built-in skeleton tracking, and much of the skeleton literature was built on that output; NTU RGB+D [7, 8] was recorded with Kinect v2 and distributes its 25-joint skeletons. Plain RGB cameras are cheapest and most widespread but carry no geometry by themselves. The development that reshaped the field was the arrival of pose estimators that recover skeletons from RGB at interactive rates.

3.3 From pixel-domain video models to skeletons

Before skeletons became the dominant representation for gesture systems, recogni-tion operated directly on video. Improved dense trajectories [9] marked the high point of hand-crafted descriptors. Deep convolutional networks [10, 11] then replaced hand engineering, and video architectures followed: two-stream networks fused appear-ance with optical flow [12]; 3D convolutions folded time into the kernel [13, 14]; I3D inflated image-pretrained filters into 3D [15]; SlowFast processed a clip at two tem-poral rates [16]; and recurrent–convolutional hybrids separated spatial encoding from temporal aggregation [17]. For hand gestures, recurrent 3D CNNs enabled online detec-tion and classification [18], and comparisons of temporal aggregation found explicit temporal modelling superior to pooling [19]. These models reach high accuracy, but their cost grows with input resolution and clip length, which motivated the turn to compact skeletal input described next.

3.4 The generic pipeline

Fig. 1 shows the stages common to almost every system reviewed here. A camera supplies frames; a pose estimator converts each frame to keypoints; normalisation

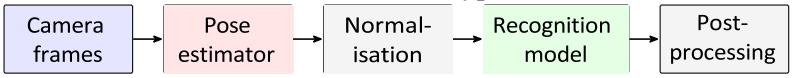


Fig. 1 Generic skeleton-based gesture recognition pipeline.

removes nuisance factors such as position in the frame and distance from the camera; a recognition model classifies a window of keypoints; and post-processing turns per-window predictions into discrete gesture events. Each stage can lose accuracy and add latency.

3.5 Why skeletons

A skeletal sequence replaces a tensor of pixels with a short list of joint coordinates per frame. The compression is several orders of magnitude, and with it comes a measure of invariance: clothing, skin tone, background and much of the lighting variation are discarded before the recognition model sees the data. The representation is geometrically interpretable, so prior knowledge about anatomy can be built into architectures. The costs are two. Information the skeleton omits—object context, fine appearance cues—is lost, and every error the pose estimator makes is inherited downstream. Both costs recur throughout this review.

4 Pose-Estimation Front-Ends

The pose estimator is the first and, in runtime terms, usually the most expensive component of a skeleton pipeline. Three lines of work dominate (Table 2).

Bottom-up multi-person estimation. OpenPose [20] detects body parts across the image and assembles them into individuals using part affinity fields that encode limb direction. Its cost is largely independent of the number of people in view, which made it the default academic front-end for several years.

High-resolution top-down estimation. HRNet [21] maintains high-resolution representations throughout the network rather than recovering resolution after heavy downsampling, and became a strong accuracy reference for 2D keypoints. Sev-eral recognition works use HRNet-style 2D poses when depth-camera skeletons are unavailable; PoseC3D [22] builds its heatmap representation on such poses.

On-device pipelines. MediaPipe [23] packages perception components as graphs optimised for mobile and desktop CPUs. Its body model, BlazePose [24], predicts 33 body landmarks; MediaPipe Hands [25] predicts 21 landmarks per hand using a palm detector followed by a landmark model; and a holistic configuration combines body, hands and face. Several recent gesture and sign recognition systems are built directly on this output [26–28].

Inherited failure modes. A recognition model cannot recover information the front-end did not deliver. Known failure cases include missed or mislocated hands under fast motion, low resolution and unusual orientation; Moryossef et al. [29] show that the hand region-of-interest step of the public MediaPipe Holistic pipeline degrades when the hand plane does not face the camera, and propose a correction. The sign

Table 2 Pose-estimation front-ends used in skeleton pipelines.

Front-end	Output	Approach	Design target
OpenPose [20]	2D body, multi-person	Bottom-up, part affinity fields	GPU
HRNet [21]	2D body keypoints	Top-down, high-resolution features	GPU, accuracy
BlazePose [24]	33 body landmarks	Detector + tracker	On-device
MediaPipe Hands [25]	21 points per hand	Palm detector + landmark model	On-device
Kinect v2 (NTU) [7]	25 body joints	Depth sensor	Dedicated hardware

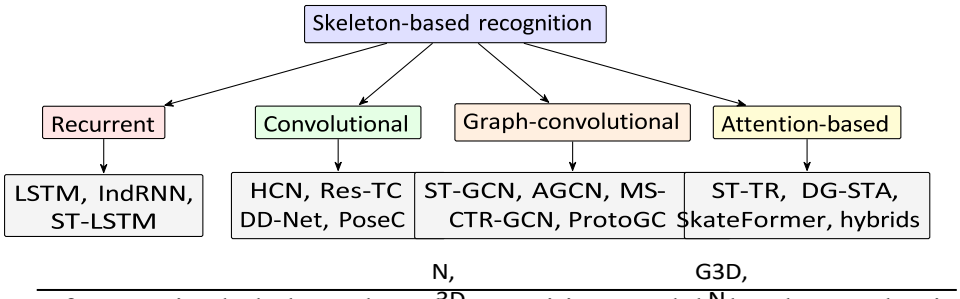


Fig. 2 Taxonomy of supervised skeleton-based recognition models by the mechanism relating joints and frames.

language literature reports the same dependency: the authors of WLASL note that errors of their off-the-shelf pose estimator may limit pose-based recognition [30]. When a hand drops out for a few frames, the downstream model sees nothing or a placeholder, and how the gap is represented affects accuracy in ways benchmark papers rarely report. Section 9 shows a further, quantitative consequence: changing the estimator that produced a benchmark's skeletons shifts reported accuracy substantially.

5 Supervised Recognition Models

Fig. 2 organises supervised models by the mechanism used to relate joints in space and frames in time. The families overlap—many strong methods combine two—but each grew from a distinct view of what a skeleton is.

5.1 Recurrent models

The first deep models on skeleton sequences treated each frame's joint coordinates as a vector and fed the sequence to a recurrent network. The LSTM [31] and GRU [32] sup-plied the gating needed to carry information across long sequences. The NTU RGB+D paper [7] reported LSTM baselines alongside the dataset, and ST-LSTM [33] extended recurrence across the spatial dimension by visiting joints in a tree-traversal order, introducing trust gates to discount unreliable joints—an early, explicit acknowledge-ment of pose noise. IndRNN [34] made neurons within a layer independent, allowing deeper and longer recurrent stacks to train stably.

The weakness of the flat-vector view is that it hides the body's structure: the net-work must learn from data that the wrist moves with the elbow. Imposing a traversal order recovers some structure at the cost of an arbitrary choice. Recurrent models nev-ertheless remain attractive for deployment because they are small and process frames incrementally, and they continue to appear as the temporal stage of hybrid designs.

5.2 Convolutional models

Convolutional approaches treat the skeleton sequence as an image-like array, with time on one axis and joints on another, or apply one-dimensional convolutions along time. HCN [35] learned co-occurrence features by moving the joint dimension into the channel axis so that convolutions aggregate global joint interactions. Temporal convolutional networks such as Res-TCN [36] replaced recurrence with stacked residual 1D convolutions, gaining parallelism and interpretability. DD-Net [37] is notable for deployment: combining joint-distance features with two-scale motion features in a lightweight 1D-convolutional network, it reports state-of-the-art results on SHREC with about 0.15 M parameters.

A distinct route converts poses back into dense form. PoseC3D [22] stacks per-joint heatmaps over time into a volume processed by a 3D CNN, and reports greater robustness to pose-estimation noise than graph methods together with straightforward fusion with RGB. The representation gives up some of the compactness that motivated skeletons in the first place.

5.3 Graph-convolutional models

Graph convolutional networks treat the skeleton as a graph whose nodes are joints and whose edges are bones. ST-GCN [38] defined a spatio-temporal graph with anatomical edges within each frame and temporal edges linking each joint to itself across frames. Removing arbitrary traversal orders and encoding an explicit anatomical prior made ST-GCN the baseline against which subsequent work was measured.

Subsequent work relaxed the fixed graph in several directions. AS-GCN [39] inferred action-specific links beyond the physical skeleton. 2s-AGCN [40] made the adjacency learnable and added a stream over bone vectors fused with the joint stream; DGNN [41] modelled the skeleton as a directed graph with jointly updated node and edge features. DC-GCN [42] decoupled the graph across channel groups and introduced a graph-specific dropout. Shift-GCN [43] replaced graph convolution with shift operations to reduce computation. MS-G3D [44] disentangled multi-scale neighbourhoods and introduced unified spatio-temporal graph convolution with cross-frame edges, and MST-GCN [45] decomposed spatial and temporal graph convolution into multi-scale sub-modules. CTR-GCN [46] learned a shared topology refined per channel; InfoGCN [47] combined attention-based graph convolution with an information-bottleneck objective.

Work since 2022 has largely refined topology learning and training objectives. PYSKL [48] consolidated good training practices into a strong common baseline. Skele-tonGCL [49] and FR-Head [50] added contrastive objectives to supervised training; GAP [51] used generated textual descriptions of actions as additional supervision;

HD-GCN [52] built hierarchically decomposed graphs to capture distant relation-ships; DS-GCN [53] introduced dynamic semantic graph construction; BlockGCN [54] redefined topology awareness while holding the model near 1.3 M parameters; and Pro-toGCN [55] represented motion as combinations of learnable prototypes to separate similar actions.

Two observations matter for deployment. First, the standard formulation presumes a single connected skeleton. The Kinect body has no detailed hands, and attaching two 21-joint hand graphs at the wrists yields a graph dominated by hand nodes. Second, most leading results are ensembles of several input streams—joint, bone and their motions—trained as separate networks and fused at the score level, with implications examined in Section 11.

5.4 Attention-based and hybrid models

Self-attention [56] relates every element of a sequence to every other through learned, content-dependent weights. ST-TR [57] applied spatial self-attention among joints and temporal self-attention across frames. STF [58] focused attention on informa-tive spatio-temporal regions, JT-GraphFormer [59] fused joint trajectories through a graph-Transformer design, and SkateFormer [60] partitioned joints and frames by type and applied attention within partitions, reaching 93.0% NTU-60 cross-subject accu-racy with joint and bone streams at about 3.6 M parameters per stream. For hand gestures, DG-STA [61] built a fully connected hand graph whose edge weights are learned by spatio-temporal attention, and STA-Res-TCN [62] added attention to tem-poral convolution. AGC-LSTM [63] is an early hybrid that embeds graph convolution in an LSTM cell and adds attention over key joints.

Attention's quadratic cost in sequence length matters little for gesture windows of a few dozen frames, which makes hybrids that place a single attention layer over a recurrent or convolutional backbone a natural fit for real-time use. The attention weights also indicate which frames or joints a prediction relied on.

6 Self-Supervised Skeleton Representation Learning

Supervised training requires labels for every class a system must recognise, which is the main obstacle to deploying gesture systems with new vocabularies. Self-supervised learning pre-trains an encoder on unlabelled skeleton sequences and then adapts it with few or no labels.

Contrastive methods adapt the instance-discrimination frameworks of Sim-CLR [64] and MoCo [65]: two augmented views of a sequence should map to nearby embeddings and different sequences to distant ones. CrosSCLR [66] exploited consistency across joint, motion and bone views; AimCLR [67] used extreme augmen-tations to enrich positive pairs; CMD [68] distilled knowledge between modalities; HaLP [69]

hallucinated latent positives; and ActCLR [70] contrasted the discriminative "actionlet" regions of a motion separately from the rest.

Generative and masked-prediction methods reconstruct or predict parts of the input. LongT GAN [71] learned representations by reconstructing sequences;

Table 3 Linear-evaluation top-1 accuracy (%) of self-supervised methods on NTU RGB+D 60, as reported in the original works or in the benchmark of the cited later works.

Method	Paradigm	Streams	X-Sub	X-View
LongT GAN [71]	Generative	1	39.1	–
CrosSCLR [66]	Contrastive	3	77.8	83.4
AimCLR [67]	Contrastive	3	78.9	83.8
CMD [68]	Contrastive	1	79.4	86.9
HaLP [69]	Contrastive	1	79.7	86.8
ActCLR [70]	Contrastive	3	84.3	88.8
MAMP [73]	Masked prediction	1	84.9	89.1

SkeletonMAE [72] applied masked autoencoding to skeleton graphs; and MAMP [73] predicted the motion of masked joints rather than their positions.

Table 3 lists reported linear-evaluation accuracy on NTU 60, where the pre-trained encoder is frozen and only a linear classifier is trained. The gap between early gen-erative pre-training and recent methods is large, and MAMP's single-stream linear result exceeds three-stream contrastive results. When fine-tuned end-to-end, MAMP reports 93.1% and 97.5% [73], comparable to the best supervised models in Table 6. For deployment, the relevance is direct: an encoder pre-trained once on abundant unlabelled motion can be adapted to a new gesture vocabulary with a small labelled set.

7 Pose-Based Sign Language Recognition

Sign language recognition (SLR) is the most demanding gesture application: signs combine hand shape, hand trajectory, body posture and facial cues, vocabularies run to thousands of signs, and many signs differ only in fine finger detail. It is also where the trade-off between pose-based and appearance-based input is most visible.

WLASL [30] introduced a word-level American Sign Language dataset of more than 2,000 glosses performed by over 100 signers, together with appearance-based and pose-based baselines. On its 2,000-gloss split, the appearance-based I3D baseline reached 32.48% top-1 accuracy and the pose-based Pose-TGCN 23.65% (Table 4). Later work narrowed and then reversed the gap. SignBERT [74] pre-trained a hand-model-aware representation on pose sequences; SAM-SLR [75] combined a sign-language graph network over keypoints with other modalities in an ensemble; NLA-SLR [76] fused RGB and keypoints with language-derived supervision; and Uni-Sign [77] pursued large-scale unified pre-training.

OpenHands [78] is particularly relevant to deployment. It standardised pose-based SLR across several languages, including the Indian Sign Language dataset INCLUDE [79], and found graph-based models to be the strongest pose-based option across datasets. Its authors also treated latency as a first-class criterion, arguing that interactive SLR needs at least about 23 FPS, and chose ST-GCN as a deployment trade-off because it ran roughly twice as fast as SL-GCN for about three points lower average accuracy [78].

Table 4 Reported top-1 accuracy (%) on WLASL2000 (per-instance) and INCLUDE. Input types differ; rows are not strictly comparable.

Method	Dataset	Input	Top-1
I3D baseline [30]	WLASL2000	RGB	32.48
Pose-TGCN [30]	WLASL2000	Pose	23.65
SL-GCN (OpenHands) [78]	WLASL2000	Pose	30.6
SignBERT [74]	WLASL2000	Hand pose + RGB, extra data	54.69
SAM-SLR [75]	WLASL2000	Multimodal ensemble	58.73
NLA-SLR [76]	WLASL2000	RGB + keypoints	61.26
Uni-Sign [77]	WLASL2000	Large-scale pre-training	63.52
SL-GCN (OpenHands) [78]	INCLUDE (Indian SL)	Pose	93.5

Table 5 Benchmark datasets for gesture and action recognition.

Dataset	Classes	Samples	Subj.	Scope	Modality
NTU RGB+D 60 [7]	60	56,880	40	Body	RGB+D, skeleton 25-joint
NTU RGB+D 120 [8]	120	114,480	106	Body	RGB+D, skeleton 25-joint
Kinetics-Skeleton [80]	400	~260k	–	Body (in the wild)	Estimated 2D pose
FineGYM [81]	99	29k	–	Gymnastics	RGB / estimated pose
Jester [82]	27	148,092	–	Hands	RGB webcam
SHREC'17 [83]	14/28	2,800	–	Hands	Depth, 22-joint hand
DHG-14/28 [84]	14/28	2,800	20	Hands	Depth, 22-joint hand
IsoGD [85]	249	47,933	21	Mixed	RGB+D
EgoGesture [86]	83	24,161	50	Hands (egocentric)	RGB+D
NVGesture [18]	25	1,532	20	Hands (in-car)	RGB+D+IR
IPN Hand [87]	13	4,218	50	Hands (continuous)	RGB

WLASL [30]	2,000	–	100+	Sign language	RGB

The contrast between WLASL2000 and INCLUDE in Table 4 illustrates how strongly vocabulary size and recording conditions govern difficulty: the same pose-based model family reaches over 90% on one dataset and about 30% on the other.

8 Benchmark Datasets and Protocols

Table 5 summarises the benchmarks most used for skeleton- and video-based gesture and action recognition.

Body-action benchmarks. NTU RGB+D 60 [7] defines *cross-subject* (X-Sub) and *cross-view* (X-View) protocols; NTU RGB+D 120 [8] doubles the classes and adds a *cross-setup* (X-Set) protocol. Kinetics-Skeleton [80] contains skeletons estimated from in-the-wild video, and FineGYM [81] tests fine-grained distinctions among gym-nastic elements. These datasets include gesture-like classes but are dominated by daily or sporting activity.

Hand-gesture benchmarks. Jester [82] contains 148,092 short clips recorded by crowd workers on their own webcams, which brings its capture conditions close to consumer deployment; its 27 classes comprise 25 gestures and two contrast classes, and frames are distributed at low resolution and a reduced frame rate. SHREC'17 [83] and DHG-14/28 [84] provide depth-camera hand skeletons for 14 gestures, with a 28-class variant separating one-finger from whole-hand executions. IPN Hand [87] targets continuous recognition and includes realistic non-gesture movement between gestures. **Multimodal benchmarks.** IsoGD [85] offers 249 isolated gesture classes with high inter-class similarity; EgoGesture [86] targets egocentric viewpoints; and NVGes-
ture [18] was recorded in a driving simulator.

Protocol pitfalls. Some works report validation rather than test accuracy on datasets with hidden test labels; ensembles and test-time augmentation are not always disclosed in summary tables; and skeleton results on NTU normally use the dataset's Kinect skeletons, which do not transfer directly to RGB pipelines whose keypoints come from a different estimator and joint layout.

9 Comparative Analysis of Reported Results

9.1 Body actions: NTU RGB+D 60 and 120

Table 6 lists results reported on NTU RGB+D in chronological groups. Recurrent baselines from 2016 sit near 60–70% cross-subject on NTU 60. Graph convolution in 2018 brought a large step, and subsequent refinements added one to three points each until results compressed toward 93–94% X-Sub and about 97–98% X-View. NTU 120, with more classes and subjects, sits about two to four points lower for the same methods.

Fig. 3 plots the NTU 60 X-Sub figures against publication year. The curve rises steeply and then flattens: from 2018 to 2021 the best reported X-Sub figure rose by about eleven points, and from 2021 to 2024 by less than two.

The "Str." column deserves attention. The strongest graph results are typically four- or six-model ensembles. ProtoGCN reports 93.0% X-Sub with two streams, 93.5% with four and 93.8% with six [55]; InfoGCN reports 92.7% with four streams and 93.0% with six [47]; HD-GCN reports 93.0% with four and 93.4% with six [52]; MS-G3D reports 89.4% for its joint stream alone and 91.5% for joint–bone fusion [44]. Ensem-bling buys roughly half a point to two points at a several-fold increase in inference cost—an exchange that is sensible on a leaderboard and questionable on a laptop.

9.2 The front-end effect: Kinetics-Skeleton and FineGYM

Table 7 shows reported results on Kinetics-Skeleton and FineGYM. On Kinetics, the earlier methods in the table cluster between about 31% and 41% top-1, whereas the methods marked †, evaluated on the skeletons provided by the PYSKL toolkit, report

Table 6 Top-1 accuracy (%) on NTU RGB+D 60 and 120 as reported in the original works or in the comparison tables of later works. "Str." is the number of fused input streams where specified.

Method	Family	Str.	NTU 60 X-Sub	NTU 60 X-View	NTU 120 X-Sub	NTU 120 X-Set
Deep LSTM [7]	Recurrent	1	60.7	67.3	–	–
ST-LSTM [33]	Recurrent	1	69.2	77.7	–	–
ST-GCN [38]	Graph	1	81.5	88.3	70.7	73.2
IndRNN [34]	Recurrent	1	81.8	88.0	–	–
HCN [35]	Conv.	–	86.5	91.1	–	–
AS-GCN [39]	Graph	–	86.8	94.2	78.3	79.8
2s-AGCN [40]	Graph	2	88.5	95.1	82.5	84.2
AGC-LSTM [63]	Hybrid	–	89.2	95.0	–	–
SGN [88]	Graph	1	89.0	94.5	79.2	81.5
DGNN [41]	Graph	–	89.9	96.1	–	–
Shift-GCN [43]	Graph	4	90.7	96.5	85.9	87.6
DC-GCN+ADG [42]	Graph	–	90.8	96.6	86.5	88.1
MS-G3D [44]	Graph	2	91.5	96.2	86.9	88.4
MST-GCN [45]	Graph	4	91.5	96.6	87.5	88.8
CTR-GCN [46]	Graph	4	92.4	96.8	88.9	90.6
STF [58]	Attention	–	92.5	96.9	88.9	89.9
PYSKL [48]	Graph	–	92.6	97.4	88.6	90.8
InfoGCN [47]	Graph attn.	+6	93.0	97.1	89.8	91.2
PoseC3D [22]	Conv. (3D)	–	94.1	97.1	86.9	90.3
SkeletonGCL [49]	Graph	–	92.8	97.1	89.8	91.2
FR-Head [50]	Graph	–	92.8	96.8	89.5	90.9
GAP [51]	Graph	–	92.9	97.0	89.9	91.1
HD-GCN [52]	Graph	6	93.4	97.2	90.1	91.6
SkateFormer [60]	Attention	2	93.0	97.4	89.4	91.0
DS-GCN [53]	Graph	–	93.1	97.5	89.2	91.1
BlockGCN [54]	Graph	–	93.1	97.0	90.3	91.5
JT-GraphFormer [59]	Graph attn.	+–	93.4	97.5	89.9	91.7
ProtoGCN [55]	Graph	6	93.8	97.8	90.9	92.2

about 49–52% [48, 53, 55]. Part of this jump reflects better recognition models, but part reflects different input: the skeletons themselves were re-extracted. The comparison is a concrete instance of the dependency stressed in Section 4—accuracy claims are claims about the model *and* its pose estimator together.

9.3 Hand gestures: SHREC'17

Table 8 collects reported results on SHREC'17. The gap between the 14- and 28-class settings is consistently larger for earlier methods, confirming that the distinction between one-finger and whole-hand execution is where models differ most. DD-Net reaches 94.6% and 91.9% with about 0.15 M parameters [37], a reminder that hand-skeleton recognition does not require large models. PointLSTM's higher figures come from point-cloud input and are listed for reference.

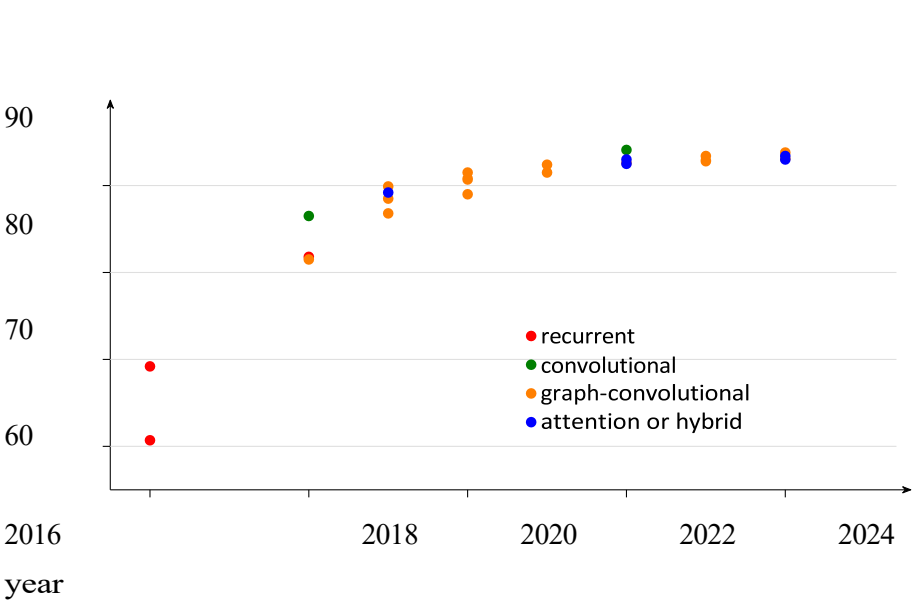


Fig. 3 Reported NTU RGB+D 60 cross-subject accuracy by publication year, coloured by model family (data from Table 6).

Table 7 Reported top-1 accuracy (%) on Kinetics-Skeleton and mean-class accuracy (%) on FineGYM. Rows marked † use skeletons provided by the PYSKL toolkit, which differ from the originally distributed Kinetics skeletons.

Method	Kinetics-Skeleton	FineGYM
ST-GCN [38]	30.7	–
AS-GCN [39]	34.8	–
2s-AGCN [40]	36.1	–
MS-G3D [44]	38.0	–
HD-GCN [52]	40.9	–
PYSKL† [48]	49.1	94.1
DS-GCN† [53]	50.6	–
ProtoGCN† [55]	51.9	95.9
PoseC3D	–	94.3

[22]

9.4 Video-based hand-gesture benchmarks

For Jester, IsoGD, NVGesture and EgoGesture, most published results are pixel-domain, because these datasets were designed for RGB or RGB-D processing. Motion Fused Frames [90] fuse motion information into the input frames and report 96.28% on Jester, 57.4% on IsoGD and 84.7% on NVGesture. The detector–classifier hierarchy of Köpüklü et al. [91], designed for real-time operation, reports 94.04% on EgoGes-ture and 83.82% on NVGesture with depth input. The IsoGD figure shows how far large-vocabulary, high-similarity gesture recognition remains from saturation.

Table 8 Accuracy (%) on SHREC'17 as reported in the original works.

Method	Input	14 gestures	28 gestures
De Smedt et al. [83]	Skeleton	88.2	81.9
Res-TCN [36]	Skeleton	91.1	87.3
STA-Res-TCN [62]	Skeleton	93.6	90.7
DG-STA [61]	Skeleton	94.4	90.7
DD-Net [37]	Skeleton	94.6	91.9
PointLSTM [89]	Point cloud	95.9	94.7

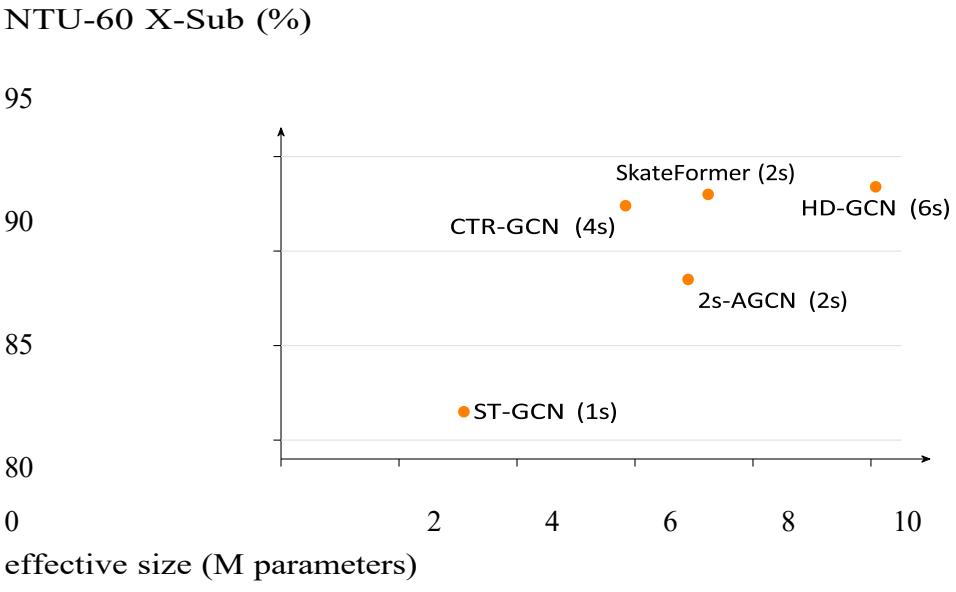


Fig. 4 Reported NTU 60 cross-subject accuracy against effective inference size (per-stream parameters × number of fused streams), from values reported for ST-GCN [38], 2s-AGCN [40], CTR-GCN [46], SkateFormer [60] and HD-GCN [52].

9.5 Accuracy against effective inference size

Fig. 4 relates reported NTU 60 X-Sub accuracy to an *effective* inference size, com-puted as the reported per-stream parameter count multiplied by the number of fused streams. Accuracy is not a simple function of size: CTR-GCN at about 5.8 M effec-tive parameters outperforms 2s-AGCN at 6.9 M, while HD-GCN's additional point over CTR-GCN comes at roughly 1.7 times the effective size. A deployment-oriented reading of

the leaderboard should therefore compare single-stream results alongside ensemble results.

Caveat on comparability. Skeleton results on NTU and SHREC use depth-camera skeletons supplied with the datasets. Pipelines that extract keypoints from RGB with an estimator such as MediaPipe operate on a different joint set with different noise characteristics, and their numbers should not be compared directly with these tables without re-running baselines on the same keypoints.

10 Input Representation, Training, and Fusion Practices

Architectural novelty attracts most attention, but several less visible choices account for a substantial share of reported gains, as the strong PYSKL baseline [48] illustrates. **Input normalisation.** Raw joint coordinates depend on where the person stands and how far they are from the camera. Common remedies translate coordinates relative to a root joint, rescale by a body-size reference such as shoulder width, and sometimes rotate the skeleton to a canonical facing direction. For hands, coordinates are expressed
relative to the wrist and scaled by palm size.

Derived streams. Beyond joint positions, many methods compute bone vectors and frame-to-frame motion. 2s-AGCN [40] popularised joint–bone two-stream training, later extended to four or six streams. DD-Net instead derives joint-distance features within one compact network [37], obtaining part of the same benefit without separate models.

Augmentation. Because augmentation operates on coordinates, it is inexpen-sive: rotation and scaling, small Gaussian jitter, temporal cropping and resampling, and horizontal reflection with left/right relabelling are standard. Skeleton augmen-tation can also simulate estimator failure, for example by dropping joints or whole hands during training. The self-supervised methods of Section 6 rely heavily on such augmentations to construct training signals [67].

Auxiliary objectives. Several recent supervised methods add objectives beyond cross-entropy: contrastive terms over features or graphs [49, 50], prototype-based reconstruction [55], information-bottleneck regularisation [47], and text-derived super-vision [51]. These change training cost but not inference cost, which makes them attractive for deployment.

Optimisation and regularisation. Most models in this review are trained with standard components—adaptive optimisers such as Adam [92], batch nor-malisation [93] and dropout [94]—typically implemented in PyTorch [95]. Training schedules and regularisation strength differ between papers and contribute to reported differences, which is one reason consolidated baselines such as PYSKL [48] are valuable.

Temporal sampling. Clips vary in length and frame rate; Jester's frames, for instance, are distributed at a lower frame rate than typical webcam capture [82]. Meth-ods resample to a fixed window, and mismatches between training and deployment frame rates are an easily overlooked source of accuracy loss.

Multimodal fusion. Where RGB, depth or optical flow are available, fusion can recover context the skeleton discards, at early, late or intermediate stages. Motion Fused Frames [90] fuse motion into appearance; PoseC3D's heatmaps can be fused with RGB early [22]; and the strongest sign language systems in Table 4 combine keypoints with RGB. For real-time systems, every added modality carries its own extraction cost.

11 Efficiency and Real-Time Evidence

Table 9 collects parameter counts and runtime figures where original works or later comparison tables report them.

Table 9 Reported model size and runtime evidence. Runtime figures come from different hardware and are not directly comparable across rows.

Method	Params	Reported runtime / notes
DD-Net [37]	0.15 M	~3,500 FPS on GPU; ~2,000 FPS on CPU
BlockGCN [54]	1.3 M	single model
CTR-GCN [46]	1.46 M	per stream; four streams fused
HD-GCN [52]	1.68 M	per stream; up to six fused

ST-GCN [38]	3.1 M	single stream
SkateFormer [60]	3.62 M	per stream; two streams fused
2s-AGCN [40]	6.9 M	two streams combined
ST-GCN vs. SL-GCN (OpenHands) [78]	–	ST-GCN ≈ 2× faster, ≈ 3 points lower

Three points emerge. First, the recognition network is rarely the bottleneck. A model of one to a few million parameters operating on a few dozen joints for a few dozen frames is cheap; the pose estimator that produces the joints typically dominates end-to-end latency on a CPU. Studies reporting recognition-only throughput therefore overstate the speed of the full system.

Second, ensembles multiply cost linearly. A six-stream result requires six forward passes per prediction plus the preprocessing that derives bone and motion streams. A deployed system must either accept that cost or choose a single-stream configuration and the half point to two points it gives up (Section 9).

Third, purpose-built lightweight designs exist and perform well. DD-Net [37] and SGN [88] were explicitly motivated by efficiency, Shift-GCN [43] replaced costly graph convolutions with shift operations, and BlockGCN [54] reaches 93.1% NTU 60 X-Sub at about 1.3 M parameters. On the video side, the Temporal Shift Module [96] adds temporal modelling to 2D CNNs at almost no extra computation. The general effi-ciency toolkit—depthwise separable convolution [97, 98], compound model scaling [99], knowledge distillation [100] and integer quantisation [101]—applies to skeleton models as well, though their small size makes aggressive compression less necessary than for pixel-domain networks.

What is missing is consistent reporting. Few works state end-to-end latency includ-ing pose estimation, the hardware it was measured on, tail latency rather than averages, or behaviour on CPU-only and edge devices. OpenHands [78] is an instruc-tive exception in treating frame rate as a selection criterion. Without such reporting, efficiency claims cannot be placed on a common scale.

12 Design Considerations for Real-Time Systems

Drawing the preceding sections together, the following considerations recur for practitioners building an interactive skeleton-based gesture system on commodity hardware.

Budget latency end to end. Interactive use tolerates roughly a tenth of a second between action and response. Since pose estimation usually dominates, the recognition model should fit the remainder of the budget, and measurement should include capture, extraction, inference and post-processing.

Prefer single strong models at deployment. Multi-stream ensembles estab-lish what accuracy is achievable, but a single model—possibly distilled from an ensemble [100]—is usually the right deployment artefact.

Treat the front-end as part of the model. Section 9 shows that changing the source of skeletons moves benchmark accuracy substantially. Train on keypoints from the same estimator used at deployment, or re-extract benchmark skeletons with it.

Match the skeleton to the vocabulary. Body-only skeletons cannot distin-guish finger configurations; hand-only skeletons cannot see arm trajectories. Mixed vocabularies require a front-end delivering both and an architecture that does not let one articulator's nodes swamp the other's.

Represent missing data explicitly. Interpolating short gaps and flagging longer ones lets a model distinguish "hand at the origin" from "hand not observed," and training with simulated dropouts prepares it for both.

Exploit pre-training for new vocabularies. Where labelled data for a new gesture set is scarce, an encoder pre-trained with self-supervision (Section 6) can be adapted with far fewer labels than training from scratch.

Decouple and smooth the pipeline. Running capture, pose estimation and classification as concurrent stages with bounded buffers prevents a momentary slowdown in one stage from stalling the others. Sliding-window inference with a stride above one frame reduces computation, and confidence thresholds with short majority-vote filters convert flickering per-window predictions into stable events.

Quantise and export. Post-training integer quantisation [101] and export to a portable runtime such as ONNX [102] reduce size and CPU latency with little accuracy cost for small models.

13 Open Problems and Research Directions

Robustness to keypoint noise. Benchmark skeletons are cleaner than those an RGB estimator delivers in a dim room, under fast motion or with partial occlusion. ST-LSTM's trust gates [33] and PoseC3D's heatmaps [22] address noise directly, but most methods do not, and few papers evaluate under systematically degraded input. Standardised perturbation suites—reduced lighting, occluded articulators, increased distance—would make robustness measurable.

Mixed hand-and-body vocabularies. Practical vocabularies mix gestures car-ried by the fingers, by the arms, and by coordination of both. Benchmarks largely separate these: NTU covers body actions with coarse hands, while SHREC and Jester cover hands alone. Holistic estimators now provide body and hand landmarks together [23, 25], but how to fuse three loosely coupled skeletons remains open, and there is no widely used benchmark for it.

Population and viewpoint generalisation. The persistent gap between X-View and X-Sub accuracy on NTU (Table 6) shows that unseen people remain harder than unseen viewpoints on that dataset. In deployment both vary at once, together with body proportions, handedness and execution style.

Consistent runtime reporting. A reporting convention would help: end-to-end latency including pose estimation, median and tail values, the hardware and software stack, and the number of fused streams.

Continuous-stream spotting. Most results assume pre-segmented clips. Real interaction is continuous: gestures must be detected as well as classified, and tran-sitional movement ignored. Detector–classifier hierarchies [91], online models [18] and continuous benchmarks such as IPN Hand [87] address this, but skeleton-based spotting is far less studied than isolated classification.

Data scarcity for new vocabularies. Deployed systems often need gestures no benchmark contains. Self-supervised pre-training (Section 6), few-shot adaptation and skeleton-level augmentation are promising, but evaluation protocols for adapting to a genuinely new vocabulary with a handful of examples per class are not standardised. **Privacy and on-device processing.** Cameras in homes, clinics and vehicles raise privacy concerns. Skeleton pipelines have an advantage: once keypoints are extracted on the device, raw video need not be stored or transmitted. Designing systems that guarantee this, and measuring what identifying information keypoint
sequences themselves still carry, is an important direction.

14 Conclusion

Skeleton-based gesture recognition has matured from recurrent networks over flat joint vectors to graph, convolutional and attention-based models that exceed 93% on NTU RGB+D 60 under the cross-subject protocol and approach 95% on SHREC'17, while self-supervised pre-training now produces encoders that rival supervised training once fine-tuned. Pose estimators that feed these models run on commodity hardware and deliver body and hand landmarks from a single RGB camera. The remaining distance to deployed interaction lies less in benchmark accuracy than in conditions benchmarks omit: noisy keypoints, mixed hand-and-body vocabularies, unseen users, honest end-to-end latency, continuous input and privacy. The consolidated results also show that leaderboard gains increasingly come from ensembling and from better input skele-tons. Lightweight single-stream models, explicit handling of missing joints, pre-training for new vocabularies, and evaluation protocols that measure robustness and runtime alongside accuracy are, in our assessment, the most productive directions for real-time gesture interfaces.

Acknowledgements. The authors thank the Department of Computer Science and Engineering, Shriram College of Engineering and Management, for its support.

Declarations

- **Funding:** No funding was received for this work.
- **Competing interests:** The authors declare no competing interests.
- **Ethics approval and consent to participate:** Not applicable; this review analyses published literature only.

- **Data availability:** No new data were generated. All figures in the tables are taken from the cited publications.
- **Code availability:** Not applicable.
- **Author contributions:** A.S. conceived the review, surveyed the literature, com-piled and verified the reported results, and wrote the manuscript. [Supervisor initials] supervised the work and revised the manuscript.
- **Use of AI tools:** A large language model-based assistant (Claude, Anthropic) was used to assist with drafting and editing portions of the text and with LaTeX format-ting. The authors reviewed all content, checked the reported figures and references against the original sources, and take full responsibility for the manuscript.

References

[1] Mitra, S., Acharya, T.: Gesture recognition: A survey. IEEE Transactions on Systems, Man, and Cybernetics, Part C **37**(3), 311–324 (2007)

[2] Rautaray, S.S., Agrawal, A.: Vision based hand gesture recognition for human computer interaction: A survey. Artificial Intelligence Review **43**(1), 1–54 (2015)

[3] Ren, B., Liu, M., Ding, R., Liu, H.: A survey on 3D skeleton-based action recognition using learning method. Cyborg and Bionic Systems **5**, 0100 (2024) https://doi.org/10.34133/cbsystems.0100

[4] Meng, X., Li, S., Yuan, M., Wang, W., Zhang, Z., Song, C., Cao, F.: Action recognition based on human skeleton: Review and prospect. Information and Control **54**(1), 1–27 (2025)

[5] Sun, Z., Ke, Q., Rahmani, H., Bennamoun, M., Wang, G., Liu, J.: Human action recognition from various data modalities: A review. IEEE Transactions on Pattern Analysis and Machine Intelligence **45**(3), 3200–3225 (2023)

[6] Foteinos, K., Cani, J., Linardakis, M., Radoglou-Grammatikis, P., Argyriou, V., Sarigiannidis, P., Varlamis, I., Papadopoulos, G.T.: Visual hand gesture recogni-tion with deep learning: A comprehensive review of methods, datasets, challenges and future research directions. arXiv preprint arXiv:2507.04465 (2025)

[7] Shahroudy, A., Liu, J., Ng, T.-T., Wang, G.: NTU RGB+D: A large scale dataset for 3D human activity analysis. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1010–1019 (2016)

[8] Liu, J., Shahroudy, A., Perez, M., Wang, G., Duan, L.-Y., Kot, A.C.: NTU RGB+D 120: A large-scale benchmark for 3D human activity understanding. IEEE Transactions on Pattern Analysis and Machine Intelligence **42**(10), 2684–2701 (2020)

[9] Wang, H., Schmid, C.: Action recognition with improved trajectories. In: IEEE International Conference on Computer Vision (ICCV), pp. 3551–3558 (2013)

[10] Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems (NeurIPS), vol. 25, pp. 1097–1105 (2012)

[11] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778 (2016)

[12] Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. In: Advances in Neural Information Processing Systems (NeurIPS), vol. 27, pp. 568–576 (2014)

[13] Ji, S., Xu, W., Yang, M., Yu, K.: 3D convolutional neural networks for human action recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence **35**(1), 221–231 (2013)

[14] Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spa-tiotemporal features with 3D convolutional networks. In: IEEE International Conference on Computer Vision (ICCV), pp. 4489–4497 (2015)

[15] Carreira, J., Zisserman, A.: Quo vadis, action recognition? A new model and the Kinetics dataset. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4724–4733 (2017)

[16] Feichtenhofer, C., Fan, H., Malik, J., He, K.: SlowFast networks for video recog-nition. In: IEEE/CVF International Conference on Computer Vision (ICCV), pp. 6202–6211 (2019)

[17] Donahue, J., Hendricks, L.A., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., Darrell, T.: Long-term recurrent convolutional networks for visual recognition and description. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2625–2634 (2015)

[18] Molchanov, P., Yang, X., Gupta, S., Kim, K., Tyree, S., Kautz, J.: Online detection and classification of dynamic hand gestures with recurrent 3D convo-lutional neural networks. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4207–4215 (2016)

[19] Pigou, L., Van Den Oord, A., Dieleman, S., Van Herreweghe, M., Dambre, J.: Beyond temporal pooling: Recurrence and temporal convolutions for gesture recognition in video. International Journal of Computer Vision **126**(2–4), 430–439 (2018)

[20] Cao, Z., Simon, T., Wei, S.-E., Sheikh, Y.: Realtime multi-person 2D pose esti-mation using part affinity fields. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 7291–7299 (2017)

[21] Sun, K., Xiao, B., Liu, D., Wang, J.: Deep high-resolution representation learning for human pose estimation. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5693–5703 (2019)

[22] Duan, H., Zhao, Y., Chen, K., Lin, D., Dai, B.: Revisiting skeleton-based action recognition. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2969–2978 (2022)

[23] Lugaresi, C., Tang, J., Nash, H., McClanahan, C., Uboweja, E., Hays, M., Zhang, F., Chang, C.-L., Yong, M.G., Lee, J., Chang, W.-T., Hua, W., Georg, M., Grundmann, M.: MediaPipe: A framework for building perception pipelines. arXiv preprint arXiv:1906.08172 (2019)

[24] Bazarevsky, V., Grishchenko, I., Raveendran, K., Zhu, T., Zhang, F., Grund-mann, M.: BlazePose: On-device real-time body pose tracking. arXiv preprint arXiv:2006.10204 (2020)

[25] Zhang, F., Bazarevsky, V., Vakunov, A., Tkachenka, A., Sung, G., Chang, C.-L., Grundmann, M.: MediaPipe Hands: On-device real-time hand tracking. arXiv preprint arXiv:2006.10214 (2020)

[26] Sundar, B., Bagyammal, T.: American Sign Language recognition for alphabets using MediaPipe and LSTM. In: Procedia Computer Science, vol. 215, pp. 642–651 (2022)

[27] Kamble, S.: SLRNet: A real-time LSTM-based sign language recognition system. arXiv preprint arXiv:2506.11154 (2025)

[28] Garg, M., *et al.*: MP-GestLSTM: Real-time gesture detection using MediaPipe and LSTM. Systems Science & Control Engineering (2025) https://doi.org/10. 1080/21642583.2025.2587853

[29] Moryossef, A., et al.: Optimizing hand region detection in MediaPipe holistic full-body pose estimation to improve accuracy and avoid downstream errors. arXiv preprint arXiv:2405.03545 (2024)

[30] Li, D., Rodriguez Opazo, C., Yu, X., Li, H.: Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison. In: IEEE Winter Conference on Applications of Computer Vision (WACV) (2020)

[31] Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation **9**(8), 1735–1780 (1997)

[32] Cho, K., Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using RNN encoder–decoder for statistical machine translation. In: Proceedings of EMNLP, pp. 1724–1734 (2014)

[33] Liu, J., Shahroudy, A., Xu, D., Wang, G.: Spatio-temporal LSTM with trust gates for 3D human action recognition. In: European Conference on Computer Vision (ECCV), pp. 816–833 (2016)

[34] Li, S., Li, W., Cook, C., Zhu, C., Gao, Y.: Independently recurrent neural net-work (IndRNN): Building a longer and deeper RNN. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5457–5466 (2018)

[35] Li, C., Zhong, Q., Xie, D., Pu, S.: Co-occurrence feature learning from skele-ton data for action recognition and detection with hierarchical aggregation. In: International Joint Conference on Artificial Intelligence (IJCAI), pp. 786–792 (2018)

[36] Kim, T.S., Reiter, A.: Interpretable 3D human action analysis with temporal convolutional networks. In: IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 1623–1631 (2017)

[37] Yang, F., Sakti, S., Wu, Y., Nakamura, S.: Make skeleton-based action recogni-tion model smaller, faster and better. In: ACM Multimedia Asia (2019)

[38] Yan, S., Xiong, Y., Lin, D.: Spatial temporal graph convolutional networks for skeleton-based action recognition. In: AAAI Conference on Artificial Intelligence, vol. 32, pp. 7444–7451 (2018)

[39] Li, M., Chen, S., Chen, X., Zhang, Y., Wang, Y., Tian, Q.: Actional-structural graph convolutional networks for skeleton-based action recognition. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3595–3603 (2019)

[40] Shi, L., Zhang, Y., Cheng, J., Lu, H.: Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 12026–12035 (2019)

[41] Shi, L., Zhang, Y., Cheng, J., Lu, H.: Skeleton-based action recognition with directed graph neural networks. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 7912–7921 (2019)

[42] Cheng, K., Zhang, Y., Cao, C., Shi, L., Cheng, J., Lu, H.: Decoupling GCN with DropGraph module for skeleton-based action recognition. In: European Conference on Computer Vision (ECCV), pp. 536–553 (2020)

[43] Cheng, K., Zhang, Y., He, X., Chen, W., Cheng, J., Lu, H.: Skeleton-based action recognition with shift graph convolutional network. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 183–192 (2020)

[44] Liu, Z., Zhang, H., Chen, Z., Wang, Z., Ouyang, W.: Disentangling and uni-fying graph convolutions for skeleton-based action recognition. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 143–152 (2020)

[45] Chen, Z., Li, S., Yang, B., Li, Q., Liu, H.: Multi-scale spatial temporal graph con-volutional network for skeleton-based action recognition. In: AAAI Conference on Artificial Intelligence, pp. 1113–

1122 (2021)

[46] Chen, Y., Zhang, Z., Yuan, C., Li, B., Deng, Y., Hu, W.: Channel-wise topology refinement graph convolution for skeleton-based action recognition. In: IEEE/CVF International Conference on Computer Vision (ICCV), pp. 13359–13368 (2021)

[47] Chi, H.-g., Ha, M.H., Chi, S., Lee, S.W., Huang, Q., Ramani, K.: InfoGCN: Rep-resentation learning for human skeleton-based action recognition. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 20186–20196 (2022)

[48] Duan, H., Wang, J., Chen, K., Lin, D.: PYSKL: Towards good practices for skeleton action recognition. In: ACM International Conference on Multimedia (ACM MM), pp. 7351–7354 (2022)

[49] Huang, X., Zhou, H., Wang, J., Feng, H., Han, J., Ding, E., Wang, J., Wang, X., Liu, W., Feng, B.: Graph contrastive learning for skeleton-based action recognition. In: International Conference on Learning Representations (ICLR) (2023)

[50] Zhou, H., Liu, Q., Wang, Y.: Learning discriminative representations for skeleton based action recognition. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 10608–10617 (2023)

[51] Xiang, W., Li, C., Zhou, Y., Wang, B., Zhang, L.: Generative action descrip-tion prompts for skeleton-based action recognition. In: IEEE/CVF International Conference on Computer Vision (ICCV), pp. 10276–10285 (2023)

[52] Lee, J., Lee, M., Lee, D., Lee, S.: Hierarchically decomposed graph convolutional networks for skeleton-based action recognition. In: IEEE/CVF International Conference on Computer Vision (ICCV), pp. 10444–10453 (2023)

[53] Xie, J., Meng, Y., Zhao, Y., Nguyen, A., Yang, X., Zheng, Y.: Dynamic semantic-based spatial graph convolution network for skeleton-based human action recognition. In: AAAI Conference on Artificial Intelligence, pp. 6225–6233 (2024)

[54] Zhou, Y., Yan, X., Cheng, Z.-Q., Yan, Y., Dai, Q., Hua, X.-S.: Block-GCN: Redefine topology awareness for skeleton-based action recognition. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2049–2058 (2024)

[55] Liu, H., Liu, Y., Ren, M., Wang, H., Wang, Y., Sun, Z.: Revealing key details to see differences: A novel prototypical perspective for skeleton-based action recognition. arXiv preprint arXiv:2411.18941 (2024)

[56] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł ., Polosukhin, I.: Attention is all you need. In: Advances in Neural Information Processing Systems (NeurIPS), vol. 30, pp. 5998–6008 (2017)

[57] Plizzari, C., Cannici, M., Matteucci, M.: Skeleton-based action recognition via spatial and temporal transformer networks. Computer Vision and Image Understanding **208–209**, 103219 (2021)

[58] Ke, L., Peng, K.-C., Lyu, S.: Towards to-a-t spatio-temporal focus for skeleton-based action recognition. In: AAAI Conference on Artificial Intelligence, pp. 1131–1139 (2022)

[59] Zheng, Y., Huang, H., Wang, X., Yan, X., Xu, L.: Spatio-temporal fusion for human action recognition via joint trajectory graph. In: AAAI Conference on Artificial Intelligence, pp. 7579–7587 (2024)

[60] Do, J., Kim, M.: SkateFormer: Skeletal-temporal transformer for human action recognition. In: European Conference on Computer Vision (ECCV) (2024)

[61] Chen, Y., Zhao, L., Peng, X., Yuan, J., Metaxas, D.N.: Construct dynamic graphs for hand gesture recognition via spatial-temporal attention. In: British Machine Vision Conference (BMVC) (2019)

[62] Hou, J., Wang, G., Chen, X., Xue, J.-H., Zhu, R., Yang, H.: Spatial-temporal attention Res-TCN for skeleton-based dynamic hand gesture recognition. In: European Conference on Computer Vision Workshops (ECCVW) (2018)

[63] Si, C., Chen, W., Wang, W., Wang, L., Tan, T.: An attention enhanced graph convolutional LSTM network for skeleton-based action recognition. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR),
pp. 1227–1236 (2019)

[64] Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: International Conference on Machine Learning (ICML), pp. 1597–1607 (2020)

[65] He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsuper-vised visual representation learning. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 9729–9738 (2020)

[66] Li, L., Wang, M., Ni, B., Wang, H., Yang, J., Zhang, W.: 3D human action repre-sentation learning via cross-view consistency pursuit. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4741–4750 (2021)

[67] Guo, T., Liu, H., Chen, Z., Liu, M., Wang, T., Ding, R.: Contrastive learn-ing from extremely augmented skeleton sequences for self-supervised action recognition. In: AAAI Conference on Artificial Intelligence, pp. 762–770 (2022)

[68] Mao, Y., Zhou, W., Lu, Z., Deng, J., Li, H.: CMD: Self-supervised 3D action representation learning with cross-modal mutual distillation. In: European Conference on Computer Vision (ECCV) (2022)

[69] Shah, A., Roy, A., Shah, K., Mishra, S., Jacobs, D., Cherian, A., Chellappa, R.: HaLP: Hallucinating latent positives for skeleton-based self-supervised learn-ing of actions. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 18846–18856 (2023)

[70] Lin, L., Zhang, J., Liu, J.: Actionlet-dependent contrastive learning for unsuper-vised skeleton-based action recognition. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2023)

[71] Zheng, N., Wen, J., Liu, R., Long, L., Dai, J., Gong, Z.: Unsupervised represen-tation learning with long-term dynamics for skeleton based action recognition. In: AAAI Conference on Artificial Intelligence (2018)

[72] Yan, H., Liu, Y., Wei, Y., Li, Z., Li, G., Lin, L.: SkeletonMAE: Graph-based masked autoencoder for skeleton sequence pre-training. In: IEEE/CVF International Conference on Computer Vision (ICCV), pp. 5606–5618 (2023)

[73] Mao, Y., Deng, J., Zhou, W., Fang, Y., Ouyang, W., Li, H.: Masked motion predictors are strong 3D action representation learners. In: IEEE/CVF Interna-tional Conference on Computer Vision (ICCV) (2023)

[74] Hu, H., Zhao, W., Zhou, W., Wang, Y., Li, H.: SignBERT: Pre-training of hand-model-aware representation for sign language recognition. In: IEEE/CVF International Conference on Computer Vision (ICCV) (2021)

[75] Jiang, S., Sun, B., Wang, L., Bai, Y., Li, K., Fu, Y.: Skeleton aware multi-modal sign language recognition. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops

(CVPRW) (2021)

[76] Zuo, R., Wei, F., Mak, B.: Natural language-assisted sign language recogni-tion. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2023)

[77] Li, Z., *et al.*: Uni-Sign: Toward unified sign language understanding at scale. In: International Conference on Learning Representations (ICLR) (2025)

[78] Selvaraj, P., NC, G., Kumar, P., Khapra, M.: OpenHands: Making sign language recognition accessible with pose-based pretrained models across languages. In: Annual Meeting of the Association for Computational Linguistics (ACL) (2022)

[79] Sridhar, A., Ganesan, R.G., Kumar, P., Khapra, M.: INCLUDE: A large scale dataset for Indian sign language recognition. In: ACM International Conference on Multimedia (ACM MM) (2020)

[80] Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., et al.: The Kinetics human action video dataset. arXiv preprint arXiv:1705.06950 (2017)

[81] Shao, D., Zhao, Y., Dai, B., Lin, D.: FineGym: A hierarchical video dataset for fine-grained action understanding. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2616–2625 (2020)

[82] Materzynska, J., Berger, G., Bax, I., Memisevic, R.: The Jester dataset: A large-scale video dataset of human gestures. In: IEEE/CVF International Conference on Computer Vision Workshops (ICCVW) (2019)

[83] De Smedt, Q., Wannous, H., Vandeborre, J.-P., Guerry, J., Le Saux, B., Filliat, D.: SHREC'17 track: 3D hand gesture recognition using a depth and skeletal dataset. In: Eurographics Workshop on 3D Object Retrieval (2017)

[84] De Smedt, Q., Wannous, H., Vandeborre, J.-P.: Skeleton-based dynamic hand gesture recognition. In: IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 1206–1214 (2016)

[85] Wan, J., Escalera, S., Anbarjafari, G., et al.: ChaLearn looking at people RGB-D isolated and continuous datasets for gesture recognition. IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) (2016)

[86] Zhang, Y., Cao, C., Cheng, J., Lu, H.: EgoGesture: A new dataset and bench-mark for egocentric hand gesture recognition. IEEE Transactions on Multimedia **20**(5), 1038–1050 (2018)

[87] Benitez-Garcia, G., Olivares-Mercado, J., Sanchez-Perez, G., Yanai, K.: IPN Hand: A video dataset and benchmark for real-time continuous hand gesture recognition. In: International Conference on Pattern Recognition (ICPR) (2020)

[88] Zhang, P., Lan, C., Zeng, W., Xing, J., Xue, J., Zheng, N.: Semantics-guided neural networks for efficient skeleton-based human action recognition. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR),
pp. 1112–1121 (2020)

[89] Min, Y., Zhang, Y., Chai, X., Chen, X.: An efficient PointLSTM for point clouds based gesture recognition. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5761–5770 (2020)

[90] Köpüklü, O., Köse, N., Rigoll, G.: Motion fused frames: Data level fusion strat-egy for hand gesture recognition. In: IEEE Conference on Computer Vision and Pattern Recognition Workshops

(CVPRW) (2018)

[91] Köpüklü, O., Gunduz, A., Kose, N., Rigoll, G.: Real-time hand gesture detection and classification using convolutional neural networks. In: IEEE International Conference on Automatic Face & Gesture Recognition (FG) (2019)

[92] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: International Conference on Learning Representations (ICLR) (2015)

[93] Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International Conference on Machine Learning (ICML), pp. 448–456 (2015)

[94] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research **15**(1), 1929–1958 (2014)

[95] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: PyTorch: An imperative style, high-performance deep learning library. In: Advances in Neural Information Processing Systems (NeurIPS), vol. 32, pp. 8024–8035 (2019)

[96] Lin, J., Gan, C., Han, S.: TSM: Temporal shift module for efficient video under-standing. In: IEEE/CVF International Conference on Computer Vision (ICCV),
pp. 7083–7093 (2019)

[97] Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: MobileNets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)

[98] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.-C.: MobileNetV2: Inverted residuals and linear bottlenecks. In: IEEE/CVF Conference on Com-puter Vision and Pattern Recognition (CVPR), pp. 4510–4520 (2018)

[99] Tan, M., Le, Q.V.: EfficientNet: Rethinking model scaling for convolutional neu-ral networks. In: International Conference on Machine Learning (ICML), pp. 6105–6114 (2019)

[100] Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015)

[101] Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H., Kalenichenko, D.: Quantization and training of neural networks for effi-cient integer-arithmetic-only inference. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2704–2713 (2018)

[102] ONNX Runtime developers: ONNX Runtime. https://onnxruntime.ai. Version
1.x; cross-platform inference engine for ONNX models (2018)