15 Keypoints Is All You Need

Michael Snower[†*]   Asim Kadav[‡]   Farley Lai[‡]   Hans Peter Graf[‡]
[†]Brown University   [‡]NEC Labs America
michael_snower@brown.edu   {asim,farleylai,hpg}@nec-labs.com

Abstract

Pose tracking is an important problem that requires identifying unique human pose-instances and matching them temporally across different frames of a video. However, existing pose tracking methods are unable to accurately model temporal relationships and require significant computation, often computing the tracks offline. We present an efficient multi-person pose tracking method, KeyTrack, that only relies on keypoint information without using any RGB or optical flow information to track human keypoints in real-time. Keypoints are tracked using our Pose Entailment method, in which, first, a pair of pose estimates is sampled from different frames in a video and tokenized. Then, a Transformer-based network makes a binary classification as to whether one pose temporally follows another. Furthermore, we improve our top-down pose estimation method with a novel, parameter-free, keypoint refinement technique that improves the keypoint estimates used during the Pose Entailment step. We achieve state-of-the-art results on the PoseTrack'17 and the PoseTrack'18 benchmarks while using only a fraction of the computation required by most other methods for computing the tracking information.

1. Introduction

Multi-person Pose Tracking is an important problem for human action recognition and video understanding. It occurs in two steps: first, estimation, where keypoints of individual persons are localized; second, the tracking step, where each keypoint is assigned to a unique person. Pose tracking methods rely on deep convolutional neural networks for the first step [48, 47, 57, 52], but approaches in the second step vary. This is a challenging problem because tracks must be created for each unique person, while overcoming occlusion and complex motion. Moreover, individuals may appear visually similar because they are wearing the same uniform. It is also important for tracking to be performed online. Commonly used methods, such
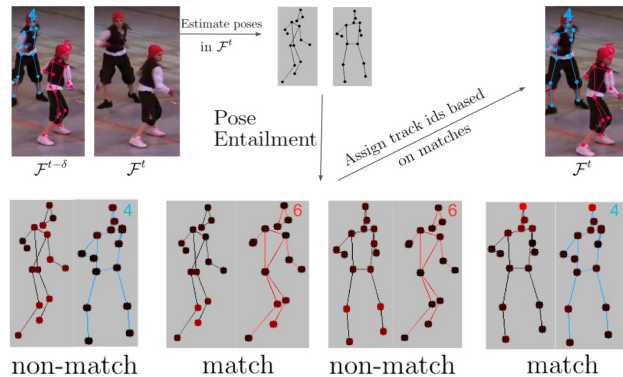
---

[*]Work done as a NEC Labs intern

Figure 1. **They look alike, how do we decide who's who?** In the Pose Entailment framework, given a video frame, we track individuals by comparing pairs of poses, using temporal motion cues to determine who's who. Using a novel tokenization scheme to create pose pair inputs interpretable by Transformers [49], our network divides its attention equally between both poses in *matching pairs*, and focuses more on a single pose in *non-matching* pairs because motion cues between keypoints are not present. We visualize this above; bright red keypoints correspond to high attention.

as optical flow and graph convolutional networks (GCNs) are effective at modeling spatio-temporal keypoint relationships [45], [35], but are dependent on high spatial resolution, making them computationally costly. Non-learning based methods, such as spatial consistency, are faster than the convolution-based methods, but are not as accurate.

To address the above limitations, we propose an efficient pose tracking method, KeyTrack, that leverages temporal relationships to improve multi-person pose estimation and tracking. Hence, KeyTrack follows the *tracking by detection* approach by first localizing humans, estimating human pose keypoints and then encoding the keypoint information in a novel *entailment* setting using transformer building blocks [49]. Similar to the textual entailment task where one has to predict if one sentence follows one another, we propose the *Pose Entailment* task, where the model learns to make a binary classification if two keypoint poses temporally follow or entail each other. Hence, rather than extracting information from a high-dimensional image representation using deep CNNs, we extract information from a sentence of 15 tokens, and each token corresponds to a key-

point on a pose. Similar to how BERT tokenizes words [14], we propose an embedding scheme for pose data that captures spatio-temporal relationships and feed our transformer network these embeddings. Since these embeddings contain information beyond spatial location, our network outperforms convolution-based approaches in terms of accuracy and speed, particularly at very low resolutions.

Additionally, in order to improve the keypoint estimates used by the transformer network, we propose a Temporal Object Keypoint Similarity (TOKS) method. TOKS refines the pose estimation output by augmenting missed detections and thresholding low quality estimates using a keypoint similarity metric. TOKS adds no learned parameters to the estimation step, and is superior to existing bounding box propagation methods that often rely on NMS and optical flow. KeyTrack makes the following contributions:

1. KeyTrack introduces Pose Entailment, where a binary classification is made as to whether two poses from different timesteps are the same person. We model this task in a transformer-based network which learns temporal pose relationships even in datasets with complex motion. Furthermore, we present a tokenization scheme for pose information that allows transformers to outperform convolutions at low spatial resolutions when tracking keypoints.

2. KeyTrack introduces a temporal method for improving keypoint estimates. TOKS is more accurate than bounding box propagation, faster than a detector ensemble, and does not require learned parameters.

Using the above methods, we develop an efficient multi-person pose tracking pipeline which sets a new SOTA on the PoseTrack test set. We achieve 61.2% tracking accuracy on the PoseTrack'17 Test Set and 66.6% on the PoseTrack'18 Val set using a model that consists of just 0.43M parameters in the tracking step. This portion of our pipeline 500X more efficient than than the leading optical flow method [45]. Our training is performed on a single NVIDIA 1080Ti GPU. Not reliant on RGB or optical flow information in the tracking step, our model is suitable to perform pose tracking using other non-visual pose estimation sensors that only provide 15 keypoints for each person [3].

2. Related Work

We are inspired by related work on pose estimation and tracking methods, and recent work on applying the transformer network to video understanding.

Pose estimation Early work on pose estimation uses graphical models to learn spatial correlations and interactions between various joints [5, 16]. These models often perform poorly due to occlusions and long range temporal relationships, which need to be explicitly modeled [12, 42, 51]. More recent work involves using convolutional neural networks (CNNs) to directly regress cartesian

Method	Estimation	Detection Improvement	Tracking	
Ours	HRNet	**Temporal OKS**	**Pose Entailment**	Top-Down
HRNet [45]	HRNet	BBox Prop.	Optical Flow	
POINet [40]	VGG, T-VGG	-	Ovonic Insight Net	
MDPN [20]	MDPN	Ensemble	Optical Flow	
LightTrack [35]	Simple Baselines	Ensemble/BBox Prop.	GCN	
ProTracker [19]	3D Mask RCNN	-	IoU	
Affinity Fields [38]	VGG/STFields	-	STFields	Bottom-Up
STEmbeddings [28]	STEmbeddings	-	STEmbeddings	
JointFlow	Siamese CNN	-	Flow Fields	

Table 1. How different approaches address each step of the Pose Tracking problem. Our contributions are in bold.

coordinates of the joints [48] or to generate heatmaps of the probability of a joint's location [47, 57, 52]. A majority of the convolutional approaches can be classified into top-down and bottom-up methods – the top-down methods use a separate detection step to identify person candidates [21, 37, 10, 24, 37]. The single person pose estimation step is then performed on these person candidates. Bottom-up methods calculate keypoints from all candidates and then correlate these keypoints into individual human joints [53, 25]. The latter method is more efficient since all keypoints are calculated in a single step; however, the former is more accurate since the object detection step limits the regression boundaries. However, top-down methods work poorly on small objects and recent work (HRNet) [45] uses parallel networks at different resolutions to maximize spatial information. PoseWarper [8] uses a pair of labeled and unlabeled frames to predict human pose by learning the pose-warping using deformable convolutions. Finally, since the earliest applications of deep learning to pose estimation [48], iterative predictions have improved accuracy. Pose estimation has shown to benefit from cascaded predictions [10] and pose-refinement methods [17, 34] refine the pose estimation results of previous stages using a separate post-processing network. In that spirit, our work, KeyTrack relies on HRNet to generate keypoints and refines keypoint estimates by temporally aggregating and suppressing low confidence keypoints with TOKS instead of commonly used bounding box propagation approaches.

Pose tracking Methods Pose tracking methods assign unique IDs to individual keypoints, estimated with techniques described in the previous subsection, to track them through time [4, 26, 27, 1]. Some methods perform tracking by learning spatio-temporal pose relationships across video frames using convolutions [50, 40, 35]. [40], in an end-to-end fashion, predicts track ids with embedded visual features from its estimation step, making predictions in multiple temporal directions. [35] uses a GCN to track poses based on spatio-temporal keypoint relationships. These networks require high spatial resolutions. In contrast, we create keypoint embeddings from the keypoint's spatial location and other information making our network less reliant
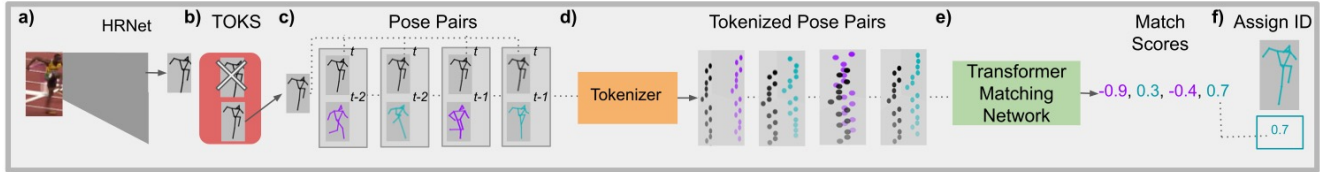
Figure 2. **a)** Keypoints are estimated with HRNet. **b)** TOKS improves detection accuracy. **c)** Pose pairs are collected from multiple past timesteps. Poses of the same color have the same track id, the color black indicates the track id is unknown. **d)** Each pair is tokenized independently from the other pairs. **e)** Our Transformer Matching Network calculates match scores independently for each pair. **f)** The maximum match score is greedily chosen and the corresponding track id is assigned.

on spatial resolution, and thus more efficient. We can also model more fine-grained spatio-temporal relationships.

Among non-learned tracking methods, optical flow propagates poses from one frame to the next to determine which pose they are most similar to in the next frame [45, 20]. This improves over spatial consistency, which measures the IoU between bounding boxes of poses from temporally adjacent frames [19]. Other methods use graph-partitioning based approaches to group pose tracks [26, 27, 29]. Another method, PoseFlow [55], uses inter/intra-frame pose distance and NMS to construct pose flows. However, our method does not require hard-coded parameters during inference, this limits the ability of non-learned methods to model scenes with complex motion and requires time-intensive manual tuning. Table 1 shows top-down methods similar to our work as well as competitive bottom-up methods.

Transformer Models Recently, there have been successful implementations of transformer-based models for image and video input modalities often substituting convolutions and recurrence mechanisms. These methods can efficiently model higher-order relationships between various scene elements unlike pair-wise methods [11, 22, 41, 56]. They have been applied for image classification [39], visual question-answering [30, 31, 46, 60], action-recognition [23, 32], video captioning [44, 61] and other video problems. Video-Action Transformer [18] solves the action localization problem using transformers by learning the context and interactions for every person in the video. BERT [13] uses transformers by pretraining a transformer-based network in a multi-task transfer learning scheme over the unsupervised tasks of predicting missing words or next sentences. Instead, in a supervised setting, KeyTrack uses transformers to learn spatio-temporal keypoint relationships for the visual problem of pose tracking.

3. Method

3.1. Overview of Our Approach

We now describe the keypoint estimation and tracking approach used in KeyTrack as shown in Figure 2. For frame $\mathcal{F}^t$ at timestep t, we wish to assign a track id to the ith

pose $p^{t,i} \in \mathcal{P}^t$. First, each of the pose's $k^j \in \mathcal{K}$ keypoints are detected. This is done by localizing a bounding box around each pose with an object detector and then estimating keypoint locations in the box. Keypoint predictions are improved with temporal OKS (TOKS). Please see 3.3 for more details. From here, this pose with no tracking id, $p_{\varnothing}^{t,i}$, is assigned its appropriate one. This is based on the pose's similarity to a pose in a previous timestep, which has an id, $p_{id}^{t-\delta,j}$. Similarity is measured with the match score, $m_{id}^{t-\delta,j}$, using Pose Entailment (3.2).

False negatives are an inevitable problem in keypoint detection, and hurt the downstream tracking step because poses with the correct track id may appear to be no longer in the video. We mitigate this by calculating match scores for poses in not just one previous frame, but multiple frames $\{\mathcal{F}^1, \mathcal{F}^2, \dots \mathcal{F}^\delta\}$. Thus, we compare to each pose $p_{id}^{t-d,j}$ where $1 \leqslant d \leqslant \delta$ and $1 \leqslant j \leqslant |\mathcal{P}^{t-d}|$. In practice, we limit the number of poses we compare to in a given frame to the n spatially nearest poses. This is just as accurate as comparing to everyone in the frame and bounds our runtime to $O(\delta n)$. This gives us a set of match scores $\mathcal{M}$, and we assign $p_{\varnothing}^{t,i}$ the track id corresponding to the maximum match score $m^* = max(\mathcal{M})$, where $id^* = m_{id}^*$. Thus, we assign the tracking id to the pose, $p_{id*}^{t,i}$.

3.2. Pose Entailment

To effectively solve the multi-person pose tracking problem, we need to understand how human poses move through time based on spatial joint configurations as well as in the presence of multiple persons and occluding objects. Hence, we need to learn if a pose in timestep t, can be inferred from timestep $t-1$. Textual entailment provides us with a similar framework in the NLP domain where one needs to understand if one sentence can be implied from the next. More specifically, the textual entailment model classifies whether a *premise* sentence implies a *hypothesis* sentence in a sentence pair [9]. The typical approach to this problem consists of first projecting the pair of sentences to an embedding space and then feeding them through a neural network which outputs a binary classification for the sentence pair.

Hence, we propose the Pose Entailment problem. More formally, we seek to classify whether a pose in a timestep
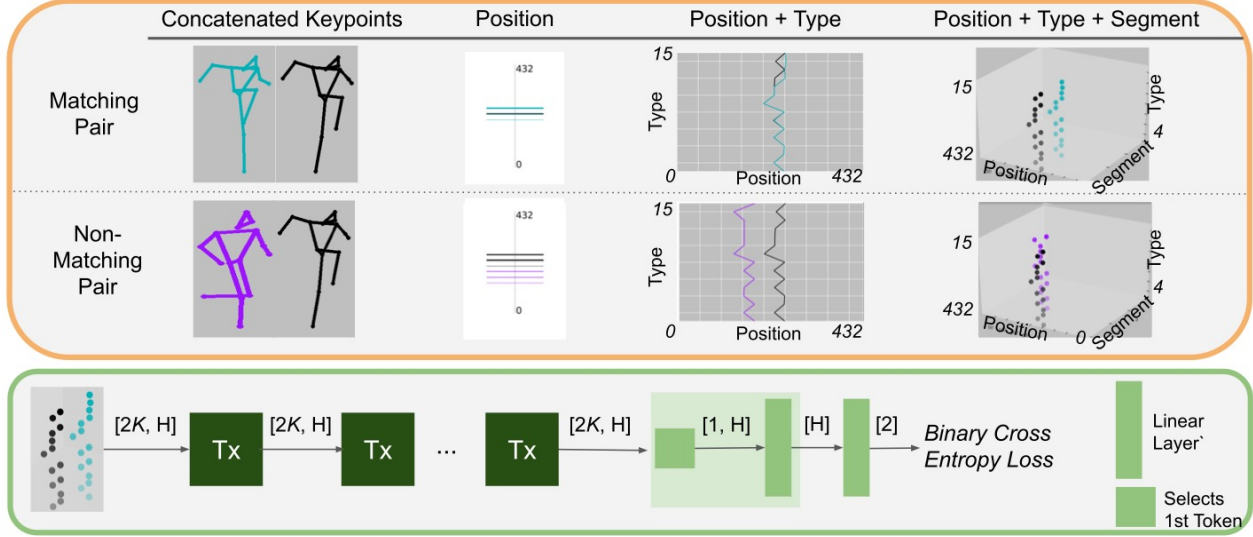
Figure 3. **Orange box:** Visualizations to intuitively explain our tokenization. In the Position column, the matching poses are spatially closer together than the non-matching ones. This is because their spatial locations in the image are similar. The axis limit is 432 because the image has been downsampled to $width * height = 432$. In the following column, the matching contours are similar, since the poses are in similar orientations. The Segment axis in the last column represents the temporal distance of the pair. **Green box:** A series of transformers (Tx) compute self-attention, extracting the temporal relationship between the pair. Binary classification follows.

$p^{t-\delta}$, i.e. the *premise*, and a pose in timestep p^t, i.e. the *hypothesis*, are the same person. To solve this problem, instead of using visual feature based similarity that incurs large computational cost, we use the set of human keypoints, $\mathcal{K}$, detected by our pose estimator. It is computationally efficient to use these as there are a limited number of them (in our case $|\mathcal{K}| = 15$), and they are not affected by unexpected visual variations such as lighting changes in the tracking step. In addition, as we show in the next section, keypoints are amenable to tokenization. Thus, during the tracking stage, we use only the keypoints estimated by the detector as our pose representation.

Tokenizing Pose Pairs The goal of tokenization is to transform pose information into a representation that facilitates learning spatio-temporal human pose relationships. To achieve this goal, for each pose token, we need to provide (i) the spatial location of each keypoint in the scene to allow the network to spatially correlate keypoints across frames, (ii) type information of each keypoint (i.e. head, shoulder etc.) to learn spatial joint relationships in each human pose, and finally (iii) the temporal location index for each keypoint within a temporal window δ, to learn temporal keypoint transitions. Hence, we use three different types of tokens for each keypoint as shown in Figure 3. There are 2 poses, and thus $2|\mathcal{K}|$ tokens of each type. Each token is linearly projected to an embedding, $E \in \mathbb{R}^{2|\mathcal{K}|, H}$ where H is the transformer hidden size. Embeddings are a learned lookup table. We now describe the individual tokens in de-

tail:

Position Token: The absolute spatial location of each keypoint is the *Position* token, ρ, and its values fall in the range $[1, w^{\mathcal{F}}h^{\mathcal{F}}]$. In practice, the absolute spatial location of a downsampled version of the original frame is used. This not only improves the efficiency of our method, but also makes it more accurate, as is discussed in 5.2. We give a general expression for the Position tokens of poses p^t and $p^{t-\delta}$, where $\rho_j^{p^t}$ corresponds to the Position token of the jth keypoint of p^t:

$$\{\rho_1^{p^t}, \rho_2^{p^t}, \ \dots \ \rho_{|\mathcal{K}|}^{p^t}, \ \rho_1^{p^{t-\delta}}, \ \rho_2^{p^{t-\delta}}, \ \dots \ \rho_{|\mathcal{K}|}^{p^{t-\delta}}\} \quad (1)$$

Type Token: The *Type* token corresponds to the unique type of the keypoint: e.g. the head, left shoulder, right ankle, etc... The *Type* keypoints fall in the range $[1, |\mathcal{K}|]$. These add information about the orientation of the pose and are crucial for achieving high accuracy at low resolution, when keypoints have similar spatial locations. A general expression for the *Type* tokens of poses p^t and $p^{t-\delta}$ is below, where j^{p^t} corresponds to the *Type* token of the jth keypoint of p^t:

$$\{1^{p^t}, 2^{p^t}, \ \dots \ |\mathcal{K}|^{p^t}, \ 1^{p^{t-\delta}}, \ 2^{p^{t-\delta}}, \ \dots \ |\mathcal{K}|^{p^{t-\delta}}\} \quad (2)$$

Segment Token: The *Segment* token indicates the number of timesteps the pose is from the current one. The segment token is in range $[1, \delta]$, where δ is a chosen constant. (We set δ to be 4.) This also allows our method to adapt

to irregular frame rates. Or, if a person is not detected in a frame, we can look back two timesteps, conditioning our model on temporal token value of 2 instead of 1.

$$\{1^{p^t}, 1^{p^t}, \; \dots \; 1^{p^t}, \; \delta^{p^{t-\delta}}, \; \delta^{p^{t-\delta}}, \; \dots \; \delta^{p^{t-\delta}}\} \quad (3)$$

After each token is embedded, we sum the embeddings, $E_{sum} = E_{Position} + E_{Type} + E_{Segment}$, to combine the information from each class of token. This is fed to our Transformer Matching Network.

Transformer Matching Network: The goal of our network is to learn motion cues indicative of whether a pose pair matches. The self-attention mechanism of transformers allows us to accomplish this by learning which temporal relationships between the keypoints are representative of a match. Transformers compute scaled dot-product attention over a set of Queries (Q), Keys (K), and Values(V) each of which is a linear projection of the input $E_{sum} \in \mathbb{R}^{2|\mathcal{K}|, H}$. We compute the softmax attention with respect to every keypoint embedding in the pair, with the input to the softmax operation being of dimensions $[2|K|, 2|K|]$. In fact, we can generate heatmaps from the attention distribution over the pair's keypoints, as displayed in 5.3. In practice, we use multi-headed attention, which leads to the heads specializing, also visualized.

Additionally, we use an attention mask to account for keypoints which are not visible due to occlusion. This attention mask is implemented exactly as the attention mask in [49], resulting in no attention being paid to the keypoints which are not visible due to occlusion. The attention equation is as follows, and we detail each operation in a single transformer in Table 5 of the Supplement:

$$\text{Attention}(Q, K, V) = \text{softmax}(\tfrac{QK^T}{\sqrt{d_k}})V \quad (4)$$

After computing self-attention through a series of stacked transformers, similar to BERT, we feed this representation to a Pooler, which "pools" the input, by selecting the first token in the sequence and then inputting that token into a learned linear projection. This is fed to another linear layer, functioning as a binary classifier, which outputs the likelihood two given poses match. We govern training with a binary cross entropy loss providing our network only with the supervision of whether the pose pair is a match. See Figure 3 for more details.

3.3. Improved Multi-Frame Pose Estimation

We now describe how we improve keypoint estimation. Top-down methods suffer from two primary classes of errors from the object detector: 1. Missed bounding boxes 2. Imperfect bounding boxes. We use the box detections from adjacent timesteps in addition to the one in the current

timestep to make pose predictions, thereby combating these issues. This is based on the intuition that the spatial location of each person does not change dramatically from frame to frame when the frame rate is relatively high, typical in most modern datasets and cameras. Thus, pasting a bounding box for the ith person in frame, $\mathcal{F}^{t-1}$, $p^{t-1,i}$, in its same spatial location in frame $\mathcal{F}^t$ is a good approximation of the true bounding box for person $p^{t,i}$. Bounding boxes are enlarged by a small factor to account for changes in spatial location from frame to frame. Previous approaches, such as [54], use standard non-maximal suppression (NMS) to choose which of these boxes to input into the estimator. Though this addresses the 1st issue of missed boxes, it does not fully address the second issue. NMS relies on the confidence score of the boxes. We make pose predictions for the box in the current frame and temporally adjacent boxes. Then we use object-keypoint similarity (OKS) to determine which of the poses should be kept. This is more accurate than using NMS because we use the confidence scores of the keypoints, not the bounding boxes. The steps of TOKS are enumerated below:

Algorithm 1 Temporal OKS

Input: $p^{t-1}, p^t, \mathcal{F}^t$
1. Retrieve bounding box, B, enclosing p^{t-1}, and dilate by a factor, α
2. Estimate a new pose, p'^t, in $\mathcal{F}^t$ from B
3. Use OKS to determine which pose to keep, $p^* = OKS(p'^t, p^t)$
Output: p^*

---

4. Experiments

4.1. The PoseTrack Dataset

The **PoseTrack 2017** training, validation, and test sets consist of 250, 50, and 208 videos, respectively. Annotations for the test set are held out. We evaluate on the PoseTrack 17 Test set because the PoseTrack 18 Test set has yet to be released. We use the official evaluation server on the test set, which can be submitted to up to 4 times. [4, 1] We conduct the rest of comparisons on the **PoseTrack ECCV 2018 Challenge Validation Set**, a superset of PoseTrack 17 with 550 training, 74 validation, and 375 test videos [2].

Metrics Per-joint Average Precision (AP) is used to evaluate keypoint estimation based on the formulation in [6]. Multi-Object Tracking Accuracy (MOTA [7], [33]) scores tracking. It penalizes False Negatives (FN), False Positives (FP), and ID Switches (IDSW) under the following formulation for each keypoint k^i, where t is the current timestep. Our final MOTA is the average of all keypoints $k^i \in \mathcal{K}$:

$$1 - \frac{\sum_t (FN_t^i + FP_t^i + IDSW_t^i)}{\sum_t GT_t^i}$$

Tracking Method	Detection Method	AP ↑ Total	% IDSW ↓								MOTA ↑ Total
			Head	Shou	Elb	Wri	Hip	Knee	Ankl	Total	
Pose Entailment			**0.7**	**0.7**	**0.6**	**0.6**	**0.6**	**0.7**	**0.7**	**0.7**	**99.3**
GCN	GT Boxes, GT Keypoints	100	1.4	1.4	1.4	1.5	1.4	1.6	1.6	1.5	98.5
Optical Flow			1.1	1.2	1.2	1.2	1.2	1.3	1.4	1.2	98.7
Pose Entailment			**0.9**	**0.9**	**0.8**	**0.8**	**0.7**	**0.8**	**0.8**	**0.8**	**72.2**
GCN	GT Boxes, Predicted Keypoints	86.7	1.6	1.6	1.6	1.6	1.3	1.5	1.4	1.5	71.6
Optical Flow			1.2	1.2	1.2	1.1	1.0	1.1	1.1	1.1	71.8
Pose Entailment			**0.9**	**1.0**	**0.9**	**0.8**	**0.7**	**0.8**	**0.8**	**0.8**	**66.6**
GCN	Predicted Boxes, Predicted Keypoints	81.6	1.7	1.7	1.7	1.7	1.4	1.5	1.4	1.6	65.9
Optical Flow			1.3	1.2	1.2	1.2	1.1	1.1	1.1	1.1	66.3

Figure 4. Compares accuracy of tracking methods on the PoseTrack 18 Val set, given the same keypoints. GT stands for Ground Truth, "predicted" means a neural net is used. Lower % IDSW is better, higher MOTA is better. "Total" averages all joint scores.

Our approach assigns track ids and estimates keypoints independently. This is also true of competing methods with MOTA scores closest to ours. In light of this, we use the same keypoint estimations to compare Pose Entailment to competing tracking methods in 4.2. This makes the IDSW the only component of the MOTA metric that changes, and we calculate $\%IDSW^i = \sum_t IDSW_t^i / \sum_t GT_t^i$. In 4.3, we compare our estimation method to others without evaluating tracking. Finally, in 4.4, we compare our entire tracking pipeline to other pipelines.

4.2. Improving Tracking with Pose Entailment

We compare with the optical flow tracking method [54], and the Graph Convolutional Network [35] (GCN) as shown in Figure 4. We do not compare with IoU because, GCN and optical flow [35], [54] have shown to outperform it, nor do we compare to the network from [40] because it is trained in an end-to-end fashion. We follow [54] for Optical Flow and use the pre-trained GCN provided by [35]. IDSW is calculated with three sets of keypoints. Regardless of the keypoint AP, we find that KeyTrack's Pose Entailment maintains a consistent improvement over other methods. We incur approximately half as many IDSW as the GCN and 30% less than Optical Flow.

Our improvement over GCN stems from the fact that it relies only on keypoint spatial locations. By using additional information beyond the spatial location of each keypoint, our model can make better inferences about the temporal relationship of poses. The optical flow CNNs are not specific to pose tracking and require manual tuning. For example, to scale the CNN's raw output, which is normalized from -1 to 1, to pixel flow offsets, a universal constant, given by the author of the original optical flow network (not [54]), must be applied. However, we found that this constant required adjustment. In contrast, our learned method requires no tuning during inference.

4.3. Improving Detection with TOKS

Table 2 shows offers a greater improvement in keypoint detection quality than other methods. In the absence of

Detection Method	AP							
	Head	Shou	Elb	Wri	Hip	Knee	Ankl	Total
GT	90.2	91.4	88.7	83.6	81.4	86.1	83.7	86.7
Det.	68.8	72.8	73.1	68.4	68.0	72.4	69.8	70.4
Det. + Box Prop.	79.3	82.0	80.8	75.6	72.4	76.5	72.4	77.1
Det. + TOKS@0.3	83.6	86.6	84.9	78.9	76.4	80.2	76.2	81.1
Det. + TOKS@0.35 (ours)	**84.1**	**87.2**	**85.3**	**79.2**	**77.1**	80.6	**76.5**	**81.6**
Det. + TOKS@0.5	83.9	87.2	85.2	79.1	77.1	**80.7**	76.4	81.5

Table 2. Per-joint AP when the pose estimator is conditioned on different boxes. GT indicates ground truth boxes are used, and serves as an upper bound for accuracy. Det. indicates a detector was used to estimate boxes. @OKS* is the OKS threshold used.

bounding box improvement, the AP performance is 6.6% lower, highlighting the issue of False Negatives. The further improvement from TOKS emphasizes the usefulness of estimating every pose. By using NMS, bounding box propagation methods miss the opportunity to use the confidence scores of the keypoints, which lead to better pose selection.

4.4. Tracking Pipeline Comparison to the SOTA

Now that we have analyzed the benefits of Pose Entailment and TOKS, we put them together and compare to other approaches. Figure 5 shows that we achieve the highest MOTA score. We improve over the original HRNet paper by 3.3 MOTA points on the Test set. [25], nearest our score on the 2018 Validation set, is much further away on the 2017 Test set. Additionally, our FPS is improved over all methods with similar MOTA scores, with many methods being offline due to their use of ensembles. (Frames per second (FPS) is calculated by diving the number of frames in the dataset by the runtime of the approach.) Moreover, our method outperforms all others in terms of AP, showing the benefits of TOKS. AP^T is also reported, which is the AP score after tracking post-processing has been applied. This post-processing is beneficial to the MOTA score, but lowers AP. See section A.3 for more details on this post-processing. As we have the highest AP, but not the highest AP^T it appears the effect of tracking post-processing varies from paper to paper. Only AP^T is given on the test set because each paper is given 4 submissions, so these are used to optimize MOTA, rather than AP.

No.	Method	Extra Data	AP^T	AP	FPS	MOTA
1.	**KeyTrack (ours)**	✗	74.3	**81.6**	1.0	**66.6**
2.	MIPAL [25]	✗	**74.6**	-	-	65.7
3.	LightTrack (offline) [35]	✗	71.2	77.3	E	64.9
4.	LightTrack (online) [35]	✗	72.4	77.2	0.7	64.6
5.	Miracle [58]	✓	-	80.9	E	64.0
6.	OpenSVAI [36]	✗	69.7	76.3	-	62.4
7.	STAF [38]	✓	70.4	-	3	60.9
8.	MDPN [20]	✓	71.7	75.0	E	50.6

No.	Method	Extra Data	AP^T	FPS	MOTA
1.	**KeyTrack (ours)**	✗	74.0	1.0	**61.2**
2.	POINet [40]	✗	72.5	-	58.4
3.	LightTrack [35]	✗	66.7	E	58.0
4.	HRNet [45]	✗	**75.0**	0.2	57.9
5.	FlowTrack [54]	✗	74.6	0.2	57.8
6.	MIPAL [25]	✗	68.8	-	54.5
7.	STAF [38]	✓	70.3	2	53.8
8.	JointFlow [15]	✗	63.6	0.2	53.1

Figure 5. Top scores on the PoseTrack leaderboards. E indicates an ensemble of detectors is used, and results in the method being offline. A check indicates external training data is used beyond COCO and PoseTrack. A "-" indicates the information has not been made publicly available. FPS calculations for JointFlow and FlowTrack are taken from [59]. HRNet FPS is approximated from FlowTrack since the methods are very similar. The AP column has the best AP score. AP^T is the AP score after tracking post-processing.
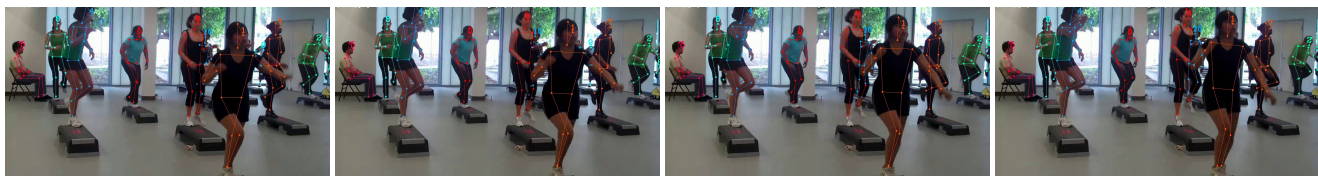

Figure 6. Qualitative results of KeyTrackon the PoseTrack PoseTrack 17 Test Set. Additional qualitative results are in the supplement.

Efficiency: Our tracking approach is efficient, not reliant on optical flow or RGB data. When processing an image at our optimal resolution, 24x18, we reduce the GFLOPS required by optical flow, which processes images at full size, from 52.7 to 0.1. [35]'s GCN does not capture higher-order interactions over keypoints and can be more efficient than our network with local convolutions. However, this translates to a ∼1ms improvement in GPU runtime. In fact, our tracking pipeline demonstrates a 30% improvement in end-to-end runtime over [35], shown in 4.4. We have the fastest FPS of Top-down approaches. Also, we do not rely on optical flow to improve bounding box propagation as [54, 45] do, instead we use TOKS. This contributes to our 5x FPS improvement over [54, 45]. Further details on the parameters and FLOPS of the GCN, Optical Flow Network, and our Transformer Matching Network are in Table 6 of the Supplement.

5. Analysis

5.1. Tracking Pipeline

Varying Tokenization Schemes and Transformer Hyper-parameters We examine the benefits of each embedding. As evident in Table 3, *Segment* embeddings are crucial because they enable the network to distinguish between the Poses being matched. *Token* embeddings give the network information about the orientation of a pose and help it interpret keypoints which are in close spatial proximity; i.e. keypoints that have the same or similar position embedding. We also train a model that uses the relative

Abs. Position	Type	Segment	Rel. Position	Match % Accuracy
✓	✓	✗	✗	72.6
✓	✗	✓	✗	90.0
✓	✓	✓	✗	**93.2 (ours)**
✗	✓	✓	✓	91.3
✓	✓	✓	✓	92.0

Table 3. Match accuracies for various embedding schemes.

keypoint distance from the pose center rather than the absolute distance of the keypoint in the entire image. We find that match accuracy deteriorates with this embedding. This is likely because many people perform the same activity, such as running, in the PoseTrack dataset, leading to them having nearly identical relative pose positions. We vary the number of transformer blocks, the hidden size in the transformer block, and number of heads in Table 7. Decreasing the number of transformer blocks, the hidden size, and attention heads hurts performance.

Number of Timesteps and Other Factors We find that reducing the number of timesteps adversely effects the MOTA score. It drops up to 0.3 points when using only a single timestep because we are less robust to detection errors. Also, in replacement of our greedy algorithm, we experimented with the Hungarian algorithm used in [19]. This algorithm is effective with ground truth information, but is not accurate when using detected poses.

Num Tx	Hidden Size	Int. Size	Num Heads	Parameters (M)	% IDSW
2	128	512	4	0.40	1.0
4	128	512	4	0.43	**0.8**
6	128	512	4	1.26	1.1
4	64	256	4	0.23	0.9
4	128	512	4	0.43	**0.8**
4	256	1024	4	3.31	1.1
4	128	128	4	0.43	**0.8**
4	128	512	4	0.86	0.8
4	128	128	2	0.43	0.9
4	128	128	4	0.43	**0.8**
4	128	128	6	0.43	0.8

Figure 7. **Left:** Transformer network hyper-parameters are varied. **Right:** A plot of IDSW rate vs. image resolution. The table on the left shows the input to each method, the conv+visual input is blurry because images are downsampled.
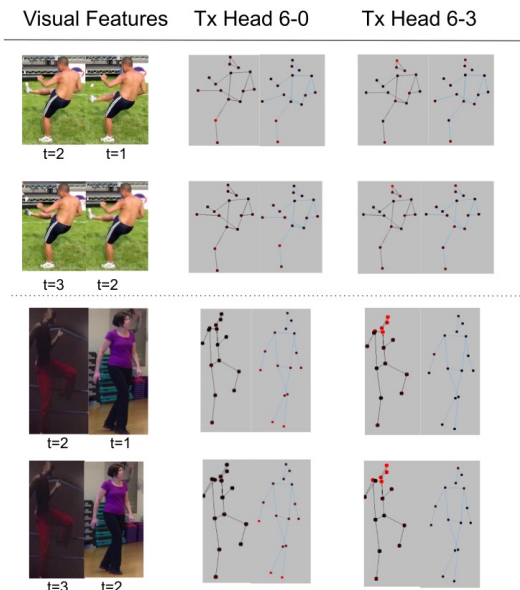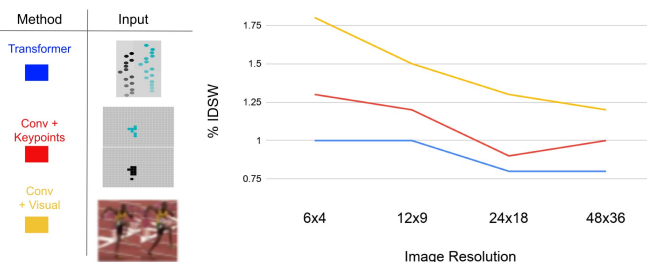

Figure 8. Attention heatmaps from two of our network's attention heads are shown. These are the 0th, and 3rd heads from our final transformer. The two pairs above the dotted line are a matching pair, while the pair below the dotted line are not (and are also from separate videos). t is the frame timestep.

5.2. Comparing Self-Attention to Convolutions

We compare transformers and CNNs by replacing our Transformer Matching Network with two convolution-based methods. One takes visual features from bounding box pose pairs as input while the other takes only key-points as input, where each unique keypoint is colored via a linear interpolation, a visual version of our *Type* tokens. Both approaches use identical CNNs, sharing an architecture inspired by VGG [43], and have approximately 4x more parameters than our transformer-based model because this was required for stable training. See A.4 of the Supplement for details.

Transformers outperform CNNs for the tracking task, as shown in Figure 7. However, we find two areas where CNNs can be competitive. First, at higher resolutions, transformers often need a large number of parameters to match CNN's performance. In NLP, when using large vocabularies, a similar behavior is observed where transformers need multiple layers to achieve good performance. Second, we also find that convolutions optimize more quickly than the transformers, reaching their lowest number of ID Switches within the first 2 epochs of training. Intuitively, CNNs are more easily able to take advantage of spatial proximity. The transformers receive spatial information via the position embeddings, which are 1D linear projections of 2D locations. This can be improved by using positional embedding schemes that better preserve spatial information [18].

In summary, CNNs are accurate at high resolutions given its useful properties such as translation invariance and location invariance. However, there is an extra computational cost of using them. The extra information, beyond the spatial location of keypoints, included in our keypoint embeddings, coupled with the transformer's ability to model higher-order interactions allows it to function surprisingly well at very low resolutions. Thus, the advantage of CNNs is diminished and our transformer-based network outperforms them in the low resolution case.

5.3. Visualizing Attention Heatmaps

We visualize our network's attention heatmaps in Fig. 8. When our network classifies a pair as non-matching, its attention is heavily placed on one of the poses over the other. Also, we find it interesting that one of our attention heads primarily places its attention on keypoints near the person's head. This specialization suggests different attention heads are attuned to specific keypoint motion cues.

6. Conclusion

In summary, we present an efficient Multi-person Pose Tracking method. Our proposed Pose Entailment method achieves SOTA performance on PoseTrack datasets without using RGB information in the tracking step. KeyTrack also benefits from improved keypoint estimates using TOKS, which outperforms bounding box propagation methods. Finally, we demonstrate how to tokenize and embed human pose information in the transformer architecture that has applications to tasks such as pose-based action recognition.

References

[1] Posetrack leaderboard, 2017 test set, 2017. 2, 5

[2] Posetrack challenge - eccv 2018, 2018. 5

[3] Abdulrahman Alarifi, AbdulMalik Al-Salman, Mansour Al-saleh, Ahmad Alnafessah, Suheer Al-Hadhrami, Mai A Al-Ammar, and Hend S Al-Khalifa. Ultra wideband indoor positioning technologies: Analysis and recent advances. *Sensors*, 16(5):707, 2016. 2

[4] Mykhaylo Andriluka, Umar Iqbal, Eldar Insafutdinov, Leonid Pishchulin, Anton Milan, Juergen Gall, and Bernt Schiele. Posetrack: A benchmark for human pose estimation and tracking. In *CVPR*, 2018. 2, 5

[5] Mykhaylo Andriluka, Stefan Roth, and Bernt Schiele. Pictorial structures revisited: People detection and articulated pose estimation. In *2009 IEEE conference on computer vision and pattern recognition*, pages 1014–1021. IEEE, 2009. 2

[6] Mykhaylo Andriluka1, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation - mpii human pose dataset. In *CVPR*, 2014. 5

[7] Keni Bernardin and Rainer Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *Journal on Image and Video Processing*, 2008:1, 2008. 5

[8] Gedas Bertasius, Christoph Feichtenhofer, Du Tran, Jianbo Shi, and Lorenzo Torresani. Learning temporal pose estimation from sparsely-labeled videos. *arXiv preprint arXiv:1906.04016*, 2019. 2

[9] Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*, 2015. 3

[10] Yilun Chen, Zhicheng Wang, Yuxiang Peng, Zhiqiang Zhang, Gang Yu, and Jian Sun. Cascaded pyramid network for multi-person pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7103–7112, 2018. 2

[11] Bo Dai, Yuqi Zhang, and Dahua Lin. Detecting visual relationships with deep relational networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 3

[12] Matthias Dantone, Juergen Gall, Christian Leistner, and Luc Van Gool. Human pose estimation using body parts dependent joint regressors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3041–3048, 2013. 2

[13] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. 3

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 2

[15] Andreas Doering, Umar Iqbal, and Juergen Gall. Joint flow: Temporal flow fields for multi person tracking, 2018. 7

[16] Pedro F Felzenszwalb and Daniel P Huttenlocher. Pictorial structures for object recognition. *International journal of computer vision*, 61(1):55–79, 2005. 2

[17] Mihai Fieraru, Anna Khoreva, Leonid Pishchulin, and Bernt Schiele. Learning to refine human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 205–214, 2018. 2

[18] Rohit Girdhar, Joao Carreira, Carl Doersch, and Andrew Zisserman. Video action transformer network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 244–253, 2019. 3, 8

[19] Rohit Girdhar, Georgia Gkioxari, Lorenzo Torresani, Manohar Paluri, and Du Tran. Detect-and-track: Efficient pose estimation in videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 350–359, 2018. 2, 3, 7

[20] Hengkai Guo, Tang Tang, Guozhong Luo, Riwei Chen, Yongchen Lu, and Linfu Wen. Multi-domain pose network for multi-person pose estimation and tracking. *Computer Vision – ECCV 2018 Workshops*, page 209–216, 2019. 2, 3, 7

[21] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 2

[22] Ronghang Hu, Marcus Rohrbach, Jacob Andreas, Trevor Darrell, and Kate Saenko. Modeling relationships in referential expressions with compositional modular networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 3

[23] Hao Huang, Luowei Zhou, Wei Zhang, Jason J Corso, and Chenliang Xu. Dynamic graph modules for modeling object-object interactions in activity recognition: Supplementary material. *BMVC*, 2019. 3

[24] Shaoli Huang, Mingming Gong, and Dacheng Tao. A coarse-fine network for keypoint localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3028–3037, 2017. 2

[25] Jihye Hwang, Jieun Lee, Sungheon Park, and Nojun Kwak. Pose estimator and tracker using temporal flow maps for limbs, 2019. 2, 6, 7

[26] Eldar Insafutdinov, Mykhaylo Andriluka, Leonid Pishchulin, Siyu Tang, Evgeny Levinkov, Bjoern Andres, and Bernt Schiele. Arttrack: Articulated multi-person tracking in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6457–6465, 2017. 2, 3

[27] Umar Iqbal, Anton Milan, and Juergen Gall. Posetrack: Joint multi-person pose estimation and tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2011–2020, 2017. 2, 3

[28] Sheng Jin, Wentao Liu, Wanli Ouyang, and Chen Qian. Multi-person articulated tracking with spatial and temporal embeddings. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5664–5673, 2019. 2

[29] Sheng Jin, Xujie Ma, Zhipeng Han, Yue Wu, Wei Yang, Wentao Liu, Chen Qian, and Wanli Ouyang. Towards multi-

person pose tracking: Bottom-up and top-down methods. In *ICCV PoseTrack Workshop*, volume 2, page 7, 2017. 3

[30] Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*, 2019. 3

[31] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *arXiv preprint arXiv:1908.02265*, 2019. 3

[32] Chih-Yao Ma, Asim Kadav, Iain Melvin, Zsolt Kira, Ghassan AlRegib, and Hans Peter Graf. Attend and interact: Higher-order object interactions for video understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6790–6800, 2018. 3

[33] Anton Milan, Laura Leal-Taixé, Ian D. Reid, Stefan Roth, and Konrad Schindler. MOT16: A benchmark for multi-object tracking. *CoRR*, abs/1603.00831, 2016. 5

[34] Gyeongsik Moon, Ju Yong Chang, and Kyoung Mu Lee. Posefix: Model-agnostic general human pose refinement network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7773–7781, 2019. 2

[35] Guanghan Ning and Heng Huang. Lighttrack: A generic framework for online top-down human pose tracking. *arXiv preprint arXiv:1905.02822*, 2019. 1, 2, 6, 7

[36] Guanghan Ning, Ping Liu, Xiaochuan Fan, and Chi Zhang. A top-down approach to articulated human pose estimation and tracking. *Computer Vision – ECCV 2018 Workshops*, page 227–234, 2019. 7

[37] George Papandreou, Tyler Zhu, Nori Kanazawa, Alexander Toshev, Jonathan Tompson, Chris Bregler, and Kevin Murphy. Towards accurate multi-person pose estimation in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4903–4911, 2017. 2

[38] Yaadhav Raaj, Haroon Idrees, Gines Hidalgo, and Yaser Sheikh. Efficient online multi-person 2d pose tracking with recurrent spatio-temporal affinity fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4620–4628, 2019. 2, 7

[39] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jonathon Shlens. Stand-alone self-attention in vision models. *arXiv preprint arXiv:1906.05909*, 2019. 3

[40] Weijian Ruan, Wu Liu, Qian Bao, Jun Chen, Yuhao Cheng, and Tao Mei. Poinet: Pose-guided ovonic insight network for multi-person pose tracking. In *Proceedings of the 27th ACM International Conference on Multimedia*, MM '19, pages 284–292, New York, NY, USA, 2019. ACM. 2, 6, 7

[41] Adam Santoro, David Raposo, David GT Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Timothy Lillicrap. A simple neural network module for relational reasoning. In *Advances in Neural Information Processing Systems*, 2017. 3

[42] Leonid Sigal and Michael J Black. Measure locally, reason globally: Occlusion-sensitive articulated pose estimation. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2041–2048. IEEE, 2006. 2

[43] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2014. 8

[44] Chen Sun, Fabien Baradel, Kevin Murphy, and Cordelia Schmid. Contrastive bidirectional transformer for temporal representation learning. *arXiv preprint arXiv:1906.05743*, 2019. 3

[45] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. *CoRR*, abs/1902.09212, 2019. 1, 2, 3, 7

[46] Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*, 2019. 3

[47] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. Efficient object localization using convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 648–656, 2015. 1, 2

[48] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1653–1660, 2014. 1, 2

[49] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017. 1, 5

[50] Xiaolong Wang, Allan Jabri, and Alexei A Efros. Learning correspondence from the cycle-consistency of time. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2566–2576, 2019. 2

[51] Yang Wang and Greg Mori. Multiple tree models for occlusion and spatial constraints in human pose estimation. In *European Conference on Computer Vision*, pages 710–724. Springer, 2008. 2

[52] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4724–4732, 2016. 1, 2

[53] Fangting Xia, Peng Wang, Xianjie Chen, and Alan L Yuille. Joint multi-person pose estimation and semantic part segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6769–6778, 2017. 2

[54] Bin Xiao, Haiping Wu, and Yichen Wei. Simple baselines for human pose estimation and tracking. In *European Conference on Computer Vision (ECCV)*, 2018. 5, 6, 7

[55] Yuliang Xiu, Jiefeng Li, Haoyu Wang, Yinghong Fang, and Cewu Lu. Pose flow: Efficient online pose tracking. *arXiv preprint arXiv:1802.00977*, 2018. 3

[56] Jiarui Xu, Yue Cao, Zheng Zhang, and Han Hu. Spatial-temporal relation networks for multi-object tracking. *arXiv preprint arXiv:1904.11489*, 2019. 3

[57] Wei Yang, Shuang Li, Wanli Ouyang, Hongsheng Li, and Xiaogang Wang. Learning feature pyramids for human pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1281–1290, 2017. 1, 2

[58] Dongdong Yu, Kai Su, Jia Sun, and Changhu Wang. Multi-person pose estimation for pose tracking with enhanced cascaded pyramid network. In *European Conference on Computer Vision*, pages 221–226. Springer, 2018. 7

[59] Jiabin Zhang, Zheng Zhu, Wei Zou, Peng Li, Yanwei Li, Hu Su, and Guan Huang. Fastpose: Towards real-time pose estimation and tracking via scale-normalized multi-task networks, 2019. 7

[60] Luowei Zhou, Hamid Palangi, Lei Zhang, Houdong Hu, Jason J Corso, and Jianfeng Gao. Unified vision-language pre-training for image captioning and vqa. *arXiv preprint arXiv:1909.11059*, 2019. 3

[61] Luowei Zhou, Yingbo Zhou, Jason J Corso, Richard Socher, and Caiming Xiong. End-to-end dense video captioning with masked transformer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8739–8748, 2018. 3