Pose and Joint-Aware Action Recognition - Supplementary Material

Anshul Shah[1] Shlok Mishra[2] Ankan Bansal[2] Jun-Cheng Chen[3]
Rama Chellappa[1] Abhinav Shrivastava[2]

[1]Johns Hopkins University [2]University of Maryland, College Park
[3]Research Center for Information Technology Innovation, Academia Sinica
{ashah95,rchella4}@jhu.edu {shlokm,ankan,abhinav2}@umd.edu pullpull@citi.sinica.edu.tw

1. Supplementary

In this supplementary material, we provide:
- Additional implementation details for the approach.
- Additional details on modality fusion.
- Other variants for Joint-Contrastive loss.
- Analyses and experiments on data augmentations.
- Experiments on clip selection.

2. Additional Implementation Details

We use Adam with a learning rate of 1E-4 for JHMDB and HMDB and 5E-4 for Charades. The learning rate is reduced by 0.1 times after the validation loss plateaus. Use of dropout after convolutional layers, as employed in [1] was beneficial for the Charades dataset. None of our experiments use the background joint in an attempt to force the network to extract richer per-joint motion information. Following [1], we additionally use flip augmentation for all experiments. Experiments on HMDB and Charades use a batch size of 128 while experiments on the much smaller JHMDB used a batch size of 16. Further, for JHMDB we activate the contrastive loss after 50 epochs. The dataset specific hyper-parameters that we used are:

J-HMDB:
- c_{dim} : 128
- λ: 0.05
- Epochs : 100
- Training and evaluation on heatmaps of size 64×86
- $\beta = 6, \gamma = 3$
- $\tau = 0.3$

HMDB:
- c_{dim} : 64
- λ: 0.5
- Epochs : 200
- Training on random crop of size 64×86 and evaluation on center crop of size 64×86
- $\beta = 4, \gamma = 0$
- $\tau = 0.1$

Charades:
- c_{dim} : 32
- λ: 0.5
- Epochs : 150
- Training on random crop of size 64×64 and evaluation on center crop of size 64×64
- $\beta = 8, \gamma = 4$
- $\tau = 0.05$

Experiments on the Mimetics dataset For experiments on Mimetics, we first train a model on Kinetics50 dataset which has the 50 classes common with Mimetics. Then, we evaluate this model on the Mimetics dataset. We used a batch size of 16, $c_{dim} = 256$ and trained the model for 100 epochs,. We used Dropout with this model. SGD with momentum was used for training this model with lr of 0.01, momentum of 0.9. We found $\lambda = 0$ to work well for this dataset. In addition, we found the use of Gumbel-Max trick [6, 3] in the weight-selector to be beneficial. Patience of 2 was used for the scheduler. $\beta, \gamma = (3, 0)$ was used to train the model. We generate heatmaps by parsing the dataset provided by [8] and adding Gaussian blobs around the keypoints. Our model gave 44.4% on Kinetics50 validation set.

Experiments on AVA dataset Since AVA is annotated with multi-person multi-label actions, the bounding box information is used to crop per-person pose information across the clip. These are then used to obtain the pose encoding. For a fair comparison, we used the bounding boxes provided by SlowFast [2]. To account for multi-person context information, we also append the pose encoding of other people in the clip as a separate channel. We use a batch size of 64 and train the model with a learning rate of 5E-4. We used $\beta, \gamma = (3, 0)$. Hyperparameters for joint-contrastive loss were set to $\tau, \lambda = 0.1, 0.5$.

Additional baseline details Since an official implementation of [1] is not publicly available, we reproduce the results using our own implementation. As mentioned in the main paper, we make use of the same aggregation function for a

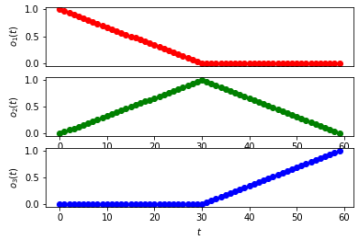

Figure 1: Channel-time encoding $o[t]$ used to encode the pose heatmaps. We use the same $o[t]$ as [1] with three channels. This example is for a video of sixty frames.

fair comparison. For completeness, we include the aggregation function used in Fig. 1.

Normalization: Normalization is essential before using a representation as an input to a neural network. The max-over-channel strategy chosen in [1] normalizes the input to lie between 0 and 1. We posit that it is also important to encode the time spent by each joint at a location. The max-over-channel strategy assigns 1.0 as the maximum value in a channel irrespective of the actual duration that a joint spent at a particular spatial location which renders different representations incomparable. To solve this, we normalize by dividing the aggregated representation p_j by the maximum possible value that a joint can accumulate at that location. Note that, in [1] the authors mention that they tried this approach, but this did not improve their performance. However, we obtain considerable improvement upon using this normalization perhaps due to better modeling.

Architecture Details Table 1 shows the architecture of our motion extractor module which is shared among all joints to give r_j. An additional 1×1 convolution is used to generate c_j. Table 2 shows the architecture of our joint selector module which generates w_j using r_j. The representation c_j is weighed using the weights w_j and passed to the classification module (Table 3) to generate the final classification scores. ReLU activation is used after all layers except for the last layer of joint selector which uses sigmoid. The projection layers used with contrastive loss are implemented as two MLP layers which take spatially pooled per-joint features. The output from the MLP is normalized to lie on a unit hypersphere.

3. Modality Fusion

Implementation Details As discussed in the main paper, we propose to use a very simple learnt fusion scheme to combine the different modalities. For single label classification tasks, we learn a single scalar weighing parameter for each of the M modalities to be fused and for a multi-label task (like Charades) with C classes, we learn $M \times C$ parameters. We first use the pretrained RGB+Flow and pose models to extract per-modality logits for each clip. We then learn the parameters using the train set. During inference, the modalities are fused using the learnt logits. Specifically for a single label classification task,

$$l_{combined} = \alpha_{RGB} l_{RGB} + \alpha_{flow} l_{flow} + \alpha_{pose} l_{pose} \quad (1)$$

Where $l_{RGB}, l_{flow}, l_{pose} \in \mathbb{R}^C$ are logits extracted from the pretrained network and $\alpha_{RGB}, \alpha_{flow}, \alpha_{pose}$ are optimized to minimize classification loss w.r.t $l_{combined}$. For JHMDB and HMDB experiments, we $L2$ normalize the extracted logits. We train all models for 100 epochs. The Adam optimizer was used for all experiments and we chose

Table 1: Architecture details of the Motion Extractor. This module extracts motion information from each joint separately. In our proposed approach JMRN, parameters of this module are shared amongst all joints except the initial BatchNorm layer.

Layer	Kernel/Stride	Output of Layer
BatchNorm	-	$3 \times 64 \times 86$
Convolution	3/2	$128 \times 32 \times 43$
Convolution	3/1	$128 \times 32 \times 43$
Convolution	3/2	$256 \times 16 \times 22$
Convolution	3/1	$256 \times 16 \times 22$

Table 2: Architecture of the joint-selector module. This module uses the motion representation from all joints to reweight the joints most useful for the task.

Layer	Kernel/Stride	Output Size
Convolution	1/1	$256 \times 16 \times 22$
Average Pool	-	$256 \times 1 \times 1$
FC	3/1	J

Table 3: Architecture of the classification module. The reweighed joints are the input to the module. This module performs the final classification.

Layer	Kernel/Stride	Output Size
Convolution	1/1	$512 \times 16 \times 22$
Convolution	3/2	$512 \times 8 \times 11$
BatchNorm	-	$512 \times 8 \times 11$
Convolution	3/1	$512 \times 8 \times 11$
BatchNorm	-	$512 \times 8 \times 11$
Global Average Pool	-	512
FC	-	C

learning rates of 1e-3, 1e-4, 5e-4 for JHMDB, HMDB, Charades respectively. Since we work with pre-extracted logits and there are a very few parameters to learn, these experiments can be performed very quickly and can be trained on a CPU. While there are a lot of approaches proposed for RGB and flow, we primarily choose models which have a high performance and are publicly available. We expect similar improvement over other models too.

In Figures 2, and 3 we visualize the improvements that we obtain on per-class metrics when we perform fusion with recent state-of-the-art methods. We see considerable improvements on classes where human motion plays a key role and this improvement is also seen in the overall performance. We believe that training schemes which involve back-propagation through the pose and RGB/flow backbones like [4] will lead to further gains and can avoid the drop that we see in a few classes during fusion. We leave this to future work.

4. Other variants for Joint-Contrastive loss

Here we experiment with different variants of our joint-contrastive loss. We first experiment with a variant that does not use label information. Specifically, only the instance and its augmented version are considered as positives. With this approach every other instance in the batch is considered as a negative for loss calculation. Note that this approach is commonly used in the self-supervised learning setting. We refer to this variant as `only-aug`. As discussed in the paper, for multi-label problems we consider an instance as a positive if it shares any label with the anchor. An alternative approach is to weigh the positive examples by the number of labels that they have in common. We call this variant `multi-label-weighted`. We refer to the variant used in the main paper as `proposed`. The results are shown in Table 4. We see that the proposed variant uses the label information effectively during training and gives a better performance compared to the other variants. But, the other variants of joint-contrastive loss still outperform our model trained without any joint-contrastive loss thus showing the efficacy of the proposed approach.

5. Analyses and experiments on data augmentation

Values of β and γ. In Fig. 4 we show the effect of β and γ on the performance of our model on the three datasets. It can be seen that almost all values of β, γ lead to improvements over $(0, 0)$ making it easier to use this method on other datasets without an extensive hyper-parameter search. Further, relatively small values of β, γ are enough to give a considerable performance improvement. To evaluate the contribution of augmentation scheme alone, we do not use contrastive loss in these experiments.

Alternative Data Augmentation Technique. We experimented with a few other data augmentation strategies. We first use a technique that was inspired by CutMix [9]. In the original paper, the authors generate new training examples by mixing patches from different images and changing the ground-truth labels based on the area of patches and the labels. We attempt to use this in our context. Specifically, during training, P joints from a pose-representation are replaced with pose-representation from another video. The ground truth labels are appropriately changed and the network is trained with the modified ground truth. In our experiments, this strategy performed considerably worse than the proposed PAA.

We also experimented with a pose-aware rotation augmentation. Specifically, we rotate the obtained representation about the center by a random amount. This can be seen to be a rotation equivalent to the global translation jitter in which the entire representation is moved by the same amount. Similar to the groupwise translation, we experiment with groupwise rotation too. We found the augmentation to be helpful and it leads to improvements over the baseline, though it performs slightly worse than translation based augmentation that was shown in the main paper. Combining the two augmentation in novel ways could lead to improved performance but we leave that to future work. It is to be noted that, unlike [1], which tried to randomly jitter each joint spatially, our approach of doing pose-aware augmentation retains the structure and hence leads to improvements. As observed in [1], we do not see any gain when we jitter or rotate each joint independently.

PAA with baseline Our data augmentation step is general and can be applied to other models. To show this, we experimented with PAA applied to the baseline PoTion model. The results are shown in Table 5. We see that the augmentation shows consistent improvement on the baseline but the results are still worse than our proposed model. Further, a joint-contrastive loss is not applicable to the baseline model since it does not extract per-joint motion features.

6. Filtering Clips for Action recognition using Pose Information

Pose guided clip selection In this section we explore application of pose trajectories for clip selection. Human trajectory information can contain significant information about the underlying activity. We posit that pose information can give crucial cues to select representative clips in a video. The intuition behind this is that the pose trajectory encodes the movement of all the joints along with redundancies and time spent at each location. Properties of this trajectory, like points of high or low curvature, time spent at a location, etc., can give cues to select clips that are discriminative for the task of action recognition. We next describe our approach for clip selection.

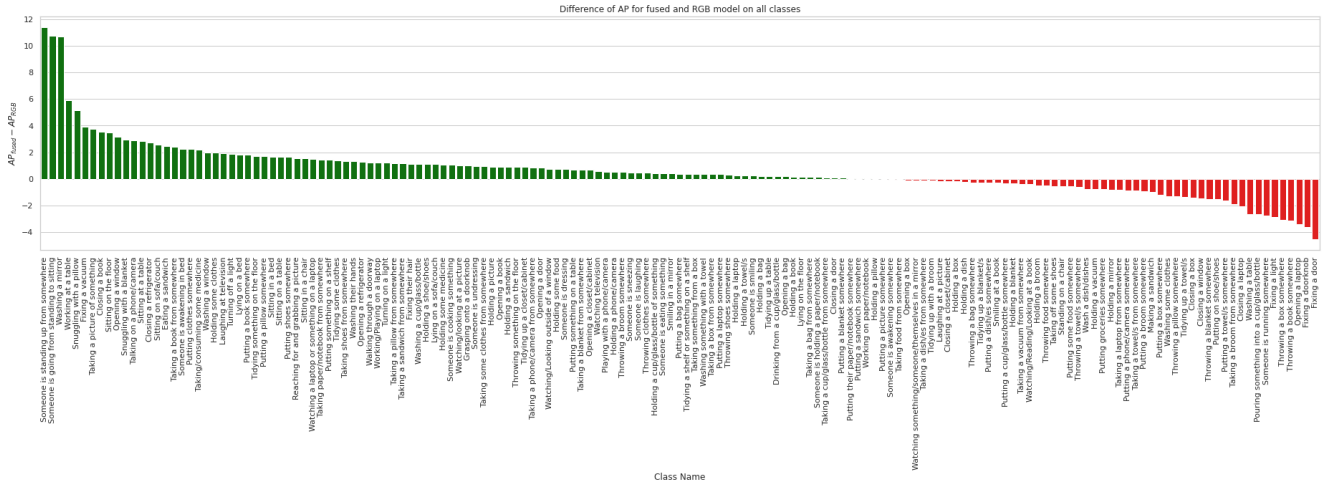

Figure 2: Per-class AP difference on Charades validation set when JMRN is combined with R101-NL-LFB[7]. We see that most classes see an improvement in per-class AP scores justifying the claim that pose has complementary information.

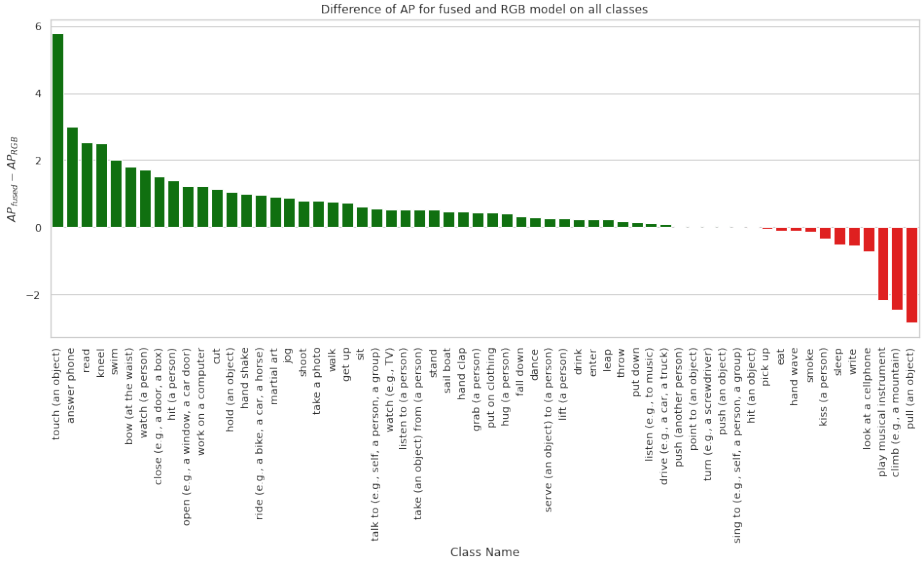

Figure 3: Per-class AP difference on AVA v2.1 validation set when JMRN is combined with SlowFast-R101-NL[2].

Table 4: Experiments with variants of joint-contrastive loss. We experiment with three variants of the joint-contrastive loss to train our model. We see that the `proposed` variant outperforms the alternatives. Further, we see that across all datasets the use of any of the variants improves performance over model trained without the joint-contrastive loss. This shows the effectiveness of the proposed loss in learning good representations.

Model	Joint-Contrastive loss variant	JHMDB-1	HMDB-1	Charades
JMRN	None	69.81	52.02	15.33
JMRN	`only-aug`	69.01	53.13	15.91
JMRN	`multi-label-weighted`	-	-	15.97
JMRN	`proposed`	**71.08**	**54.05**	**16.2**

We assume a trained clip classifier f that takes as input clips $c_i \in \mathbb{R}^{3\times H\times W\times T}$ for $i \in 1, \cdots, N$ from a video with N clips and returns normalized logits z_i that give the probability of occurrence of each class. The baseline model

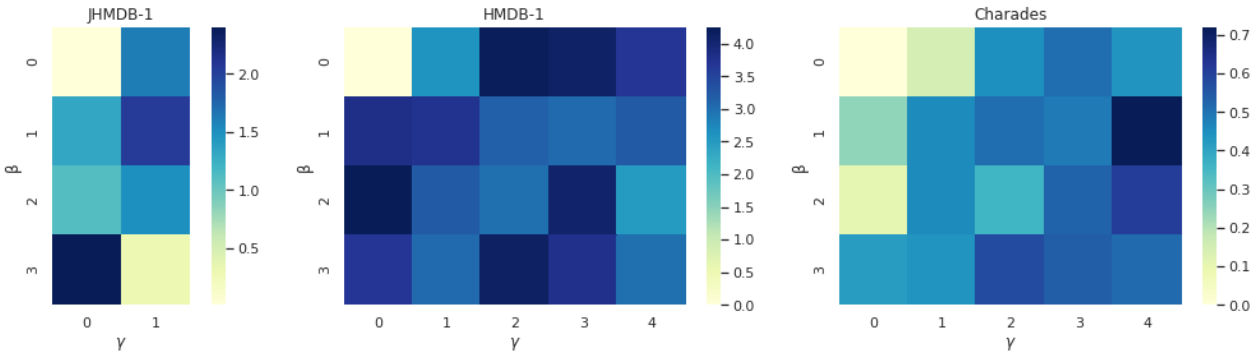

Figure 4: Effect of data augmentation hyperparameters β, γ. We visualize the improvement in accuracy compared to a model which does not use the proposed data augmentation step. We observe that even small values of β, γ lead to a significant improvement which shows the effectiveness of the approach. Further, we notice that the performance is not very sensitive to the hyperparameter values and all values lead to an improvement.

Table 5: PAA with baseline. The proposed augmentation step is not tied to JMRN but can give performance improvements to other methods. The baseline shows consistent improvement when used with PAA. Our proposed model still outperforms Baseline + PAA

Approach	**JHMDB-1**	**HMDB-1**	**Charades**
PoTion	59.44	42.04	13.54
+ PAA	65.92	49.76	15.16
Ours	**71.08**	**54.05**	**16.2**

densely goes over each clip and uses a consensus function $g(z_1, \cdots, z_N)$ to give a single class-wise score for each video. Typically, `max` or `avg` is used as the consensus function.

Inspired by [5], we generate 'oracle' scores using the training data and the pre-trained clip classifier and then train a model to mimic the 'oracle'. For clip selection experiments, we train using pose-representation extracted from clips of sixteen frames. Oracle for the trained clip classifier is first obtained using $\mathcal{O}_i = \text{argtopK}\ (f_{y_i}(c_j))$ which selects the K clips that give the highest confidence about the correct class. These Oracle clips are then used to generate ground truth labels: label$[i, j] = 1$ if $c_j \in \mathcal{O}_i$ and 0 otherwise. Thus instead of selection being guided by RGB data, we use features extracted from JMRN.

Next, we train a saliency ranker which ranks two clips during training time. In each batch, we sample an equal number of Oracle and non-oracle clips. During inference, we rank the clips according to the saliency given by the model for all clips in the video and select the topK.

In Table 6 we show the results of clip selection using our approach on average of 3 splits on HMDB dataset. Using JMRN to extract pose features helps select salient clips and gives improvements over all baselines. We also show the middle frames corresponding to the 5 most salient clips in Fig. 5. The selected clips are plausible for the activity of 'turn' and 'swing baseball' respectively.

Table 6: Clip selection comparison average of 3 splits on HMDB. Our model outperforms the dense clip selector baseline by 0.7% while utilizing only 60% of clips.

Method	Clips used	Accuracy(%)
Dense	24	75.64
Random	14	74.72
Uniform	14	75.11
Empirical	14	74.96
JMRN + PAA + TAN	14	**76.34**

References

[1] Vasileios Choutas, Philippe Weinzaepfel, Jérôme Revaud, and Cordelia Schmid. Potion: Pose motion representation for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[2] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6202–6211, 2019.

[3] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations*, 2017.

[4] Hamid Reza Vaezi Joze, Amirreza Shaban, Michael L Iuzzolino, and Kazuhito Koishida. Mmtm: Multimodal transfer module for cnn fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13289–13299, 2020.

[5] Bruno Korbar, Du Tran, and Lorenzo Torresani. Scsampler: Sampling salient clips from video for efficient action recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

Figure 5: Here we show the middle frame corresponding to 5 most salient clips selected using our approach on the HMDB dataset. The upper row has frames from a video of 'turn' whereas the second row is from a video of 'swing baseball'. The frames are shown in order of their saliency scores. The selected clips are plausible for the underlying action.

[6] Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *5th International Conference on Learning Representations*, 2017.

[7] Chao-Yuan Wu, Christoph Feichtenhofer, Haoqi Fan, Kaiming He, Philipp Krahenbuhl, and Ross Girshick. Long-term feature banks for detailed video understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[8] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

[9] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019.