UNIVERSITY OF
BIRMINGHAM

Analysis of Non-Human Primates Pose Estimation with Transformer

Jiwon Park (2226442)
BSc Computer Science
2023

Supervisor:
Hyungjin Chang

Word Count: **6096**

Abstract

Convolutional Neural Networks (CNN) have been widely employed in human pose estimation tasks. However, recently, transformers with self-attention features have shown outstanding performance when applied to large datasets. Despite this trend, there is a scarcity of approaches addressing Non-Human Primate (NHP) pose estimation using transformers. This project aims to bridge this gap by utilizing a transformer-based model, an underexplored method in NHP pose estimation. This project adapts one of the most recent and effective models with transformer, previously employed in multiple human pose estimation tasks, and fine-tunes it using an NHP dataset. The inherent limitations of transformer models, such as the need for extensive data and high computational costs, are overcome through transfer learning with a pre-trained model. A comparison with a CNN-based model on the same dataset reveals that adopting a transformer-based approach for NHP pose estimation enhances performance. Moreover, ablation study addresses the decision to freeze layers and provides evidence of underfitted tail keypoints. These findings suggest that the self-attention feature of transformers is effective not only for human pose estimation tasks but also for NHP pose estimation. This project has potential to inspire further research in other vision tasks with limited data, demonstrating that transformers can offer viable solutions when combined with transfer learning. The code used for this study can be found at: Gitlab or Github.

Acknowledgement

I want to express my gratitude to everyone who helped me with this project. I want to start by expressing my gratitude to Dr. HyungJin Chang, the project's supervisor. He offered considerate encouragement and criticism to help me improve this work. Next, I would like to thank Inspector Dr. Alexander Krull for his feedback on the development of the contents of project. Finally, I'm grateful to University of Birmingham for providing the research environment and fostering the knowledge required to finish this study.

Contents

1 Introduction

In recent years, the study of non-human primates (NHP) pose estimation has become a critical research area due to its applicability and significance across a diverse range of fields, such as neuroscience, psychology, anthropology, epidemiology, and ecology. Further, the development of automatic pose detection methodologies plays a crucial role in improving NHP welfare and conservation efforts. Nevertheless, NHP pose estimation research presents several challenges, including the homogeneity of body textures and the vast number of potential pose configurations, complicating the identification of instances and joints in images. Moreover, the limited availability of datasets and research on NHP pose estimation has resulted in the field being less developed compared to human pose estimation.

Historically, early approaches to pose estimation were reliant on physical marker-based methods [29], which proved inefficient and difficult to apply, particularly for monkeys. The advent of deep learning, however, has spurred the exploration of various techniques to address pose estimation tasks [18, 19, 20, 31], such as employing convolutional neural networks (CNN) based models for NHP pose estimation [25, 26, 27]. Recently, transformers [1], initially designed for natural language processing tasks, have gained significant attention in the computer vision domain. For instance, Vision Transformers (ViT) [2] have demonstrated the feasibility of using image patches in transformer encoders without CNN. Drawing inspiration from object detection tasks with transformers [33, 34], recent research has reported promising results for multi-person pose estimation using transformers [3, 4, 13]. However, NHP pose estimation faces unique challenges due to the lack of extensive datasets. To overcome this limitation, the transfer learning approach presented in "Transferring Dense Pose to Proximal Animal Classes" [27] offers a viable solution by adapting human pose models to chimpanzee models. In this project, a NHP pose estimation model is introduced using transformers and transfer learning from human models.

Proposed approach of this project adopts the architecture of PETR [13], one of the recent high-performance end-to-end transformer-based multi-person human pose estimation model. The PETR model is fundamentally grounded in the basic transformer architecture, as introduced by [1]. Owing to its relative simplicity in comparison to alternative transformer, such as the Vision Transformer (ViT) [2], which are tailored for visual tasks, the basic transformer demonstrates greater adaptability and suitability for fine-tuning and transfer learning. The architecture of model can be divided into three components: a backbone network, an encoder, and a decoder. ResNet-50 serves as the backbone, extracting multi-scale feature maps from images, which are then fed into a fully connected layer to obtain concatenated feature tokens. Subsequently, the encoder refines the multi-scale visual feature memory. The pose decoder predicts multiple body poses in parallel, employing multiple pose queries and visual feature memory with pose-to-pose attention and feature-to-pose attention. A separate joint decoder refines the poses and outputs the results, following processes similar to the pose decoder. Building on the success of prior work [27], this work utilizes a pre-trained model using the COCO Keypoint 2017 dataset [37] as a starting point for transfer learning. Data augmentation techniques, such as random cropping, flipping, and resizing, are applied to overcome data scarcity with transfer learning. The optimal epoch for fine-tuning the pre-trained model is determined using the Open Monkey Challenge validation set [32].

Training only on the Open Monkey Challenge dataset, the project achieves performance comparable to models trained on larger datasets. The project conducts a comprehensive evaluation of the model and assesses individual keypoint performance by comparing this approach to other CNN-based models on the same dataset. These results demonstrate that employing transformers for NHP pose estimation can be a successful option, overcoming limitations through transfer learning. In summary, the contributions of this project can be divided into three parts:

- Adapt one of the recent high performance transformer-based multi-person human pose estimation model to address the non-human primate (NHP) pose estimation task.
- Utilize transfer learning and data augmentation to overcome the main limitation of transformer-based architecture, which is the requirement for a large dataset.

- Upon examining the performance of Transformer-based models in comparison to CNN-based models on the same dataset, as presented in previous research [32], it is evident that the Transformer-based approach demonstrates similar or higher performance over its CNN-based counterparts.

2 Background

2.1 2D Multiple Object Pose Estimation

Pose estimation is a method for detecting key point of the object and classifying it. Input types are usually image or video, and output of the pose estimation model is the coordinates of the keypoint in the images or video. Number and type of keypoint can vary, and there are several ways to approach to model such as skeleton-based model and contour-based model. However, the most important part of pose estimation is how it can be approached.

It can be classified into three main categories depending on the approach:

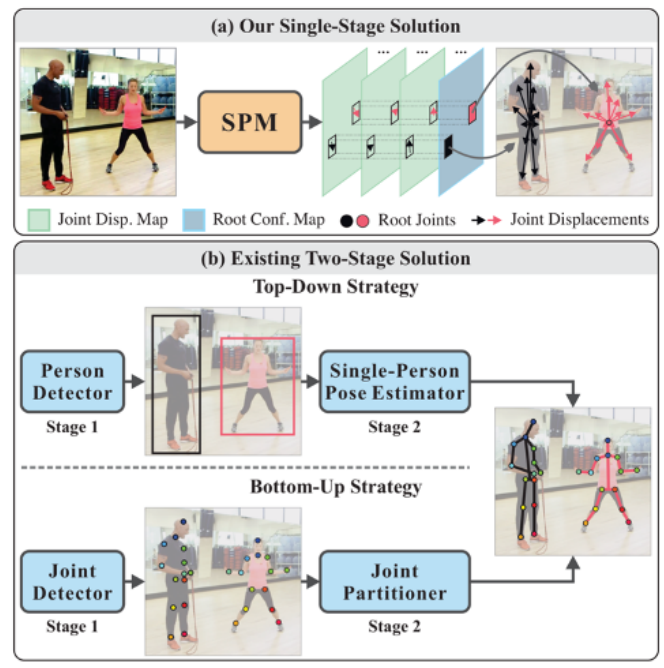


Figure 1: Comparing three methods of Pose Estimation [7]

- Top - down method detects each person (object detection) first and with crop boxes from object detector it inferences key points [5]. With cropped box single person detection can be used. Due to reliance on object detection, performance of detector is very important, and if there are many instances, slow inference can occur.

- Bottom – up method detects all the key points first. After that it groups the key points in instance level. However, grouping is very challenging due to hardness for finding relationship between key points. For efficient post-processing some methods were proposed: Part Affinity Fields (PAFs) [19] and associative embedding

(AE) [6]. It is usually faster than top-down method, but performance is often lower than the top-down method.

- Single – stage method is the method SPM [7] proposed to overcome limitations of above two methods. It regresses predicted value of pose to location of each object. There are several methods to improve accuracy from SPM such as using keypoint heatmap or dynamic instance-aware convolutions [8, 9].

2.2 Transformer

Transformer was first introduced in "Attention is all you need" [1]. It is sequence-to-sequence model which can maintain information about order. Also, transformer efficiently overcomes limitation of other sequence-to-sequence models which is losing information by attention method. Because of this advantage transformer is now widely used in natural language process area [14, 15]. Recently, ViT [2] introduced transformer also can be efficiently used in vision task such as object detection [34] and segmentation [17].

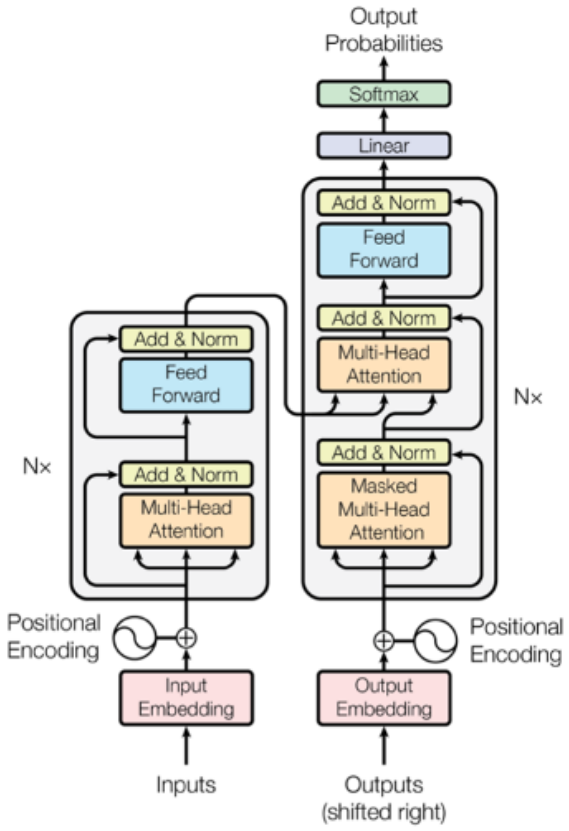


Figure 2: Architecture of Basic Transformer [1]

Function of transformer can be divided into three parts:

- Attention mechanisms enable transformer to calculate attention score between each token. Computation of attention score can be done by using Query, Key, and Value vector. With attention score, transformer can understand relationship between information and decide which part of information it has to focus. In self-attention, all Query, Key, and Value vectors originate from the same source, as the attention is directed towards the input's own elements.

$$\text{FFNN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{1}$$

- Feed Forward Neural Network is used to process the output from previous attention layer to transfer suit format of next attention layer.

$$\text{RC}(x) = \text{Sublayer}(x) + x \tag{2}$$

$$\text{LN} = \text{LayerNorm}(x + \text{sublayer}(x)) \tag{3}$$

- Residual Connection and Layer Normalization (Add & Norm) can help training transformer. Residual connections involve adding input and output of sublayer, effectively creating a shortcut that helps preserve information from previous layers. Layer normalization can be performed after residual connection. During this process, it calculates the mean and variance of last dimension tensor, and it uses this to normalize the values to aid in stabilizing training.

With these functions, encoders and decoders can be constructed. Each of the N encoders comprises two sublayers. The first sublayer is a multi-head self-attention layer which means multiple self-attention mechanisms operate in parallel. The second sublayer is a feed forward neural network, and after each sublayer, add & norm operation supports training.

The decoder consists of three sublayers. The first sublayer is masked multi-head self-attention layer which has similar role as first sublayer of encoder except masking is applied. The second sublayer is encoder - decoder attention layer. Different from two previous attention layers, it is not self-attention layer. Query vectors are from the decoder vector, but the Key and Value vectors are from the encoder vector, and this allows target sequence pays attention to input sequence. Therefore, it measures attention scores of how input sequence influence target sequence. The third sublayer and add & norm work same as second sublayer of encoder. Embedding and positional encoding are required before input data for encoder and decoder to ensure proper representation and encoding of positional information of sequences.

2.3 Transfer Learning

Acquiring sufficient dataset and computational resources, particularly GPUs, can pose a challenge in achieving satisfactory model performance. However, the technique of transfer learning itself has emerged as a practical solution to this problem. It involves utilizing a pre-trained model from a different task and fine-tuning it with respect to the target dataset. This approach has been shown to yield high-performance results with less computational resources, as compared to learning from scratch.

Transfer learning has been demonstrated to be effective across various computer vision tasks, including human pose estimation [11], image classification [10], and object detection [34]. Building upon the success of previous

research that demonstrated the effectiveness of transfer learning from human pose estimation to animal pose estimation [27], this project employs a transfer learning approach from human pose estimation to NHP pose estimation with transformer.

3 Related Works

3.1 Multiple Human Pose Estimation with Transformer

Convolutional Neural Networks (CNN) have been the representative approach for multi-person human pose estimation tasks, with various studies employing this method [18, 19, 20, 21, 22]. However, recently, various approaches based on transformers have been introduced, and they have achieved remarkable performance. Notably, TFPose and PRTR [3, 4] employ the sequence-to-sequence aspect of transformers to predict K-length sequential coordinates (where K denotes the number of keypoints) and, in turn, address the regression-based pose estimation problem using transformers. PETR [13] has further demonstrated that transformers can facilitate a fully end-to-end process for pose estimation.

Transformers offer a distinct advantage through their attention mechanism, which facilitates the computation of attention scores between keypoints, consequently assisting in connecting individual keypoints. Moreover, transformers have the potential to achieve superior performance compared to CNN-based approaches, provided that the model is sufficiently deep. However, training an complicate transformer-based model demands a more extensive dataset because of a greater number of parameters than a CNN-based model.

3.2 Non-Human Primates Pose Estimation

Compared to human pose estimation, research on non-human primate (NHP) pose estimation is relatively scarce. Initial studies in NHP pose estimation employed marker-based approaches [23]. As deep learning-based human pose estimation methods emerged, analogous techniques were gradually introduced for animal pose estimation [24, 25]. However, these works were limited to detecting single primates. In response, DeepLabCut (DLC) [26], a deep learning framework for single-animal body feature detection, was developed with the potential for future multi-animal detection. Building on this foundation, recent work approached using OpenPose to solve multiple NHP pose estimation task [28].

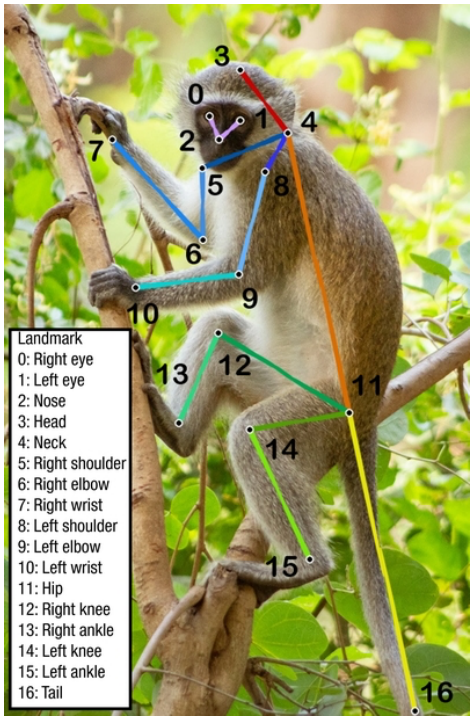
Despite the successes of transformer-based approaches in various computer vision tasks, such as DETR [34] for object detection and SAANet [16] for crowd counting, there has been no exploration of transformer-based methods for NHP pose estimation. Recently, "Transferring Dense Pose to Proximal Animal Classes" [27] demonstrated that transfer learning from pre-trained models on human datasets could lead to substantial achievements in animal pose estimation tasks. This research gives inspiration that using transfer learning can be an effective solution for overcoming the limitation of transformers, which typically require large dataset, thus offering a useful approach in field with limited data. Consequently, the present study aims to demonstrate the effective application of transformers to the NHP pose estimation task, building upon the insights gained from transfer learning techniques.

Building upon the exceptional performance of transformer-based models in human pose estimation and the demonstrated success of transfer learning from pre-trained models on human datasets, the current study presents a pioneering approach to non-human primate pose estimation utilizing transformers. Overcoming dataset limitations is made possible through the application of transfer learning from pre-trained human keypoint models, as evidenced by [27]. Moreover, data augmentation techniques, proved by [9], are employed to further enhance the model's performance and generalization capabilities.

4 Methodology

4.1 Dataset / Baseline Model

4.1.1 Dataset Description




```
{
  "filename": string,
  "species": string,
  "bbox": [x, y, w, h],
  "landmarks": [x1, y1, ..],
  "visibility": [v1, v2, ..]
}
```

Figure 3: Open Monkey Challenge Dataset Description and Annotation [32]

This project utilizes the open monkey challenge dataset to develop a model for non-human primate (NHP) pose estimation. The dataset is comprised of 111,529 images of NHPs collected from the Internet, three National Primate Research Centers, and the Minnesota Zoo. 111,529 images are divided into three subsets: training (60%, 66,917 images), validation (20%, 22,306 images), and test (20%, 22,306 images). Each image is annotated with five variables: filename, species, bbox, landmarks, and visibility.

Filename variable contains the name of each image, while the species variable identifies the species of the NHP in each image. Since this project focuses on pose estimation, it does not consider the species variable. Bbox variable provides the bounding box information for each NHP in each image. The x and y coordinates correspond to the upper left point of the bounding box, while the w and h variables represent the width and height of the bounding box. This variable can also be used for training object detection tasks.

Landmarks and visibility variables, in combination with the bbox variable, play a crucial role in training, validation and evaluation in the project. The landmarks variable provides the coordinates and visibility for each keypoint used in training, validation and evaluation. Specifically, X1 and Y1 represent the coordinate of the first keypoint, while V1 denotes its visibility.

4.1.2 Baseline Model

This study adopts on the high-performance capabilities of a multiple human pose estimation transformer model which names PETR [13]. PETR employs a basic transformer [1] to address the multiple human pose estimation task. Compared to alternative transformers specifically designed for visual tasks, the basic transformer boasts a more straightforward architecture, thereby facilitating transfer learning. Additionally, PETR is an end-to-end and single-stage method model, so it eliminates the need for hand-crafted modules.

The pre-trained model utilized in this study is trained on the COCO 2017 Keypoint dataset [37]. This dataset contains multiple instances annotated with 17 keypoints, which coincides with the number of keypoints in the Open Monkey Challenge dataset [32]. As a result, employing this pre-trained model streamlines the fine-tuning process by maintaining a consistent number of keypoints for prediction. Moreover, the chosen model demonstrates superior performance compared to models employing larger backbone networks [42, 43]. Consequently, with reduced computational complexity and memory requirements, the pre-trained model outperforms its counterparts while optimizing resource usage.

4.2 Proposing architecture

4.2.1 Architecture Details

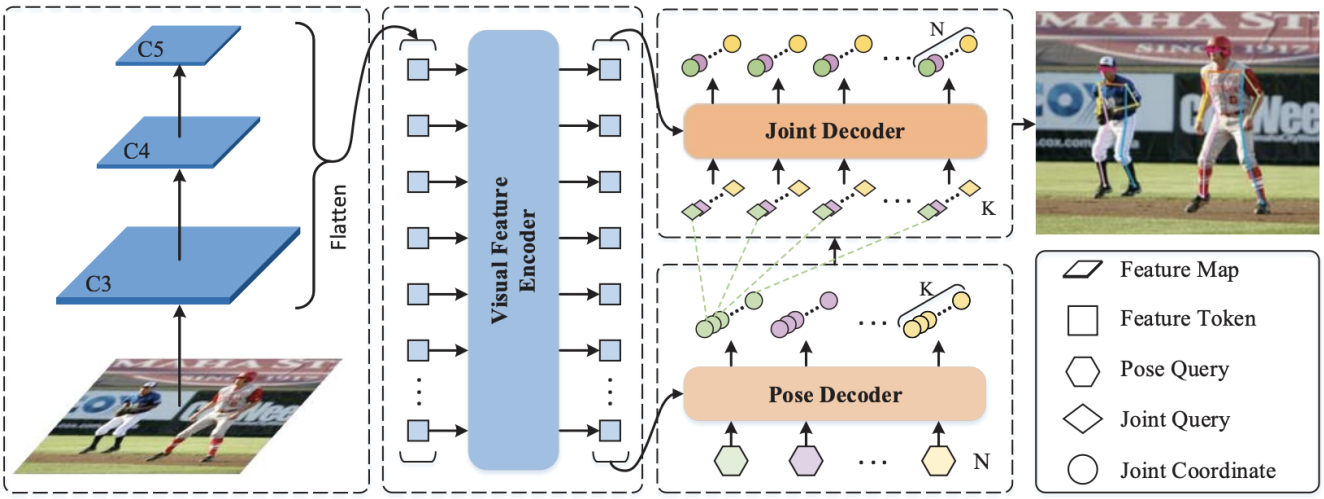


Figure 4: Architecture of PETR [13]

Architecture of PETR which achieved success in multiple human pose estimation can be divided into three components: the backbone network, encoder, and decoder. The ResNet [30] serves as the backbone network, extracting multi-scale feature maps from the input image. These multi-scale feature maps are subsequently projected onto 256 channels through a spatial-wise fully-connected (FC) layer, ultimately yielding flattened feature tokens. Afterwards, the concatenated feature tokens are fed into the visual encoder, which passes them through six deformable encoder layers, each containing a multi-scale deformable attention module and a feed-forward nerual network (FFNN). This process produces a multi-scale visual feature memory as an output.

The decoder's architecture is partitioned into two modules: the pose decoder, which predicts the pose, and the joint decoder, which refines the prediction from the pose decoder. The pose decoder receives embedded pose queries and calculates an attention score that enables the queries to interact with one another in a self-attention module (pose-to-pose attention). Afterward, each query extracts features from the visual multi-scale feature memory obtained from the encoder using a deformable cross-attention module (feature-to-pose attention). This process generates instance-aware query features, which are then fed into multi-scale prediction heads comprising a classification head and a pose regression head. The classification head's linear projection layer (FC) predicts the confidence score for each object, while the pose regression head's multi-layer perceptron predicts the relative offsets (K reference points).

The joint decoder receives K randomly initialized joint queries, predicted by preceding pose decoder and calculates an attention score that enables the queries to interact with one another in a self-attention module (joint-to-joint attention). After this, each query extracts features from the visual multi-scale feature memory obtained from the encoder using a deformable cross-attention module (feature-to-joint attention). The pose regression head's multi-layer perceptron predicts 2D joint displacement $\Delta J = (\Delta x, \Delta y)$. Furthermore, each pose and joint decoder layer comprises three individual layers arranged in the structure described above, and these decoder layers operate progressively to estimate the pose coordinates.

4.2.2 Data Preprocessing

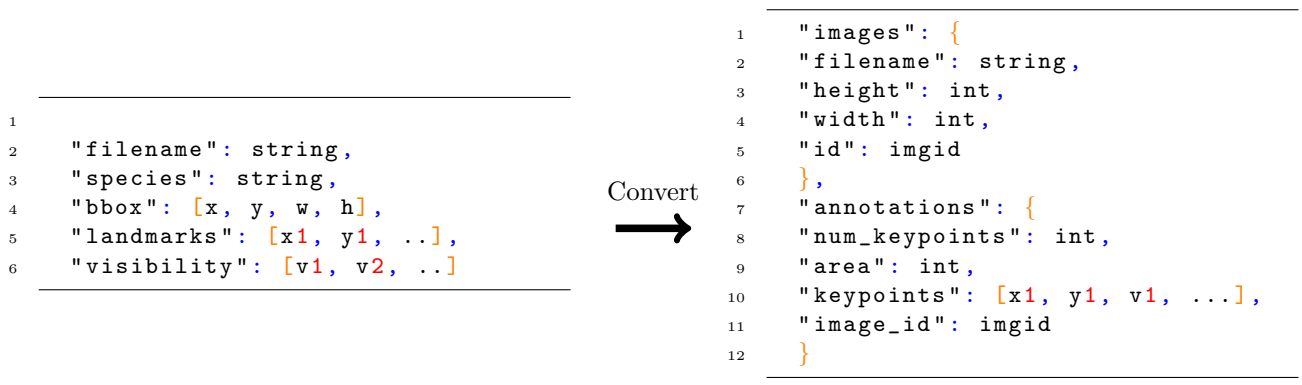


Figure 5: Converting Annotation format: Open Monkey Challenge dataset [32] to COCO keypoint dataset [37]

Data pre-processing is an essential step in preparing data for model training. In this project, it is imperative to note that the COCO keypoint type dataset [37] annotation format is the only supported format for training. As such, it becomes necessary to convert the data to this format, a process which will be explicated below.

Specifically, the filename in the image category of the COCO annotation type is directly transferred to the filename of the open monkey challenge dataset [32] annotation. The height and width of the COCO format correspond to the third and fourth elements of the bbox of the open monkey dataset, respectively. For the ID attribute, an integer value is extracted from the filename itself, that a file named "train_0001555.jpg" would have an ID of 1555.

Moreover, the num_keypoints attribute within the annotation box should be set to 17, and the area should be calculated as the product of the height and width. The keypoints attribute stores the coordinates and visibility information for each keypoint. The x1 and y1 attributes represent the x and y coordinates of the first landmark, while the v1 attribute indicates whether the first landmark is visible or not. Lastly, the image_id has the same information as the id attribute found in the images box.

Furthermore, in order to address the challenge of limited data availability, the present study applies a technique referred to as data augmentation. This technique can be divided into three methods: random resize, random flip, and random crop. Random resize method involves the manipulation of image dimensions either by zero-padding or by employing a resizing function. In instances where the initial image is smaller than the desired dimensions, zero-padding is utilized to enlarge the image. Conversely, when the original image exceeds the target size, a resizing function is employed to reduce the dimensions accordingly. Random flip can be done by flipping input images either horizontally or vertically with respect to symmetric pairs. In this project, symmetric pairs can be defined as both eyes, shoulders, elbows, wrists, knees, and ankles. Finally, random crop is implemented by selecting regions within the input images and cropping them to a decided width and height. If the decided size is bigger than the given image, zero padding is applied to make the given image to the decided size.

4.2.3 Fine Tuning

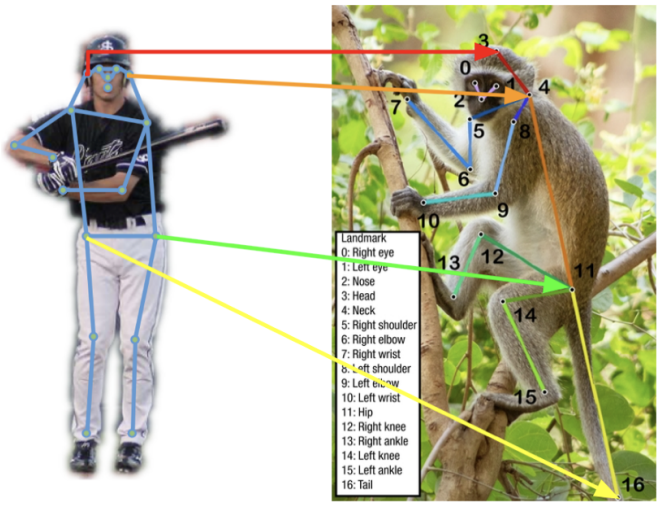


Figure 6: Fine tuning from COCO 2017 dataset [37] to Open monkey challenge dataset [32]

Fine-tuning is a crucial step that involves modifying a pre-trained model to achieve the best performance for a particular task. When substantial gaps exist between the pre-trained task and the target task, modification to the pre-trained model's architecture is necessary. However, in the context of human pose estimation and non-human primate pose estimation, these tasks share sufficient similarities, and the number of keypoints for the COCO 2017 keypoint dataset [37] and the Open Monkey Challenge dataset [32] is identical. Consequently, the same architecture can be employed for both the human pose estimation model and the non-human primate pose estimation model.

To prevent the loss of information from the pre-trained model during fine-tuning, the learning rate is scaled to 10% of the previous training. Additionally, to reduce training time, parameters other than the encoder and decoder are frozen. The process of selecting which layers to freeze is carefully considered through ablation study, which will be discussed later.

The self-attention module is a crucial component of the pose estimation model as it enables pose queries to interact with one another by computing attention scores. This feature is particularly useful in the context of the PETR [13] model, which can learn new features from non-human primate datasets by updating the trained features

from human datasets. Figure 6 demonstrates that this process can be achieved by using the most similar keypoint from the pre-trained model to update new keypoints that were not initially annotated in the human dataset. Especially, the head and neck keypoints of non-human primates can be updated from the ear keypoints of the pre-trained model, and the hip keypoint can be updated from either the left or right hip keypoints. Keypoints that are annotated for the same part as the human dataset will also be updated to fit the non-human primate feature. However, updating the tail keypoint of non-human primates from the keypoint of another hip is a challenging task as the pre-trained model lacks a comparable feature for non-human primates. This issue will be thoroughly evaluated and addressed in evaluation and ablation study.

4.3 Training / Validation

4.3.1 Training

This project employs a transfer learning methodology to address the challenge of non-human primate pose estimation which is inspired from [27]. A pre-trained model based on the COCO 2017 keypoint dataset with 100 epochs is utilized, leveraging its existing knowledge to enhance the estimation of 17 keypoints for one non-human primate in each of the 66,916 images. Given the inherent similarities between the human and non-human primate pose estimation tasks, as well as the identical number of keypoints in both datasets, this project adopts the same architecture without modifications before updating the pre-trained model's parameters. This transfer learning technique enables the project to realize two significant advantages: minimizing training time and compensating for the shortage of available data.

The primary objective of training is to minimize the error value computed by appropriate loss functions, which plays a vital role in the successful execution of the given task. In this study, four loss functions are employed during the training process, namely, Classification loss, L1 loss, OKS loss, and Heatmap loss, as proposed in PETR [13]. The Classification loss function is used to update the parameter of the classification head in the pose decoder, which predicts the confidence score, like deformable DETR [34].

$$OKS(P, P^*) = \frac{\sum_{i=1}^{K} exp(-\frac{\|P_i - P_i^*\|^2}{2s^2k_i^2})\delta(v_i > 0)}{\sum_{i=1}^{K} \delta(v_i > 0)}$$

Figure 7: Formula for Object Keypoint Similarity (OKS) loss [37]

L1 loss, which functions similarly to Manhattan distance, and the Object Keypoint Similarity (OKS) loss functions operate as the optimization criteria for fine-tuning the parameters of the pose and joint regression head. This module is tasked with estimating the pose, and subsequently refining the predicted pose through the employment of the pose and joint decoder. Nevertheless, the L1 loss function demonstrates a limitation concerning the inconsistent scales for small and large poses, despite possessing similar relative errors. To counter this drawback, the OKS loss function which consider area of the bounding box (s^2) is integrated alongside the L1 loss function, considering both the area of the bounding box and the accuracy of pose prediction. Furthermore, OKS loss will be discussed more in the validation part.

Heatmap loss function is applied to update the parameters of the visual feature encoder by computing a modified version of the focal loss [35], which assesses the disparity between the ground truth heatmap and the predicted heatmap. The generation of the predicted heatmap involves extracting feature tokens from the encoder output and subsequently reshaping these tokens to their original spatial dimensions. A deformable transformer encoder is then

applied to the reshaped feature tokens, yielding the predicted heatmap. As a result, the overall loss function can be expressed as illustrated in Figure 8.

$$\text{Overall Loss} = \text{Classification loss} + \lambda_1(\text{L1 loss}) + \lambda_2(\text{OKS loss}) + \lambda_3(\text{Heatmap Loss})$$

Figure 8: Overall loss (λ_1, λ_2 and λ_3 are weights of loss)

Moreover, for optimization, the Adaptive Moment Estimation optimizer [36] (Adam) with a base learning rate of 2×10^{-5}, a momentum of 0.9, and a weight decay of 1×10^{-4}, is selected. The learning rate is scaled to 10% of the pre-training model's learning rate, as indicated in fine-tuning. The Adam optimizer incorporates an adaptive learning rate and momentum feature, which allows the model to fit the gradients of varying magnitudes and smooth out the optimization process.

4.3.2 Validation

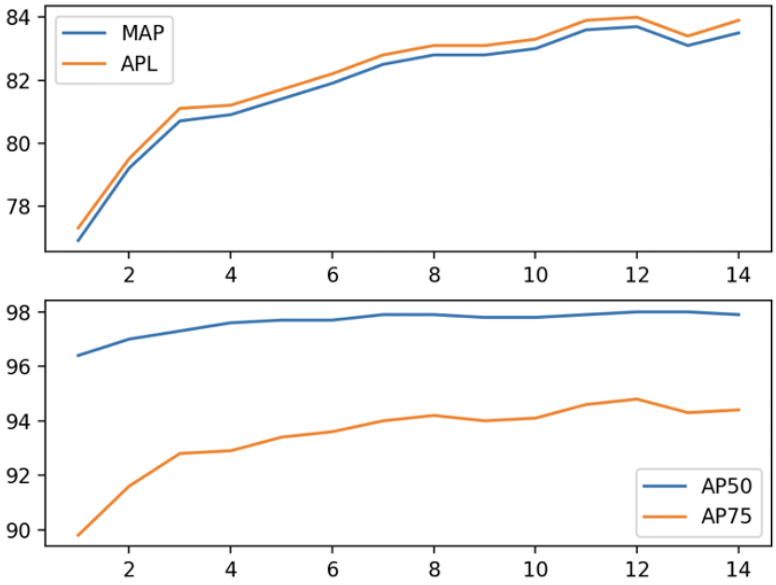


Figure 9: Graph of accuracy for each epoch on Validation Set

$$\text{AP@}\varepsilon = \frac{1}{17J}\sum_{j=1}^{J}\sum_{i=1}^{17}\delta(\text{OKS}_{ij} \geq \varepsilon)$$

Figure 10: Formula of Average Precision at ε

Validation is a critical step in the training of a machine learning model to prevent overfitting and determine the optimal point to stop training. In this project, a validation set consisting of 22,306 images with single non-human primate annotation is used.

To evaluate the accuracy of the model on the validation set, the average precision metric is employed. It is based on the object keypoint similarity (OKS) loss which can be found from figure 7 and 10. The OKS loss is calculated using the Euclidean distance between the predicted keypoint and the ground truth keypoint ($|P_i - P_i^*|$), where s^2 denotes the area of the bounding box. Also, the visibility of the ith keypoint (v_i) assists the model in determining if the ith keypoint should be included when calculating the loss. The average precision metric considers the individual weight of each keypoint (k_i) which is a part of the OKS loss and is suitable for overall evaluation for each epoch.

Specifically, the AP50 and AP75 metrics represent the average precision at 0.5 and 0.75, respectively. The mean average precision (MAP) is calculated by calculating mean of the average precision across all object scales, ranging from 0.5 to 0.95, in increments of 0.05. The average precision large (APL) metric which is used to evaluate the MAP for images with an area larger than 96^2, proves helpful for assessing the model's performance on larger objects.

The graph presented in figure 9 shows the model's performance on the validation set at different epochs. The accuracy of the model does not improve beyond epoch 12 in any of the evaluation criteria. Therefore, it can be defined that the model reaches a saturation point at epoch 12. These findings suggest that due to the transfer learning process from a pre-trained model that underwent 100 epochs, the optimal epoch for the current model was attained at an accelerated pace.

5 Experimental Evaluation

5.1 Experimental setting

In this study, the proposed PETR-based model for NHP pose estimation was trained and evaluated using a virtual environment powered by Amazon Web Services (AWS). The hardware specifications for this virtual environment include an Intel(R) Xeon(R) Platinum 8259CL CPU @ 2.50GHz, a Tesla-T4 GPU, and 16GB of memory. The software stack employed for this project consists of Python as the programming language, PyTorch [44] as the deep learning framework, and OpenMMLab which is composed with mmcv [46] and mmdetection [45] as the platform for the implementation of the model for Non Human Primates pose estimation. The evaluations were conducted on the test set of the Open Monkey Challenge dataset [32] for fair comparison.

5.2 Quantitative results

$$PCK@\epsilon = \frac{1}{17J} \sum_{j=1}^{J} \sum_{i=1}^{17} \delta \left(\frac{\|P_i - P_i^*\|}{W} < \epsilon \right) \tag{4}$$

Figure 11: Formula of PCK at ϵ [19]

Evaluation is a crucial aspect of deep learning in which the performance of a model is assessed. This project is evaluated on Open Monkey challenge test set which is composed of 22,306 images for unbiased evaluation. For evaluation metrics, this project utilizes two widely adopted evaluation metrics, namely the Average Precision (AP) and Probability of Correct Keypoint (PCK).

Both metrics rely on Euclidean distance as a measure of accuracy. Specifically, Average Precision (AP) was utilized as the primary metric for overall assessment, taking into account the varying weights assigned to each keypoint, similar to the validation process. Meanwhile, the Percentage of Correct Keypoints (PCK) was employed for the evaluation of individual keypoints. In line with previous research [32], this project also uses the COCO keypoint challenge weights (k_i) for evaluation on the same dataset, disregarding visibility for OKS loss to ensure a fair comparison with earlier approaches. Thus, the key difference between the OKS loss applied in training and validation and the one used in evaluation is whether visibility is considered or not. PCK, on the other hand, which was employed to determine the accuracy of individual keypoints, calculates the results based on number of instances J , width of the bounding box W, and spatial tolerance for correct detection ϵ that determines if the predicted keypoint is considered correct.

The two aforementioned metrics were utilized to evaluate the performance of the current model relative to other models that employ different CNN architectures. The project compares the following models which are introduced by [32]: Deeplabcut [26], CPM [38], Hourglass [39], HRNet-W32, HRNet-W48 [40], SimpleBaseline (ResNet 152), and SimpleBaseline (ResNet 101) [41] which are top-down methods, HigherHRNet-W32, and HigherHRNet-W48 [42] which are bottom-up methods.

5.2.1 Overall Evaluation

Model	AP@0.5	AP@0.6	AP@0.7	AP@0.8	AP@0.9	mAP
DeepLabCut	92.3	89.7	83.9	74.1	52.6	73.2
CPM	91.8	86.1	78.9	69.6	54.2	72.9
Hourglass	91.3	85.7	80.8	74.7	63.8	74.5
HRNet-W32	89.4	80.6	71.0	64.6	65.7	70.7
HRNet-W48	90.2	85.7	80.8	74.7	63.8	76.5
SimpleBaseline (ResNet 152)	89.5	84.8	81.2	76.9	**67.8**	78.5
SimpleBaseline (ResNet 101)	**97.2**	82.6	65.9	46.9	31.4	65.3
HigherHRNet-W32	88.0	79.5	57.3	32.0	20.0	59.1
HigherHRNet-W48	91.5	82.6	65.9	46.9	31.4	65.3
My Model	93.8	**93.7**	**92.4**	**87.9**	60.0	**80.7**

Table 1: Comparison with AP matrix with different threshold on OpenMonkeyChallenge [32] test set.

Table 1 displays the performance of the model developed in this project, by comparing it with other models by using the average precision. Each row on the table represents the mean precision at different threshold values ranging from 0.5 to 0.9 with the last row being the mean. The average performance is represented by the mAP across average of various threshold values ranging from 0.5 to 0.95 in an increasing pattern of 0.05. Additionally, all the values on the table are rounded off to one decimal place to make it intelligible for comparison.

Upon comparison, it is evident that the model developed in this project demonstrates superior overall performance, except at the threshold value of 0.5 and 0.9. This outcome implies that a transformer-based approach could potentially deliver enhanced results for non-human primate pose estimation tasks compared to a CNN-based method. It is crucial to agree that the high performance of the transformer model might be attributable to the increased number of parameters used. While the transformer model necessitates a larger dataset, it is capable of achieving high performance levels.

Additionally, the technique employed in this project to overcome the limitation of a small dataset was successful. The results indicate that transfer learning from a model trained on human datasets can be an effective solution for non-human primate pose estimation tasks. Overall, these findings suggest that the transformer model may offer a promising approach for improving the accuracy of non-human primate pose estimation tasks.

5.2.2 Keypoint Evaluation

Model	R.Eye	L.Eye	Nose	Head	Neck	R.Shoulder	R.Elbow	R.Wrist	L.Shoulder	L.Elbow	L.Wrist	Hip	R.Knee	R.Ankle	L.Knee	L.Ankle	Tail	Mean
DeepLabCut	0.937	0.938	0.936	0.926	0.922	0.906	0.874	0.814	0.908	0.875	0.81	0.855	0.868	0.819	0.862	0.816	0.747	0.871
CPM	0.995	0.995	0.994	0.96	0.945	0.887	0.825	0.785	0.886	0.869	0.814	0.892	0.893	0.902	0.924	0.854	0.809	0.896
Hourglass	0.997	0.996	0.996	0.96	0.925	0.864	0.823	0.825	0.839	0.836	0.846	0.869	0.881	0.891	0.91	0.853	0.814	0.89
HRNet-W48	0.997	0.996	0.996	0.951	0.94	0.876	0.858	0.846	0.867	0.856	0.864	0.885	0.893	0.906	0.923	0.863	0.842	0.903
HRNet-W32	0.997	**0.997**	0.996	0.958	0.934	0.874	0.844	0.828	0.86	0.857	0.844	0.867	0.898	0.897	0.921	0.854	0.83	0.897
SimpleBaseline (ResNet 152)	**0.997**	0.996	**0.996**	0.954	0.942	0.868	0.855	0.842	0.864	0.858	0.855	0.883	0.892	0.907	0.921	0.855	0.838	0.901
SimpleBaseline (ResNet 101)	0.995	0.995	0.994	**0.983**	0.877	0.837	0.776	0.757	0.82	0.798	0.794	0.827	0.87	0.868	0.897	0.828	0.756	0.863
HigherHRNet-W32	0.962	0.979	0.981	0.978	0.856	0.798	0.724	0.782	0.661	0.62	0.804	0.71	0.899	0.804	0.897	0.82	0.633	0.818
HigherHRNet-W48	0.986	0.986	0.985	0.965	0.86	0.796	0.777	0.82	0.721	0.73	0.831	0.779	0.863	0.834	0.874	0.827	0.715	0.844
My Model	0.953	0.953	0.953	0.952	**0.95**	**0.944**	**0.931**	**0.911**	**0.945**	**0.932**	**0.912**	**0.91**	**0.927**	**0.913**	**0.929**	**0.914**	**0.846**	**0.928**

Table 2: Model comparison with PCK@0.2 metric of each Keypoint on OpenMonkeyChallenge [32] test set.

Table 2 presents an analysis of the accuracy of each keypoint compared with multiple models by using probability of correct keypoint (PCK). Also, all values are rounded to three decimal places to make them easier to understand and compare. The suggested model demonstrates the highest accuracy for keypoints, except for the eyes, nose and head. While the accuracy of the suggested model for both eyes, nose and head was lower than the best-performing model for these keypoints, it still exhibits a high level of accuracy, suggesting that further adjustments are unnecessary. However, particular attention must be given to improving the accuracy of the tail keypoint, as compared to other keypoints, it exhibited a comparatively lower level of accuracy. This can be attributed to two reasons: the visual ambiguity of the tail keypoint and the transfer learning approach employed in the model.

Firstly, the tail keypoint is visually more ambiguous than other keypoints, as shown in Figure 13. It is often far away from the body and can be easily confused with branches. Secondly, due to the transfer learning approach used in the model, the pre-trained model on a human dataset does not include the tail feature. In the fine-tuning process, the feature of the hip was updated to represent the tail feature of the new model. However, the features of the tail and hip are vastly different, and the model requires further training to fit the tail feature appropriately. Ablation study will address the underfitting of the tail feature and propose a solution to overcome this limitation.

5.3 Ablation Study

Ablation study addressed two objectives: justifying the decision to freeze layers during the fine-tuning process, and investigating the cause of low accuracy in tail keypoint detection. The first study involved comparing the performances of four models with different frozen layers, using the same accuracy measurement methods employed in validation. In the second study, the project aimed to determine if transfer learning was responsible for the low accuracy observed in tail keypoint detection by comparing an overfitted model with the optimal one. The validation set was employed in both studies because it is more closely aligned with the training and validation processes, and the evaluation should remain unbiased.

5.3.1 Justification of freezing layer decision

Model	AP50	AP75	APL	mAP
Epoch_1	96.3	90.2	77.4	77.1
Encoder Only	96.8	**91.8**	79.4	79.1
Decoder Only	96.1	90.4	77.9	77.6
Epoch_2	**96.9**	91.6	**79.5**	**79.2**

Table 3: Comparison between models with different updated layers on OpenMonkeyChallenge [32] validation set

Table 3 illustrates the comparison of four different models, where Epoch_1 is trained for one epoch from the pretrained model, Encoder Only is solely updated for parameters of the encoder from Epoch_1, Decoder Only is updated exclusively for parameters of the decoder from Epoch_1 and Epoch_2 is updated for both the encoder's and decoder's parameters from epoch 1.

While the model updated only for the encoder showed considerable improvement in comparison to Epoch_1, Epoch_2 which was trained for both the encoder and the decoder produced the best results. Moreover, the performance of the model updated solely for the decoder did not surpass that of Epoch_1, which served as the starting point for this model.

These findings imply that training the encoder alone achieves better performance than training only the decoder, suggesting that if the visual feature memory is not sufficiently refined by the encoder, fitting the decoder becomes unattainable. Consequently, the model fails to achieve high performance, even when the decoder is trained extensively. In contrast, when both the encoder and decoder are effectively trained, the model can reach high performance. These results guided the decision to freeze layers during the fine-tuning process.

5.3.2 Proof of underfitted tail keypoint

Model	R.Eye	L.Eye	Nose	Head	Neck	R.Shoulder	R.Elbow	R.Wrist	L.Shoulder	L.Elbow	L.Wrist	Hip	R.Knee	R.Ankle	L.Knee	L.Ankle	Tail
Epoch_12	**0.95**	**0.95**	**0.95**	**0.948**	**0.947**	**0.942**	0.925	0.906	**0.941**	**0.928**	**0.909**	**0.909**	0.924	**0.909**	**0.924**	**0.91**	0.844
Epoch_14	0.949	0.949	0.949	0.947	0.946	0.94	0.925	0.906	0.94	0.927	0.908	0.907	0.924	0.906	0.923	0.908	**0.846**

Table 4: Model comparison with PCK@0.2 metric of each Keypoint on OpenMonkeyChallenge [32] validation set.

In the evaluation, transfer learning is considered the cause of low tail keypoint accuracy. During validation, epoch_12 can be viewed as the saturation point, representing the optimal model, while epoch_14 signifies a model where overfitting has occurred. Consequently, this study presents a comparison between these two models in Table 4.

If the pretrained model encompasses every feature of keypoints for non-human primates, epoch_12 should achieve higher accuracy than epoch_14 for all keypoints. In line with this expectation, epoch_12 exhibits equal or better accuracy for the majority of keypoints. However, epoch_14 outperforms epoch_12 in terms of tail keypoint accuracy. The most plausible explanation for this discrepancy is that epoch_12 is underfitted, having not been trained with enough epochs. This outcome provides evidence supporting the analysis of low tail keypoint accuracy.

To address this issue, the project proposes modifying the loss function. Specifically, OKS loss [37] incorporates a weight (k_i) for each keypoint, where the weight for the tail keypoint can be increased to yield a higher error value.

This adjustment would enable the model to reach the optimal point for predicting tail keypoints more rapidly. Consequently, determining the appropriate weight value and rescaling the number of epochs could be future work aimed at enhancing accuracy for all keypoints by concentrating on improving tail keypoint accuracy.

5.4 Qualitative results

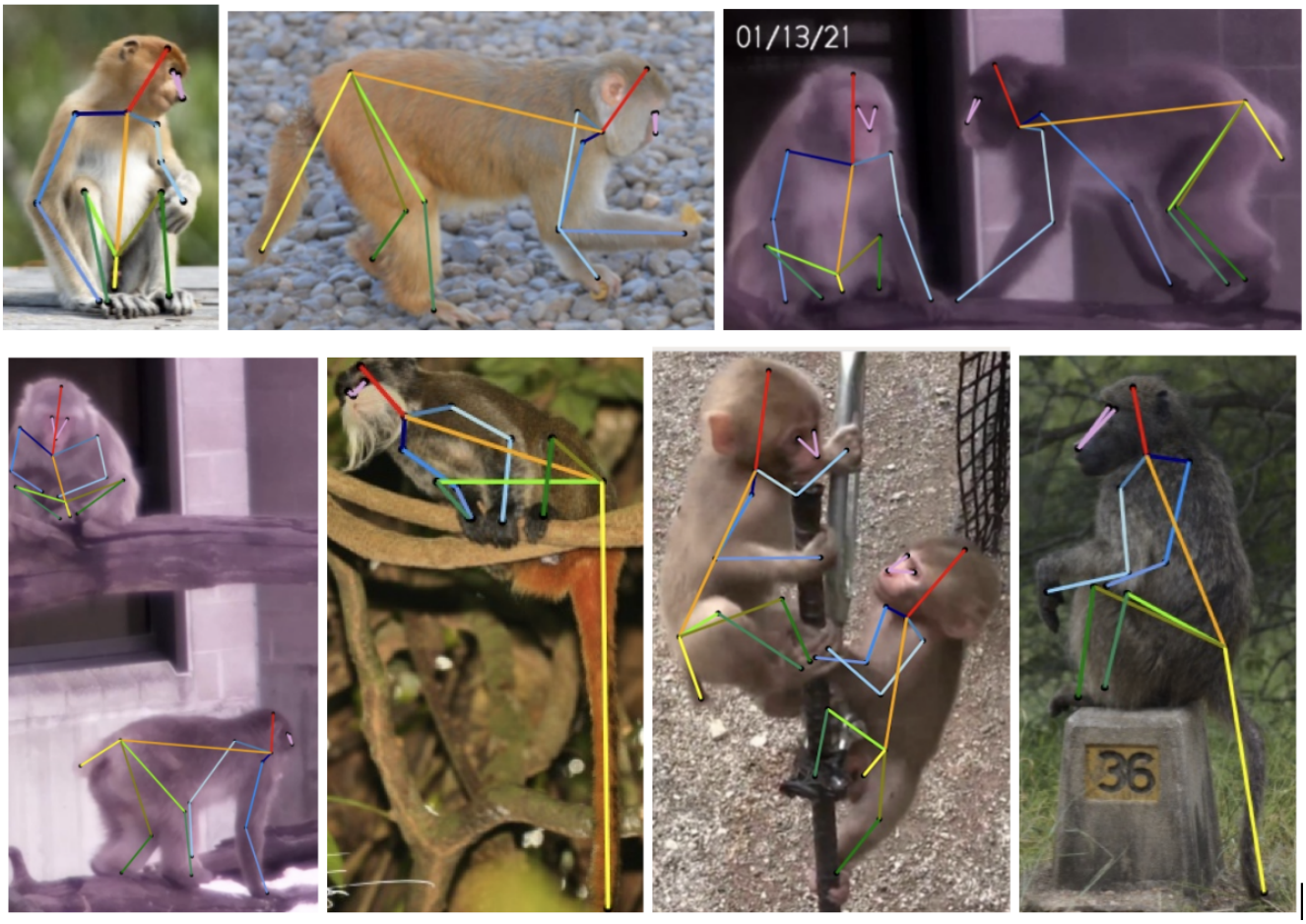


Figure 12: Visualization results on OpenMonkeyChallenge [32] test Set

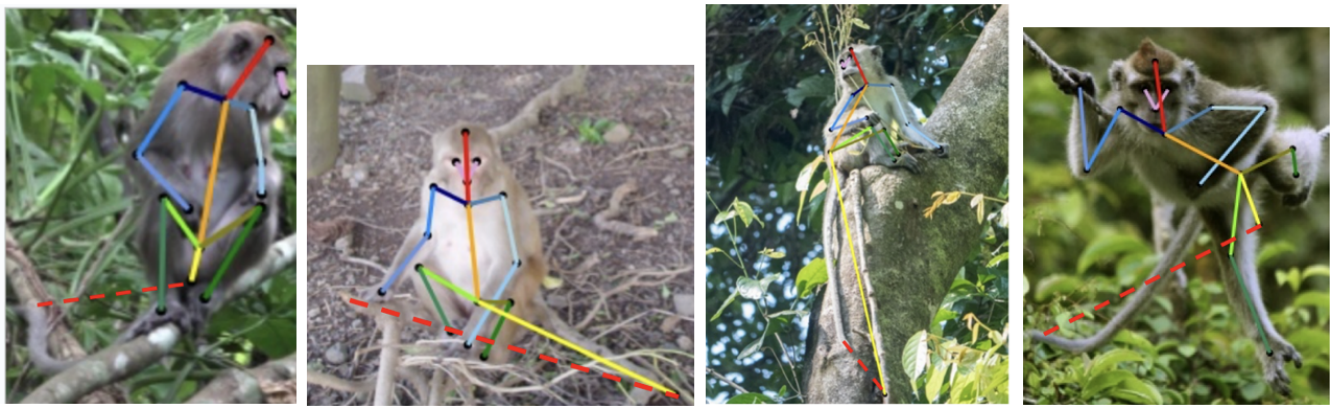


Figure 13: Failure cases about tail (Red dotted line shows distance between ground truths and predicted points)

5.5 Discussion

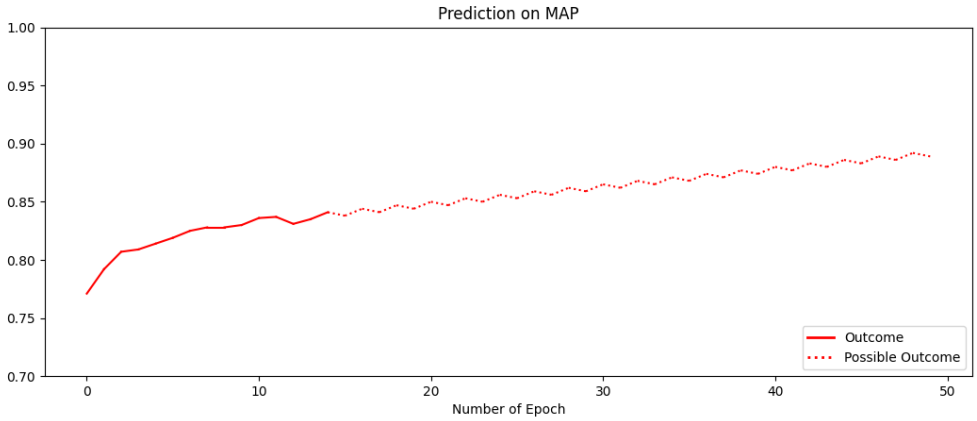


Figure 14: Observed validation performance (solid line) and potential trends with more epochs (dotted line)

In the present study, the optimal model performance was observed at the 12th epoch among the 14 models trained with varying epoch numbers. However, as previously mentioned, it is conceivable that a model trained for more than 14 epochs could yield improved performance. In such a scenario, the accuracy graph may exhibit a gradual, oscillatory upward trend as Figure 14. Notably, the ablation study revealed that the tail keypoint in the optimal model (12th epoch), is underfitted.

Moreover, this investigation is employed in the entire dataset without distinguishing between species. As a result, future research can focus on three key objectives: checking whether the 12th epoch indeed represents the optimal model, enhancing the accuracy of tail keypoint predictions by adjusting the weights assigned to each keypoint in the Object Keypoint Similarity (OKS) loss function, and partitioning the dataset according to species and subsequently improving accuracy by training models specific to each species.

6 Limitation / Conclusion

6.1 Limitation

Owing to the extensive duration of the training process for transformer models and the limited computational resources allocated to such projects, evaluating performance beyond 14 epochs becomes impractical. Nevertheless, the observed trend indicates that the accuracy reaches a saturation point at 12 epochs, with the potential for further improvement if more epochs were to be utilized. Additionally, the Open Monkey Challenge dataset exhibits an intrinsic constraint in assessing the model's capacity to predict poses for multiple instances, given that it exclusively comprises annotations related to a single monkey.

6.2 Conclusion

In conclusion, this study has successfully showed the capabilities of transformer-based models in non-human primate pose estimation, broadening their use beyond human pose estimation. By employing transfer learning and data augmentation techniques, the challenge of limited data availability has been effectively tackled. Moreover, this research has pinpointed a potential enhancement in tail keypoint estimation through loss function refinement.

This work significantly enriches the existing research on non-human primate pose estimation methods, emphasizing the versatility and robustness of transformer-based models in diverse research domains. By harnessing transfer learning in a well-resourced domain, the study proposes a viable solution to overcome data scarcity challenges in various fields. As a result, this investigation paves the way for future advancements and innovations using transformers in a range of vision tasks.

References

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszko- reit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[2] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929, 2020.

[3] KeLi,ShijieWang,XiangZhang,YifanXu,WeijianXu,and Zhuowen Tu. Pose recognition with cascade transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1944–1953, 2021.

[4] Weian Mao, Yongtao Ge, Chunhua Shen, Zhi Tian, Xinlong Wang, and Zhibin Wang. Tfpose: Direct human pose estimation with transformers. arXiv preprint arXiv:2103.15320, 2021.

[5] Hao-Shu Fang, Shuqin Xie, Yu-Wing Tai, and Cewu Lu. Rmpe: Regional multi-person pose estimation. In *Proceedings of the IEEE international conference on computer vision*, pages 2334–2343, 2017.

[6] Alejandro Newell, Zhiao Huang, and Jia Deng. Associative embedding: End-to-end learning for joint detection and grouping. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[7] Xuecheng Nie, Jiashi Feng, Jianfeng Zhang, and Shuicheng Yan. Single-stage multi-person pose machines. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6951–6960, 2019.

[8] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. arXiv preprint arXiv:1904.07850, 2019.

[9] Weian Mao, Zhi Tian, Xinlong Wang, and Chunhua Shen. Fcpose: Fully convolutional multi-person pose estimation with dynamic instance-aware convolutions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9034–9043, 2021.

[10] Taranjit Kaur and Tapan Kumar Gandhi, Deep convolutional neural networks with transfer learning for automated brain image classification, *Machine Vision and Applications*, vol. 31, no. 20, 2020. https://doi.org/10.1007/s00138-020-01069-2

[11] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5693–5703, 2019.

[12] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun, Faster R-CNN: Towards real-time object detection with region proposal networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1467-1475, 2015.

[13] Dahu Shi, Xing Wei, Liangq Li, Ye Ren, and Wenming Tan, End-to-End Multi-Person Pose Estimation with Transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11069-11076, 2021.

[14] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, Dario Amodei, Language models are few-shot learners. In *Proceedings of the Advances in Neural Information Processing Systems*, 2020.

[15] 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.

[16] Xing Wei, Yuanrui Kang, Jihao Yang, Yunfeng Qiu, Dahu Shi, Wenming Tan, and Yihong Gong. Scene-adaptive attention network for crowd counting. arXiv preprint arXiv:2112.15509, 2021.

[17] Yuqing Wang, Zhaoliang Xu, Xinlong Wang, Chunhua Shen, Baoshan Cheng, Hao Shen, and Huaxia Xia. End-to-end video instance segmentation with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[18] Yuanhao Cai, Zhicheng Wang, Zhengxiong Luo, Binyi Yin, Angang Du, Haoqian Wang, Xiangyu Zhang, Xinyu Zhou, Erjin Zhou, and Jian Sun. Learning delicate local representations for multi-person pose estimation. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 455–472, 2020.

[19] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, Yaser Sheikh. OpenPose: Realtime multi-person 2D pose estimation using part affinity fields. In *Proceedings of IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 1, pp. 172–186, 2019, doi: 10.1109/tpami.2019.2929257.

[20] Joao Carreira, Pulkit Agrawal, Katerina Fragkiadaki, and Jitendra Malik. Human pose estimation with iterative error feedback. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4733–4742, 2019.

[21] Xuecheng Nie, Jiashi Feng, Jianfeng Zhang, and Shuicheng Yan. Single-stage multi-person pose machines. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6951–6960, 2019.

[22] Bowen Cheng, Bin Xiao, Jingdong Wang, Honghui Shi, Thomas S Huang, and Lei Zhang. Higherhrnet: Scale-aware representation learning for bottom-up human pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5386–5395, 2020.

[23] Tomoya Nakamura, Jumpei Matsumoto, Hiroshi Nishimaru, Rafael Vieira Bretas, Yusaku Takamura, Etsuro Hori, Taketoshi Ono, and Hisao Nishijo. A markerless 3D computerized motion capture system incorporating a skeleton model for monkeys. PLOS ONE, vol. 11, no. 11, p. e0166154, 2016, doi: 10.1371/journal.pone.0166154.

[24] Rollyn Labuguen, Vishal Gaurav, Salvador Negrete Blanco, Jumpei Matsumoto, Kenichi Inoue, and Tomohiro Shibata. Monkey Features Location Identification Using Convolutional Neural Networks. bioRxiv, Jul. 2018, doi: 10.1101/377895.

[25] Rollyn Labuguen, Jumpei Matsumoto, Salvador Blanco Negrete, Hirosh Nishimaru, Hisao Nishijo, Masahiko Takada, Yasuhiro Go, Ken-ichi Inoue, and Tomohiro Shibata. MacaquePose: A novel "in the wild" macaque monkey pose dataset for markerless motion capture. bioRxiv, p. 2020.07.30.229989, 2020.

[26] Alexander Mathis, Pranav Mamidanna, Kevin M Cury, Taiga Abe, Venkatesh N Murthy, Mackenzie Weygandt Mathis, and Matthias Bethge. DeepLabCut: markerless pose estimation of user-defined body parts with deep learning. *Nature Neuroscience*, vol. 21, no. 9, pp. 1281–1289, Aug. 2018, doi: 10.1038/s41593-018-0209-y.

[27] Artsiom Sanakoyeu, Vasil Khalidov, Maureen S. McCarthy, Andrea Vedaldi, and Natalia Neverova. Transferring Dense Pose to Proximal Animal Classes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5232-5241, 2020.

[28] Salvador Blanco Negrete, Rollyn Labuguen, Jumpei Matsumoto, Yasuhiro Go, Ken-ichi Inoue, and Tomohiro Shibata. Multiple Monkey Pose Estimation Using OpenPose. bioRxiv, 2021, doi: 10.1101/2021.01.28.428726.

[29] Mostafa A. Nashaat, Hatem Oraby, Laura Blanco Peña, Sina Dominiak, Matthew E. Larkum, and Robert N. S. Sachdev. Pixying Behavior: A Versatile Real-Time and Post Hoc Automated Optical Tracking Method for Freely Moving and Head Fixed Animals. *Eneuro*, vol. 4, no. 1, p. ENEURO.0245-16.2017, 2017.

[30] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 770-778, 2016.

[31] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2D human pose estimation: New benchmark and state of the art analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3686-3693, 2014.

[32] Yuan Yao, Abhiraj Mohan, Eliza Bliss-Moreau, Kristine Coleman, Sienna M. Freeman, Christopher J. Machado, Jessica Raper, Jan Zimmermann, Benjamin Y. Hayden6, and Hyun Soo Park. Openmonkeychallenge: Dataset and benchmark challenges for pose tracking of non-human primates. bioRxiv, 2021, doi: 10.1101/2021.09.08.459549.

[33] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision (ECCV)*, pages 213–229. Springer, 2020.

[34] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR 2021: The Ninth International Conference on Learning Representations*, 2021.

[35] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European conference on computer vision (ECCV)*, pages 734–750, 2018.

[36] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, 9, 2015.

[37] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision (ECCV)*, pages 740–755. Springer, 2014.

[38] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *2016 IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[39] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European Conference on Computer Vision (ECCV)*, 2016.

[40] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

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

[42] Bowen Cheng, Bin Xiao, Jingdong Wang, Honghui Shi, Thomas S. Huang, and Lei Zhang. HigherHRNet: Scale-aware representation learning for bottom-up human pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[43] Zhengxiong Luo, Zhicheng Wang, Yan Huang, Liang Wang, Tieniu Tan, and Erjin Zhou. Rethinking the heatmap regression for bottom-up human pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13264–13273, 2021.

[44] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, vol. 32, pp. 8026–8037, 2019, [Online]. Available: https://arxiv.org/pdf/1912.01703.pdf.

[45] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open MMLab Detection Toolbox and Benchmark. 2019, [Online]. Available: https://arxiv.org/pdf/1906.07155.pdf

[46] MMCV Contributors. MMCV: OpenMMLab Computer Vision Foundation. 2018, version 0.7.6, https://github.com/open-mmlab/mmcv.

A Appendix

A.1 Installation

- Clone the Final project git repository to `root directory`.
- Clone the MMCV git repository to `Root/jxp042/thirdparty/`.
- In `Root/jxp042/thirdparty/mmcv` directory, run:

```
$ MMCV_WITH_OPS=1 pip install -e .
```

- Clone the MMDetection git repository to `Root/jxp042/thirdparty/`.
- In `Root/jxp042/thirdparty/mmdetection` directory, run:

```
$ pip install -e .
```

- In Root/jxp042/, run:

```
$ pip install -r requirements.txt
$ pip install -e .
```

A.2 Requirements

- Linux
- Python 3.7+
- PyTorch 1.8+
- CUDA 10.1+
- MMCV
- MMDetection

A.3 Dataset & Pre-trained model

- Download **Dataset**
  Open Monkey Challenge Dataset
- Download **COCO keypoint format annotation files**
  Annotation file for Training
  Annotation file for Validation
  Annotation file for Test
- Download **Pre-trained model**
  Optimal model (12 epoch)
- Generate monkey_dataset folder, and it has to get dataset, and annotation files. (Root/jxp042/monkey_dataset (train/, val/, cocoMonkeyTrain.json, cocoMonkeyVal.json, cocoMonkeyTest.json))

A.4 Operate Code

All code is written for executing on jxp042/

- **Inference:**

```
$ python3 demo/image_demo.py --out-file (output filename) (input file directory)
configs/petr/petr_r50_monkey_coco.py (checkpoint directory)
```

- **Training:**

```
$ bash tools/dist_train.sh configs/petr/petr_r50_monkey_coco.py 1
--work-dir monkeyDir --gpu-id 0 --resume-from (checkpoint directory)
```

- **Evaluation: Average Precision (AP)**
  Run this code to obtain mAP, APL, AP50, and AP75:

```
$ bash tools/dist_test.sh configs/petr/petr_r50_monkey_coco.py
(checkpoint directory) 1 —eval keypoints
```

- **Evaluation: Probability of Correct Keypoint (PCK)**
  After running code for AP evaluation, obtain the `predicted_result.json` file.
  Or download the already generated result file on Test set.
  Then, compile the following code to get PCK @ 0.2 results for each keypoint:

```
$ python3 pck@0.2.py
```

- **Average Precision with different Threshold**
  You can manually modify the threshold in averagePrecision.py file and compile:

```
$ python3 averagePrecision.py
```