Fine-tuning Vision Transformer-Based Model for Pose-Estimation

A Preprint

Jinxuan Liang, Yihua Zhou, Abdullah Azhar, Anjana Manjunath
University of California, Berkeley

February 7, 2024

Abstract

With the development of the Computer Vision discipline within Deep Learning, Pose Estimation tasks have seen increasing interest and growth in the past decade. Pose Estimation refers to the problem of localizing specific body parts in images and videos and encoding the spatial information into a caption relating human skeletal joints with pixel coordinates. While Pose Estimation models have largely been Convolutional Neural Network (CNN)-based, the advent of Vision Transformers (ViT) has opened new areas of research. In this study, we fine-tune a transformer-based model, BLIP, to generate accurate positional captions and explore attention mechanisms under this task. Starting with a low-complexity version of our problem, we scaled up to our task of fine-tuning BLIP for Pose Estimation. After fine-tuning BLIP on varying hyperparameters, we found that the model consistently performed well during tuning. Within a restricted threshold of 1 pixel, it retained an 81% validation accuracy and an average error of about 4 pixels.

1 Introduction

Pose Estimation tasks endeavor to estimate the spatial configuration of body parts, providing machines with a technique with which to detect figures and movements in visual media. The spatial and geometric information Pose Estimation provides has a wide range of applications and sub-tasks, resulting in Pose Estimation models taking on many forms.

1.1 Related Work

Pose Estimation for the human body is naturally called Human Pose Estimation (HPE), and the development of HPE models has long had a symbiotic relationship with action recognition and motion analysis problems. Most action recognition and motion analysis tasks need accurate estimates of human skeletal joints to benchmark each action. Simultaneously, action recognition and motion analysis prompt new niches to explore for Pose Estimation models; odd movements and body configurations challenge Pose Estimation models to generate accurate positional labels for the human skeletal joints in the image/video input [13].
Additional fields of applications of HPE models are robotics, augmented and virtual reality, and accessibility technology. HPE models can focus on specific body parts as well; hand sign language recognition centers HPE models on human hand movements and poses [12]. Applications of Pose Estimation models extend beyond HPE as well, with Animal Pose Estimation models becoming helpful in understanding animal behavior and migration patterns of endangered species [5].
In this paper, we focus on Human Pose Estimation, specifically 2-dimensional HPE models for single-person estimation. Two-dimensional (2D) Human Pose Estimation models take in a preprocessed image of a person as input and then output a list of up to sixteen key points. Each key point is associated with a human body part or joint. Like several computer vision tasks, 2D HPE models have seen significant development from the introduction of Deep Learning methods. In 2014, Google's DeepPose [15] was released, which utilized a cascade of Convolutional Neural Networks to perform pose regression [16]. Models like Hourglass built on DeepPose's cascading structure and reached higher accuracy rates [10], others pair the CNNs with a heat map method to increase accuracy [14] or integrate Graph Neural

Network (GNN) based methods [9]. Most state-of-the-art models for HPE continue to follow these strategies and are largely CNN-based. OpenPose, a common benchmark, utilizes these methods for real-time estimation [2].
Since the adaptation of Transformers for the computer vision context, Vision Transformers (ViT) have been applied to tasks that only CNNs' had once interacted with. Pose Estimation has become one of these tasks, and in the past three years, a variety of ViT architectures have been constructed. ViTPose employs a plain transformer model with a lightweight decoder for feature extraction and the pose estimation task [17]. ViT's expanded receptive field relative to a CNN allows ViT-based models to better understand spatial constraints between human joints. Several other models also apply ViTs to the Pose Estimation domain, but they continue to incorporate CNNs' in the model structure for additional feature extraction [8, 16, 18, 20].
With the development of transformer-based models, multi-modal Vision and Language models have demonstrated great performance on a variety of tasks, including HPE and action recognition tasks [6]. Since HPE is naturally an image-to-text problem, studying if Vision-Language pre-trained methods (VLP) for image-to-text can be applied to the Pose Estimation domain has been a very recent line of research. Currently, the zero-shot pre-trained model CLIP [11], is the primary model being adapted for Pose Estimation and adjacent tasks. Methodologies like LAMP (Language Assisted Multi-person Pose estimation) [4] explore its robustness in understanding human poses even under occlusion and multi-person settings.

1.2 Study Objectives

Following current research on employing multi-modal VLP methods for Pose Estimation, this paper explores whether a fine-tuned **BLIP**, a pre-trained transformer-based model, can generate positional captions for human anatomical key points. BLIP has a unique multi-modal structure composed of different modules and losses [7], and studying it in the Pose Estimation context allows us to analyze the model's internal dynamics and attention mechanisms under HPE's spatial constraints. BLIP has also demonstrated higher accuracy than CLIP in general image caption-generating tasks [7], a key reason why we are interested in using BLIP to generate positional captions.
Due to a lack of literature on employing BLIP for Pose Estimation, we approach our main Pose Estimation task first with simplified versions of it to verify that BLIP can learn poses and human anatomical key points. After checking that the model can learn these key points, we train the model to identify and caption a single-body joint using the entire input dataset.
Once we confirm that BLIP can learn and be trained for Pose Estimation, we train and fine-tune BLIP to generate a caption with all the positional information about the sixteen key points. The fine-tuning is done with three sets of parameters: learning rate, batch size, and optimizer type. Based on the optimal parameters found after twenty-five epochs, the final model was trained and validated with two evaluation metrics. Through our experiments, we see that BLIP can perform Pose Estimation and generate positional captions. The model displays relatively low training loss and high validation accuracy, considering the nature of the model and the limited compute.

2 Methodology

As HPE is primarily an image-to-text task, in this paper, the model's input is an image of a person, and the output is a caption that provides spatial information about the human anatomical key points that appear on the image. Hence, for each image in the data, the model is trained on a comprehensive dataset to generate a caption with the (x, y) pixel coordinates for each of the key points that are visible in the image.

2.1 Data

The dataset that we utilize to work with BLIP in the context of Pose Estimation is the MPII Human Pose dataset. Aggregated in 2014 from public YouTube videos, the dataset serves as one of the state-of-the-art benchmarks to evaluate HPE and action recognition models. Containing around twenty-five thousand images with over forty thousand people with annotated anatomical key points, it pairs (x, y) pixel coordinates of each anatomical key point that appears in the image with its respective MPII Joint Indices. The key points and corresponding index are listed in Table 1. These labels serve to ground truth the fine-tuning that is done [1].
Additionally, the test set includes more detailed labels with notes regarding occlusions and 3D information like torso and head angles/orientations [1].

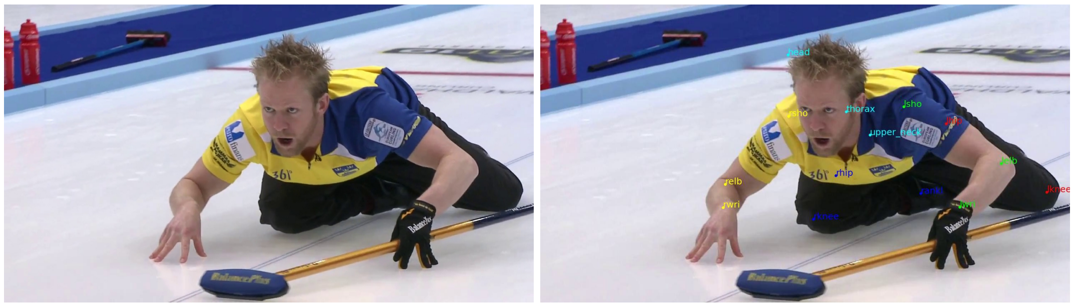

Figure 1: **Example of an image from the MPII Human Pose dataset.** The image on the left is unlabeled, and the image on the right is the same image labeled. Each visible key point is labeled with a dot and a shorthand string representation of the anatomical key point.

As the MPII Human Pose dataset stores the image identification (ID) names and annotations in a MATLAB file, to access the dataset and create a working pipeline for this experiment, we begin by processing this file. After downloading the file, the MATLAB fields and object structures are converted to a Python key/value structure using the `SciPy` Python library.
In order to refine the dataset to the scope of our question and mitigate dataset-related issues down the line, we only retain images (and their corresponding labels) that contain a single person with all sixteen key points visible. Once the appropriate subset of the MPII dataset is compiled, we split it into a training and validation set. Each subset accounts for 80% and 20%, respectively. The number of images in the training and validation sets are listed in the following table 1.

Total Number of Images	Training Size	Validation Size
1475	1170	293

Table 1: Number of images in the training and validation sets based on the 80/20 training and validation split.

2.1.1 Data Preprocessing

One key data preprocessing step for smooth BLIP training involves resizing the images from the MPII dataset. After loading the images and annotations from the MPII that only contained one person, the images were resized to 384 pixels $\times$ 384 pixels, and the ground-truth positional labels were also treated to account for any offsets. Once the images and annotations were resized and adapted, eight generation functions were employed to create eight keys with which we could approach slightly different labels depending on the nature of the captioning task. The first two keys are 'Image Name' and 'Joint Points,' which provide image identification and the sixteen positional coordinates based on the body parts visible in the image. Corresponding MPII indices and anatomical key points are listed in the table below for reference.

0	1	2	3	4	5	6	7
R. Ankle	R. Knee	R. Hip	L. Ankle	L. Knee	L. Hip	Pelvis	Thorax
8	9	10	11	12	13	14	15
Upper Neck	R. Elbow	R. Wrist	Head	R. Shoulder	L. Shoulder	L. Elbow	L. Wrist

Table 2: MPII Joint Indices and associated body parts. (Right and Left are denoted as 'R.' and 'L.', respectively.)

The other six keys involve organizing and simplifying this positional information depending on the task at hand. 'Text' provides the target output, 'Simple Text' provides the target output for one joint, 'Visibility text' and 'Simple Visibility Text' provide a binary output of whether the joint is visible in an image (if so, 1), or not visible (0). Their normalized counterparts, 'Normalized Text' and 'Normalized Simple Text,' normalize the positional (x, y) pixel values to fall between 0 and 1. For our experiments, the positional key formats we use are 'Text,' and to troubleshoot and debug, 'Simple Text.' We used all these formats for data to conduct some experiments before the official training process. We ended up using un-normalized text with coordinates round to the closest integers for simplicity.
Once preprocessed, the images and annotations are saved via the Python `Pickle` module for straightforward recon-

struction when needed. The processed data set and code to process it from the MPII Human Pose dataset for BLIP is available in section 6 of this paper.

2.2 Model

The model we are fine-tuning is BLIP, Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation [7]. As with models used for Pose Estimation tasks, this model's input is an image and the output is a caption containing the positional information of the body parts visible in the image [7]. BLIP is not a Pose Estimation model but rather a VLP method. However, BLIP's multi-modal Encoder-Decoder framework allows us to leverage it to generate synthetic positional joint captions for each image. Once the model generates captions via bootstrapping, it filters out noisy captions.

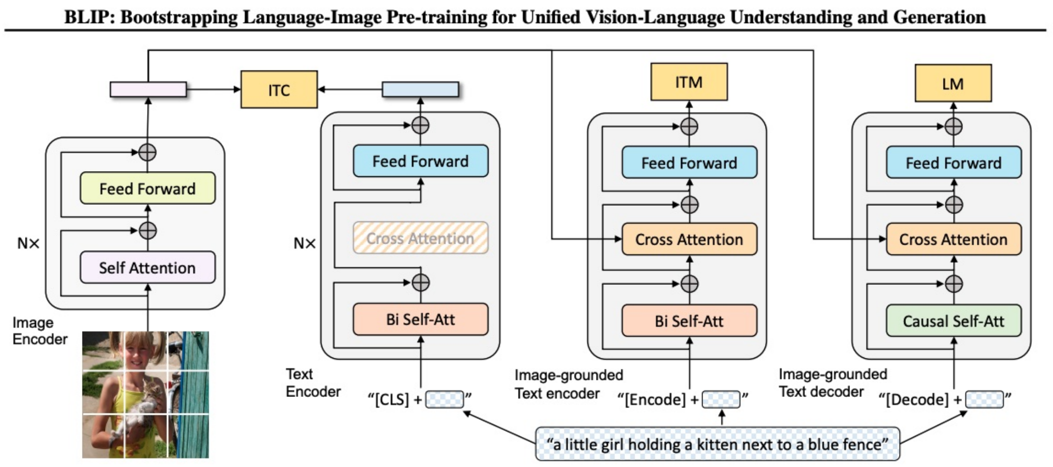
BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation
ITC
ITM
LM
Feed Forward
Self Attention
Cross Attention
Bi Self-Att
Causal Self-Att
N×
Image Encoder
Text Encoder
Image-grounded Text encoder
Image-grounded Text decoder
"[CLS] +"
"[Encode] +"
"[Decode] +"
"a little girl holding a kitten next to a blue fence"

Figure 2: **BLIP Architecture** Diagram of architectures involved from the original paper [7].

BLIP uses a visual transformer (ViT) for image encoding, and it operates with a unified model to have both understanding and generation abilities. As a result, the model can control each of the three functionalities. The **unimodal encoder**, which encodes the image and text separately, with a text encoder similar to BERT's [3]. The second functionality is the **image-grounded text encoder**; this is taken care of by including a cross-attention layer in each transformer block between the bi-directional self-attention layer and feed-forward network. Distinct from the first two functionalities, the **image-grounded text decoder** edits the image-grounded text encoder by replacing the self-attention layers with causal self-attention layers [7].
BLIP was pre-trained to optimize these three objectives jointly, so it has three losses that each activates one key functionality [7].
Image-Text Contrastive Loss (ITC), which activates the separate image and text encoders (the unimodal encoder) and aligns the ViT and text transformer feature space by assigning positive values to the image-text pairs with higher similarity scores and negative values to pairs with low similarity scores. The ITC loss BLIP uses is defined as a sum between cross-entropy losses and has a momentum encoder to account for cases where a high similarity pair ends up in the negative space [7]. Mathematically:

$$\mathcal{L}_{ITC} = \frac{1}{2}\mathbb{E}_{(I,T)\sim D}\Big(H(y_{i2t}, p_{i2t}) + H(y_{t2i}, p_{t2i})\Big)$$

where $H(\cdot,\cdot)$ is the cross-entropy loss, $y_{(\cdot,\cdot)}$ refers to the ground-truth similarity, and $p_{(\cdot,\cdot)}$ is the normalized softmax similarity between the image (i) and text (t), and vice versa [7, 19].
The **Image-Text Matching Loss** (ITM) is a binary classification task that activates the image-text encoder. It utilizes a linear layer to predict whether an image-text pair is matched or unmatched depending on its multi-modal structure. The formula for ITM loss looks like:

$$\mathcal{L}_{ITM} = \mathbb{E}_{(I,T)\sim D}\big(H(y_{itm}, p_{itm})\big)$$

[19]. The last loss function, **Language Modeling Loss** (LM), activates the decoder and is done by training the model to autoregressively maximize the likelihood of the text. Through this, a cross-entropy loss is optimized [7, 19].
The BLIP model and relevant functions are imported from Hugging Face's transformer Python library and Salesforce's

Python deep learning library for Language and Vision Research (LAVIS). The specific BLIP transformer class we utilize is called `BlipForConditionalGeneration`.

2.3 Fine-tuning Procedure

Our choice of hyper-parameters to fine-tune with respect to included batch size, learning rate, and the optimizer choice for gradient descent during model training. Due to the lack of literature on using BLIP for Pose Estimation and consequently a lack of prior knowledge on a hyperparameter 'baseline' for fine-tuning BLIP for HPE, we used the generic BLIP baselines.
We set our baseline choice of hyper-parameters to be a batch size of 4, a learning rate of 5×10^{-5}, and AdamW as the default choice of the optimizer. While we could also potentially explore the choice of β_1, the exponential decay rate for the first-moment estimates, and β_2, the exponential decay rate for the second-moment estimates, we opted to use the default values of 0.9 for β_1 and 0.999 for β_2 to avoid divergence related problems during training time.

Hyperparameter	Variable Name	Choice of Values
Batch Size	`batch_size`	$[1, 2, 4]$
Learning Rate	`lr`	$[5 \times 10^{-6}, 2 \times 10^{-5}, 5 \times 10^{-5}]$
Optimizer	`torch.optim`	`SGD, Adam, AdamW`

Table 3: Choice of Hyper-parameters

Table 3 above shows the choices of hyper-parameters that we experimented with. Ideally, we would have explored all twenty-seven possible hyperparameter combinations. However, due to our constraints with computing power, we experimented with nine choices sequentially. For each training loop, we trained our model on twenty-five epochs and saved our model check-points at the 1st, 15th, and 23rd epoch. At the end of each epoch, we save the training loss, validation accuracies, and average errors.

2.4 Evaluation

The primary evaluation metric we use is **Mean Absolute Error** (MAE). The MAE can be calculated by comparing the positional captions generated by the fine-tuned BLIP model with the positional annotations provided by the MPII dataset. More specifically, it is calculated as the average per joint offset from the ground truth.
Apart from that, we also measure the ability of the model to perform the Pose Estimation task with **Pose Estimation Accuracy**, which is defined to be the number of correctly estimated samples divided by the total number of samples. We define a correctly estimated sample as one whose predicted joint positions are close enough to the ground truth. We evaluate the accuracy with three different thresholds: 25 pixels, 5 pixels, and 1 pixel. These specify the maximum distance between the predicted points and the ground truth in terms of the number of pixels.

3 Results

Before training the model on our final goal, generating a positional caption for the sixteen anatomical key points, we carried out an initial experiment. We verify that the model has the potential to learn the positional information of the key points in the image. To check this, we randomly select several images and use the 'simple text' key to overfit the model. A demonstration of this simple task is provided in section 6. From this experiment, we assess that the model can make predictions that match the ground truth.

3.1 Simple Task (Baseline): Training for a Single Body Joint

After verifying that the model can learn what we are interested in, we scale up the size of the input dataset from five to a thousand. From this, we can confirm that the model has the ability to learn from a larger data set.
As mentioned in the Methodology section 2, the input to the model is a preprocessed image and the desired output is the pixel coordinate corresponding to a single body joint (e.g. `neck: (100, 200)`).
The model was trained using the AdamW optimizer with a learning rate of 5×10^{-5} and a batch size of four. The model was trained for a hundred and thirty epochs, with each epoch taking approximately three minutes to run. The accuracy metric is calculated every five epochs for the validation set and ten epochs for the training set. Figure 3 displays the loss and the accuracy change for training and validation:

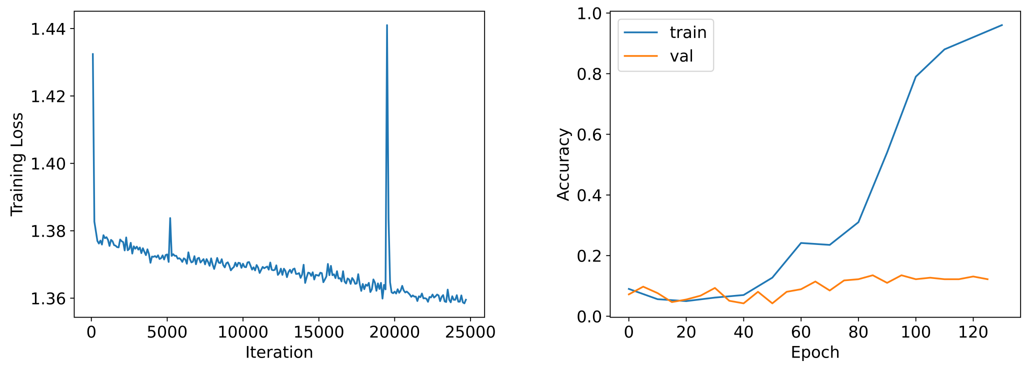

Figure 3: Comparing the loss (left) and accuracy (right) plots under the 'simple text' scheme, for both training and validation.

In the plot on the left for Figure 3, we see that the training loss generally decreases during the training process. However, we notice that there is a spike in the loss curve around the twenty-thousandth iteration. We attribute this to a divergence that occurs during training, and so after resetting the model to its nearest checkpoint, the loss continues to decrease.
One interesting observation from the other plot on the right in Figure 3 is that the training accuracy increases drastically around the 70^{th} epoch and finally goes above 90% by the final epochs. However, the accuracy for validation does not perform well and the changes are slower, with the final accuracy around 15% hence indicating a pattern of overfitting.

3.2 Complex Task: Training for all 16 Key Points

The previous warm-up and baseline section serves as the preparation for our main target - training for all key points of the body joints in one image.
As mentioned in the fine-tuning section, we select three sets of hyperparameters: learning rate, batch size, and optimizer, to carry out fine-tuning. By comparing the training loss and validation accuracy, we train our final best model based on the set of parameters with the best performance after twenty-five epochs.

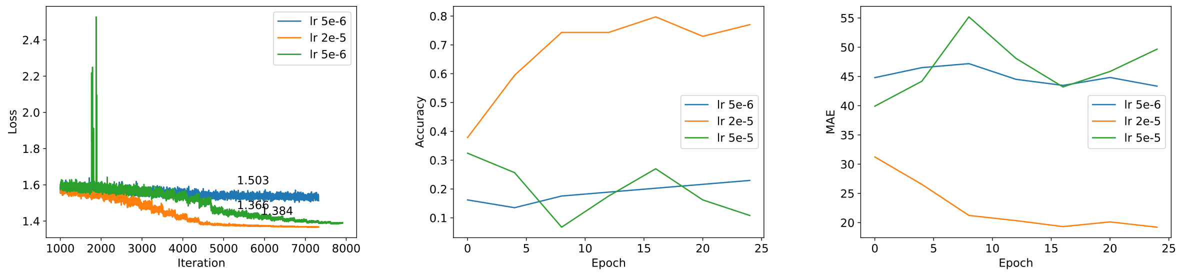

Figure 4: Comparing the training loss (left), validation accuracy (middle), and validation MAE (right) plots with different learning rates, but with AdamW and batch size 4.

As is shown in the plots, the magnitude of the learning rate plays a vital role in training the model. We adopt 2×10^{-5} to be the learning rate for our final model since it provides the best performance for both accuracy and MAE. In terms of batch size, different batch sizes do not show drastic changes in the evaluation metrics. Considering the speed of training, we adopt batch size to be 4.
For the choice of optimizer, we adopt AdamW, even though Adam's performance looks better with a learning rate of 5×10^{-5}. The combination of AdamW, a learning rate of 2×10^{-5}, and a batch size of four can provide the best results. Using the fine-tuned BLIP, we evaluate the positional captions during the early, middle, and late stages of training in Figure 7.

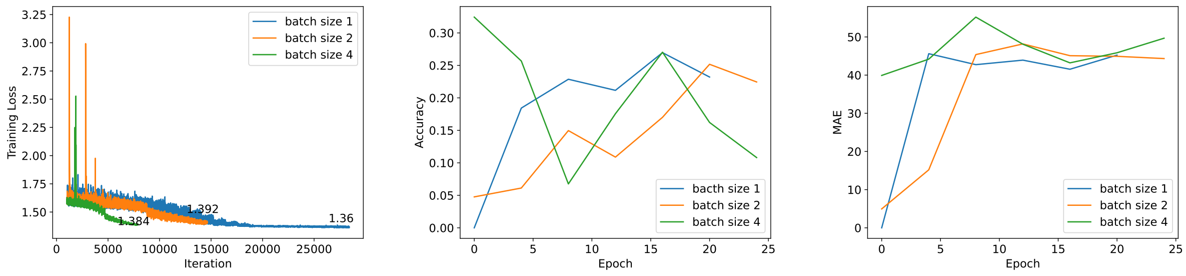

Figure 5: Comparing the training loss (left), validation accuracy (middle), and validation MAE (right) plots with different batch sizes, but with AdamW and a learning rate of 5×10^{-5}.

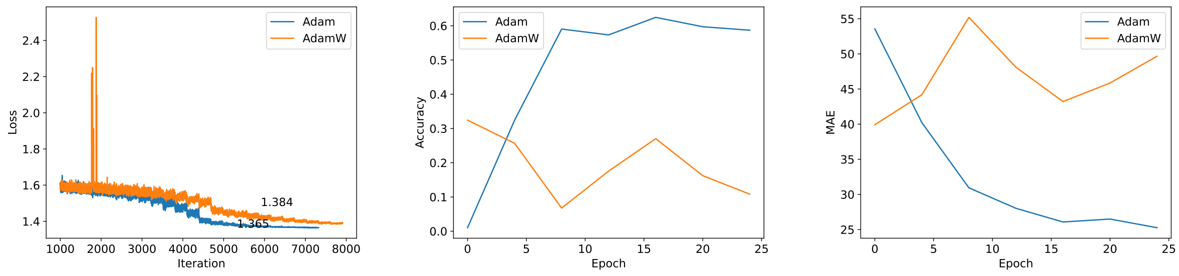

Figure 6: Comparing the training loss (left), validation accuracy (middle), and validation MAE (right) plots with two optimizers. SGD does not converge.

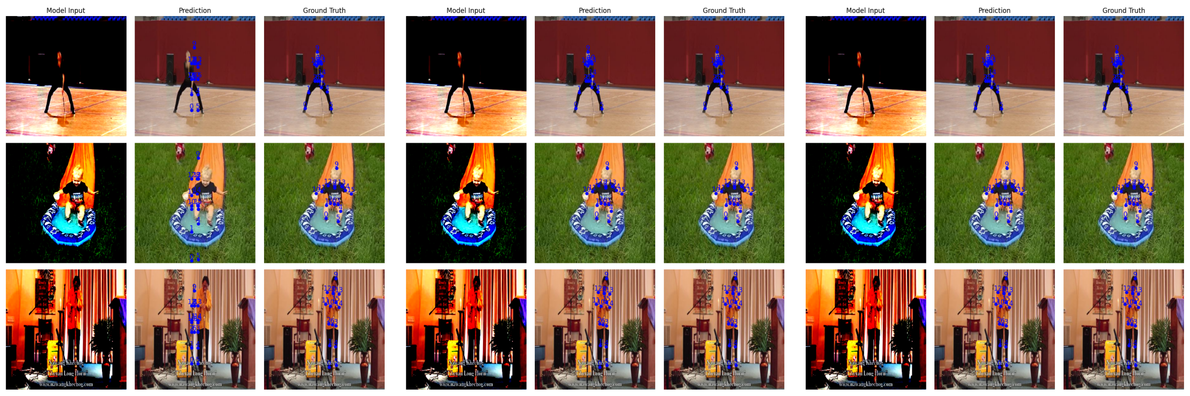

Figure 7: Caption outputs overlaid on corresponding training images during the early, middle, and late stages of training.

Based on the validation test results in Figure 10 we get after fine-tuning, we conclude that the pre-trained BLIP image captioning model can indeed do pose estimation tasks.
The 'best' model was validated with about 300 images randomly drawn from the pre-processed MPII dataset. The validation accuracy of the model is at about 92% with an MAE threshold of 25 pixels and an average estimated key point error of 5 pixels. Given such a low average error, we explored the precision of the model further by finding the accuracy at lower thresholds. Evaluating the model with a lower accuracy threshold of 1 pixel, which is the smallest offset possible in our setting, the accuracy is around 81%.

Validation Result with Different Threshold			
Threshold	Total	Correct	Accuracy
25	293	271	0.925
5	293	239	0.816
1	293	237	0.809

Table 4: Validation results and accuracy at each threshold level.

4 Discussion

4.1 Why this works?

One phenomenon that arises is the simple task's inability to generate accurate captions, while the complex version of the task attains great performance both in training and validation data sets. Although this result seems counter-intuitive, there seems to be a subtle reason behind this related to the self-attention mechanisms in the model.
The difference between the simple baseline task and the complex task is that the complex task generates a positional caption for all sixteen anatomical key points simultaneously, whereas the simple baseline task generates a caption for only one. We think that when the model is trained over a set of spatially related key points, the spatial restrictions between different joints/body parts in the human body are learned as well. This helps the model infer the coordinates using both the image embeddings and the positions of the previously generated points. Specifically, knowing the position of the right knee, left knee, and left ankle of a person in the image restricts the potential positions where a person's right ankle can be. Similarly, by knowing the position of the shoulders, the model verifies the positions of the other body joints. In order to verify our intuition about why this could work, we conducted a few experiments. Our exploration of this relationship is demonstrated through Figure 8, where we mask different parts of a person's body and analyze how it localizes the key points.

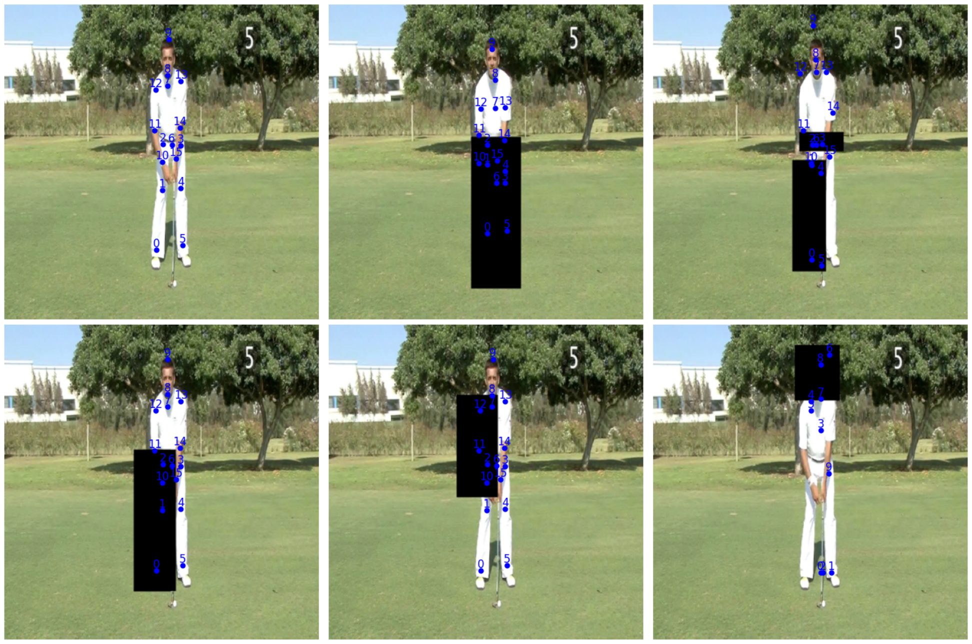

Figure 8: Example of how the Self-Attention mechanism behaves under localizing for human anatomical key points.

We see in Figure 8 that when we mask out the person's lower body, the accuracy of the positional caption worsens. Specifically, if we mask out joints number one to four, the model does even worse compared to cases where other joints are occluded. We hypothesize that this phenomenon occurs due to the self-attention layers in the image-grounded text decoder. As the joint positions are generated in sequential order, the model takes previously generated positions into account when it is creating the next one (the order of the joints/key points is in Table 2). This mechanism provides the model with the ability to learn the positional relationships and the spatial restrictions between joints. As a result, the

mechanism provides the model some flexibility in the presence of occlusions, allowing it to achieve good performance even with them.
However, we note from Figure 8 that the ‘flexibility’ is conditioned on which joint points are being masked and where they are in the sequence. If we mask out the body parts associated with the first few joints during the generation process, the model doesn’t have correct points to reference, resulting in mistakes during the next generation. In the last image in the first row of Figure 8, we see that the model incorrectly infers the spatial relationship for the upper body when substantial parts of the body key points in the positional sequence are occluded. On the other hand, the model generates relatively accurate captions even under the occlusion in the two images in the second column of Figure 8. We observe that the correctness of the first four points (MPII joint indices 0-4) generated in the caption seems to greatly affect the whole inference because blocking these points leads to bad estimations. It seems that as long as the model has some correct and relevant baseline points to reference, it is tolerant of occlusions within the image.
We further investigate how the self-attention mechanism may help with generating correct key points by plotting the attention map after the softmax layer of the text decoder, which is shown in Figure 9. These attention maps are taken from the eighth of 12 heads in the text decoder. The y-axis represents the generated tokens. The labeled rows correspond to the tokens that are converted to coordinates, whereas the unlabeled rows are descriptive words and punctuation. The brighter a row is, the more weight its associated token has. We see that, when predicting the position of the left ankle (lankl), the most activated rows are the left knee (lknee) and right ankle (rankl). We can observe that when predicting the coordinates of a joint, the model will attend to the coordinates of previously generated joints. It also seems to weigh each previously generated joint by how much they are spatially related to the joint that is about to be generated because when predicting the position of the left wrist, the most activated row is the left elbow (lelb). We think this also explains why the model does bad at predicting one joint and does well at predicting all sixteen joints. With sixteen joints, the model can always refer back to previously generated points, whereas with only one point, it can only utilize the visual information for prediction.
In short, based on our observations of the model’s behavior and our experiments, we believe the attention mechanism plays a crucial role in doing an accurate and occlusion-resistant pose estimation.

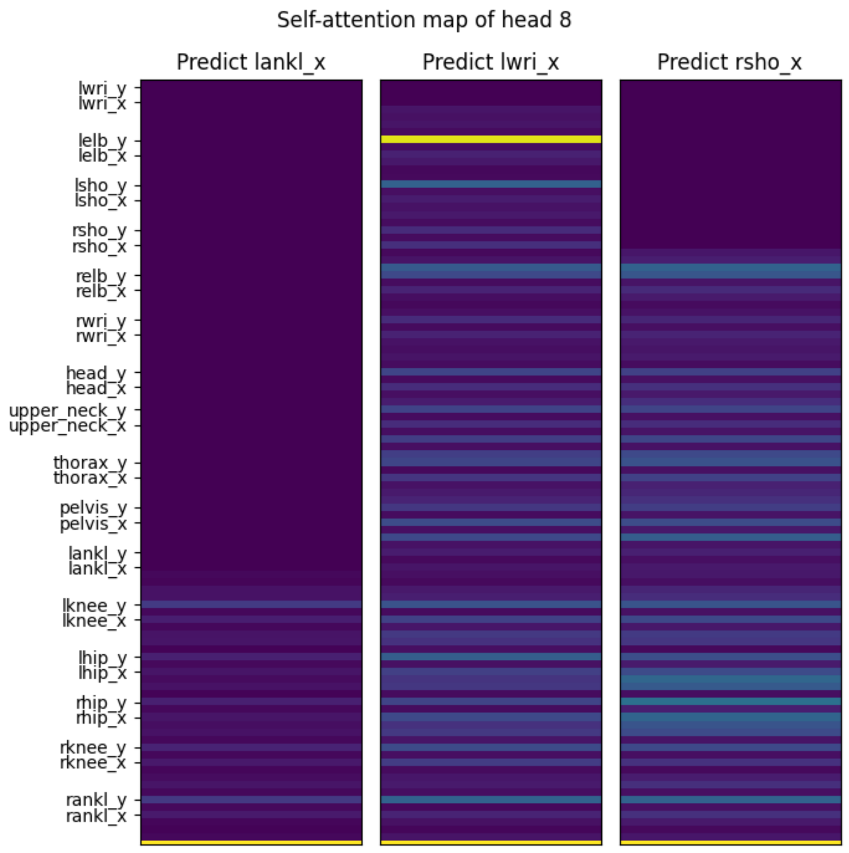

Figure 9: The self-attention map of attention-head 8 of the text decoder.

4.2 Robustness

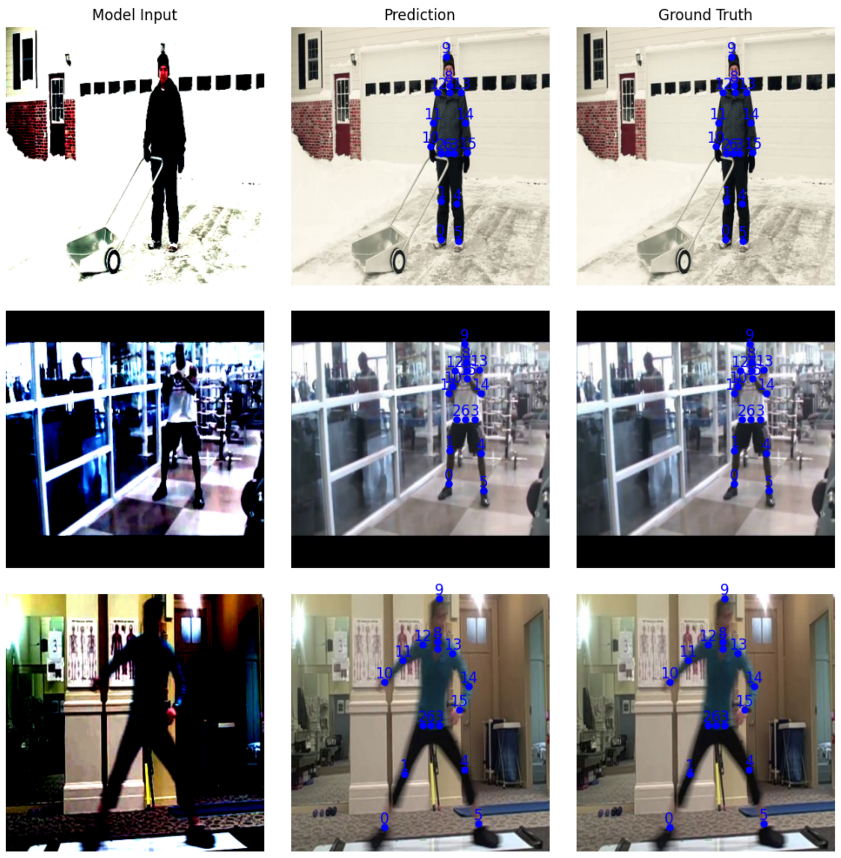

Figure 10: Sample outputs from the fine-tuned BLIP model lined up with the model input and ground truth images.

Generally, it seems that the model performed well during training and validation but poorly during test time. Even the best-generated captions would still have some key points far away from the positions of the actual body parts seen in Figure 10. However, this behavior is not all that surprising when taking into account which images we retained during training/validation dataset preprocessing. We believe that this discrepancy arose since we did not train our model with a sufficiently large dataset.
Based on our observations, the average errors seem to be dominated by the poorly estimated samples from the validation set. That is to say, if the model recognizes a pose, it will output the joint coordinates with almost 100% accuracy. If it cannot recognize a pose, then it will produce a result that deviates significantly from the ground truth.
Apart from validating the model with data drawn from the MPII dataset, we also tried to test model robustness by feeding it some images from the internet, i.e., out of distribution. The results from testing the fine-tuned model on images outside of the training/validation dataset can be seen in Figure 11.
As mentioned earlier in the paper, the original MPII data set contains more than twenty-five thousand images. After pre-processing the data to yield a training/validation set where only one person appears in each image with all body joints visible, we naturally limit the poses the model 'sees'. Also, the MPII dataset typically contains images of people in similar poses and similar environments. Hence, we can think of the training data and validation data to be coming from the same distribution. The good validation performance in these experiments tells us that the model can generalize well within the distribution defined by the subset of the MPII dataset. However, since this distribution only covers a limited number of possible human poses, the model doesn't perform as well on the out-of-distribution test data.
During training, we used about 1100 images with annotated key points from the MPII dataset, which is not a lot of data for these types of tasks. Hence, we cannot expect the model to generalize well on random images from the internet due to the model's lack of inductive bias. This seems to be one of the key limitations of a fine-tuned BLIP for Pose Estimation. To improve the robustness of the model, we will need to train the model on tens of thousands of images, including those with possible occlusions and missing labels, so that the model can learn a greater variety of human poses.

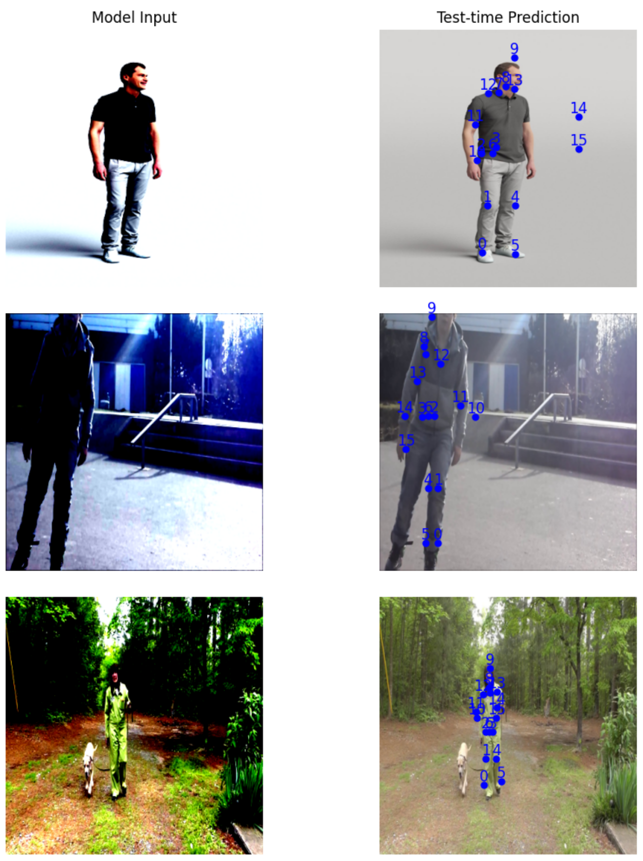

Figure 11: Results from testing fine-tuned BLIP on images from the internet (out of training/validation dataset).

4.3 Future Work

As mentioned earlier, to increase the model performance, one thing that can be done is to train the model on more data given enough computational resources. We would like to not only increase the size of the training and validation dataset and introduce a greater variety of poses but also test the model on more out-of-distribution images. The other experiment we can do further is to conduct a more detailed analysis of the model's architectures, including all the attention layers, weights, and gradients, to better understand how the attention mechanism helps with pose estimation. Although we observe how the model attends to previously generated joints when predicting new joints, more in-depth analysis is still needed to reach that conclusion.

5 Conclusion

In this project, we fine-tune BLIP, a multi-modal vision-language model in the context of a Pose Estimation. Starting from simple processed data, we first verified that the model can do pose estimation by training on a small data set and training it to generate a positional caption for one human anatomical key point. After recognizing that it can learn, we increased the training data set, and while the model performed well during training, it encountered overfitting. Finally, we trained and fine-tuned the model on a larger data set and with a more complex task, generating a caption for all sixteen key points. As a whole, the fine-tuned BLIP displays a good performance with respect to Pose Estimation when trained on a subset of the MPII dataset. Additionally, we revealed a strong connection between self-attention and generating positional captions for all sixteen human anatomical key points. This connection allows us to study how a well-placed attention mechanism can potentially allow the model to tolerate occlusions.

6 Data and Code Availability

Code to reproduce our experiments and plots can be found here: `https://github.com/RichZhou1999/cs282_final_project_codebase/tree/main`.
The link for the demo notebook can be found here: Demo on simple text.
Notice that all the notebooks are intended to be run on Kaggle. Running them in Colab or locally may cause unexpected errors.

References

[1] M. Andriluka, L. Pishchulin, P. Gehler, and B. Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[2] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[4] S. Hu, C. Zheng, Z. Zhou, C. Chen, and G. Sukthankar. Lamp: Leveraging language prompts for multi-person pose estimation, 2023.

[5] L. Jiang, C. Lee, D. Teotia, and S. Ostadabbas. Animal pose estimation: A closer look at the state-of-the-art, existing gaps and opportunities. *Computer Vision and Image Understanding*, 222:103483, 2022.

[6] C. Ju, T. Han, K. Zheng, Y. Zhang, and W. Xie. Prompting visual-language models for efficient video understanding, 2022.

[7] J. Li, D. Li, C. Xiong, and S. C. H. Hoi. BLIP: bootstrapping language-image pre-training for unified vision-language understanding and generation. *CoRR*, abs/2201.12086, 2022.

[8] Y. Li, S. Zhang, Z. Wang, S. Yang, W. Yang, S. Xia, and E. Zhou. Tokenpose: Learning keypoint tokens for human pose estimation. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 11293–11302, Los Alamitos, CA, USA, oct 2021. IEEE Computer Society.

[9] A. Newell, Z. Huang, and J. Deng. Associative embedding: End-to-end learning for joint detection and grouping, 2017.

[10] A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation, 2016.

[11] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning transferable visual models from natural language supervision, 2021.

[12] R. Rastgoo, K. Kiani, and S. Escalera. Hand sign language recognition using multi-view hand skeleton. *Expert Systems with Applications*, 150:113336, 2020.

[13] L. Song, G. Yu, J. Yuan, and Z. Liu. Human pose estimation and its application to action recognition: A survey. *Journal of Visual Communication and Image Representation*, 76:103055, 2021.

[14] J. Tompson, A. Jain, Y. LeCun, and C. Bregler. Joint training of a convolutional network and a graphical model for human pose estimation, 2014.

[15] A. Toshev and C. Szegedy. Deeppose: Human pose estimation via deep neural networks. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, June 2014.

[16] D. Wang, W. Xie, Y. Cai, X. Li, and X. Liu. Transformer-based rapid human pose estimation network. *Computers Graphics*, 116:317–326, 2023.

[17] Y. Xu, J. Zhang, Q. Zhang, and D. Tao. ViTPose: Simple vision transformer baselines for human pose estimation. In *Advances in Neural Information Processing Systems*, 2022.

[18] S. Yang, Z. Quan, M. Nie, and W. Yang. Transpose: Keypoint localization via transformer, 2021.

[19] J. Yu, Z. Wang, V. Vasudevan, L. Yeung, M. Seyedhosseini, and Y. Wu. Coca: Contrastive captioners are image-text foundation models, 2022.

[20] Y. Yuan, R. Fu, L. Huang, W. Lin, C. Zhang, X. Chen, and J. Wang. Hrformer: High-resolution vision transformer for dense predict. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 7281–7293. Curran Associates, Inc., 2021.

7 Response to Reviews

As suggested by our reviewers, we did not provide a very detailed explanation of the BLIP model's attention mechanism along with some evidence that can help support our thoughts about how the attention mechanism makes the model perform better. Hence, in this version, we add an attention map showing the weights used to generate joint coordinates. This plot clearly shows that the model is attending to previously generated joint positions, with more attention on the most relevant body parts. We also provide more explanation on what we can learn from the plot to help explain why we think the attention mechanism makes the model perform well on pose estimation tasks.
Suggestions regarding the organization and clarity of certain sections in the paper were also acted upon. Due to the length of the 'Related Work' discussion in the paper, one reviewer suggested that we create a separate subsection for it, and we did. An additional formatting-related comment that we addressed was the centering Table 4.
With respect to the content, one reviewer suggested providing more motivation as to why we are exploring BLIP for pose estimation. Hence, we've built upon the 'Related Work' and 'Study Objectives' sections to provide a more clear narrative. Apart from a brief compare and contrast between Vision Transformers and CNN-based methods, we added a paragraph in the 'Study Objectives' subsection to discuss our motivations more explicitly. A reviewer also mentioned a lack of clarity in portions of the Discussion section, and so those were edited and improved upon.
We also highlighted certain aspects of our studies - reviewers mentioned testing robustness via masking and testing on images from the internet. While we had done this for the first draft, we had not discussed the Figures that demonstrated them. Hence, we discuss Figures 8, 10, and 11 further in the Discussion portion.

Limited by the submission time, we are not able to address all the suggestions pointed out by the reviewers.
The comparison between CNN based model and our model is a meaningful issue to talk about. However, a systematic analysis can not be performed in a short time. There are a bunch of CNN models. Selecting one model arbitrarily and testing on some arbitrary data is not meaningful, since it is hard to determine whether the difference is caused by the data or the model's performance. We have some ideas to address this issue, but we would need several baseline data sets and then test our model along with other CNN-based models. Since this is too heavy for a two-day revision, we would like to solve this issue in the future. Also, since the BLIP model is not pre-trained for doing vision classification tasks like pose estimation, we are quite certain that it will not outperform the traditional CNN-based pose-estimation models, especially when we have a limited amount of data and computational resources. Therefore, we think it will not be very meaningful to compare our model with other robust pose-estimation models.
Although an ablation study would allow us to better explore BLIP, significant experimentation would take over two days to perform. The vision encoder and the text decoder each have 12 attention layers with 12 heads. Conducting an ablation study on them would take much time. Also, we did try to study the layers by plotting the cross-attention. However, we cannot find any patterns from it, which makes it a bit hard to interpret why a part of the ViT network attention layers is contributing more to the final results.
Due to the nature of the Pose Estimation positional captions, we utilize standard generative caption metrics like SPICE, and CIDEr have less meaning. Thus, we continue to work with mean absolute error and Pose Estimation Accuracy primarily.
One last suggestion that was made was to attempt fine-tuning the model with some of the Parameter Efficient Fine-Tuning techniques mentioned in class, however, we utilized a traditional hyperparameter tuning method due to precedent. The literature we referenced and 'related work' did not use PEFT methods. Since there is no literature on BLIP being used for Pose Estimation/Action Recognition tasks, we wanted to stick with certain methodologies that have worked in the Pose Estimation domain.