LwPosr: Lightweight Efficient Fine Grained Head Pose Estimation

Naina Dhingra
ETH Zurich
ndhingra@ethz.ch

Abstract

This paper presents a lightweight network for head pose estimation (HPE) task. While previous approaches rely on convolutional neural networks, the proposed network LwPosr uses mixture of depthwise separable convolutional (DSC) and transformer encoder layers which are structured in two streams and three stages to provide fine-grained regression for predicting head poses. The quantitative and qualitative demonstration is provided to show that the proposed network is able to learn head poses efficiently while using less parameter space. Extensive ablations are conducted using three open-source datasets namely 300W-LP, AFLW2000, and BIWI datasets. To our knowledge, (1) LwPosr is the lightest network proposed for estimating head poses compared to both keypoints-based and keypoints-free approaches; (2) it sets a benchmark for both overperforming the previous lightweight network on mean absolute error and on reducing number of parameters; (3) it is first of its kind to use mixture of DSCs and transformer encoders for HPE. This approach is suitable for mobile devices which require lightweight networks.

1. Introduction

Analysis of facial gestures and landmarks have been actively researched since past few decades [4, 22]. Some of the sub-topics involving facial analysis are face recognition, face detection and alignment, head pose estimation (HPE), facial emotion recognition, etc. HPE is crucial for non-verbal communication (NVC) in helping blind people, advanced driver assistance systems, behaviour analysis, in-person attention modeling, human-computer interaction, virtual/augmented reality, etc.

Depth images are used in [24, 25] to predict head poses since estimating yaw, pitch, and roll angles are a 3D vision problem. Using depth images, it is possible to achieve high prediction performance but they require RGB-D cameras. Past works on HPE have also used facial landmark for predicting 3D vectors [20, 4]. But it leads to expensive computation costs and larger models. Therefore, these techniques are not suitable to be utilized on platforms which require low computation power and have limited memory. This calls for designing lightweight networks as they are serviceable for employing them on mobile devices such as drones, movable robots, mobile phones, advanced driver assistance systems.

Using holistic approach of predicting 3D head pose from single image has shown to perform better than the keypoints-based methods [30]. Keypoints-based techniques are required to have good detection accuracy of keypoints. Furthermore, it makes the network unnecessarily complex. Also, its performance depends on the standard of the 3D head model which is required for establishing correspondences with keypoints. It is common to jointly predict facial landmarks and head poses but the primary goal in these works is generally to predict the landmarks [22, 28, 29]. So, being the secondary task, predicted head poses are not very accurate. In this work, a complete end-to-end learning based network is designed for HPE keeping in mind to use less number of parameters and at the same time to have a good performance.

Transformers [35] have recently become famous in computer vision problems considering their good performance for natural language processing applications. They have been used in image classification [14], image enhancement [44], image segmentation [38], and object detection [7]. They are being preferred for various different modalities. The positive point about their use is that they help in reducing the inductive biases that come into designing the networks. They are also shown to get an enormous parameter number reduction while maintaining the estimation performance [14]. Attention in the form of self-attention mechanism is the main building basis of transformers. It also makes it possible to learn the alignments and order which exist inside the data [35, 2]. Hence, it helps to remove the requirement for watchfully designing the inductive biases to a large extent. To our best knowledge, this is the first work using transformer encoders in a network with depthwise separable convolutions (DSCs) for designing a lightweight network for HPE.

The main contributions of this paper are given as:

- A novel architecture is proposed based on fine-grained regression using DSCs and transformer encoders. This network is able to predict head pose given by roll, yaw, and pitch angles. It consists of two streams having three stages each. Each stage consists of DSCs and/or transformer encoder and each stage has its own prediction of head poses. The weighted mean of the three head pose estimations is the final head pose prediction.

- The proposed *LwPosr* is compared with the state-of-the-art methods based on its performance and computational parameters.
- Qualitative results are illustrated to compare them with the ground truth and with FSA-Net [46]. Quantitative results are also visualized on a log-scale plot.
- Extensive ablation study is presented to understand the nuances of varying several parameters in the network.
- It is the lightest network proposed for HPE to our best knowledge. The proposed network outperforms the state-of-the-art methods which have comparable number of parameters. This work will promote the use of transformer encoders for future research in designing the algorithms for HPE with less number of parameters. This work will also have an impact on the research in similar domains to employ *LwPosr* variants which can have good estimations using less number of parameters.

2. Related Work

2.1. Head Pose Estimation

Head pose estimation can be solved in two ways. One way is to use the facial landmarks (or keypoints) along with a reference 3D head model to predict the head pose from these landmarks. The other way is to utilize the complete face appearance to predict the pose using either direct relation from image to pose or using a appearance face model.

Keypoints-Based Approaches: In these approaches, keypoints are estimated initially and then utilized for predicting the head pose. 3D vision techniques are used as in [10] where first 2D face keypoints are detected. Some techniques [5, 42, 43, 23, 12] sketch rough faces and then modify these faces incrementally to align them with the real faces using regression. Model-based techniques generate human face models using landmarks and generate correspondences on real faces using learned appearance models. In [18], an optimization technique to regress 3DMM parameters is utilized and prediction of rotation matrix is performed using a network. In [22], a heatmap-CNN is defined which refines the facial landmarks iteratively and HPE is a side product of the total task. Techniques employing deep learning to predict 3D face models using trained network perform better than the traditional approaches. These keypoint based techniques under-perform as compared to non-keypoints based approaches.

Non-keypoints-Based Approaches: Non-keypoints based methods have the benefit of not requiring the acquisition of the landmarks which makes the task less intensive. Different approaches have used different kind of modalities to solve HPE. Many recent approaches have employed a single RGB image for estimating the head poses [8, 29, 46, 30, 28, 22, 53, 50]. In [30], a ResNet50 is used along with multi-loss technique where loss consists of binned pose classification and regression for all three angles separately. In [46], a stage-wise regression is performed using a CNN based network. A new feature aggregation approach with attention mechanism is used to combine features from each stage. In [53], multi-loss approach is used to work on full range of yaws. In [50], a feature decoupling model is used to learn discriminative feature for individual pose angle using a cross-category loss. In [6], rotation matrix is used to represent three vectors for depicting the head pose. They used three sub-networks which are very similar to that used in [46] to estimate the three vectors, i.e., total nine values instead of three angles. In this paper, non-keypoints-based technique is used as it is shown to have several advantages over the keypoints-based techniques.

2.2. Transformers

Transformers have recently dominated the natural language modeling using their self-attention mechanism which has shown to have the ability to capture the global information [35, 11, 27]. They have been recently used for computer vision tasks such as image classification [14, 34], image generation [26], image enhancement [44, 9], image segmentation [36, 38], etc. Various different versions of vision transformers have been proposed and shown to perform well on large data (such as on ImageNet-22k, JFT-300M) [37, 40, 49, 34]. In [16], a hybrid neural network is developed for fast image classification where a mixture of convolution and transformer is used with new attention bias. In [39], a new network based on a mixture of convolutions and vision transformer is designed to have the positives of both convolutions and transformer. In [33] also, a mixture of CNN with a transformer is utilized for multi-instance pose estimation. Inspired by the these works, this work employs mixture of convolutions and transformer to get best characteristics of both for HPE.

3. Method

LwPosr consists of two streams and Q stages, where $Q = 3$ as shown in Figure 1. Each of the two streams extract a feature map at Q^{th} stage. The outputs from both the streams are fused together by element-wise multiplication followed by 1×1 convolutional layer (conv) to transform the features to c channels. The output of all the stages have same dimensions. The individual stage output is fed into a linear layer to predict the pose. The predicted head poses p_q's are

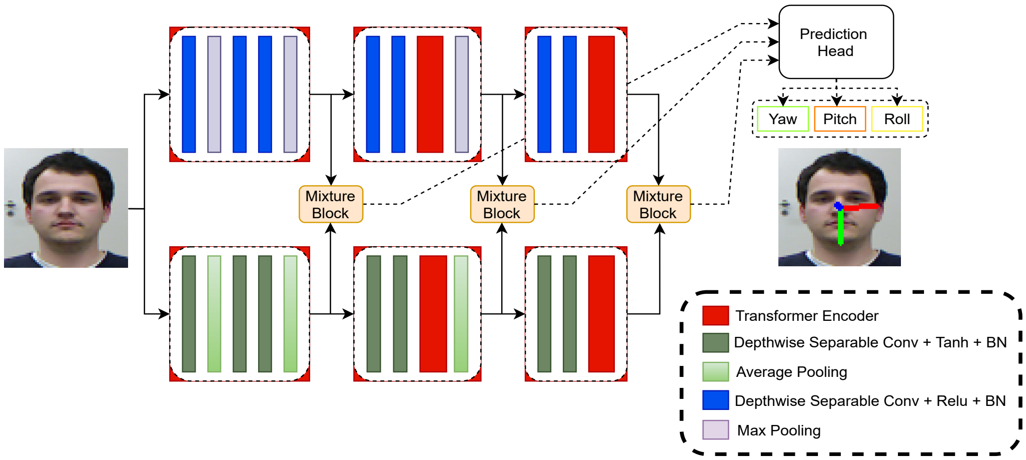


Figure 1: Architecture of *LwPosr*. It consists of two streams, each stream has three stages. Each of the three stage contributes to the estimation of pose directly in a fine-grained fashion. Mixture Block consists of element-wise multiplication of the output from two streams. Prediction head consists of 1 × 1 convolutional layer and a fully connected layer which regresses three head pose predictions from three stages. The weighted mean of all the three predictions is calculated to get final head pose estimation, i.e., yaw, pitch, and roll. The network is trained end-to-end.

combined using Eq. 1, in which P is the final predicted head pose and p_q is the q^{th} predicted pose. The weighted term α_q is found empirically and is shown in section 4 ablation study. The problem of HPE has a regression formulation. So, mean absolute error (MAE) is used as a loss function and evaluation metric similar to used in [46].

$$P = \Sigma_{q=1}^{Q} \alpha_q p_q \tag{1}$$

3.1. Single Stage Architecture:

Each of the stage consists of stacked depthwise separable filters (DSF). The technique of replacing standard convolutional filter with two consecutive layers, i.e., with depthwise and pointwise convolution filters for mobile networks is adapted from [19, 46]. DSC is a structured convolutional layer which factorizes the conventional convolutional layer into a depthwise and a pointwise (also known as 1 × 1 conv) convolutional layer. In a conventional convolution, filtering and combining of input to output features takes place in a single step. DSC breaks this single step into two individual layers, i.e., a layer for filtrating and a second layer for combining. This splitting step helps in considerably reducing the computation power required and further aids in reducing the model size. The DSC is followed by a transformer encoder. The convolution layer helps in capturing the local and global dependencies, and transformer encoder aids in understanding the global information. In [33] and [45], transformer encoder is used with convolutional based networks to understand the image-based features. Motivated by these works, in this paper, the output from convolutional layer is converted in the form of sequential features to use as an input to transformer encoder. The description of the functioning and decreased dimension of the transformer encoder is presented in section 3.3.

3.2. Reducing Number of Parameters using Depthwise Separable Filters:

In a standard convolutional layer, $F_m \times F_n \times C_{in}$ are given as input I to the layer, where $F_m \times F_n$ is the spatial size of the input feature map, C_{in} is the input channel size. Output of this convolution layer has $F'_m \times F'_n \times C_o$, where $F'_m \times F'_n$ is spatial feature size and C_o is the output channel size. The convolutional kernel $\hat{\mathbf{K}}$ in standard convolutional layer is parameterized by $F_k \times F_k \times C_{in} \times C_o$ where $F_k \times F_k$ is the spatial size of the kernel, and C_i is input channel size and C_o is output channel size. The output of a convolutional layer with stride one and padding is given by Eq. 2.

$$\hat{\mathscr{F}}_{x,y,n} = \sum_{i,j,m} \hat{\mathbf{K}}_{i,j,m,n} \cdot \mathbf{I}_{x+i-1,y+j-1,m} \tag{2}$$

The computational cost of this layer as given in Eq. 3 depends on the kernel size $F_k \times F_k$, dimension of input and output channels, i.e., C_{in} and C_o, and $F_m \times F_n$.

$$CompCost_{\text{conv}} = F_k \cdot F_k \cdot C_{in} \cdot C_o \cdot F_m \cdot F_n \tag{3}$$

DSC consists of two separate layers: (1) depthwise convolution to apply a single filter for each input depth or channel; (2) pointwise convolution to create a linear combination of the features generated by the depthwise convolutional layer. Unlike [19], in this work, batchnorm and relu are used only after the pointwise convolution layer.

The output of a depthwise convolution with one filter for each input channel is given by Eq. 4. The $\mathbf{K}$ kernel for depthwise convolutional layer has size $F_k \times F_k \times C_{in}$. The n^{th} filter of the kernel $\hat{\mathbf{K}}$ is used on the n^{th} channel to give n^{th} output feature map.

$$\mathscr{F}_{x,y,n} = \sum_{i,j} \mathbf{K}_{i,j,n} \cdot \mathbf{I}_{x+i-1,y+j-1,n} \tag{4}$$

The corresponding computational cost for the depthwise convolutional (DC) layer is .

$$CompCost_{\mathrm{DC}} = F_k \cdot F_k \cdot C_{in} \cdot F_m \cdot F_n \tag{5}$$

This depthwise convolution provides with the filtration of the input channels. But the combination of these filtered output does not take place by the depthwise convolutional layer. Therefore, a pointwise convolutional layer is used to combine them. It linearly combines the output features from the depthwise convolutional layer. Both the layers, i.e., depthwise and pointwise convolution form a DSC. The computation cost of the DSC is given by Eq. 6

$$CompCost_{\mathrm{DSC}} = F_k \cdot F_k \cdot C_in \cdot F_m \cdot F_n + C_{in} \cdot C_o \cdot F_m \cdot F_n \tag{6}$$

By comparing both the computational costs given in Eq. 3 and 6, it results into Eq. 7 which shows that DSC have lower number of parameters than conventional convolutional layer.

$$\frac{F_k \cdot F_k \cdot C_{in} \cdot F_m \cdot F_n + C_{in} \cdot C_o \cdot F_m \cdot F_n}{F_k \cdot F_k \cdot C_{in} \cdot C_o \cdot F_m \cdot F_n} = \frac{1}{C_o} + \frac{1}{F_k^2} \tag{7}$$

This work also uses 3×3 DSCs which use approximately 8 to 9 times lower computational power than conventional convolutional layer as shown in [19].

3.3. Transformer Encoder:

Each stage structure except the first one has a transformer encoder, i.e., second and third stage in both the streams. Transformer encoder follows a very similar structure as proposed in [35]. To reform the output from the DSC, a reshaper is used. It converts the output features into a sequential form which can be used as an input to transformer encoder. The reshaper performs this operation by multiplying the height and width of the features and then performs permutation on them. Consider the output features from DSC layers $\mathscr{F} \in \mathbb{R}^{B \times C \times H \times W}$, where B is the batch-size, C is the number of channels, H stands for the height, and W for the width. Reshaper transforms it to $\mathscr{F} \in \mathbb{R}^{B \times A \times C}$, where $A = H \times W$. This output $\mathscr{F}$ is in sequential form and is apt to be fed into a transformer encoder.

Transformer encoder uses multi-head self attention mechanism. It gets a sequential input $\mathscr{F} \in \mathbb{R}^{B \times A \times C}$. The inputs are used to calculate values $V \in \mathbb{R}^{A \times C}$, keys $K \in \mathbb{R}^{A \times C}$, and queries $Q \in \mathbb{R}^{A \times C}$ as shown in Eq. 8.

$$Q_h = \mathscr{F} * W_h^Q \text{ and } K_h = \mathscr{F} * W_h^K \text{ and } V_h = \mathscr{F} * W_h^V \tag{8}$$

Here, h represents used attention head, and $W_h^Q \in \mathbb{R}^{C \times C_k}$, $W_h^K \in \mathbb{R}^{C \times C_k}$, and $W_h^V \in \mathbb{R}^{C \times C_v}$ represent parameter matrices. In this work, position embeddings are used in transformer encoder. Without position embeddings, transformer encoder would be permutation-invariant, hence would not consider the spatial formation of the features. The output Z_i of the scaled dot-product attention head is given by Eq 9

$$Z_h = softmax\left(\frac{Q_h * K_h^T}{\sqrt{C_k}}\right) \tag{9}$$

The output Z_h from attention heads are concatenated to form

$$Z = concatenate(Z_1, Z_2, ..., Z_{n_{heads}}). \tag{10}$$

The Add & Norm layer is used for adding the residual connection of the input with the output of previous multi-head attention and for normalization by calculating the mean and variance of the cumulative inputs using the technique given in [1]. It also alleviates the problem of 'covariate shift'. This is followed by a feed-forward layer and activation function (see Section 4.6 for results on two different activation functions). It is followed by another Add & Norm layer. For detailed working of transformer, refer to [35].

3.4. Architecture Details:

There are two streams in the network as used in [47, 48, 46]. Similar to [46], two basic DSC blocks are used: $DSC_r(c) \equiv [DSC(3 \times 3, c) - BN - ReLU]$ and $DSC_t(c) \equiv [DSC(3 \times 3, c) - BN - Tanh]$, where c is a parameter and BN stands for batch-normalization. $TransEnc$ denotes transformer encoder with 8 heads, 3 encoder layers, 32 number of expected features in the encoder (denoted as k), 64 feedforward dimension. The first stream has $[DSC_r(16) - AvgPool(2 \times 2) - DSC_r(32) - DSC_r(32) - AvgPool(2 \times 2)] - [DSC_r(32) - DSC_r(32) - TransEnc - AvgPool(2 \times 2)] - [DSC_r(32) - DSC_r(32) - TransEnc]$. A stage is formed by each pair of square brackets. The second stream has same structure as the first stream except that it uses DSC_t instead of DSC_r and $MaxPool$ instead of $AvgPool$. Mixture block consists of element-wise multiplication of the features from the two streams. Prediction head consists of 1×1 convolutional layer which lowers the number of channels from 32 to 16 (first stage has AvgPool after 1×1 conv). The flattened array is then fed into linear layer having three output units, one each for yaw, pitch, and roll. The final head pose is calculated using the three predicted head poses (one from each stage) by taking their weighted mean as given in Eq. 1.

4. Experiments

This section describes the used datasets, implementation details, evaluation protocols, quantitative and qualitative results, comparison with the state-of-the-art, and extensive ablation study.

4.1. Datasets

Three open-source popular datasets are adopted in this work, i.e., the 300W-LP [54], AFLW2000 [55], and BIWI [15] datasets.

300W-LP Dataset: It is acquired from the 300W dataset [31] which is an amalgamation of several sub-datasets. Some of the examples for sub-datasets are: AFW [56], LFPW [3], HELEN [52], IBUG [31], etc. Face profiling is used with 3D meshing to expand 300W dataset to produce 61,225 image samples for large poses [54]. Flipping is also used to further expand the dataset images to 122,450 synthesized image samples. Therefore, the resulting synthesized dataset by [54] has its name as the 300W across large poses (300W-LP).

AFLW2000 Dataset: It has first 2,000 images present in AFLW [21] dataset. It includes ground-truth 3D faces along with 68 landmarks for each face. These 3D faces have a large variations in their illuminations, expressions, and poses.

BIWI Dataset: It has total of 15,678 frames generated out of 24 videos having 20 different subjects. These videos are taken in the controlled indoor environment. This dataset does not inculcate human heads bounding boxes. So, in this work, the human faces are detected and cropped to get the bounding box of the face using MTCNN [51] similar to [6, 46],

4.2. Experiment Protocols

Protocol *P1*: Similar settings are followed as in [46, 13, 6] to have fair comparison of the results. Training is performed on synthetic 300W-LP dataset. Testing is performed on real-world datasets: (1) AFLW2000, (2) BIWI datasets. Similar to [46, 13, 6], this protocol does not utilize tracking and uses rotation angles between [-99°, 99°] with MTCNN [51] face detection algorithm.

Protocol *P2*: It uses BIWI dataset which is divided into 70% training and 30% testing dataset, same as performed by [6, 46]. MTCNN is used to detect the faces with empirical tracking method so that all the faces get detected successfully. This protocol has been used with several different modalities in the literature. In this work, single RGB image is used.

4.3. Implementation Details:

Pytorch library is used for the implementation of the LwPosr. The implementation parameters are kept same as described by [46, 6] so that the comparison of results is fair. During training, random cropping and scaling is used by a factor of 0.8-1.2. The training is carried on for 90 epochs using an Adam optimizer and with initial learning rate of 0.001. This learning rate is reduced after 30 steps by a factor of 0.1. Batch-size used is 16 images for protocol P_1 and 8 images for protocol P_2. The experimental implementation was carried on an GeForce RTX 2080 Ti GPU (training time is 8 hours).

4.4. Quantitative Results

LwPosr is compared to existing methods (with both lightweight and heavier techniques) for HPE on open-source benchmark datasets. Table 1 and 2 illustrate the comparison of the number of parameters and MAE for the compared frameworks. They are as follows. (1) 3DDFA [54] uses a cascaded CNN network which tends to fit 3D model to a single image. It is described to perform well on occlusions. (2) KEPLER [22] uses Heatmap-CNN to understand the local and global features to detect keypoints for face alignment and HPE is a by-product task. (3) Dlib [20] is a standard well used library for faces for applications of HPE, face detection, keypoints prediction. (4) FAN [4] utilizes landmark localization network combined with residual block forming a multi-stage network with multi-scale features. (5) Hopenet [30] is ResNet based fine-grained structure using classification and regression loss. (6) Shao [32] works on the adjustment of margins of detected bounding box for faces using a CNN architecture. (7) SSR-Net-MD [48] uses soft stage-wise regression. (8) FSA-Caps-Fusion [46] learns feature aggregation for fine-grained regression. (9) TriNet [6] uses ResNet along with feaure agregagtion as used in [46] for three different streams. It is a memory intensive network. (10) WHENet [53] is also one of the low memory networks (having 4.4×10^6 number of parameters) which uses efficientNet (lightweight backbone) and uses multi loss (classification and regression loss). (11) VGG16 [17] and VGG16+RNN [17] are highly memory expensive models with CNN and CNN with RNN respectively, using Bayesian filters analysis. (12) DeepHeadPose [25] uses depth images (low resolution) and uses regression and classification for each angle prediction. (13) Martin [24] too uses depth images and performs registration to a 3D head model.

Protocol P_1: To our best knowledge, *LwPosr* is the lowest memory requiring network with just 0.159×10^6 parameters. TriNet has lowest MAE on AFLW2000 dataset with a difference of 0.68 on MAE but it has $\sim$ 163.5 times more number of parameters. WHENet has least MAE on BIWI dataset with a difference of 0.24 but it requires 27.7 times more number of parameters than *LwPosr*. In the literature, SSR-Net-MD has the least number of parameters in the network. *LwPosr* has outperformed it on MAE on both the datasets by 0.6 on BIWI and 0.66 on AFLW2000 dataset simultaneously reducing the number of parameters. Hence, *LwPosr* has significantly improved the benchmark for networks with least number of parameters for HPE to our best knowledge and as is also evident in Table 1. *LwPosr* α has performed better

Table 1: Results for Protocol P_1 trained on 300W-LP dataset, tested on BIWI [15] (left) and AFLW2000 [54] (right) dataset

		BIWI				AFLW2000			
Method	Param 10^6	Yaw	Pitch	Roll	MAE	Yaw	Pitch	Roll	MAE
3DDFA [54]	-	36.20	12.30	8.78	19.10	5.40	8.53	8.25	7.39
KEPLER [22]	-	8.80	17.3	16.2	13.9	-	-	-	-
Dlib (68 points) [20]	6-24	16.8	13.8	6.19	12.2	23.1	13.6	10.5	15.8
FAN (12 points) [4]	~36.6	6.36	12.3	8.71	9.12	8.53	7.48	7.63	7.88
Hopenet(a = 1) [30]	23.9	4.81	6.61	3.27	4.90	6.92	6.64	5.67	6.41
Hopenet(a = 2) [30]	23.9	5.12	6.98	3.39	5.18	6.47	6.56	5.44	6.16
Shao [32]	24.6	4.59	7.25	6.15	6.00	5.07	6.37	4.99	5.48
SSR-Net-MD [48]	0.2	4.49	6.31	3.61	4.65	5.14	7.09	5.89	6.01
FSA-Caps-Fusion [46]	1.2	4.27	4.96	2.76	4.00	4.50	6.08	4.64	5.07
TriNet [6]	~26	4.11	4.76	3.05	3.97	4.04	5.78	4.20	4.67
WHENet [53]	4.4	3.99	4.39	3.06	3.81	5.11	6.24	4.92	5.42
EVA-GCN [41]	~3.3	4.46	5.34	4.11	4.64	4.01	4.78	2.98	3.92
LwPosr	**0.15**	4.11	4.87	3.19	4.05	4.80	6.38	4.88	5.35
LwPosr α	**0.15**	4.41	5.11	3.24	4.25	4.44	6.06	4.35	4.95

Table 2: Results for Protocol P_2 using BIWI dataset as 70% training and 30% testing data (using only RGB images).

	Param 10^6	Yaw	Pitch	Roll	MAE
RGB-based					
DeepHeadPose [25]	-	5.67	5.18	-	-
SSR-Net-MD [48]	0.2	4.24	4.35	4.19	4.26
VGG16 [17]	~138	3.91	4.03	3.03	3.66
FSA-Caps-Fusion [46]	1.2	2.89	4.29	3.60	3.60
TriNet	~ 26	2.99	3.04	2.44	2.80
LwPosr	**0.15**	3.62	4.65	3.78	4.01
RGB+Depth					
DeepHeadPose [25]	-	5.32	4.76	-	-
Martin [24]	-	3.6	2.5	2.6	2.9
RGB+Time					
VGG16+RNN [17]	~138+	3.14	3.48	2.6	3.07

than *LwPosr* on AFLW2000 dataset. The only difference is that *LwPosr* α uses Gelu activation function in transformer encoders instead of Relu.

Protocol P_2: It is evident from Table 2 that *LwPosr* has the least number of parameters compared to all other networks in this protocol (to our best knowledge). It has improved the performance over SSR-Net-MD (previous benchmark for least number of parameters) by 0.25 on MAE. TriNet performs quite efficiently on P_2 but it has $\sim$ 163.5 times more parameters ($\sim 26 \times 10^6$ as compared to 0.159×10^6). VGG16 and VGG16+RNN has 870 times more number of parameters than *LwPosr*. Thus, even though these networks have slightly better performance than *LwPosr* on P_2, these are difficult to be used on small mobile devices.

4.5. Visualization of Results

Figure 2a shows a plot with MAE and number of parameters plotted for the compared HPE techniques. It is evident that *LwPosr* dominates the other networks (TriNet, WHENet, FSA-Caps-Fusion) in parameter number domain and is sub-dominant in MAE. Rest of the networks have more number of parameters and also perform lower than the proposed architecture.

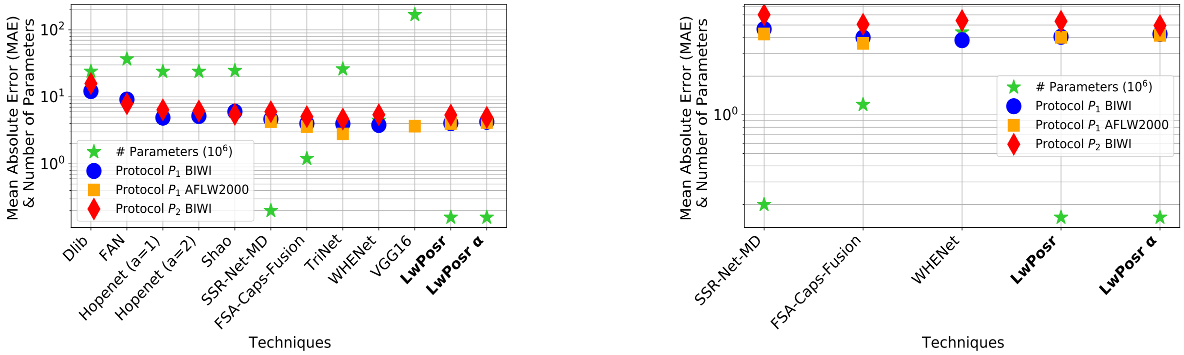
Figure 2b shows the networks with number of parameters less than 5×10^6. SSR-Net-MD which has the least number of parameters in literature has 0.41×10^6 more paramters than *LwPosr*. FSA-Caps-Fusion has almost 10 times more number of parameters than *LwPosr*.

Figure 3 shows the qualitative examples of the head pose illustration on the test images of AFLW2000 dataset using protocol P_1. The illustrative images in Figure 3 also show the comparison with FSA-Net results. It is seen that *LwPosr* performs equally well or even better on the given examples. For instance, in the first column the blue line is better corresponding to the ground truth as compared to the FSA-Net results. Results are added for BIWI dataset in supplementary material.

4.6. Ablation Study

All the results corresponding to ablations performed using Protocol P_1 are concisely described in Table 3 (parameter being studied is varied and other parameters are kept constant to 3 encoder layers, 4 heads, relu activation function, sine embedding, 0.001 learning rate, $[0.5, 0.5, 2]/3$ weights in the weighted mean, MAE loss, 3 stages).

Weights in the Weighted Mean: Eq. 1 shows that the outputs from all the mixture models have own individual head pose predictions. These predictions are combined by weighted means. The experiments performed for $\alpha_1, \alpha_2, \alpha_3$ for stages 1, 2, and 3, respectively, having sets of values [0.5, 0.5, 2]/3, [2, 0.5, 0.5]/3, and [1, 1, 1]/3 show that the set of values [0.5, 0.5, 2]/3 performs better than other sets. It is intuitive that the last stage rectifies the features learnt by the initial stages. So, the performance is better when more weight is given to the last stage. Investigation is also



(a) All Compared Techniques

(b) Techniques with Number of Parameters $< 5 \times 10^6$

Figure 2: Visualization of Mean Absolute Error (MAE) and number of parameters for the compared techniques of head pose estimation. It is evident that *LwPosr* has the least number of parameters compared to other techniques. Its performance is also significantly better as compared to most of the networks (surpassing the state-of-the-art memory efficient network, i.e. SSR-Net-MD)

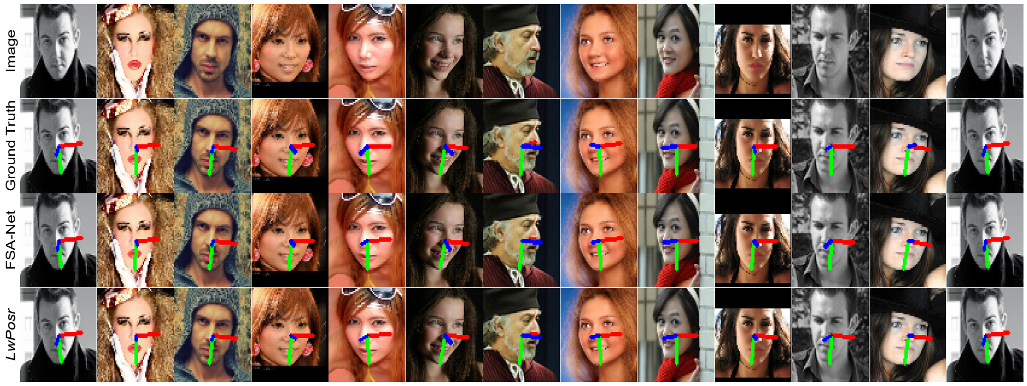


Figure 3: Qualitative visualization of the head pose estimation illustration for ground-truth, FSA-Net [46], *LwPosr*

performed on learnable set (where $\alpha_1, \alpha_2, \alpha_3$ are learnable parameters). But it performs worse than the fixed sets.

Learning Rate: Initial learning rates are investigated for values of 0.01, 0.001, 0.0001. It is evident from Table 3 that training gets stuck in local mimina with the learning rate of 0.01. The learning rate of 0.001 performs better than the other two.

Activation Function in Transformer Encoder: Two activation functions are checked for in transformer encoder, i.e., Relu and Gelu functions. For BIWI dataset, Relu shows to perform better than Gelu function and for AFLW2000 dataset Gelu performs better than Relu function in the proposed network.

Position Embeddings: The 2D spatial structure is learnt using position embedding in transformer. Three tests are performed: (1) with no position embedding, (2) learnable embedding, and (3) sine embedding. Intuitively, network with no position embedding performs the worst. It makes the transformer encoder permutation-invariant. Sine embedding performs better than learnable embedding for *LwPosr* (details of sine-embedding are mentioned in Supplementary material).

Size Scaling of Transformer Encoder: Number of encoder layers are varied to check their effect in *LwPosr* on the HPE. The experiments are performed for one to six encoder layers. It is seen that the optimized number of encoder layers is three. If the number of encoders is increased further than three, then the MAE increases.

Number of Heads in Transformer Encoder: Experiment by varying the number of heads in transformer encoder are performed. It is seen that the performance increases by increasing the number of heads until eight heads and then starts declining.

Loss Functions: Orthogonality (Ortho) loss as used in [6] is investigated by using Ortho loss and MAE together for optimization. It is seen in the experiments that using Ortho loss in addition to MAE does not improve the performance in the network (details on Ortho loss in supplementary material).

Table 3: Ablation Study on *LwPosr* network. The ablations are performed on varying the parameters of transformer encoder, changing hyper-parameters, using different weighted mean for final prediction.

		BIWI				AFLW			
	Param 10^6	Yaw	Pitch	Roll	MAE	Yaw	Pitch	Roll	MAE
No. of Encoder Layers in Transformer Encoder									
1	0.091	4.66	5.29	3.73	4.56	5.04	6.33	4.99	5.45
2	0.126	4.50	5.07	3.19	4.25	4.97	6.39	5.00	5.45
3	0.159	4.11	4.87	3.19	**4.05**	4.80	6.38	4.88	**5.35**
4	0.194	4.40	5.29	3.33	4.34	4.89	6.46	5.03	5.46
5	0.228	9.87	9.90	5.96	8.57	12.39	8.71	9.43	10.17
6	0.262	10.15	10.21	5.94	8.77	12.24	8.75	9.27	10.09
No. of Heads in Transformer Encoder									
1	0.159	17.62	21.06	6.97	15.22	23.25	14.47	12.53	16.75
2	0.159	5.07	7.08	4.21	5.45	6.88	7.70	6.96	7.18
4	0.159	5.28	5.95	4.37	5.2	6.55	7.80	6.68	6.99
8	0.159	4.11	4.87	3.19	**4.05**	4.80	6.38	4.88	**5.35**
16	0.159	4.69	7.76	4.18	5.54	7.25	8.35	7.52	7.80
Activations in Transformer Encoder									
Relu	0.159	4.11	4.87	3.19	**4.05**	4.80	6.38	4.88	5.35
Gelu	0.159	4.41	5.11	3.24	4.25	4.44	6.06	4.35	**4.95**
Position Embeddings in Transformer Encoder									
No Embedding	0.139	4.34	5.15	3.31	4.26	5.35	6.39	5.30	5.69
Learnable	0.159	4.71	5.73	3.42	4.62	4.86	6.38	4.99	5.41
Sine	0.159	4.11	4.87	3.19	**4.05**	4.80	6.38	4.88	**5.35**
Learning Rate									
0.01	0.159	10.27	9.57	5.67	8.50	12.30	8.77	9.41	10.16
0.001	0.159	4.11	4.87	3.19	**4.05**	4.80	6.38	4.88	**5.35**
0.0001	0.159	5.16	6.43	3.87	5.15	6.48	7.57	6.52	6.86
Weights in the Weighted Mean									
$[0.5, 0.5, 2]/3$	0.159	4.11	4.87	3.19	**4.05**	4.80	6.38	4.88	5.35
$[2, 0.5, 0.5]/3$	0.159	4.26	5.10	3.40	4.25	4.85	6.21	4.79	**5.28**
$[1, 1, 1]/3$	0.159	4.50	5.46	3.08	4.34	4.93	6.33	5.10	5.45
Learnable	0.159	4.50	5.37	3.20	4.35	4.74	6.63	5.18	5.52
Loss Functions									
MAE	0.159	4.11	4.87	3.19	**4.05**	4.80	6.38	4.88	5.35
MAE + Ortho loss	0.159	4.29	5.24	3.38	4.30	4.84	6.31	4.77	**5.31**
No. of stages in *LwPosr*									
3	0.159	4.11	4.87	3.19	**4.05**	4.80	6.38	4.88	5.35
4	0.221	4.69	5.63	3.51	4.61	5.22	6.62	4.97	**5.27**

5. Different Stages in *LwPosr* Network

We implemented 4 stages in *LwPosr*, where $Q = 4$ instead of $Q = 3$ (as mentioned in section 3). Table 3 illustrates the number of parameters and results for the corresponding stages. It shows that increasing the stages increases the number of parameters which have to be learnt. Hence, the network size increases. The quantitative results on average do not change. So, network with three stages is more optimized in terms of number of parameters and evaluation results.

6. Conclusion

This paper presented a novel architecture *LwPosr* for head pose predictions. In contrast to previous work, this model is able to predict poses efficiently by taking a very small number of parameters and memory. A key insight is that transformer encoder helps in learning the global information. Using transformer encoder in combination with the depth-wise separable convolutions gets the best of both entities and helps to learn spatial as well as global features. Furthermore, transformer encoders take less number of parameters, hence small network size. It is demonstrated that proposed model outperforms previous memory efficient approaches on 1) lowering mean absolute error and 2) decreasing the number of parameters. It is believed that in future crucial concepts of the presented work can be used in other domains and tasks that have a similar problem statement.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[3] Peter N Belhumeur, David W Jacobs, David J Kriegman, and Neeraj Kumar. Localizing parts of faces using a consensus of exemplars. *IEEE transactions on pattern analysis and machine intelligence*, 35(12):2930–2940, 2013.

[4] Adrian Bulat and Georgios Tzimiropoulos. How far are we from solving the 2d & 3d face alignment problem?(and a dataset of 230,000 3d facial landmarks). In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1021–1030, 2017.

[5] Xudong Cao, Yichen Wei, Fang Wen, and Jian Sun. Face alignment by explicit shape regression. *International Journal of Computer Vision*, 107(2):177–190, 2014.

[6] Zhiwen Cao, Zongcheng Chu, Dongfang Liu, and Yingjie Chen. A vector-based representation to enhance head pose estimation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1188–1197, 2021.

[7] 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*, pages 213–229. Springer, 2020.

[8] Feng-Ju Chang, Anh Tuan Tran, Tal Hassner, Iacopo Masi, Ram Nevatia, and Gerard Medioni. Faceposenet: Making a case for landmark-free face alignment. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 1599–1608, 2017.

[9] Hanting Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yiping Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, Chao Xu, and Wen Gao. Pre-trained image processing transformer. *arXiv preprint arXiv:2012.00364*, 2020.

[10] Daniel F DeMenthon and Larry S Davis. Model-based object pose in 25 lines of code. *International journal of computer vision*, 15(1-2):123–141, 1995.

[11] 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.

[12] Piotr Dollár, Peter Welinder, and Pietro Perona. Cascaded pose regression. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1078–1085. IEEE, 2010.

[13] Bardia Doosti, Shujon Naha, Majid Mirbagheri, and David J Crandall. Hope-net: A graph-based model for hand-object pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6608–6617, 2020.

[14] 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.

[15] Gabriele Fanelli, Matthias Dantone, Juergen Gall, Andrea Fossati, and Luc Van Gool. Random forests for real time 3d face analysis. *International journal of computer vision*, 101(3):437–458, 2013.

[16] Ben Graham, Alaaeldin El-Nouby, Hugo Touvron, Pierre Stock, Armand Joulin, Hervé Jégou, and Matthijs Douze. Levit: a vision transformer in convnet's clothing for faster inference. *arXiv preprint arXiv:2104.01136*, 2021.

[17] Jinwei Gu, Xiaodong Yang, Shalini De Mello, and Jan Kautz. Dynamic facial analysis: From bayesian filtering to recurrent neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1548–1557, 2017.

[18] Jianzhu Guo, Xiangyu Zhu, Yang Yang, Fan Yang, Zhen Lei, and Stan Z Li. Towards fast, accurate and stable 3d dense face alignment. *arXiv preprint arXiv:2009.09960*, 2020.

[19] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[20] Vahid Kazemi and Josephine Sullivan. One millisecond face alignment with an ensemble of regression trees. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1867–1874, 2014.

[21] Martin Koestinger, Paul Wohlhart, Peter M Roth, and Horst Bischof. Annotated facial landmarks in the wild: A large-scale, real-world database for facial landmark localization. In *2011 IEEE international conference on computer vision workshops (ICCV workshops)*, pages 2144–2151. IEEE, 2011.

[22] Amit Kumar, Azadeh Alavi, and Rama Chellappa. Kepler: Keypoint and pose estimation of unconstrained faces by learning efficient h-cnn regressors. In *2017 12th ieee international conference on automatic face & gesture recognition (fg 2017)*, pages 258–265. IEEE, 2017.

[23] Donghoon Lee, Hyunsin Park, and Chang D Yoo. Face alignment using cascade gaussian process regression trees. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4204–4212, 2015.

[24] Manuel Martin, Florian Van De Camp, and Rainer Stiefelhagen. Real time head model creation and head pose estimation on consumer depth cameras. In *2014 2nd International Conference on 3D Vision*, volume 1, pages 641–648. IEEE, 2014.

[25] Sankha S Mukherjee and Neil Martin Robertson. Deep head pose: Gaze-direction estimation in multimodal video. *IEEE Transactions on Multimedia*, 17(11):2094–2107, 2015.

[26] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *International Conference on Machine Learning*, pages 4055–4064. PMLR, 2018.

[27] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

[28] Rajeev Ranjan, Vishal M Patel, and Rama Chellappa. Hyperface: A deep multi-task learning framework for face detection,

landmark localization, pose estimation, and gender recognition. *IEEE transactions on pattern analysis and machine intelligence*, 41(1):121–135, 2017.

[29] Rajeev Ranjan, Swami Sankaranarayanan, Carlos D Castillo, and Rama Chellappa. An all-in-one convolutional neural network for face analysis. In *2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, pages 17–24. IEEE, 2017.

[30] Nataniel Ruiz, Eunji Chong, and James M Rehg. Fine-grained head pose estimation without keypoints. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 2074–2083, 2018.

[31] Christos Sagonas, Georgios Tzimiropoulos, Stefanos Zafeiriou, and Maja Pantic. 300 faces in-the-wild challenge: The first facial landmark localization challenge. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 397–403, 2013.

[32] Mingzhen Shao, Zhun Sun, Mete Ozay, and Takayuki Okatani. Improving head pose estimation with a combined loss and bounding box margin adjustment. *arXiv preprint arXiv:1905.08609*, 2019.

[33] Lucas Stoffl, Maxime Vidal, and Alexander Mathis. End-to-end trainable multi-instance pose estimation with transformers. *arXiv preprint arXiv:2103.12115*, 2021.

[34] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *arXiv preprint arXiv:2012.12877*, 2020.

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

[36] Huiyu Wang, Yukun Zhu, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. Max-deeplab: End-to-end panoptic segmentation with mask transformers. *arXiv preprint arXiv:2012.00759*, 2020.

[37] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. *arXiv preprint arXiv:2102.12122*, 2021.

[38] Yuqing Wang, Zhaoliang Xu, Xinlong Wang, Chunhua Shen, Baoshan Cheng, Hao Shen, and Huaxia Xia. End-to-end video instance segmentation with transformers. *arXiv preprint arXiv:2011.14503*, 2020.

[39] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. *arXiv preprint arXiv:2103.15808*, 2021.

[40] Lemeng Wu, Xingchao Liu, and Qiang Liu. Centroid transformers: Learning to abstract with attention. *arXiv preprint arXiv:2102.08606*, 2021.

[41] Miao Xin, Shentong Mo, and Yuanze Lin. Eva-gcn: Head pose estimation based on graph convolutional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1462–1471, 2021.

[42] Xuehan Xiong and Fernando De la Torre. Supervised descent method and its applications to face alignment. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 532–539, 2013.

[43] Xuehan Xiong and Fernando De la Torre. Global supervised descent method. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2664–2673, 2015.

[44] Fuzhi Yang, Huan Yang, Jianlong Fu, Hongtao Lu, and Baining Guo. Learning texture transformer network for image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5791–5800, 2020.

[45] Sen Yang, Zhibin Quan, Mu Nie, and Wankou Yang. Transpose: Towards explainable human pose estimation by transformer. *arXiv preprint arXiv:2012.14214*, 2020.

[46] Tsun-Yi Yang, Yi-Ting Chen, Yen-Yu Lin, and Yung-Yu Chuang. Fsa-net: Learning fine-grained structure aggregation for head pose estimation from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1087–1096, 2019.

[47] Tsun-Yi Yang, Jo-Han Hsu, Yen-Yu Lin, and Yung-Yu Chuang. Deepcd: Learning deep complementary descriptors for patch representations. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3314–3322, 2017.

[48] Tsun-Yi Yang, Yi-Hsuan Huang, Yen-Yu Lin, Pi-Cheng Hsiu, and Yung-Yu Chuang. Ssr-net: A compact soft stagewise regression network for age estimation. In *IJCAI*, volume 5, page 7, 2018.

[49] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. *arXiv preprint arXiv:2101.11986*, 2021.

[50] Hao Zhang, Mengmeng Wang, Yong Liu, and Yi Yuan. Fdn: Feature decoupling network for head pose estimation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 12789–12796, 2020.

[51] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, 23(10):1499–1503, 2016.

[52] Erjin Zhou, Haoqiang Fan, Zhimin Cao, Yuning Jiang, and Qi Yin. Extensive facial landmark localization with coarse-to-fine convolutional network cascade. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 386–391, 2013.

[53] Yijun Zhou and James Gregson. Whenet: Real-time fine-grained estimation for wide range head pose. *arXiv preprint arXiv:2005.10353*, 2020.

[54] Xiangyu Zhu, Zhen Lei, Xiaoming Liu, Hailin Shi, and Stan Z Li. Face alignment across large poses: A 3d solution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 146–155, 2016.

[55] Xiangyu Zhu, Zhen Lei, Junjie Yan, Dong Yi, and Stan Z Li. High-fidelity pose and expression normalization for face recognition in the wild. In *Proceedings of the IEEE con-*

ference on computer vision and pattern recognition, pages 787–796, 2015.

[56] Xiangxin Zhu and Deva Ramanan. Face detection, pose estimation, and landmark localization in the wild. In *2012 IEEE conference on computer vision and pattern recognition*, pages 2879–2886. IEEE, 2012.