Enhancing Lightweight Neural Networks by Modifying Reference Image Selection for Improved Head Pose Estimation

Nguyen Quang Khai

Department of Electrical Engineering

National Chung Cheng University, Chia-Yi 621, Taiwan, R.O.C.

Abstract

Head pose estimation plays a crucial role in applications such as human-machine interaction and virtual reality. While prior research has applied deep metric learning to this task, accuracy, particularly in terms of Mean Absolute Error (MAE), achieved with lightweight models remains limited. This study aims to enhance head pose estimation accuracy by optimizing model size and introducing new methods for selecting reference images during training.

Instead of using traditional random sampling, two novel approaches, Gaussian distribution and uniform distribution, are applied to select reference images. These methods improve estimation accuracy by focusing on reference images that are more closely related to the current head pose angles.

Experiments were conducted on three standard datasets: 300W-LP, AFLW2000, and BIWI, ensuring fair comparisons with existing techniques. The results demonstrate that our proposed approach achieves significant improvements in accuracy, effectively balancing model complexity and performance while advancing the state of head pose estimation.

1 Introduction

Deep learning, a subset of machine learning, involves training artificial neural networks with multiple layers to automatically learn and extract features from raw data. Its remarkable success across various domains is attributed to its ability to handle large datasets and model complex patterns. Deep learning techniques, especially Convolutional Neural Networks (CNN) [1], have revolutionized fields such as computer vision, natural language processing, and speech recognition by providing state-of-the-art performance on a wide range of tasks.

Deep Metric Learning (DML) [2], an advanced form of deep learning, focuses on learning data representations by optimizing the distances between data points in a fea-

ture space. Unlike traditional fixed feature extraction methods, DML adjusts the embedding space so that similar items cluster together while dissimilar items are spaced apart. This method, known as Deep Distance Metric Learning, is particularly effective in tasks requiring fine differentiation, such as face recognition and head pose estimation.

Head pose estimation (HPE) is a critical component for applications like human-computer interaction and virtual reality. Recent research has focused on improving accuracy while also making models more lightweight to be suitable for resource-limited environments [3]. This study builds on the work of Phung Huu Tai [4], aiming to further reduce the Mean Absolute Error (MAE) of head pose estimation while maintaining the model's complexity. Unlike traditional methods that rely on random sampling to select reference images, this study explores two new methods for reference image selection: Gaussian distribution and uniform distribution. By using these distributions, the model can focus on reference images that are more representative of the actual head pose, thereby improving accuracy. The proposed method is validated on three widely used datasets 300W-LP, AFLW2000, and BIWI and compared with existing state-of-the-art techniques.

2 Method description

The primary objective of this research is to enhance the accuracy of head pose estimation (HPE) by using new methods for reference image selection, specifically Gaussian distribution and uniform distribution, as opposed to the original method of random sampling. By adopting these new reference selection methods, the goal is to reduce the Mean Absolute Error (MAE) and improve the robustness of HPE models across various challenging scenarios. This approach aims to provide a more accurate and practical solution for real-world applications, where both precision and efficiency are essential.

2.1 Datasets

In our experiments, we employed three distinct datasets: 300W-LP, AFLW2000, and BIWI, each contributing unique characteristics essential for the evaluation of head pose estimation techniques.

The 300W-LP dataset, a comprehensive collection combining four subsets (AFW, LFPW, HELEN, and IBUG), features horizontally flipped images to enhance diversity and includes RGB images, 3D facial landmarks, and Euler angles, making it ideal for training models on diverse head poses. The AFLW2000 dataset, containing 2000 images annotated with 3D landmarks and head poses, presents challenges due to varying lighting and extreme poses. Finally, the BIWI dataset comprises 24 Kinect-recorded video sequences with depth and RGB images, annotated with rotation matrices and 3D translation vectors, covering yaw angles from -75 to 75 degrees.

2.2 Model Architecture

The model architecture for head pose estimation, as proposed in Phung Huu Tai's thesis, is shown in Figures 1 and 2. It uses a $K+1$-stream CNN-based framework, where the input image $\mathbf{x}^{in}$ and K reference images $\mathbf{x}_K^{ref}$ are processed through separate, identical streams. Each stream employs a CNN encoder with several 2D convolutional layers and a modified Convolutional Bottleneck Attention Module (CBAM) [5] to extract features.

The features $\dot{\mathbf{z}}^{in}$ and $\dot{\mathbf{z}}^{ref}$, which are then used to compute concentration factors of a matrix Fisher distribution, are reshaped into a $3 \times m$ matrix and normalized into a triplet of unit vectors, called a triple-vector $\hat{\mathbf{z}}^{in}$. Distances between the input triple-vector and K reference triple-vectors are computed, and the vectors are passed through a rotation estimator implemented as a fully connected layer. The Separated-Concentration Matrix Fisher distribution Estimation (SCME) [6] [7] module is used to learn rotation matrix parameters, enhancing pose estimation accuracy.

Increasing the number of reference images K impacts both the model's performance and complexity. A higher K improves head pose estimation accuracy by offering more diverse comparisons, potentially lowering the Mean Absolute Error (MAE). However, this also increases model size and computational demands, leading to longer training and inference times. Conversely, a smaller K reduces computational load but may compromise accuracy. Therefore, selecting the optimal K requires balancing accuracy with computational efficiency.

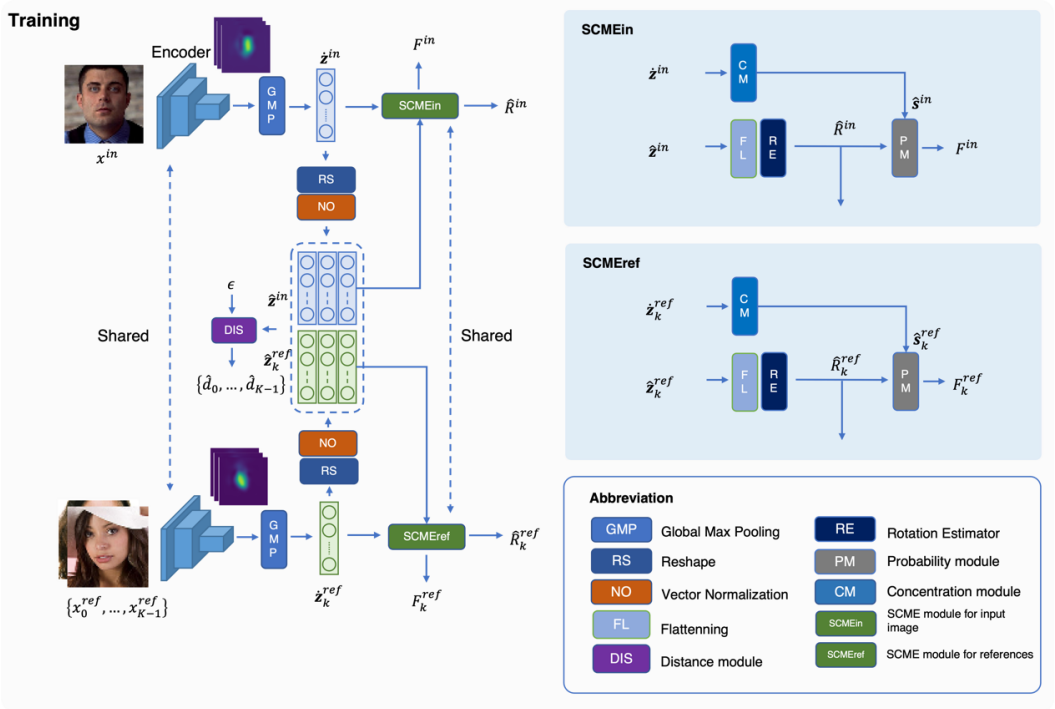

Figure 1: *Architecture of the training phase of our MHPE.*

During inference, the architecture is streamlined for efficiency. As shown in Fig. 2, only the input stream and rotation estimator are used, omitting the distance module and concentration factor branch. The model processes the input image through the CNN encoder, and the rotation estimator calculates the mean of the predicted rotation matrix from the extracted features. This approach enhances efficiency while maintaining high accuracy for real-time head pose estimation.

2.3 Data Pre-processing

The input to a head pose estimator is often a cropped face ROI based on a face detector or tracker. Many prior works require a fixed input size, so the cropped face ROI is typically resized to a square before passing it through the neural network. However,

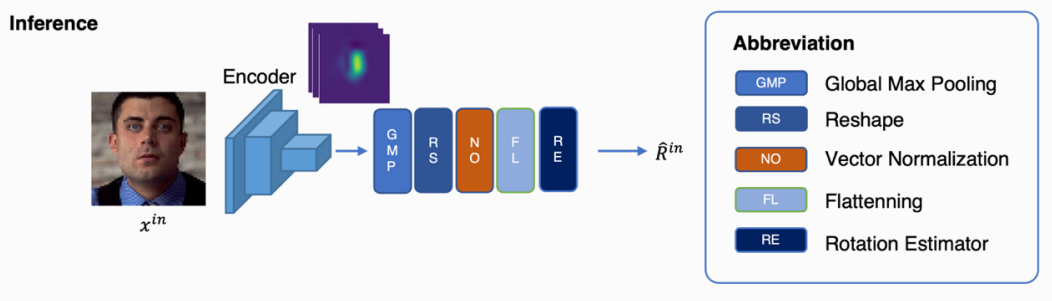

Figure 2: *Architecture of the inference phase of our MHPE.*

resizing can lead to distortions due to the difference in aspect ratio of the face bounding box, which can negatively impact the estimated head pose. Resizing without maintaining the aspect ratio causes the head width to increase, making the head appear more oriented to the left side.

In thesis, to avoid distortion, the bounding box size is adjusted before resizing. The center of the bounding box is preserved, and the bounding box size is re-defined to maintain the aspect ratio.

Additionally, augmentation techniques are used to improve estimation performance by generating more variants of a sample, thus increasing the diversity and size of the training dataset. In this project, two popular image augmentation methods are employed: random crop and Cutout. Cutout simulates occlusion by removing parts of the image, and the size and number of removed regions are pre-defined. The removed parts can be replaced with objects from the VOC dataset.

2.4 Encoder architecture

The encoder, shown in Fig. 3, is a convolutional neural network (CNN) designed to extract useful features from RGB images. To achieve a lightweight encoder for head pose estimation, several techniques are applied. There are three key considerations for designing an efficient neural network: input image resolution, number of feature channels, and network depth.

Higher resolution images require more computations but provide more detailed features. In our architecture, the input image size is kept relatively small at 64x64 pixels to minimize computational cost while still maintaining sufficient feature extraction. Experiments showed that 64x64 pixels is a suitable size for lightweight models.

Regarding the number of feature channels, increasing the channels improves recognition capability. In CNNs, shallow layers typically have fewer channels, while deeper layers increase the number of channels to enhance representation power. The network depth, which refers to the number of layers along a certain path, also plays a crucial role in enhancing nonlinearity and enabling the model to learn more complex input patterns. Our model employs three residual blocks and one attention module. Residual blocks are implemented using ResNet [8], with skip connections to prevent vanishing gradients.

To improve performance further, we introduce a new attention module called Modified Convolutional Bottleneck Attention Module (MCBAM), which is a modification of the CBAM [5]. The MCBAM consists of a Channel Attention Module (CAM) and a Spatial Attention Module (SAM). While the CAM remains unchanged, the SAM is modified to address limitations in the original CBAM. In the original CBAM, spatial

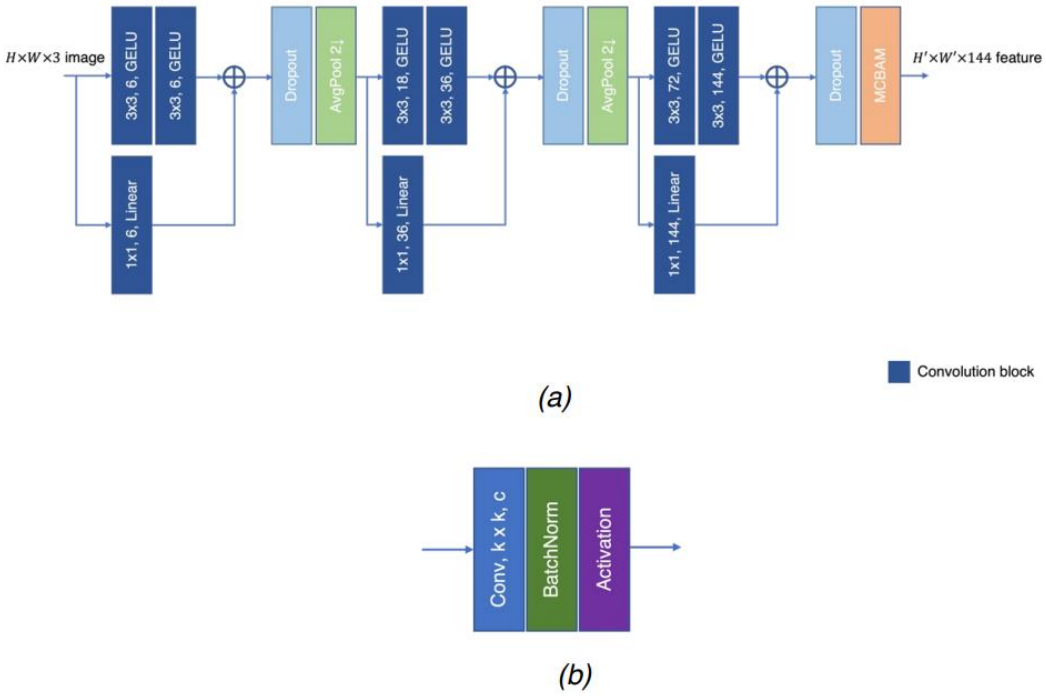

Figure 3: *(a) The architecture of the encoder, (b) design of a convolution block.*

attention is applied after pooling, which may lead to some loss of information. To overcome this, we replace the pooling layers with two convolutional layers to expand the receptive field and capture more useful information.

The architecture of the encoder and the design of the convolution block are shown in Fig. 3, and the architecture of our MCBAM along with the modified spatial attention is shown in Fig. 4.

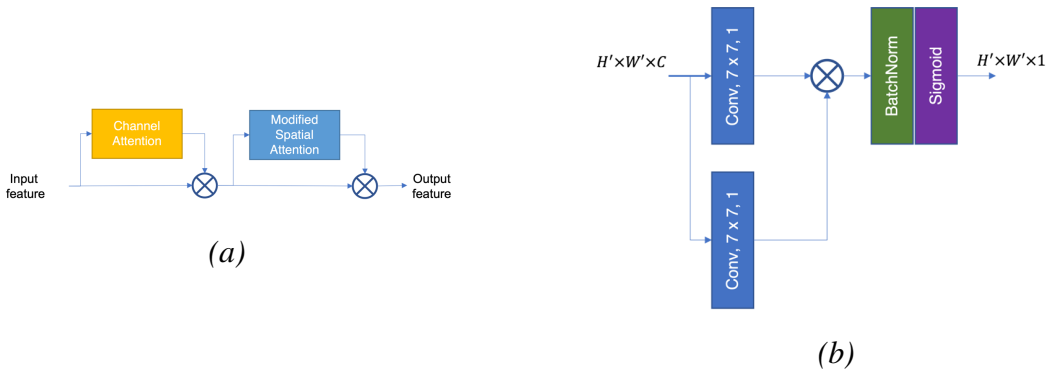

Figure 4: *(a) The architecture of our MCBAM, (b) the modified spatial attention.*

2.5 Distance Module

In the original research by Phung Huu Tai, the Distance module [9] was based on the calculation of the distance between the triple-vectors of the input image $\hat{\mathbf{z}}^{in}$ and the reference image $\hat{\mathbf{z}}^{in}$. The distance was computed using the trace product of the matrices derived from these vectors:

$$d(\hat{\mathbf{z}}^{\text{in}}, \hat{\mathbf{z}}^{\text{ref}}) = \sqrt{2(3 - \text{trace}(D))}$$

where $D = (\hat{\mathbf{z}}^{\text{ref}})^T$. $\hat{\mathbf{z}}^{\text{in}}$ and $trace(.)$ is the trace function that sums the diagonal elements of the matrix.

2.6 Evaluation Metrics

To assess the accuracy of head pose estimation, the metrics from Phung Huu Tai's research are utilized. The primary metric is the mean absolute error (MAE) in degrees of Euler angles, expressed as:

$$\text{MAE} = \frac{1}{3}\left(|\hat{a}_{\text{yaw}} - a_{\text{yaw}}| + |\hat{a}_{\text{pitch}} - a_{\text{pitch}}| + |\hat{a}_{\text{roll}} - a_{\text{roll}}|\right)$$

where $\hat{y}_{\text{yaw}}$, $\hat{y}_{\text{pitch}}$, and $\hat{y}_{\text{roll}}$ are the predicted yaw, pitch, and roll angles, and y_{yaw}, y_{pitch}, and y_{roll} are the ground truth values.

To address the potential ambiguity of Euler angles, particularly with gimbal lock, the mean absolute error of the angle between vectors (MAEV) is also employed. Given an estimated rotation matrix $\hat{R} = [\hat{v}_1\ \hat{v}_2\ \hat{v}_3]$ and the ground truth $R = [v_1\ v_2\ v_3]$, MAEV is defined as follows:

$$\text{MAEV} = \frac{1}{3}\left(\cos^{-1}(\hat{v}_1^T v_1) + \cos^{-1}(\hat{v}_2^T v_2) + \cos^{-1}(\hat{v}_3^T v_3)\right)$$

The vector angle errors on v_1, v_2, v_3 are called left vector error, down vector error, and front vector error, respectively.

2.7 Protocols of experiments

Two experimental protocols were designed to evaluate the effectiveness of the proposed head pose estimation model:

Protocol 1: The model was trained on the 300W-LP dataset and tested on the BIWI dataset. This protocol evaluates how well the model generalizes to unseen data, particularly with challenging head poses.

Protocol 2: The model was trained on the 300W-LP dataset and tested on the AFLW2000 dataset. This protocol measures the model's performance across a diverse range of head poses and facial variations.

These protocols provide a comprehensive evaluation of the model's robustness and ability to generalize to different datasets.

3 Experiment results

The experiments conducted aimed to evaluate the impact of changes in image size and the number of reference images K on the performance of head pose estimation. Additionally, the effectiveness of an improved distance calculation method was assessed. The experiments were conducted on two protocols:

Protocol 1: Training on 300W-LP and testing on BIWI.

Protocol 2: Training on 300W-LP and testing on AFLW2000.

To benchmark the effectiveness of the proposed modifications, the results obtained from Phung Huu Tai's original implementation were first replicated. The key hyperparameters used in Tai's method include:

Here are the results from Phung Huu Tai's research on the two protocols:

The following section details the results achieved through my enhancements:

Hyperparameter	MHPE/MHPE-PB
Batch size	100
Optimizer	Adam
Learning rate	0.01 (multiplied by 0.1 at 20^{th}, 50^{th}, and 80^{th} epoch)
Weight decay	1e-6

Table 1. Hyperparameters in Tai's implementation.

Protocol	**Method**	**Euler angle errors**				**Vector angle errors**			
		Yaw	**Pitch**	**Roll**	**MAE**	**Left**	**Down**	**Front**	**MAEV**
Protocol 1	**MHPE**	3.99	5.19	3.06	4.08	5.29	5.60	6.76	5.88
Protocol 2	**MHPE**	3.65	5.88	4.23	4.58	5.12	5.29	5.95	5.45

Table 2: MAE research results of Tai

3.1 Impact of using reference images in bins

When selecting reference images, the yaw angle of the input image determines the bin from which reference images are drawn. This process ensures that the selected references are relevant to the input pose.

Comparison of MAE results between Tai's thesis, Original code and Experiment using reference images in bins.

Result	**Range of bins**	**Protocol**	**Yaw**	**Pitch**	**Roll**	**MAE**
Tai Thesis		Protocol 1	3.99	5.19	3.06	4.08
		Protocol 2	3.65	5.88	4.23	4.58
Original code		Protocol 1	4.01	5.16	3.28	4.13
		Protocol 2	3.85	6.12	4.62	4.71
Experiment	10	Protocol 1	3.64	5.90	3.16	4.23
		Protocol 2	3.76	6.19	4.45	4.83
	20	Protocol 1	3.93	5.22	3.37	4.17
		Protocol 2	3.74	5.97	4.39	4.71
	40	Protocol 1	4.23	4.96	3.20	4.13
		Protocol 2	3.74	6.12	4.41	4.78
	60	Protocol 1	4.06	5.11	3.19	4.11
		Protocol 2	3.79	6.14	4.63	4.81

1. **Identify the bin:**

- Determine the bin corresponding to the yaw angle of the input image.
- For example, if the input image has $yaw = -10^\circ$, and the bins are divided as $[-15^\circ, 0^\circ]$, $[0^\circ, 15^\circ]$, etc., the image belongs to the bin $[-15^\circ, 0^\circ]$.

2. **Retrieve reference images:**

   - Select reference images from the identified bin.
   - If the identified bin does not contain enough reference images, retrieve additional images from the neighboring bins.
   - For example, if the bin $[-15^\circ, 0^\circ]$ is insufficient, additional images can be retrieved from $[0^\circ, 15^\circ]$ or $[-30^\circ, -15^\circ]$.

This approach ensures that the selected reference images are as relevant as possible while maintaining robustness in cases where certain bins lack sufficient data.

3.2 Impact of using Gaussian distribution

The use of the Gaussian distribution in this context is a method for selecting reference images based on their proximity to a given input yaw angle. This approach aims to give more weight to images whose yaw angles are closer to the input angle, providing a more focused and representative set of reference images. Below is a theoretical explanation of how the Gaussian distribution is applied:

1. Motivation for Using Gaussian Distribution:

In head pose estimation, the input to the model is often a specific yaw angle, and the task is to estimate the corresponding head pose. However, instead of selecting random reference images for training or testing, it is more effective to sample images that are close to the input yaw angle. This ensures that the references are relevant and the model learns from a focused subset of the data. By using a Gaussian distribution, we can assign higher weights to images that are near the input angle and lower weights to those that are farther away.

2. The Gaussian Distribution Formula:

The Gaussian distribution is defined as:

$$W(x) = e^{-\frac{(x-\mu)^2}{2\sigma^2}},$$

where:

- $W(x)$ is the weight assigned to the yaw angle x,
- μ is the mean of the Gaussian distribution (which represents the input yaw angle),
- σ is the standard deviation of the Gaussian distribution (which controls the spread of the weights),
- x is the center of the bin.

The Gaussian function assigns higher weights to values of x that are closer to the mean μ and lower weights to values that are farther away. The weight decreases exponentially as the difference between x and μ increases.

3. Dividing the Range into Bins:
The yaw angle range $[-180^\circ, 180^\circ]$ is divided into discrete bins (e.g., 20° bins). For each bin, the center of the bin is used as the value of x in the Gaussian function. The Gaussian weight for each bin is computed using the formula, where the mean μ is the input yaw angle. This ensures that bins that are closer to the input angle have higher weights.

4. Normalization of Weights:
Since the computed weights can vary in magnitude, it is important to normalize them. Normalization ensures that the sum of all weights equals 1, making the weight distribution comparable across different experiments. The normalization is done by dividing each weight by the sum of all weights:

$$\text{Normalized Weight}(x) = \frac{W(x)}{\sum W(x)}.$$

5. Sampling the References:
Once the weights are normalized, the number of references to be sampled from each bin is determined by multiplying the normalized weight by the total number of references k required. This ensures that bins with higher weights contribute more references, while bins with lower weights contribute fewer references:

$$\text{References from Bin} = \text{Normalized Weight}(x) \times k.$$

6. Final Sampling:
After calculating the number of references to be sampled from each bin, the final references are chosen. The number of references from each bin is proportional to the normalized Gaussian weight, ensuring that the selected references are concentrated around the input yaw angle.

Comparison of MAE results between Tai's thesis, Original code and Experiment using reference images by Gaussian distribution.

Result	Range of bins	Std Dev	Protocol	Yaw	Pitch	Roll	MAE
Tai Thesis			Protocol 1	3.99	5.19	3.06	4.08
			Protocol 2	3.65	5.88	4.23	4.58
Original code			Protocol 1	4.01	5.16	3.28	4.13
			Protocol 2	3.85	6.12	4.62	4.71
Experiment	20	15	Protocol 1	3.72	5.41	3.30	4.14
			Protocol 2	4.09	6.30	4.60	4.95
	20	20	Protocol 1	3.65	5.49	3.14	4.09
			Protocol 2	3.91	6.10	4.58	4.71
	20	30	Protocol 1	3.58	5.30	3.10	3.98
			Protocol 2	3.91	6.09	4.52	4.69
	20	37.5	Protocol 1	3.60	5.39	3.11	4.03
			Protocol 2	3.90	6.09	4.58	4.78
	20	45	Protocol 1	3.83	5.39	3.34	4.22
			Protocol 2	3.83	6.36	4.61	4.92

Example:
The lookup table divides the yaw angle range $[-180^\circ, 180^\circ]$ into 20° bins. For the input

yaw angle $\mu = 10^\circ$, we compute the weights using a Gaussian distribution and sample references accordingly.

Steps:

1. Setting the Mean (μ) and Standard Deviation (σ): For the input yaw angle:

$$\mu = 10^\circ \quad \text{(Gaussian mean)}, \quad \sigma = 30^\circ \quad \text{(standard deviation)}.$$

We calculate weights for each bin using the formula:

$$W(x) = e^{-\frac{(x-\mu)^2}{2\sigma^2}}.$$

2. Computing Gaussian Weights for Each Bin: For bins centered on x, compute the Gaussian weight:

Bin Range	Center (x)	Gaussian Weight ($\mu = 10^\circ, \sigma = 30^\circ$)
$-20^\circ \to 0^\circ$	-10°	$e^{-\frac{(-10-10)^2}{2(30)^2}} \approx 0.800$
$0^\circ \to 20^\circ$	10°	$e^{-\frac{(10-10)^2}{2(30)^2}} = 1.000$
$20^\circ \to 40^\circ$	30°	$e^{-\frac{(30-10)^2}{2(30)^2}} \approx 0.800$
$40^\circ \to 60^\circ$	50°	$e^{-\frac{(50-10)^2}{2(30)^2}} \approx 0.367$
$-40^\circ \to -20^\circ$	-30°	$e^{-\frac{(-30-10)^2}{2(30)^2}} \approx 0.367$

Other bins have negligible weight.

3. Normalizing the Weights: To normalize, divide each weight by the sum of all weights:

$$\text{Total Weight} = 0.367 + 0.800 + 1.000 + 0.800 + 0.367 = 3.334.$$

The normalized weights are:

Bin Range	Weight	Normalized Weight
$-20^\circ \to 0^\circ$	0.800	$\frac{0.800}{3.334} \approx 0.240$
$0^\circ \to 20^\circ$	1.000	$\frac{1.000}{3.334} \approx 0.300$
$20^\circ \to 40^\circ$	0.800	$\frac{0.800}{3.334} \approx 0.240$
$40^\circ \to 60^\circ$	0.367	$\frac{0.367}{3.334} \approx 0.110$
$-40^\circ \to -20^\circ$	0.367	$\frac{0.367}{3.334} \approx 0.110$

4. Sampling References: Suppose we need $k = 10$ references. The number of references from each bin is:

$$\text{Images from Bin} = \text{Normalized Weight} \times k.$$

The number of images sampled from each bin is as follows:

Bin Range	Normalized Weight	Images to Sample($k = 10$)
$-20^\circ \to 0^\circ$	0.240	$0.240 \times 10 \approx 2$
$0^\circ \to 20^\circ$	0.300	$0.300 \times 10 \approx 3$
$20^\circ \to 40^\circ$	0.240	$0.240 \times 10 \approx 2$
$40^\circ \to 60^\circ$	0.110	$0.110 \times 10 \approx 1$
$-40^\circ \to -20^\circ$	0.110	$0.110 \times 10 \approx 1$

Final Sampling: The $k = 10$ references are distributed as follows:

- 3 references from $0^\circ \to 20^\circ$, - 2 references each from $-20^\circ \to 0^\circ$ and $20^\circ \to 40^\circ$, - 1 reference each from $40^\circ \to 60^\circ$ and $-40^\circ \to -20^\circ$.

Conclusion:

The use of the Gaussian distribution in this setup is designed to focus the reference selection process on images that are near the input angle, ensuring that the model learns from the most relevant data. The Gaussian distribution provides a smooth, probabilistic way of emphasizing nearby reference images and is especially useful when dealing with angles that are close to the input, improving model performance for head pose estimation.

3.3 Impact of using Reverse Gaussian Distribution

In some cases, we may need to use a reverse Gaussian distribution, where instead of computing weights for the reference angles based on a given input angle, we estimate the input angle (μ) from the known weights or distribution of reference angles. This approach can be helpful when we have predefined reference images with their associated weights and want to estimate which input angles could have generated these weights.

In the inverse Gaussian method, the weights are computed such that the weight distribution is the opposite of the standard Gaussian distribution. Specifically, bins farther from the mean value (μ) will have higher weights, while bins closer to the mean will have lower weights.

Suppose we have yaw bins with a mean (μ) of 10° and a standard deviation (σ) of 30°, the weights for each bin are computed using the inverse Gaussian formula:

$$W(x) = e^{\frac{(x-\mu)^2}{2\sigma^2}}$$

Below is the weight table for the yaw bins:

Bin Range	Weight (Inverse Gaussian)
$-20^\circ \to 0^\circ$	1.56
$0^\circ \to 20^\circ$	1.00
$20^\circ \to 40^\circ$	1.56
$40^\circ \to 60^\circ$	5.93
$-40^\circ \to -20^\circ$	5.93

After normalizing the weights to ensure their sum equals 1, we get the following normalized weight table:

$$\text{Total weight} = 1.56 + 1.00 + 1.56 + 5.93 + 5.93 = 16.98$$

Bin Range	Normalized Weight	Number of References (k = 10)
$-20^\circ \to 0^\circ$	0.092	1
$0^\circ \to 20^\circ$	0.059	1
$20^\circ \to 40^\circ$	0.092	1
$40^\circ \to 60^\circ$	0.349	3
$-40^\circ \to -20^\circ$	0.349	3

Thus, in the inverse Gaussian method, references are sampled more heavily from bins farther from the mean value (μ) and less from bins closer to it.

Comparison of MAE results between Tai's thesis, Original code and Experiment using reference images by Reverse Gaussian distribution.

Result	Range of bins	Std Dev	Protocol	Yaw	Pitch	Roll	MAE
Tai Thesis			Protocol 1	3.99	5.19	3.06	4.08
			Protocol 2	3.65	5.88	4.23	4.58
Original code			Protocol 1	4.01	5.16	3.28	4.13
			Protocol 2	3.85	6.12	4.62	4.71
Experiment	20	20	Protocol 1	9.85	7.05	4.48	7.13
			Protocol 2	11.45	8.85	7.39	9.24
	20	30	Protocol 1	6.29	10.55	4.44	7.09
			Protocol 2	10.09	8.71	7.46	8.75
	20	45	Protocol 1	5.37	8.16	4.56	6.03
			Protocol 2	6.23	8.01	6.62	6.96

3.4 Impact of using Uniform Distribution

In this method, we use a **uniform distribution** to sample reference images from each bin. Instead of using a Gaussian distribution, where weights are calculated based on the distance from the mean, we simply select one image from each bin until the required number of references (k) is reached. If there are fewer bins than the required references, the process will repeat, sampling from the bins again until the total number of references is collected.

Steps:

- Divide the range: Divide the yaw angle range $[-180^\circ, 180^\circ]$ into equal-sized bins.
- Sampling references: From each bin, one reference image is selected. This means that for each bin, you choose one image that corresponds to the yaw angle within the range of that bin.
- Repetition if necessary: If the number of bins is insufficient to meet the required references (k), the process repeats, sampling from the bins in the same order until k references are collected.

Example:

For instance, if there are 5 bins and 10 references are needed, the sampling process would look as follows:
- Bins: Bin 1, Bin 2, Bin 3, Bin 4, Bin 5
- References Sampled:
1. Bin 1 Reference 1
2. Bin 2 $\rightarrow$ Reference 2
3. Bin 3 $\rightarrow$ Reference 3
4. Bin 4 $\rightarrow$ Reference 4
5. Bin 5 $\rightarrow$ Reference 5
6. Bin 1 $\rightarrow$ Reference 6
7. Bin 2 $\rightarrow$ Reference 7
8. Bin 3 $\rightarrow$ Reference 8
9. Bin 4 $\rightarrow$ Reference 9
10. Bin 5 $\rightarrow$ Reference 10

Comparison of MAE results between Tai's thesis, Original code and Experiment using reference images in bins.

Result	Range of bins	Protocol	Yaw	Pitch	Roll	MAE
Tai Thesis		Protocol 1	3.99	5.19	3.06	4.08
		Protocol 2	3.65	5.88	4.23	4.58
Original code		Protocol 1	4.01	5.16	3.28	4.13
		Protocol 2	3.85	6.12	4.62	4.71
Experiment	20	Protocol 1	18.57	14.67	7.57	13.61
		Protocol 2	18.91	13.62	9.54	14.03

Thus, references are uniformly distributed across the bins, and if there are fewer bins than required, the bins will be sampled again to meet the reference count.

The experiments are run on a compute with an Intel Core i9 9900k for CPU and a RTX 3080 for GPU.

4 Discussion

Comparison of MAE results between Tai's thesis, Original code and Experiment.

Result	Range of bins	Std Dev	Protocol	Yaw	Pitch	Roll	MAE
Tai Thesis	-	-	Protocol 1	3.99	5.19	3.06	4.08
			Protocol 2	3.65	5.88	4.23	4.58
Original code	-	-	Protocol 1	4.01	5.16	3.28	4.13
			Protocol 2	3.85	6.12	4.62	4.71
Reference images in bins	20	-	Protocol 1	4.21	5.13	3.16	4.17
			Protocol 2	3.61	6.07	4.34	4.71
Gaussian distribution	20	30	Protocol 1	3.58	5.30	3.10	3.98
			Protocol 2	3.91	6.09	4.58	4.69
Reverse Gaussian distribution	20	45	Protocol 1	5.37	8.16	4.56	6.03
			Protocol 2	6.23	8.01	6.62	6.69
Uniform distribution	20	-	Protocol 1	18.57	14.67	7.57	13.61
			Protocol 2	18.91	16.62	9.54	14.03

The experiment of replacing random sampling with Gaussian distribution showed significant improvement compared to the original results. I used the results from the Gaussian distribution to compare with the original code because, when running Tai's original code, some parameters might have been altered, preventing it from achieving the best possible results as in Tai's original work. Therefore, I only used the results from the Gaussian distribution to compare with the original code I received. This demonstrates that Gaussian distribution, when used correctly, brings improvements across both protocols.

Specifically, with a standard deviation of 30, the results from the Gaussian distribution showed optimal performance across the tests. However, when using the inverse Gaussian distribution to compare with the original code, the results significantly worsened across both protocols. This suggests that, when selecting reference images, factors

such as the proximity of yaw angles between the input image and the reference images are crucial. The reference images should include a large number of yaw angles close to the input image's yaw and a smaller number of angles that are directly opposite to the input yaw. This helps the model learn features similar to the input's yaw while also incorporating some diverse features that improve accuracy in handling more complex cases.

The results from the experiment using uniform distribution also demonstrated that if only one image is selected per bin and there is no substantial number of images with yaw angles close to the input yaw, it results in a lack of focus on critical angles: Uniform distribution treats all yaw angles as equally important. However, not all angles have an equivalent impact on learning. Yaw angles close to the input image's yaw often provide more useful information, while opposing or unrelated angles can introduce noise. Additionally, selecting only one reference image per bin limits the diversity in the sample data. The model is not provided with enough information from various yaw angles, leading to inefficient learning, especially when handling complex cases.

The results from the experiment using reference images grouped into bins (where each bin contains images with yaw angles closest to the input image) also reinforce this observation. The results obtained by selecting reference images with yaw angles closest to the input image show accuracy close to the original results. However, using only the closest images did not lead to optimal improvement. This is because of the lack of diversity in distinct features, especially when compared with opposite yaw angles. If there is a combination of both similar and diverse features, the model can make better-informed decisions, ultimately achieving the best results.

5 Conclusion

This study highlights the significant impact of adopting Gaussian distribution in the reference image sampling process for head pose estimation. The shift from random sampling to Gaussian-based sampling demonstrates clear improvements in accuracy across both Protocol 1 and Protocol 2, particularly when the standard deviation is set to 30. This result underscores the importance of selecting reference images with yaw angles closely aligned to the input image, while also incorporating a smaller number of images with opposing yaw angles. Such a strategy ensures the model can effectively learn key features that are both similar to the input and sufficiently diverse to improve generalization.

Additionally, the findings reveal that merely focusing on reference images with yaw angles closest to the input is insufficient for optimal performance. While this approach preserves proximity, it lacks the diversity needed to differentiate finer details of head orientations. A balanced strategy that combines similar and contrasting angles emerges as critical for achieving high-quality estimations.

Overall, this research underscores the importance of carefully designing the sampling and selection mechanisms for reference images. By leveraging Gaussian distribution and optimizing the balance of yaw angles, it is possible to achieve substantial performance gains. Future work should explore further refinements to these sampling strategies, as well as evaluate their applicability across more diverse datasets and real-world scenarios. This will pave the way for more robust and accurate head pose estimation models.

Acknowledgements

I would like to extend my heartfelt gratitude to the College of Engineering, National Chung Cheng University, Taiwan, Republic of China, for their generous financial and technical support, which has been crucial for the completion of this study. My deepest thanks go to Professor Wen-Nung Lie for his outstanding guidance, invaluable mentorship, and insightful feedback throughout this research. I am also sincerely grateful to my colleagues and senior researchers for their steadfast support and thoughtful contributions, which have played a pivotal role in the success of this work.

References

[1] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2012, pp. 1097–1105.

[2] J. Wang, F. Zhou, S. Wen, X. Liu, and Y. Lin, "Deep metric learning with angular loss," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Venice, Italy, 2017, pp. 2593–2601.

[3] W. N. Lie, M. Yim, L. Aing, and J. C. Chiang, "3d head pose estimation based on graph convolutional network from a single rgb image," in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, Bordeaux, France, 2022.

[4] P. H. Tai and W.-N. Lie, "Mhpe: A lightweight neural network based on novel deep metric learning for head pose estimation from a single rgb image," https://hdl.handle.net/11296/e8x7y4, 2024, national Digital Library of Theses and Dissertations in Taiwan.

[5] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "Cbam: Convolutional block attention module," in *European Conference on Computer Vision (ECCV)*, 2018.

[6] T. Lee, "Bayesian attitude estimation with the matrix fisher distribution on so(3)," *IEEE Transactions on Automatic Control*, vol. 63, no. 10, pp. 3377–3392, 2018.

[7] D. Mohlin, G. Bianchi, and J. Sullivan, "Probabilistic orientation estimation with matrix fisher distributions," in *Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 2016.

[9] D. Q. Huynh, "Metrics for 3d rotations: Comparison and analysis," *Journal of Mathematical Imaging and Vision*, vol. 35, pp. 155–164, 2009.