Spatio-Temporal Attention and Gaussian Processes for Personalized Video Gaze Estimation (Supplementary Materials)

Swati Jindal[1] Mohit Yadav[2] Roberto Manduchi[1]

[1]University of California Santa Cruz [2] University of Massachusetts Amherst

1. Proposed Method – Omitted Details

We provide the mathematical formulation of Dual-SAM and Cross-SAM in Algorithm 1 and 2, respectively.

Algorithm 1 Dual-Spatial Attention Module (Dual-SAM)

Input: X_{t-1}, X_t $\in \mathbb{R}^{h\times w\times k}$

Output: $\mathbf{z}_t$ $\in \mathbb{R}^{3\cdot k}$

$X'_{t-1} = [X_{t-1}; X_t - X_{t-1}]$

$X'_t = [X_t; X_t - X_{t-1}]$ $\in \mathbb{R}^{h\times w\times 2\cdot k}$

$A_{t-1} = \sigma(\text{conv}(\text{ReLU}(\text{conv}(X'_{t-1}))))$

$A_t = \sigma(\text{conv}(\text{ReLU}(\text{conv}(X'_t))))$ $\in \mathbb{R}^{h\times w\times 1}$

$\mathbf{v}_{t-1} = \sum_{h,w} A_{t-1} \odot X_{t-1}$

$\mathbf{v}_t = \sum_{h,w} A_t \odot X_t$ $\in \mathbb{R}^{k}$

$\mathbf{z}_t = [\mathbf{v}_{t-1}; \mathbf{v}_t - \mathbf{v}_{t-1}; \mathbf{v}_t]$ $\in \mathbb{R}^{3\cdot k}$

return $\mathbf{z}_t$

Algorithm 2 Cross-Spatial Attention Module (Cross-SAM)

Input: X_{t-1}, X_t $\in \mathbb{R}^{h\times w\times k}$

Output: $\mathbf{z}_t$ $\in \mathbb{R}^{3\cdot d}$

$X_{t-1} = \text{flat}(\text{conv}(X_{t-1}) + \mathbf{1}_{h,w} \odot P_{2d})$

$X_t = \text{flat}(\text{conv}(X_t) + \mathbf{1}_{h,w} \odot P_{2d})$ $\in \mathbb{R}^{h\cdot w\times d}$

$X_{t-1} = \text{crossatten}(X_{t-1}, X_t, X_t)$

$X_t = \text{crossatten}(X_t, X_{t-1}, X_{t-1})$ $\in \mathbb{R}^{h\cdot w\times d}$

$\mathbf{v}_{t-1} = \sum_{h,w} \text{unflat}(X_{t-1}, h\times w)$

$\mathbf{v}_t = \sum_{h,w} \text{unflat}(X_t, h\times w)$ $\in \mathbb{R}^{d}$

$\mathbf{z}_t = [\mathbf{v}_{t-1}; \mathbf{v}_t - \mathbf{v}_{t-1}; \mathbf{v}_t]$ $\in \mathbb{R}^{3\cdot d}$

return $\mathbf{z}_t$

Transformer Block. Figure 1 shows the architecture of a single transformer layer used in the temporal sequence model of the STAGE method. MLP is a Multi-Perceptron layer, and we use L layers stacked together in the TSM.

We incorporate learned temporal position embeddings to enable the transformer model to discern temporal relationships within the input feature sequence. These embeddings are uniquely associated with each position, providing the model with explicit information about the relative ordering of elements within the sequence. The embedded features are then passed through multiple layers, each consisting of masked multi-head attention, LayerNorm (LN), and MLP. Masked multi-head attention allows the transformer model to attend to only past frame features. The output of the TSM is a feature sequence passed through an LN layer, similar to the GPT-2 model [3].

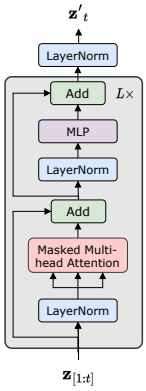

Figure 1. Block diagram of transformer temporal sequence model.

2. Additional Implementation Details

The Dual-SAM consists of two convolutional layers with kernel size 1 and output feature maps of 64 and 1, respectively. The first convolutional layer has a group normalization layer [4] applied to the output features, followed by a dropout layer with $p = 0.5$. In Cross-SAM and Hybrid-SAM, we project the incoming features to higher channels through a convolution layer with $d = 512$ and a kernel size 1. After adding 2D positional embeddings to the projected feature maps, they go through the cross-attention encoder, which consists of four heads and two layers with an embedding size of 64.

Method	Full	180°	20°
Dual-SAM(1-block)+Tx	10.13	9.93	7.23
Hybrid-SAM(1-block)+Tx	10.10	9.90	7.33
Dual-SAM(4-blocks)+Tx	12.13	11.68	9.33
Hybrid-SAM(4-blocks)+Tx	10.25	10.08	7.27

(a) Within-dataset evaluation

Method	EyeDiap	Full	180°
Dual-SAM(1-blocks)+Tx	6.77	23.99	23.38
Hybrid-SAM(1-blocks)+Tx	6.54	23.77	23.17
Dual-SAM(4-blocks)+Tx	7.27	23.34	22.74
Hybrid-SAM(4-blocks)+Tx	7.55	23.52	22.91

(b) Cross-dataset evaluation

Table 1. **Ablation Study:** Comparison of different numbers of SAM blocks employed in our STAGE method. Tx is transformer-based TSM, and training is performed for within-data and cross-data settings in (a) and (b), respectively. The metric reported is mean angular errors (in degrees).

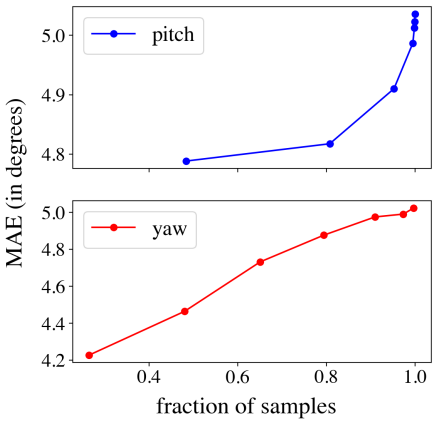

Figure 2. Comparison of Mean Angular Error (in degrees) of gaze components (yaw or pitch) with increasing fraction of test samples sorted with respect to the uncertainty of GP predictions. Plots exhibit that GPs are more accurate when the prediction is relatively more confident (with less variance).

The TSM model has two variants: an LSTM variant and a transformer variant. The LSTM variant consists of one unidirectional LSTM layer with a hidden dimension of 128. The transformer variant is based on GPT-2 [3] network with 6-heads and 6-layers, operating on a dimension of $d = 128$, and initialized randomly. The gaze prediction layer consists of two fully connected (FC) layers. The first FC layer has a SeLU activation function and a hidden dimension of the same size as the input dimension. The second FC layer outputs the 2D gaze direction angles, pitch and yaw.

Our STAGE model is implemented in PyTorch [2]. We set $\lambda = 0.001$ for cross-data and $\lambda = 0$ for within-data evaluations. For GP hyper-parameter optimization, we use Adam optimizer with a learning rate of 0.001, implemented using GPytorch [1]. Our code and trained models will be made publicly available in the future and are zipped in supplementary.

3. Ablation Study

In the ablation study, we study the impact of adding multiple SAM blocks in the STAGE model, where the output of one SAM goes as input to the next. The ablation study on the number of Dual- and Hybrid-SAM blocks (four blocks vs. one block) for within-data and cross-data settings are shown in Tables 1(a) and (b), respectively. We observe no significant improvements over a single block of SAM, indicating that one SAM block is enough to provide spatial motion cues between consecutive frame features and improve performance.

4. Additional Results for GP Evaluation

For assessing the effectiveness of the GP model's uncertainty, we provide additional analysis of gaze predictions, as illustrated in Figure 2. Our evaluation begins with an analysis of the GP's posterior variance diagonal. We arrange this in ascending order and then apply different uncertainty thresholds to it. For each selected threshold, we compute the MAE on test samples that exhibit uncertainty levels below the threshold. This procedure is repeated across a range of different thresholds to evaluate performance. Figure 2 presents a comparison of the MAE for yaw and pitch against increasing fractions of test data samples. These samples are sorted according to the uncertainty in the GP prediction. This analysis demonstrates that GPs tend to deliver more accurate results when their variance is lower, signifying greater confidence in the predictions. Therefore, the uncertainty measure in the GP model can act as an effective indicator to avoid making inaccurate predictions.

We also provide additional visualizations of the predictions from personalized GP on top of the STAGE model, similar to Figure 6 in the main manuscript. Figure 3a and 3b respectively show certain and uncertain prediction images from the EYEDIAP dataset after performing GP personalization. The ground truth and predicted gaze directions are respectively shown with blue and pink colored arrows, and the corresponding uncertainty of prediction is shown with the green colored triangle.

References

[1] Jacob R Gardner, Geoff Pleiss, David Bindel, Kilian Q Weinberger, and Andrew Gordon Wilson. Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu accelera-

(a) Certain predictions for EYEDIAP dataset

(b) Uncertain Predictions for EYEDIAP dataset

Figure 3. The figure depicts a few confident 3a and uncertain 3b predictions for gaze directions after GP's personalization on the EYEDIAP dataset. Blue and pink arrows show ground truth and predicted gaze directions, respectively. The green-colored region offers uncertainty of the predictions in pink arrows. The uncertainty region often covers the ground truth, *i.e.*., the pink arrows are in the green-colored area.

tion. In *Advances in Neural Information Processing Systems*, 2018. 2

[2] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 2

[3] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019. 1, 2

[4] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision*, pages 3–19, 2018. 1