STAPLE: Siamese Transformer Assisted Pseudo Label Ensembling for Unsupervised Domain Adaptation in No-Reference IQA

Arshita Gupta
Samsung Research America
a7.gupta@samsung.com

Zhe Zhu
Samsung Research America
zhe.zhu@samsung.com

Tien Bau
Samsung Research America
t.bau@samsung.com

1. Supplementary

1.1. Mean and Variance of Sample Mean

Let X_1, X_2, ..., X_n be a random sample of size n from a distribution with mean μ, variance σ^2 and sample mean $\bar{X}$. The Expected value of the sample mean, that is $E[\bar{X}]$, is given by:

$$E[\bar{X}] = E\left[\frac{X_1 + X_2 + ... + X_n}{n}\right] \tag{1}$$

The linear operator property 1 of expectation is:

$$E[X + Y] = E[X] + E[Y] \tag{2}$$

and property 2 is:

$$E[cX] = E[c].E[X] = cE[x] \tag{3}$$

here c is a constant. Using Equation 2 and 2 in Equation 1 we get:

$$E[\bar{X}] = \frac{1}{n}(E[X_1] + E[X_2] + ... + E[X_n]) \tag{4}$$

As X_i are identically distributed, they have the same mean μ. We then get:

$$E[\bar{X}] = \frac{1}{n}(\mu + \mu + ... + \mu) = \frac{n\mu}{n} \tag{5}$$

$$E[\bar{X}] = \mu \tag{6}$$

Equation 6 shows that the expected value (or mean) of sample mean $\bar{X}$ is μ, which is the mean of individual X_i.

Similarly, for variance of sample mean $\bar{X}$ is:

$$Var(\bar{X}) = Var\left(\frac{X_1 + X_2 + ... + X_n}{n}\right) \tag{7}$$

Since variance scales by the square of a constant, we can factor out $\frac{1}{n^2}$ from above equation:

$$Var(\bar{X}) = \frac{1}{n^2}Var(X_1 + X_2 + ... + X_n) \tag{8}$$

$$Var(\bar{X}) = \frac{1}{n^2}(Var(X_1) + Var(X_2) + ... + Var(X_n)) \tag{9}$$

similar to mean, each X_i has the same variance σ^2:

$$Var(\bar{X}) = \frac{1}{n^2}(\sigma^2 + \sigma^2 + ... + \sigma^2) = \frac{1}{n^2}(n\sigma^2) \tag{10}$$

$$Var(\bar{X}) = \frac{\sigma^2}{n} \tag{11}$$

Equation 11 indicates that the variance of the sample mean $\bar{X}$ is inversely proportional to sample size n. This implies that $Var(\bar{X})$ decreases as sample size n increases, thus leading to a more accurate estimate.

1.2. Decomposition of Expected MSE into Bias and Variance

The bias-variance decomposition can be used to decompose the mean-squared error of an predictor $\bar{y}$ into two parts which are its bias and variance.

Firstly consider a predictor $\bar{y}$ and true value y. The bias is defined as the deviation of its expectation value from the true value that we want to predict.

$$Bias(\bar{y}) = E[\bar{y} - y] \tag{12}$$

while the variance is defined as the squared deviation from its expectation value given by:

$$Var(\bar{y}) = E[(\bar{y} - E[\bar{y}])^2] \tag{13}$$

The expected MSE (Mean Squared Error) of predictor $\bar{y}$ is given by:

$$MSE(\bar{y}) = E[(\bar{y} - y)^2] \tag{14}$$

Expanding the right hand term in Equation 14:

$$E[(\bar{y} - y)^2] = E[\bar{y}^2 + y^2 - 2\bar{y}y] \tag{15}$$

Using properties in equations 2 and 3

$$E[(\bar{y}-y)^2] = E[\bar{y}^2] + E[y^2] - E[2\bar{y}y] \quad (16)$$

$$E[(\bar{y}-y)^2] = E[\bar{y}^2] + y^2 - 2yE[\bar{y}] + E[\bar{y}]^2 - E[\bar{y}]^2 \quad (17)$$

rearranging the above equation:

$$E[(\bar{y}-y)^2] = (E[\bar{y}]-y)^2 + E[\bar{y}^2] - E[\bar{y}]^2 \quad (18)$$

$$E[(\bar{y}-y)^2] = (E[\bar{y}]-y)^2 + E[\bar{y}^2] + E[\bar{y}]^2 - 2E[\bar{y}]^2 \quad (19)$$

$$E[(\bar{y}-y)^2] = (E[\bar{y}]-y)^2 + E[\bar{y}^2] + E[E[\bar{y}]^2] - 2E[\bar{y}E[E[\bar{y}]]] \quad (20)$$

$$E[(\bar{y}-y)^2] = (E[\bar{y}]-y)^2 + E[\bar{y}^2 + E[\bar{y}]^2 - 2\bar{y}E[\bar{y}]] \quad (21)$$

$$E[(\bar{y}-y)^2] = (E[\bar{y}]-y)^2 + E[\bar{y}^2 + E[\bar{y}]^2 - 2\bar{y}E[\bar{y}]] \quad (22)$$

$$E[(\bar{y}-y)^2] = (E[\bar{y}]-y)^2 + E[(\bar{y}-E[\bar{y}])^2] \quad (23)$$

In Equation 24, the first term indicates the bias while the second term indicates the variance as shown in Equations 12 and 13 respectively. Thus we get the bias-variance decomposition which is a simple equation given by:

$$MSE(\bar{y}) = E[(\bar{y}-y)^2] = Bias(\bar{y})^2 + Var(\bar{y}) \quad (24)$$

1.3. Evaluation Against Vision-Language Based IQA Methods

We evaluate STAPLE, trained on KonIQ-10K as $\mathcal{D}_S$ and $\mathcal{D}_T$ with Gaussian Noise, Pixelation, and Impulse Noise combined (as described in second part of section 5.3 from main paper). While vision-language models like Q-Align, LIQE, and CLIP-IQA leverage both image and text supervision, STAPLE remains purely vision-based. Despite this, STAPLE achieves competitive performance as shown in Figure 1. Notably, while Q-Align performs best overall, STAPLE consistently outperforms or matches models like CLIP-IQA and LIQE, demonstrating its strong generalization capability even without language supervision.

Additionally in Figure 2, we evaluate all methods in a real-world application like perceptual quality assessment of super-resolved images generated via diffusion (as described in Section 5.4 from main paper). Consistent with earlier results, STAPLE performs competitively and tracks perceptual progression more reliably across diffusion steps than some of the vision-language methods. This further reinforces STAPLE's robustness in challenging and practical deployment scenarios, even without leveraging textual guidance.

1.4. Algorithms

Algorithm 1 represents STN baseline with no pseudo-label training. While Algorithm 2 represents STAPLE training. The blue indicates the steps required for pseudo-label generation. In Algorithm 3, we show algorithm at inference using $ref = 1$ reference image.

Algorithm 1 Training STN Model (No pseudo label)

Input: Number of iterations N, labeled dataset D, Initialize STN f_θ
Output: f_θ
Set: STN to Train mode $\rightarrow f_\theta$
for N iterations **do**
  Sample $(x_1, y_1), (x_2, y_2) \sim D$
  $L_{Lb} = \mid f_\theta(x_i, x_j) - (y_i - y_j) \mid$
  $L = L_{Lb}$
  $\theta \leftarrow \theta - \eta \nabla_\theta L_{Lb}$
end for
return: f_θ

Algorithm 2 STN Model with Pseudo label

Input Number of iterations N, labeled dataset D, unlabeled dataset U , Initialize STN f_θ
Output: f_θ
for N iterations **do**
  Sample $(ux_1) \sim U$
  Set: STN to eval mode $\rightarrow f_\theta^{'}$
  Calculate Pseudo label $\tilde{up}_i$ for ux_1
  for reference t = 1 to T **do**
    Sample $(x_t, y_t) \sim D$
    $y_{p_i,t} = f_\theta(ux_1, x_{ref,t}) + y_{ref,t}$
  end for
  $\tilde{up}_i = \frac{1}{T} \sum_{t=1}^{T} y_{p_i,t}$
  Set: STN to Train mode $\rightarrow f_\theta$
  Sample $(x_1, y_1), (x_2, y_2)$ and $(x_3, y_3) \sim D$
  $L_{ULb} = \mid f_\theta(ux_1, x_3) - (\tilde{up}_i - y_3) \mid$
  $L_{Lb} = \mid f_\theta(x_1, x_2) - (y_1 - y_2) \mid$
  $L = L_{Lb} + \lambda L_{ULb}$
  $\theta \leftarrow \theta - \eta \nabla_\theta L$
end for
return: f_θ

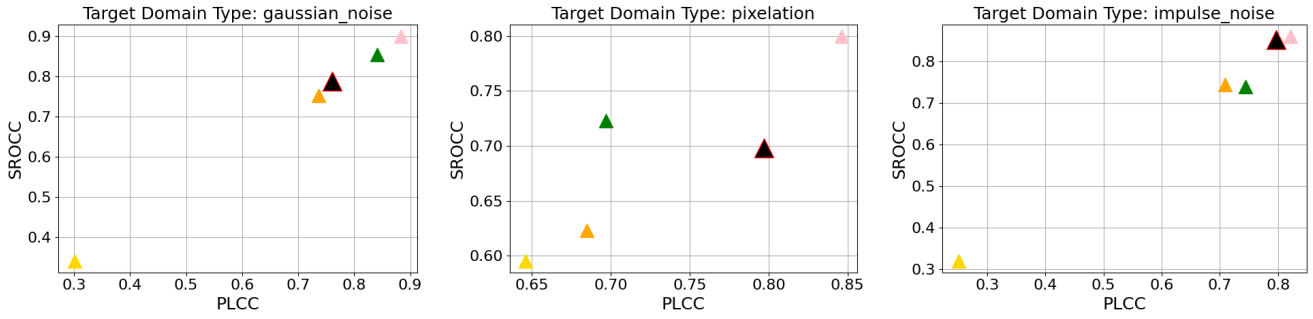


Figure 1. PLCC and SROCC scores for KonIQ-10K test set ($\mathcal{D}_S$) and Distortions from KADID-10K ($\mathcal{D}_T$). CLIPIQA (yellow), LIQE (orange), LIQEmix (green), Qalign (pink), STAPLE (black)

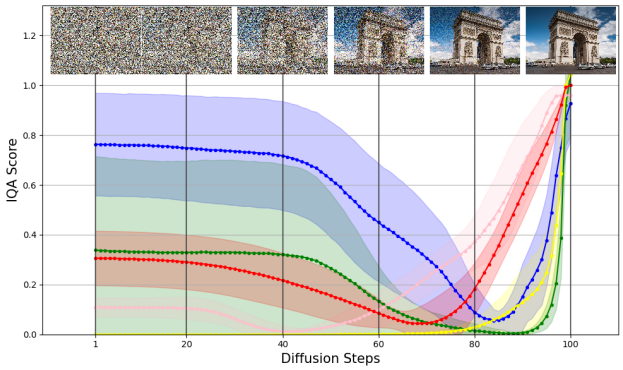


Figure 2. Performance of CLIPIQA (blue), LIQE (green), LIQEmix (yellow), Qalign (pink) and STAPLE(red) on Super-Resolution images generated using Stable Diffusion across 100 timesteps. Solid line indicates average and shaded area indicates standard deviation.

Algorithm 3 Inference of STN Model

Input x_{test}, $(x_{ref}, y_{ref}) \sim D$, trained STN f_θ
Set: STN to eval mode $\rightarrow f_\theta^{'}$
$y_{pred} = f_\theta^{'}(x_{test}, x_{ref}) + y_{ref}$

1.5. Experimental Setup for Scenario 1

As described in Section 5.1 of the main paper, Scenario 1 involves training STAPLE on six distortion types as $\mathcal{D}_S$ while reserving the seventh as $\mathcal{D}_T$, conducting a separate evaluation for each distortion type. This results in seven independent experiments. For all experiments, λ is initialized at 0.1 and increased by 0.1 every 5 epochs. To prevent over-fitting on the target domain, we randomly sample from both $\mathcal{D}_S$ training set and $\mathcal{D}_T$ during unsupervised learning, using a 6:4 sampling ratio. STAPLE is fine-tuned for 50 epochs using Stochastic Gradient Descent (SGD) with a learning rate of 0.0001 and a cosine scheduler.

1.6. Experimental Setup for Scenario 2

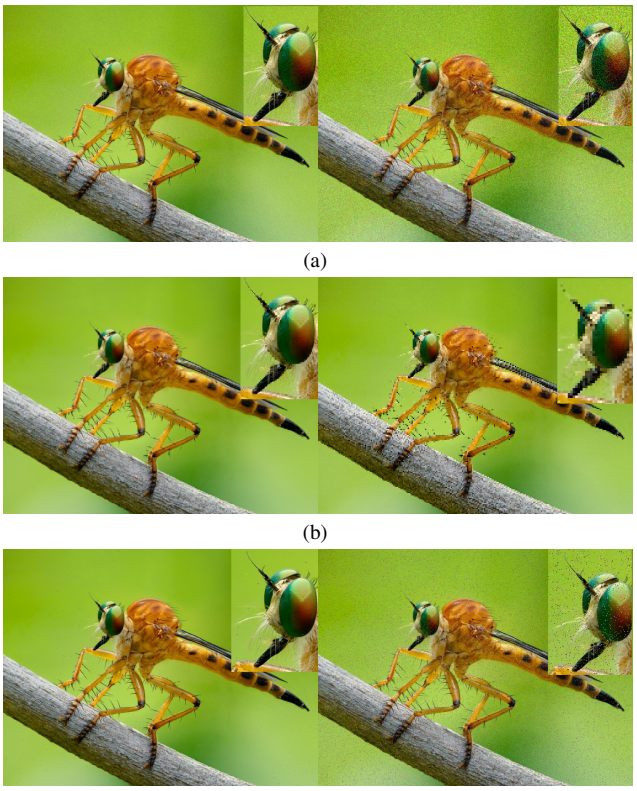
As described in Section 5.2 of the main paper, Scenario 2 involves training STAPLE on KADID-10K as $\mathcal{D}_S$ while utilizing KonIQ-10K, LIVEC, and BID as unlabeled $\mathcal{D}_T$ in three separate experiments, each running for 50 epochs. For KonIQ-10K, λ is kept steady at 0.5 for the first 20 epochs and then increased to 1.25 in increments of 0.25 every 10 epochs. The $\mathcal{D}_S$ to $\mathcal{D}_T$ sampling ratio is set to 6:4. Due to the smaller dataset size of BID, the sampling ratio is adjusted to 2:8. For LIVEC, λ is initialized at 0.25 for the first 10 epochs, then maintained at 0.5, with a 4:6 sampling ratio. The rest of the training strategy follows the approach used in Scenario 1.

1.7. Experimental Setup for Scenario 3

For Scenario 3, supervised learning (L_{Lb}) is conducted on 80% of the KonIQ-10K training set, while unsupervised learning (L_{ULb}) involves random sampling from either the 80% KonIQ-10K training set (GT labels discarded) or the simulated target dataset using an 8:2 sampling ratio. λ values for Gaussian Noise are initially set at 0.1, gradually increasing to 0.3 in increments of 0.1 every 10 epochs, before being decreased at the same rate. A similar trend is followed for Pixelation and Impulse Noise, but with λ adjusted from 0.1 to 0.5 in increments of 0.2. The training strategy for STN and STAPLE remains consistent with Scenario 1.

As outlined in Section 5.3 of the main paper, STAPLE is trained on a simulated $\mathcal{D}_T$ instead of using KADID-10K distortions directly, ensuring KADID-10K remains a separate test set. To generate this simulated dataset, we create an image-only dataset with three types of distortions: Gaussian Noise, Pixelation, and Impulse Noise. Specifically, we select 9% of the highest-quality images from KonIQ-10K, based on their GT MOS, and apply distortions at varying levels during training. Examples of minimum and maximum distortion levels for each type are shown in Figure 3.

Additionally, as mentioned in the main paper, we train



(a) (b) (c)

Figure 3. Minimum and maximum distortions for (a) Gaussian Noise (b) Pixelation (c) Impulse Noise

STAPLE on all three target distortions simultaneously. Figure 4 shows that STAPLE effectively handles multiple target domains, further improving accuracy across all distortions.

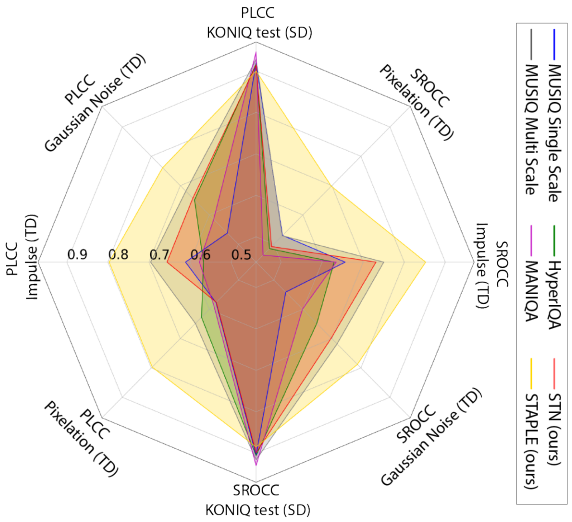


Figure 4. Performance when STAPLE is trained with $\mathcal{D}_{\mathcal{T}}$ consisting of multiple distortions

1.8. Simulated target distortions in Experiment Setup 1

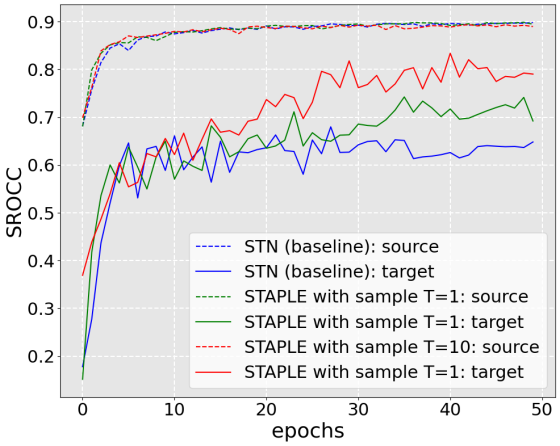


Figure 5. SROCC Comparison in performance of STN vs STNAPL throughout Training.

1.9. Effect of self-supervised Learning

In Fugure 5, we compare performance of STN to STAPLE with reference sample number T=1 and T=10 on SROCC metric. Here, we follow Setup 1 as mentioned in original paper and consider target domain as Gaussian Noise.