Appendix

A. Overview

This Appendix is structured as follows. First, more methodology details are described in Appendix B, followed by more ablation studies and qualitative results in Appendix C. Then, some extensions are illustrated in Appendix D. Finally, we discuss the limitations in Appendix E.

B. More Details

B.1. Details of Dataset Construction

Special designs for PIPAL dataset. The PIPAL dataset is annotated using pair-wise comparisons and Elo ratings, instead of the conventional five-point standard rating. As a result, variance information is not provided. To integrate the PIPAL dataset into our training framework, we assign a pseudo variance derived from other datasets. The statistics of the other three training datasets are shown in Tab. A1. Based on these statistics, we manually set the ratio of the mean standard deviation (std) to the score range for the PIPAL dataset as 20%. The minimum and maximum scores in the PIPAL dataset are 934.95 and 1835.99, respectively, thus the pseudo std is set as $20\% \times (1835.99 - 934.95) = 180.21$. Note that all scores are normalized to [1,5] during training, with the std / variance normalized accordingly.

Table A1. **Statistics** of score range and standard deviation (std).

Datasets	KonIQ	SPAQ	KADID
score range (max - min)	2.91	91.67	3.93
mean std	0.57	13.93	0.86
mean std / score range	19.73%	15.20%	21.90%

Degradation to linear interpolation when the variance is quite small. When the variance is extremely small, integration calculations can introduce errors. These errors can lead to strange solutions when directly solving the two constraint conditions for post-adjustment, resulting in α and β values that deviate significantly from 1 and 0, respectively. Consequently, the adjusted "probabilities" can become substantially smaller than 0 or larger than 1, which is unreasonable as the training label. Recall that the quality score, x, is modeled as a Gaussian distribution, $\mathcal{N}(\mu, \sigma^2)$. To address this chanllenge, when the variance is extremely small, we approximate the probability density function, $f(x)$, of the score's Gaussian distribution as a unit impulse function:

$$\lim_{\sigma \to 0} f(x) = \delta(x - \mu), \tag{A1}$$

where $\delta(\cdot)$ is the unit impulse function. In this case, the soft label is calculated through linear interpolation between the two nearest center points. Suppose that $c_j < \mu \leq c_{j+1}$, and recall that $d = c_{j+1} - c_j = 1$, the soft label is obtained as:

$$p_i = \begin{cases} c_{j+1} - \mu, & \text{if } i = j, \\ \mu - c_j, & \text{if } i = j+1, \\ 0, & \text{otherwise.} \end{cases} \tag{A2}$$

According to the 3σ rule, nearly all samples fall within the range $[\mu - 3\sigma, \mu + 3\sigma]$. Thus, if 3σ is less than half of the level interval, only the two nearest levels have non-zero probabilities. This criterion provides a way to define the threshold for small variance as $3\sigma \leq d/2$, which simplifies to $\sigma \leq 0.17$. To allow for a slightly relaxed threshold, the threshold for small (normalized) variance is set to $(0.2)^2$.

B.2. Details of Methodology

Model architecture. DeQA-Score adopts the architecture of mPLUG-Owl2 [75], structured as follows. Specifically, the input images and the question texts are first tokenized, then fused, finally processed by the Large Language Model (LLM) for response generation. (a) Tokenizing input images and texts. We use a pre-trained CLIP pre-trained ViT-L/14 [49] as the image encoder to convert the input images into visual tokens, with each token having a channel of 1024. The texts are tokenized into textual tokens using the SentencePiece tokenizer [27], with each token having a channel of 4096. To bridge the different embedding spaces of visual and textual tokens, a trainable image abstractor, which is a six-layer transformer network, is implemented to map the vision tokens to the hidden dimension of the LLM, which is 4096. The abstractor can also significantly reduce the number of vision tokens to 64, relieving the computing pressure. (b) Token fusion. We integrate the visual tokens into pre-defined positions within the textual tokens as token fusion. (c) Response generation using LLM. The fused tokens are fed into an LLM, which is LLaMA-2-7B [62], to generate the final response. The LLM can be either LoRA-tuned [22] or fully tuned, and their results are similar when the vision components are trainable, as shown in Tab. A7.

Examples of dataset variation. As discussed in the main paper, different IQA datasets have distinct distributions. To illustrate this, we present six images sampled from various IQA datasets in Fig. A1. Although these images share similar mean quality scores (*i.e.*, linearly normalized scores), they exhibit significantly different visual quality.

B.3. Details of Training and Inference

Training Setup. The pre-trained weights of mPLUG-Owl2 are used for model initialization. The loss weighting term is set to 0.05, bringing the two loss terms to roughly the same scale. We adopt AdamW [23] as the optimizer, with an initial learning rate of 2e-5 that gradually decays using a cosine decay strategy. A warmup strategy is applied with a warmup ratio of 0.03. Our model is trained with a batch size

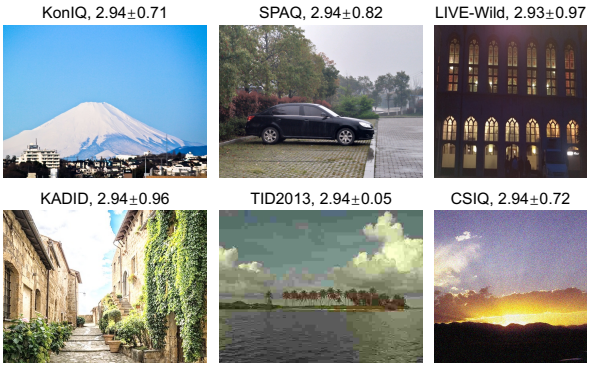

Figure A1. **Examples of dataset variation**. Images from different IQA datasets have similar mean quality scores (*i.e.*, linearly normalized scores), but exhibit significantly different visual quality. The image from KonIQ dataset has the best quality than others.

Table A2. **Probability sum of the five levels** when applying softmax function on all textual tokens.

Datasets	KonIQ	SPAQ	KADID	PIPAL	LIVE-Wild	AGIQA-3K
Prob. Sum	0.9998	0.9998	0.9996	0.9997	0.9997	0.9996

of 64 for 3 epochs. Both the vision encoder and abstractor are trainable, and the LLM is fully tuned unless specified. As shown in Tab. A7, LoRA-tuned LLM achieves comparable performance to fully-tuned LLM, providing an alternative for scenarios with limited computation resources. Using 8 RTX A6000 GPUs, training on the KonIQ dataset is completed in about 1.5 hours, while training on the KonIQ, SPAQ, and KADID datasets takes around 4 hours.

Explanation of closed-set softmax during inference. In the main paper, we follow Q-Align [70] by applying a closed-set softmax on the five levels to compute the probabilities p_i^{pred}, thereby avoiding the influence of other textual tokens. As shown in our statistics in Tab. A2, the probability sum of the five levels after training is very close to 1, indicating that the trained MLLM consistently predicts one of the five levels. Thus, applying softmax over the five levels or across all textual tokens yields nearly identical results.

C. More Results

Results of score distribution prediction with KL divergence as metric. In Tab. 5, we provide the distribution prediction results with JS divergence and Wasserstein distance as metrics. Here we further add the KL divergence as metric. The KL divergence between two Gaussian distributions, $p_1 = \mathcal{N}(\mu_1, \sigma_1^2)$, $p_2 = \mathcal{N}(\mu_2, \sigma_2^2)$, is calculated as:

$$\text{KL}(p_1||p_2) = \log(\frac{\sigma_2}{\sigma_1}) + \frac{\sigma_1^2 + (\mu_1 - \mu_2)^2}{2\sigma_2^2} - \frac{1}{2}. \quad \text{(A3)}$$

The results are given in Tab. A3, where the KL divergence of Q-Align is quite large. We explain this as follows. As

Table A3. **Score distribution prediction results** with KL divergence between the predicted Gaussian distribution and the human labeled Gaussian distribution as metric. Models are trained on the KonIQ, SPAQ, and KADID datasets. DeQA-Score achieves a much closer alignment with human annotations.

	KonIQ	SPAQ	KADID	LIVE-Wild	AGIQA-3K
Q-Align [70]	109.039	2.329	1229.530	3.980	76.414
DeQA-Score	**0.058**	**0.241**	**0.142**	**0.249**	**0.534**

Table A4. **Results on AI-generated images** with PLCC / SRCC metrics. The models are trained on KonIQ, SPAQ, and KADID.

	AIGCIQA2023	AGIN	AGIQA-1K	AGIQA-3K
Q-Align [70]	0.809 / 0.783	0.655 / 0.615	0.650 / 0.442	0.788 / 0.733
DeQA-Score	**0.826 / 0.799**	**0.674 / 0.626**	**0.715 / 0.514**	**0.808 / 0.745**

in Figs. 6 and A3 to A5, Q-Align tends to predict a single level token, bringing a quite small predicted variance. That means, the first variance, σ_1^2 in Eq. (A3), is quite small ($\to 0$), thus the KL divergence goes extremely large ($\to +\infty$).

Results on more AI-generated images. In the main paper, we have included an IQA dataset with AI-generated images, AGIQA-3K [30], for evaluation. Here we further provide evaluation results on three additional AIGC datasets including AIGCIQA2023 [65], AGIN [11], and AGIQA-1K [90] in Tab. A4. Both Q-Align and our DeQA-Score are trained on the KonIQ, SPAQ, and KADID datasets, and then directly evaluated on these unseen AIGC datasets. Our DeQA-Score consistently outperforms the baseline method.

More baselines on multi-dataset training. In the main paper, we primarily compare with Q-Align. Here we add the comparison results with more baselines including LIQE [89] and Compare2Score [98] in Tab. A5. Our DeQA-Score consistently outperforms these baselines.

Ablation studies on level number are conducted in Tab. A6. The performance improves with moderately larger level numbers (*i.e.*, 6∼8), as discretization becomes more accurate. However, the performance declines with much larger level numbers (*i.e.*, 10∼12), due to the increased difficulty in level prediction (more classification categories).

Ablation studies on training / fixing components are summarized in Tab. A7. The vision encoder and abstractor can be either fixed or trained, while the LLM can be LoRA-tuned [22] or fully tuned. First, comparing #0 with #1, or #3 with #4, training the vision abstractor significantly enhances performance. Second, similarly, from #1 to #2, or #4 to #5, training the vision encoder also leads to performance improvements. This improvement may be attributed to the fact that training either the vision encoder or abstractor helps extract more relevant features for IQA. Third, comparing #0 with #3, when the vision components are fixed, fully-tuned LLM shows a substantial advantage over LoRA-tuned LLM. Finally, from #1 to #4, or #2 to #5, when the vision components are trainable, fully-tuned LLM demonstrates

Table A5. **Score regression results of co-training on multiple IQA datasets** with the PLCC / SRCC metrics. The models are trained on the KonIQ, SPAQ, and KADID datasets.

Methods	KonIQ	SPAQ	KADID	PIPAL	LIVE-Wild	AGIQA-3K	TID2013	CSIQ
LIQE [89]	0.907 / 0.922	0.916 / 0.921	0.928 / 0.929	**0.503** / 0.493	0.855 / 0.822	0.689 / 0.650	**0.881 / 0.854**	0.792 / 0.794
Compare2Score [98]	0.941 / 0.929	0.929 / 0.927	0.952 / 0.949	0.446 / 0.440	0.868 / 0.856	0.787 / 0.733	0.836 / 0.809	0.879 / 0.830
Q-Align [70]	0.945 / 0.938	0.933 / 0.931	0.935 / 0.934	0.409 / 0.420	0.887 / 0.883	0.788 / 0.733	0.829 / 0.808	0.876 / 0.845
DeQA-Score	**0.957 / 0.944**	**0.938 / 0.934**	**0.955 / 0.953**	0.495 / **0.496**	**0.900 / 0.887**	**0.808 / 0.745**	0.852 / 0.820	**0.900 / 0.857**

Table A6. **Ablation studies on level numbers** with PLCC / SRCC metrics. We use numerical names (one / two / three / four / five / ...) as level names because they are easy to extend to different numbers. The models are trained on the KonIQ, SPAQ, and KADID datasets.

Level Number	KonIQ	SPAQ	KADID	PIPAL	LIVE-Wild	AGIQA-3K	TID2013	CSIQ
5	0.945 / 0.932	0.934 / 0.930	0.955 / 0.952	**0.421 / 0.424**	0.873 / 0.857	0.766 / 0.712	**0.820 / 0.794**	**0.882 / 0.835**
6	0.947 / 0.935	0.935 / 0.931	**0.956 / 0.953**	0.404 / 0.408	0.879 / 0.867	**0.789 / 0.732**	0.808 / 0.789	0.877 / 0.830
7	0.947 / 0.935	0.935 / **0.932**	0.952 / 0.949	0.411 / 0.413	**0.883 / 0.869**	0.777 / 0.720	0.812 / 0.789	0.879 / 0.827
8	**0.949 / 0.936**	**0.936 / 0.932**	0.949 / 0.946	0.420 / 0.419	0.872 / 0.854	0.772 / 0.709	0.795 / 0.779	0.874 / 0.824
10	0.932 / 0.916	0.930 / 0.927	0.948 / 0.942	0.409 / 0.403	0.861 / 0.837	0.737 / 0.654	0.784 / 0.763	0.880 / 0.831
12	0.930 / 0.916	0.930 / 0.927	0.947 / 0.943	0.415 / 0.412	0.864 / 0.840	0.722 / 0.643	0.804 / 0.773	0.870 / 0.819

Table A7. **Ablation studies on training / fixing various model components** with PLCC / SRCC metrics. "Enc." means vision encoder, and "Abs." represents vision abstractor. "LLM LoRA" or "LLM Full" represents the LLM is LoRA-tuned [22] or fully tuned. "✔" means this component is trained. The models are trained on the KonIQ, SPAQ, and KADID datasets. The results show that training vision encoder and abstractor significantly improves the performance. When vision encoder and abstractor are trained, fully-tuned LLM only shows a slight advantage over LoRA-tuned LLM.

#	Enc.	Abs.	LLM LoRA	LLM Full	KonIQ	SPAQ	KADID	PIPAL	LIVE-Wild	AGIQA-3K	TID2013	CSIQ
0			✔		0.826 / 0.797	0.871 / 0.867	0.814 / 0.803	0.440 / 0.427	0.728 / 0.695	0.804 / 0.750	0.740 / 0.692	0.818 / 0.751
1		✔	✔		0.909 / 0.890	0.919 / 0.916	0.898 / 0.892	0.429 / 0.425	0.806 / 0.768	0.699 / 0.672	0.749 / 0.682	0.876 / 0.821
2	✔	✔	✔		0.955 / 0.942	**0.938 / 0.934**	0.953 / 0.950	0.479 / 0.473	0.898 / 0.884	**0.810 / 0.756**	0.849 / **0.824**	**0.900 / 0.861**
3				✔	0.889 / 0.867	0.913 / 0.910	0.873 / 0.866	0.429 / 0.417	0.784 / 0.747	0.762 / 0.711	0.743 / 0.670	0.843 / 0.759
4		✔		✔	0.911 / 0.893	0.920 / 0.916	0.905 / 0.899	0.439 / 0.429	0.819 / 0.781	0.764 / 0.705	0.765 / 0.699	0.878 / 0.820
5	✔	✔		✔	**0.957 / 0.944**	**0.938 / 0.934**	**0.955 / 0.953**	**0.495 / 0.496**	**0.900 / 0.887**	0.808 / 0.745	**0.852** / 0.820	**0.900** / 0.857

similar performance to LoRA-tuned LLM. This provides an alternative if the computation resources are limited.

Qualitative results are provided in Figs. A3 to A5. Q-Align, which is trained on one-hot labels, tends to predict a single label, resulting in higher KL divergence. Our DeQA-Score can predict the score distribution that aligns well with human annotations under wide circumstances:

- Different categories of distortions, including *in-the-wild* images with authentic distortions in Fig. A3, *artificial* distortions in Fig. A4, *AI-generated* images in Fig. A5.
- Multiple quality levels, such as, *poor* quality (Fig. A3f, Fig. A4ah), *fair* quality (Fig. A3e, Fig. A4g), and *high* quality (Fig. A3hj, Fig. A5c).
- Various image contents, including *animals* (Fig. A3e, Fig. A4bd), *humans* (Fig. A3fg, Fig. A5ad), *nature scenes* (Fig. A4agh), *urban scenes* (Fig. A3ch, Fig. A4f), *indoor scenes* (Fig. A3abd), and *sports* (Fig. A3i, Fig. A4c).

D. Extensions

Score regression helps quality description. We investigate whether score regression tasks can enhance general low-level perception tasks, *i.e.*, language-based quality description tasks. The low-level perception tasks are evaluated on Q-Bench [68]. For score regression, we adopt the same training techniques as in the main paper, and train the model on the KonIQ, SPAQ, and KADID datasets. The low-level perception tasks are trained using the next token prediction paradigm, with Q-Instruct [69] as the training dataset. Q-Instruct [69] has shown that co-training with or pre-training on high-level tasks can improve performance on low-level perception tasks. Similarly, we evaluate two strategies: (a) co-training with score regression tasks, and (b) pre-training on score regression tasks. Some questions in the Q-Instruct dataset are quite similar to the questions of score regression, which may confuse the model. Therefore, we append the questions of score regression with a specific suffix, "Answer the question with levels.", to specify the task.

The experimental results are presented in Tab. A8. First, comparing #0 with #3 & #4, both co-training with and pre-training on score regression tasks consistently improve low-level perception performance. Second, comparing #1 with #3, or #2 with #4, the benefits from score regression tasks are greater than those from high-level tasks, likely because score regression is more closely related to low-level perception. Finally, from #3 to #4, pre-training on score regression tasks achieves better results than co-training.

Table A8. **Low-level perception results on Q-Bench [68]** of co-training with or pre-training on score regression tasks. The results of #0, #1, and #2 are borrowed from [69]. Co-training with or pre-training on score regression tasks stably improves the performance.

#	Method	Yes / No	What	How	Distortion	Other	IC Distortion	IC Other	Overall
0	From the scratch [69]	0.7218	0.5796	0.5619	0.5668	0.6921	0.5329	0.7265	0.6161
1	High-level co-training [69]	0.7564	0.6704	0.5903	0.7101	0.6528	0.6316	0.6980	0.6756
2	High-level pre-training [69]	0.7600	0.6504	0.6166	0.6595	0.6875	0.6546	0.7388	0.6796
3	Score co-training	0.7727	0.6615	0.6308	0.6965	0.6875	0.6349	0.7633	0.6925
4	Score pre-training	**0.7927**	**0.7323**	**0.6410**	**0.7276**	**0.7060**	**0.6974**	**0.7837**	**0.7258**

Table A9. **Co-training with Q-Instruct [69] leads to an obvious reduction in score regression performance**. Exploring better strategies to combine language-based quality description tasks with score regression tasks is left as our future work.

	KonIQ	SPAQ	KADID	PIPAL	LIVE-Wild	AGIQA-3K	TID2013	CSIQ
Co-training with Q-Instruct	0.916 / 0.926	0.870 / 0.816	0.812 / 0.810	0.286 / 0.289	0.825 / 0.812	0.778 / 0.732	0.680 / 0.656	0.798 / 0.790
Only score regression	**0.957 / 0.944**	**0.938 / 0.934**	**0.955 / 0.953**	**0.495 / 0.496**	**0.900 / 0.887**	**0.808 / 0.745**	**0.852 / 0.820**	**0.900 / 0.857**

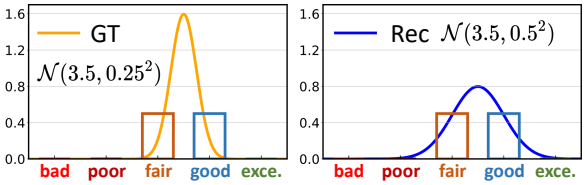

Figure A2. **Illustration of discretization errors when the variance is quite small**. A score distribution, $\mathcal{N}(3.5, 0.25^2)$, is discretized into a soft label, where both "fair" and "good" share a probability of 0.5. However, the recovered distribution from this soft label becomes $\mathcal{N}(3.5, 0.5^2)$, resulting in a larger variance and, consequently, a flatter curve of the distribution density function.

Quality description harms score regression. Considering the promising results in Tab. A8, we aim to explore whether it is possible to co-train a model for both accurate score regression and language-based quality description. We therefore evaluate the score regression results of the co-trained model in Tab. A9. These preliminary results indicate that co-training with the instruction-tuning dataset, Q-Instruct, leads to a noticeable decrease in score regression. The reason can be that, in quality description datasets, the words to describe the quality levels are very diverse, greatly beyond the pre-defined five levels for score regression. This may confuse the model when predicting the level tokens.

E. Limitations and Future Works

Simple co-training with quality description cannot improve score regression. As shown in Appendix D, score regression tasks can enhance quality description results, while co-training with quality description tasks reduces the score regression performance. How to better co-train these two tasks remains an open question. However, first, the performance of the co-training model is still reasonable and higher than many non-MLLM-based IQA methods. Second, with the rapid development of MLLM-based quality description research, better quality description datasets may help.

Our discretization introduces errors when the variance is very small. While our discretization method effectively handles most distributions, it still introduces errors in the variance of the distribution when the variance is very small. For instance, as shown in Fig. A2, a score distribution with a small variance, $\mathcal{N}(3.5, 0.25^2)$, is discretized into a soft label where both "fair" and "good" share a probability of 0.5. The recovered distribution from this soft label becomes $\mathcal{N}(3.5, 0.5^2)$, resulting in a larger variance and a flatter curve in the probability density function. However, samples with extremely small variance are rare, *e.g.*, only 0.29% of samples in the KonIQ dataset have a variance smaller than 0.5^2. Thus, the overall influence of this issue is relatively small. How to better preserve the distribution characteristics during discretization for such cases is our future work.

High memory consumption. As our DeQA-Score is based on an MLLM with 7B parameters, it requires 15.2G storage space (in bfloat16 format) and 15.8G (batch size 1) / 23.0G (batch size 64) CUDA memory for inference. Despite this, it remains deployable on consumer GPUs like 4090, and the memory consumption can be reduced through quantization.

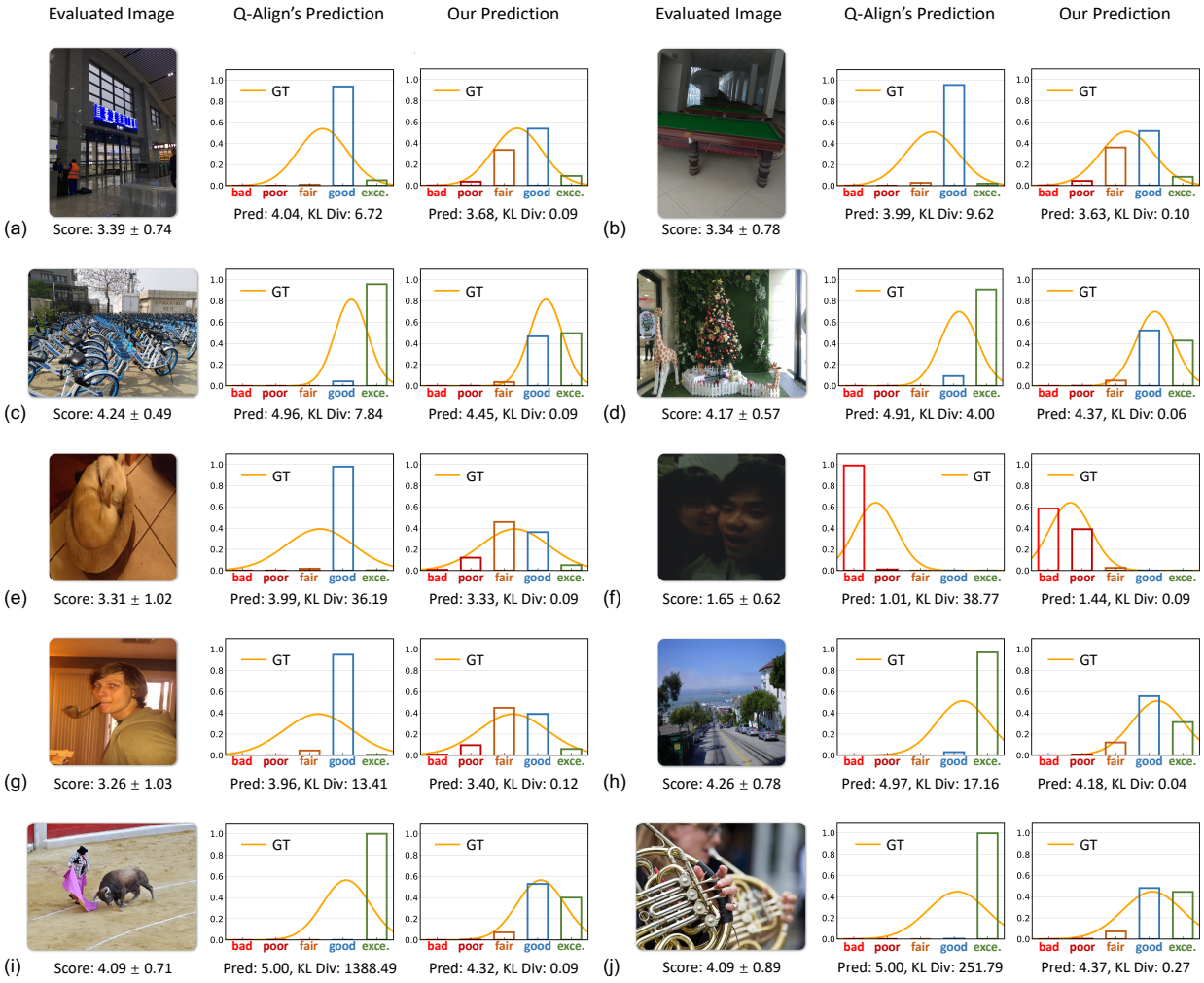

Figure A3. **Qualitative results** on in-the-wild IQA datasets, sampled from the KonIQ, SPAQ, and LIVE-Wild datasets.

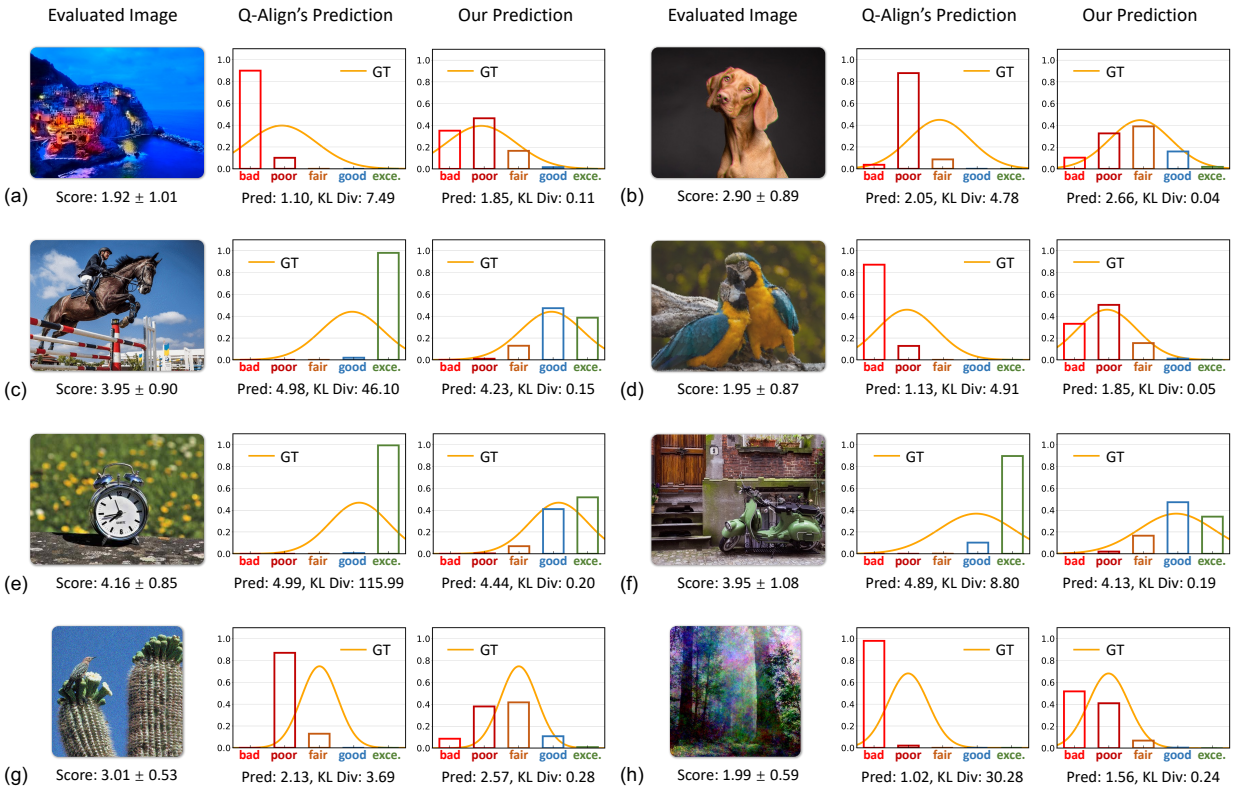

Figure A4. **Qualitative results** on IQA datasets with artificial distortions, sampled from the KADID and CSIQ datasets.

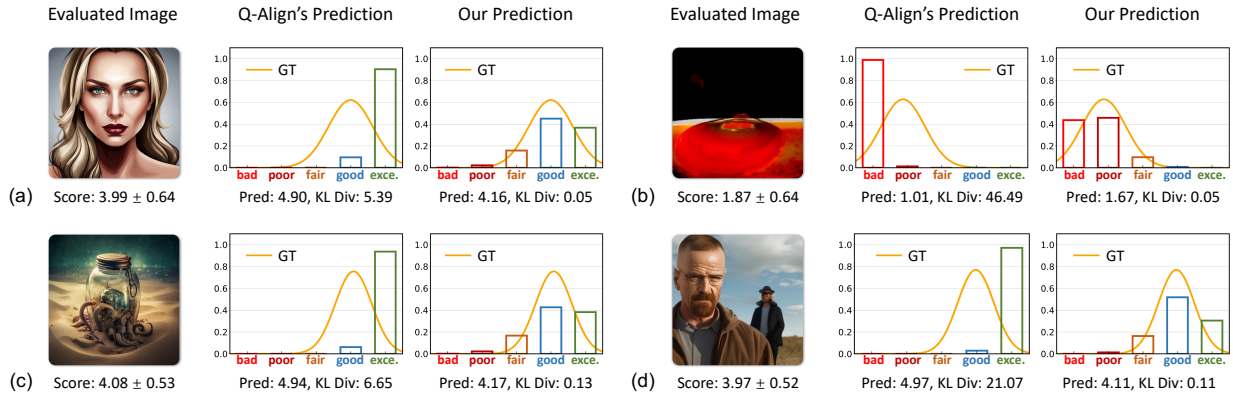

Figure A5. **Qualitative results** on IQA datasets with AI-generated images, sampled from the AGIQA-3K dataset.