Blindly Assess Image Quality in the Wild Guided by A Self-Adaptive Hyper Network

1. Introduction

- IQA 목표
  - 사람이 보는 것과 유사하게 컴퓨터가 image quality를 판단
- Synthetically distorted image에 대한 평가는 잘함.
- 실제로는 NR-IQA가 중요하다.
- NR-IQA가 어려운 이유
  - 부분적인 distortion이 많다.
  - 기존의 모델은 classification과 image quality를 같이 판단한다.

2. Related Work

- 2.1. IQA for Synthetically Distorted Images
  - hand-crafted feature based IQA
    - require expertly design and are time-consuming
    - Global view에서만 image quality를 측정.
  - learning feature based IQA
    - CNN 이용해서 성능이 좋아짐.
    - synthetic databases에만 성능이 좋다.
    - content variation, diverse distortion types 고려 안함.

2. Related Work

- 2.2. IQA for Authentically Distorted Images
  - semantic features가 image quality에 영향을 주는 것처럼 보임.
  - 최근 모델은 semantic features를 quality prediction에 사용.
  - 두 가지 단점이 존재.
    - Image semantics와 quality perception 관계를 고려 안함.
    - (사람은 image content 판단 후 quality를 판단)
    - global scale에서 image feature를 정하기 때문에 local distortion을 판단 못함.
  - image semantic features are learned first,
  - quality is predicted based upon what content the image delivers.

3. Proposed Method

Hyper IQA

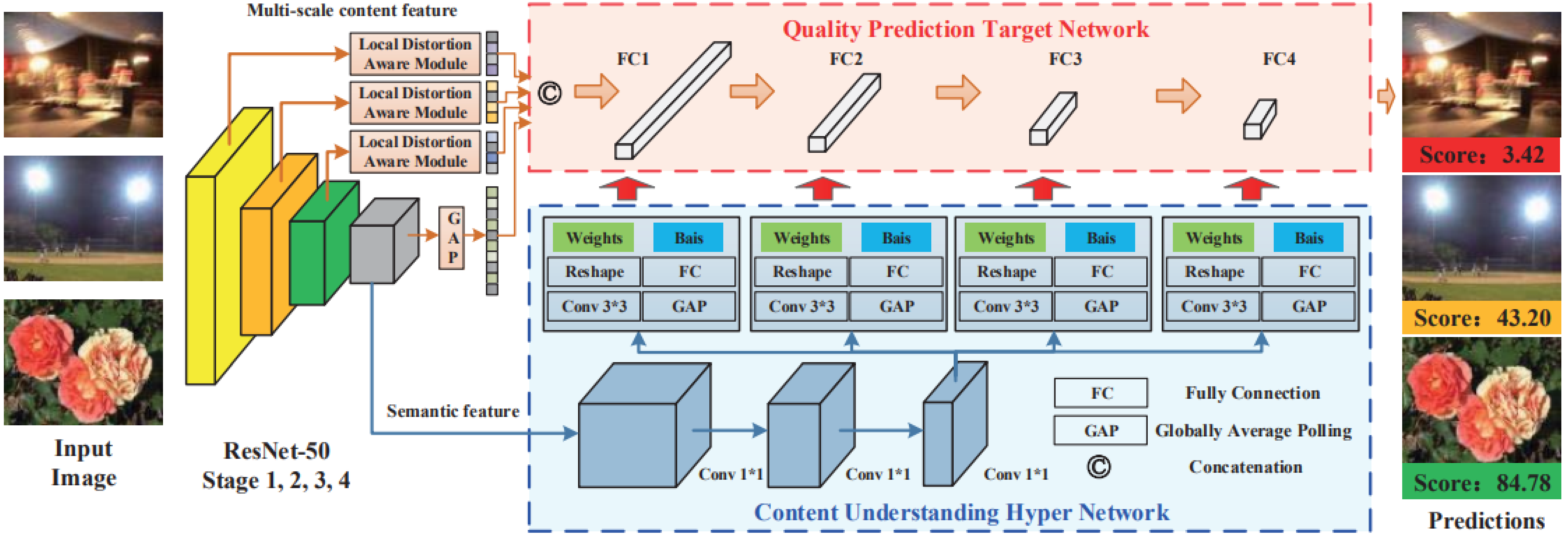


Figure 2. The pipeline of the proposed network. Given an image, we first extract semantic features from the basic model ResNet50, and import them to a hyper network which generates weights for a quality prediction target network. The input of the quality prediction target network is from aggregating multi-scale content features of the image, capturing both local and global distortions. In our module, the hyper network plays the role of formulating quality perception rule according to image content, and the target network makes quality prediction based on what an image specifically exhibits.

3. Proposed Method

- 3.1. Self-Adaptive IQA Model

$$\varphi(\boldsymbol{x}, \theta) = q,$$

- Φ = network model
- X = input image
- Θ = weight parameters

$$\varphi(\boldsymbol{x}, \theta_{\boldsymbol{x}}) = q,$$

- Θx = network parameters
  (dependent on the image itself instead of being fixed for all inputs)

$$\theta_{\boldsymbol{x}} = H(S(\boldsymbol{x}), \gamma),$$

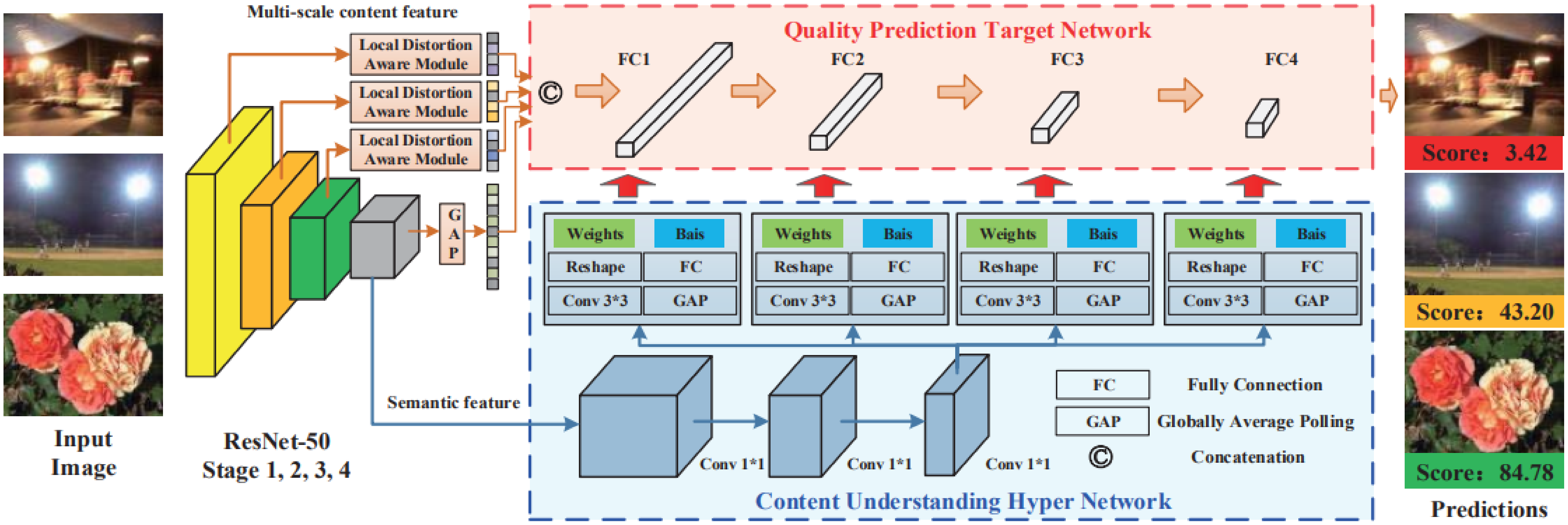
- H = a hyper network mapping function
- γ = hyper network parameters
- S(x) = semantic features extracted from the input image x

3. Proposed Method

- 3.1. Self-Adaptive IQA Model

$$\varphi(\boldsymbol{v_x}, H(S(\boldsymbol{x}), \gamma)) = q.$$

  - content aware vector Vx = Sms(x)



3. Proposed Method

- 3.2. Semantic Feature Extraction Network

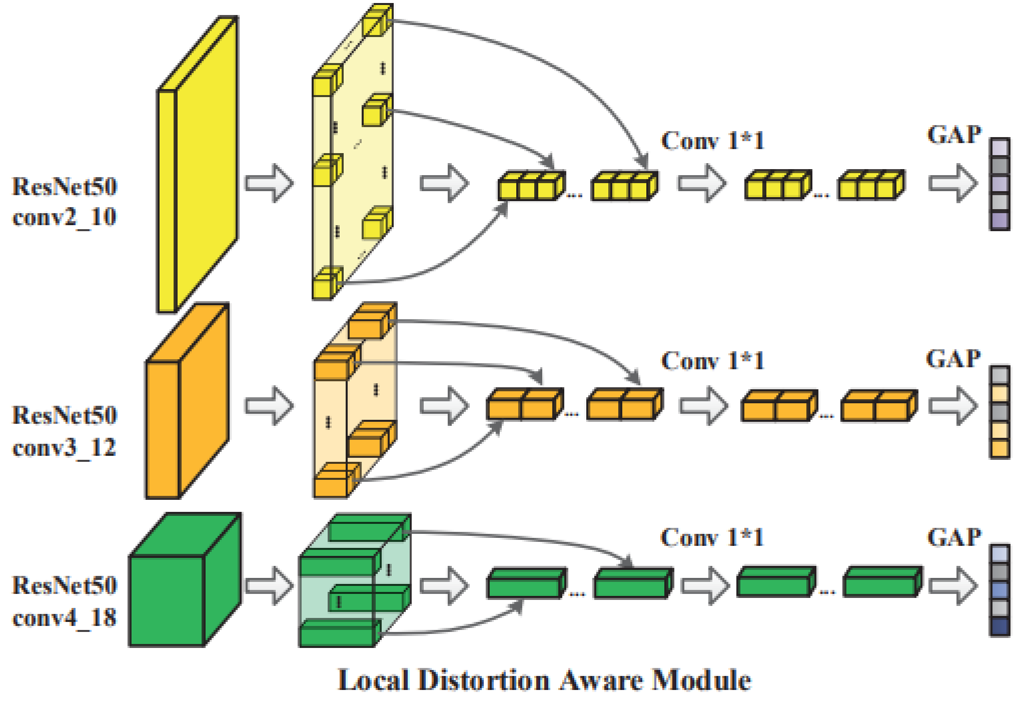
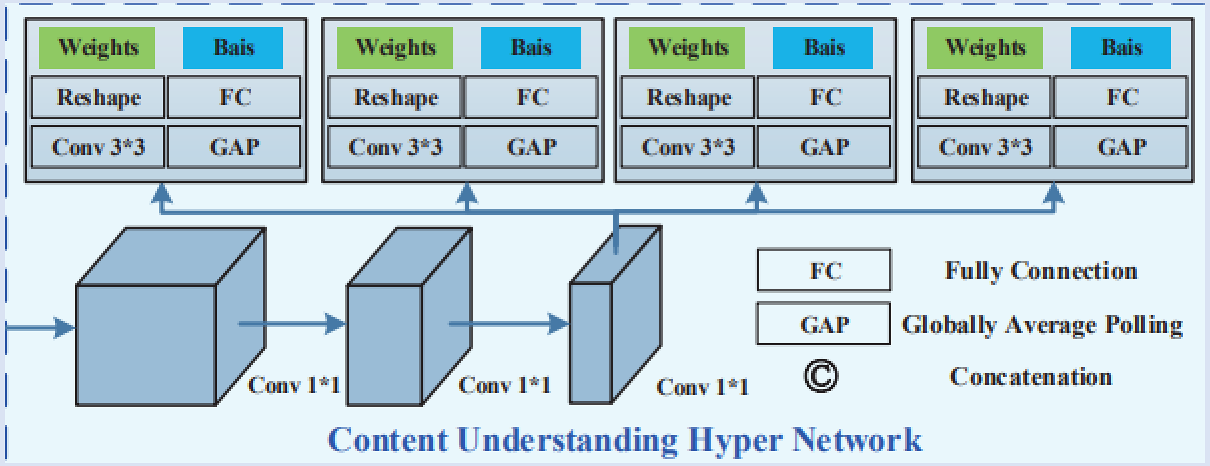


Figure 3. The architecture of the proposed local distortion aware module.

- ResNet50
- S(x) (semantic feature)
  - directly fed to hyper network for weight generation
- Sms(x) (multi-scale content feature)
  - input of the target network

3. Proposed Method

- 3.3. Hyper Network for Learning Perception Rule



- consists of three 1×1 convolution layers
- several weight generating branches
- fully connected layers are used as basic target network component

- The generated weights are regarded as the rule of perceiving image quality

3. Proposed Method

- 3.4. Target Network for Quality Prediction

- the function of the target network is simply mapping learned image contents to a quality score.
- consists of four fully connected layers
- sigmoid function as the activation function

3. Proposed Method

- 3.5. Implementation Details

$$\ell = \frac{1}{N}\sum_{i}^{N} ||\varphi(\boldsymbol{v}_{\boldsymbol{p_i}}, H(S(\boldsymbol{p_i}), \gamma)) - Q_i||_1,$$

- randomly sample and horizontally flipping 25 patches with size 224×224 pixels from each training image for augmentation.

- pi = i-th training patch
- Qi = ground truth score
- Adam optimizer with weight decay 5 × 10−4 to train our model for 15 epochs
- mini-batch size = 96
- Learning rate = 2 × 10−5 reduced by 10 after every 5 epochs.
- Xavier initialized

4. Experiments

- 4.1. Datasets
  - LIVE Challenge (LIVEC)
    - 1162 images
    - complex and composite distortions
  - KonIQ-10k
    - 10073 images
    - sense of brightness, colorfulness, contrast and sharpness
  - BID
    - 586 images
    - realistic blur distortions such as motion blur and out of focus
  - LIVE and CSIQ
    - 779 and 866 synthetically distorted images

4. Experiments

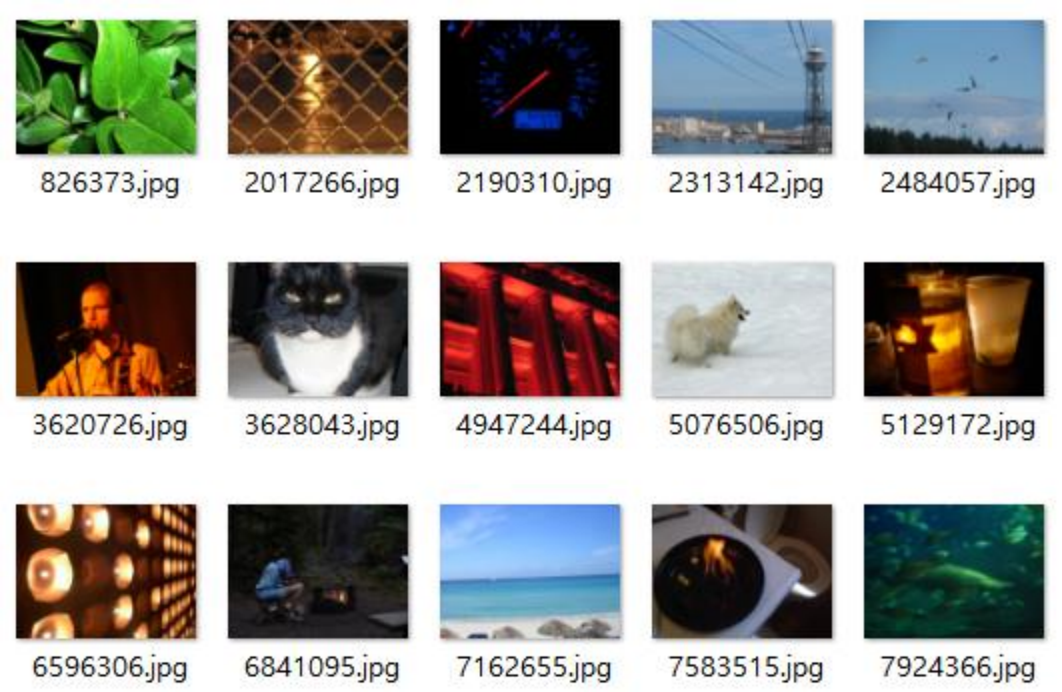
- 4.1. Datasets
  - KonIQ-10k
    - 10073 images
    - sense of brightness, colorfulness, contrast and sharpness

TABLE 1: Comparison of existing IQA databases with KonIQ-10k.

Database	Year	Content	No. of distorted images	Distortion type	No. of distortion types	No. of rated images	Ratings per image	Environment
IVC [17]	2005	10	185	artificial	4	185	15	lab
LIVE [18]	2006	29	779	artificial	5	779	23	lab
TID2008 [19]	2009	25	1,700	artificial	17	1,700	33	lab
CSIQ [20]	2009	30	866	artificial	6	866	5∼7	lab
TID2013 [21]	2013	25	3,000	artificial	24	3,000	9	lab
CID2013 [22]	2013	8	474	authentic	12∼14	480	31	lab
LIVE-itW [23]	2016	1,169	1,169	authentic	N/A	1,169	175	crowdsourcing
Waterloo Exploration [24]	2016	4,744	94,880	artificial	4	0	0	lab
MDID [25]	2017	20	1,600	artificial	5	1,600	33∼35	lab
KADID-10k [26]	2019	81	10,125	artificial	25	10,125	30	crowdsourcing
KonIQ-10k	2018	10,073	10,073	authentic	N/A	10,073	120	crowdsourcing

4. Experiments

- 4.1. Datasets
  - KonIQ-10k
    - 10073 images
    - sense of brightness, colorfulness, contrast and sharpness



4. Experiments

- 4.1. Datasets
  - KonIQ-10k
    - 10073 images
    - sense of brightness, colorfulness, contrast and sharpness

image_id	brightness	contrast	colorfulness	sharpness	quality_factor	bitrate	hxw	deep_feature
826373	0.390904834	0.253917609	0.432910646	22.76193439	79	0.576666667	3145728	100
2017266	0.268971143	0.184486318	0.274606168	15.92623475	73	0.247070313	3145728	10
2190310	0.046951947	0.126259043	0.096060146	8.148866586	90	0.121119748	5038848	42
2704811	0.44946323	0.310170134	0.365717795	7.452023631	97	0.340017731	4255763	188

image_name	c1	c2	c3	c4	c5	c_total	MOS	SD	MOS_zscore
10004473376.jpg	0	0	25	73	7	105	3.828571429	0.527277894	77.38362069
10007357496.jpg	0	3	45	47	1	96	3.479166667	0.580003025	68.72857143
10007903636.jpg	1	0	20	73	2	96	3.78125	0.527219619	78.62857143

4. Experiments

- 4.1. Datasets
  - KonIQ-10k
    - 10073 images
    - sense of brightness, colorfulness, contrast and sharpness

```python
class Koniq_10kFolder(data.Dataset):

    def __init__(self, root, index, transform, patch_num):
        imgname = []
        mos_all = []
        csv_file = os.path.join(root, 'koniq10k_scores_and_distributions.csv')
        with open(csv_file) as f:
            reader = csv.DictReader(f)
            for row in reader:
                imgname.append(row['image_name'])
                mos = np.array(float(row['MOS_zscore'])).astype(np.float32)
                mos_all.append(mos)

        sample = []
        for i, item in enumerate(index):
            for aug in range(patch_num):
                sample.append((os.path.join(root, '1024x768', imgname[item]), mos_all[item]))

        self.samples = sample
        self.transform = transform
```

4. Experiments

- 4.2. Evaluation Metrics
  - Spearman's rank order correlation coefficient (SRCC)

```
scipy.stats.spearmanr(a, b=None, axis=0, nan_policy='propagate', alternative='two-sided')
```

  - Pearson's linear correlation coefficient (PLCC)
  - 80% images are used for training
  - rest 20% are used for testing
  - synthetic image databases LIVE and CSIQ
  - run 10 and the median SRCC and PLCC values

4. Experiments

FastCampus For Business

Hyper IQA

- 4.2. Evaluation Metrics

```
def test(self, data):
    """Testing"""
    self.model_hyper.train(False)
    pred_scores = []
    gt_scores = []

    for img, label in data:
        # Data.
        img = torch.tensor(img.cuda())
        label = torch.tensor(label.cuda())

        paras = self.model_hyper(img)
        model_target = models.TargetNet(paras).cuda()
        model_target.train(False)
        pred = model_target(paras['target_in_vec'])

        pred_scores.append(float(pred.item()))
        gt_scores = gt_scores + label.cpu().tolist()

    pred_scores = np.mean(np.reshape(np.array(pred_scores), (-1, self.test_patch_num)), axis=1)
    gt_scores = np.mean(np.reshape(np.array(gt_scores), (-1, self.test_patch_num)), axis=1)
    test_srcc, _ = stats.spearmanr(pred_scores, gt_scores)
    test_plcc, _ = stats.pearsonr(pred_scores, gt_scores)

    self.model_hyper.train(True)
    return test_srcc, test_plcc
```

4. Experiments

- 4.3. Comparison with the State-of-the-art Methods

Table 1. Overall performance evaluation on five image databases.

SRCC	LIVEC	BID	KonIQ	LIVE	CSIQ
BRISQUE [29]	0.608	0.562	0.665	0.939	0.746
ILNIQE [3]	0.432	0.516	0.507	0.902	0.806
HOSA [37]	0.640	0.721	0.671	0.946	0.741
BIECON [15]	0.595	0.539	0.618	0.961	0.815
WaDIQaM [2]	0.671	0.725	0.797	0.954	**0.955**
SFA [22]	0.812	0.826	0.856	0.883	0.796
PQR [44]	0.857	0.775	0.880	0.965	0.873
DBCNN [46]	0.851	0.845	0.875	**0.968**	0.946
Ours	**0.859**	**0.869**	**0.906**	0.962	0.923

PLCC	LIVEC	BID	KonIQ	LIVE	CSIQ
BRISQUE [29]	0.629	0.593	0.681	0.935	0.829
ILNIQE [3]	0.508	0.554	0.523	0.865	0.808
HOSA [37]	0.678	0.736	0.694	0.947	0.823
BIECON [15]	0.613	0.576	0.651	0.962	0.823
WaDIQaM [2]	0.680	0.742	0.805	0.963	**0.973**
SFA [22]	0.833	0.840	0.872	0.895	0.818
PQR [44]	0.882	0.794	0.884	0.971	0.901
DBCNN [46]	0.869	0.859	0.884	**0.971**	0.959
Ours	**0.882**	**0.878**	**0.917**	0.966	0.942

4. Experiments

- 4.3. Comparison with the State-of-the-art Methods

Table 2. SRCC comparisons on individual distortion types on the LIVE and CSIQ databases.

Database	LIVE					CSIQ					
Type	JP2K	JPEG	WN	GB	FF	WN	JPEG	JP2K	FN	GB	CC
BRISQUE [29]	0.929	0.965	**0.982**	**0.964**	0.828	0.723	0.806	0.840	0.378	0.820	0.804
ILNIQE [3]	0.894	0.941	0.981	0.915	0.833	0.850	0.899	0.906	0.874	0.858	0.501
HOSA [37]	0.935	0.954	0.975	0.954	**0.954**	0.604	0.733	0.818	0.500	0.841	0.716
BIECON [15]	0.952	**0.974**	0.980	0.956	0.923	0.902	**0.942**	0.954	0.884	0.946	0.523
WaDIQaM [2]	0.942	0.953	**0.982**	0.938	0.923	**0.974**	0.853	0.947	0.882	**0.979**	**0.923**
PQR [44]	0.953	0.965	0.981	0.944	0.921	0.915	0.934	0.955	0.926	0.921	0.837
DBCNN [46]	**0.955**	0.972	0.980	0.935	0.930	0.948	0.940	0.953	**0.940**	0.947	0.870
Ours	0.949	0.961	**0.982**	0.926	0.934	0.927	0.934	**0.960**	0.931	0.915	0.874

4. Experiments

- 4.3. Comparison with the State-of-the-art Methods

Table 3. SRCC evaluations on cross database tests.

Training	Testing	PQR	DBCNN	Ours
LIVEC	BID	0.714	**0.762**	0.756
	KonIQ	0.757	0.754	**0.772**
BID	LIVEC	0.680	0.725	**0.770**
	KonIQ	0.636	**0.724**	0.688
KonIQ	LIVEC	0.770	0.755	**0.785**
	BID	0.755	0.816	**0.819**
LIVE	CSIQ	0.719	**0.758**	0.744
CSIQ	LIVE	0.922	0.877	**0.926**

Table 4. D-Test, L-Test and P-Test results on the Waterloo Exploration Database.

Model	D-Test	L-Test	P-Test
BRISQUE [29]	0.9204	0.9772	0.9930
GM-Log [38]	0.9203	0.9106	0.9748
CORNIA [43]	0.9290	0.9764	0.9947
HOSA [37]	0.9175	0.9647	0.9983
dipIQ [26]	0.9346	**0.9846**	**0.9999**
deepIQA [2]	0.9074	0.9467	0.9628
MEON [27]	0.9384	0.9669	0.9984
Two Stream CNN [41]	0.9301	0.9765	0.9952
DB-CNN [46]	**0.9387**	0.9527	0.9984
Ours	0.9006	0.9747	0.9971

4. Experiments

- Waterloo Exploration Database
  - 기존의 IQA 데이터베이스
    - 너무 제한된 content variation
  - 4744개의 원본 이미지
  - 94880개의 왜곡 이미지
  - 원본 이미지는 4개의 왜곡 유형
  - 5개의 왜곡 레벨로 왜곡

COMPARISON OF IQA DATABASES

Database	# of Pristine Images	# of Distorted Images	Subjective Testing Methodology
LIVE [3]	29	779	single-stimulus continuous scale
TID2008 [8]	25	1,700	paired comparison
TID2013 [4]	25	3,000	paired comparison
CSIQ [9]	30	866	multi-stimulus absolute category
LIVE MD [10]	15	405	single-stimulus continuous scale
LIVE Challenge [11]	—	1,162	single-stimulus continuous scale with crowdsourcing
Waterloo Exploration	4,744	94,880	need-based

https://bskyvision.com/entry/%EB%85%BC%EB%AC%B8-%EC%A0%95%EB%A6%AC-Ma-Waterloo-Exploration-Database-New-Challenges-for-Image-Quality-Assessment-Models-TIP2017

4. Experiments

- Waterloo Exploration Database
  - D-test (pristine/distorted image discriminability test)
    - IQA 모델이 왜곡 이미지로부터 원본 이미지를 잘 분리했는지를 시험
  - L-test (listwise ranking consistency test)
    - IQA 모델이 같은 콘텐츠 및 같은 왜곡 유형이지만 다른 정도로 왜곡된 이미지들의 순위를 잘 매길 수 있는지를 시험
    - 같은 콘텐츠 및 같은 왜곡 유형이지만 다른 왜곡 정도를 가진 이미지들을 모아서 SRCC와 KRCC를 구한 것을 평균
  - P-test (pairwise preference consistency test)
    - 품질차이를 느낄 수 있는 이미지 쌍(quality-discriminable image pair, DIP)을 제시했을 때 IQA 모델이 더 나은 품질의 것을 잘 선택할 수 있는지를 시험

https://bskyvision.com/entry/%EB%85%BC%EB%AC%B8-%EC%A0%95%EB%A6%AC-Ma-Waterloo-Exploration-Database-New-Challenges-for-Image-Quality-Assessment-Models-TIP2017

4. Experiments

- 4.3. Comparison with the State-of-the-art Methods

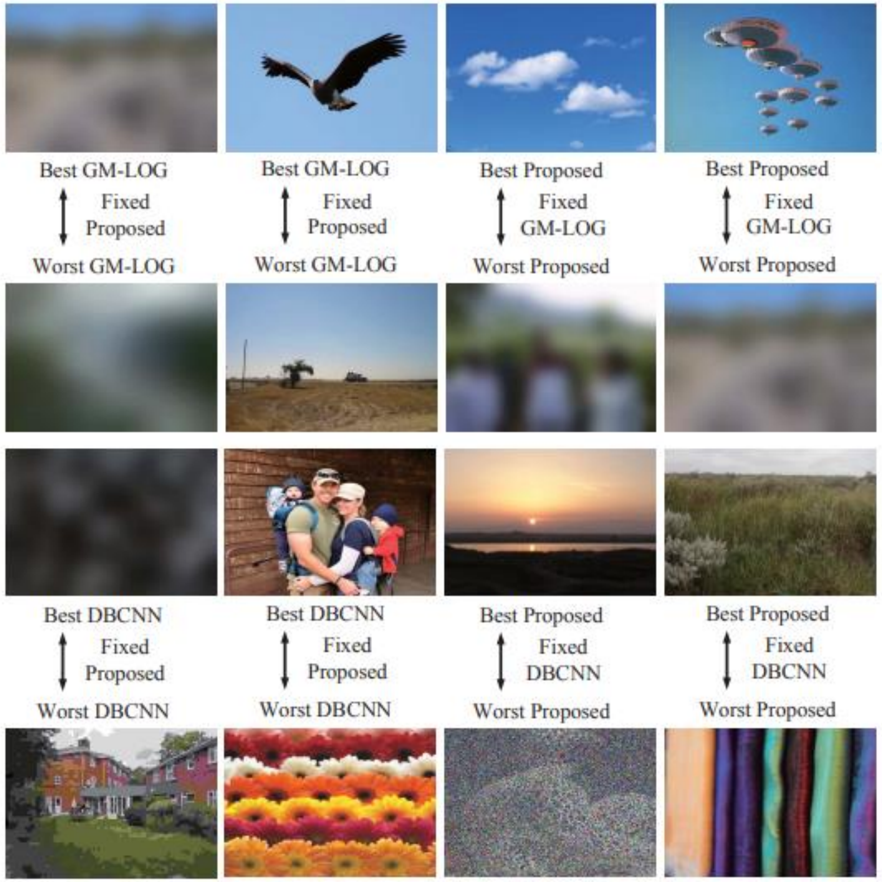


Figure 4. gMAD competition results on the Waterloo Database against GM-LOG [38] and DBCNN [46].

4. Experiments

- 4.4. Visualization of Self-Adaptive Weights

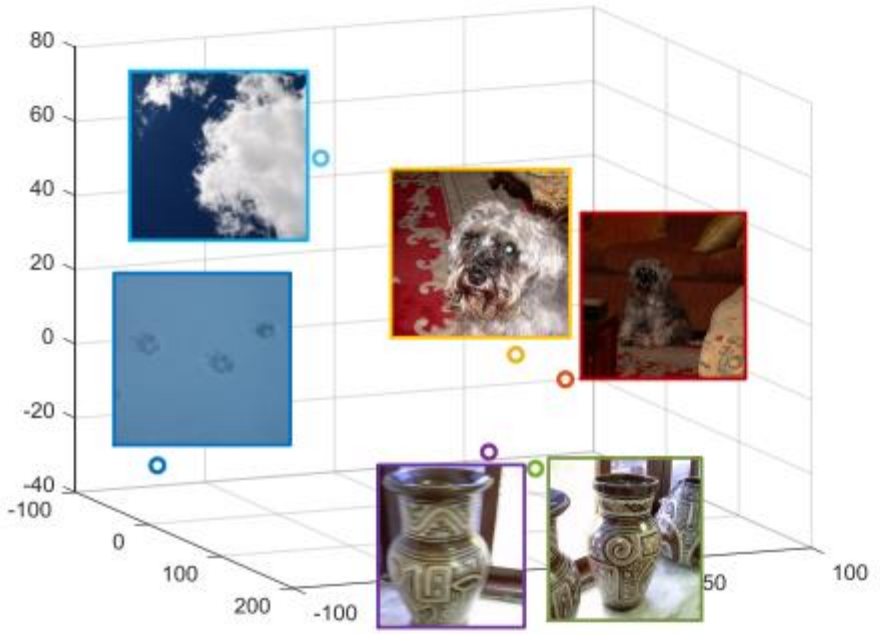


Figure 5. Generated weights of different images are plotted in the 3D space after PCA transformation. This figure shows the weights extracted from the first layer of the target network, weights from other layers also exhibits similar distribution.

4. Experiments

- 4.5. Ablation Study

Table 5. Ablation results on LIVE Challenge and LIVE databases.

Components	LIVE Challenge		LIVE	
	SRCC	PLCC	SRCC	PLCC
Res50	0.827	0.852	0.923	0.947
Res50+MS	0.836	0.859	0.954	0.963
Res50+Hyp	0.854	0.879	0.944	0.959
Res50+MS+Hyp	**0.859**	**0.882**	**0.962**	**0.966**

5. Conclusion

- 다른 모델이 image feature와 image quality를 함께 고려할 때,
- 우리 모델은 global image feature를 먼저 고려하고
- Local image feature를 image quality를 계산하는데 적용하여
- 사람이 image quality를 판단하는 기준과 유사한 모델을 만들었다.

5. Conclusion

FastCampus
For Business

Hyper
IQA

```python
pred_scores = []
for i in range(10):
    img = pil_loader(im_path)
    img = transforms(img)
    img = torch.tensor(img.cuda()).unsqueeze(0)
    paras = model_hyper(img)  # 'paras' contains the network weights conveyed to target network

    # Building target network
    model_target = models.TargetNet(paras).cuda()
    for param in model_target.parameters():
        param.requires_grad = False

    # Quality prediction
    pred = model_target(paras['target_in_vec'])  # 'paras['target_in_vec']' is the input to target net
    pred_scores.append(float(pred.item()))
score = np.mean(pred_scores)
# quality score ranges from 0-100, a higher score indicates a better quality
print('Predicted quality score: %.2f' % score)
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:5: UserWarning: To copy construct from a tenso
  """
Predicted quality score: 77.37
```