Convolutional Neural Networks for No-Reference Image Quality Assessment

Le Kang[1], Peng Ye[1], Yi Li[2], and David Doermann [1]

[1]University of Maryland, College Park, MD, USA
[2]NICTA and ANU, Canberra, Australia
[1]{lekang,pengye,doermann}@umiacs.umd.edu [2]yi.li@cecs.anu.edu.au

Abstract

In this work we describe a Convolutional Neural Network (CNN) to accurately predict image quality without a reference image. Taking image patches as input, the CNN works in the spatial domain without using hand-crafted features that are employed by most previous methods. The network consists of one convolutional layer with max and min pooling, two fully connected layers and an output node. Within the network structure, feature learning and regression are integrated into one optimization process, which leads to a more effective model for estimating image quality. This approach achieves state of the art performance on the LIVE dataset and shows excellent generalization ability in cross dataset experiments. Further experiments on images with local distortions demonstrate the local quality estimation ability of our CNN, which is rarely reported in previous literature.

1. Introduction

This paper presents a Convolutional Neural Network (CNN) that can accurately predict the quality of distorted images with respect to human perception. The work focuses on the most challenging category of objective image quality assessment (IQA) tasks: general-purpose No-Reference IQA (NR-IQA), which evaluates the visual quality of digital images without access to reference images and without prior knowledge of the types of distortions present.

Visual quality is a very complex yet inherent characteristic of an image. In principle, it is the measure of the distortion compared with an ideal imaging model or perfect reference image. When reference images are available, Full Reference (FR) IQA methods [14, 22, 16, 17, 19] can be applied to directly quantify the differences between distorted images and their corresponding ideal versions. State of the art FR measures, such as VIF [14] and FSIM [22], achieve a very high correlation with human perception.

However, in many practical computer vision applications there do not exist perfect versions of the distorted images, so NR-IQA is required. NR-IQA measures can directly quantify image degradations by exploiting features that are discriminant for image degradations. Most successful approaches use Natural Scene Statistics (NSS) based features. Typically, NSS based features characterize the distributions of certain filter responses. Traditional NSS based features are extracted in image transformation domains using, for example the wavelet transform [10] or the DCT transform [13]. These methods are usually very slow due to the use of computationally expensive image transformations. Recent development in NR-IQA methods – CORNIA [20, 21] and BRISQUE [9] promote extracting features from the spatial domain, which leads to a significant reduction in computation time. CORNIA demonstrates that it is possible to learn discriminant image features directly from the raw image pixels, instead of using handcrafted features.

Based on these observations, we explore using a Convolutional Neural Network (CNN) to learn discriminant features for the NR-IQA task. Recently, deep neural networks have gained researchers' attention and achieved great success on various computer vision tasks. Specifically, CNN has shown superior performance on many standard object recognition benchmarks [6, 7, 4]. One of CNN's advantages is that it can take raw images as input and incorporate feature learning into the training process. With a deep structure, the CNN can effectively learn complicated mappings while requiring minimal domain knowledge.

To the best of our knowledge, CNN has not been applied to general-purpose NR-IQA. The primary reason is that the original CNN is not designed for capturing image quality features. In the object recognition domain good features generally encode local invariant parts, however, for the NR-IQA task, good features should be able to capture

The partial support of this research by DARPA through BBN/DARPA Award HR0011-08-C-0004 under subcontract 9500009235, the US Government through NSF Awards IIS-0812111 and IIS-1262122 is gratefully acknowledged.

NSS properties. The difference between NR-IQA and object recognition makes the application of CNN nonintuitive. One of our contributions is that we modified the network structure, such that it can learn image quality features more effectively and estimate the image quality more accurately.

Another contribution of our paper is that we propose a novel framework that allows learning and prediction of image quality on local regions. Previous approaches typically accumulate features over the entire image to obtain statistics for estimating overall quality, and have rarely shown the ability to estimate local quality, except for a simple example in [18]. By contrast, our method can estimate quality on small patchs (such as 32×32). Local quality estimation is important for the image denoising or reconstruction problems, applying enhancement only where required.

We show experimentally that the proposed method advances the state of the art. On the LIVE dataset our CNN outperforms CORNIA and BRISQUE, and achieves comparable results with state of the art FR measures such as FSIM [22]. In addition to the superior overall performance, we also show qualitative results that demonstrate the local quality estimation of our method.

2. Related Work

Previously researchers have attempted to use neural networks for NR-IQA. Li et al. [8] applied a general regression neural network that takes as input perceptual features including phase congruency, entropy and the image gradients. Chetouani et al. [3] used a neural network to combine multiple distortion-specific NR-IQA measures. These methods require pre-extracted handcrafted features and only use neural networks for learning the regression function. Thus they do not have the advantage of learning features and regression models in a holistic way, and these approaches are inferior to the state of the art approaches. In contrast, our method does not require any handcrafted features and directly learns discriminant features from normalized raw image pixels to achieve much better performance.

The use of convolutional neural networks is partly motivated by the feature learning framework introduced in CORNIA [20, 21]. First, the CORNIA features are learned directly from the normalized raw image patches. This implies that it is possible to extract discriminative features from spatial domain without complicated image transformations. Second, supervised CORNIA [21] employs a two-layer structure which learns the filters and weights in the regression model simultaneously based on an EM like approach. This structure can be viewed as an empirical implementation of a two layer neural network. However, it has not utilized the full power of neural networks.

Our approach integrates feature learning and regression into the general CNN framework. The advantages are two fold. First, making the network deeper will raise the learning capacity significantly [1]. In the following sections we will see that with fewer filters/features than CORNIA, we are able to achieve the state of the art results. Second, in the CNN framework, training the network as a whole using a simple method like backpropagation enables the possibility of conveniently incorporating recent techniques designed to improve learning such as dropout [5] and rectified linear unit [7]. Furthermore, after we make the bridge between NR-IQA and CNN, the rapid developing deep learning community will be a significant source of novel techniques for advancing the NR-IQA performance.

3. CNN for NR-IQA

The proposed framework of using CNN for image quality estimation is as follows. Given a gray scale image, we first perform a contrast normalization, then sample non-overlapping patches from it. We use a CNN to estimate the quality score for each patch and average the patch scores to obtain a quality estimation for the image.

3.1. Network Architecture

The proposed network consists of five layers. Figure 1 shows the architecture of our network, which is a $32 \times 32 - 26 \times 26 \times 50 - 2 \times 50 - 800 - 800 - 1$ structure. The input is locally normalized 32×32 image patches. The first layer is a convolutional layer which filters the input with 50 kernels each of size 7×7 with a stride of 1 pixel. The convolutional layer produces 50 feature maps each of size 26×26, followed by a pooling operation that reduces each feature map to one max and one min. Two fully connected layers of 800 nodes each come after the pooling. The last layer is a simple linear regression with a one dimensional output that gives the score.

3.2. Local Normalization

Previous NR-IQA methods, such as BRISQUE and CORNIA, typically apply a contrast normalization. In this work, we employ a simple local contrast normalization method similar to [9]. Suppose the intensity value of a pixel at location (i, j) is $I(i, j)$, we compute its normalized value $\hat{I}(i, j)$ as follows:

$$\begin{aligned} \hat{I}(i,j) &= \frac{I(i,j) - \mu(i,j)}{\sigma(i,j) + C} \\ \mu(i,j) &= \sum_{p=-P}^{p=P} \sum_{q=-Q}^{q=Q} I(i+p, j+q) \\ \sigma(i,j) &= \sqrt{\sum_{p=-P}^{p=P} \sum_{q=-Q}^{q=Q} \big(I(i+p, j+q) - \mu(i,j)\big)^2} \end{aligned} \tag{1}$$

where C is a positive constant that prevents dividing by zero. P and Q are the normalization window sizes. In [9],

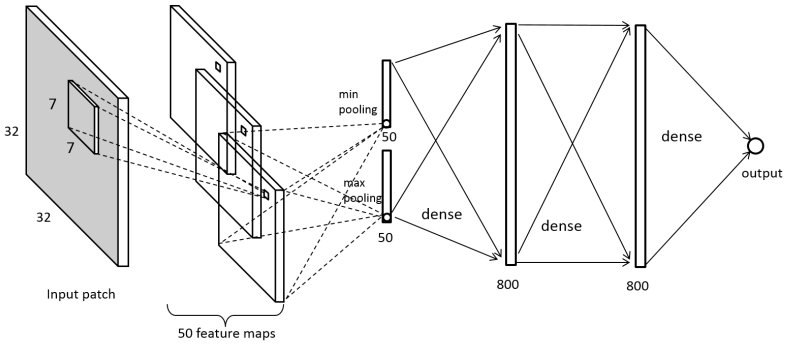


Figure 1: The architecture of our CNN

it was shown that a smaller normalization window size improves the performance. In practice we pick $P = Q = 3$ so the window size is much smaller than the input image patch. Note that with this local normalization each pixel may have a different local mean and variance.

Local normalization is important. We observe that using larger normalization windows leads to worse performance. Specifically, a uniform normalization, which applies the mean and variance of the entire image patch to each pixel, will cause about a 3% drop on the performance.

It is worth noting that when using a CNN for object recognition, a global contrast normalization is usually applied to the entire image. The normalization not only alleviates the saturation problem common in early work that used sigmoid neurons, but also makes the network robust to illumination and contrast variation. For the NR-IQA problem, contrast normalization should be applied locally. Additionally, although luminance and contrast change can be considered distortions in some applications, we mainly focus on distortions arising from image degradations, such as blur, compression and additive noise.

3.3. Pooling

In the convolution layer, the locally normalized image patches are convolved with 50 filters and each filter generates a feature map. We then apply pooling on each feature map to reduce the filter responses to a lower dimension. Specifically, each feature map is pooled into *one* max value and *one* min value, which is similar to CORNIA. Let $R^k_{i,j}$ denote the response at location (i, j) of the feature map obtained by the k-th filter, then the max and min values of u_k and v_k are given by

$$\begin{aligned} u_k &= \max_{i,j} R^k_{i,j} \\ v_k &= \min_{i,j} R^k_{i,j} \end{aligned} \tag{2}$$

where $k = 1, 2, ..., K$ and K is the number of kernels. The pooling procedure reduces each feature map to a 2 dimensional feature vector. Therefore, each node of the next fully connected layer takes an input of size $2 \times K$. It is worth noting that although max pooling already works well, introducing min pooling boosts the performance by about 2%.

In object recognition scenario, pooling is typically performed on every 2×2 cell. In that case, selecting a representative filter response from each small cell may keep some location information while achieving robustness to translation. This operation is particularly helpful for object recognition since objects can typically be modeled as multiple parts organized in a certain spatial order. However, for the NR-IQA task we observe that image distortions are often locally (if not globally) homogeneous, i.e. the same level of distortion occurs at all the locations of a 32×32 patch, for example. The lack of obvious global spatial structure in image distortions enables pooling without keeping locations to reduce the cost of computation.

3.4. ReLU Nonlinearity

Instead of traditional sigmoid or tanh neurons, we use Rectified Linear Units (ReLUs) [11] in the two fully connected layers. [7] demonstrated in a deep CNN that ReLUs enable the network to train several times faster compared to using tanh units. Here we give a brief description of ReLUs. ReLUs take a simple form of nonlinearity by applying a thresholding function to the input, in place of the sigmoid or tanh transform. Let g, w_i and a_i denote the output of the ReLU, the weights of the ReLU and the output of the previous layer, respectively, then the ReLU can be mathematically described as $g = \max(0, \sum_i w_i a_i)$.

Note that ReLUs only allow nonnegative signals to pass through. Due to this property, we do not use ReLUs but use linear neurons (identity transform) on the convolutional and pooling layer. The reason is that the min pooling typically produce negative values and we do not want to block the information in these negative pooling outputs.

3.5. Learning

We train our network on non-overlapping 32×32 patches taken from large images. For training we assign each patch

a quality score as its source image's ground truth score. We can do this because the training images in our experiments have homogeneous distortions. During the test stage, we average the predicted patch scores for each image to obtain the image level quality score. By taking small patches as input, we have a much larger number of training samples compared to using the whole image on a given dataset, which particularly meets the needs of CNNs.

Let x_n and y_n denote the input patch and its ground truth score respectively and $f(x_n; w)$ be the predicted score of x_n with network weights w. Support Vector Regression (SVR) with ϵ-insensitive loss has been successfully applied to learn the regression function for NR-IQA in previous work [21, 9]. We adopt a similar objective function as follows:

$$\begin{aligned} L &= \frac{1}{N}\sum_{n=1}^{N} \|f(x_n; w) - y_n\|_{l_1} \\ w' &= \min_{w} L \end{aligned} \tag{3}$$

Note that the above loss function is equivalent to the loss function used in ϵ-SVR with $\epsilon = 0$. Stochastic gradient decent (SGD) and backpropagation are used to solve this problem. A validation set is used to select parameters of the trained model and prevent overfitting. In experiments we perform SGD for 40 epochs in training and keep the model parameters that generate the highest Linear Correlation Coefficient (LCC) on the validation set.

Recently successful neural network methods [7, 5] report that dropout and momentum improve learning. In our experiment we also find these two techniques boost the performance.

Dropout is a technique that prevents overfitting in training neural networks. Typically the outputs of neurons are set to zero with a probability of 0.5 in the training stage and divided by 2 in the test stage. By randomly masking out the neurons, dropout is an efficient approximation of training many different networks with shared weights. In our experiments, since applying dropout to all layers significantly increases the time to reach convergence, we only apply dropout at the second fully connected layer.

Updating the network weights with momentum is a widely adopted strategy. We update the weights in the following form:

$$\begin{aligned} \Delta w^t &= r^t \Delta w^{t-1} - (1 - r^t)\epsilon^t \langle \nabla_w L \rangle \\ w^t &= w^{t-1} + \Delta w^t \\ \epsilon^t &= \epsilon_0 (d)^t \\ r^t &= \begin{cases} \frac{t}{T} r_e + (1 - \frac{t}{T}) r_s, & t < T \\ r_e, & t \geqslant T \end{cases} \end{aligned} \tag{4}$$

where w^t is weight at epoch t, $\epsilon_0 = 0.1$ is learning rate, $d = 0.9$ is decay for the learning rate, $r_s = 0.9$ and $r_e = 0.5$ are starting and ending momentums respectively, $T = 10$ is a threshold to control how the momentum changes with the number of epochs. Note that unlike [5] where momentum starts off at a value of 0.5 and stays at 0.99, we use a large momentum at the beginning and reduce it as the training progresses. We found through experiments that this setting can achieve better performance.

4. Experiment

4.1. Experimental Protocol

Datasets: The following two datasets are used in our experiments.

(1) LIVE [15]: A total of 779 distorted images with five different distortions – JP2k compression (JP2K), JPEG compression (JPEG), White Gaussian (WN), Gaussian blur (BLUR) and Fast Fading (FF) at 7-8 degradation levels derived from 29 reference images. Differential Mean Opinion Scores (DMOS) are provided for each image, roughly in the range $[0, 100]$. Higher DMOS indicates lower quality.

(2) TID2008 [12]: 1700 distorted images with 17 different distortions derived from 25 reference images at 4 degradation levels. In our experiments, we consider only the four common distortions that are shared by the LIVE dataset, i.e. JP2k, JPEG, WN and BLUR. Each image is associated with a Mean Opinion Score (MOS) in the range $[0, 9]$. Contrary to DMOS, higher MOS indicates higher quality.

Evaluation: Two measures are used to evaluate the performance of IQA algorithms: 1) Linear Correlation Coefficient (LCC) and 2) Spearman Rank Order Correlation Coefficient (SROCC). LCC measures the linear dependence between two quantities and SROCC measures how well one quantity can be described as a monotonic function of another quantity. We report results obtained from 100 train-test iterations where in each iteration we randomly select 60% of reference images and their distorted versions as the training set, 20% as the validation set, and the remaining 20% as the test set.

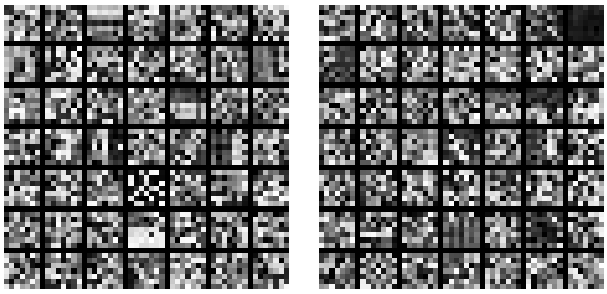
4.2. Evaluation on LIVE

On the LIVE dataset, for distortion-specific experiment we train and test on each of the five distortions: JP2K, JPEG, WN, BLUR and FF. For non-distortion-specific experiments, images of all five distortions are trained and tested together without providing a distortion type.

Table 1 shows the results of the two experiments compared with previous state of the art NR-IQA methods as well as FR-IQA methods. Results of the best performing NR-IQA systems are in bold. The FR-IQA measures are evaluated by using 80% of the data for fitting a non-linear logistic function, then testing on 20% of the data. We can see from Table 1 that our approach works well on each of

SROCC	JP2K	JPEG	WN	BLUR	FF	ALL
PSNR	0.870	0.885	0.942	0.763	0.874	0.866
SSIM	0.939	0.946	0.964	0.907	0.941	0.913
FSIM	0.970	0.981	0.967	0.972	0.949	0.964
DIIVINE	0.913	0.910	**0.984**	0.921	0.863	0.916
BLIINDS-II	0.929	0.942	0.969	0.923	0.889	0.931
BRISQUE	0.914	0.965	0.979	0.951	0.877	0.940
CORNIA	0.943	0.955	0.976	**0.969**	0.906	0.942
CNN	**0.952**	**0.977**	0.978	0.962	**0.908**	**0.956**
LCC	JP2K	JPEG	WN	BLUR	FF	ALL
PSNR	0.873	0.876	0.926	0.779	0.870	0.856
SSIM	0.921	0.955	0.982	0.893	0.939	0.906
FSIM	0.910	0.985	0.976	0.978	0.912	0.960
DIIVINE	0.922	0.921	**0.988**	0.923	0.888	0.917
BLIINDS-II	0.935	0.968	0.980	0.938	0.896	0.930
BRISQUE	0.922	0.973	0.985	0.951	0.903	0.942
CORNIA	0.951	0.965	0.987	**0.968**	0.917	0.935
CNN	**0.953**	**0.981**	0.984	0.953	**0.933**	**0.953**

Table 1: SROCC and LCC on LIVE. Italicized are FR-IQA methods for reference.



(a) (b)

Figure 2: Learned convolution kernels on (a) JPEG (b) ALL on LIVE dataset

the five distortions, especially on JPEG, JP2K and FF. For the overall evaluation, our CNN outperformed all previous state of the art NR-IQA methods and approached the state of the art FR-IQA method FSIM.

We visually examine the learned convolution kernels, and find only a few kernels present obvious structures related to the type of distortion. Figure 2 shows the kernels learned on JPEG and all distortions combined respectively. We can see that blockiness patterns are learned from JPEG, and a few blur-like patterns exist for kernels learned from all distortions. It is not surprising that the kernels learned by CNN tend to be noisy patterns instead of presenting strong structure related to certain distortions as shown in CORNIA[20]. This is because CORNIA's feature learning is unsupervised and belongs to generative model while our CNN is supervisedly trained and learns discriminative features.

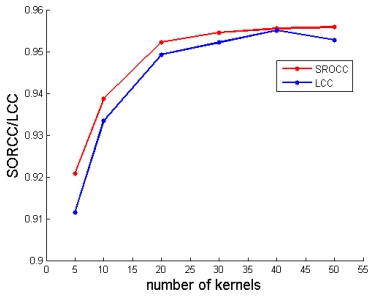


Figure 3: SROCC and LCC with respect to number of convolution kernels

size	5×5	7×7	9×9
SROCC	0.953	0.956	0.955
LCC	0.951	0.953	0.955

Table 2: SROCC and LCC under different kernel sizes

4.3. Effects of Parameters

Several parameters are involved in the CNN design. In this section, we examine how these parameters affect the performance of the network on the LIVE dataset.

Number of kernels Figure 3 shows how the performance varies with the number of convolution kernels. It is not surprising to find that the number of filters significantly affects the performance. In general, the use of more kernels leads to better performance. But little performance increase is gained when the number of kernels exceeds 40.

Kernel size We train and test the network with different kernel sizes while fixing the rest of structure. Table 2 shows how the performance changes with the kernel size. We can see from Figure 2 that all tested kernel sizes show similar performance. The proposed network is not sensitive to kernel size.

Patch size Since in our experiment the whole image score is simply the average score of all patches sampled, we examine how the patch sampling strategy affects performance. This includes two aspects, patch size and number of patches per image. It is worth noting that if we keep sampling patches in a non-overlapping way, larger patch size leads to fewer patches. For example, if we double the patch size, the number of patches per image will drop to one fourth of the original number. To avoid this situation, we allow overlap sampling and use a fixed sampling stride (32) for different patch sizes. In this way the number of patches per image remains roughly the same (ignoring the border effect) when patch size varies. Table 3 shows the change of performance with respect to patch size. From Table 3 we see that larger patch results in better performance. The performance increases slightly as the patch size increases

size	48	40	32	24	16
SROCC	0.959	0.958	0.956	0.950	0.946
LCC	0.957	0.955	0.953	0.947	0.946

Table 3: SROCC and LCC on different patch size

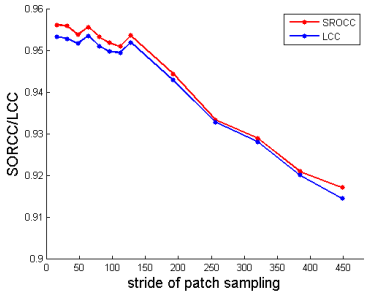


Figure 4: SROCC and LCC with respect to the sampling stride

from 8 to 48. However larger patches not only lead to more processing time but also reduce spatial quality resolution. Therefore we prefer the smallest patch that yields the state of the art performance.

Sampling stride To observe how the number of patches affects the overall performance, we fix the patch size and vary the stride. Changing the stride does not change the structure of the network. For simplicity at each iteration of the 100 iteration experiment, we use the same model trained at stride 32, and test with different different stride values. Figure 4 shows the change of performance with respect to the stride. A larger stride generally leads to lower performance since less image information is used for overall estimation. However, it is worth noting that state of the art performance is still maintained even when the stride increases up to 128, which roughly corresponds to $1/16$ of the original number of patches. This result is consistent with the fact that the distortions on the LIVE data are roughly homogeneous across entire image, and also indicates that our CNN can accurately predict quality score on small image patches.

4.4. Cross Dataset Test

Tests on TID2008 This set of experiment is designed to test the generalization ability of our method. We follow the protocol of previous work [9, 20] to investigate cross dataset performance between the two datasets by training our CNN on LIVE and testing on TID2008[1]. Only the four types of distortions that are shared by LIVE and TID2008 are examined in this experiment. The DMOS scores in LIVE range from 0 to 100, while the MOS scores in TID2008 fall in the range 0 and 9. To make a fair comparison, we adopt the same method as [20] to perform a nonlinear mapping on the predicted scores produced by the model trained on LIVE. A nonlinear mapping based on a logistic function is usually applied to FR measures for transforming the quality measure into a certain range. We randomly split the TID2008 into two parts of 80% and 20% 100 times. Each time 80% of data is used for estimating parameters of the logistic function and 20% is used for testing, i.e. evaluating the transformed prediction scores. Results of the cross dataset test are shown in Table 4. We can see that our CNN outperforms previous state of the art methods.

[1]We have observed that some images in TID2008 share the same content as images in LIVE. However, their resolutions are different.

	CORNIA	BRISQUE	CNN
SROCC	0.890	0.882	**0.920**
LCC	0.880	0.892	**0.903**

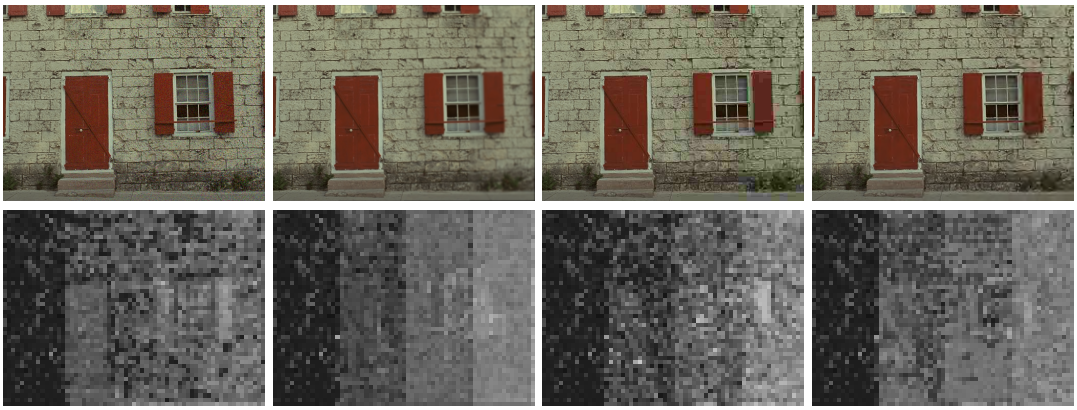
Table 4: SROCC and LCC obtained by training on LIVE and testing on TID2008

4.5. Local Quality Estimation

Our CNN measures the quality on small image patches, so it can be used to detect low/high quality local regions as well as giving a global score for the entire image.

We select an undistorted reference image from TID 2008 (which is not included in LIVE) and divide it into four vertical parts. We then replace the second to the fourth parts with distorted versions at three different degradation levels. Four synthetic images are generated in this way, one for each types of distortions including WN, BLUR, JPEG and JP2K. We then perform local quality estimation on these synthetic images using our model trained on LIVE. We scan 16×16 patches with a stride of 8 and normalize the predicted scores into the range $[0, 255]$ for visualization. Figure 5 shows estimated quality map on the synthetic images. We can see that our model properly distinguishes the clean and the distorted parts of each synthetic image.

To better examine the local quality estimation power of our model, we consider several types of distortions in TID2008 that are not used in previous experiments, and find three types that can only affect local regions: JPEG transmission, JPEG2000 transmission and blockwise distortion. Again from TID2008 we pick several images that are not shared by LIVE, and test on their distorted versions with the above three distortions. Figure 6 shows the local quality estimation results. We find our model locates the distorted regions with reasonable accuracy and the results generally fit human judgement. It is worth noting that our model locates the blockwise distortion very well although this type of distortion is not contained in the training data from LIVE. In the images of the third row in Figure 6, the stripes on the window are mistaken as a low quality region. We speculate that it is because the local patterns on the stripes resem-



(a) (b) (c) (d)

Figure 5: Synthetic examples and local quality estimation results. The first row contains images distorted in (a) WN, (b) BLUR, (c) JPEG (d) JP2K. Each image is divided into four parts and three of them are distorted in different degradation level. The second row shows the local quality estimation results, where brighter pixels indicate lower quality.

Figure 6: Local quality estimation results on examples of non-global distortion from TID2008. Column 1,3,5 show (a) jpeg transmission errors (b) jpeg2000 transmission errors (c) local blockwise distortion. Column 2,4,6 show the local quality estimation results, where brighter pixels indicate lower quality.

ble blockness distortion. Contextual information may be needed to overcome such problems.

4.6. Computational Cost

Our CNN is implemented using the Python library Theano [2]. With Theano we are able to easily run our algorithm on a GPU to speed up the process without much optimization. Our experiments are performed on a PC with 1.8GHz CPU and GTX660 GPU. We measure the processing time on images of size 512×768 using our model of 50 kernels with 32×32 input size, and test the model using part of those strides that give the state of the art performance in

the experiments on LIVE. Table 5 shows the average processing time per image under different strides. Note that our implementation is not fully optimized. For example, the normalization process for each image is performed on the CPU in about 0.017 sec, which takes a significant portion of the total time. From Table 5 we can see that with a sparser sampling pattern (stride greater than 64), real time processing can be achieved while maintaining state of the art performance.

stride	32	64	96	128
time(sec)	0.114	0.041	0.029	0.023

Table 5: Time cost under different strides.

5. Conclusion

We have developed a CNN for no-reference image quality assessment. Our algorithm combines feature learning and regression as a complete optimization process, which enables us to employ modern training techniques to boost performance. Our algorithm generates image quality predictions well correlated with human perception, and achieves state of the art performance on standard IQA datasets. Furthermore we demonstrated that our algorithm can estimate quality in local regions, which is rarely reported in previous literature and has many potential applications in image reconstruction or enhancement.

References

[1] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(8):1798–1828, 2013.

[2] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, June 2010.

[3] A. Chetouani, A. Beghdadi, S. Chen, and G. Mostafaoui. A novel free reference image quality metric using neural network approach. In *Int. Workshop Video Process. Qual. Metrics Cons. Electron.*, pages 1–4, Jan. 2010.

[4] D. C. Ciresan, U. Meier, and J. Schmidhuber. Multi-column deep neural networks for image classification. In *Computer Vision and Pattern Recognition*, pages 3642–3649, 2012.

[5] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. arxiv:1207.0580, 2012.

[6] K. Kavukcuoglu, P. Sermanet, Y.-L. Boureau, K. Gregor, M. Mathieu, and Y. LeCun. Learning convolutional feature hierachies for visual recognition. In *Advances in Neural Information Processing Systems (NIPS)*, 2010.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, volume 1, page 4, 2012.

[8] C. Li, A. Bovik, and X. Wu. Blind image quality assessment using a general regression neural network. *IEEE Transactions on Neural Networks*, 22(5):793–799, 2011.

[9] A. Mittal, A. Moorthy, and A. Bovik. No-reference image quality assessment in the spatial domain. *IEEE Transactions on Image Processing*, 21(12):4695–4708, 2012.

[10] A. K. Moorthy and A. C. Bovik. Blind image quality assessment: From natural scene statistics to perceptual quality. *IEEE Transactions on Image Processing*, 20(12):3350–3364, Dec. 2011.

[11] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, pages 807–814, 2010.

[12] N. Ponomarenko, V. Lukin, A. Zelensky, K. Egiazarian, M. Carli, and F. Battisti. TID2008 - a database for evaluation of full-reference visual quality assessment metrics. *Advances of Modern Radio Electronics*, 10:30–45, 2009.

[13] M. Saad, A. Bovik, and C. Charrier. Blind image quality assessment: A natural scene statistics approach in the DCT domain. *IEEE Transactions on Image Processing*, 21(8):3339–3352, Aug. 2012.

[14] H. R. Sheikh, A. C. Bovik, and G. de Veciana. An information fidelity criterion for image quality assessment using natural scene statistics. *IEEE Transactions on Image Processing*, 14(12):2117–2128, Dec. 2005.

[15] H. R. Sheikh, Z. Wang, L. Cormack, and A. C. Bovik. LIVE image quality assessment database release 2. Online, http://live.ece.utexas.edu/research/quality.

[16] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[17] Z. Wang and Q. Li. Information content weighting for perceptual image quality assessment. *IEEE Transactions on Image Processing*, 20(5):1185–1198, 2011.

[18] W. Xue, L. Zhang, and X. Mou. Learning without human scores for blind image quality assessment. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 995–1002, 2013.

[19] W. Xue, L. Zhang, X. Mou, and A. Bovik. Gradient magnitude similarity deviation: A highly efficient perceptual image quality index. *Image Processing, IEEE Transactions on*, 23(2):684–695, Feb 2014.

[20] P. Ye, J. Kumar, L. Kang, and D. Doermann. Unsupervised feature learning framework for no-reference image quality assessment. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1098–1105, 2012.

[21] P. Ye, J. Kumar, L. Kang, and D. Doermann. Real-time no-reference image quality assessment based on filter learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 987–994, 2013.

[22] L. Zhang, D. Zhang, X. Mou, and D. Zhang. FSIM: A feature similarity index for image quality assessment. *IEEE Transactions on Image Processing*, 20(8):2378–2386, 2011.