GENERATING ADAPTIVE AND ROBUST FILTER SETS USING AN UNSUPERVISED LEARNING FRAMEWORK

Mohit Prabhushankar, Dogancan Temel, and Ghassan AlRegib

Center for Signal and Information Processing, School of Electrical and Computer Engineering,
Georgia Institute of Technology, Atlanta, GA

ABSTRACT

In this paper, we introduce an adaptive unsupervised learning framework, which utilizes natural images to train filter sets. The applicability of these filter sets is demonstrated by evaluating their performance in two contrasting applications - image quality assessment and texture retrieval. While assessing image quality, the filters need to capture perceptual differences based on dissimilarities between a reference image and its distorted version. In texture retrieval, the filters need to assess similarity between texture images to retrieve closest matching textures. Based on experiments, we show that the filter responses span a set in which a monotonicity-based metric can measure both the perceptual dissimilarity of natural images and the similarity of texture images. In addition, we corrupt the images in the test set and demonstrate that the proposed method leads to robust and reliable retrieval performance compared to existing methods.

***Index Terms*—** Unsupervised Learning, ZCA Whitening, Adaptive Filter Sets, Image Quality Assessment, Texture Retrieval

1. INTRODUCTION

Unsupervised learning is a branch of machine learning tasked with inferring functions that describe hidden structures from unlabeled data. In terms of neural networks, these functions are a set of affine transformations subjected to nonlinearity. Each function is described as a layer, and stacking layers leads to deep neural nets. These deep networks can be used as end to end systems utilizing all the underlying layers as one entity. An example of vanilla neural networks are autoencoders. The authors in [1] describe autoencoders as neural networks that are trained to copy inputs to outputs. By placing multiple constraints, the network learns to extract hidden characteristics from data. Autoencoders can extract overcomplete basis functions which can be used for denoising. Stacking layers of denoising autoencoders, which are trained layer-wise to locally denoise corrupted versions of their inputs, led to developement of Stacked Denoising Autoencoders [2].

In this article, we show that the individual layers can be repurposed to tasks different from what they were trained for. In other words, we use neural networks as tools to generate affine functions. Mathematically, these functions act as filters that process inputs to span a response set in which we can perform various image processing tasks. To test the applicability of generated filters, we focus on image quality assessment and texture retrieval. These applications are a challenge for the traditional deep learning networks due to lack of large labeled datasets that are essential for training. To overcome label requirements and application specific data, we learn filters from natural images in an unsupervised fashion. In practice, we need to train filters that are insensitive to lower order statistics of natural images, so that the trained model can work on disparately distributed images during testing. We achieve this by proposing a simple extension to ZCA whitening procedure during preprocessing in Section 2.1, and describe the learning architecture that generates the filter sets in Section 2.2. We integrate the preprocessing and the learning blocks to obtain the Unsupervised Learning Framework which has two tuning parameters. These parameters can be adjusted to generate multiple filter sets capturing multi-order statistics as discussed in Section 3. We utilize the generated filters to perform image quality assessment in Section 4 and texture retrieval in Section 5. Finally, we conclude our work in Section 6.

2. UNSUPERVISED LEARNING FRAMEWORK

In this section, we describe the Unsupervised Learning Framework ($ULF_{k,h}$), whose block diagram is shown in Fig. 1. The framework takes in an input I_D and generates weights W and bias b, which span our filter sets. The subscripts k and h correspond to tuning parameters that determine filter characteristics.

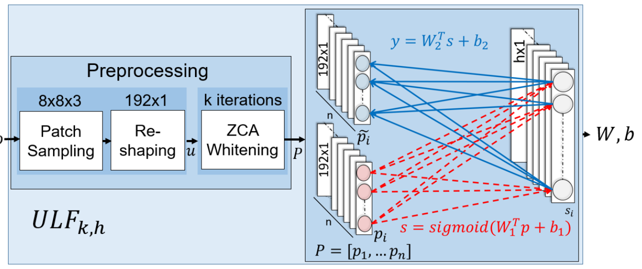


Fig. 1: Unsupervised Learning Framework (ULF).

2.1. Preprocessing

As shown in Fig. 1, the preprocessing stage converts raw images from input database I_D into patch-based vectors u, and feeds them into the Zero-Phase Component Analysis (ZCA) whitening algorithm [3]. ZCA whitening is a decorrelating procedure that removes the first and the second order statistics of input data and forces the ensuing learning stage to capture higher order statistics. Essentially, this specifies the covariance matrix of output P to be the identity matrix I. Therefore, the sample covariance matrix of P is given by

$$PP^T = (n-1)I, \tag{1}$$

where n is the number of input vectors u. For a large value of n, the sample covariance estimate used in Eq. 1, which is the maximum likelihood estimate (MLE), would be close to the actual covariance matrix. ZCA whitening attempts to find a set of linear, symmetric filters W_z that transform u to P. The output of the preprocessing block is given by

$$P = W_z u, \tag{2}$$

where u is assumed to be zero centered. Following the approach described by the authors in [3] and [4], we calculate W_z as

$$W_z \propto (uu^T)^{-1/2}. \tag{3}$$

The outer product uu^T is symmetric and orthogonally diagonalizable. Hence, applying eigenvalue decomposition to uu^T, we obtain

$$uu^T = V\Lambda V^T, \tag{4}$$

where V are the eigenvectors and Λ are the eigenvalues of uu^T. Raising both sides of Eq. 4 to the power $-1/2$, we get

$$(uu^T)^{-1/2} = V\Lambda^{-1/2}V^T. \tag{5}$$

Combining Eq. 5 and Eq. 3, we obtain an expression for W_z as

$$W_z \propto V\Lambda^{-1/2}V^T. \tag{6}$$

The matrix uu^T is, however, ill conditioned. Therefore, a regularization term ϵ is added to the eigenvalues, so that the inverse of the smaller eigenvalues do not arbitrarily scale the resultant. Hence, the expression for the filters of a standard ZCA whitening algorithm is given by

$$W_z \propto V(\Lambda + \epsilon)^{-1/2}V^T. \tag{7}$$

Applying W_z on u produces whitened patches whose covariance matrix is ideally I. In practice, this is not always the case. Fig. 2(a) shows uu^T while Fig. 2(b) shows PP^T using Eq. 2 and Eq. 3. The nine distinct blocks are present in Fig. 2(a) because of the three separate channels (RGB) whose distributions are different but not entirely uncorrelated. These blocks are still present in Fig. 2(b) because of the approximation caused by the addition of ϵ. Such an approximation is tolerable as long as the testing data distribution is similar to the training data distribution. Since we propose utilizing texture images for testing, which do not adhere to the natural scene statistics, an approximate identity matrix for PP^T is not acceptable. Hence, we propose a simple extension to the standard ZCA algorithm in which we apply Eq. 2 iteratively on the initial input (u_0), k times. k is specified as an input to the algorithm. Using this extension, the intermediate whitened vector denoted by u_i, at every i^{th} iteration is given by

$$\begin{gathered} u_i = W_z^i u_{i-1}, \forall i \in [1:k], \forall k > 0 \\ W_z^i \propto (u_{i-1}u_{i-1}^T)^{-1/2}, \end{gathered} \tag{8}$$

where W_z^i is the intermediate whitening matrix at iteration i. As u_i can be completely described with the knowledge of u_{i-1}, the sequence of matrices $u_i, \forall i = [1,k]$ satisfy the markov property. Hence after calculating all the W_z^i matrices using Eq. 8, we express the final whitened patches P as

$$P = W_z^k u_{k-1}, \tag{9}$$

where W_z^k and u_{k-1} are the whitening filters, and input vectors at k^{th} iteration. $k = 1$ gives us the standard ZCA algorithm. Taking k to be $1/\epsilon$, with $\epsilon = 0.1$, we obtain the covariance matrix as shown in Fig. 2(c). Essentially, by performing whitening multiple times, the effect of ϵ is reduced. This ensures that Eq. 1 holds and the lower order statistics from natural images is eliminated.

In common practice, the whitening filters (W_z) calculated during training are used to transform the test data as well, because there is not enough test data (n) to estimate sample covariance matrix in Eq. 1. However, W_z is trained on natural images and is not suitable to transform texture or distorted images. Hence, we calculate W_z during testing as well. Even though n during testing is insufficient to estimate covariance matrix precisely, it is not singular and for the present we overlook this approximation.

2.2. Linear Decoder

Linear decoder is a variation of standard autoencoder in which the nonlinearity of the final reconstructed output decoder is replaced with a linear activation function. The primary task of this network is to learn representative structure in its input data while attempting to reconstruct it. Adding a sparsity criterion forces the network to learn unique statistical features from the data rather than an identity function in the hidden layer [1]. The hidden layer responses (s), with a sigmoid nonlinearity activation, are obtained as

$$\begin{gathered} s = sigmoid(W_1^T P + b_1), \\ \forall s \in \Re^{h\times n}, W_1 \in \Re^{d\times h}, P \in \Re^{d\times n}, b_1 \in \Re^h, \end{gathered} \tag{10}$$

where W_1 and b_1 are the forward weights and bias, and P is the input patch matrix. Eq. 10 is an affine function followed by a non linearity. The rows of weights W_1^T represent a set of h filters of dimension d that linearly transform P. If h is greater than d, we obtain an overcomplete basis which, coupled with sparsity, learns localized features [1]. If h is lower than d, then we have undercomplete basis set in which the network is forced to learn the most salient features. In this work, we use h as a tuning parameter of the learning framework. The responses s are used to reconstruct the patches $\tilde{P}$, using a set of backward weights W_2, and bias, b_2, as

$$\tilde{P} = W_2^T s + b_2. \tag{11}$$

Backpropagation is used to train weights (W) and bias (b) by setting up the objective function $J(W, b)$ as

$$J(W,b) = \|\tilde{P} - P\|_2^2 + \beta \sum_{j=1}^{h} \mathrm{KL}(\rho||\hat{\rho}_j) + \lambda\|W\|_2^2, \tag{12}$$

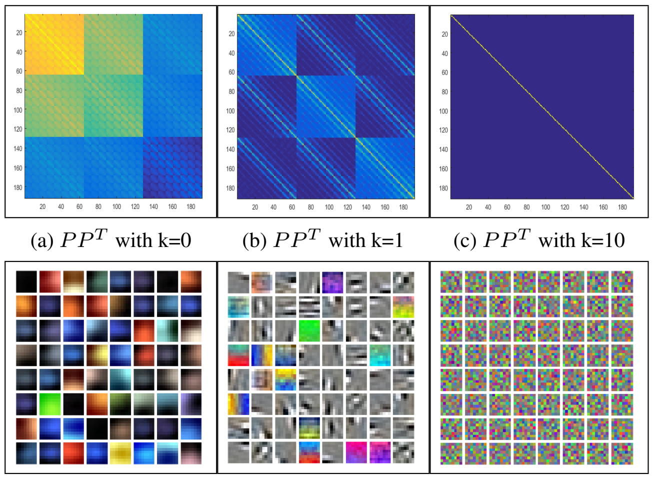
where the first term is the reconstructed ℓ_2-norm error, the second term is the sparsity constraint, and the third term is the weight decay term used for regularization [5]. The sparsity constraint is on the hidden layer responses s. KL-Divergence, which is differentiable at the origin, is used as a proxy for ℓ_1 sparsity. Differentiability ensures that Eq. (12) is viable for L-BFGS minimization. ρ is the desired average response, which is set to a small value of 0.035 [6]. $\hat{\rho}_j$ is the actual average response of j^{th} hidden unit averaged over all n inputs, and is calculated as,

$$\hat{\rho}_j = \frac{1}{n}\sum_{i=1}^{n} s_i^{(j)}. \tag{13}$$

Minimizing the KL-Divergence between ρ and $\hat{\rho}_j$ ensures that the majority of the h responses are close to 0, hence promoting sparsity. A detailed description is provided in [6]. The values for β and λ are set to 5 and $3e^{-3}$, respectively, as suggested by the authors in [6].

3. FILTER SET GENERATION

We use the proposed framework to generate filter sets that are trained with natural images from ImageNet database [7]. 100 random patches of dimensions $8 \times 8 \times 3$ are sampled from each of 1000 random images to form I_D. These patches are preprocessed with different values of k to train W_1. The weights are visualized in Fig. 2(d)-(f) for $h = 64$. The number of neurons (h), in the hidden layer, represents the number of filters for that set. The covariance matrices of patches P that generate these filters are visualized in Fig. 2(a)-(c). Without preprocessing, filters primarily learn the first order statistics. Edges are predominant when $k = 1$, which are the independent components of natural images [3]. When $k = 10$, natural scenes lose pixel-wise correlations and hence there are no discernible simple structures including edges.



(a) PP^T with k=0 (b) PP^T with k=1 (c) PP^T with k=10

(d) Filter sets for k = 0 (e) Filter sets for k = 1 (f) Filter sets for k = 10

Fig. 2: Covariance matrix of patches and their generated filter sets.

4. IMAGE QUALITY ASSESSMENT

Perceptual image quality assessment (IQA) is a challenging field whose objective is to analyze an image and estimate its quality as perceived by humans. Database generation in this domain is a challenge, since it requires gathering subjective scores of distorted images. Learning based approaches are commonly employed in the literature to estimate quality [8][9]. However, all these methods are trained with distortion specific images that may not be available or sufficient. To overcome this limitation of insufficient data, we proposed an unsupervised approach to image quality assessment called UNIQUE [10] and it's extension MS-UNIQUE [11], that use only generic natural images during training. In this section we recast both these estimators using the framework described in Section 2.

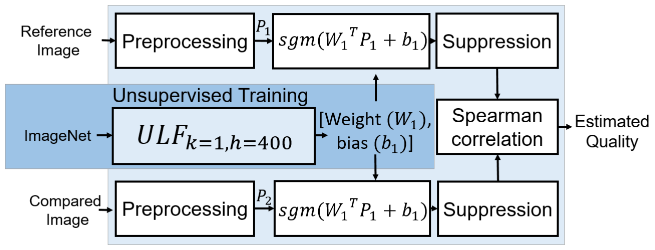


Fig. 3: Unsupervised image quality estimation.

4.1. UNIQUE: Unsupervised Image Quality Estimation

The block diagram in Fig. 3 shows UNIQUE trained with the proposed framework. The proposed training uses a standard ZCA procedure with $k = 1$ and an overcomplete architecture with $h = 400$ [6]. The feature vectors are obtained by filtering non overlapping preprocessed patches of both reference and distorted images. The two vectors are compared using Spearman correlation to estimate quality. A complete description of the algorithm is provided in [10].

4.2. MS-UNIQUE: Multi-Model and Sharpness-Weighted Unsupervised Image Quality Estimation

MS-UNIQUE was designed to have a wide configuration with independent networks modeling the same input using different number

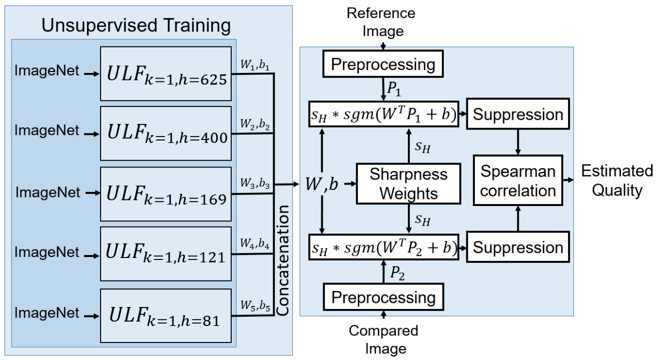


Fig. 4: Multi-model and sharpness-weighted unsupervised image quality estimation.

of neurons. Such a task is directly achieved by varying h in the proposed framework, hence casting MS-UNIQUE as an instance of the framework. We use 5 instances of h spanning both undercomplete and overcomplete states. In MS-UNIQUE, we also classify the filters based on sharpness as either being activated by edge or color content. A detailed description of the algorithm is provided in [11].

4.3. Validation

A detailed validation of UNIQUE and MS-UNIQUE is provided in [10] and [11]. Table 1 lists the results of state of the art estimators on LIVE [12] and TID 2013 [13] databases. The estimated scores from each method is validated against the subjective scores. Validation is performed utilizing metrics that analyze accuracy (RMSE), consistency (Outlier Ratio), linearity (Pearson correlation), and monotonic behavior (Spearman correlation). It is evident that both UNIQUE and MS-UNIQUE are among the top performing estimators.

Table 1: Performance of image quality estimators.

Methods	**PSNR HMA** [14]	**MS SSIM** [15]	**SR SIM** [16]	**FSIMc** [17]	**UNI QUE** [10]	**MS-UNIQUE** [11]
	Outlier Ratio					
TID13	0.670	0.697	**0.632**	0.727	0.640	**0.611**
	Root Mean Square Error					
LIVE	**6.58**	7.43	7.54	7.20	6.76	**6.61**
TID13	0.69	0.68	0.61	0.68	**0.60**	**0.57**
	Pearson Correlation Coefficient					
LIVE	**0.958**	0.946	0.945	0.950	0.956	**0.958**
TID13	0.827	0.832	0.866	0.832	**0.870**	**0.884**
	Spearman Correlation Coefficient					
LIVE	0.944	0.951	**0.955**	**0.959**	0.952	0.949
TID13	0.817	0.785	0.807	0.851	**0.860**	**0.870**

5. TEXTURE RETRIEVAL

The aim of texture retrieval is to detect images that are sampled from the same repeating texture as the given query image. Algorithms designed for recognizing and retrieving textures need to filter the entire contents of an image rather than non-overlapping patches of it. Hence, to perform a global filtering operation, we need to design a pyramidal structure of filter banks, which can be mimicked by stacking multiple hidden layers during training.

5.1. Training Filter Sets

The block diagram in Fig. 5 gives an overview of the filter set generation module. We divide the filtering process into two phases based on the data characteristics we wish to capture - color based filtering and structure based filtering. For color based filtering, we train 400 filters with $k = 0$. Thus, the network is limited to learn only the first order statistics from unwhitened data. The filters are visualized in Fig. 2(d). For capturing structure, we process the entire image by building multiple layers of filter sets. Each layer is a self contained filter set, acting on a specific subregion. Following the illustration provided on the right side of Fig. 5, we train filters to act on $8 \times 8 \times 3$ P_2 subregions. The responses of nine P_2 patches are concatenated to process $24 \times 24 \times 3$ P_3 subregions. The responses of adjoining nine P_3 patches are passed through a max pooling layer to obtain the maximum 64 values from each P_3 subregion. Pooling is required to limit the input dimensions, since it is challenging for an autoencoder to extract features from high dimensional data. Moreover, pooling provides translation invariance. The pooled feature responses are concatenated to obtain a response vector that includes values from every part of an image. A final filter set is trained to transform these responses to obtain the final feature vector.

5.2. Retrieval

The texture images in the database are first resized to get patches P_1 of dimensions $8 \times 8 \times 3$, as shown on the top right of Fig. 5. These low resolution images, essentially represent the mean color. Without any preprocessing, all the images are filtered using the generated color filter sets. The images with relatively similar colors to the query image are selected and examined for structure. Each texture image is resized to dimensions of $72 \times 72 \times 3$. The block diagram in Fig. 5 shows the progression of spatial filtering to obtain a final feature vector that encompasses the entire image. Once all feature vectors are obtained, Spearman correlation is calculated between all features in the database. Based on the similarity coefficients, similar texture images to the query image are retrieved.

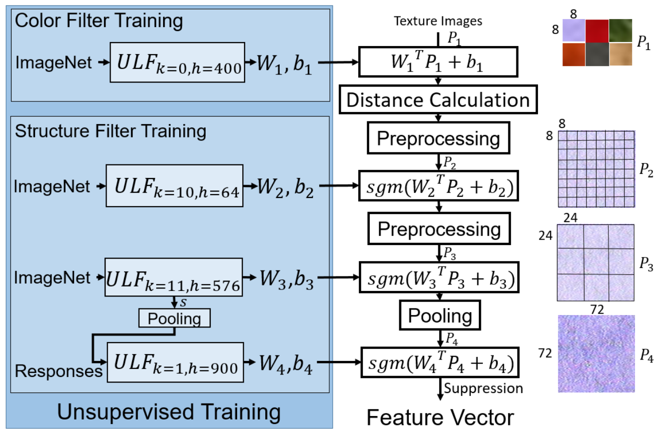


Fig. 5: Texture retrieval through unsupervised learning framework.

5.3. Validation

We test the retrieval process on the CUReT database [18]. Non overlapping patches of size $128 \times 128 \times 3$ were extracted from all images with viewing condition number 55 as detailed by the authors in [19]. There are 61 different texture classes, each of which have 3 samples. To quantify the results of the proposed method, we use three standard retrieval metrics including Precision at one (P@1), Mean Reciprocal Rank (MRR), and Mean Average Precision (MAP). All of these metrics produce results in the range $[0, 1]$ with 1 being the ideal score. The results are presented in Table 2. The proposed method is measured against commonly utilized and state of the art techniques. It can be observed that the proposed method is always among the top two performing metrics, second only to STSIM-1 [20], which is a handcrafted technique designed to measure texture similarity.

Table 2: Performance Validation on CUReT

Method	P@1	MRR	MAP
S-SSIM [15]	0.0546	0.0952	0.0935
CW-SSIM [21]	0.1530	0.2612	0.1925
STSIM-1 [20]	**0.9071**	**0.9447**	**0.9048**
STSIM-2 [20]	0.8852	0.9248	0.8500
STSIM-M [20]	0.8798	0.9170	0.8516
Proposed	**0.9126**	**0.9371**	**0.8922**

5.4. Robustness

We examine the proposed method's robustness against noise. Texture retrieval is performed after adding random noise to CUReT database, with standard deviation σ in the range $[0, 100]$. The MAP results of top metrics are presented in Table 3. The results clearly indicate that the drop off of MAP values for the proposed method is lower than the compared techniques. Robustness of the proposed method is rooted in the expectation that a higher level representation should be stable under corruptions to the input. Using the k extension to the standard ZCA algorithm decorrelates adjoining pixel values in the natural images, hence disrupting local structure in the data - a behavior equivalent to adding noise. The proposed method has been trained on decorrelated data and has learnt to consider only the underlying principal components that constitute an image ignoring the lower order statistics.

Table 3: MAP values for random Gaussian noise with zero mean and standard deviation σ

σ	CW-SSIM	STSIM-1	STSIM-2	STSIM-M	Proposed
5	0.1939	**0.9017**	0.8433	0.8579	0.8931
25	0.1700	0.8576	0.7478	0.8033	**0.8600**
50	0.1185	0.7450	0.5839	0.6990	**0.8535**
75	0.1204	0.6825	0.4460	0.6218	**0.8351**
100	0.0876	0.5465	0.3487	0.4826	**0.8219**

6. CONCLUSION

In this paper, we proposed the Unsupervised Learning Framework (ULF) that is based on a data driven approach, to solve challenges which lack sufficient domain-specific data and annotations. An extension to the classical ZCA algorithm was proposed that would, in practice, orthogonalize the input vectors thereby eliminating all the lower order natural scene statistics. Orthogonalization allows the linear decoder to learn higher order characterizations from the data. We modified the number of neurons in the linear decoder to obtain multiple filter sets modeling the same input and used them in parallel. We demonstrated the use of these filter sets on image quality assessment (IQA) and texture retrieval. In IQA, we showed that the already proposed estimators, UNIQUE and MS-UNIQUE are instances of the framework and that the filter sets spanned a response space wherein perceptual dissimilarity could be measured. We also showed that similarity is quantifiable in the same response space, by retrieving texture images. We established the robustness of the proposed method towards noisy input data. In conclusion, we illustrated utilizing unsupervised learning to generate robust and adaptive filter sets that can be used in various image processing applications.

7. REFERENCES

[1] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016.

[2] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *Journal of Machine Learning Research*, vol. 11, no. Dec, pp. 3371–3408, 2010.

[3] A. J. Bell and T. J. Sejnowski, "The independent components of natural scenes are edge filters," *Vision research*, vol. 37, no. 23, pp. 3327–3338, 1997.

[4] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," 2009.

[5] A. Ng, "Sparse autoencoder," *CS294A Lecture notes*, vol. 72, no. 2011, pp. 1–19, 2011.

[6] A. Ng, J. Ngiam, C. Y. Foo, Y. Mai, and C. Suen, "Ufldl tutorial," 2012.

[7] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al., "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[8] C. Charrier, O. Lézoray, and G. Lebrun, "Machine learning to design full-reference image quality assessment algorithm," *Signal Processing: Image Communication*, vol. 27, no. 3, pp. 209–219, 2012.

[9] H. Chang and M. Wang, "Sparse correlation coefficient for objective image quality assessment," *Signal processing: Image communication*, vol. 26, no. 10, pp. 577–588, 2011.

[10] D. Temel, M. Prabhushankar, and G. AlRegib, "Unique: Unsupervised image quality estimation," *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1414–1418, 2016.

[11] M. Prabhushankar, D. Temel, and G. AlRegib, "Ms-unique: Multi-model and sharpness-weighted unsupervised image quality estimation," *Electronic Imaging*, vol. 2017, no. 12, 2017.

[12] H. R. Sheikh, M. F. Sabir, and A. C. Bovik, "A statistical evaluation of recent full reference image quality assessment algorithms," *IEEE Transactions on image processing*, vol. 15, no. 11, pp. 3440–3451, 2006.

[13] N. Ponomarenko, L. Jin, O. Ieremeiev, V. Lukin, K. Egiazarian, J. Astola, B. Vozel, K. Chehdi, M. Carli, F. Battisti, et al., "Image database tid2013: Peculiarities, results and perspectives," *Signal Processing: Image Communication*, vol. 30, pp. 57–77, 2015.

[14] N. Ponomarenko, O. Ieremeiev, V. Lukin, K. Egiazarian, and M. Carli, "Modified image visual quality metrics for contrast change and mean shift accounting," in *CAD Systems in Microelectronics (CADSM), 2011 11th International Conference The Experience of Designing and Application of*. IEEE, 2011, pp. 305–311.

[15] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[16] L. Zhang and H. Li, "Sr-sim: A fast and high performance iqa index based on spectral residual," in *Image Processing (ICIP), 2012 19th IEEE International Conference on*. IEEE, 2012, pp. 1473–1476.

[17] L. Zhang, L. Zhang, X. Mou, and D. Zhang, "Fsim: A feature similarity index for image quality assessment," *IEEE transactions on Image Processing*, vol. 20, no. 8, pp. 2378–2386, 2011.

[18] K. J. Dana, B. van Ginneken, S. Nayar, and J. Koenderink, "Curet: Columbia-utrecht reflectance and texture database," 1999.

[19] M. Alfarraj, Y. Alaudah, and G. AlRegib, "Content-adaptive non-parametric texture similarity measure," in *Multimedia Signal Processing (MMSP), 2016 IEEE 18th International Workshop on*. IEEE, 2016, pp. 1–6.

[20] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 7, pp. 971–987, 2002.

[21] M. P. Sampat, Z. Wang, S. Gupta, A. C. Bovik, and M. K. Markey, "Complex wavelet structural similarity: A new image similarity index," *IEEE transactions on image processing*, vol. 18, no. 11, pp. 2385–2401, 2009.