Comparative Review of Machine Learning and Deep Learning Techniques for Texture Classification

Shantanu Kumar[1(✉)] and Amey Gupta[2]

[1] Department of Computer Science and Information Systems, Birla Institute of Technology and Science, Pilani, Hyderabad Campus, Hyderabad, India
f20190375@hyderabad.bits-pilani.ac.in

[2] Department of Electrical and Electronics Engineering, Birla Institute of Technology and Science, Pilani, Pilani Campus, Pilani, India

Abstract. In this paper, we present a review and analysis of different methods and resources for texture classification, presenting the most popular techniques that have been used in the past decade. The paper covers some of the most traditional approaches involving texture descriptors like Gray Level Co-occurrence Matrix and Gabor Filter Banks, to some of more recent approaches such as Convolutional Neural Network (CNN) and ScatNets. These methods are tested and their performance is evaluated on six standard datasets.

Keywords: Texture Classification · Gabor Filter · Convolutional Neural Networks · Feature Extraction · Image Processing

1 Introduction

Textures enable us to identify and differentiate objects around us, therefore they play a significant role in computer vision and pattern recognition applications like satellite imaging, remote sensing, and biomedical imaging [1, 2]. Texture classification, segmentation, and defect detection are essential issues in computer vision, and they play a key role in a variety of applications. The texture categorization procedure may be summarised as follows: The features that are extracted by the model from the training set are used to assign the label to test data. The set and number of texture characteristics retrieved determines how efficient the texture classification method is. The texture features can be divided into five major categories: statistical, geometrical, model based, structural, and signal processing features. Because of optimum localization in both frequency and spatial domain, feature extraction approaches such as using different filter bank methods like Gabor filters have become one of the most popular. On the other hand, since the breakthrough in 2012 ILSVRC [3], the usage of Convolutional Neural Networks has widened, which has resulted in improvement in state of art of many tasks in computer vision [4, 5]. Unlike object recognition, where shape is one of the most important features to distinguish one object from other, texture recognition depends on

V. E. Balas et al. (Eds.): AITEES 2022, AHIS 3, pp. 95–112, 2023.
https://doi.org/10.2991/978-94-6463-074-9_10

the way the pixels are arranged. For this reason, texture recognition generally depends more on methods that capture patterns of such arrangements, instead of detecting salient points as in object recognition. Given this standpoint, we provide a review of some of the most relevant techniques in recent decades. The aim of our paper is to implement and evaluate 5 different feature extraction techniques using machine learning and deep learning based models which provide an understanding for the most popular methods for this task.

2 Texture Descriptors

2.1 Gray Level Co-occurrence Matrix

A Gray Level Co-occurrence Matrix (GLCM) is a well known texture descriptor used for feature extraction. It is defined over an image to be the distribution of co-occurring pixel values at a given offset. Statistical expressions are then computed on this matrix and these are used as features in the image descriptor.

In addition to the three statistical texture descriptor features (namely Mean, Variance and Entropy) we use six other GLCM features listed below [6].

$$ASM = \sum_{i,j=0}^{N-1} \left(P_{ij}\right)^2 \tag{1}$$

$$Energy = \sqrt{\sum_{i,j=0}^{N-1} \left(P_{ij}\right)^2} \tag{2}$$

$$Entropy = \sum_{i,j=0}^{N-1} -ln\left(P_{ij}\right)P_{ij} \tag{3}$$

$$Homogeneity = \sum_{i,j=0}^{N-1} \frac{P_{ij}}{1+(i-j)^2} \tag{4}$$

$$Contrast = \sum_{i,j=0}^{N-1} P_{ij}(i-j)^2 \tag{5}$$

$$Dissimilarity = \sum_{i,j=0}^{N-1} P_{ij}|i-j| \tag{6}$$

$$Correlation = \sum_{i,j=0}^{N-1} P_{ij}\left[\frac{(i-\mu_i)(j-\mu_i)}{\sqrt{\sigma_i^2\sigma_j^2}}\right] \tag{7}$$

2.2 Gabor Filter Banks

The Gabor filter is a linear bandpass filter with significant importance in texture classification and analysis. It examines if the image has any certain frequency content in specific directions in a localized region surrounding the analysis region or point. Gabor filter orientation and frequency representations have been shown to be very useful for texture representation and differentiation. The filter comprises two components, one real and one imaginary, that represent orthogonal directions [7]. The two components may either be used separately or formed into a complex number as shown in equations below:

$$g(x, y; \lambda, \theta, \psi, \sigma, \gamma) = exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right) exp\left(i\left(2\pi\frac{x'}{\lambda}\right) + \psi\right) \tag{8}$$

$$g(x, y; \lambda, \theta, \psi, \sigma, \gamma) = exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right) cos\left(2\pi\frac{x'}{\lambda} + \psi\right) \tag{9}$$

$$g(x, y; \lambda, \theta, \psi, \sigma, \gamma) = exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right) sin\left(2\pi\frac{x'}{\lambda} + \psi\right) \tag{10}$$

where $x = xcos\theta + ysin\theta$ and $y = -xsin\theta + ycos\theta$.

The Gabor function's frequency and orientations are an important component in describing the wide variety of textures that might occur in an image. Gabor filters allow for the variation of a large number of parameters, including frequency, eccentricity, orientation, and symmetry, and the collection of such parameters is referred to as Gabor Filter Banks. Such large number of parameters can also be one of the major drawbacks. Because determining the ideal set of parameters can be challenging, filter bank design has arisen with a way for determining the ideal set of filter banks for a problem [8].

2.3 Discrete Wavelet Transforms

A discrete wavelet transform (DWT) decomposes a signal into a number of sets, each set including a time series of coefficients that describe the signal's time evolution in the associated frequency band. The family of wavelets we chose for our approach with the Gabor filters were the Daubechies Wavelets.

3 Deep Learning Approaches

3.1 ResNet

ResNets, short for Residual Networks, is a specific type of neural network introduced in 2015 [9]. Residual networks are much deeper than the networks that were used previously and are intended to make training easier. They accomplish this by using shortcuts, or skip connections, to bypass some levels. Residual Networks are based on pyramidal cell constructions in the cerebral cortex. They accomplish this by using shortcuts, or skip connections, to which helps them bypass some levels. The two primary advantages of employing skip connections are that they avoid the problem of vanishing gradients and that they mitigate the degradation problem, which was previously caused by adding more layers to deep networks resulting in large training errors (Fig. 1).

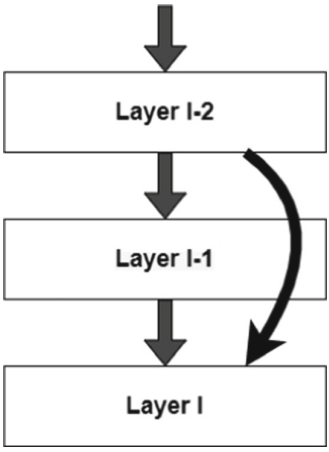

Fig. 1. A Residual Block

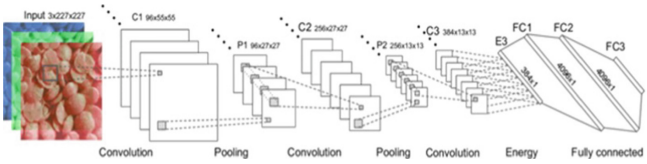

Fig. 2. Architecture of Texture CNN-3 [5]

3.2 T-CNN

Texture Convolutional Neural Network, or T-CNN, is a network built from AlexNet that tries to construct learnable filter banks that are incorporated in the CNN architecture. The network uses a new energy layer as described below along with multiple configurations of varying numbers of convolutional layers. The feature maps are pooled in the energy layer by averaging their activation output. This gives each feature map a single value, equivalent to an energy response to a filter bank [10] (Fig. 2).

3.3 ScatNet

ScatNet is a wavelet scattering network which helps in computing an image representation which is translation invariant. It is deformation stable and also retains high frequency information for classification. It cascades wavelet transform convolutions using non-linear modulus and averaging operators [11]. The convolution of the input with a mother wavelet which is rotated by θ and also dilated by 2^j is used to perform 2-D wavelet transform:

$$\psi_{j,\theta}((u)) = 2^{-j}\psi\left(2^{-j}R_{-\theta}(u)\right) \tag{11}$$

Here the rotation matrix is denoted by R, the scale is indexed by $1 \leq j \leq J$, and θ is indexed by $1 \leq k \leq K$ to produce K angles between 0 and π. The wavelet transform can then be

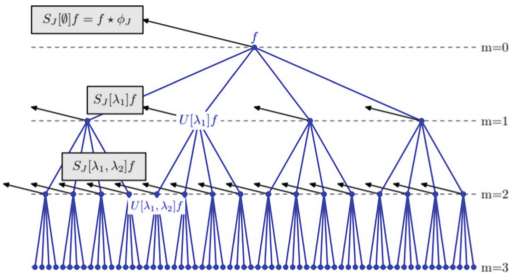

Fig. 3. Wavelet Scattering Transform [6]

rewritten after combining the indices into $\lambda = (j, k)$ followed by the representation of all the λs in the set as Λ, resulting in $|\Lambda| = JK$:

$$W_{x(c,u)} = \{x(c, u) * \phi_J(u), x(c, u) * \psi_\lambda(u)\}_{\lambda \in \Lambda} \tag{12}$$

where $\phi_J(u)$ is a low pass filter. Taking the modulus (non-linearity) of the wavelet coefficients, eliminates the output signal's high frequency oscillations while maintaining the energy of coefficients throughout the frequency range covered by $\psi_\lambda(u)$, thereby resulting in:

$$W_{x(c,u)} = \{x(c, u) * \phi_J(u), |x(c, u) * \psi_\lambda(u)|\}_{\lambda \in \Lambda} \tag{13}$$

The modulus terms can be redefined as $U\ [\lambda]\ x = |x * \psi_\lambda|$. The path can be defined as a sequence of λs given by $p = (\lambda_1, \lambda_2, \ldots \lambda_m)$ and the modulus propagator acting on path p by:

$$U[p]x = U[\lambda_m \ldots U[\lambda_2]U[\lambda_1]x \tag{14}$$

The above equation is rewritten as:

$$U[p]x = ||\ldots \left|x * \psi_{\lambda_1}\right| * \psi_{\lambda_2}| \ldots * \psi_{\lambda_m}| \tag{15}$$

The scattering coefficients can then be obtained by averaging out $U[p]x$ over a window of 2^j by the low pass filter ψ_j

$$S[p]x(u) = U[p]x * \psi_J(u) \tag{16}$$

The calculation of scattering coefficients is an iterative process as demonstrated in Fig. 3.

4 Texture Classifiers

4.1 Logistic Regression

Logistic regression, one of the simplest and widely popular machine learning algorithm uses a statistical approach to predict outputs based on a dataset's past observations. As the prediction is based on the past data values, the model becomes more and more accurate as and when more relevant data comes in. In common practice, logistic regression is used with data where the outcome is known to be binary. However, it can be used for data classification by adopting a "one versus rest" approach repeatedly to come to a conclusion.

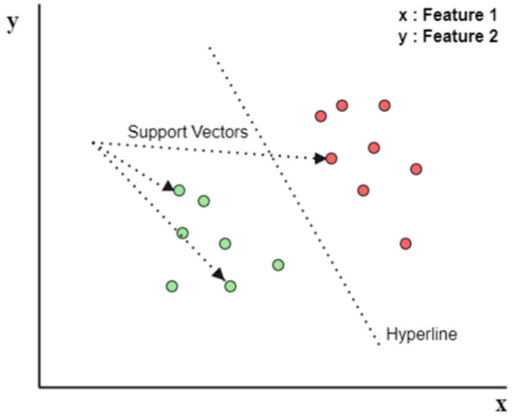

Fig. 4. A simple representation of SVM classifier

4.2 Support Vector Machines

Support-Vector Machines are machine learning models majorly used for regression and binary classification problems. It aims at modeling the data values as points in an N-dimensional plane (N being the number of features) and dividing it into two classes with a hyper-plane. The choice of the plane is done such that its distance from all the points is optimally maximum. Since, it requires the past observed values to come to this conclusion, it is a supervised learning approach. Similar to LR, SVM is also a binary classifier but it does not depend on a probabilistic theorem making it a non-probabilistic binary linear classifier. Although they are used mainly for linear classification, SVMs are also used for Non-Linear Classification using a kernel trick, where inputs are mapped in higher-dimensional feature spaces (Fig. 4).

4.3 Random Forest

A decision tree is a machine learning model which predicts an output based on simple decision making skills based on the training data. It is a kind of supervised learning approach used for regression and classification. A collection of several decision trees working at once form a random forest. Each tree in the random forest comes to a certain predicted value for the output. Then a count of each and every tree's prediction is taken and the prediction supported by the majority of the population is assumed to be the correct output. One advantage of using Random Decision Forests is that they correct the problem of overfitting on the training set by the decision trees.

4.4 K Nearest Neighbours

KNN, short for K-Nearest-Neighbours is a supervised learning approach which classifies the incoming data point based on it's distance from the prior observations from the dataset. It looks at the state of 'k' (fixed) data points in the vicinity of the given data value and the class which holds the majority of these points is predicted as the output. The range is arbitrarily determined, but the point is to take a sample of the data (Fig. 5).

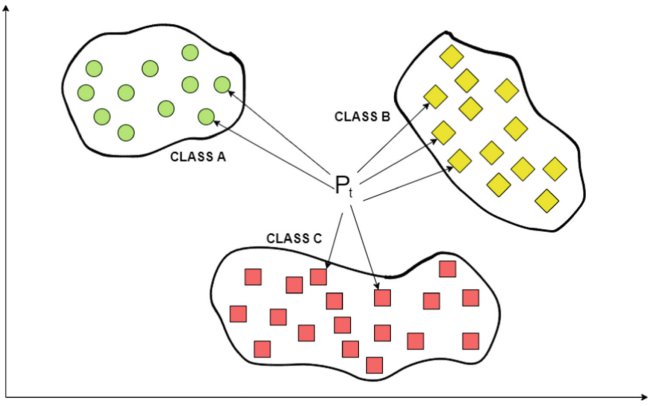

Fig. 5. An example of the KNN algorithm. Class B is the chosen class here.

4.5 Naïve Bayes

Naïve Bayes uses a simple probabilistic theorem called Bayes Theorem to make classification decisions.

$$P(A|B) = P(A)\frac{P(B|A)}{P(B)} \tag{17}$$

It gets its name from the "naive" assumption it makes about the data. It works on the assumption of independence among predictors which assumes that the features used have an equal and independent contribution and are not overlapping or dependent on each other, which is often not the case.

5 Texture Image Datasets

5.1 UMD

UMD [12] is a image dataset designed by The University of Maryland Institute for Advanced Computer Studies (UMIACS), a research group in The University of Maryland, Baltimore, USA. Consisting of 1000 images divided into 25 classes having 40 grayscale images, the dataset helps in real-life texture analysis and classification. The images in this high-resolution texture dataset (the UMD dataset) are of size 1280×960 (Fig. 6).

5.2 RawFoot

The Raw Food Texture database (RawFooT) [13] designed in The University of Milan-Bicocca, Italy focuses on testing the robustness of image descriptors and classifiers with variations in lighting conditions and color. The images are captured under 46 varying lighting and illumination settings differing in direction, color, and intensity. It is divided into 68 classes with each class containing images of raw food. For example fruits, meats, cereals, fish, grains, pulses etc. Each class has 46 different images with various lighting and color conditions, giving the dataset a total strength of 3128 images (Fig. 7).

Fig. 6. Real life grayscale texture images in UMD [7]

Fig. 7. Examples of images in RawFooT dataset [8]

5.3 CureT

Columbia-Utrecht Reflectance and Texture Database [14] (CureT) consists of 61 classes of common surfaces with images having specific focus on geometrical and lighting variations. The categories include a variety of surfaces like aluminum foil, plaster, corn husk, leather, pebbles, brick, sponge, artificial grass, lettuce etc. It covers a lot of different kinds of materials and is hence a very important dataset which is often considered when texture classification is approached.

5.4 KTH-TIPS

KTH-TIPS (Textures under varying Illumination Pose and Scale) is a texture dataset with a main aim to provide scale variations by using images captured at different distances into the training set [15] for sampling. The KTH-TIPS database has been designed to complement the CureT database with scale variations, with the pose and illumination. This database covers ten materials; sandpaper, crumpled aluminium foil, cotton, bread, styrofoam, corduroy, sponge, orange peel, linen, cotton, and crackers. Each of these is captured at nine different distances from the camera with three different directions of illumination and three different poses for each distance giving a total of 9 images per scale, and 81 images per material. For each image, a 200×200 pixels region was selected to remove the background and they were added to the dataset (Fig. 8).

5.5 KTH-TIPS-2b

The KTH-TIPS-2b dataset [16], like KTH-TIPS provides scale variations by incorporating different samples imaged at different distances. The dataset is divided into 11 classes and each class consists of 432 images. These 432 images are come from 4 different samples (with 108 images per sample) each under different lighting, pose and scale conditions. The 11 classes covered by this dataset are: tin foil, cork, white and brown bread, corduroy, lettuce, cotton, wool, cracker, linen and wood.

Fig. 8. Examples of images in KTH-TIPS database [10]

Fig. 9. Examples of images in DreTex database [12]

5.6 DreTex

The DreTex (Drexel Texture) dataset is a large dataset with over 40,000 images demonstrating the descriptiveness of the scale-space representation and the importance of geometric texture analysis [17]. DreTex covers 20 textures at different distances, with different rotations around the plane.

The database contains both regular well as irregular textures like wooden bark, carpet, toasted bread, various fabrics, sandpaper etc. (Fig. 9).

6 Implementing Texture Analysis Algorithms

We implement and compare the feature extraction and classification capabilities of different texture analysis algorithms. The datasets images are passed through different models for feature extraction and then classified on the basis of their texture. Moreover, for the CNN models, to augment the dataset, we applied various transformation techniques on our images. Each image of the dataset was cropped, resized, and normalized according to the expected input dimensions of the pretrained models. The images were then divided randomly into training, validation and testing sets as shown in Table 1.

Additionally, the CNN models were tested with different hyperparameter settings. Finally the best performing settings on each of the dataset were used as shown in Table 2.

Table 1. Number of Train, Validation and Test Images

Dataset	Training set	Validation set	Testing set
UMD	800	100	100
RawFoot	2502	312	314
CureT	11480	1435	1435
KTH-TIPS	648	81	81
KTH-TIPS-2b	2376	1188	1188
DreTex	17879	17827	5443

Table 2. Hyperparameter settings used in CNN models

Dataset	ResNet18		TS-CNN-3		Hybrid ScatNet	
	Epochs	Batch Size	Epochs	Batch Size	Epochs	Batch Size
UMD	25	32	25	32	50	32
RawFoot	15	64	10	64	20	64
CureT	15	128	25	128	50	128
KTH-TIPS	25	32	25	32	50	32
KTH-TIPS-2b	25	60	25	60	50	60
DreTex	25	128	25	128	50	128

6.1 Gabor Filter Banks with GLCM

Initially for the Machine Learning based classification we passed each image of the dataset through a set of Gabor Filter Banks (as explained in Sect. 2) with finetuned parameters. From these filtered images, we extracted 3 basic statistical features; mean, standard deviation and entropy. For large datasets, increasing the number of features extracted results in an increase in classification accuracy since the classifier is supplied with more discriminative power. For these filtered images a Gray Level Co-occurrence Matrix was then computed which further provided us with 6 additional features namely; energy, dissimilarity, homogeneity, contrast, correlation and angular second moment (ASM). In total this provided us with 252 features per image in the dataset as the size of the feature vector.

6.2 Gabor Filter Banks and Discrete Wavelet Transforms with GLCM

Each image was passed through a set of Gabor Filters like earlier however now the filtered images thus obtained were separated out and a discrete wavelet transform (refer to Sect. 2) was applied to them. The wavelet we used were Daubechies Wavelets (a family of orthogonal wavelets giving the maximal number of vanishing moments of the input) present in the `pywt` python library (an inbuilt discrete wavelet transform function

from the python wavelets library) and passed our 28 different filtered images that we received from the Gabor Filter into it, and performed a 3-layered Daubechies Wavelet Transform on it to receive 280 different images. Further we extracted features similar to the previous method using GLCM, extracting 9 features per image to get a total of 2520 features into our feature vector for each image in the dataset.

6.3 ResNet18

The network was first pretrained on the ImageNet database containing more than 1 million images in 1000 different classes [18]. The architecture is then modified by adding a new fully connected layer with output equal to number of classes in each of the datasets.

Following the training, the entire model is then finetuned to recognize different textures in our datasets. Finetuning the model greatly improves our model performance and counters the problem of model overfitting.

6.4 TS-CNN-3

The design of T-CNN presented in Sect. 3 allows for an effective integration with AlexNet. This new and modified architecture is illustrated in Fig. 10. We see that T-CNN-3 energy layer is extracted from the classic CNN architecture and its output gets concatenated with the last convolution layer's flattened output.

The outputs of the last pooling layer P5 and energy layer are first flattened out using `torch.flatten()`. They are then concatenated to form a concatenation layer and then connected to the fully connected layers without any modification from the AlexNet layers. Using this method, helps analyze both the shape and texture of the images thereby enabling to do overall shape analysis by combining both shape and texture features in the same network. All the previous and following layers (C1, C2, C3 and FC6, FC7, FC8) remain unchanged which helps to keep the memory consumption and computation time close to original AlexNet model and much less than the ResNet18 network.

6.5 Hybrid Scattering Network

Cascading modern CNN architectures on top of the scattering network can lead to high performance classification systems [19]. Consequently, we used a hybrid convolutional network with a VGG16 model created on top of applying a wavelet scattering transform.

For the applying 2D scattering transform on our images, we use the `Kymatio` package in python. For applying the 2D wavelet transform, we use $J = 2$ leading to an

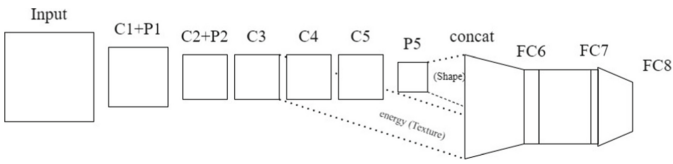

Fig. 10. TS-CNN-3 Architecture

output which is 8×8 spatially and 243 in the channel depth. We use the Morlet wavelet to define our high pass filter bank $\psi(u)$. We also apply batch normalization at the first and final layer which helps us lead to a better conditioning of the optimization.

7 Results and Discussion

The models were evaluated on the accuracy and precision scores obtained by their predictive performance on the six datasets. The following formulae were used for for the calculation of the metrics:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \times 100 \tag{18}$$

$$\text{Precision} = \frac{\text{True positives}}{\text{True positives} + \text{False positives}} \times 100 \tag{19}$$

7.1 Machine Learning Based Techniques

Using classical machine learning techniques we achieved below par results with some models but some models were able to achieve very high (>95% Accuracy) scores (Table 3).

Using only Gabor filters along with GLCM, the model which proved to be the most impressive was SVM, We were able to achieve unexpectedly high accuracies from the simple model with a significantly small feature vector. Another model which outperformed it's expectations was Logical Regression. LR proved to be just marginally

Table 3. Accuracy and Precision obtained using Gabor Filters and GLCM

		NB	LR	SVM	RF	KNN
UMD	Accuracy	77.50	97.50	97.50	94.00	91.50
	Precision	77.96	96.51	97.47	93.21	91.53
Rawfoot	Accuracy	44.00	86.30	88.50	88.00	84.20
	Precision	53.40	86.40	89.30	88.50	84.70
CureT	Accuracy	22.85	82.22	88.53	78.78	67.83
	Precision	24.11	82.42	88.04	79.48	67.31
KTH Tips	Accuracy	53.08	91.35	91.97	91.35	85.18
	Precision	60.55	91.73	92.42	92.27	87.09
KTH Tips 2b	Accuracy	47.13	58.24	60.18	57.07	52.60
	Precision	49.97	60.63	61.92	57.17	52.34
DreTex	Accuracy	48.75	85.81	86.84	81.82	72.05
	Precision	47.26	86.09	87.15	81.83	72.80

lagging behind SVM on difficult datasets like KTH Tips and DreTex. Random Forests also gave us good results. Naive Bayes owing to it's "naive" assumption of independence among predictors performed the worst compared to the other complex algorithms with accuracies as low as 22% where it's competitors were scoring in the range of 80%. This indicates that the predictors used were closely linked to each other making the assumption fail drastically. KNN performed as per its expectations, it did not give extraordinary results like SVM or LR but produced an acceptable set.

Further, on increasing the feature vector size ten folds by performing the wavelet transform on the filtered images we were able to obtain a surge in the scores. Well performing models like SVM, LR and Random Forests each observed an increase of 0.5%–2.5% on most of the datasets but for some datasets the results remained almost unchanged (Table 4).

All the models seemed to not work well for KTH-TIPS-2b, owing to the fact of having wide variety of variations in the images which makes it a very challenging dataset for the classifiers. Nevertheless, we observe significant improvement in accuracy when using discrete wavelet transform with Gabor filters. This is also evident from the confusion matrices in Figs. 11 and 12. Without applying the transform, the model wrongly identifies corduroy as white bread and wool as corduroy with normalized scores of 0.067 and 0.057 respectively. However after applying the wavelet transform, the model is able to correctly identify both corduroy and wool with normalized scores 0.051 and 0.057 respectively.

Additionally we observed that the classifiers were able to produce better results on smaller datasets like UMD, RawFoot and KTH-TIPS when compared with larger datasets like CureT and DreTex.

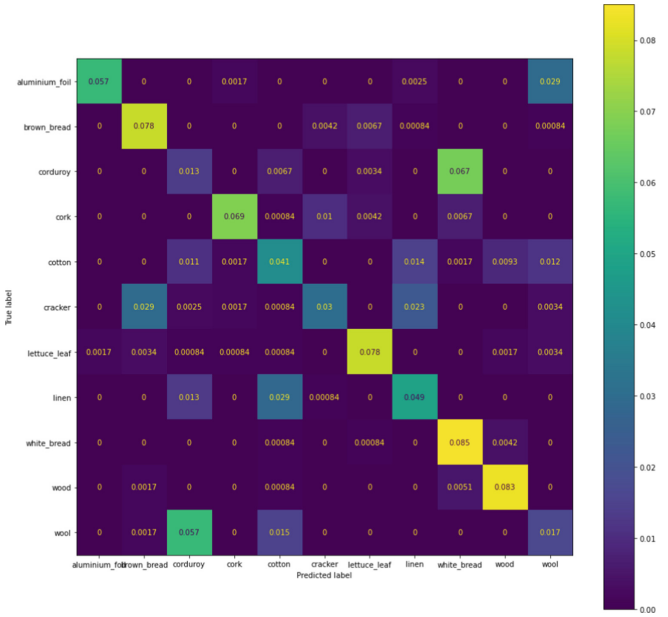

Fig. 11. Confusion Matrix for KTH-TIPS-2b without applying wavelet transform with SVM classifier

Table 4. Accuracy and Precision obtained by applying discrete wavelet transform

		NB	LR	SVM	RF	KNN
UMD	Accuracy	82.50	98.00	98.50	96.50	96.50
	Precision	81.62	97.26	98.05	96.63	95.94
Rawfoot	Accuracy	73.32	97.60	96.32	96.80	95.04
	Precision	77.52	97.94	96.99	96.9	95.42
CureT	Accuracy	22.51	87.49	88.95	77.45	73.10
	Precision	26.39	87.31	88.63	79.35	72.49
KTH Tips	Accuracy	65.43	91.97	94.44	92.59	91.35
	Precision	71.90	92.11	95.08	93.48	92.18
KTH Tips 2b	Accuracy	52.69	65.31	67.59	66.32	58.16
	Precision	58.69	65.96	70.46	68.52	60.16
DreTex	Accuracy	61.49	78.92	81.33	81.20	72.23
	Precision	66.76	79.78	82.96	81.99	75.09

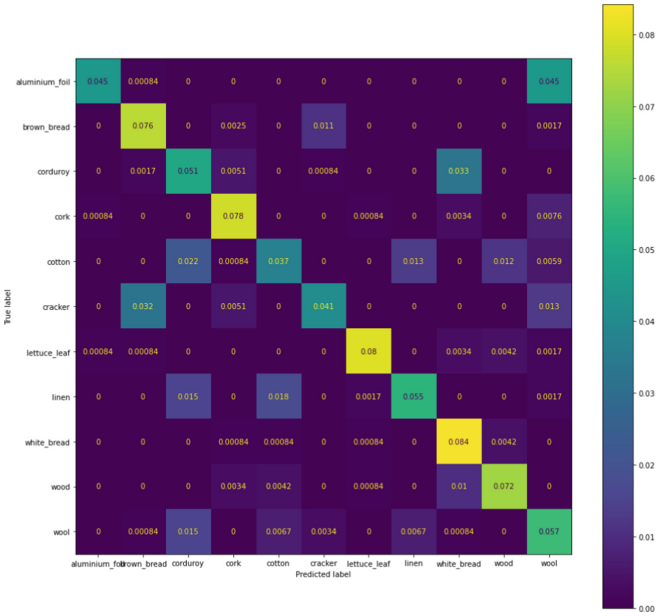

Fig. 12. Confusion Matrix for KTH-TIPS-2b applying wavelet transform with SVM classifier

7.2 CNN Based Deep Learning Techniques

The CNNs were able to produce significantly better results than the machine learning methods as expected. ResNet18 came out to be the best performing network with a 100%

accuracy in three of the six test datasets namely UMD, RawFoot and KTH-Tips, and matched the state of the art accuracy for CureT with 99.7% (Table 5).

TS-CNN-3, despite being based on the AlexNet having only 5 convolution layers is able to match ResNet18, which has 18 layers, in three of the datasets and is only marginally below for CureT, achieving an accuracy of 99.51%. It is therefore possible to obtain improved results after combining the texture and overall shape information.

On the other hand, Hybrid ScatNet was able to match TS-CNN-3 for datasets KTH-TIPS, RawFoot, and CureT but could not perform as well as its counterparts on the other datasets.

On KTH-Tips-2b the score obtained was low with an accuracy of 40.80% compared to 77.21% with ResNet18 and 71.10% with TS-CNN-3. The stark difference between

Table 5. Accuracy obtained from the CNNs

	ResNet18	Hybrid ScatNet	TS-CNN-3
UMD	100	95.00	100
RawFoot	100	99.71	100
CureT	99.70	98.40	99.51
KTH-TIPS	100	100	100
KTH-TIPS-2b	77.21	40.80	71.10
DreTex	99.10	94.21	97.20

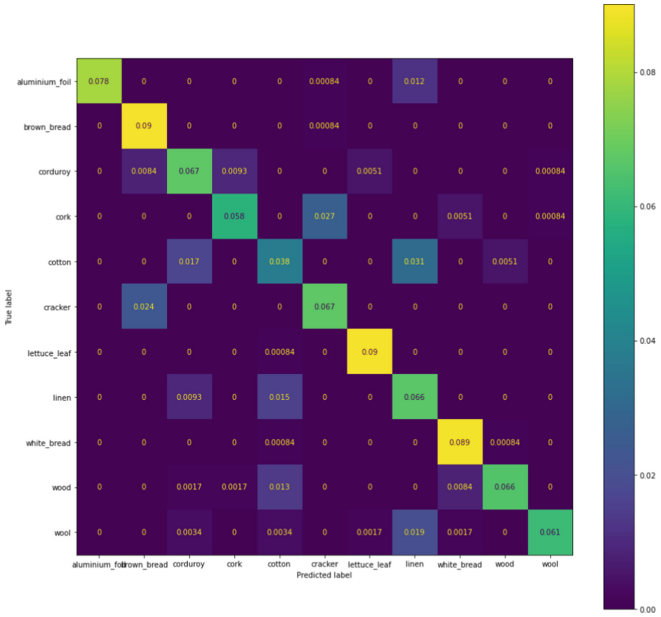

Fig. 13. Confusion Matrix for KTH-TIPS-2b with ResNet18

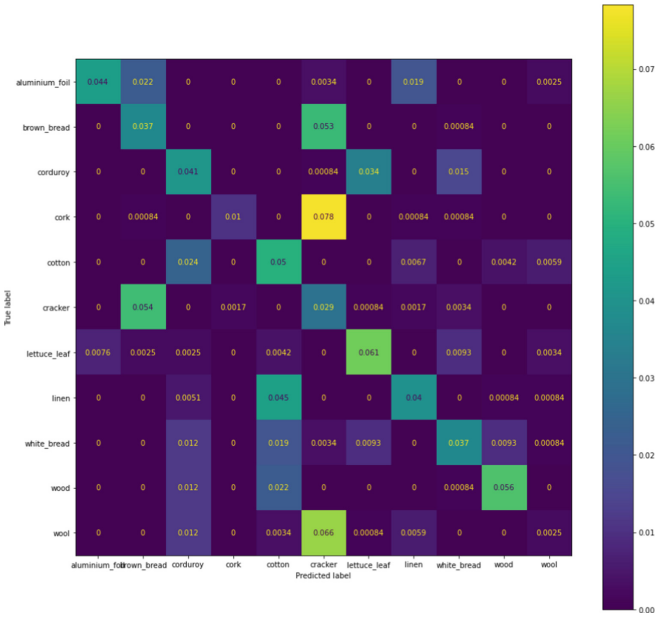

Fig. 14. Confusion Matrix of KTH-Tips-2b with Hybrid ScatNet

the accuracies of ResNet18 and our chosen Hybrid ScatNet can also be observed from the confusion matrices in Figs. 13 and 14. The ScatNet model has great difficulty in identifying 5 of the 11 classes (brown bread, cork, cracker, linen, and wool). On the other hand, the ResNet18 model is performing significantly better with only some ambiguity in identifying cork and cotton. However, the ScatNet surpasses ResNet18 in correctly identifying cotton.

The ScatNet model also saw a lower accuracy on UMD with 95% whereas the other two networks produced a perfect score. The main reason for this is the choice of hyperparameters for the scattering transform (such as number of scales and angles for wavelet transform) which were manually tuned and also consistent for all the datasets and for a simple comparison we decided not to adapt them.

KTH-TIPS-2b again witnessed the lowest accuracies among all the chosen datasets. Its wide variations in images succeeded to confuse the CNNs as well as they did with the ML Models.

8 Conclusion

In this paper we have compared and analysed various machine learning and deep learning models for classifying texture images. We first compare the results obtained using classical machine learning algorithms on Gabor filters. There is a significant improvement in accuracy and precision after applying discrete wavelet transform on images. Upon extending our research to deep learning methodologies there is a further gain in accuracy. We were able to achieve perfect scores for some datasets and matched the state

of the art for some others via our three CNN models under study. Through this paper we have compared the results of both the approaches and discussed upon the best and worst performing models and discussed and analysed the reason behind such behaviours. We also see that there is even more scope of improvement in the neural network models. Using a deeper layered network on top of these CNNs with more trainable parameters can result in even better results and higher accuracies.

Acknowledgments. We would like to thank Birla Institute of Technology and Science, Pilani for introducing us to this topic via their practice school programme. We would also like to extend our thanks to Central Electronics Engineering Research Institute, Madras Complex for providing us with access to their workstations and resources making it possible for us to run such heavy programs and work with such large datasets.

References

1. Lu, Y., Yi, S., Zeng, N., Liu, Y., Zhang, Y.: Identification of rice diseases using deep convolutional neural networks. Neurocomputing **267**, 378–384 (2017)
2. Roth, H.R., et al.: Anatomy-specific classification of medical images using deep convolutional nets. In: IEEE 12th ISBI, pp. 101–104 (2015)
3. Russakovsky, O., et al.: ImageNet large scale visual recognition challenge. Int. J. Comput. Vis. **115**(3), 211–252 (2015). https://doi.org/10.1007/s11263-015-0816-y
4. Basu, S., et al.: Deep neural networks for texture classification—a theoretical analysis. Neural Netw. **97**, 173–182 (2018)
5. Cimpoi, M., Maji, S., Vedaldi, A.: Deep filter banks for texture recognition and segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3828–3836 (2015)
6. Haralick, R.M., Shanmugam, K., Dinstein, I.H.: Textural features for image classification. **SMC-3**(6), 610–621 (1973). https://doi.org/10.1109/tsmc.1973.4309314
7. Henriksen, J.J.: 3D surface tracking and approximation using Gabor filters. South Denmark University, 28 March 2007
8. Bianconi, F., Fernández, A.: Evaluation of the effects of Gabor filter parameters on texture classification. Pattern Recogn. **40**(12), 3325–3335 (2007). https://doi.org/10.1016/j.patcog.2007.04.023
9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016). https://doi.org/10.1109/cvpr.2016.90
10. Andrearczyk, V., Whelan, P.F.: Using filter banks in convo- lutional neural networks for texture classification. Pattern Recogn. Lett. **84**, 63–69 (2016). https://doi.org/10.1016/j.patrec.2016.08.016
11. Bruna, J., Mallat, S.: Invariant scattering convolution networks. IEEE Trans. Pattern Anal. Mach. Intell. **35**(8), 1872–1886 (2013). https://doi.org/10.1109/tpami.2012.230
12. Xu, Y., Ji, H., Fermüller, C.: Viewpoint invariant texture description using fractal analysis. Int. J. Comput. Vis. **83**(1), 85–100 (2009). https://doi.org/10.1007/s11263-009-0220-6
13. Cusano, C., Napoletano, P., Schettini, R.: Evaluating color texture descriptors under large variations of controlled lighting conditions. J. Opt. Soc. Am. A **33**(1), 17 (2015). https://doi.org/10.1364/josaa.33.000017

14. Dana, K.J., van Ginneken, B., Nayar, S.K., Koenderink, J.J.: Reflectance and texture of real-world surfaces. ACM Trans. Graph. **18**(1), 1–34 (1999). https://doi.org/10.1145/300776.300778
15. Fritz, M., Hayman, E., Caputo, B., Eklundh, J.O.: THE KTH-TIPS database (2004)
16. Mallikarjuna, P.B., Targhi, A.T., Fritz, M., Hayman, E., Caputo, B., Eklundh, J.O.: THE KTH-TIPS 2 database (2006)
17. Oxholm, G., Bariya, P., Nishino, K.: The scale of geometric texture. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012. LNCS, vol. 7572, pp. 58–71. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33718-5_5
18. Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., Fei-Fei, L.: ImageNet: a large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition (2009). https://doi.org/10.1109/cvpr.2009.5206848
19. Oyallon, E., et al.: Scattering networks for hybrid representation learning. IEEE Trans. Pattern Anal. Mach. Intell. **41**(9), 2208–2221 (2019). https://doi.org/10.1109/tpami.2018.2855738