An Innovative Approach of Textile Fabrics Identification from Mobile Images using Computer Vision based on Deep Transfer Learning

Antonio Carlos da Silva Barros[‡§], Elene Firmeza Ohata[*§], Suane Pires P. da Silva[*§],
Jefferson Silva Almeida[§†] and Pedro Pedrosa Rebouças Filho[*§]
[*]Programa de Pós-Graduação em Engenharia de Teleinformática (PPGETI),
Universidade Federal do Ceará, Fortaleza, Ceará, Brazil
[†]Programa de Pós-Graduação em Engenharia Elétrica (PPGEE),
Universidade Federal do Ceará, Fortaleza, Ceará, Brazil
[‡]Universidade da Integração Internacional da Lusofonia Afro-Brasileira (Unilab), Fortaleza, Ceará, Brazil
[§]Laboratório de Processamento de Imagens, Sinais e Computação Aplicada (LAPISCO),
Instituto Federal do Ceará, Fortaleza, Ceará, Brazil
Email: {carlosbarros, elene.ohata, suanepires, jeffersonsilva}@lapisco.ifce.edu.br, pedrosarf@ifce.edu.br

***Abstract*—The identification of different textile fabrics is a task commonly learned in practice and, therefore, is considered a very strenuous and costly form of learning, causing annoyance to the individual who performs it. Based on this context, this paper proposes a new method for classifying textile fabrics, based on the development of a computer vision system using Convolutional Neural Network (CNN). CNN works as a feature extractor by incorporating the concept of Transfer Learning. Using Transfer Learning allows a pre-trained CNN model to be reused for a new problem. In order to highlight the high performance of CNN, an analysis is performed with feature extractors established in the literature. Parameters such as Accuracy, F1-Score, and processing time are considered to evaluate the efficiency of the proposed approach. For the classification were used Bayesian Classifier, Multi-layer Perceptron (MLP), k-Nearest Neighbor (kNN), Random Forest (RF), and Support Vector Machine (SVM). The results show that the best combination is the CNN architecture DenseNet201 with SVM (RBF), obtaining an accuracy of 94% and F1-Score of 94.2%.**

***Index Terms*—Textile Fabrics, Convolutional Neural Network, Transfer Learning, Computer Vision**

I. Introduction

The need for man to use a fabric that can cover his body goes back to the beginnings when there was a need for man to protect his body from cold or heat. The use and manufacture of fabrics became a basic need and later an item of luxury and social status. Clothing evolved with humanity and became a reflection of social, political, religious, and moral aspects of all stages experienced by human beings [1], [2].

Nowadays, because of the Industrial Revolution and subsequent scientific progress, few people need to know how to spin or weave, but they need to know how to judge the quality of the cloths made by the machines considering their duration. Therefore, the study of textiles becomes essential for all consumers, as well as the manufacture of the cloth, and the identification of the fiber assumes greater significance [1], [2].

There are people who learn to judge the quality of fabrics, through practice and experience, over time, but the method of "learning by making mistakes" is costly and full of hassles for a human being [1], [2].

The computer vision area has solved several problems. It is a research field that has helped humankind to customize and automate different tasks. For instance, it can be used to aid in the textile industry tasks. In the literature, some works have applied this field of knowledge to the classification of textile fibers, classification of flat fabrics, detection of defects, or inspection. In [3], an automatic system for identification of fabric structure was developed, employing the principal component analysis (PCA) and fuzzy clustering. In [4], the authors used the Local Binary Patterns and Gray-Level Co-occurence Matrix, along with artificial neural networks, to detect fabric defects. In [5], the authors proposed an algorithm for detecting defects in fabrics, which was based on biological vision modeling. In [6], a method based on lattice segmentation and lattice templates was developed to automatically identifies the defects of fabric images. While in [7], the authors presented a method for detecting fabric defects based on autoencoder. In [8], the authors used a CNN to identify fabrics, but they used a dataset with 19,894 images.

This article proposes an innovative approach to classify textile fabrics. The approach consists of the use of CNN for feature extraction, based on the definition of Transfer Learning. We evaluated the deep extractors with five classifiers. Because it is a complex method, vision-based classification relies on accurate calculations and fast processing times. Thus, to verify the performance of each classifier, two evaluation metrics were used: Accuracy (Acc) and F1-Score (F1S). Criteria such as extraction time and classification time were also measured.

The results show that DenseNet201, DenseNet169, and DenseNet121 combined with SVM reached 94.35%, 93.34%, and 93.52%, respectively, in Accuracy, demonstrating to be

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

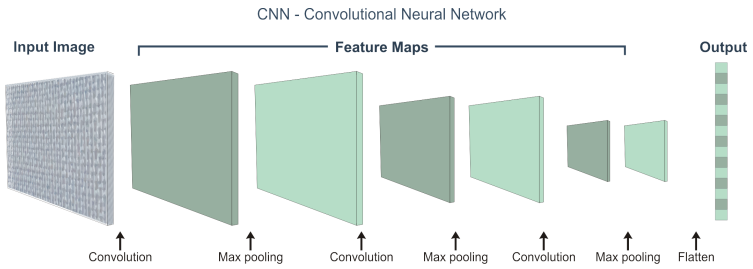


Fig. 1: Generic CNN model for Transfer Learning.

a valid and reliable approach for the classification of textile fabrics.

This article is structured as follows. Section II shows the main features of CNN and its application in our approach. Section III presents a brief summary of machine learning techniques employed. Section IV details the methodology adopted. Session V exposes the results, emphasizing and highlighting the best values. Finally, Section VI presents the conclusion and future work.

II. Convolutional Neural Network as Feature Extractor

Despite the improvement in segmentation and object detection by modern CNN architectures, it may cause overfitting problems when applied to small datasets. These architectures are usually tested in large datasets, such as ILSVRC [9] and JFT [10], resulting in the use of millions of parameters.

In order to solve this problem, researches have been using an approach called Transfer Learning [11]–[13]. The central idea of this technique is to try to transfer the "knowledge" acquired on a generic task to a specific one. For instance, we can use what was done to classify objects to identify a type of cerebral vascular accident on tomography exams. The transfer learning concept has been applied in several papers [14]–[18].

In order to be possible to apply the transfer learning on CNNs, it is necessary to follow a few steps. First, build a model, like the VGG network or Resnet. Then, train the model in a large dataset, like the ImageNet. After that, remove the last fully connected layer. Then, apply the remaining architecture, that now is a fixed transformation, to each image of the destination dataset, saving them in a new dataset with their respective labels. Finally, train a new classifier, such as Support Vector Machines or Multilayer Perceptron, using this new dataset.

Generically, Figure 1 illustrates the operation of CNN employing the concept of Transfer Learning. First, a sequence of filters is applied to the input image (convolutional layer). Then, the results are resized, resulting in a one-dimensional vector (flatten layer). After this procedure, the one-dimensional vector will store the attributes returned by CNN, based on the input image.

III. Review of Machine Learning Techniques

The classification task runs after using CNNs as a feature extractor utilizing the transfer learning approach. The outputs from the transfer learning approach are used as inputs for the classification methods. In this section, five consolidated machine learning techniques are summarized.

Bayesian Classifier, henceforth called Naive Bayes, consists of a technique of machine learning based on Bayes' theory. This method belongs to the probabilistic and supervised learning category [19].

Multilayer perceptron (MLP) is a neural network structure composed of multiple layers formed by a perceptron arrangement, with the objective of solve non-linearly separable tasks [20].

k-Nearest Neighbor (kNN) classifier defines the class to which a sample is related according to the attributes of the k nearest neighbors, obtained from the training set. It is defined as a supervised machine learning method [21].

Random Forest (RF) is classified as a supervised machine learning method and has as objective to form decision trees from a feature collection, which was arbitrarily selected from an initial set [22].

Support Vector Machines (SVM) has as its primary goal to determine classes with surfaces, or hyperplanes, that increase the distance between them [23].

IV. Methodology

In this section, we describe the proposed methodology to classify the commercial textile fabrics using CNNs as feature extractors associated with machine learning algorithms.

Figure 2 presents the steps of the methodology. The first step consists in the dataset construction with textile fabric images using an Iphone 6. The following step, which is detailed in Section IV-B, is the feature extraction from the images. The extracted features are used as input to the classifiers to build the model. The training and test steps are detailed on Section IV-C.

A. Dataset Construction

The dataset was built from a set of images divided into 14 classes of commercial fabrics. These classes are listed in Table I as well as the number of images acquired for each class. The classes are based on the work of [24], who affirmed that these classes of commercial fabrics are the most widespread in the clothing market.

A textile fabric specialist captured 48 images for each class with different capture distances. Initially, the camera was placed 5 cm away from the fabric, and then the first capture

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

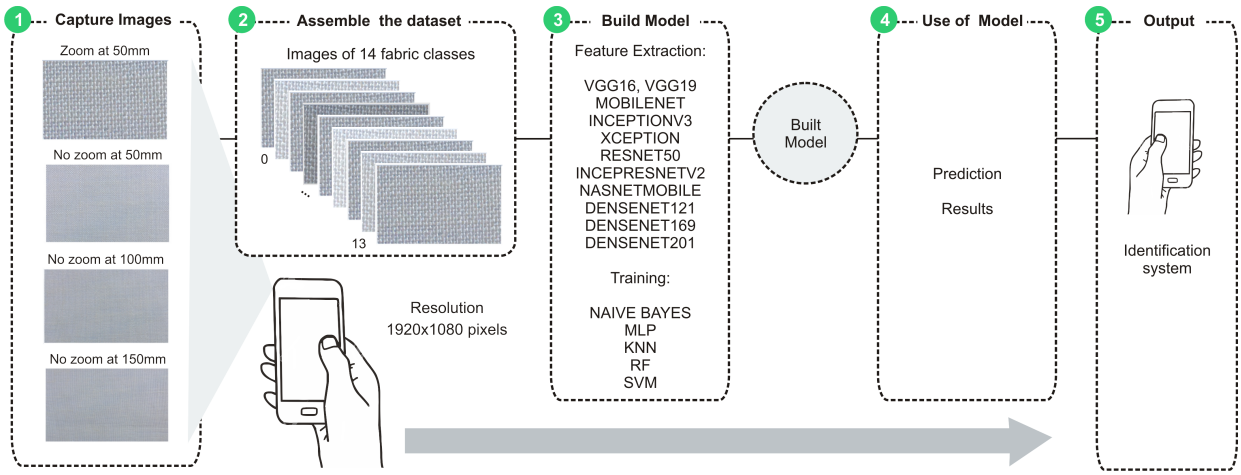


Fig. 2: Methodology of the proposed approach.

TABLE I: Classes of commercial textile fabrics used to build the training base.

Class	Textile Fabrics Type	Number of Samples
0	Denim 100% cotton	48
1	Satin 100% acetate	48
2	Satin 100% polyester	48
3	Elastane satin	48
4	Crepe 100% polyester	48
5	100% Wool	48
6	Linen 100% cotton	48
7	Mixed linen	48
8	Knits	48
9	Ramie	48
10	Satin	48
11	Textoleen 50-50	48
12	Tricoline 100% cotton	48
13	100% viscose	48

was made by observing the image with the necessary zoom to make the fabric pattern evident. The second capture occurred at the same distance as the first capture, but without the zoom feature. For the third capture, the camera was moved further 5 cm from the fabric. Finally, for the last capture, the camera and the fabric had a distance of 15 cm. Each image has a resolution of 1920 x 1080 pixels.

The sequence of images presented in Figure 3 shows the result of the image acquisition for one sample of the Ramie class. Figure 3(a) illustrates the result of capture with zoom in a 5 cm distance, while Figure 3(b) presents the fabric at same distance than Figure 3(a), but without zoom. Figures 3(c) and 3(d) show the result of a capture without zoom with distances of 10 cm and 15 cm, respectively.

B. Feature Extraction

In this paper, we use the CNN with the Transfer Learning concept. The CNN architectures considered for this work were: VGG16 [25], VGG19 [25], MobileNet [26], InceptionV3 [27], Xception [28], ResNet50 [29], InceptionResNetV2 [27], NASNetMobile [30], DenseNet121 [9], DenseNet169 [9] and DenseNet201 [9], which were pre-trained with the ImageNet database, and then we used as feature extractors in a computer vision system to identify commercial textile fabrics, composing the proposed approach. The configuration of each architecture is specified in its paper. In addition, we extract features from three well known consolidated feature extraction methods from the published literature based on texture and structure in order to compare the results obtained with the transfer learning technique. These feature extraction methods are: Gray-Level Co-Occurence Matrix (GLCM), Local Binary Patterns (LPB) and Hu's Moments (Hu). Table II presents the number of features extracted by each extractor.

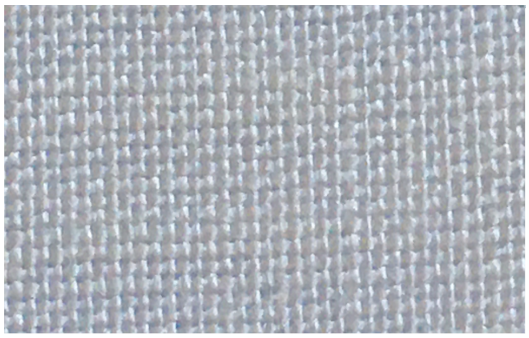
TABLE II: Number of features extracted by each extractor.

Extractor	Number of Features
VGG16	512
VGG19	512
MobileNet	1024
InceptionV3	2048
Xception	2048
ResNet50	2048
InceptionResNetV2	1536
NASNetMobile	1056
DenseNet121	1024
DenseNet169	1664
DenseNet201	1920
GLCM	9
LBP	108
Hu	7

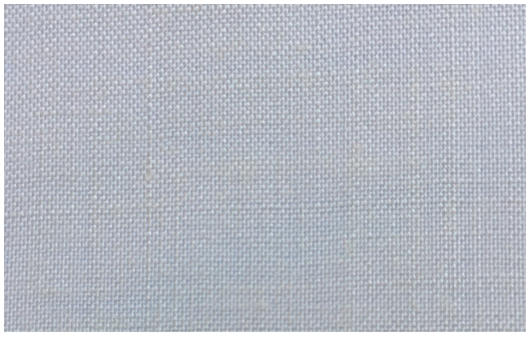
C. Model Training and Test

The classifiers used in this work have different principles, allowing the coverage of various aspects. The Naive Bayes classifier is based on probability and statistics; MLP is based on neural networks; kNN is a technique based on instances. Random Forest presents characteristics of a bagging classifier, and SVM is based on an optimal hyperplane.

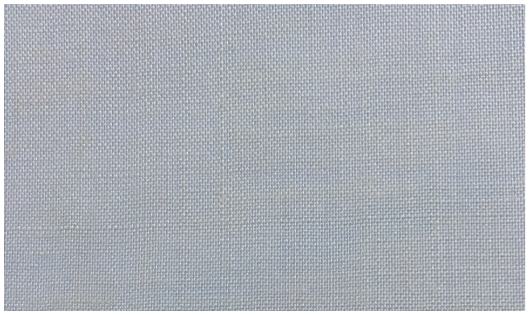
978-1-7281-6926-2/20/$31.00 ©2020 IEEE



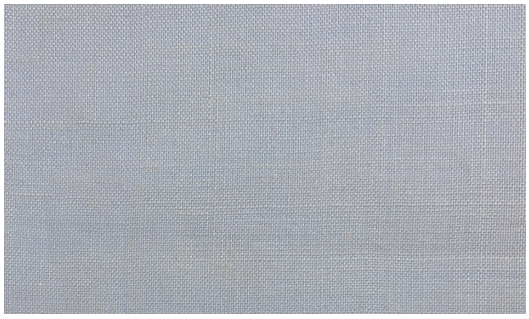
(a) With zoom and 5cm camera distance.



(b) No zoom and 5cm camera distance.



(c) No zoom and 10cm camera distance.



(d) No zoom and 15cm camera distance.

Fig. 3: Sample of *Ramie* class. (a) Image with zoom and camera placed 5 cm away of the fabric, (b) image without zoom and camera placed 5 cm away of the fabric, (c) image without zoom and camera placed 10 cm away, and (d) image without zoom and camera placed 15 cm away.

TABLE III: Hyperparameters of classifiers.

Classifier	Hyperparameter Search	Parameter	Setup
Bayes	-	-	Gaussian probability density function
kNN	Grid Search	k	3, 5, 7, 9, 11
MLP	Random Search	algoritm neuron in hidden layer	Levenberg-Marquardt method 2 to 1000
RF	Random Search	number of estimators criterion maximum depth of the tree use bootstrap samples	100 to 3000 Gini or entropy None to 6 True or False
SVM Linear	Random Search	C	2^5, 2^4, ...,2^{15}
SVM Polynomial	Random Search	C degree	2^5, 2^4, ...,2^{15} 3, 5, 7, 9
SVM RBF	Random Search	C γ	2^5, 2^4, ..., 2^{15} 2^{15}, 2^{14}, ..., 2^3

In this work, each dataset formed by the resulting attributes after the feature extraction step, had its patterns divided into two sets, where eighty percent of the samples were destined for the training phase and the remainder was submitted for the test step. We performed a 10-fold cross validation, in which an average of these 10-folds are presented on Section V.

1) Model Training: For the training step, we considered the values presented on Table III in order to find the best parameter for the training set. We investigated the Linear, Polynomial and Radial Basis Function (RBF) kernels for the SVM classifier. In the Grid Search and Random Search methods, we determined a 10-fold cross validation. The Random Search method performed a 20 iterations-search. The model with the best estimator is saved to use in the next step.

2) Model Test: With the trained model, it is possible to perform the test in the remaining dataset. The system can accept the class with the highest score. It is also possible to use the model to identify a fabric by submitting this new image to the classification algorithms, which will return a score for each class of tissue.

D. Results Evaluation

In order to evaluate the proposed methodology, we consider two evaluation metrics: Accuracy and F1-Score [31]. These metrics are calculated according to the confusion matrix. It is important to note that the classes correspond to the type of commercial fabric.

TABLE IV: Confusion matrix for the textile fabrics classification problem.

Real Class	Predicted Class				
	1	2	3	...	14
1	TP	FN	FN	FN	FN
2	FP	TN	FN	FN	FN
3	FP	FN	TP	FN	FN
...	FP	FN	FN	TN	FN
14	FP	FN	FN	FN	TN

Table IV presents the confusion matrix that portrays the types of commercial fabrics predicted correctly or incorrectly by the classification method. It is observed that the matrix is of order 14 since the number of classes is 14.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

V. Results

The inspection and identification of fabrics have several applications and is a problem that can be solved with computer vision techniques. For this reason, it is proposed, in this work, a system capable of identifying different types of textile fabrics, assisting professionals and academics.

The results of this work were obtained using several combinations of CNN architectures with classifiers. The analysis is obtained from the dataset, which was built by a specialist. Finally, the best combinations have their processing times analyzed.

The experiments were performed on a computer with a Linux Ubuntu 16.04 operating system, Core i5 2.5 GHz processor, and 8GB RAM. We should highlight that the computer does not have a Graphical Processing Unit (GPU). Both extractors and classifiers were implemented in the Python programming language.

Tables V and VI show the mean values and standard deviations of Accuracy and F1-Score, respectively, calculated combining the CNN architectures with the classifiers. The best values are highlighted in green.

Looking at the Table V, we can say that the best accuracy values were achieved by the SVM(RBF) in association with the entire CNN architecture, with the highest values being 94.350%, 93.524% and 93.339%, in combination with the DenseNet201, DenseNet121 and DenseNet169 architectures. The lowest Accuracy values were returned by combinations with GLCM, LBP, and Hu Moments.

Analyzing the Table VI, we observed that the best F1-Score values were achieved by the combinations of the DenseNet201, DenseNet121, DenseNet169 architecture with SVM(RBF), returning 94.296%, 93.454% and 93.331% in F1-Score, respectively. Still analyzing the Table VI, we can also affirm that the second best classifier, in this evaluation metric, was the kNN combined with the VGG19 architecture, reaching the value of 90.887% in F1-Score. The lowest F1-Score values were returned by combinations with GLCM, LBP, and Hu Moments.

Tables VII and VIII show the training and test times, respectively, for each classifier to perform the prediction task, considering the attributes returned by the CNN architectures and classic feature extractors.

TABLE V: **Accuracy** achieved by CNN architectures and classic extractors in combination with the classifiers.

	Classifiers						
CNN architectures	Bayes	MLP	kNN	RF	SVM(Linear)	SVM(Polyn.)	SVM(RBF)
VGG16	32.574±2.860	83.298±5.602	89.064±2.272	89.445±3.044	89.602±2.225	10.369±3.190	89.987±2.761
VGG19	31.409±3.830	88.806±4.180	90.869±0.739	90.303±2.515	89.959±2.239	10.377±3.011	90.356±2.166
MobileNet	70.259±2.871	82.865±3.978	90.074±2.319	89.823±1.413	90.077±2.306	87.963±1.596	90.685±1.637
InceptionV3	64.349±2.077	87.836±4.032	89.551±3.064	86.746±2.718	89.908±2.630	8.857±2.455	89.908±2.630
Xception	67.154±2.732	88.628±2.811	90.873±2.440	91.275±3.135	91.255±1.964	10.534±4.155	91.834±1.464
ResNet50	35.571±4.214	51.160±10.190	85.704±3.114	85.704±3.507	86.611±1.331	14.007±3.624	87.795±1.553
InceptionResNetV2	65.844±3.208	88.595±2.162	89.553±1.439	87.264±2.490	90.496±1.535	12.712±2.910	90.496±1.535
NASNetMobile	60.150±5.399	77.073±8.832	89.088±4.335	89.068±4.355	89.506±3.985	64.932±6.758	90.089±3.726
DenseNet121	58.343±2.605	90.414±1.436	92.371±1.497	92.476±2.866	92.319±2.523	11.496±4.381	**93.524±1.666**
DenseNet169	53.342±4.123	92.402±1.142	92.785±1.383	92.203±2.486	93.131±1.910	10.919±4.141	**93.339±1.792**
DenseNet201	56.120±2.557	93.177±1.729	92.798±0.767	93.023±2.017	93.547±1.452	9.049±2.314	**94.350±1.177**
GLCM	13.439±0.427	17.065±1.973	26.381±4.332	64.298±4.302	16.895±2.650	11.783±2.198	16.346±1.243
LBP	13.237±0.249	28.557±8.618	67.583±3.395	67.611±3.215	65.377±5.756	8.324±1.318	82.059±1.028
Hu	13.023±0.890	17.806±3.550	69.129±2.612	73.812±4.592	35.232±3.597	37.199±2.437	51.071±6.374

TABLE VI: ***F1-Score*** achieved by CNN architectures and classic extractors in combination with the classifiers.

	Classifiers						
CNN architectures	Bayes	MLP	kNN	RF	SVM(Linear)	SVM(Polyn.)	SVM(RBF)
VGG16	32.270±3.648	88.284±3.076	89.130±2.381	89.437±3.090	89.618±2.399	3.144±2.366	89.938±2.904
VGG19	30.601±5.164	88.804±4.150	**90.887±0.807**	90.224±2.611	89.910±2.244	3.870±3.175	90.334±2.142
MobileNet	68.548±4.677	82.283±4.251	90.068±2.344	89.821±1.442	90.067±2.352	88.024±1.639	90.695±1.632
InceptionV3	64.479±2.523	87.845±3.895	89.421±3.020	86.413±2.814	89.720±2.754	1.590±0.846	89.720±2.754
Xception	66.014±2.614	88.250±2.990	90.932±2.423	91.317±3.162	91.277±1.850	2.663±2.093	91.808±1.454
ResNet50	33.146±5.804	48.914±11.683	85.559±3.121	85.740±3.317	86.479±1.231	9.104±4.631	87.694±1.490
InceptionResNetV2	64.507±3.407	88.615±2.206	89.472±1.450	87.107±2.568	90.506±1.537	8.094±3.219	90.506±1.537
NASNetMobile	59.280±5.183	76.568±9.399	89.087±4.419	88.988±4.415	89.454±4.131	64.317±6.526	90.081±3.834
DenseNet121	58.041±2.820	90.339±1.578	92.350±1.481	92.414±2.968	92.277±2.567	4.699±4.411	**93.454±1.673**
DenseNet169	50.863±4.610	92.374±1.155	92.869±1.302	92.105±2.554	93.139±1.896	4.513±4.148	**93.331±1.767**
DenseNet201	54.283±2.851	93.137±1.690	92.757±0.801	93.030±1.951	93.509±1.406	1.778±0.797	**94.296±1.170**
GLCM	3.187±0.191	7.217±1.001	25.428±4.173	64.039±4.725	13.079±3.115	5.881±2.184	11.730±1.929
LBP	3.187±0.191	7.217±1.001	25.428±4.173	64.039±4.725	13.079±3.115	5.881±2.184	11.730±1.929
Hu	3.124±0.217	10.153±2.740	68.807±2.773	73.611±4.986	31.965±3.345	32.567±2.442	48.926±6.521

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

TABLE VII: **Training** time, in seconds, obtained by the classifiers.

	Classifiers						
CNN architectures	Bayes	MLP	kNN	RF	SVM(Linear)	SVM(Polin.)	SVM(RBF)
VGG16	0.021±0.014	82.076±5.683	0.007±0.003	7.265±0.543	0.437±0.006	0.963±0.124	0.513±0.045
VGG19	0.025±0.007	101.236±14.732	**0.006±0.001**	14.967±0.117	0.417±0.012	0.867±0.018	0.480±0.015
MobileNet	0.019±0.004	82.949±14.151	0.015±0.001	5.264±0.035	0.976±0.014	0.963±0.052	1.616±0.027
InceptionV3	0.030±0.014	153.560±16.144	0.066±0.029	18.532±0.464	2.929±0.179	3.696±0.475	3.078±0.106
Xception	0.033±0.017	41.710±6.431	0.038±0.008	34.248±1.013	2.037±0.278	3.497±0.464	2.040±0.206
ResNet50	0.027±0.006	119.065±49.501	0.049±0.023	23.486±0.219	1.433±0.029	3.445±0.127	2.113±0.040
InceptionResNetV2	0.025±0.005	149.043±35.772	0.015±0.001	22.713±0.291	1.448±0.033	2.537±0.054	1.543±0.044
NASNetMobile	0.018±0.010	70.123±10.366	0.012±0.002	5.129±0.063	1.057±0.014	1.648±0.026	1.243±0.088
DenseNet121	0.023±0.015	132.330±16.677	0.016±0.001	30.174±0.309	0.830±0.006	1.696±0.035	1.524±0.028
DenseNet169	0.030±0.013	123.957±15.914	0.026±0.013	8.248±0.255	1.576±0.176	3.043±0.586	2.126±0.220
DenseNet201	0.031±0.010	138.051±12.765	0.027±0.013	24.287±0.754	1.744±0.035	3.380±0.265	2.502±0.242
GLCM	0.001±0.000	13.704±4.254	0.001±0.000	16.147±0.084	0.100±0.006	0.056±0.006	0.144±0.015
LBP	0.004±0.005	61.965±33.930	0.001±0.000	32.286±1.431	0.424±0.038	0.148±0.010	0.345±0.021
Hu	0.002±0.000	9.663±3.686	0.001±0.000	10.501±0.609	0.390±0.016	0.134±0.011	0.242±0.022

TABLE VIII: **Testing** time, in seconds, obtained by the classifiers.

	Classifiers						
CNN architectures	Bayes	MLP	kNN	RF	SVM(Linear)	SVM(Polin.)	SVM(RBF)
VGG16	**0.001±0.001**	0.009±0.004	0.013±0.001	0.179±0.009	0.008±0.000	0.009±0.001	0.008±0.000
VGG19	**0.001±0.001**	0.011±0.001	0.014±0.001	0.162±0.001	0.008±0.000	0.009±0.000	0.008±0.000
MobileNet	0.002±0.001	0.020±0.002	0.028±0.001	0.163±0.002	0.016±0.001	0.016±0.001	0.018±0.001
InceptionV3	**0.001±0.001**	0.026±0.005	0.058±0.003	0.160±0.006	0.037±0.002	0.037±0.002	0.034±0.003
Xception	**0.001±0.001**	0.005±0.001	0.053±0.004	0.178±0.004	0.029±0.001	0.035±0.003	0.033±0.003
ResNet50	**0.001±0.001**	0.025±0.010	0.039±0.004	0.167±0.002	0.029±0.001	0.035±0.001	0.032±0.001
InceptionResNetV2	**0.001±0.001**	0.017±0.008	0.041±0.002	0.167±0.002	0.023±0.001	0.026±0.001	0.024±0.001
NASNetMobile	**0.001±0.001**	0.020±0.002	0.029±0.001	0.161±0.002	0.016±0.001	0.018±0.001	0.017±0.001
DenseNet121	**0.001±0.001**	0.017±0.003	0.027±0.001	0.157±0.001	0.015±0.001	0.018±0.000	0.017±0.000
DenseNet169	**0.001±0.001**	0.018±0.005	0.041±0.002	0.167±0.016	0.023±0.001	0.029±0.001	0.025±0.002
DenseNet201	**0.001±0.001**	0.023±0.004	0.045±0.002	0.153±0.003	0.028±0.001	0.032±0.001	0.030±0.002
GLCM	0.001±0.000	0.004±0.002	0.003±0.000	0.152±0.001	0.001±0.000	0.001±0.000	0.001±0.000
LBP	0.001±0.000	0.005±0.002	0.003±0.000	0.157±0.002	0.001±0.000	0.002±0.000	0.002±0.000
Hu	0.001±0.000	0.005±0.002	0.003±0.000	0.156±0.001	0.001±0.000	0.001±0.000	0.001±0.000

TABLE IX: Extraction time for each extractor.

CNN Architectures	**Extraction Time** (ms)
VGG16	**164.42±0.50**
VGG19	**188.11±0.14**
MobileNet	**94.01±0.55**
InceptionV3	274.61±0.30
Xception	264.12±0.42
ResNet50	188.30±0.64
InceptionResNetV2	632.33±0.32
NASNetMobile	298.43±0.08
DenseNet121	507.37±0.02
DenseNet169	666.03±0.11
DenseNet201	832.14±0.20
GLCM	17.615±9.19
LBP	502.998±194.10
Hu	50.570±20.696

Analyzing the Table VII, we observed that the kNN classifier provides the best training times in combination with all CNN architectures, the lowest value being returned when combined with VGG19 architecture, with 0.006s. On the other hand, MLP has the worst training time values, in combination with all CNN architectures, with its highest value being 153.560s.

Looking at Table VIII, we can see that the fastest execution times were returned by the Naive Bayes classifiers, their lowest value being 0.001s, in combination with all CNN architectures, except the MobileNet architecture, with which the Naive Bayes reached a time of 0.002s. We can also verify that the Random Forest classifier had the worst test times, its highest value being 0.179s when in association with the VGG16 architecture.

Regarding the extraction time, Table IX shows the time of each CNN architecture to accomplish its task. Looking at Table IX, it is noted that the best extraction times are returned by the MobileNet, VGG16 and VGG19 architectures, with the values of 94.01s, 164.42s and 188.11s, respectively.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

The CNN architecture MobileNet can extract features from an image in the shortest time. This allows its use on an embedded system. However, the best combination of this architecture is with the KNN classifier with a F1-Score value of 90.068%. On the other hand, DenseNet201 architecture takes longer to extract the features of an image, nonetheless, when associated with the SVM(RBF) classifier, it returned the best Accuracy and F1-Score (94.350% and 94.296%, respectively). The combination DenseNet201 with SVM(RBF) classifier still a potential combination since the time to the extract and test a new image would be less than 1 second.

Therefore, we can note from the presented tables that the classic methods of feature extractions are not viable for this application. Even though they achieved proper extraction and test times, they did not reach satisfactory accuracy and F1-Score. This phenomenon can be explained by the images with different distances, thus preventing a precise extraction, mainly of texture.

VI. Conclusions and Future Works

This paper proposed a system to identify textile fabrics utilizing CNNs to extract features from the images from the transfer learning concept.

According to results, we can conclude that images of a real-world scenario from a high resolution camera can be used in the task of identifying fabrics for making clothing. Furthermore, CNN architectures can be used as feature extractors for this problem; the combination DenseNet201-SVM(RBF) achieved the best performance in terms of accuracy, reaching 94.350%. This work also developed a prediction system that can be used as a service for several applications, since we have the built models (extractor-classifier) saved.

For future works, we intent to acquire new images from the same classes of the dataset used, and add new classes to the dataset. Other CNN architectures can also be used, such as ResNeXt-50 [32] and CapsNet [33], as well as other machine learning methods, such as Optimum-Path Forest [34]. In addition, a mobile application can be developed to integrate the proposed system with a smartphone camera.

Acknowledgments

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001. Also Pedro Pedrosa Rebouças Filho acknowledges the sponsorship from the Brazilian National Council for Research and Development (CNPq) via Grants Nos. 431709/2018-1 and 311973/2018-3. The authors would like to thank The Ceará State Foundation for the Support of Scientific and Technological Development (FUNCAP) for the financial support (6945087/2019).

References

[1] Jacintha and S. karthikeyan, "A survey on various fabric defect detection methods," in *2019 International Conference on Communication and Signal Processing (ICCSP)*, April 2019, pp. 0801–0805.

[2] R. Divyadevi and B. V. Kumar, "Survey of automated fabric inspection in textile industries," in *2019 International Conference on Computer Communication and Informatics (ICCCI)*, Jan 2019, pp. 1–4.

[3] X. Wang, N. D. Georganas, and E. M. Petriu, "Automatic woven fabric structure identification by using principal component analysis and fuzzy clustering," in *2010 IEEE Instrumentation Measurement Technology Conference Proceedings*, May 2010, pp. 590–595.

[4] O. Kaynar, Y. E. Işik, Y. Görmez, and F. Demirkoparan, "Fabric defect detection with lbp-glmc," in *2017 International Artificial Intelligence and Data Processing Symposium (IDAP)*, Sep. 2017, pp. 1–5.

[5] C. Li, G. Gao, Z. Liu, M. Yu, and D. Huang, "Fabric defect detection based on biological vision modeling," *IEEE Access*, vol. 6, pp. 27 659–27 670, 2018.

[6] L. Jia, J. Zhang, S. Chen, and Z. Hou, "Fabric defect inspection based on lattice segmentation and lattice templates," *J. Franklin Institute*, vol. 355, no. 15, pp. 7764–7798, 2018.

[7] H. Tian and F. Li, "Autoencoder-based fabric defect detection with cross-patch similarity," in *2019 16th International Conference on Machine Vision Applications (MVA)*, May 2019, pp. 1–6.

[8] X. Wang, G. Wu, and Y. Zhong, "Fabric identification using convolutional neural network," in *International Conference on Artificial Intelligence on Textile and Apparel*. Springer, 2018, pp. 93–100.

[9] R. Olga, D. Jia, S. Hao, K. Jonathan, S. Sanjeev, M. Sean, H. Zhiheng, K. Andrej, K. Aditya, B. M. S., B. A. C., and L. Fei-Fei, "Imagenet large scale visual recognition challenge." *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[10] G. Hinton, "Neural networks for machine learning - coursera.org," 2012.

[11] K. R. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning." *J. Big Data*, vol. 3, p. 9, 2016.

[12] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks." in *CVPR*. IEEE Computer Society, 2014, pp. 1717–1724.

[13] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, Oct 2010.

[14] T. Grubinger, G. Chasparis, and T. Natschläger, "Generalized online transfer learning for climate control in residential buildings," *Energy and Buildings*, vol. 139, pp. 63–71, 01 2017.

[15] C. Mazo, J. Bernal, M. Trujillo, and E. Alegre, "Transfer learning for classification of cardiovascular tissues in histological images," *Computer Methods and Programs in Biomedicine*, vol. 165, 08 2018.

[16] H. K. Suh, J. IJsselmuiden, J. W. Hofstee, and E. J. van Henten, "Transfer learning for the classification of sugar beet and volunteer potato under field conditions," *Biosystems Engineering*, vol. 174, pp. 50 – 65, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1537511016308777

[17] Q. Lin, J. Hong, Z. Liu, B. Li, and J. Wang, "Investigation into the topology optimization for conductive heat transfer based on deep learning approach," *International Communications in Heat and Mass Transfer*, vol. 97, pp. 103 – 109, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0735193318301593

[18] Q. Wang, P. Du, J. Yang, G. Wang, J. Lei, and C. Hou, "Transferred deep learning based waveform recognition for cognitive passive radar," *Signal Processing*, vol. 155, pp. 259 – 267, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0165168418303256

[19] S. Theodoridis and K. Koutroumbas, *Pattern Recognition, Fourth Edition*. Academic Press, 2009.

[20] S. Haykin, *Neural networks and learning machines*. Pearson Upper Saddle River, NJ, USA:, 2009, vol. 3.

[21] K. Fukunaga and P. M. Narendra, "A branch and bound algorithm for computing k-nearest neighbors," *IEEE Transactions on Computers*, vol. C-24, no. 7, pp. 750–753, July 1975.

[22] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, Oct 2001.

[23] V. N. Vapnik, *Statistical Learning Theory*. Wiley-Interscience, 1998.

[24] L. Hendrick, *Apostila de tecnologia têxtil. Curso de Economia Doméstica*, Universidade Federal do Ceará, Fortaleza-Ce, 2013.

[25] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, cite arxiv:1409.1556.

[26] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel, "Handwritten digit recognition with a back-propagation network." in *NIPS*, D. S. Touretzky, Ed. Morgan Kaufmann, 1989, pp. 396–404.

[27] C. Szegedy, Wei Liu, Yangqing Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 1–9.

[28] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," 2016.
[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.
[30] H. Pham, M. Guan, B. Zoph, Q. Le, and J. Dean, "Efficient neural architecture search via parameters sharing," in *Proceedings of the 35th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, J. Dy and A. Krause, Eds., vol. 80. Stockholmsmässan, Stockholm Sweden: PMLR, 10–15 Jul 2018, pp. 4095–4104.
[31] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing Management*, vol. 45, no. 4, pp. 427 – 437, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0306457309000259
[32] S. Xie, R. B. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, 2017, pp. 5987–5995.
[33] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," *CoRR*, vol. abs/1710.09829, 2017.
[34] T. M. Nunes, A. L. V. Coelho, C. A. M. Lima, J. P. Papa, and V. H. C. de Albuquerque, "EEG signal classification for epilepsy diagnosis via optimum path forest - A systematic assessment," *Neurocomputing*, vol. 136, pp. 103–123, 2014.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE