Scientific Research Publishing

Journal of Computer and Communications, 2019, 7, 158-165
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

Classification of 3D Film Patterns with Deep Learning

John Mlyahilu, Youngbong Kim, Jongnam Kim

Department of IT Convergence and Application Engineering, Pukyong National University, Busan, Korea
Email: johnmlyahilu@gmail.com, ybkim@pknu.ac.kr, jnkim1225@gmail.com

How to cite this paper: Mlyahilu, J., Kim, Y. and Kim, J. (2019) Classification of 3D Film Patterns with Deep Learning. *Journal of Computer and Communications*, **7**, 158-165.
https://doi.org/10.4236/jcc.2019.712015

Received: December 10, 2019
Accepted: December 24, 2019
Published: December 27, 2019

Abstract

Researches on pattern recognition have been tremendously performed in various fields because of its wide use in both machines and human beings. Previously, traditional methods used to study pattern recognition problems were not strong enough to recognize patterns accurately as compared to optimization algorithms. In this study, we employ both traditional based methods to detect the edges of each pattern in an image and apply convolutional neural networks to classify the right and wrong pattern of the cropped part of an image from the raw image. The results indicate that edge detection methods were not able to detect clearly the patterns due to low quality of the raw image while CNN was able to classify the patterns at an accuracy of 84% within 1.5 s for 10 epochs.

Keywords

Patterns, Recognition, Accuracy, CNN, Edge Detection, Classifications

1. Introduction

Before the rise of researches in pattern recognition field, images of objects and other structures were traditionally represented by vector spaces [1]. This was easily presented by the use of front, left, right and rear elevation views [2]. Generally, the traditional ways to represent objects usually oversimplify the nature, shape and size of the objects. Substantially, objects are made by patterns that can be simply observed physically and or hardly observed by applying different mathematical algorithms. In this context, we define pattern recognition as the process of data classification based on the statistical information extracted from the objects' representation [3]. As per its essence in solving classification and clustering problems, pattern recognition is widely applied in various fields including natural language processing, medical diagnosis, image

and video processing, document recognition, a list is too long to mention as referenced in [4] and [5].

Within all the fields of application mentioned thereof, a pattern recognition problem uses raw data which are processed to a format that a machine can comprehend. Pattern recognition problems are practically solved by different approaches including discriminant analysis, statistical decision theory, structural pattern recognition, combinatorial approaches, support vector machines and syntactic pattern recognition [6]. As its essence has been outlined in the previous paragraphs of this work, we introduce a pattern recognition problem that involves detection of wrong and right patterns and classification of the detected patterns with faults using Convolutional Neural Networks.

2. Related Works

Several works have employed Laplacian, Sobel and Canny edge detectors [7] as preliminary steps for pattern recognition. In this work, we apply Laplacian edge detector because it uses kernel or filter and calculates the second order derivatives in a single pass of an image intensity. The Laplacian of an image is given by the following equation.

$$L(x,y)=\frac{\partial^2 I}{\partial x^2}+\frac{\partial^2 I}{\partial y^2} \tag{1}$$

where $I(x,y)$ is the intensity of an image in reference.

We employed Sobel operator that performs a 2-D operator spatial gradient measurement on an Image that emphasizes the regions of high spatial frequency that correspond to the edges.

$$|G|=\sqrt{G_x^2+G_y^2} \tag{2}$$

where G_x and G_y is the rotation and magnitude of the gradient respectively.

We lastly employed Canny edge detection which is a multi-step algorithm that can detect edges with noise suppressed at the same time. This process involves smoothing, gradient computing, thresholding M, suppressing non-maxima pixels and linking the edge segments to form continuous edges. The following are the procedures for canny edge detection.

- Define the Gaussian filter $G_\sigma=\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{m^2+n^2}{2\sigma^2}\right)$ to reduce noise and unwanted details and textures.
- Smoothen the image with the Gaussian filter $g(m,n)=G_\sigma(m,n)*f(m,n)$.
- Compute gradient of $g(m,n)$ using any of the gradient operators (Roberts, Sobel, Prewitt, etc) to get $M(m,n)=\sqrt{g_m^2(m,n)+g_m^2(m,n)}$ and $\theta(m,n)=\tan^{-1}\left[g_n(m,n)/g_m(m,n)\right]$.
- Determine the minimum threshold M, mathematically defined as $M_T(m,n)=\left\{\begin{matrix}M(m,n)\ if\ M(m,n)>T\\ 0\ otherwise\end{matrix}\right\}$.

If the above simple techniques do not perform better, some machine learning

methods can be introduced to serve the purpose. Machine learning metric measures are used to validate whether the images are suitable for pattern recognition and segmentation [8] and [9]. The precise measures for thresholding are F_1-Score and Mathew Correlation Coefficient (MCC) that uses the original and predicted edges of an image.

We define a confusion matrix with two elements as Foreground and Background for each edge cases of the image to be recognized and segmented from the raw image. The following are the edge cases that considered during the calculation of the thresholding measurements.

From **Table 1** we define the accuracy, F_1-score and MCC measure of the classification as Equation (3), (4) and (5) respectively.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{3}$$

$$\text{F}_1\text{-score} = \frac{2 * TP}{2 * TP + FP + FN} \tag{4}$$

$$\text{MCC} = \frac{TP * TN - FP * FN}{\sqrt{(TP + FT)(TP + FN)(TN + FP)(TN + FN)}} \tag{5}$$

F_1-score ranges from 0 to 1 showing that the higher the score close to 1 the better the prediction for the edges, while MCC lies between −1 and +1. MCC has double meaning whereby −1 states the negative correlation between the original edges and the detected edges (mismatching) while +1 shows the relationship between the predicted and original edges also known as absolute matching [9].

Selecting a Template

The figure below represents workflow that indicates the procedures on how to obtain the expected results using the proposed algorithms. The figure describes the process from raw image data to the measure of performance of the Convolutional Neural Networks model.

Figure 1 indicates that the colored image will be converted to gray scale by either open CV or matplot python libraries and then edge detection methods will be applied. Depending on the nature of the image, if the edge detection methods will not perform better then machine learning metrics will be applied for thesholding process. When the accuracy achieved will be smaller than 75% as described in [9] and [12], then the process goes back to the initial step for cropping different patterns for binary classification processing in CNN. Otherwise, the algorithm produces clear output for the raw image with faults.

Table 1. Confussion matrix.

		Original Edges	
Predicted Edges		Foreground(255)	Background(0)
	Foreground(255)	True Positive (TP)	False Positive (FP)
	Background(0)	False Negative (FN)	True Negative (TN)

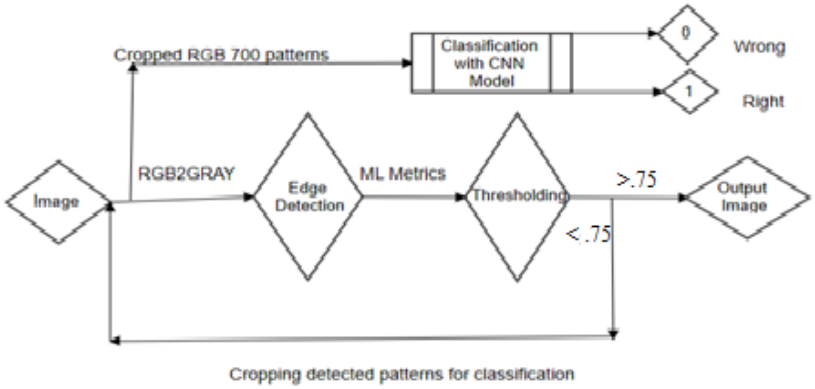

Figure 1. Proposed workflow for image processing, manipulation and classification.

3. Methodology and Experiments

Input data was obtained as an image with two different patterns and with non-uniform intensity. The image used was 383 by 676 in pixels and 324.7 KB where traditional methods were applied to detect the similar patterns in the colored image. We applied the pattern detection algorithm based on the patterns and the color intensity of the image. We also used template matching method to determine the patterns on the image where we were able to detect different patterns on the colored image. The methods used for edge detections are Laplacian, Sobel and Canny edge detectors.

We used the cropped template in **Figure 2** and OpenCV library for image manipulation. Based on that template we were able to detect the matched patterns as shown in the following figure.

Figure 3 shows matched templates when the maximum matching threshold was set to 0.7 to avoid overwriting of matched templates on the colored image. From the colored image in **Figure 3**, we cropped different patterns of which some have wrong and right patterns. We had 700 images whose sizes were different because of cropping process and some of the images were rotated at an angle of 10˚. The images were resized to 40 by 40 pixels in order to be compatible with CNN. The data set was divided into training set and test sets whereby the former had 450 images and the later had 250 images respectively. We employed Convolutional Neural Network to classify the binary classification problem and determine the efficiency of our model. The CNN architecture accomplished both feature extraction and classification activities for the images.

The CNN architecture shown in **Figure 4** was implemented in Python using Keras and Tensorflow libraries with GPU having two convolution layers with 32 and 64 nodes respectively. Two maximum pooling layers were put after every convolution layer to reduce the number of parameters in order to control overfitting and computational costs. The activation function for convolution layers was ReLU, applied for the sake of converting all negative pixel values to zero and for the last fully connected layer was Softmax that normalizes the probabilities to the interval [0, 1]. We used "adam" as the optimization function because it computes individual learning rates for different parameters [13].

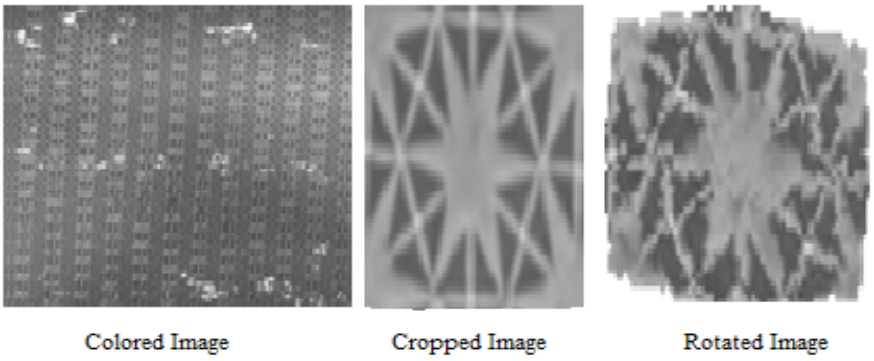

Figure 2. Showing colored image, cropped pattern and rotated pattern.

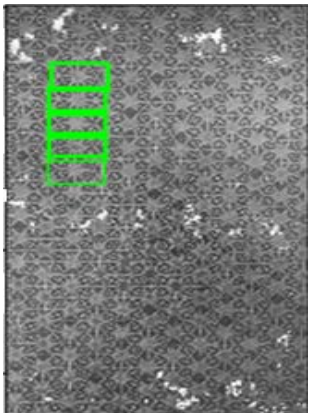

Figure 3. Recognized patterns from template matching process.

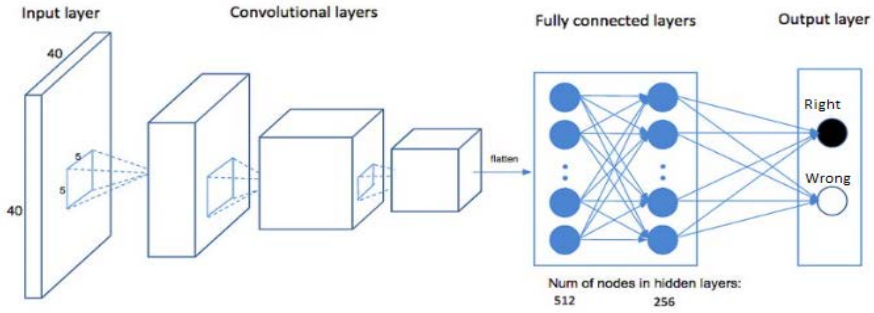

Figure 4. Convolutional Neural Network archtecture for binary classification referred from [10].

4. Results and Discussion

The results are presented as obtained by edge detectors' methods, thresholding metric measures and Convolutional Neural Networks. With these three algorithms, the results were displayed in a figure where all the algorithms are applied. To obtain precise and accurate results, images must of high quality, detectable background and foreground, and uniform intensity. As shown in **Figure 2** the original image was colored however, its intensity was not uniform. **Figure 5** shows all the methods used to determine patterns in the original image as explained in the methodology and experiment section.

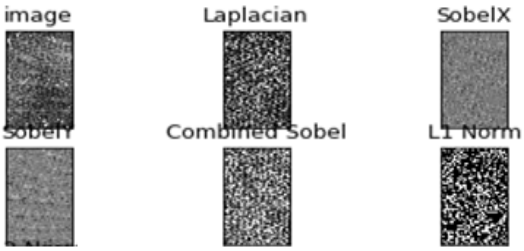

Figure 5. Edge detection results for Laplacian, Sobel and Canny methods.

4.1. Results from Edge Detection Methods

The following figure presents the results obtained by using the edge detection methods that are introduced in section two of the study.

Figure 5 indicates that Sobel edge detectors can detect the pattern edges but has many noises along the horizontal and vertical gradients. The Combined Sobel detects edges with a little noise compared to its individual gradients. Laplacian edge detector is somehow cleaner than Sobel even though its results cannot surpass the results obtained by Canny edge detectors when L1 norm is allowed.

4.2. Machine Learning Thresholding Metrics

Given that, the results in Section 4.1 were not clearly observed that the edges were not clearly detected, then the ML thresholding metrics were employed to determine whether the colored image was suitable for segmentation and further processing. The Machine learning thresholding metrics were applied and accuracy was 45% smaller than the criterion mentioned in [9] and [10]. When the accuracy is small, this means that, there is no background pixels that were detected while F_1 score was approximately equal to 1.41×10^{-5} that recommends the use of other sophisticated thresholding methods. The value of MCC was −0.035 indicating that there is a negative correlation of pixels in an image due to high noise and non-uniform color intensity [9].

4.3. Convolution Neural Networks Results

As introduced in the methodology part, CNN was used to classify and determine the efficiency of the training model used for classification. The efficiency of the model was inversely proportional to the learning rate *i.e.* when the learning rate is small then model efficiency increased even though the computational time was almost similar [11]. We started with a batch 200 increasing to 700 images, the accuracy ranged from 45.6% to 84% while the execution time was 1.5 s for 10 epochs. The following figure shows the predicted templates were wrong as shown in **Figure 6**.

5. Results and Discussion

Even though the edge detection methods were not able to detect the patterns accurately because of low quality and poor intensity of the raw image but CNN was able to classify the images that have right and wrong with faulty patterns.

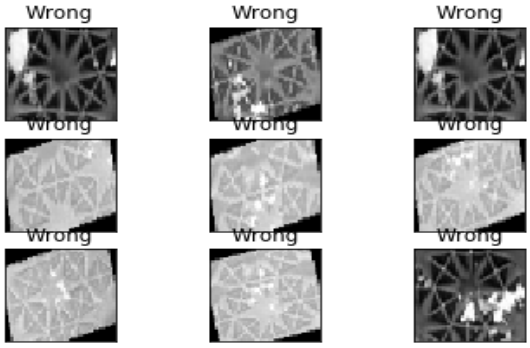

Figure 6. Results from CNN classification model for wrong and right pattern.

The model used revealed the truth that the more increase in the input mages, the more accurately the classification problem is handled. We conclude that, the expected results could be precise and accurate for both edge detection methods and CNN if the quality of the raw image is maintained.

Acknowledgements

This research was supported by the Ministry of Trade, Industry and Energy for its financial support of the project titled "the establishment of advanced marine industry open laboratory and development of realistic convergence content", PKNU LINC+ Project, and SMTECH Innovation Startup Project.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Szeliski, R. (2010) Computer Vision: Algorithms and Applications. Springer, London, UK.

[2] Shin, D., Fowlkes, C.C. and Hoiem, D. (2018) Pixels, Voxels, and Views: A Study of Shape Representations for Single Vie 3D Object Shape Prediction. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3061-3069. arXiv:1804.06032 https://doi.org/10.1109/CVPR.2018.00323

[3] Dutt, V., Chaudhry, V. and Khan, I. (2012) Pattern Recognition: An Overview. *American Journal of Intelligent Systems*, **2**, 23-27.

[4] Wojcik, W., Gromaszek, K. and Junisbekov, M. (2016) Face Recognition: Issues, Methods and Alternative Applications. *Face Recognition-Semisupervised Classification, Subspace Projection and Evaluation Methods*, 7-28. https://doi.org/10.5772/62950

[5] Verma, B. and Blumenstein, M. (2008) Pattern Recognition Technologies and Applications: Recent Advances. *Information Science Reference*, USA. https://doi.org/10.4018/978-1-59904-807-9

[6] Jain, A.K., Duin, R. and Mao, J. (2000) Statistical Pattern Recognition: A Review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22**, 4-37. https://doi.org/10.1109/34.824819

[7] Rosebrock, A. (2019) Historically-Nested Edge Detection with OpenCV and Deep Learning. https://www.pyimagesearch.com/2019/03/04/

[8] Bhargavi, K. and Jyothi, S. (2012) A Survey on Thresholding Based Segmentation Technique in Image Processing. *International Journal of Innovative Research & Development*, **3**, 234-239.

[9] Vemuri, P.V.N. (2019) Image Segmentation with Python. https://kite.com/blog/python/image-segmentation-tutorial/

[10] Boughorbel, S., Jarray, F. and El-Anbari, M. (2017) Optimal Classifier for Imbalanced Data Using Mathews Correlation Coefficient Metric. *PLoS ONE*, **12**, 1-17. https://doi.org/10.1371/journal.pone.0177678

[11] Weng, S. (2019) Automating Breast Cancer Detection with Deep Learning with Deep Learning. https://blog.insightdatascience.com/automating-breast-cancer-detection-with-deep-learning-d8b49da17950

[12] .Bhamare, D. and Suryawanshi, P. (2019) Review on Reliable Pattern Recognition with Machine Learning Techniques. *Fuzzy Information and Engineering*, 1-16. https://doi.org/10.5013/IJSSST.a.20.02.08

[13] Kingma, D.P. and Ba, J. (2015) Adam: A Method for Stochastic Optimization. *Conference Paper at the* 3rd *International Conference for Learning Representations*, San Diego, 2015. arXiv:1412.6980ss