CLASSIFICATION OF LOMBOK SONGKET CLOTH IMAGE USING CONVOLUTION NEURAL NETWORK METHOD (CNN)

Hambali [1]; Mahayadi [2]; Bahtiar Imran [3*)]

Technical Information[1,2], Computer System Engineering[3]
Universitas Teknologi Mataram
https://utmmataram.ac.id/
08mi071@gmail.com[1], lp2mutm@gmail.com[2] , bahtiarimranlombok@gmail.com[3]*
(*) Corresponding Author

Abstract— The diversity of tribes makes Indonesia rich in culture that characterizes it, one of which is traditional cloth. Through a variety of patterns and motifs that exist in traditional fabrics, reflecting the life, customs, and culture that exist in an area. Lombok is one of the areas that produce a typical songket cloth. The famous songket craft centers in Lombok are located in the Pringgasela area, Pringgasela District, Sade Village is in Pujut District, Central Lombok Regency and Sukarara is in Jonggat District, Central Lombok Regency. Each area of the center for songket craftsmen has its characteristics both in terms of the name, motif, and texture. When viewed with the naked eye, the texture of each songket will look the same, to be able to know the differences in the texture of each songket, it is necessary to do a classification using computers or technology. Today's society still does not know much information about the textures of songket cloth. The method used to classify the typical Lombok songket in this study uses the Convolution Neural Network (CNN) method. The results obtained from the use of 64 datasets, with details of 40 types of Sade songket and 24 types of Pringgasela songket, after the dataset is trained it produces 86.36% accuracy, 87% precision, 86% recall, and 86% F1-Score.

Keywords: Histogram Equalization, Convolution Neural Network, Songket Cloth.

Abstract— *Keberagaman suku menjadikan Indonesia kaya akan kebudayaan yang menjadi ciri khas, salah satunya adalah kain tradisional. Melalui ragam corak serta motif yang ada pada kain tradisional, mencerminkan kehidupan, adat istiadat, dan kebudayaan yang ada pada suatu daerah. Lombok merupakan salah satu daerah yang menghasilkan kain songket yang khas. Sentra pengrajin kain songket di Lombok yang terkenal berada di daerah Pringgasela Kecamatan Pringgasela, Desa Sade berada di Kecamatan Pujut Kabupaten Lombok Tengah dan Sukarara berada di Kecamatan Jonggat Kabupaten Lombok Tengah. Setiap daerah sentra pengrajin kain songket mempunyai ciri khas masing-masing baik dari segi nama, motif dan teksturnya, Jika dilihat menggunakan kasat mata, tekstur dari setiap kain songket akan kelihatan sama, untuk dapat mengetahui perbedaan dari tekstur dari setiap kain songket, perlu dilakukan sebuah klasifikasi dengan memanfaatkan komputer atau teknologi. Masyarakat saat ini masih belum banyak mengetahui informasi tentang tekstur-tekstur dari kain songket. Metode yang digunakan untuk melakukan klasifikasi terhadap kain songket khas Lombok dalam penelitian ini menggunakan metode Convolution Neural Network (CNN). Hasil yang diperoleh dari penggunaan sebanyak 64 dataset, dengan rincian songket Sade 40 jenis dan songket Pringgasela 24 jenis, setelah dataset di training menghasilkan akurasi 86.36%, precision sebesar 87%, recall sebesar 86%, dan F1-Score sebesar 86%.*

Kata Kunci*: Histogram Equalization, Convolution Neural Network, Kain Songket.*

INTRODUCTION

Indonesia has a variety of cultures that must be preserved, one of which is batik and songket which have the characteristics of each region. The diversity of tribes makes Indonesia rich in culture that characterizes it, one of which is traditional cloth. Through a variety of patterns and motifs that exist in traditional fabrics, reflecting the life, customs, and culture that exist in an area (Putu Aryasuta Wicaksana, I Made Sudarma, 2019). On October 2, 2009, UNESCO determined that woven fabrics and batik are Indonesian cultural heritages and are one of the icons for the Indonesian nation (Fonda, 2020; Mawan, 2020).

The motifs and textures on woven fabrics in Indonesia vary according to the characteristics of each region (Amalia, 2018). One area that has a characteristic in songket cloth is Lombok. Lombok is one of the islands in the province of West Nusa Tenggara. Lombok is a tourist area that offers cultural diversity, one of which is the typical Lombok songket cloth. Lombok is one of the areas that produce a typical songket cloth. The famous songket craft centers in Lombok are located in the

Pringgasela area, Pringgasela District, Sade Village is in Pujut District, Central Lombok Regency and Sukarara is in Jonggat District, Central Lombok Regency.

Each area of the center for songket craftsmen has its characteristics, both in terms of names, motifs, and textures. When viewed with the naked eye, the texture of each songket will look the same, to be able to know the difference in the texture of each songket, it is necessary to classify by using a computer or technology. Today's society still does not know much information about the textures of songket cloth. This is because there is no good data collection computerized (Amalia, 2018). Classification of an object can be done indirectly by classifying the image of the object based on its features (Setiohardjo & Harjoko, 2014)

Several studies use Convolution Neural Network (CNN) to solve the problem. Study (Fonda, 2020) using Convolution Neural Network (CNN) for classification of Riau batik, Classification using CNN resulted in the bureau and not bureau with an accuracy of 65%. Research (Mawan, 2020) using Convolution Neural Network to calibrate batik achieves an average accuracy of 65% while the CNN model combined with Grayscale achieves an average accuracy of 70%. From research (Putu Aryasuta Wicaksana, I Made Sudarma, 2019) CNN method for pattern recognition of gringsing has woven fabric motifs, and able to complete 100 epoch training in 19.33 hours, and has an accuracy value of 76%, precision of 74.1%, recall 72.3%, and F-measure of 0, 73. While research (Willyanto et al., 2021) using CNN architecture VGG-16 to identify handwritten Japanese hiragana script and get the best scenario with the Adam optimizer and at a learning rate of 0.0001 with an accuracy value of 97.6%, a precision of 97.9%, a recall of 98% and a value of f1 score of 97.5%. On research (Salawazo et al., 2019) implementation of Convolution Neural Network (CNN) on the introduction of CCTV video objects, with the application of the methods carried out can be analyzed based on certain characteristics of the object to be detected or recognized, research (Harani et al., 2019) Convolutional Neural Network (CNN) method for object detection and vehicle number plate character recognition by concluding that the proposed method can detect well.

Currently, there have been many studies conducted related to songket cloth. Researchers use different methods with different results. The methods used also vary from feature extraction using GLCM and Artificial Neural Networks (Imran & Efendi, 2020), SIFT Feature Extraction, Bag of Features and Support Vector Machine to classify Batik (Azhar et al., 2015), Chain Code Algorithm (Yuhandri et al., 2017) and Content-Based Image Retrieval (Minarno et al., 2016), Comparing cropping methods (Yuhandri, 2019), Filter Gabor (Leonardo, 2020), Content-Based Image Retrieval with a combination of Tamura Texture Feature and Gabor Texture Feature (Imran, 2019), Convolution Neural Network with VGG-16 Architecture (Willyanto et al., 2021). From the various methods, methods of presentation, and objects used, researchers get different results.

From several studies that have been carried out, researchers use different objects and architectures. However, this research is to create an architecture and a model for classifying Lombok's typical songket cloth using the Convolution Neural Network (CNN) method using Lombok's typical songket cloth object, the data used in this study was taken directly at the existing songket craft center. in the Lombok area.

MATERIALS AND METHODS

The stages of the research method used in this research are 1) Literature Study 2) Image Acquisition; 3) Pre-processing; 4) Image Improvement with Histogram Equalization; 5) Classification using Convolution Neural Network (CNN); 6) Results and conclusions.

1. Pre-processing

At this stage, the songket image that has been taken is then cropped so that its size becomes smaller, the initial size at the time of data collection was 3456 x 2304 pixels, then changed to 800 x 800 pixels. Image cropping is done to simplify the process of classifying image data.

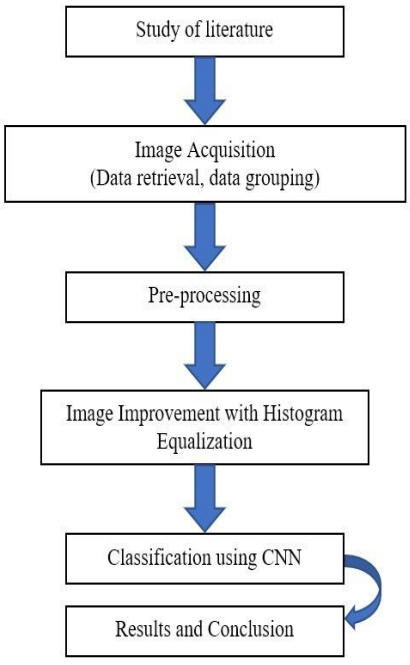

Figure 1. Research Method

2. Image improvement with Histogram Equalization

At this stage, image improvement needs to be done to get better image results. At the time of data collection, things that are not wanted happen such as noise, inconsistent movement, and other disturbances, for that this stage needs to be done. Image improvement used is Histogram Equalization. Figure 3 shows an example of image repair that will be carried out.

3. Classification with CNN

Convolutional Neural Network (CNN) is a development of MLP (Multi-Layer Perceptron) (Amalia, 2018) inspired by human neural networks and designed to process two-dimensional data (Putu Aryasuta Wicaksana, I Made Sudarma, 2019). CNN is included in the Deep Neural Network because CNN has a high network depth and its architecture consists of many layers. At this stage, after the image is pre-processed then it has passed the image repair stage, after that the image data is then classified using CNN. CNN application is built using Python language with Google Collaboratory platform. This application will classify the Songket according to its area. If the application is wrong in classifying, then the data is considered unsuccessful in classification.

RESULTS AND DISCUSSION

1. Data Retrieval

In this study, data retrieval was carried out in 3 areas of the center for songket fabric craftsmen typical of Lombok. The areas used for data collection include Pringgasela District, East Lombok Regency, Sukarara Village, Jonggat District, Central Lombok Regency, and Sade District, Pujut District, Central Lombok Regency. Of the three areas of the center of Lombok's typical songket fabric, all three have different songket motifs and names that make each region's songket fabric has its charm for tourists. The image of the songket cloth is taken using a camera, each songket cloth is taken 10-15 times. This study uses songket image data for data processing.

2. Image Acquisition

Image acquisition was carried out to classify songket image data that had been taken from various areas of songket fabric craftsmen. At the time of data collection, the data is still random and the officers who carry out data collection are also different, in this case, it is necessary to do image acquisition. The original data of the songket image is 3456 x 2304 pixels, then cropped to 500x500 pixels. Grouping is done to facilitate the training process and the testing process.

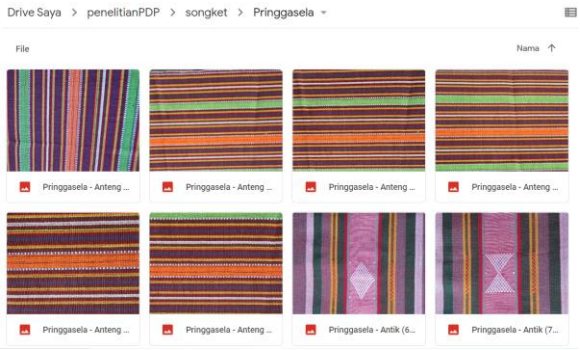

Figure 2. Connect Google Colab and Google Drive

3. Dataset Usage

The dataset was obtained from songket craftsmen on the island of Lombok. The datasets are grouped into two polders, namely songket polder 1 and songket 2. The dataset used in this study is 64 data, with details songket 1 = 40 and songket 2 = 24.

4. Implementation CNN

CNN is included in the type of Deep Neural Network because of its deep network level and is widely implemented in image data (Mawan, 2020).

a. Dataset collection

The dataset is grouped into 2 polders and named Songket 1 and Songket 2. In this study, Google Colab was used as a platform to create Python-based applications for the classification of Lombok songket fabrics. while for data storage using the Google Drive application. The dataset is uploaded in zip form on Google Drive and we start to create applications with Google Colab by connecting Colab with Google Drive, picture 3 is the command to connect Google Colab with Google Drive where the dataset that we will use is stored.

```
# CONNECT GOOGLE DRIVE
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

Figure 3. Connecting Google Colab and Google Drive

```
!unzip "/content/drive/My Drive/penelitianPDP/songket.zip" -d "/content/drive/My Drive/penelitianPDP/songket/"

Archive:  /content/drive/My Drive/penelitianPDP/songket.zip
   creating: /content/drive/My Drive/penelitianPDP/songket/Sade/
  inflating: /content/drive/My Drive/penelitianPDP/songket/Sade/Rarang (2).jpg
  inflating: /content/drive/My Drive/penelitianPDP/songket/Sade/Rarang (3).jpg
  inflating: /content/drive/My Drive/penelitianPDP/songket/Sade/Rarang (4).jpg
  inflating: /content/drive/My Drive/penelitianPDP/songket/Sade/Rarang (5).jpg
  inflating: /content/drive/My Drive/penelitianPDP/songket/Sade/Rarang (6).jpg
  inflating: /content/drive/My Drive/penelitianPDP/songket/Sade/Sade - Keker (1).jpg
  inflating: /content/drive/My Drive/penelitianPDP/songket/Sade/Sade - Keker (2).jpg
  inflating: /content/drive/My Drive/penelitianPDP/songket/Sade/Sade - Keker (3).jpg
  inflating: /content/drive/My Drive/penelitianPDP/songket/Sade/Sade - Keker (4).jpg
  inflating: /content/drive/My Drive/penelitianPDP/songket/Sade/Sade - Keker (5).jpg
  inflating: /content/drive/My Drive/penelitianPDP/songket/Sade/Sade - Keker (6).jpg
  inflating: /content/drive/My Drive/penelitianPDP/songket/Sade/Sade - Pengginang (1).jpg
  inflating: /content/drive/My Drive/penelitianPDP/songket/Sade/Sade - Pengginang (2).jpg
  inflating: /content/drive/My Drive/penelitianPDP/songket/Sade/Sade - Pengginang (3).jpg
  inflating: /content/drive/My Drive/penelitianPDP/songket/Sade/Sade - Pengginang (4).jpg
  inflating: /content/drive/My Drive/penelitianPDP/songket/Sade/Sade - Pengginang (5).jpg
  inflating: /content/drive/My Drive/penelitianPDP/songket/Sade/Sade - Rarang.jpg
  inflating: /content/drive/My Drive/penelitianPDP/songket/Sade/Sade - Subhanale .jpg
  inflating: /content/drive/My Drive/penelitianPDP/songket/Sade/Subhanale (2).jpg
  inflating: /content/drive/My Drive/penelitianPDP/songket/Sade/Subhanale (3).jpg
```

Figure 4. Creating and Extracting Data

Figure 4 is a code snippet of the dataset that we have connected from Google Drive to Google Colab. From the picture, it can be seen that the dataset that we

have saved into Google Drive can be read and extracted so that we can carry out further processes.

b. Reading Dataset

A dataset that has been uploaded to Google Drive containing image data from songket cloth and then read using the Open CV tool available in Python. The file that has been read is then stored in the variable X for the songket image and y for the label {"Sade", Pringgasela"l}. Figure 5 is a code snippet to display the number of datasets used.

```
import os, re, glob, cv2, numpy as np

path = os.path.abspath('penelitian.ipynb') #absolute path of program
path = re.sub('[a-zA-Z\s._]+$', '', path) #remove unintended file
dirs = os.listdir('/content/drive/My Drive/penelitianPDP/songket/') #list directory in Land Use Images folder
label = 0
im_arr = []
lb_arr = []
X = []
y = []
for i in dirs: #loop all directory
    count = 0
    for pic in glob.glob('/content/drive/My Drive/penelitianPDP/songket/'+i+'/*'):
        im = cv2.imread(pic) #open image
        #im = cv2.resize(im,(70,70))
        im = np.array(im) #change into array
        count = count + 1
        X.append(im)
        y.append(label)
        if(count <= 10): #SAmple
            im_arr.append((str(i):im))
    print("Songket "+str(i)+" : "+str(count))
    label = label + 1
    lb_arr.append(i)
X = np.array(X)
y = np.array(y);
```

Figure 5. Displays the number of datasets used.

```
Songket Sade : 40
Songket Pringgasela : 24
```

Figure 6. Data on the number of datasets used

Figure 6 shows the amount of data that will be used in this study, for songket sade as many as 40 and songket pringgasela 24. Songket sade and songket sukerare are combined into one file because they come from the same area, namely the Central Lombok district.

a. Show Data

At this stage, it displays the dataset used whether the code used has been successful or not. Figure 7 is a code snippet to display the dataset used.

```
import matplotlib.pyplot as plt

fig, axs = plt.subplots(3, 7, figsize=(25, 10))
cnt = 0
row = 0
col = 0
for i in im_arr:
    for key, value in i.items():
        if(cnt==7):
            row = row + 1
            col = 0
            cnt = 0
        axs[row, col].imshow(value)
        axs[row, col].set_title(key)
        cnt = cnt + 1
        col = col + 1
plt.show()
```

Figure 7. Displaying data on Google Colab

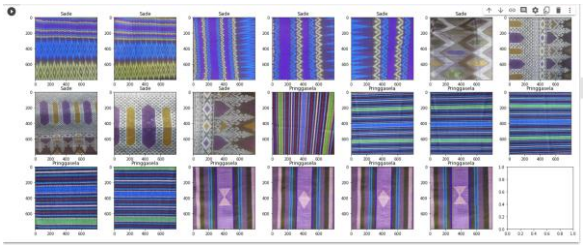

Figure 8. Display Dataset

Figure 8 shows the display of the dataset used with a size of 500x500 pixels, the dataset needs to be resized to facilitate the image classification process.

```
plt.imshow(X[0])
```

```
<matplotlib.image.AxesImage at 0x7fce0c1f3710>
```

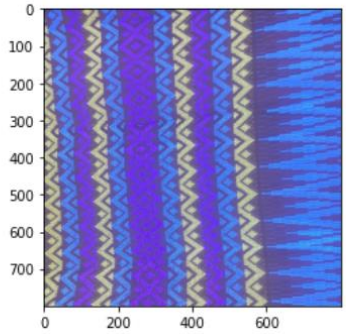

Figure 9. Displays per dataset

Figure 9 is an example of displaying data per index, in the example shown is an image that has an index to [0], in this case, we use the plt.imshow command to call one of the data using the index.

```
X_train
```

```
[[[0.15294118, 0.10588235, 0.3019608 ],
  [0.1254902 , 0.08627451, 0.2509804 ],
  [0.14509805, 0.11372549, 0.23137255],
  ...,
  [0.40784314, 0.34509805, 0.4392157 ],
  [0.27450982, 0.18431373, 0.33333334],
  [0.25882354, 0.16078432, 0.33333334]]],


[[[0.42352942, 0.34117648, 0.23529412],
  [0.45490196, 0.35686275, 0.22352941],
  [0.5254902 , 0.39607844, 0.21960784],
  ...,
  [0.16470589, 0.19215687, 0.627451  ],
  [0.17254902, 0.16862746, 0.62352943],
  [0.2627451 , 0.2509804 , 0.7058824 ]],

 [[0.41568628, 0.32156864, 0.24313726],
  [0.40392157, 0.29803923, 0.19215687],
  [0.5019608 , 0.3764706 , 0.20784314],
  ...,
  [0.16862746, 0.19607843, 0.6313726 ],
  [0.16862746, 0.17254902, 0.61960787],
  [0.25490198, 0.2509804 , 0.7058824 ]],

 [[0.4       , 0.3019608 , 0.2627451 ],
  [0.39215687, 0.28627452, 0.20784314],
  [0.4392157 , 0.3137255 , 0.17254902],
  ...,
  [0.1764706 , 0.20392157, 0.6392157 ],
  [0.16078432, 0.18431373, 0.627451  ],
  [0.20784314, 0.23137255, 0.6745098 ]],
```

Figure 10. X_train display

Figure 10 is a snippet of the results after we saw the extraction of the data used as training data, in this case, we use the X_train command to display the data. Meanwhile, to see the extraction data for label Y can be seen in Figure 11, in this case, we use the y_train command to display the results.

```
y_train

array([[1., 0.],
       [0., 1.],
       [1., 0.],
       [0., 1.],
       [0., 1.],
       [0., 1.],
       [1., 0.],
       [1., 0.],
       [0., 1.],
       [0., 1.],
       [1., 0.],
       [1., 0.],
       [1., 0.],
       [1., 0.],
       [0., 1.],
       [1., 0.],
       [1., 0.],
       [0., 1.],
       [0., 1.],
       [1., 0.],
       [1., 0.],
       [0., 1.],
       [0., 1.],
```

Figure 11. y_train display

```
X_test

array([[[[0.30980393, 0.09411765, 0.08235294],
         [0.32941177, 0.11372549, 0.10196079],
         [0.3254902 , 0.10980392, 0.09803922],
         ...,
         [0.49803922, 0.21960784, 0.23137255],
         [0.43137255, 0.14901961, 0.17254902],
         [0.34117648, 0.06666667, 0.09019608]],

        [[0.32156864, 0.10588235, 0.09411765],
         [0.32941177, 0.11372549, 0.10196079],
         [0.31764707, 0.10196079, 0.09019608],
         ...,
         [0.35686275, 0.06666667, 0.08235294],
         [0.39215687, 0.10980392, 0.13333334],
         [0.36862746, 0.09411765, 0.11764706]],

        [[0.30980393, 0.09411765, 0.08235294],
         [0.34117648, 0.1254902 , 0.11372549],
         [0.30588236, 0.09019608, 0.07843138],
         ...,
         [0.43529412, 0.14901961, 0.15686275],
         [0.41960785, 0.14117648, 0.15294118],
         [0.36078432, 0.08235294, 0.09411765]],
```

Figure 12. X_test display

Figure 12 is the extraction display from the dataset used as test data, in this case using the X_test command to display the data, while Figure 13 is the extraction result from the Y label, in this case using the y_test command to display the data.

```
y_test

array([[0., 1.],
       [0., 1.],
       [1., 0.],
       [0., 1.],
       [1., 0.],
       [1., 0.],
       [1., 0.],
       [1., 0.],
       [1., 0.],
       [0., 1.],
       [0., 1.],
       [1., 0.],
       [0., 1.],
       [1., 0.],
       [1., 0.],
       [0., 1.],
       [1., 0.],
       [1., 0.],
       [0., 1.],
       [1., 0.],
       [1., 0.],
       [1., 0.]], dtype=float32)
```

Figure 13. y_test display

a. Preprocessing

```
# PREPROCESSING
from sklearn.model_selection import train_test_split
from tensorflow.keras.utils import to_categorical
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=42)

X_train = X_train.astype('float32') #set x_train data type as float32
X_test = X_test.astype('float32') #set x_test data type as float32
X_train /= 255 #change x_train value between 0 - 1
X_test /= 255 #change x_test value between 0 - 1
y_train = to_categorical(y_train) #change label to binary / categorical: [1 0 0 0] = 0, [0 1 0 0] = 1, so on
y_test = to_categorical(y_test) #change label to binary / categorical
```

Figure 14. Preprocessing

Figure 14 is the CNN Preprocessing of training data and test data, the library used is train_test_split, and train_test_split is the default from the sklearn library, then the X_test variable and X-train variable we make the data type into Float.

b. Creating CNN Architecture

```
# ARSITEKTUR
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D, Dropout, Flatten, Dense
model = Sequential() #model = sequential
model.add(Conv2D(32, kernel_size=(3, 3),activation='relu',input_shape=(70,70,3))) #layer convolutional 2D
model.add(MaxPooling2D(pool_size=(2,2))) #max pooling with stride (2,2)
model.add(Conv2D(32, (3, 3), activation='relu')) #layer convolutional 2D
model.add(MaxPooling2D(pool_size=(2,2))) #max pooling with stride (2,2)
model.add(Dropout(0.25)) #delete neuron randomly while training and remain 75%
model.add(Flatten()) #make layer flatten
model.add(Dense(128, activation='relu')) #fully connected layer
model.add(Dropout(0.5)) #delete neuron randomly and remain 50%
model.add(Dense(2, activation='softmax')) #softmax works
```

Figure 15. CNN Architecture

Figure 15 is a Convolution Neural Network (CNN) based model, in this case, we use a library of hardware and a sequential model. In creating this model we need conv2D, maxpooling2D, dropout, flatten, and dense layers. In the code snippet above, we create a variable with the model name, and then we call each layer above.

c. Compile

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 68, 68, 32)        896
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 34, 34, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 32, 32, 32)        9248
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 16, 16, 32)        0
_________________________________________________________________
dropout (Dropout)            (None, 16, 16, 32)        0
_________________________________________________________________
flatten (Flatten)            (None, 8192)              0
_________________________________________________________________
dense (Dense)                (None, 128)               1048704
_________________________________________________________________
dropout_1 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_1 (Dense)              (None, 2)                 258
=================================================================
Total params: 1,059,106
Trainable params: 1,059,106
Non-trainable params: 0
```

Figure 16. CNN Compile Results

```
# COMPILE
from tensorflow.keras.optimizers import SGD
epochs = 25
lrate = 0.01
decay = lrate/epochs
sgd = SGD(lr=lrate, momentum=0.9, decay=decay, nesterov=False)
model.compile(loss='categorical_crossentropy', optimizer=sgd, metrics=['accuracy'])
print(model.summary())
```

Figure 17. CNN Compile Results

Figure 17 is a code snippet from the compile, in this case, we use the library from hard, then with lrrate with static. In this case, we also add metrics by adding measurements in the form of accuracy. Figure 16 is the result after we compile it.

d. Training

The training was conducted to provide knowledge of the Convolution Neural Network (CNN) architecture that was built. Figure 16 is the code used to perform training data, which includes training data (x_train) and test data (y_train) using validation data between the labels x_test and y_test. Then the epoch used in this study was 25, from the results of the training carried out, the accuracy obtained was 86.63%. The results of the training can be seen in Figure 19.

```
#TRAINING
model.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=epochs, batch_size=32)
scores = model.evaluate(X_test, y_test, verbose=0)
print("Accuracy: %.2f%%" % (scores[1]*100))
```

Figure 18. Training Code

```
Epoch 1/25
2/2 [==============================] - 0s 149ms/step - loss: 0.2104 - accuracy: 0.9524 - val_loss: 0.2474 - val_accuracy: 0.909
1
Epoch 2/25
2/2 [==============================] - 0s 106ms/step - loss: 0.0942 - accuracy: 0.9762 - val_loss: 0.2354 - val_accuracy: 0.909
1
Epoch 3/25
2/2 [==============================] - 0s 113ms/step - loss: 0.0749 - accuracy: 0.9762 - val_loss: 0.2742 - val_accuracy: 0.863
6
Epoch 4/25
2/2 [==============================] - 0s 105ms/step - loss: 0.1047 - accuracy: 0.9762 - val_loss: 0.2172 - val_accuracy: 0.909
1
Epoch 5/25
2/2 [==============================] - 0s 108ms/step - loss: 0.1011 - accuracy: 0.9762 - val_loss: 0.2054 - val_accuracy: 0.909
1
Epoch 6/25
2/2 [==============================] - 0s 76ms/step - loss: 0.0950 - accuracy: 0.9762 - val_loss: 0.4201 - val_accuracy: 0.8182
Epoch 7/25
2/2 [==============================] - 0s 77ms/step - loss: 0.0983 - accuracy: 0.9524 - val_loss: 0.2188 - val_accuracy: 0.9091
Epoch 8/25
2/2 [==============================] - 0s 80ms/step - loss: 0.0801 - accuracy: 1.0000 - val_loss: 0.2195 - val_accuracy: 0.8636
Epoch 9/25
2/2 [==============================] - 0s 82ms/step - loss: 0.0642 - accuracy: 0.9762 - val_loss: 0.7771 - val_accuracy: 0.8182
Epoch 10/25
2/2 [==============================] - 0s 76ms/step - loss: 0.1950 - accuracy: 0.8810 - val_loss: 0.4042 - val_accuracy: 0.8182
Epoch 11/25
2/2 [==============================] - 0s 74ms/step - loss: 0.1930 - accuracy: 0.9048 - val_loss: 0.3127 - val_accuracy: 0.8636
Epoch 13/25
2/2 [==============================] - 0s 74ms/step - loss: 0.1500 - accuracy: 0.9524 - val_loss: 0.6451 - val_accuracy: 0.5909
Epoch 14/25
2/2 [==============================] - 0s 78ms/step - loss: 0.4923 - accuracy: 0.7143 - val_loss: 0.4195 - val_accuracy: 0.8182
Epoch 15/25
2/2 [==============================] - 0s 84ms/step - loss: 0.3056 - accuracy: 0.9524 - val_loss: 0.4303 - val_accuracy: 0.8636
Epoch 16/25
2/2 [==============================] - 0s 80ms/step - loss: 0.2354 - accuracy: 0.9286 - val_loss: 0.6481 - val_accuracy: 0.7727
Epoch 17/25
2/2 [==============================] - 0s 77ms/step - loss: 0.3911 - accuracy: 0.8810 - val_loss: 0.5526 - val_accuracy: 0.8636
Epoch 18/25
2/2 [==============================] - 0s 74ms/step - loss: 0.3046 - accuracy: 0.8810 - val_loss: 0.3520 - val_accuracy: 0.8636
Epoch 19/25
2/2 [==============================] - 0s 73ms/step - loss: 0.1617 - accuracy: 0.9286 - val_loss: 0.3933 - val_accuracy: 0.7727
Epoch 20/25
2/2 [==============================] - 0s 78ms/step - loss: 0.2297 - accuracy: 0.9286 - val_loss: 0.3322 - val_accuracy: 0.8636
Epoch 21/25
2/2 [==============================] - 0s 77ms/step - loss: 0.1440 - accuracy: 0.9762 - val_loss: 0.4475 - val_accuracy: 0.8636
Epoch 22/25
2/2 [==============================] - 0s 75ms/step - loss: 0.2171 - accuracy: 0.9048 - val_loss: 0.3613 - val_accuracy: 0.9091
Epoch 23/25
2/2 [==============================] - 0s 75ms/step - loss: 0.1340 - accuracy: 0.9286 - val_loss: 0.2520 - val_accuracy: 0.9091
Epoch 24/25
2/2 [==============================] - 0s 80ms/step - loss: 0.1441 - accuracy: 1.0000 - val_loss: 0.2440 - val_accuracy: 0.9091
Epoch 25/25
2/2 [==============================] - 0s 79ms/step - loss: 0.0954 - accuracy: 1.0000 - val_loss: 0.3465 - val_accuracy: 0.8636
Accuracy: 86.36%
```

Figure 19. Training results

e. Classification Results

After going through the stages of architecture, pre-processing, compiling, and training, then classification is carried out on the data that is used as training data. From the results of the classification carried out for precision and recall on the data used, the results obtained are precision and recall for Songket Pringgasela = 0.89 and recall = 0.80, while the results of precision and recall on songket sade are precision = 0.85 and Recall = 0.92. The results of this classification get an accuracy of 0.86. The classification results can be seen in Figure 20.

	precision	recall	f1-score	support
Pringgasela	0.89	0.80	0.84	10
Sade	0.85	0.92	0.88	12
accuracy			0.86	22
macro avg	0.87	0.86	0.86	22
weighted avg	0.87	0.86	0.86	22

Figure 20. Classification results

CONCLUSION

By using the Convolution Neural Network method for the classification of Lombok songket, it provides information that there is a difference in texture between the songket cloth in Sade Village and the songket cloth in Pringgasela. From the results of the classification carried out, the architecture of the proposed CNN can provide a good classification as evidenced by the results of research and testing on the CNN method. for classification, it can be concluded that the use of the convolutional neural network method has an Accuracy value of 86%, Precision of 87%, Recall of 86%, and F1-Score of 86%. In total there are 64 datasets, with details of 40 types of Sade songket and 24 types of Pringgasela songket. The conclusion of this study can be said that the classification using the CNN method on Lombok's typical songket cloth is quite good.

REFERENCES

Amalia, I. (2018). Ekstraksi Fitur Citra Songket Berdasarkan Tekstur Menggunakan Metode Gray Level Co-occurrence Matrix (GLCM). *Jurnal Infomedia*, *3*(2), 64–68. https://doi.org/10.30811/jim.v3i2.715

Azhar, R., Tuwohingide, D., Kamudi, D., Sarimuddin, & Suciati, N. (2015). Batik Image Classification Using SIFT Feature Extraction, Bag of Features, and Support Vector Machine. *Procedia Computer Science*, *72*, 24–30. https://doi.org/10.1016/j.procs.2015.12.101

Fonda, H. (2020). Klasifikasi Batik Riau Dengan Menggunakan Convolutional Neural Networks (Cnn). *Jurnal Ilmu Komputer*, *9*(1), 7–10. https://doi.org/10.33060/jik/2020/vol9.iss1.144

Harani, N. H., Prianto, C., & Hasanah, M. (2019). Deteksi Objek Dan Pengenalan Karakter Plat Nomor Kendaraan Indonesia Menggunakan Metode Convolutional Neural Network (CNN) Berbasis Python. *Jurnal Teknik Informatika*, *11*(3), 47–53.

Imran, B. (2019). Content-Based Image Retrieval Based on Texture and Color Combinations Using Tamura Texture Features and Gabor Texture Methods. *American Journal of Neural Networks and Applications*, *5*(1), 23. https://doi.org/10.11648/j.ajnna.20190501.14

Imran, B., & Efendi, M. M. (2020). The Implementation Of Extraction Feature Using Glcm And Back-Propagation Artificial Neural Network To Clasify Lombok Songket Woven Cloth. *Jurnal Techno Nusa Mandiri*, *17*(2), 131–136.

Leonardo, L. (2020). Penerapan Metode Filter Gabor Untuk Analisis Fitur Tekstur Citra Pada Kain Songket. *Jurnal Sistem Komputer Dan Informatika (JSON)*, *1*(2), 120. https://doi.org/10.30865/json.v1i2.1942

Mawan, R. (2020). Klasifikasi motif batik menggunakan Convolutional Neural Network. *Jnanaloka*, 45–50. https://doi.org/10.36802/jnanaloka.2020.v1-no1-2

Minarno, A. E., Kurniawardhani, A., & Bimantoro, F. (2016). Image retrieval based on multi-structure co-occurrence descriptor. *Telkomnika (Telecommunication Computing Electronics and Control)*, *14*(3), 1175–1182. https://doi.org/10.12928/TELKOMNIKA.v14i3.3292

Putu Aryasuta Wicaksana, I Made Sudarma, D. C. K. (2019). Pengenalan Pola Motif Kain Tenun Gringsing Menggunakan Metode Convolutional Neural Network Dengan Model Arsitektur. *Jurnal SPEKTRUM*, *6*(3), 159–168.

Salawazo, V. M. P., Gea, D. P. J., Gea, R. F., & Azmi, F. (2019). Implementasi Metode Convolutional Neural Network (CNN) Pada Peneganalan Objek Video Cctv. *Jurnal Mantik Penusa*, *3*(1), 74–79.

Setiohardjo, N. M., & Harjoko, A. (2014). Analisis Tekstur untuk Klasifikasi Motif Kain (Studi Kasus Kain Tenun Nusa Tenggara Timur). *IJCCS (Indonesian Journal of Computing and Cybernetics Systems)*, *10*(1), 177. https://doi.org/10.22146/ijccs.6545

Willyanto, A., Alamsyah, D., & Irsyad, H. (2021). *Identifikasi Tulisan Tangan Aksara Jepang Hiragana Menggunakan Metode CNN Arsitektur VGG-16*. *2*(1), 1–11.

Yuhandri, Madenda, S., Wibowo, E. P., & Karmilasari, karmila@staff gunadarma ac id. (2017). Object feature extraction of songket image using chain code algorithm. *International Journal on Advanced Science, Engineering and Information Technology*, *7*(1), 235–241. https://doi.org/10.18517/ijaseit.7.1.1479

Yuhandri, Y. (2019). Perbandingan Metode Cropping Pada Sebuah Citra Untuk Pengambilan Motif Tertentu Pada Kain Songket Sumatera Barat. *Komtekinfo*, *6*(1), 95–105. https://doi.org/10.35134/komtekinfo.v6i1.273