Homework 2 Machine Learning
Train a convolutional neural network that will be able to classify weather conditions in an urban scenario with CCTV views

Vannoli Marco matr.1860363*

Master's degree in artificial intelligence and robotics Department of computer engineering, automatic e management "Antonio Ruberti" University of Sapienza Roma

(Dated: December 13, 2019)

1. INTRODUCTION

The work carried out consists of classifying a set of images in the four climate-related categories: haze, rainy, snowy and sunny. For this purpose two types of models have been performed:

- a network based on a traditional convolution neural network (CNN),
- a neural network based on transfer learning technic (VGG16).

During the paper, the choices that led to a choice between the two models will be evaluated both from an accuracy point of view and from a precision point of view of the classification of the images. This evaluation, as we shall see, will focus on the provision of classes for a set of images produced by myself to evaluate both models even with images not deriving from the datasets provided. The datasets are provided by the SMART-I company and concern urban scenarios taken from the CCTV (The China Central Television) which is the largest television network in mainland China. These scenarios take scenarios with four weather conditions, i.e (see figure 1):

- HAZE,
- RAINY,
- SNOWY,
- SUNNY.

FIG. 1. some random images

It will be noted, that the predictions concerning the classification of test images are mainly influenced by everything that makes up the individual images given to the model, for example by the type of subjects that surround the image (machines, people, things), image quality (the size of the image, the quality and the truth of the colors) and so on. The images belonging to the blind dataset which have been produced by my self have been analyzed in order to consider many aspects that characterized the images with which the model has been trained. the results obtained from a classification problem of this type can be exploited in the tech field to optimize processes that a short time ago were unthinkable or too expensive to be put into action.

2. DATASET

The datasets provided come from CCTV divided by the four classes mentioned above. For the training phase, the "MWI-Dataset-1.1_2000" dataset was used, which is composed of 2000 images while the "MWI-Dataset-

* vannolimarco@hotmail.it

1.1.1_400" dataset was used for the test data set, containing 400 images. This last dataset was not the only data set used as we will see, the dataset supplied by the SMART-I company was also taken into consideration ("TestSet_Weahter/Weahter_Dataset", but containing only images belonging to three classes: RAINY, SNOWY and SUNNY (no HAZE). Moreover, as far as the blind test dataset is concerned, a series of images taken from myself consisting of 40 images was collected (10 for each class-homogeneous dataset)

3. PREPROCESSING

all images were processed to channel them into the model for the training phase. The preprocessing of each image of the dataset chosen for the training of the model touched the following phases:

- rescale = the resizing of a digital image (1/255)
- zoom range = the range for random zoom
- rotation range = the range for rotation image
- width shift range = random shifting of training data on width
- height shift range = random shifting of training data on height
- horizontal flip = Randomly flip inputs horizontally.
- vertical flip = Randomly flip inputs vertically.

The first step of preprocessing is the rescale of image because it aims to transform every pixel value from range [0,255] to [0,1] to manage all images in the same manner, allowing the use of one typical learning rate. the second step is the setting of other parameters that help us to obtain good results during training, for example, the zoom range which allow to zooming with some range inside the image, the rotation range which indicates the range from 0 to 360 with which rotate the image, the width/height shift range which moving all pixels of the image in one direction (horizontally or vertically) while keeping the image dimensions the same and the horizontal/vertical flip which reverse the rows or columns of pixels in the case of a vertical or horizontal flip respectively. In order to define image data preparation and augmentation, the "ImageDataGenerator" class of Keras has been used and the parameters above have been set in this way:

Values of parameters for image augmentation	
Parameters	Values
rescale	1. / 255
zoom_range	0.1
rotation_range	10
width_shift_range	0.1
height_shift_range	0.1
horizontal_flip	True
vertical_flip	False

TABLE I. values chosen from ImageDataGenerator class of keras

Once the parameters of images were set, the images have been taken from the datasets provided and elaborate through a method of ImageDataGenerator class: flow_from_directory(). Thi last method allow to read images from a big numpy array and folders containing images. Its parameters that were set are the following:

- directory= the path where your 'n'
- classes of folders are present
- target_size = (118, 224)
- color_mode = "rgb"
- batch_size = 64
- class_mode = "categorical"
- shuffle = False

To aim to have the same number of sets for both training and test a batch size has been set with value equals to 64. This last parameter is the number that divides your total number of images in your test set or training set, so the model will do the training among 64 images between set training and set testing one at time. Each image from directory has been elaborated in such a way to have an input equals to (118,224,3) and to have as output a 2D one-hot encoded labels (set "categorical" in class_mode). Thus the labels have been encoded in this way:

labels of multi-class of weather		
Parameters	index	one-hot encoding
HAZE	0	1 0 0 0
RAINY	1	0 1 0 0
SNOWY	2	0 0 1 0
SUNNY	3	0 0 0 1

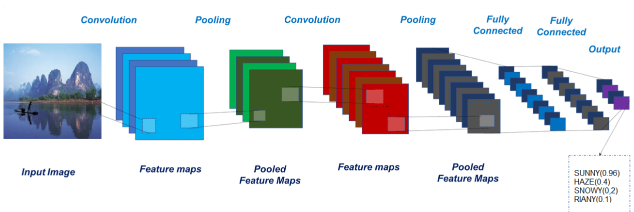
TABLE II. representation of labels

The one-hot encoding has been applied as a representation of labels because offer a more nuanced set of predictions than a single label and provides the representation of categorical data to be more expressive. For the training, the datasets "MWI-Dataset-1.1_2000" and "MWI-Dataset-1.1.1_400", the first as training set and the second as test set (validation) were taken into consideration. The first dataset contains a 2000 images while the second 400 maintaining a homogeneity in the various classes. In order to have a good evaluation and a more confront about dataset, the dataset provided by SMART-I has been used as a set of test.

4. MODELS

The models that have been trained respect a convolution neural network architecture. The convolution neural network, as well called with acronymous CNN, is a class of deep neural networks applied to analyzing visual imagery. It consists of an input and an output layer that perform a convolution operation (see figure 2). The convolution operation which is performed by convolution layer involves moving the kernel from left to right over the input image with a certain stride value until it has analyzed the entire image. In the case of images based on multiple channels (RGB channels), the kernel has the same depth of the input image and the nal result is the sum of the results obtained with the individual kernels (i.e. 3) giving us a squashed one-depth channel convoluted Feature Output. Thus each input image passes into a series of convolution layers with filters or kernels, Pooling, fully connected layers (flatters) and apply "Softmax" function to classify an object with probabilistic values between 0 and 1. The layer as Pooling and Flatter aim to reduce the number of parameters when the images are too large and to flat the matrix provided from convolution layers into a vector.

FIG. 2. The convolution neural network architecture



The models that have been performed are two:

- Convolution neural network.
- Convolution neural network with transfer learning technic through VGG16 pre-training model using as weights the research project 'ImageNet'.

These two different architectures have been trained and evaluated in order to classify in the best way the set of test image produced by my self. Thus, one of these models has been choose at the end of work. In the next subsections, they will be analyzed in deep.

A. Convolution Neural Networks

The structure chosen for the convolution neural network is composed of two convolution layer each of them followed by one "MaxPooling" layer and one dropout layer. After these two convolution layer blocks a flatten layer is performed in order to pass from a matrix to a single vector. The last part is composed of three dense layers with two dropout layer which aim to regularize and disable non-useful neurons. The first two dense layers have as activation function the 'relu' function with 128 and 64 neurons while the last dense layer has a 'softmax' activation function in order to obtain a probability distribution of a list of potential outcomes. This last layer has 4 neurons because have four class as labels (see figure 3).

FIG. 3. The convolution neural network architecture used

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 116, 222, 16)	448
max_pooling2d_1 (MaxPooling2	(None, 38, 74, 16)	0
dropout_1 (Dropout)	(None, 38, 74, 16)	0
conv2d_2 (Conv2D)	(None, 36, 72, 64)	9280
max_pooling2d_2 (MaxPooling2	(None, 12, 24, 64)	0
dropout_2 (Dropout)	(None, 12, 24, 64)	0
flatten_1 (Flatten)	(None, 18432)	0
dense_1 (Dense)	(None, 128)	2359424
dropout_3 (Dropout)	(None, 128)	0
dense_2 (Dense)	(None, 64)	8256
dropout_4 (Dropout)	(None, 64)	0
dense_3 (Dense)	(None, 4)	260

As the figure 3 shows, a 'MaxPooling' layer is used as Pooling layer because it works better than other kind of pooling layer: the results are downsampled or pooled feature maps that highlight the most present feature in the patch, not the average presence of the feature in the case of average pooling. In order to train this model we have to define three entity:

- **batch size**:the number of samples in each mini-batch. For the work is used a batch size with a value of 64.
- **steps per epoch**: the number of batch iterations before a training epoch is considered finished. It is used when there is a huge set of data. In a few words, it is equal to the ratio between the number of samples about the training set and the value of batch size. In this case, the training set is composed of 2000 images, so the steps per epochs is equals to 31 (2000/64).
- **validation steps**: steps per epochs but for the validation data set instead of the training data. Thus, it is equal to the ratio between the number of samples used for the test set and the value of batch size.

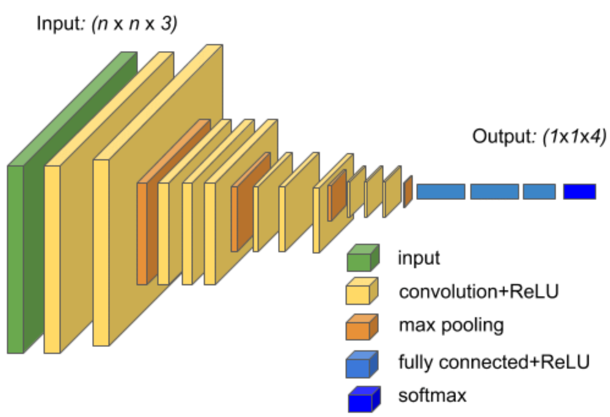
In this case, the test set is composed of 400 images, so the steps per epochs is equal to 7 (400/64).

These three entities characterize the process of training for both models which have been performed for this work. The dropout layer has been used in order to avoid a co-dependency among neurons of the network which can lead an over-fitting. Moreover, the size of kernels chosen is a matrix of (3,3) as the size of the max pool used is also.[4][1] The training was performed with 100 epochs.

B. Convolution Neural Networks with Transfer Learning

The alternative model was created with the transfer learning process. This last process aims to exploit the acquired knowledge to solve a problem and apply it to another problem [2][1]. In order to make an implementation of the transfer learning process a pre-training network was used, so the 'VGG16' has been chosen. The 'VGG16' is a convolutional neural network model (see figure 4). It get 92.7% top-5 test accuracy in ImageNet. Imagenet is a dataset composed of over 14 million images belonging to 1000 classes.

FIG. 4. The VGG16 structure



The features extrapolated from this pre-trained network have been incorporated as input to a further network composed of two dense layers followed by two dropout layers and two batch normalization layers that apply to transformation that maintains the mean activation close to 0 to 1. The optimizer that has been used is 'Adam' because it can update network weights iterative based on training data.

5. EVALUATION MODELS

The model that had the best result about accuracy was the model trained using the transfer learning. As the bottom table show, the convolution neural network trained without transfer learning obtain good accuracy among training and validation but a little lower rather those obtained by network performed with transfer learning.

accuracy obtained by models trained			
Models	acc. train	acc. test	testset
ConvNet	**91%**	**90%**	MWI-Dataset-1.1.1_400
TranferNet (VGG16)	**99%**	**98.5%**	MWI-Dataset-1.1.1_400
ConvNet	**88%**	45%	TestSet_Weather
TranferNet (VGG16)	**98%**	50%	TestSet_Weather

TABLE III. values of accuracy obtained by models

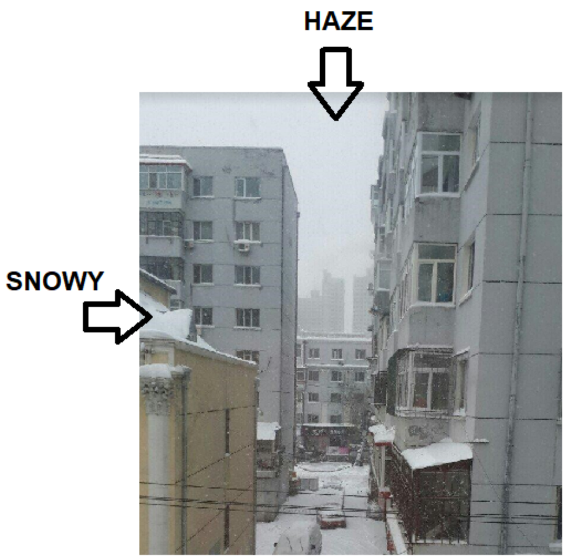
FFrom the accuracy obtained by using the homogeneous MWI-Dataset-1.1.1 _400 dataset as a test set, both models can be seen to be a good learning model, highlighting an excellent network behavior developed through transfer learning. The latter fact leads us to argue that the network trailed with a pre-trained net like the VGG16 classifies the images best thanks to the weights get with ImageNet which is a dataset of over 15 million labeled high-resolution images belonging to roughly 22,000 categories. The figures 7-8 on page 6 show all trends on accuracy and loss function of both models during training, the first respect to dataset MWI with 400 images and the second respect to the dataset TestSet_Weather. The values about accuracy obtained as test set the dataset provided by SMART-I company show an overfitting trend. This latter behavior is due to a larger number of samples compared to the number of training samples (2000 samples of training vs. 3000 samples of test), to the presence of images inside the dataset (taken from the motorway cameras) that move slightly away from the template of images taken by the CCTV and to the having images classified in three classes and not 4 (no haze class). Thus the model chosen as the final model is the model performed with the transfer learning process through the VGG16 pre-trained network. The next section shows the result from a point of view of the confusion matrix.

6. EVALUATION RESULTS FROM CONFUSION MATRIX

The confusion matrix is a method that is able to evaluate classification problem through, given a dataset, the showing if the classes are correctly predicted for each image or not. This method was useful in order to note which classes are confused with other classes for both methods. The gure 9 on page 7 shows the confusion matrix considering the dataset MWI composed of 400 images. From both confusion, matrix is possible to note that the network performed by transfer learning classify images with a minimum error. The only error that is present on predictions made by TL network is to confuse the class

'HAZE' with the class 'SNOWY' (7 images) and the class 'RAINY' with class 'SNOWY' (1 images). In fact, if we look at the dataset of training we notice that some images can belong to both two distinct classes:

FIG. 5. An example of image where the model has to evaluate an image that can belongs to two different class



The confusion matrix provided by model CNN, instead, is influenced by more errors which lead a misclassification. This behavior is due to training with a reduced number of samples, unlike the transfer learning model which uses a pre-training network with a dataset that contains over 14 million images bellowing to 1000 different classes. Therefore it can be affirmed that the model performed with the transfer learning technique optimized by VGG16 has a better prediction trend than the implementation model following a classic CNN.[3]

7. EVALUATION ABOUT BLIND TEST SET

A collection of images produced by my self has been taken in order to create a blind dataset of test and seeing how models predict them. The figure 10 on page 7 shows the confusion matrices get from models trained with their relative errors about classes predicted. The results are not very faithful to the promises, especially from the outcomes predicted by the CNN model. This, however, is not a problem of model but data presented as a blind test. The images collected are not close to the context of images taken from the CCTV dataset, produced with a lower resolution (many of them produced with a smartphone) and with a climate representation not totally evident, as the bottom example images show:

FIG. 6. An example of image taken from dataset blind test set

Finally, bringing the attention to the model transfer learning, the results are acceptable: it is able to classify the images provided by blind dataset with more generality and precision with respect to the CCN model halving or eliminating classification errors (CCN = RAINY in HAZE = 10% vs CCN-TL = RAINY in HAZE = 5%) The results relative to the blind images of test have been placed in a special folder (.../resources/1860363.csv w.r.t transfer learning model) inside the predicted class folder.

8. CONCLUSION

In conclusion, it can be affirmed that the transfer learning model with the same dataset and implementation process turns out to be an excellent classifier that allows to classify with greater generality rather than to a CNN network without TL method which needs to be trained with an enormous quantity of images in order to generalize the images in own classes.

9. REFERENCE

[1] *Rock images classification by using deep convolution neural network* Guo, Wenhui Journal of Physics: Conference Series 2017/08/01

[2] *A survey of transfer learning* Weiss, Karl, Khoshgoftaar, Taghi, Wang, DingDing Journal of Big Data 2016/12/01

[3] *Weather Prediction using Classification* Abrar, Muhammad,Sim, Alex,Shah, Dilawar,Khusro, Shah,Abdusalam 2014/12/25

[4] *Implementation of Training Convolutional Neural Networks* Liu, Tianyi,Fang, Shuangsang,Zhao, Yuehui,Wang, Peng,Zhang, Jun 2015/06/03

FIG. 7. **Trend of accuracy and loss fucntion of both models using as testset the MWI-Dataset-1.1.1_400 with 400 images (100 for each class: HAZE,RAINY,SNOWY and SUNNY)**

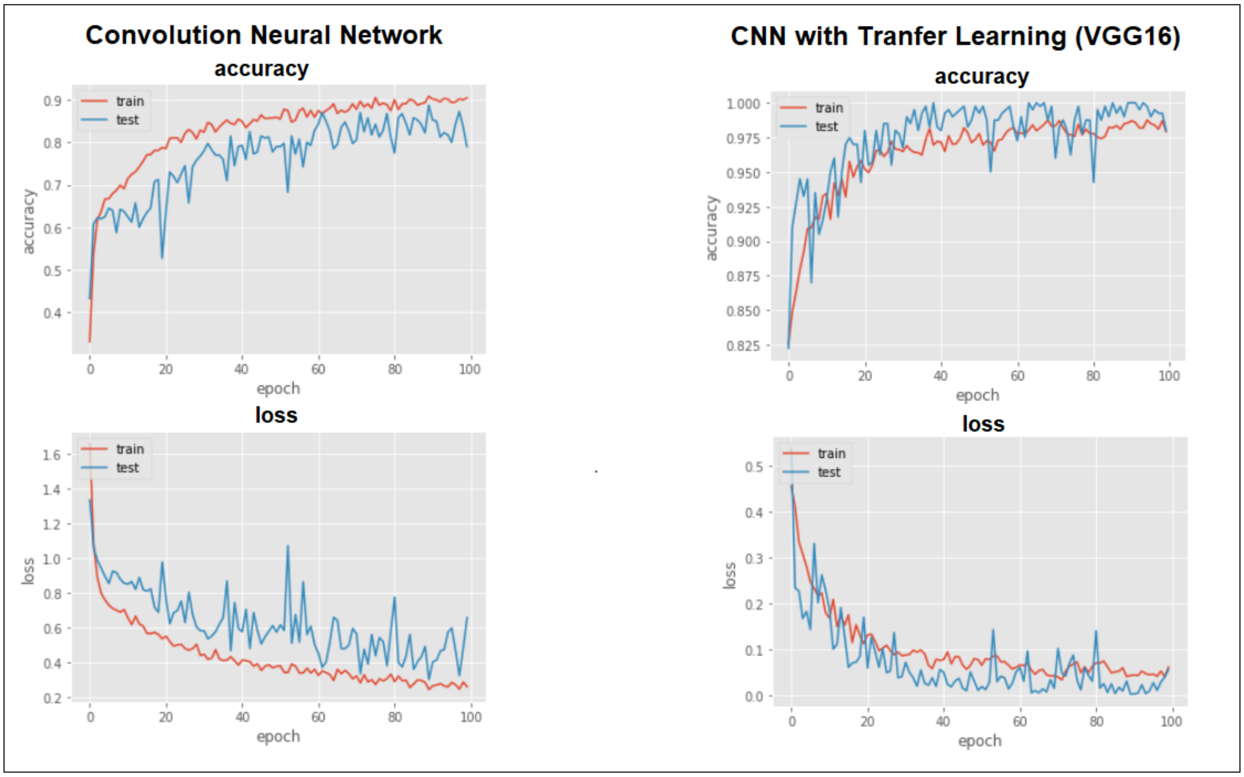


FIG. 8. **Trend of accuracy and loss fucntion of both models using as testset the TestSet**$_{Dataset}$)

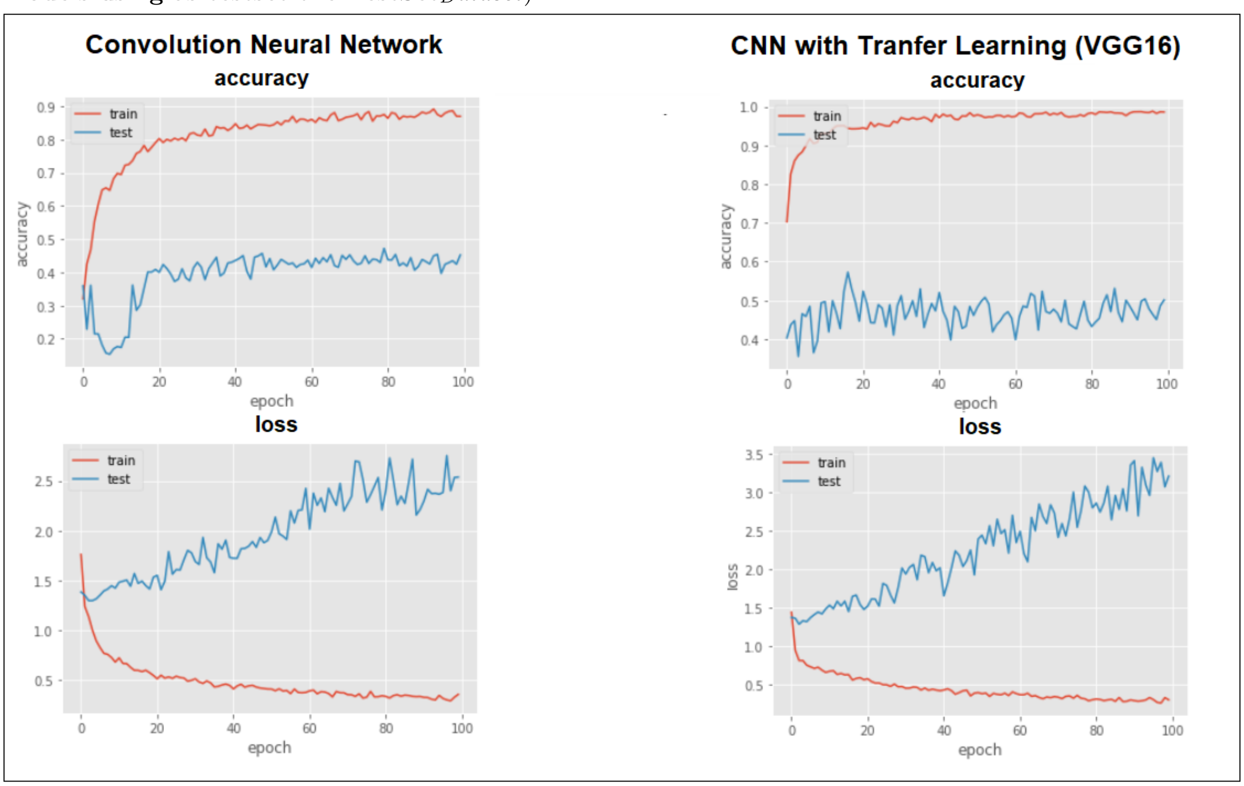


FIG. 9. The convolution neural network architecture used

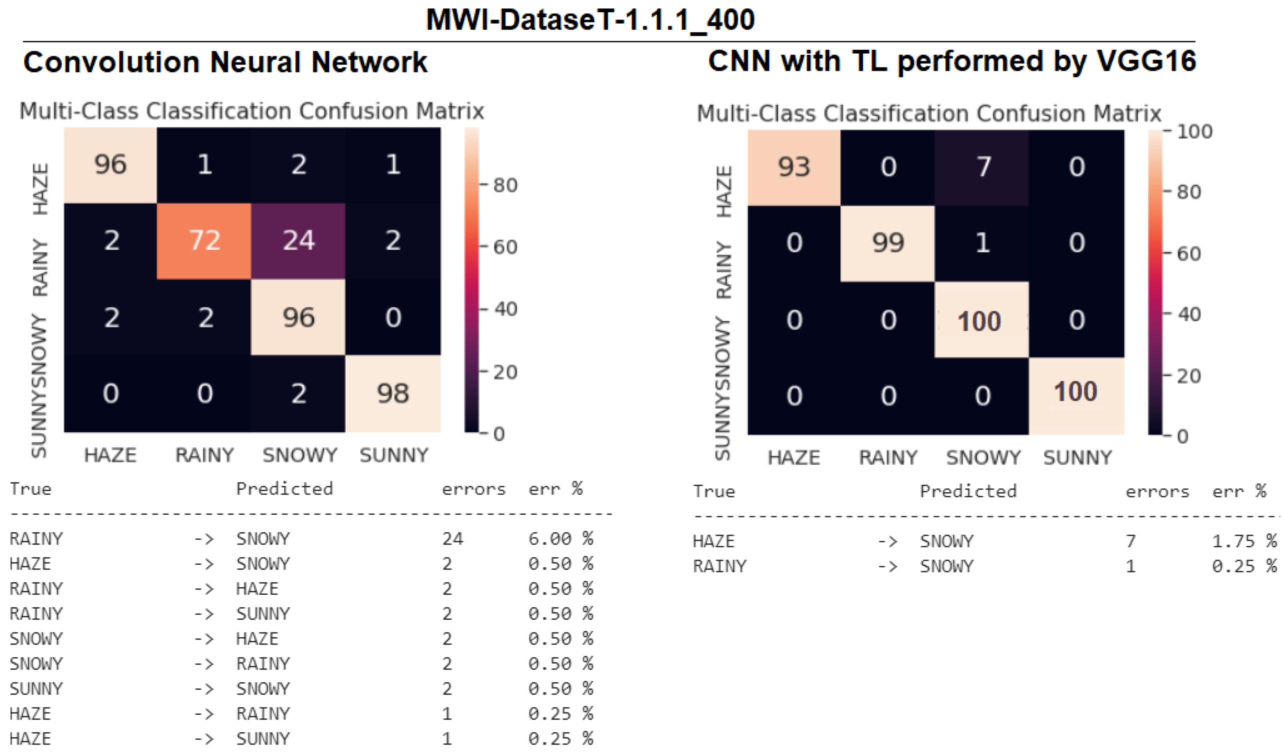


FIG. 10. The convolution neural network architecture used