import org.datavec.api.io.labels.ParentPathLabelGenerator;

import org.datavec.api.split.FileSplit;

import org.datavec.image.loader.NativeImageLoader;

import org.datavec.image.recordreader.ImageRecordReader;

import org.deeplearning4j.nn.api.OptimizationAlgorithm;

import org.deeplearning4j.nn.conf.MultiLayerConfiguration;

import org.deeplearning4j.nn.conf.NeuralNetConfiguration;

import org.deeplearning4j.nn.conf.inputs.InputType;

import org.deeplearning4j.nn.conf.layers.OutputLayer;

import org.deeplearning4j.nn.transferlearning.TransferLearning;

import org.deeplearning4j.nn.transferlearning.TransferLearningHelper;

import org.deeplearning4j.transferlearning.vgg16.VGG16Helper;

import org.deeplearning4j.zoo.model.VGG16;

import org.deeplearning4j.zoo.PretrainedType;

import org.nd4j.linalg.activations.Activation;

import org.nd4j.linalg.dataset.api.iterator.DataSetIterator;

import org.nd4j.linalg.dataset.api.iterator.MultipleEpochsIterator;

import org.nd4j.linalg.dataset.api.preprocessor.VGG16ImagePreProcessor;

import org.nd4j.linalg.learning.config.Adam;

import org.nd4j.linalg.lossfunctions.LossFunctions;

import java.io.File;

import java.util.Random;

public class PoultryDiseaseClassifier {

public static void main(String[] args) throws Exception {

int height = 224;

int width = 224;

int channels = 3;

int batchSize = 16;

int epochs = 3;

long seed = 123;

// Dataset directory

File dataDir = new File("poultry\_dataset");

FileSplit fileSplit = new FileSplit(dataDir, NativeImageLoader.ALLOWED\_FORMATS, new Random(seed));

ParentPathLabelGenerator labelMaker = new ParentPathLabelGenerator();

// Image loading and preprocessing

ImageRecordReader recordReader = new ImageRecordReader(height, width, channels, labelMaker);

recordReader.initialize(fileSplit);

DataSetIterator dataIter = new RecordReaderDataSetIterator(recordReader, batchSize, 1, recordReader.getLabels().size());

dataIter.setPreProcessor(new VGG16ImagePreProcessor());

// Load pretrained VGG16 model

VGG16 pretrained = VGG16.builder().build();

System.out.println("Loading pretrained VGG16...");

org.deeplearning4j.nn.graph.ComputationGraph vgg16 = (org.deeplearning4j.nn.graph.ComputationGraph) pretrained.initPretrained(PretrainedType.IMAGENET);

// Transfer Learning: Modify final layers

ComputationGraph model = new TransferLearning.GraphBuilder(vgg16)

.setFeatureExtractor("fc2") // Freeze up to fc2

.removeVertexAndConnections("predictions")

.addLayer("new\_predictions",

new OutputLayer.Builder(LossFunctions.LossFunction.NEGATIVELOGLIKELIHOOD)

.nIn(4096).nOut(recordReader.getLabels().size())

.activation(Activation.SOFTMAX).build(), "fc2")

.setOutputs("new\_predictions")

.build();

model.fit(dataIter, epochs);

System.out.println("Training complete!");

}

}