unesco
University Network
Member

Ministry of
Education

Object Detection using Deep Learning

Sung Soo Hwang

YOLO

- Example code

```
#include "cv.hpp"
#include <iostream>
#include <opencv2/dnn.hpp>
#include <fstream>

using namespace cv;
using namespace std;
using namespace dnn;

int main(int argc, char** argv)
{
		String modelConfiguration = "deep/yolov2-tiny.cfg";
		String modelBinary = "deep/yolov2-tiny.weights";

		Net net = readNetFromDarknet(modelConfiguration, modelBinary);

		VideoCapture cap("deep/downtown_road.mp4");

		vector<String> classNamesVec;
		ifstream classNamesFile("deep/coco.names");

		if (classNamesFile.is_open()) {
				string className = "";
				while (std::getline(classNamesFile, className)) classNamesVec.push_back(className);
		}
```

YOLO

- Example code

```
while (1)
{
        Mat frame;
        cap >> frame; // get a new frame from camera/video or read image
        if (frame.empty()) {
                waitKey();
                break;
        }

        if (frame.channels() == 4) cvtColor(frame, frame, COLOR_BGRA2BGR);

        // Convert Mat to batch of images
        Mat inputBlob = blobFromImage(frame, 1 / 255.F, Size(416, 416), Scalar(), true, false);
        net.setInput(inputBlob, "data");                //set the network input
        Mat detectionMat = net.forward("detection_out");    //compute output

        float confidenceThreshold = 0.24; //by default

        for (int i = 0; i < detectionMat.rows; i++)   {
                const int probability_index = 5;
                const int probability_size = detectionMat.cols - probability_index;
                float *prob_array_ptr = &detectionMat.at<float>(i, probability_index);
                size_t objectClass = max_element(prob_array_ptr, prob_array_ptr + probability_size) - prob_array_ptr;
                // prediction probability of each class
                float confidence = detectionMat.at<float>(i, (int)objectClass + probability_index);
```

YOLO

- Example code

```
// for drawing labels with name and confidence
if (confidence > confidenceThreshold) {
                float x_center = detectionMat.at<float>(i, 0) * frame.cols;
                float y_center = detectionMat.at<float>(i, 1) * frame.rows;
                float width = detectionMat.at<float>(i, 2) * frame.cols;
                float height = detectionMat.at<float>(i, 3) * frame.rows;

        Point p1(cvRound(x_center - width / 2), cvRound(y_center - height / 2));
        Point p2(cvRound(x_center + width / 2), cvRound(y_center + height / 2));
        Rect object(p1, p2);
        Scalar object_roi_color(0, 255, 0);

        rectangle(frame, object, object_roi_color);
        String className = objectClass < classNamesVec.size() ? classNamesVec[objectClass] :
        cv::format("unknown(%d)", objectClass);
        String label = format("%s: %.2f", className.c_str(), confidence);
        int baseLine = 0;

        Size labelSize = getTextSize(label, FONT_HERSHEY_SIMPLEX, 0.5, 1, &baseLine);
```

YOLO

- Example code

```
                                        rectangle(frame, Rect(p1, Size(labelSize.width, labelSize.height + baseLine)), object_roi_color, FILLED);
                                        putText(frame, label, p1 + Point(0, labelSize.height), FONT_HERSHEY_SIMPLEX, 0.5, Scalar(0, 0, 0));
                                    }
                                }
                                imshow("YOLO: Detections", frame);
                                if (waitKey(1) >= 0) break;
                            }
                            return 0;
}
```

YOLO

- What does the result matrix of YOLO mean?
  - Each row represents each bounding box.
  - 845 rows
    - YOLO divides the image into 13x13 grids, and each grid predicts 5 bounding boxes.
  - 85 columns
    - 4 position values of each bounding box
      : center x, center y, width, height
    - 1 box confidence
    - 80 class confidence

YOLO

- In the example code …
  - Mat detectionMat
    : result matrix of YOLO
  - const int probability_index = 5
    : offset to get the probability of each class because there are 5 values about bounding boxes
  - size_t objectClass
    : index of the largest confidence class
  - float confidence
    : the largest probability among 80 detection results
  - float x_center, y_center, height, width
    : position values of each bounding box