You Only Look Once (YOLO) Real-Time Object Detection

Gordon Keil, Computer Engineering

Faculty Advisor: Dr. Salomon Oldak

Kellogg Honors College Capstone Project

Abstract: Computer vision is an area of technology that has become useful in the development of autonomous vehicles, video surveillance systems, and manufacturing systems. Specifically, object detection is utilized in the aforementioned computer vision applications. You Only Look Once (YOLO) is a new approach to object detection that outperforms other detection methods such as Deformable Parts Models (DPM) and Region Convolutional Neural Network (R-CNN). This object detection method makes use of a single convolutional neural network (CNN) to predict multiple bounding boxes around objects of interest in an image or video frame and class probabilities for those boxes. This research project explains how YOLO works and shows the results of training the neural network on a custom dataset.

Introduction

You Only Look Once (YOLO) sees an image to predict what objects are present and where they are. This object detection method is very simple. A single convolutional neural network (CNN) simultaneously predicts multiple bounding boxes and class probabilities for those boxes. YOLO trains on full images and directly optimizes detection performance. YOLO has several benefits over other traditional methods of object detection. First, YOLO is extremely fast. A neural network is simply run on a new image at test time to predict detections. Second, YOLO reasons globally about the image when making predictions. Unlike sliding window and region-based techniques, YOLO sees the entire image during training and test time so it implicitly encodes contextual information about classes as well as their appearance. Third, YOLO learns generalizable representations of objects. Since this is the case, YOLO is less likely to break down when applied to new domains or unexpected inputs [1].

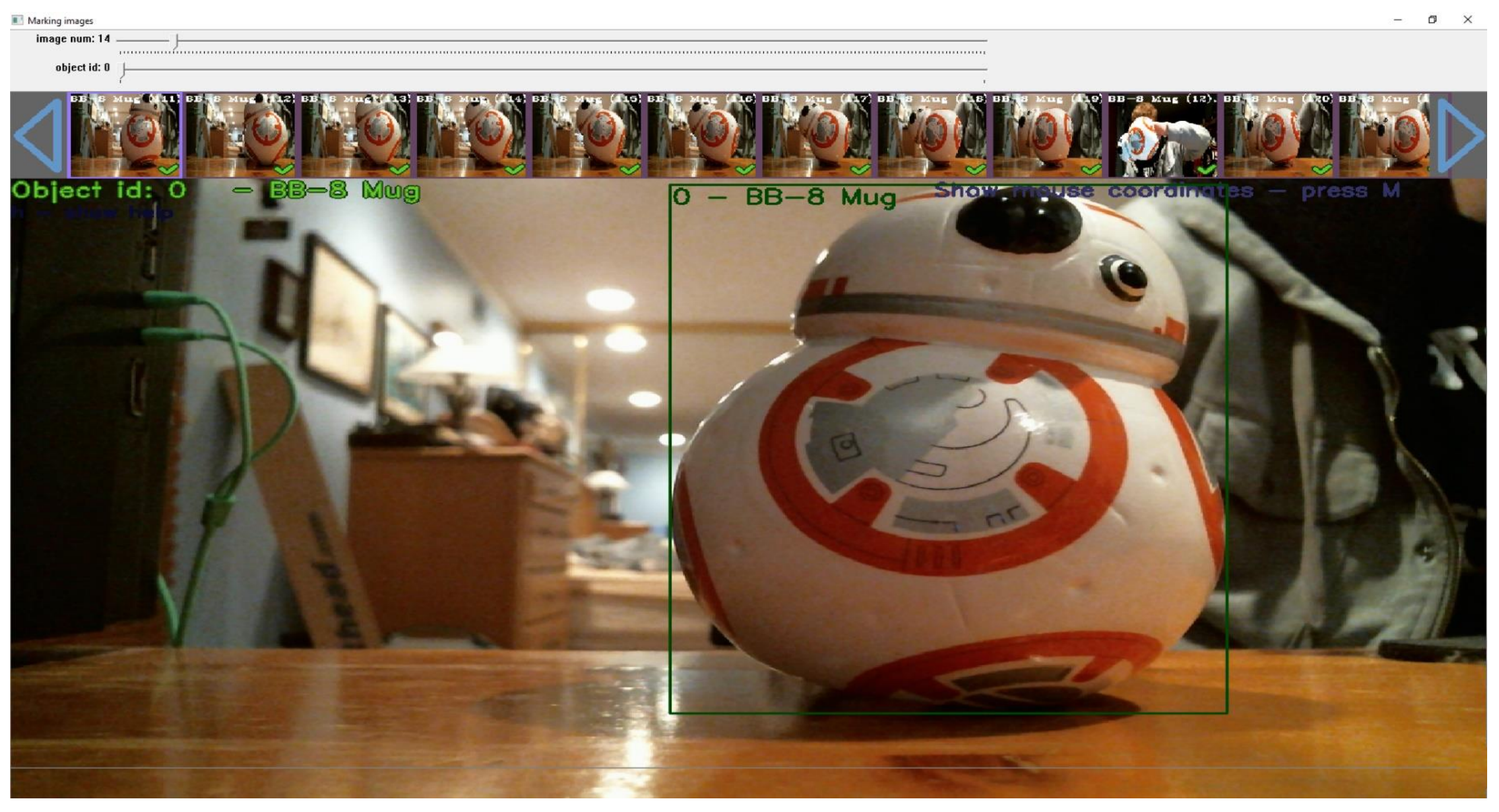
Comparison to Other Detection Systems

Deformable Parts Models (DPM): This system uses a sliding window approach to object detection. DPM uses a disjoint pipeline to extract static features, classify regions, etc. YOLO replaces all of these disparate parts with a single CNN. The CNN behind YOLO performs feature extraction, bounding box prediction, non-maximal suppression, and contextual reasoning all concurrently.

R-CNN: This system and its variants use region proposals instead of sliding windows to find objects in images. Selective Search generates bounding boxes, a CNN extracts features, an SVM scores the boxes, a linear model adjusts the bounding boxes, and non-max suppression eliminates duplicate detections. Each stage of this complex pipeline must be precisely tuned independently and the resulting system is very slow, taking more than 40 seconds per image at test time [1].

Training YOLO On a Custom Dataset

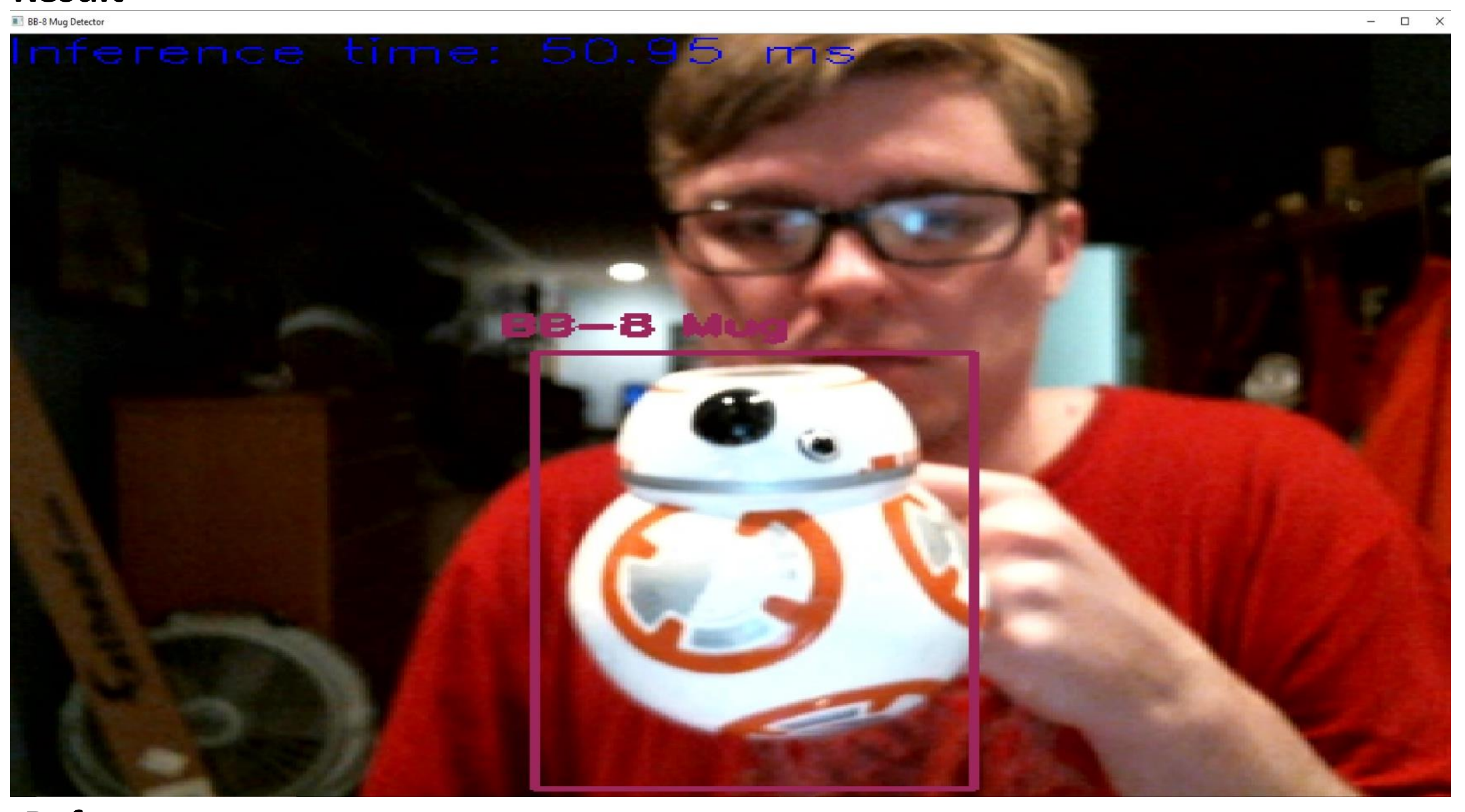
The object of interest for the YOLO real-time object detection system was a BB-8 mug. A program called "YOLO mark" [4] was used to mark this object in a total of 228 images with different angles and lighting conditions. This program is a graphical user interface that allows you to draw a bounding box around the object in each image. A text file is generated for each image that has an integer representing the object, float values that store the center of the object relative to the width and height of the image, and float values that store the width and height of the object itself. Below is a picture of the "YOLO mark" program running with the bounding box highlighting the object:



Execution

The CNN used for this object detector was a pre-made neural network called "Darknet53" [5]. This CNN had to be trained on all of the images taken with their respective text file. A python program ran with the path to the weights file generated from training on Darknet53. The result can be seen below. The time taken for the object detector to identify the object (inference time) is also shown.

Result



Overview of Convolutional Neural Networks (CNNs)

A CNN is the main system behind identifying objects with YOLO. This algorithm can take an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other.

Components of a CNN:

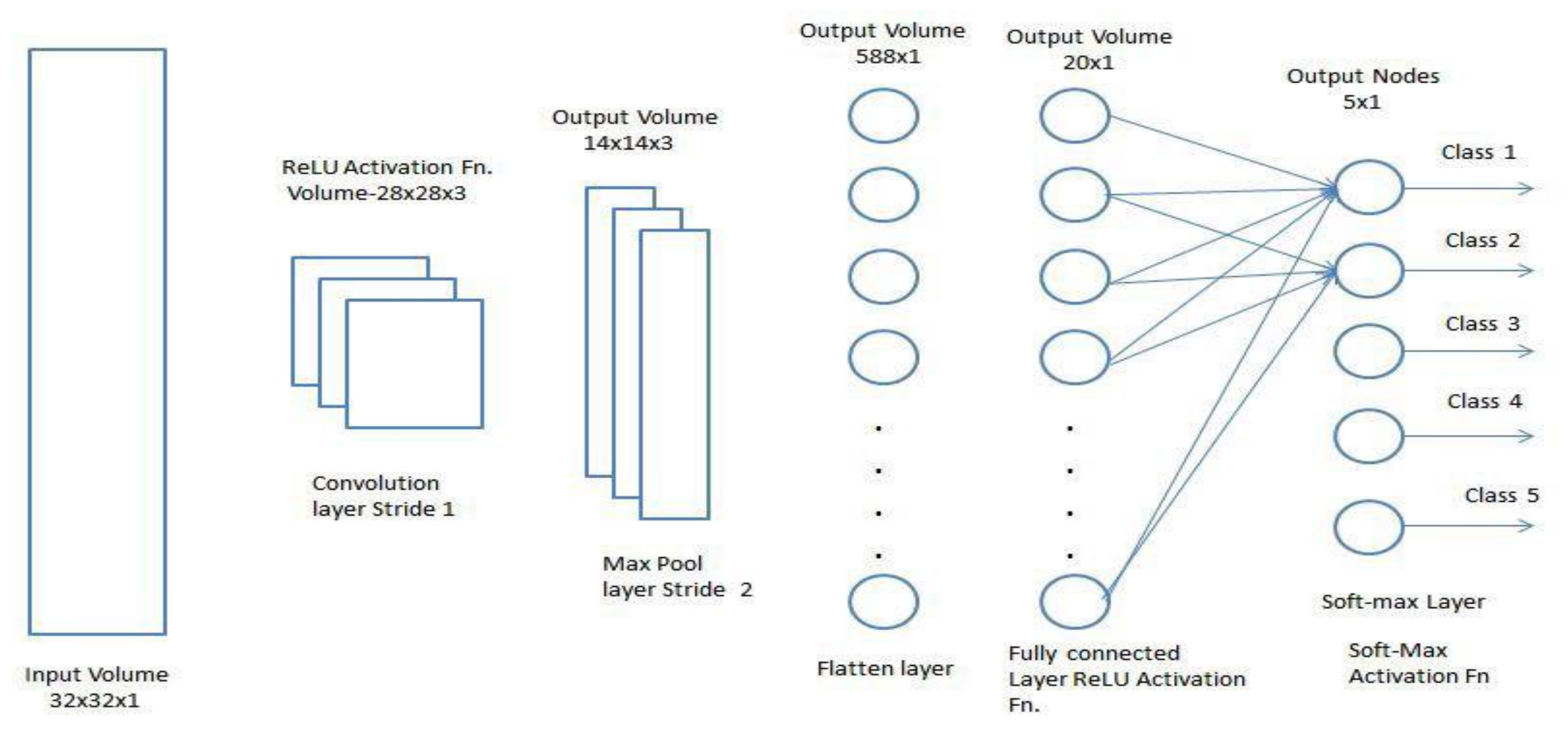


Figure 1: Convolutional Neural Network
Source: Adapted from [3]

1. **Input image**. The role of a CNN is to reduce the images into a form which is easier to process, without losing features which are critical for getting a good prediction.

2. **Convolution Layer**. Below is an illustration of what happens in this step. The green section resembles a 5x5x1 input image. The element involved in carrying out the convolutional operation in the first part of a Convolutional Layer is called the Kernel/Filter, K, represented in yellow. K is selected as a 3x3x1 matrix. The Kernel shifts over the input image and performs a matrix multiplication operation between K and the portion P of the image over which the kernel is hovering.

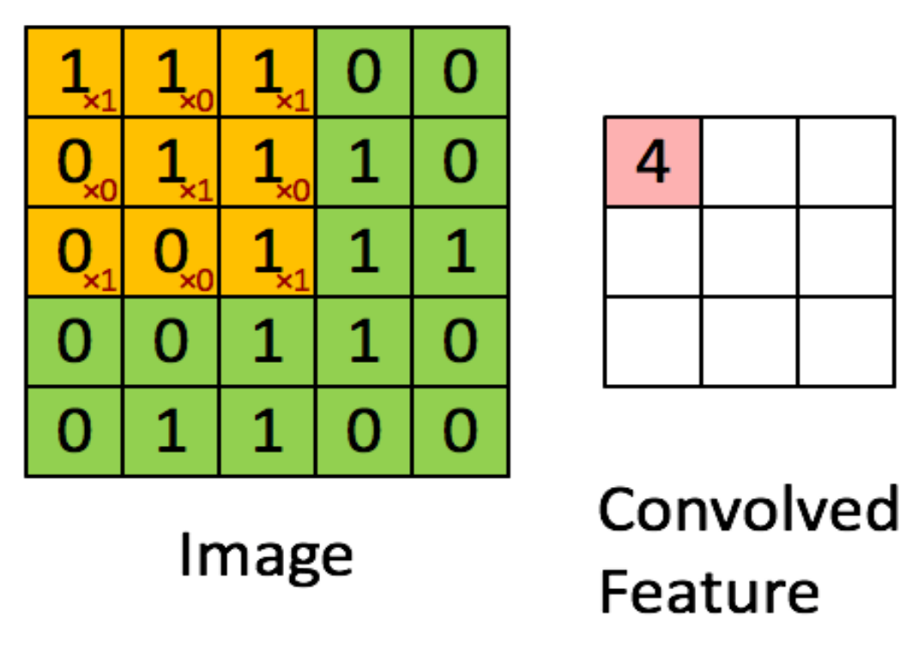


Figure 2: Convolution Operation
Source: Adapted from [3]

The objective of the Convolution Operation is to extract the high-level features such as edges, from the input image. The first convolutional layer is responsible for capturing the Low-Level features such as edges, color, gradient, orientation, etc. With added layers, the architecture adapts to the high-level features as well, resulting in a network which has a complete understanding of the images in the dataset.

3. **Pooling Layer**. This layer is responsible for reducing the spatial size of the Convolved Feature obtained in the Convolutional Layer. This is to decrease the computational power required to process the data through dimensionality reduction. In addition, it is useful for extracting dominant features which are rotational and positional invariant, thus maintaining the process of effectively training the model.

The Convolutional Layer and the Pooling Layer, together form the i-th layer of a CNN. After going through the above process, we have successfully enabled the model to understand the features.

4. **Flatten Layer**. After the image is converted into a suitable form, the image is flattened into a column vector. The flattened output is fed to a feed-forward neural network and backpropagation applied to every iteration of training.

5. **Classification – Fully Connected Layer (FC Layer)**. This layer is for learning non-linear combinations of the high-level features as represented by the output of the convolutional layer. The Fully-Connected layer is learning a possibly non-linear function.

6. **Soft-max Layer**. Over a series of epochs, the model is able to distinguish between dominating and certain low-level features in images and classify them using the Softmax Classification technique. Basically, this provides a list of different objects that could be in the image each with a probability. After this stage, the object is identified [3].

YOLO CNN Architecture

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
1×	Convolutional	64	3 × 3	
1×	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
2×	Convolutional	128	3 × 3	
2×	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
8×	Convolutional	256	3 × 3	
8×	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
8×	Convolutional	512	3 × 3	
8×	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
4×	Convolutional	1024	3 × 3	
4×	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Figure 3: Darknet53 Architecture
Source: Adapted from [2]

Description

This CNN is used for performing feature extraction for YOLO object detection. This network uses successive 3x3 and 1x1 convolutional layers. There are 53 convolutional layers in total and hence the name "Darknet53" for this network [2].

References

[1] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," *Computer Vision and Pattern Recognition Expo 2016*. [Online]. Available: https://pjreddie.com/media/files/papers/yolo_1.pdf. [Accessed Oct. 28, 2018].

[2] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," University of Washington. [Online]. Available: https://pjreddie.com/media/files/papers/YOLOv3.pdf. [Accessed Jan. 8, 2019].

[3] S. Saha, "A Comprehensive Guide to Convolutional Neural Networks – the ELI5 way," *Medium: Towards Data Science*, Dec. 15, 2018. [Online], Available: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53. [Accessed Jan. 10, 2019].

[4] AlexeyAB, "Yolo-mark," May, 2018. [Online]. Available: https://github.com/AlexeyAB/Yolo_mark [Accessed Feb. 6, 2019].

[5] AlexeyAB, "Yolo-v3 and Yolo-v2 for Windows and Linux," January, 2019. [Online]. Available: https://github.com/AlexeyAB/darknet#how-to-train-to-detect-your-custom-objects [Accessed Feb. 6, 2019]