Machine Learning Capstone Project

Smart Car

Capstone Proposal Overview

Creating accurate machine learning models capable of localizing and identifying multiple objects in a single image or video remains a core challenge in computer vision.

Domain Background

college research for detect objects on videos or images inside the college to use it to get information about the place in real time detection with smart car embedded system that will detect.

we focus to detect this objects in the college:

- potted plant
- trees
- persons
- bags
- backpack
- lecture tables
- projector screens
- dining tables
- tv
- dogs

Research on Computer Vision-Based Object Detection and Classification

https://link.springer.com/chapter/10.1007/978-3-642-36124-1_23

Problem Statement

Learning smart car how to detect and define objects when it's move in the real time inside the college and tell us about it.

It's classification problem, to give objects its label, the input will be pictures and videos and the output is the detection and classification of the objects on it.

Data Sets and Inputs

We will download the model which is trained on the ***COCO dataset***. COCO stands for **Common Objects in Context,** this dataset contains around 330K labeled images. http://cocodataset.org/#home

- There will be around 15 images examples and 1 video to show the result.
- It contains around 8 balanced classes.
- The training set will be from coco dataset and the validation and testing set will be images and videos from college that, the ratio will be 1:3

The input cases:
- images from the college.
- real time capturing inside college.
- videos from the college.

Input constrains:
- the images can be colored or black and white, the dimensions is more than 200*200 pixels

Solution Statement

The Tensor Flow Object Detection API is an
open source framework built on top of Tensor Flow that makes it easy to construct, train and deploy object detection models.

Object Detection can be done via multiple ways:

- Feature-Based Object Detection
- Viola Jones Object Detection
- SVM Classifications with HOG Features
- Deep Learning Object Detection

In this Object Detection project, we'll focus on **Deep Learning Object Detection** as Tensor flow uses Deep Learning for computation.

```
MODEL_NAME =
'ssd_mobilenet_v1_coco_11_06_2017'
```

Benchmark Model

From tensor flow API:

```
MODEL_NAME = 'ssd_mobilenet_v1_coco_2017_11_17'
```

Evaluation Metrics

tf.metrics.precision

Computes the precision of the predictions with respect to the labels.

The `precision` function creates two local variables, `true_positives` and `false_positives`, that are used to compute the precision. This value is ultimately returned as `precision`, an idempotent operation that simply divides `true_positives` by the sum of `true_positives` and `false_positives`.

For estimation of the metric over a stream of data, the function creates an `update_op` operation that updates these variables and returns the `precision`. `update_op` weights each prediction by the corresponding value in `weights`.

what are all of the steps that you'll take during pre-processing?

installation:

- GPU supported Nvidea card with compute capability 3.0 or higher.

- Python v3.5 or v3.6 installed.

- Installing CUDA v9.0

- Installing cuDNN v7.0

- Installing TensorFlow

- Installing the required dependencies

Will you need to re-size images?

yes, IMAGE_SIZE = (12, 8) inches

What types of deep learning do you plan on using?

Convolutional Neural Networks (CNNs)

Project Design

The general sequence of steps are as follows:

a. **Data Visualization:** Visual representation of data to find the degree of correlations between predictors and target variable and find out correlated predictors. Additionally, we can see ranges and visible patterns of the predictors and target variable.
b. **Data Preprocessing:** Scaling and Normalization operations on data and splitting the data in training, validation and testing sets.
c. **Feature Engineering:** Finding relevant features, engineer new features using methods like PCA if feasible.
d. **Model Selection:** Experiment with various algorithms to find out the best algorithm for this use case.
e. **Model Tuning:** Fine tune the selected algorithm to increase performance without overfitting.
f. **Testing:** Test the model on testing dataset.