भारतीय प्रौद्योगिकी संस्थान जम्मू
INDIAN INSTITUTE OF TECHNOLOGY JAMMU
विद्याधनं सर्वधन प्रधानम्

7/12/2019

Image Tagging

Project Report

Submitted to :

Prof. Virendra Singh

Indian Institute of Technology Bombay

Summer Internship 2019

20th May – 10th July 2019

Kabir Juneja, Manas Ghai

IIT JAMMU

Preface

*This report document is the work done during the summer internship at **IIT Bombay** on Image Tagging, under the supervision of **Dr. Virendra Singh**. The report will give an overview of the tasks completed during the period of internship with technical details. Then the results obtained are discussed and analyzed. We have tried our best to keep report simple yet technically correct. We hope we succeed in our attempt.*

Kabir Juneja

Manas Ghai

Acknowledgement

We take this opportunity to acknowledge all the people who have helped us whole heartedly in every stage of this project.

We are indebtedly grateful to *Dr. Virendra Singh* for his support. We are also grateful to our Project guide *Dr. Chandramani Chaudhary* for her valuable guidance.

We also extend our sincere thanks to *People Invited by Prof. Virendra Singh* for their valuable inspiration. We also extend our sincere thanks to all other PhD scholars of Electrical Engineering Department and our Colleagues.

We would like to thank authorities of *Indian Institute of Technology Bombay* to let us be a part of their institution and learn from it.

Introduction

This project is taken as an attempt to Tag Satellite Images(Application Based Research) using Deep Neural Networks and their analysis and the dataset is taken from Kaggle. Kaggle is a platform for predictive modeling and Analytics competitions. Here organization and researchers post the data. Statisticians and data scientists from all over the world compete to produce the best models.

Satellite images are part of the big data revolution. These images are captured through remote sensing technologies – like drones, aerial photographs and satellite sensors – without physical contact or firsthand experience. Algorithms refine these data to describe places and phenomena on the Earth's surface and in the atmosphere.

Satellite images are processed to demonstrate seasonal vegetation change. This information now helps to monitor vegetative health and track droughts around the world.

Images also provide evidence of compelling stories about the power of disasters. For example, in 1986, combined data modeled from satellite images and weather data tracked the plume of radiation from the explosion of the Chernobyl reactor in the USSR.

Processing satellite images is computationally intensive. At best, satellite images are interpretations of conditions on Earth – a "snapshot" derived from algorithms that calculate how the raw data are defined and visualized.

Convolutional Neural Networks (CNNs) :

In neural networks, Convolutional neural network (ConvNets or CNNs) is one of the main categories to do images recognition, images classifications. Objects detections, recognition faces etc., are some of the areas where CNNs are widely used.

CNN image classifications takes an input image, process it and classify it under certain categories (Eg., Dog, Cat, Tiger, Lion). Computers sees an input image as array of pixels and it depends on the image resolution. Based on the image resolution, it will see h x w x d(h = Height, w = Width, d = Dimension). Eg., An image of 6 x 6 x 3 array of matrix of RGB (3 refers to RGB values) and an image of 4 x 4 x 1 array of matrix of grayscale image.

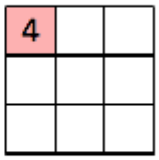
Technically, deep learning CNN models to train and test, each input image will pass it through a series of convolution layers with filters (Kernals), Pooling, fully connected layers (FC) and apply Softmax function to classify an object with probabilistic values between 0 and 1

Convolution Layer

Convolution is the first layer to extract features from an input image. Convolution preserves the relationship between pixels by learning image features using small squares of input data. It is a mathematical operation that takes two inputs such as image matrix and a filter or kernel

Consider a 5 x 5 whose image pixel values are 0, 1 and filter matrix 3 x 3 as shown

Image

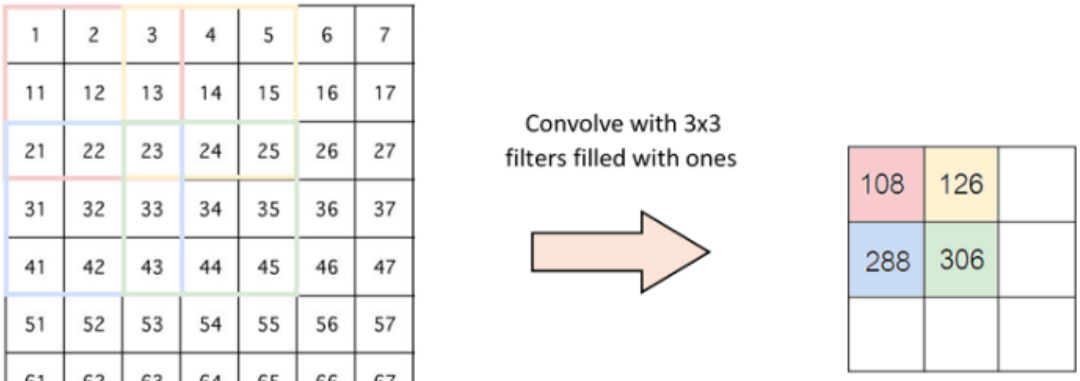


Convolved Feature

Then the convolution of 5 x 5 image matrix multiplies with 3 x 3 filter matrix which is called **"Feature Map"** as output shown in above.
Convolution of an image with different filters can perform operations such as edge detection, blur and sharpen by applying filters.

Strides

Stride is the number of pixels shifts over the input matrix. When the stride is 1 then we move the filters to 1 pixel at a time. When the stride is 2 then we move the filters to 2 pixels at a time and so on. The below figure shows convolution would work with a stride of 2.



Padding
Sometimes filter does not fit perfectly fit the input image. We have two options:

1. Pad the picture with zeros (zero-padding) so that it fits
2. Drop the part of the image where the filter did not fit. This is called valid padding which keeps only valid part of the image.

Non Linearity(ReLU)

ReLU stands for Rectified Linear Unit for a non-linear operation. The output is ***ƒ(x) = max(0, x).***

Why ReLU is important : ReLU's purpose is to introduce non-linearity in our ConvNet. Since, the real world data would want our ConvNet to learn would be non-negative linear values.

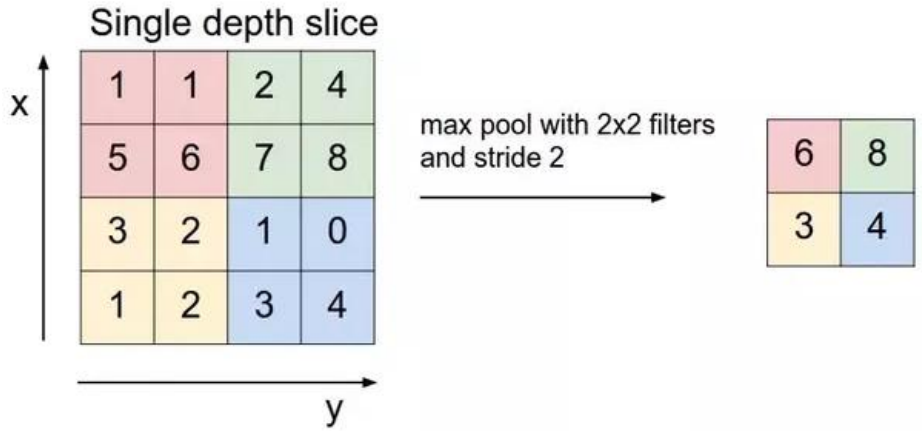
There are other non linear functions such as tanh or sigmoid can also be used instead of ReLU. Most of the data scientists uses ReLU since performance wise ReLU is better than other two.

Pooling Layer

Pooling layers section would reduce the number of parameters when the images are too large. Spatial pooling also called subsampling or downsampling which reduces the dimensionality of each map but retains the important information. Spatial pooling can be of different types:

1. Max Pooling
2. Average Pooling
3. Sum Pooling

Max pooling take the largest element from the rectified feature map. Taking the largest element could also take the average pooling. Sum of all elements in the feature map call as sum pooling.



Problem Statement

There are images of Earth taken by Satellite with tags associated to each image, where tags are like Cultivation, Partly cloudy, Clear, Water etc. Each pixel has a single pixel value associated with it, indicating the lightness or darkness of that pixel, with higher meaning darker. The pixel value is an integer between 0 and 255, inclusive.

Goal

Take a Satellite Image, Give tags to that Image

128 * 128 (pixel values)

agriculture
artisinal_mine
clear
cultivation
habitation
primary
road
water

Model building

Develop the Analysis Plan:- For the conceptual model establishment, we need to understand the selected techniques and model implementation issue. Here we will establish predictive model, which will be in based on Deep Neural Networks. Plan is to treat this as an Image Classification since we have a Total of 17 Tags(to be associated with different Images).

Data Preprocessing:- Before model estimation and model planning, we need to preprocess our data. As suggested by our mentor Dr. Chandramani Chaudhary, We first loaded data images to a target pixel size of 128 * 128.

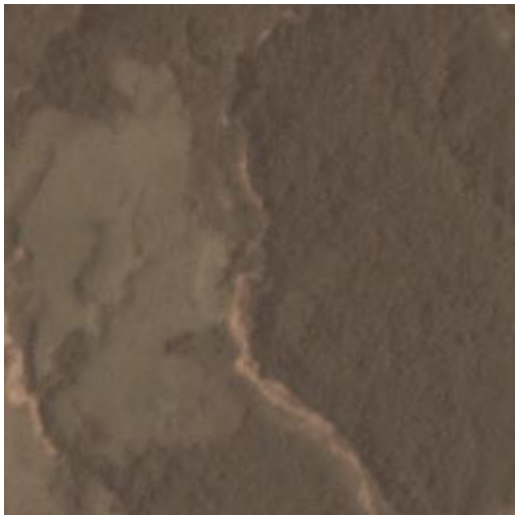
Scaling :
Since the pixel values lie between 0 and 255(inclusive), this is a whole lot of variation in pixels for our upcoming model.
Therefore, we need to **scale the pixel values** in range 0 to 1.
This is achieved just by dividing all the pixel values by 255.0

***Data Augmentation* :**
Sometimes we change our Dataset by rotating , shrinking , expanding, flipping the images in our Original dataset and then train the model on new Dataset. For this purpose, we have analysed the model using keras's ImageDataGenerator for **in-place** Data Augmentation.

Estimate the Model:- The analysis proceeds to the actual estimation of the model and assessment of overall model fit. In the estimation process, we plan to build a training model similar to VGG-16(Not exactly same but similar). Our CNN model is explained later in this report visually and in detail.

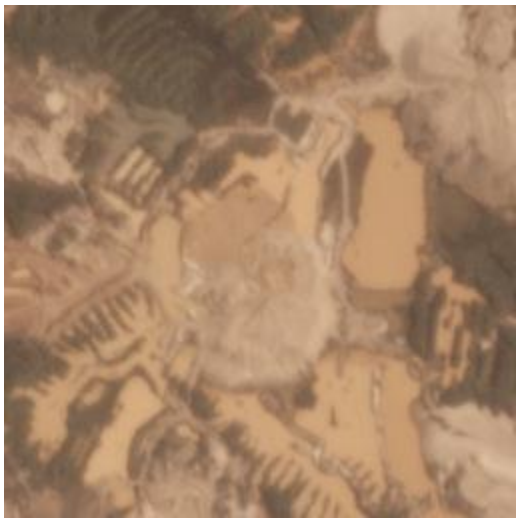
Training and Testing:- Once the model is prepared, all we need is to train our model and then run the model on the test set to obtain results that we require.

Dataset Visuals : Type of images in our Dataset ->



agriculture clear primary water

Partly Cloudy Primary



Clear Primary

Clear Primary Agriculture

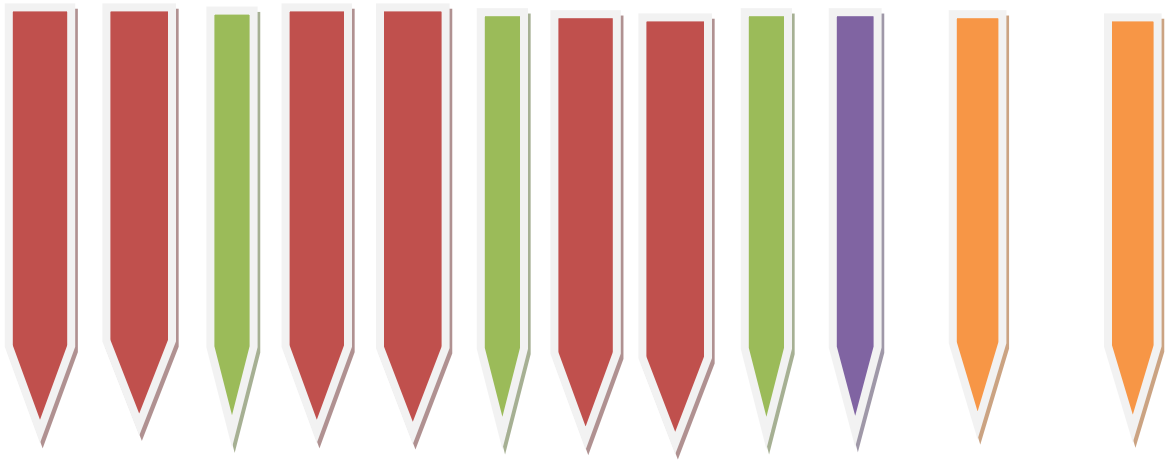
agriculture clear primary slash_burn water

bare_ground clear Water

	image_name	tags
0	train_0	haze primary
1	train_1	agriculture clear primary water
2	train_2	clear primary
3	train_3	clear primary
4	train_4	agriculture clear habitation primary road
5	train_5	haze primary water
6	train_6	agriculture clear cultivation primary water
7	train_7	haze primary
8	train_8	agriculture clear cultivation primary
9	train_9	agriculture clear cultivation primary road
10	train_10	agriculture clear primary slash_burn water
11	train_11	clear primary water
12	train_12	cloudy
13	train_13	clear primary
14	train_14	cloudy
15	train_15	clear primary
16	train_16	clear primary
17	train_17	partly_cloudy primary
18	train_18	clear primary
19	train_19	agriculture clear primary road
20	train_20	agriculture clear primary water

Other Examples of Tags Associated to Images

CNN MODEL DESIGN



Conv(3 * 3)	POOL	Conv(3 * 3)	POOL	Conv(3 * 3)	POOL	Flatten	Dense1	Dense2	
32	2*2	64	2*2	128	2*2		128	<RELU>	17

Conv Layers

Pooling Layers

Flatten Layers

Dense Layers

Above model is similar to the VGG-16 model but possess different values of parameters and different layers. In this model there also are some dropout layers as mode of regularization to reduce overfitting. Since, we found promising results in first attempt in this model, we didn't really run the actual VGG-16 model.

Model Summary :

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 128, 128, 32)      896
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 128, 128, 32)      9248
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 64, 64, 32)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 64, 64, 32)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 64, 64, 64)        18496
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 64, 64, 64)        36928
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 32, 32, 64)        0
_________________________________________________________________
dropout_2 (Dropout)          (None, 32, 32, 64)        0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 32, 32, 128)       73856
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 32, 32, 128)       147584
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 16, 16, 128)       0
_________________________________________________________________
dropout_3 (Dropout)          (None, 16, 16, 128)       0
_________________________________________________________________
flatten_1 (Flatten)          (None, 32768)             0
_________________________________________________________________
dense_1 (Dense)              (None, 128)               4194432
_________________________________________________________________
dropout_4 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_2 (Dense)              (None, 17)                2193
=================================================================
Total params: 4,483,633
Trainable params: 4,483,633
Non-trainable params: 0
```

Model Optimization Techniques

Stochastic Gradient Descent

Parameters:

Learning Rate: **0.01**

Momentum: **0.9**

Loss: **binary_crossentropy (built-in in keras sequential model compilation)**

Training parameters:

Training epochs: **20**

Batch size: **128**

Model Compilation / Optimizer Setting

```
# model compilation
# using stochastic gradient descent for optimizing our model
from keras.optimizers import SGD
opt = SGD(lr=0.01, momentum=0.9)
model.compile(optimizer=opt, loss='binary_crossentropy', metrics=['accuracy'])
```

Results

On Training Set:

Loss: 0.1970
Accuracy: 0.9210

On Test Set:

Loss: 0.1859
Accuracy: 0.9241

Assumptions/Setting: We are only taking the classes with a probability > 0.1 of being tagged to an image

```
print(photos_test.shape)
print(targets_test.shape)
# score evaluation or accuracy on the test set
score = model.evaluate(photos_test,targets_test)
score
```

```
(5000, 128, 128, 3)
(5000, 17)
5000/5000 [==============================] - 100s 20ms/step

[0.1859953994989395, 0.9241646843910217]
```

```
print(ypred.shape)
# setting all the classes with probability > 0.1 as 1 and others as 0
# probability > 0.08 and 0.15 have also been analysed but 0.1 gives promising results
y_pred_temp = np.empty((5000,17))
for i in range(5000):
    for j in  range(17):
        if(ypred[i][j]>=0.1):
            y_pred_temp[i][j]=1
        else:
            y_pred_temp[i][j]=0
print(y_pred_temp)
```

Limitations Faced

Limited by computational powers, the original Training Dataset was of size 50k images(Large enough to not been modelled by CPU), we had to reduce the size to 15k images and then apply the inplace Data Augmentation. Training Dataset was also reduced to 5k in size but it was not augmented since all the images were already new.

Limited by Time, we could not analyse other different existing models and hence came up with **our own modification to VGG-16 model**.

Limited by Knowledge and Time(again), we could not handle the case of unseen tags since it is still a big research topic going on and could not be done in a span of few weeks.

Scope of Image Annotation

In many real-life scenarios, an object can be categorized into multiple categories. E.g., a newspaper column can be tagged as "political", "election", "democracy"; an image may contain "tiger", "grass", "river"; and so on. These are instances of multi-label classification, which deals with the task of associating multiple labels with single data. It is a difficult problem because one needs to consider the intricate correlations that exist among different labels.

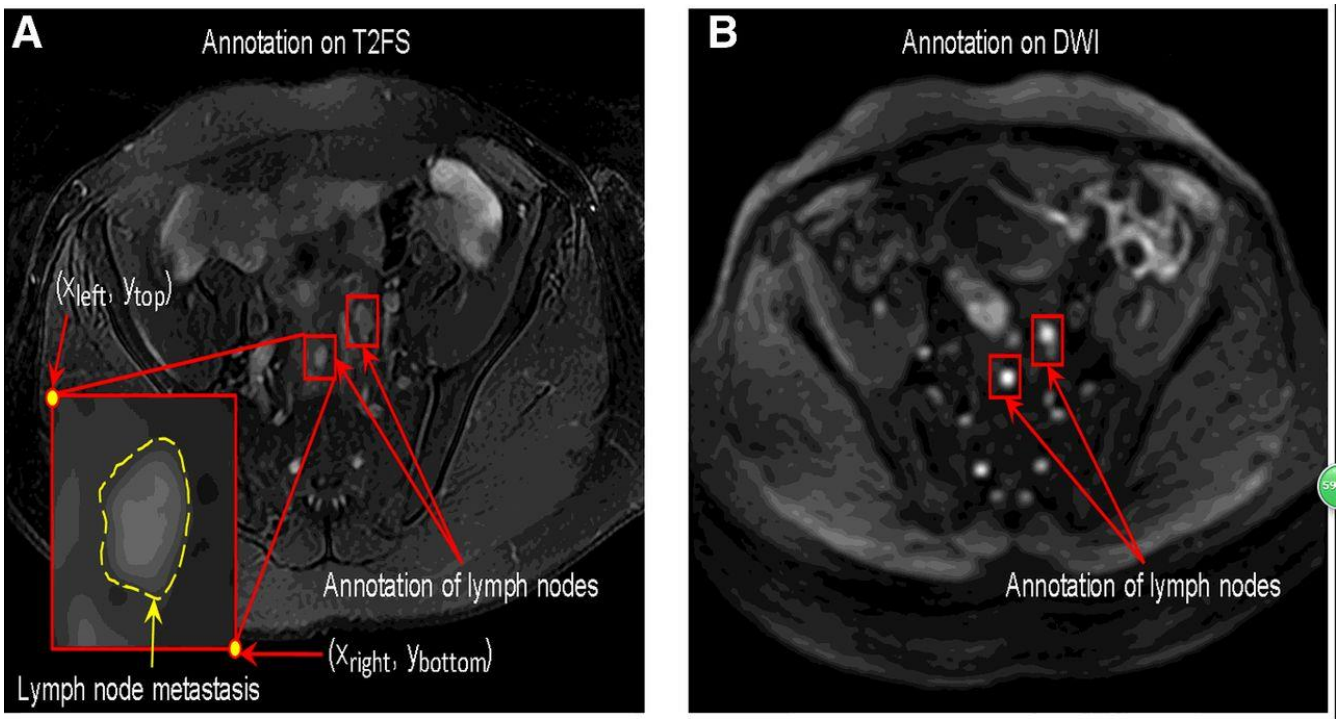
Automatic image annotation is a multi-label classification problem that aims at associating a set of textual with an image that describe its semantics. It has potential applications in image retrieval, image description, etc. Recent outburst of multimedia content on the Internet and as personal collections has raised the demand for auto-annotation methods; due to which this has become an active area of research.

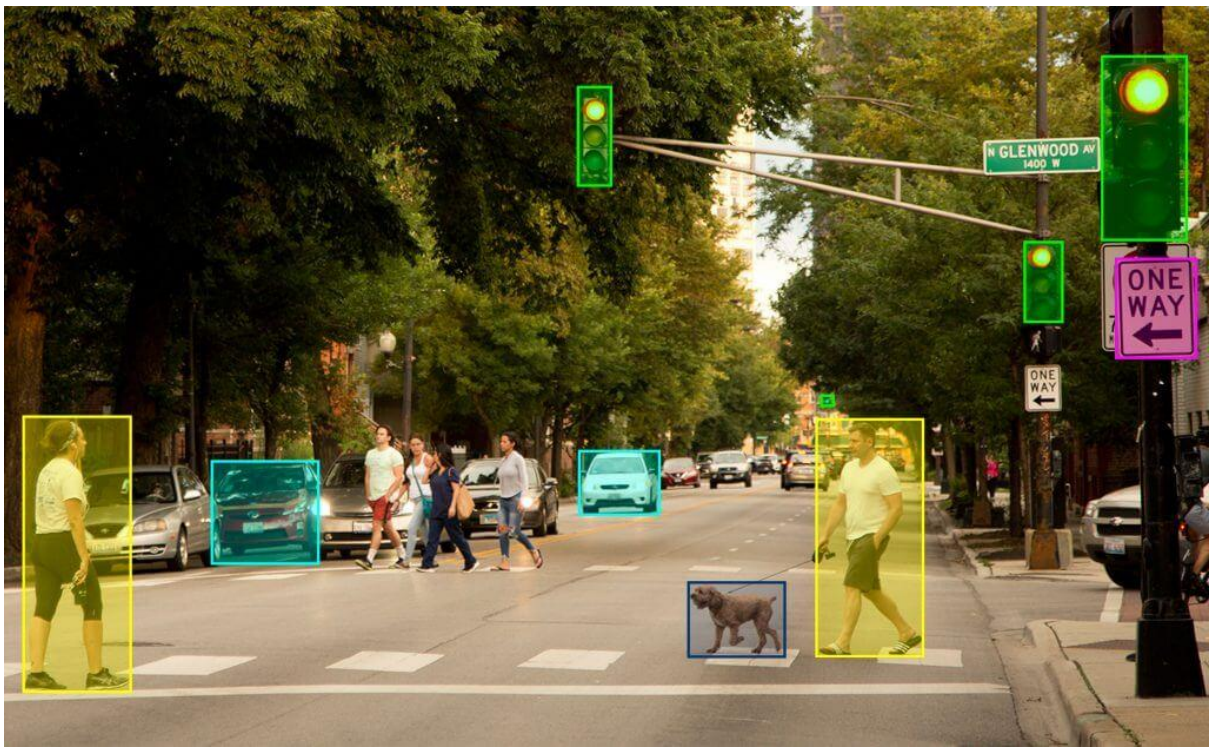
A self-driving car is one of the best examples where annotated images have been used to train the model working into the auto functioning of such vehicles. The autonomous vehicles recognize the object on the street and move accordingly to avoid any crash. It is like making a child learn about new things with varied data sets so that it can be recognized easily when used in autonomous vehicles to drive on the road into real-world.

Healthcare is another important filed where annotated medical imaging used to develop such machines that can detect various types of maladies including life-threatening diseases like cancer at the initial stage of developments with better accuracy. Medical images like X-ray, MRI and CT Scan reports are annotated manually by humans that are used to train the machine learning to detect such diseases without the help of humans.

Apart from autonomous vehicle driving, there are many other fields, like healthcare, retail, agriculture, security surveillance and sports analytics are other areas image annotation is playing an important role to make the images easily recognizable for machines allowing machine learning developers to build a right model at affordable pricing.

The main purpose and objective of image annotation are well-described here with a various set of examples showing the use and purpose of image annotation in machine learning and AI. The right applications of image annotation are only possible when annotations are accurate so that models can get accurate data sets to learn and give the right predictions.


A
Annotation on T2FS
(Xleft, Ytop)
Annotation of lymph nodes
(Xright, Ybottom)
Lymph node metastasis
B
Annotation on DWI
Annotation of lymph nodes


N GLENWOOD AV
1400 W
ONE WAY
ONE WAY

Research Papers Read

Fast Image Tagging, M Chen, A Zheng, K Weinberger, International conference on machine learning, 2013

TagProp: Discriminative metric learning in nearest neighbor models for image auto-annotation, Matthieu Guillaumin ; Thomas Mensink ; Jakob Verbeek ; Cordelia Schmid, 2009 IEEE 12th International Conference on Computer Vision

A Hybrid Probabilistic Model for Unified Collaborative and Content-Based Image Tagging, Ning Zhou, William K. Cheung, Guoping Qiu, and Xiangyang Xue, IEEE Transactions on Pattern Analysis and Machine Intelligence (Volume: 33 , Issue: 7 , July 2011)

An efficient color representation for image retrieval, Yining Deng ; B.S. Manjunath ; C. Kenney ; M.S. Moore ; H. Shin, IEEE Transactions on Image Processing (Volume: 10 , Issue: 1 , Jan 2001)

Fast Zero-Shot Image Tagging, Yang Zhang ; Boqing Gong ; Mubarak Shah , 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)

Deep Collaborative Embedding for Social Image Understanding, Zechao Li ; Jinhui Tang ; Tao Mei, IEEE Transactions on Pattern Analysis and Machine Intelligence (Early Access)