PROFILER
Instagram account classification using Deep Learning and NLP

Ashwin Unnikrishnan, Ashwini Shaktivel Kumar, Shamekh Mohammad Siddiqui
Khoury College of Computer Sciences, Northeastern University
unnikrishnan.a@northeastern.edu, kumar.ashwi@northeastern.edu, siddiqui.sh@northeastern.edu

Abstract

With more than a billion Instagram accounts, and accounts covering a wide range of contents like food, travel, nature, sports, etc it gets hard to determine what type of account it is. Even though Instagram allows users to select an account type, most of the time the account might have content from other categories as well. Classifying the accounts based on all the content posted in the profile might make it useful for users to understand if it matches with the preferences they have and whether they want to follow the account. The categorization will also help the content creators to keep a check on themselves to see if they are going off track from what they want the account to look like. Hence, we propose a pipeline involving web scraping, object detection (semantic segmentation), and category label classification divided into two phases. We are selecting 3 different object detection models and 3 different classification models and evaluating the accuracy of these models.

I. Introduction

Social network services (SNS), such as Facebook and Instagram, have fascinated a lot of the population and this has led to increased usages of such services. People share contents of wide varieties from day to day life, like the food they eat, places they travel, sports they see or play, etc, in such SNS platforms. This brings potential commercial opportunities to commercial companies as well as aspiring content creators to follow established people and make their own content. It would be helpful for other users (commercial companies or aspiring creators) to understand the contents that are posted by a particular user in a button's click. SNS platforms allow features where one can label the type of account it is, but sometimes this information might not match with the type of account. Going through 100s of images just to check if the account actually posts the contents as per the label selected is a hectic job. And even the content creators sometimes might go off the track and post stuff that doesn't match with the kind of account they want to create. It's always better if someone can keep monitoring the posts and see if the content being posted is on track with what kind of profile they are building. This will help advertising companies as well when selecting profiles with related interests to advertise their products. The Instagram platform supports images, videos, reels. There is no proper way of analyzing the type of content with captions. So to analyze the posts and a profile we can use the images and the objects present in those images.

We have fundamental tasks like image classification that will understand the entire details of an image considering the image as a whole. Image classification classifies the image into a particular category. When we do a traditional image classification, it concentrates on one major object appearing in the image, in such instances we might miss out on other details that are part of the background or are present in smaller quantities. Classifying the images and then summarizing the profile based on different classes of images present in the profile might sometimes give a wrong result. Object detection approaches are used to more realistically understand and analyze the entire profile, it helps detect multiple objects in the same image.

In this paper, we propose a framework that uses object detection to understand the objects present in an image and combine all the objects and the frequency of these objects across the images in the profile. We store this list of objects detected across the profile as a document and a Natural Language Processing (NLP) model is used to label the account with matching types (an account might be classified as food and sports if it has both food and sports content with some threshold 50-50). This will give a better idea for the user, rather than classifying it as 1 label.

To test our idea and method, we collected image dataset from COCO test dataset to evaluate the accuracy of our object detection models and simulated Instagram accounts to train our classification model. The main contributions of this paper are listed as follows: an

- A high-level profile classification framework that combines an object detection model and an NLP model.
- Simulating an Instagram profile dataset, using the most seen objects in each category.
- Training and validation for the proposed framework.

II. Background

Dataset:

There are several great object detection and image segmentation data sets available on the internet, amongst which we elected to work with the following:

- **MS COCO dataset:** The COCO dataset sponsored by Microsoft comprises over 330K images with over 200K images labeled (used as training and validation data-set). The dataset has 80 labeled object categories. The COCO team has collaborated with Fifty-One, an open-source data-set visualization, and evaluation tool. This can be used to evaluate the subject model's performance.
- **PASCAL VOC dataset:** The PASCAL VOC project provides standardized image data sets and annotations for object detection, allowing for quick evaluation and comparison of various models. The most recent version of the data collection, which dates back to 2012, contains around 11.5K photos and 20 object class labels.
- **Open Images dataset:** The OID sponsored and collected by Google is a data set of millions of images with rich object labels and segmentation annotations. The dataset is annotated with over 19K image-level multi-classes and 600 object localization categories.

Object Detection:

Our primary goal, as stated in the introduction, was to identify significant objects in a given image that could subsequently be used to categorize a series of photographs in the social media profile. Our investigations are built on the foundation of cutting-edge object detection techniques. To obtain more information about the prominence of the object in the image, we used Semantic Segmentation on the photos in parallel.

Semantic Segmentation:

Semantic Segmentation is a computer vision task that assigns each pixel a class to which it belongs. 'Semantic' means that the model uses language to convey meaning, reference, or truth. It puts parts of an image that belong to the same class together. An outline of the object and a list of object names represent the semantic label assignment of objects found in the image, such as 'human,' 'potted plant,' and 'traffic light.' Once all pixels of an object belonging to the same class are identified, it is colored uniquely. In essence, the image's categorized areas are overlaid with a segmentation mask that defines the class's color. The output contains a mask in which the input pixel value (ranging from 0 to 255) is translated into a normalized label value (0, 1, 2,... n).

The following images Figure 1 and 2, form an example of Semantic Segmentation, the output image categorizing pixels of a bus in *pastel green* color, and the people standing in *blue* color. The model tends to return color mapping and the list of objects present which we then use towards our end model application.

A deep CNN is generally preferred, wherein many convolutional layers are stacked with tuned padding and weights

Figure 1: Input Image

.

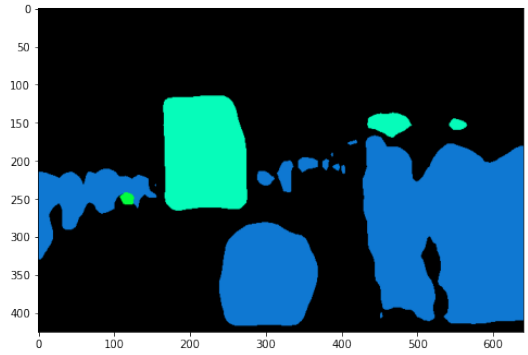


Figure 2: Semantic Segmentation Output Image

.

to get a feature mapping of the pixel semantics. The earlier layers tend to learn the pattern seen in the pixels, while the later layers generate feature mappings. Many breakthroughs are made with an increase in the number of feature channels through the levels of CNN to prevent any loss of image pixel information. However, this increased the overall computational requirement. To prevent the same and achieve high-quality semantic mappings, the Semantic Segmentation models are briefly divided into two stages:

- **Encoder:** A pre-trained classification neural network like Residual Neural Network or MobileNet trained on large-scale datasets like MS COCO and PASCAL-VOC.
- **Decoder:** A network recovering spatial information and overlaying discriminative features (segmentation mask) extracted by the encoder onto the pixel space of the input image.

In the encoder stage *down-samples* the spatial resolution of the given image to offload the requirement of additional computational layers, which is then used by the decoder to generate feature masks that are used to discriminate pixels of an image to different classes. The developed feature map is then *up-sampled* to the input image resolution.

We have experimented on the following models comprising of modified encoder and decoder stages to achieve different levels of precision in object detection and segmentation.

Deeplab v3 by Google AI:

Deeplab was initially introduced by the Google AI team, Chen et. al. in the year 2017 for advanced semantic image segmentation, based on Deep Convolutional Neural Networks backbone. The Deeplab model was then revised three times, following which the current widely used version is the Deeplab v3. The Deeplab model also follows the encoder-decoder architecture as seen in a variety of Semantic Segmentation models at present.

The latest version adds a slightly tuned decoder module, which refines the segmentation results along the boundaries of identified pixel groups. The up-sampling unit in the decoder now implements **atrous convolutions**.

The convolutional layers of a neural network are generally defined based on the following parameters:

- **Kernel Size:** A kernel or convolution matrix is an nxn matrix used for edge detection, pixel categorization, image enhancement, etc. wherein the kernel is convoluted with the image pixels. The common kernel size is 3x3 for 2D image processing.
- **Padding:** As the term suggests, an additional padding layer of pixels is added around the input image to prevent loss of information around the image edges.
- **Stride:** Step-size of the kernel while convoluting through a group of pixels, generally used while down-sampling.

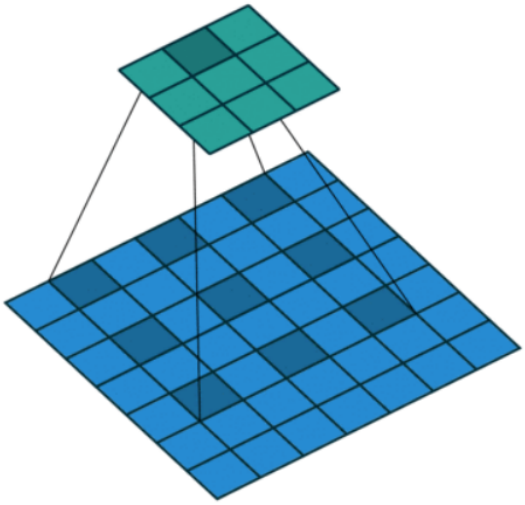


Figure 3: Atrous Convolution Dilation Rate

.

Atrous Convolutions: A *dilation rate* parameter is added to the kernel used to design and fine-tune the convolutional layers. This defines the spacings between each matrix unit. For example, a simple 3x3 kernel described earlier with a dilation rate of 1 unit, now has a 5x5 matrix field of view. The matrix convolutes with 9 image pixels (as in 3x3 matrix), with a wider view of 5x5 matrix as shown in Figure 3. [convo]

The atrous convolutions are used as a replacement for multiple convolutions or large kernels, for dense computer vision prediction tasks.

Chen et. al. have proposed an Atrous Spatial Pyramid Pooling (ASPP) technique, for capturing useful image context on multiple scales. The ASPP traverses through the convolutional feature layer with multiple atrous dilation rates, thus forming a pyramid of kernel operations with gradually increasing dilation, Figure 4. The final fused field-of-view thus captures image information at multiple scales.

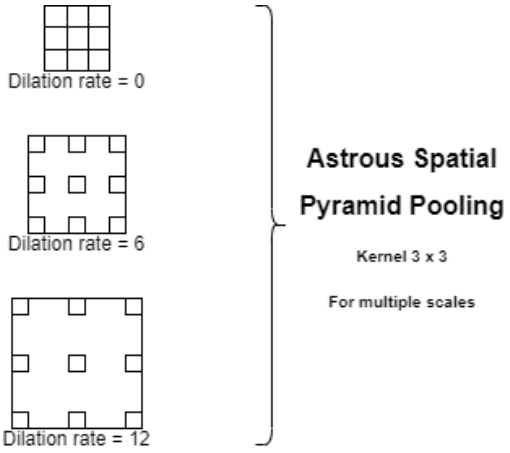


Figure 4: ASPP layering

Conditional Random Field: The Deeplab model then executes a fully connected Conditional Random Field to improve localization accuracy, which is weighed down during the down-sampling performed by the encoder. The CRF is used to maximize relational potential between similar pixels of an object class providing a belief map covering pixels believed to be part of an object in the image.

We implemented the Deeplab v3 model with a backbone of ResNet 50, ResNet 101, and MobileNet architectures as a part of our experimentation.

Fully Convolutional Network:

A fully Convolutional Network is one of the early models developed for semantic segmentation. Classical convolutional networks process and produce labels for only images of a fixed size. The FCN is an extension to the same, by having the capability of processing arbitrarily sized input images.

Traditional convolutional networks employ dense layers wherein an image of only a particular size is taken as input and then downsized to achieve classification. There is no flexibility in taking input different image sizes. Fully Convolutional networks use the concept of upsampling via deconvolution which is to increase the size of the output so we can get pixel-wise accuracy for semantic segmentation instead of just one label for classification with the help of 1x1 convolutions which are akin to the function of dense layers.

Therefore we get a mapping of each classification for each pixel at the spatial position. Using ResNet50 as a backbone, we predict up to 20 classes for each pixel in the semantic segmentation. We build this using two-dimensional convolution layers and a mechanism to prevent overfitting called regularization.

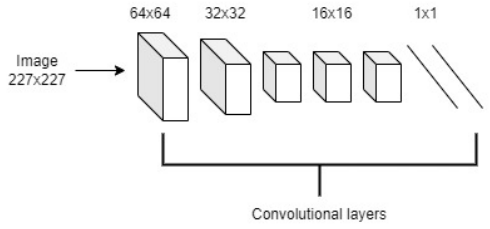


Figure 5: Dense layers providing one classification

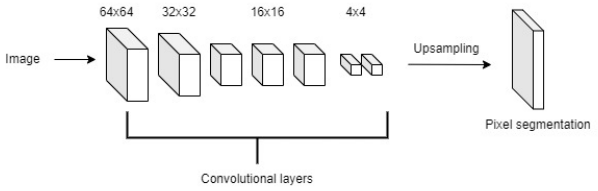


Figure 6: Upsampling and mapping classification to each pixel for semantic segmentation

In figure 5, the traditional network with dense layers downsizes the image and gives one classification output. Figure 6 shows how an FCN can achieve semantic segmentation using pixel-wise mapping of each classification using upsampling.

ResNet Architecture:

In the traditional convolutional network used for images, we try to use additional layers to improve accuracy. However, increasing layers gives a rise to complexity and training difficulty thereby decreasing accuracy. RESNET uses the concept of skipping layers so that training time is less, it achieves this using a skip connection in residual blocks. Due to this, the gradient can flow through the shortcut path.

$$H(x) = f(wx + b)$$
$$H(x) = f(x) + x$$

Estimating H(x) output in a traditional network would require stacked layers whereas RESNET uses a residual function f(x) to estimate H(x). The initial ResNet architecture uses 34 layers residual network i.e it takes the 34 layers plain network and uses shortcuts to convert it into the residual network. We will be using ResNet50 which is 50 layers deep with DeepLabV3 and FCN backbones.

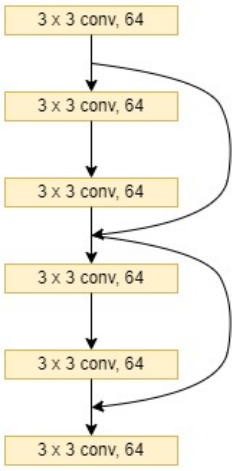


Figure 7: Skipping layers using residual function

Residual mapping taking place every two layers. This reduces the number of stacked layers through which x needs to go to estimate H(x).

MobileNet Architecture:

MobileNet is Tensorflow's exclusive mobile and embedded system-specific computer vision model. Since it works on restricted computational resources (latency and size), the aim of its development is to have a lightweight deep neural network. MobileNet is based on **depth-wise separable convolutions**, under which the model observes a significant trimming of parameters which eventually makes it a lightweight model.

Depth-wise separable convolutions: The MobileNet structure has a modified width multiplier and resolution multiplier hyper-parameters as discussed by Howard et. al. The standard convolution structure is factorized into depth-wise separable convolution with a combination of:

- **Depth-wise convolution:** This layer applies a unit filter to each input channel. The filter adds up to the efficiency of the overall model, however, it does not cater to combining the input features to create new features. This deficiency is then solved by the point-wise convolution layer.
- **Point-wise convolution:** The point-wise convolution layer generates new features by linearly combining outputs from the filters in the depth-wise layer.

This layer combination of depth-wise and point-wise convolution is addressed as Depth-wise separable convolutions. A standard MobileNet has 28 layers while counting depth-wise and point-wise separately. We have implemented MobileNet as the backbone of one of our Deeplab models to utilize its lightweight and high-efficiency peculiarity.

Multinomial Naive Bayes

This classification technique that follows the probabilistic naïve Bayes method is used primarily for categorical text

data. Naive Bayes is given by the equation:

$$P(A|B) = P(A) * P(B|A)/P(B)$$

Where P(B) and P(A) is the prior probability of B and class A respectively. $P(B|A)$ is the probability of occurrence of B given A.

$$p(f1, f2, .., fn|c) = \prod_{i=1}^{n} p(f_i|c)$$

Here each feature is independent of the other, and the model uses a high probability threshold with discrete features in order to give the category based on the text. This algorithm has a similar simple implementation and accuracy rate like naïve Bayes however multinomial nb takes the count of the word into consideration.

Logistic Regression

This classification technique performs predictive analysis using a logistic function logit(p) as defined below.

$$logit(p) = log(\frac{p}{1-p})$$

This model describes the relationship between data using the probability of an outcome, thereby giving the category of the sample data. The log function applies a nonlinear transformation such that the linear mapping between input and output variables is transformed.

$$z = wx + b$$

Z is passed through the activation function which is sigmoid function in this case $\frac{1}{1+e^{-x}}$. Based on the value of Z, we achieve binary classification, with label 1 if Z tends to infinity and 0 if it tends to negative infinity.

Linear SVC

A linear support vector classification model creates a hyperplane to categorize data points using a decision boundary. Linear SVC is a supervised machine learning algorithm that is used for binary as well as multiclass classification, regression, and noise detection. This model is usually used when we encounter input with higher dimensionality. Different kernels are used based on the type of input/dimensionality which gives us different results.

$$H(x) = w(x)^T x + b(x)$$
$$H(X) = \sum_{i=1}^{n} w_i(x)x_i + b(x)$$

Bag of Words

This is a method used for natural language processing to generate a feature set from textual data. It uses the occurrence/count of words in no particular order to create a 'bag' of words. Each feature set contains boolean values for words for whether a particular word is present or absent. However, with the increase in the size of the corpus, we may get a vector with a very large range that can reduce performance hence it is essential to preprocess data to remove punctuations, articles and create a group of words in the form of bigrams or trigrams.

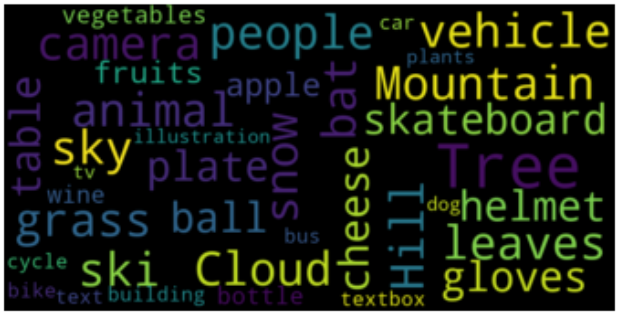


Figure 8: Word cloud of labels

TF-IDF

To counter the pitfalls of just counting words in a document and risk extracting meaningless information based on high occurrence, TF-IDF uses term frequency and Inverse document frequency to check rarity of words across all documents in addition to finding word frequencies. The rarity of the word across all other documents if high essentially means that the word is of significance for a document. The weightage to words that are rare in the entire corpus is given importance over weightage to common words.

III. Related Work

With the rapid increase of people using Social network services, many parallel applications and solutions have been developed in the recent past. This includes processing and gaining valuable insights from text, images, relational links between users, location, and usage statistics. The retrieved data is then used towards up-scaling and improvement of the user interface. We found the following relevant work which guided us through the planning stages of the project:

Metaeyes:

Metaeyes is an online subscription based visual-monitoring tool for Instagram. It analyses images without the requirement of hashtag or natural language captions to identify objects, scenes, content type of the given images. The application is largely based on image recognition and machine learning methods. It detects wide set of object categories extending from nature to vehicles and describes the context of a particular image. Subscribers receive this analytic treasure in the form of summary and granular reports.

Image captioning for targeting influencers on SNS:

Xiaohong Yu et. al. have disclosed an image classification and captioning method based on a combination of computer

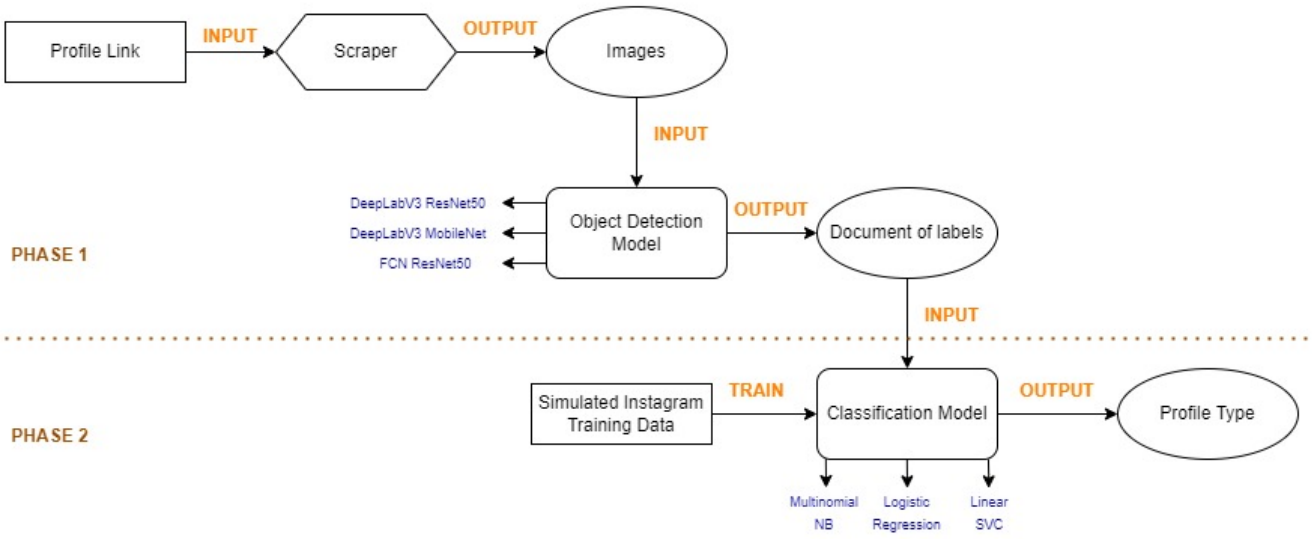


Figure 9: High level Architecture of Profiler

vision and BERT-based natural language processing methods. It is observed that images from social networking sites contain a large diversity of objects, belonging to complex scenarios. This information is retrieved by the team by synergizing image captioning model and natural language processing model together.

The first stage comprises an image captioning model, which understands the image and provides scene-based captions, which is then processed by the second stage of the application, which is the NLP-BERT model. The second stage classifies generated text descriptions of the images. A Tensorflow-based visual attention framework with inception v3 model as backbone and BERT model is used in Stage 1 and 2 respectively.

IV. Project Implementation

Object Detection and Classification

As shown in Figure 9 the Profiler can be divided into two phases. Phase 1 expects the link of the profile to be classified. The scraper will pull all the images from that profile and send images one at a time to the object detection model, where the object detection model detects objects in each image and sums up the entire profile as a document of labels(objects found across all the images of the profile). In phase 2, we train a classification model based on simulated Instagram training data. This trained classification model is provided with the document of labels from phase 1 as input and the classification model outputs the probability of the profile belonging to each of the 7 given profile types. Based on a threshold we output all the profile types whose probability is more than the given threshold.

Object Detection

Figure 10 illustrates the Phase 1 architecture of our implementation. We can divide phase 1 into two blocks. In the first block, we have a scraper, that takes a profile as input, scrapes through the profile and collects all the images in the profile, and feeds this into the Object Detection model used. The object detection block is the second block that identifies the objects in each image and combines all these into a document of labels. With this step, we have converted an Instagram profile into a document of labels format.

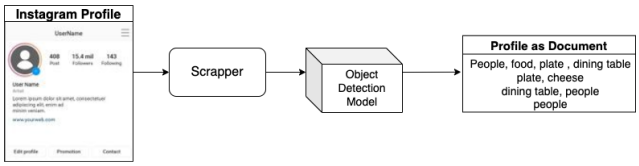


Figure 10: Conversion of an Instagram Profile to Document of labels

Models trained on open source images, pascal and COCO datasets were used in the object detection phase. We tested the accuracy of various models and based on the results identified chose 3 models for our use case. Based on the initial observations, we used DeepLabV3_Resnet50, DeeplabV3_MobileNet, FCN_ResNet models trained on the COCO dataset and fine tuned to Pascal dataset. We compared these 3 models based on the objects identified and checked for its accuracy.

Simulated Instagram Dataset

We identified the most frequent objects that are seen for each of the 7 different categories of Instagram profiles. We created 100 test data for each category by creating a document of labels using this simulation.

Profile Classification

Using the simulated Instagram dataset we are training 3 different text classification models such as Multinomial Naive

Profile	Type
people grass Mountain Cloud animal Tree Mounta...	Nature
camera grass camera grass camera tree camera v...	Wildlife
table cheese people building apple fruits buil...	Food
bike building car cycle people bus bus car cyc...	Vehicle
animal textbox text animal animal animal peopl...	Meme
cycle cycle people people dog building people ...	Travel
ski people bat skateboard bat skateboard ski b...	Sports

Figure 11: Example of document to label for each type of account

Bayes, Linear Regression, and Linear Support Vector Classifier and comparing the performance of these 3 models based on our requirements. The classification model takes the output of phase 1 i.e. the document of labels and based on the training gives the probability of the document belonging to each of the 7 labels we trained it for. We are feeding this output as an input to a threshold checker, it outputs all the labels above the threshold of 0.7.

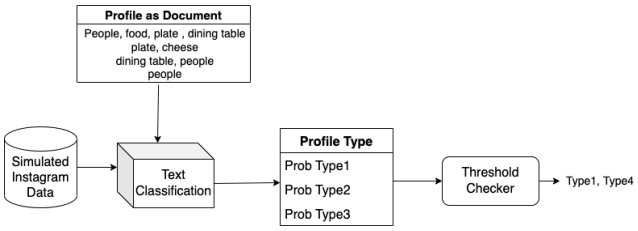


Figure 12: Document Classification and providing type of account

Instead of classifying each document into one, we are identifying the similarity of the document of labels to each of the 7 and printing the categories that have a probability of more than 0.7. This covers the scenario where a user might be posting vehicle and food-related content, so instead of just showing one of the two we are marking it with both the categories so that it gives a proper idea of the type of content being posted in the account.

V. Experiments and Results

We discuss the experiments conducted and the results in this section. During the initial experimentation to identify the best object detection models to cover our use case. Our main goal was to find a model that could detect as many objects as possible. Open Image Dataset (OID) trained model could detect up to 600 objects, but initial trial runs with sample images we could see the detection rates were very less, moreover the accuracy of such models were very low. We can see in Figure 13, a dog is getting detected as limousine. So instead of taking a big leap we tried using models trained to

Figure 13: Dog getting classified as limousine

detect 20 objects i.e. the Pascal dataset, that would give us better accuracy.

To collect images from Instagram to validate our Object Detection model and to train the classification model we used python libraries to scrape through the Instagram and fetch images of the profiles provided. Due to the Instagram policies we were unable to proceed with the same and decided to use COCO test and validation dataset to test our Object Detection model accuracy and created Instagram training dataset by simulation.

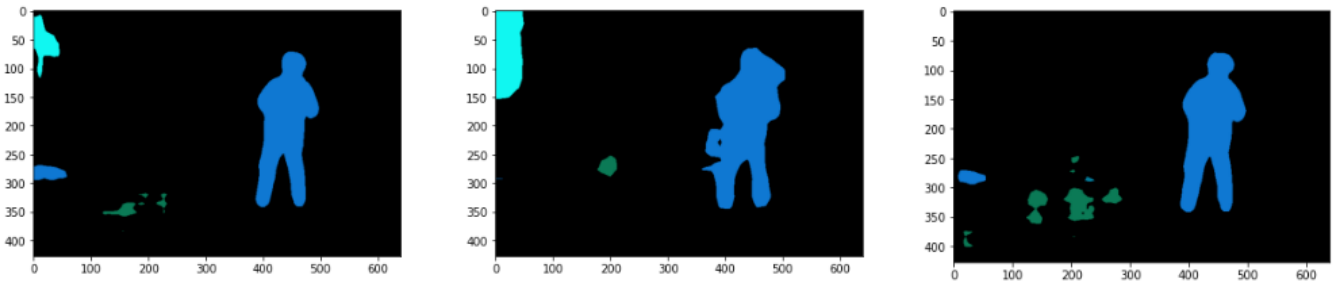
Phase I: Object Detecion

As discussed in the background section, we implemented Deeplab and FCN model architectures to obtain semantic segmentation and list of objects in an image. We integrated pre-trained object detection models by the Pytorch team into our application pipeline. The models are trained on COCO dataset (dataset with over 80 object categories) maintained the same height and width of the model tensors, fine-tuned to output classes according to the PASCAL-VOC (dataset with 20 object classes) to achieve high accuracy.

The following Figure 14 is a test-image randomly given as an input to the phase I models, wherein we see a man, a dining table, a few potted plants and plate. This stands as a point of reference while comparing with fellow models:

Figure 14: Original Test Image

.



A.) DeepLab v3 ResNet50 B.) DeepLab v3 MobileNet C.) FCN ResNet50

Figure 15: Object Label Results

The input images to the pre-trained models are normalized in a standard manner comprising of mini-batches of three RGB images with attributes (N, 3, H, W), wherein:

- **N:** Denotes number of input images.
- **H:** Height of the images with at least 225 pixel dimension.
- **W:** Width of the images with at least 225 pixel dimension.

Pre-processing of the loaded images includes normalization in the range 0 and 1, using mean and standard deviation masks set to mean = [0.485, 0.456, 0.406] and Standard Deviation = [0.229, 0.224, 0.225]. The output of the models encompasses unnormalized probabilities of object class predictions at all image parts. We then retrieve the maximum probability of each class object using the *argmax* method. The semantic segmentation plot of objects presents a segmentation mask overlaid over the identified object locations based on a normalized color pallete.

As seen in Figure 15, we observed a noticeable difference in the attained results, the DeepLab models seem to miss 'bottle' objects, while the FCN model misses out 'potted plants' in the image. The figure comprises a sample of 15 image results, used to compare the results of the models. This list of objects corresponding to each image is then documented and taken as an input to phase II.

We consolidated FiftyOne to our project pipeline, an open-source COCO dataset visualization and access tool to measure the accuracy of our models. The tool aided us to retrieve label annotations from the validation dataset, used to compare and improve our deep learning models. FiftyOne can be imported to the machine learning system by installing it as a background support framework.

Figure 16 is a sample list of images uniquely identified using *sample id* list of objects originally labelled in the image by the dataset developers. This stands as a strong point of reference while metric calculation. Since the retrieved annotations are from the COCO dataset, we had to negate class labels that were not a part of the PASCAL VOC to get appropriate metrics.

```
61b7c8943829e4b68d05c2b4 : ['cell phone', 'clock', 'person', 'person']
61b7c8943829e4b68d05c2b5 : ['train', 'person', 'person', 'person', 'person', 'person', 'person']
61b7c8943829e4b68d05c2b6 : ['sandwich', 'bowl']
61b7c8943829e4b68d05c2b7 : ['person', 'surfboard']
61b7c8943829e4b68d05c362 : ['laptop', 'mouse', 'keyboard', 'tv', 'mouse']
61b7c8943829e4b68d05c363 : ['car', 'car', 'car', 'car', 'truck', 'car', 'car', 'car']
```

Figure 16: Labels retrieved from annotations

.

The following Figure 17 discloses observed mean Intersection over Union of the models and a pixel accuracy measure compared with the COCO validation dataset.

Object Detection Model	mIOU	Per Pixel Accuracy
DeepLab ResNet 50	66.4	92.4
DeepLab ResNet 101	67.4	92.4
DeepLab MobileNet	60.3	91.2
FCN Resnet 50	60.5	91.4

Figure 17: IOU and Accuracy comparisons

.

Phase II: Profile Classification

In the second phase we train multinomial Naive Bayes, logistic regression, and support vector classifier using the simulated Instagram dataset to identify the types of content the provided profile contains. Phase 2 takes a document of labels and outputs the most similar categories the account matches with.

Initially, we tested the classification models to classify each profile(document of labels) into one category.

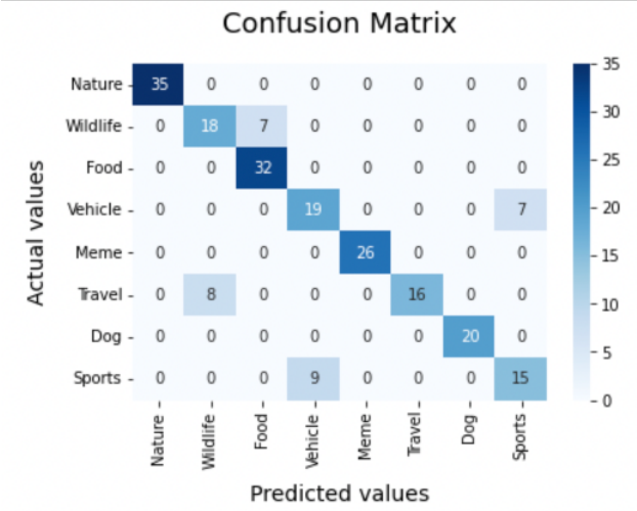


Figure 18: Confusion Matrix for SVM and Linear Regression

.

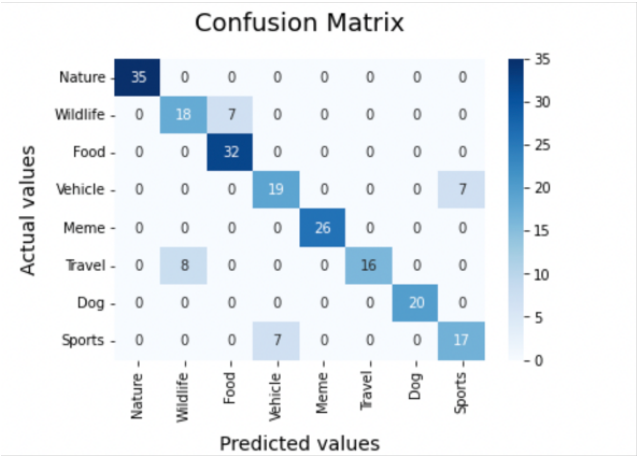


Figure 19: Confusion Matrix Multinomial NB

.

We can see from Figure 18 and 19 that the Multinomial NB has much better prediction as it did not misclassify sports as the vehicle. So we selected multinomial NB as our Profile classification model. We calculated the precision, recall, and F1-score of the Multinomial NB. From Figure 19 and 20 we can see that when we are classifying an account into one category, we are seeing a loss of accuracy. Upon further exploration, we found that a profile might contain images of two different categories in almost similar frequencies ex: travel accounts might contain wildlife-related images in equal amounts, due to which sometimes we might get the wrong classification. So instead of classifying into one category, we take the predicted probability of an account belonging to a category and output all the categories that fall above the threshold of 0.7. As per the example provided above of misclassifying travel as wildlife, the output in such instances would be travel, wildlife both, instead of either one of them. This provides users more insight into the contents of an account, by providing multiple categories where the majority of the images map to.

	precision	recall	f1-score	support
Dog	1.00	1.00	1.00	35
Food	0.69	0.72	0.71	25
Meme	0.82	1.00	0.90	32
Nature	0.68	0.73	0.70	26
Sports	1.00	1.00	1.00	26
Travel	1.00	0.67	0.80	24
Vehicle	1.00	1.00	1.00	20
Wildlife	0.68	0.62	0.65	24
accuracy			0.85	212
macro avg	0.86	0.84	0.85	212
weighted avg	0.86	0.85	0.85	212

Figure 20: Precision, Recall and F1-score of Multinomial NB

.

VI. Conclusion

In this paper, we provide a method to understand the type of content a particular Instagram profile consists of given the Instagram ID. In the first phase, we are using object detection models to convert an Instagram profile to a document of labels based on the objects detected across all the images in that profile. In the second phase, we are using text classification to identify the matching categories of the profile. According to our experiment, multinomial naive Bayes was able to classify better with an f1 score of 0.85 for single-label classification. In this paper, we only selected 7 categories namely Sports, Vehicle, Travel, Food, Nature, Meme, Wildlife.

VII. Future Work

For object detection we had used models trained on Pascal Dataset that covers only 20 objects, in the future, we can use models trained with COCO dataset that covers overall 90 objects. To make better predictions and accuracy we can add labels that are not covered with the COCO dataset by implementing a new object detection model using transfer learning and train to identify objects like the sky, building, bridge, monument, etc. We can use the same methodology

to categorize users across different Social Networking Services(SNS), by just changing the scraper block according to the new SNS. We can extend even the object detection part where we identify the most prominent objects in the image and only store that in the document of labels, rather than storing objects that are not prominent.

VIII. References

[1] Evgeniou, Theodoros Pontil, Massimiliano. (2001). Support Vector Machines: Theory and Applications. 2049. 249-257. 10.1007/3-540-44673-7-12.

[2] Long, Jonathan Shelhamer, Evan Darrell, Trevor. (2015). Fully convolutional networks for semantic segmentation. 3431-3440. 10.1109/CVPR.2015.7298965.

[3] X. Yu, Y. Ahn and J. Jeong, "High-level Image Classification by Synergizing Image Captioning with BERT," 2021 International Conference on Information and Communication Technology Convergence (ICTC), 2021, pp. 1686-1690, doi: 10.1109/ICTC52510.2021.9620954.

[4] Howard, Andrew G., et al. "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications." ArXiv Preprint ArXiv:1704.04861, 2017.

[5] L. -C. Chen, G. Papandreou, I. Kokkinos, K. Murphy and A. L. Yuille, "DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 40, no. 4, pp. 834-848, 1 April 2018, doi: 10.1109/TPAMI.2017.2699184.

[6] Chen, L. C., Papandreou, G., Schroff, F., Adam, H. (2017). Rethinking atrous convolution for semantic image segmentation. arXiv preprint arXiv:1706.05587.

[7] Chen, L. C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A. L. (2017). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE transactions on pattern analysis and machine intelligence, 40(4), 834-848.

[8] Liang-Chieh Chen and Yukun Zhu, "Semantic Image Segmentation with DeepLab in TensorFlow", googleaiblog (blog), 12th March, 2018, https://ai.googleblog.com/2018/03/semantic-image-segmentation-with.html

[9] Pytorch team, "DEEPLAB", https://pytorch.org/hub/pytorch-vision-deeplabv3-resnet101/

[10] Pytorch Team, "FCN",https://pytorch.org/hub/pytorch-vision-fcn-resnet101/

[11] Paul-Louis Pröve, "An Introduction to different Types of Convolutions in Deep Learning", towardsdatasscience (blog) , 22nd July, 2017, https://towardsdatascience.com/types-of-convolutions-in-deep-learning-717013397f4d

[12] Lin, Tsung-Yi, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár and C. Lawrence Zitnick. "Microsoft COCO: Common Objects in Context." ECCV (2014).

[13] He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770-778. 2016.

[14] Zhang, Yin Jin, Rong Zhou, Zhi-Hua. (2010). Understanding bag-of-words model: A statistical framework. International Journal of Machine Learning and Cybernetics. 1. 43-52. 10.1007/s13042-010-0001-0.