Semantic Segmentation (depth effect in photographs and self driving cars)

Nikita Goswami
Master of Science – Data Science
Univeristy at Buffalo
email : nikitago@buffalo.edu

Vimal Kumarasamy
Master of Science – Data Science
Univeristy at Buffalo
email : vimalkum@buffalo.edu

Semantic Segmentation

***Abstract*— Deep Learning has pushed the limits of what was possible in the domain of Digital Image Processing. However, that is not to say that the traditional computer vision techniques which had been undergoing progressive development in years prior to the rise of DL have become obsolete. There is large consent that successful training of deep networks requires many thousand annotated training samples. In this paper, we present a network and training strategy that relies on the strong use of transfer learning and data augmentation to use the available annotated samples more efficiently and requires less time and computation to train the network. The architecture consists of a contracting path to capture context and a symmetric expanding path that enables precise localization. We show that such a network can be trained end-to-end from very few images and outperforms the prior best method (a sliding-window convolutional network) for the image segmentation problem. Moreover, the network is very fast. Segmentation of any sized image set takes less than a second on a latest GPU. The full implementation and the trained networks are available at https://github.com/Nikita018/Image-Segmentation.**

Keywords— Computer vision, Deep Learning, Image segmentation, Transfer Learning, MobileNet, Transposed Convolutions, U-Net, Image Augmentation, Convolution

I. INTRODUCTION

Deep Learning (DL) is used in the domain of digital image processing to solve difficult problems (e.g. image colorization, classification, segmentation and detection). DL methods such as Convolutional Neural Networks (CNNs) mostly improve prediction performance using big data and plentiful computing resources and have pushed the boundaries of what was possible. Problems which were assumed to be unsolvable are now being solved with super-human accuracy. Image classification is a prime example of this. Since being reignited by Krizhevsky, Sutskever and Hinton in 2012 [1], DL has dominated the domain ever since due to a substantially better performance compared to traditional methods .

Semantic image segmentation, also called pixel-level classification, is the task of clustering parts of image together which belong to the same object class (Thoma 2016). Two other main image tasks are image level classification and detection. Classification means treating each image as an identical category. Detection refers to object localization and recognition. Image segmentation can be treated as pixel-level prediction because it classifies each pixel into its category. Moreover, there is a task named instance segmentation which joints detection and segmentation together. More details can refer to literature (Lin et al. 2014; Li et al. 2017a). Semantic image segmentation has multiple applications, such as detecting road signs (Maldonado-Bascon et al. 2007), colon crypts segmentation (Cohen et al. 2015), land use and land cover classification (Huang et al. 2002). Also, it is widely used in medicine field, such as detecting brains and tumors (Moon et al. 2002), and detecting and tracking medical instruments in operations (Wei et al. 1997). Several applications of segmentation in medicine are listed in Dzung et al. (1999). In Advanced Driver Assistance Systems (ADAS) or self driving car area, scene parsing is of great significance and it heavily relies on semantic image segmentation (Fritsch et al. 2013; Menze and Geiger 2015; Cordts et al. 2016). Since the re-rising of DNN (Deep Neural Network), the segmentation accuracy has been significantly enhanced. In general, the methods before DNN are called traditional method. we also comply with this convention in the following sections. Traditional segmentation methods are briefly reviewed in this paper. More importantly, it will focus on the recent progress made by adopting DNN and organize them in several aspects. Moreover, we have carried out a survey on datasets of image segmentation and evaluation metrics

In the last few years, deep convolutional networks have outperformed the state of the art in many visual recognition tasks, e.g. [1 , 3]. While convolutional networks have already existed for a long time [2], their success was limited due to the size of the available training sets and the size of the considered networks. The breakthrough by Krizhevsky et al. [1] was due to supervised training of a large network with 8 layers and millions of parameters on the ImageNet dataset with 1 million training images. Since then, even larger and deeper networks have been trained [3].

The typical use of convolutional networks is on classification tasks, where the output to an image is a single class label. However, in many visual tasks the desired output should include localization, i.e., a class label is supposed to be assigned to each pixel. Moreover, thousands of training images are usually beyond reach in most tasks. Hence, Ciresan et al. [1] trained a network in a sliding-window setup to predict the class label of each pixel by providing a local region (patch) around that pixel as input. First, this network can localize. Secondly, the training data in terms of patches is much larger than the number of training images. The resulting network won the EM segmentation challenge at ISBI 2012 by a large margin.

II. DEEP LEARNING

A. WHAT IS DEEP LEARNING

To gain a fundamental understanding of DL we need to consider the difference between descriptive analysis and predictive analysis. Descriptive analysis involves defining a comprehensible mathematical model which describes the

phenomenon that we wish to observe. This entails collecting data about a process, forming hypotheses on patterns in the data and validating these hypotheses through comparing the outcome of descriptive models we form with the real outcome [4]. Producing such models is precarious however because there is always a risk of unmodelled variables that scientists and engineers neglect to include due to ignorance or failure to understand some complex, hidden or non-intuitive phenomena [5]. Predictive analysis involves the discovery of rules that underlie a phenomenon and form a predictive model which minimize the error between the actual and the predicted outcome considering all possible interfering factors [4]. Machine learning rejects the traditional programming paradigm where problem analysis is replaced by a training framework where the system is fed a large number of training patterns (sets of inputs for which the desired outputs are known) which it learns and uses to compute new patterns [6]. DL is a subset of machine learning. DL is based largely on Artificial Neural Networks (ANNs), a computing paradigm inspired by the functioning of the human brain. Like the human brain, it is composed of many computing cells or 'neurons' that each perform a simple operation and interact with each other to make a decision [7]. Deep Learning is all about learning or 'credit assignment' across many layers of a neural network accurately, efficiently and without supervision and is of recent interest due to enabling advancements in processing hardware [1]. Self-organisation and the exploitation of interactions between small units have proven to perform better than central control, particularly for complex non-linear process models in that better fault tolerance and adaptability to new data is achievable [1].

B. ADVANTAGES OF DEEP LEARNING

Rapid progressions in DL and improvements in device capabilities including computing power, memory capacity, power consumption, image sensor resolution, and optics have improved the performance and cost-effectiveness of further quickened the spread of vision-based applications. Compared to traditional CV techniques, DL enables CV engineers to achieve greater accuracy in tasks such as image classification, semantic segmentation, object detection and Simultaneous Localization and Mapping (SLAM). Since neural networks used in DL are trained rather than programmed, applications using this approach often require less expert analysis and fine-tuning and exploit the tremendous amount of video data available in today's systems. DL also provides superior flexibility because CNN models and frameworks can be re-trained using a custom dataset for any use case, contrary to CV algorithms, which tend to be more domain-specific. Taking the problem of object detection on a mobile robot as an example, we can compare the two types of algorithms for computer vision: The traditional approach is to use well-established CV techniques such as feature descriptors (SIFT, SURF, BRIEF, etc.) for object detection. Before the emergence of DL, a step called feature extraction was carried out for tasks such as image classification. Features are small "interesting", descriptive or informative patches in images. Several CV algorithms, such as edge detection, corner detection or threshold segmentation may be involved in this step. As many features as practicable are extracted from images and these features form a definition (known as a bag-of-words) of each object class. At the deployment stage, these definitions are searched for in other images. If a significant number of features from one bag-of-words are in another image, the image is classified as containing that specific object (i.e. chair, horse, etc.). The difficulty with this traditional approach is that it is necessary to choose which features are important in each given image. As the number of classes to classify increases, feature extraction becomes more and more cumbersome. It is up to the CV engineer's judgment and a long trial and error process to decide which features best describe different classes of objects. Moreover, each feature definition requires dealing with a plethora of parameters, all of which must be fine-tuned by the CV engineer. DL introduced the concept of end-to-end learning where the machine is just given a dataset of images which have been annotated with what classes of object are present in each image [1]. Thereby a DL model is 'trained' on the given data, where neural networks discover the underlying patterns in classes of images and automatically works out the most descriptive and salient features with respect to each specific class of object for each object. It has been well-established that DNNs perform far better than traditional algorithms, albeit with trade-offs with respect to computing requirements and training time. With all the state-of-the-art approaches in CV employing this methodology, the workflow of the CV engineer has changed dramatically where the knowledge and expertise in extracting hand-crafted features has been replaced by knowledge and expertise in iterating through deep learning architectures as depicted in Fig. 1.

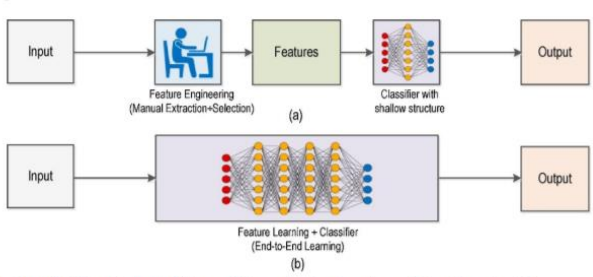

Fig. 1. (a) Traditional Computer Vision workflow vs. (b) Deep Learning workflow. Figure from [8].

The development of CNNs has had a tremendous influence in the field of CV in recent years and is responsible for a big jump in the ability to recognize objects [9]. This burst in progress has been enabled by an increase in computing power, as well as an increase in the amount of data available for training neural networks. The recent explosion in and wide-spread adoption of various deep-neural network architectures for CV is apparent in the fact that the seminal paper ImageNet Classification with Deep Convolutional Neural Networks has been cited over 3000 times [2]. CNNs make use of kernels (also known as filters), to detect features (e.g. edges) throughout an image. A kernel is just a matrix of values, called weights, which are trained to detect specific features. As their name indicates, the main idea behind the CNNs is to spatially convolve the kernel on a given input image check if the feature it is meant to detect is present. To provide a value representing how confident it is that a specific feature is present, a convolution operation is carried out by

computing the dot product of the kernel and the input area where kernel is overlapped (the area of the original image the kernel is looking at is known as the receptive field). To facilitate the learning of kernel weights, the convolution layer's output is summed with a bias term and then fed to a non-linear activation function. Activation Functions are usually non-linear functions like Sigmoid, TanH and ReLU (Rectified Linear Unit). Depending on the nature of data and classification tasks, these activation functions are selected accordingly. For example, ReLU is known to have more biological representation (neurons in the brain either fire or they don't). As a result, it yields favorable results for image recognition tasks as it is less susceptible to the vanishing gradient problem and it produces sparser, more efficient representations.

To speed up the training process and reduce the amount of memory consumed by the network, the convolutional layer is often followed by a pooling layer to remove redundancy present in the input feature. For example, max pooling moves a window over the input and simply outputs the maximum value in that window effectively reducing to the important pixels in an image [7]. As shown in Fig. 2, deep CNNs may have several pairs of convolutional and pooling layers. Finally, a Fully Connected layer flattens the previous layer volume into a feature vector and then an output layer which computes the scores (confidence or probabilities) for the output classes/features through a dense network. This output is then passed to a regression function such as Softmax [12], for example, which maps everything to a vector whose elements sum up to one [7].

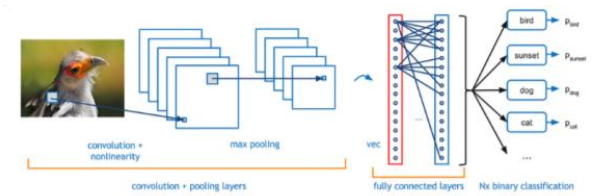

Fig. 2. Building blocks of a CNN. Figure from [13]

But DL is still only a tool of CV For example, the most common neural network used in CV is the CNN

CONVOLUTION OPERATION

The convolution of f and g is written f∗g, using an asterisk or star. It is defined as the integral of the product of the two functions after one is reversed and shifted. As such, it is a particular kind of integral transform

$$(f * g)(t) \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} f(\tau)\, g(t-\tau)\, d\tau = \int_{-\infty}^{\infty} f(t-\tau)\, g(\tau)\, d\tau \quad \text{(commutativity)}$$

C. Traditional Methods

Before DNN is proposed, features and classification methods refer to the most important topics. In the computer vision and image processing area, feature is a piece of information which is relevant for solving the computational tasks. In general, this is the same sense as feature in machine learning and pattern recognition. Variety of features are used for semantic segmentation, such as Pixel color, Histogram of oriented gradients (HOG) (Dalal and Triggs 2005; Bourdev et al. 2010), Scale-invariant feature transform (SIFT) (Lowe 2004), Local Binary Pattern (LBP) (He and Wang 1990), SURF (Bay et al. 2008), Harris Corners (Derpanis 2004), Shi-Tomasi (Shi et al. 1994), Sub-pixel Corner (Medioni and Yasumoto 1987), SUSAN (Smith and Brady 1997), Features from Accelerated Segment Test (FAST) (Rosten and Drummond 2005), FAST- ER (Rosten et al. 2010), AGAST (Mair et al. 2010) and Multiscale AGAST (Leutenegger et al. 2011) Detector, Bag-of-visual-words (BOV) (Csurka et al. 2004), Pselets (Brox et al. 2011), and Textons (Zhu et al. 2005), just to name a few. Approaches in image semantic segmentation include unsupervised and supervised ones. To be specific, the simple one is thresholding methods which are widely used in gray images. Gray images are very common in medical area where the collection equipment is usually X-ray CT scanner or MRI (Magnetic Resonance Imaging) equipment (Zheng et al. 2010; Hu et al. 2001; Xu et al. 2010). Overall, thresholding methods are quite effective in this area.

K-means clustering refers to an unsupervised method for clustering. The k-means algorithm requires the number of clusters to be given beforehand. Initially, k centroids are randomly placed in the feature space. Furthermore, it assigns each data point to the nearest centroid, successively moves the centroid to the center of the cluster, and continues the process until the stopping criterion is reached (Hartigan and Hartigan 1975). The segmentation problem can be treated as an energy model. It derives from compression based method which is implemented in Mobahi et al. (2010). Intuitively, edge is important information for segmentation. There are also many edgebased detection researches (Kimmel and Bruckstein 2003; Osher and Paragios 2003; Barghout 2014; Pedrycz et al. 2008; Barghout and Lee 2003; Lindeberg and Li 1997). Besides, edge-based approaches and region-growing methods (Nock and Nielsen 2004) are also other branches. Support vector machine (SVMs): SVMs are well-studied binary classifiers which perform well on many tasks. The training data is represented as (xi, yi) where xi is the feature vector and yi ∈ {−1, 1} the binary label for training example i ∈ {1,…., m}. Where w is a weight vector and b is the bias factor. Solving SVM is an optimization problem described as Eq. 5.

$$\min_{w,b} = \frac{1}{2}||w||^2$$
$$s.t.\ \forall_{i=1}^{m} y_i \cdot (< w, x_i > + b) \geq 1$$

Slack variables can solve linearly inseparable problems. Besides, kernel method is adopted to deal with inseparable tasks through mapping current dimensional features to higher dimension. Markov Random Network (MRF) is a set of random variables having a Markov property described by an undirected graph. Also, it is an undirected graphical model. Let x be the input, and y be the output. MRF learns the distribution P(y, x). In contrast to MRF, A CRF (Russell et al. 2009) is essentially a structured extension of logistic regression, and it models the conditional probabilities P(Y |X). These two models and their variations are widely used and have reached the best performance in segmentation (http://host.robots.ox.ac.uk/pascal/VOC/voc2010/results/index.html; He et al. 2004; Shotton et al. 2006).

D. Recent DNN in segmentation

Artificial Neural Network (ANN) is inspired by biologic neurons. The basic element of ANN is artificial neuron. Each single artificial neuron has some inputs which are weighted and summed up. Followed by a transfer function or activation function, the neuron outputs a scale value. An example of neural model is illustrated in Fig. 3. Based on artificial neuron, different stacking of the neurons forms Auto-encoder (Bengio 2009), Restricted Boltz- mann Machine (RBM) (Larochelle and Bengio 2008), Recurrent Neural Network or Recursive Neural Network (RNN), Convolutional Neural Network (CNN) (LeCun and Bengio 1995), Long Short Term Memory (LSTM) (Hochreiter and Schmidhuber 1997) and other types of ANNs. The basic architecture is illustrated in Fig. 4. Convolutional Neural Network (CNN) (LeCun and Bengio 1995) uses shared-weight architecture, which is inspired by biological processes. The connectivity pattern between neurons is mimic of the organization of the animal visual cortex. Another important concept is receptive field, and it means that individual cortical neurons respond to stimuli only in a

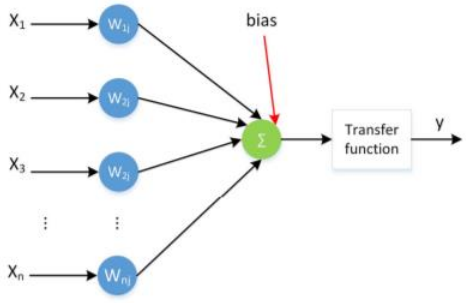

Fig. 3 Artificial neuron model

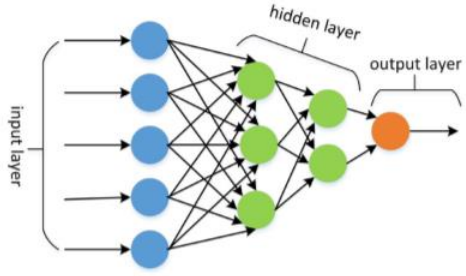

Fig. 4 An example of artificial neural network model

restricted region of the visual field. Also, they have the property of shift invariant or space invariant, based on their shared-weight architecture and translation invariance characteristics. Due to the excellent structure, CNN has obtained remarkable results on image classification, segmentation, and detection. The following part will present the recent progresses by applying CNNs in image semantic segmentation.

III. DATASET

We have used OXFORT-IIIT Pet Dataset[8], a 37 category pet dataset with roughly 200 images for each class. The images have a large variation in scale, pose and lighting. All images have an associated ground truth annotation of breed, head ROI, and pixel level trimap segmentation. This dataset is an inbuilt dataset in TensorFlow examples.

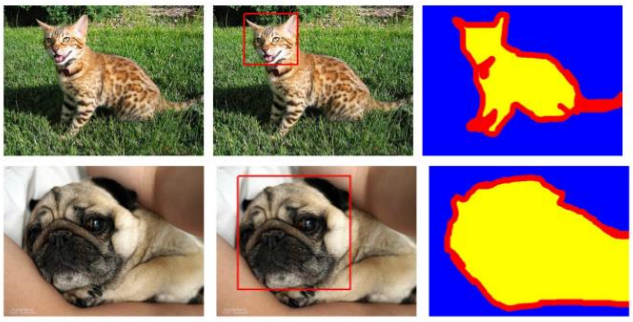

Fig.5: Annotation Examples

IV. IMAGE AUGMENTATION

Data Augmentation approaches overfitting from the root of the problem, the training dataset. This is done under the assumption that more information can be extracted from the original dataset through augmentations. These augmentations artificially inflate the training dataset size by either data warping or oversampling. Data warping augmentations transform existing images such that their label is preserved. This encompasses augmentations such as geometric and color transformations, random erasing, adversarial training, and neural style transfer. Oversampling augmentations create synthetic instances and add them to the training set.

Fig.6 - Data Augmentations based on basic image manipulations

V. TRANSFER LEARNING

Transfer Learning is a Machine Learning technique whereby a model is trained and developed for one task and is then re-used on a second related task. It refers to the situation whereby what has been learnt in one setting is exploited to improve optimization in another setting. Transfer Learning is usually applied when there is a new dataset smaller than the original dataset used to train the pre-trained model.

This paper proposes a system which uses a model (MobileNetV2) in which was first trained on a base dataset (MobileNet), and is now being repurposed to learn features (or

transfer them), to be trained on a new dataset (OXFORD-IIIT PET DATASET)

A. MOBILENETV2

MobileNets are based on a streamlined architecture that uses depth wise separable convolutions to build light weight deep neural networks. There are two simple global hyperparameters that efficiently tradeoff between latency and accuracy. These hyper-parameters allow the model builder to choose the right sized model for their application based on the constraints of the problem.

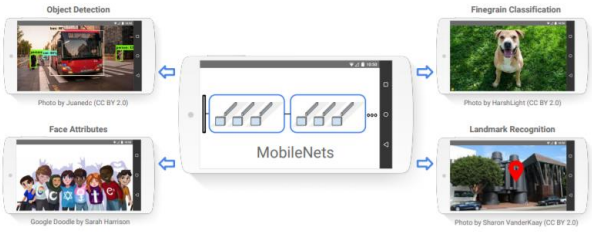

Fig.6 MobileNet models can be applied to various recognition tasks for efficient on device intelligence

The MobileNet structure is built on depth wise separable convolutions as mentioned in the previous section except for the first layer which is a full convolution. By defining the network in such simple terms, we are able to easily explore network topologies to find a good network. The MobileNet architecture is defined in Table 1. All layers are followed by a batchnorm and ReLU nonlinearity with the exception of the final fully connected layer which has no nonlinearity and feeds into a SoftMax layer for classification. Down sampling is handled with stride convolution in the depth wise convolutions as well as in the first layer. A final average pooling reduces the spatial resolution to 1 before the fully connected layer. Counting depthwise and pointwise convolutions as separate layers, MobileNet has 28 layers.

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5\times$ Conv dw / s1 Conv / s1	$3 \times 3 \times 512$ dw $1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$ $14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

Table 1. MobileNet Body Architecture

Fig.7 Example Object Detection results using MobileNet SSD

VI. U-NET ARCHITECTURE

The network architecture is illustrated in Figure 8 below. It is an encoder-decoder based architecture with skip connection between them. The encoder part retains the context specific information and the decoder part retains the localization information. Encoding part can be called the contracting path (left side) and decoding part can be called the expansive path (right side). For the contracting part, we have used pre-trained weights from MobilenetV2. The contracting path follows the typical architecture of a convolutional network. It consists of the repeated application of two 3x3 convolutions (unpadded convolutions), each followed by a rectified linear unit (ReLU) and a 2x2 max pooling operation with stride 2 for down sampling. At each down sampling step we double the number of feature channels. The weights for Every step in the expansive path consists of an up sampling of the feature map followed by a 2x2 convolution ("up-convolution") that halves the number of feature channels, a concatenation with the correspondingly cropped feature map from the contracting path, and two 3x3 convolutions, each followed by a ReLU. The cropping is necessary due to the loss of border pixels in every convolution. At the final layer a 1x1 convolution is used to map each 64- component feature vector to the desired number of classes. In total the network has 23 convolutional layers. To allow a seamless tiling of the output segmentation map, it is important to select the input tile size such that all 2x2 max-pooling operations are applied to a layer with an even x- and y-size.

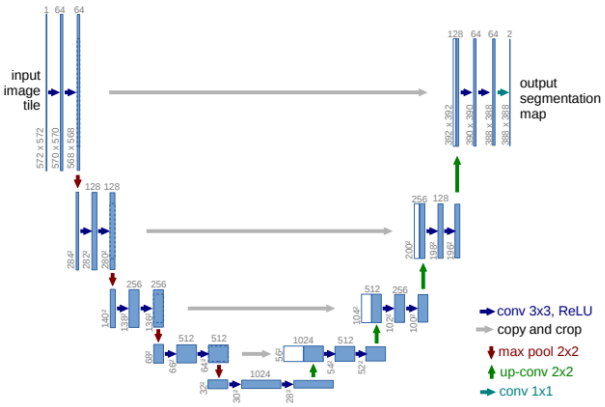

Fig.8: U-net architecture (example for 32*32 pixels in the lowest resolution). Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge

of the box. White boxes represent copied feature maps. The arrows denote the different operations.

The main idea is to supplement a usual contracting network by successive layers, where pooling operators are replaced by up sampling operators. Hence, these layers increase the resolution of the output. In order to localize, high resolution features from the contracting path are combined with the up sampled. output. A successive convolution layer can then learn to assemble a more precise output based on this information.

One important modification in our architecture is that in the up sampling part we have also a large number of feature channels, which allow the network to propagate context information to higher resolution layers. As a consequence, the expansive path is more or less symmetric to the contracting path, and yields a u-shaped architecture. The network does not have any fully connected layers and only uses the valid part of each convolution, i.e., the segmentation map only contains the pixels, for which the full context is available in the input image. This strategy allows the seamless segmentation of arbitrarily large images by an overlap-tile strategy (see Figure 9 below). To predict the pixels in the border region of the image, the missing context is extrapolated by mirroring the input image. This tiling strategy is important to apply the network to large images, since otherwise the resolution would be limited by the GPU memory.

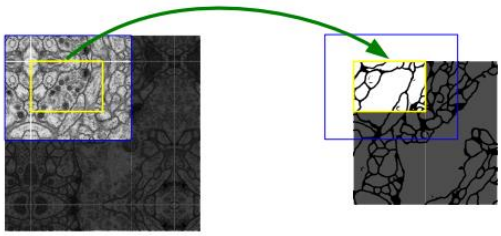

Fig.9 : Overlap-tile strategy for seamless segmentation of arbitrary large images. Prediction of the segmentation in the yellow area, requires image data within the blue area as input. Missing input is extrapolated by mirroring

Transposed convolution can be seen as the backward pass of a corresponding traditional convolution. It is also known as deconvolution and fractionally strided convolution. To stay consistent with most literature we use the term “deconvolution”. Contrary to the traditional convolution that connects multiple input activations to a single activation, deconvolution associates a single activation with multiple output activations. Figure 10 below shows a deconvolution operation of 3 × 3 kernel over a 4 × 4 input using unit stride and zero padding. The stride of deconvolution gives the dilation factor for the input feature map. Specifically, the deconvolution will first up sample the input by a factor of the stride value with padding, then perform convolution operation on the up sampled input. Recently, deconvolution has been widely used for visualization, recognition, localization, semantic segmentation, visual question answering , and super-resolution.

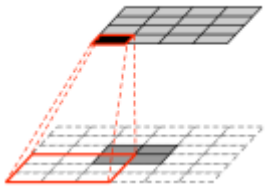

Fig.10 Deconvolution

VII. DATA PRE-PROCESSING

The train and the test images and labels are normalized to make the image pixel intensities range between 0 and 1. The images are then resized to 128*128 and 224*224 for trying out the weights trained on two variations of MobileNet which take image input of the following size. For enabling better model generalization, 50% of the training images and the corresponding labels have been flipped. The test images and labels are not flipped

VIII. UTILITY FUNCTION

- To enable easier prediction and model fetching, the below utility functions are built
- Saving checkpoints: After every successful epoch the checkpoints are saved in a location (provided by the user)
- These checkpoints will be used when the user wants to predict the classes on a new dataset
- Upload the image: The users are also enabled to directly upload an image on the notebook and the prediction is done
- U-net building: A function that builds u-net framework on top of the given MobileNet pre-trained weights
- Display image: At multiple instances, to check the quality of the prediction a function to showcase the prediction has been built
- Callbacks: When a user preferred accuracy level has been met, the function can terminate further training the checkpoints will be saved, this is also has been added

IX. HYPER-PARAMETER TUNING

A. Adam Configuration Parameters

Learning Rate: 0.001
Exponential decay rate for first moment (Beta1) = 0.9
Exponential decay rate for first moment (Beta2) = 0.999
Epsilon (small number to prevent division by zero):1e-08

B. Sparse Categorical Cross Entropy

Reduction – Setting the reduction option to be determined the usage context

C. Base Model for downsample

MobileNetV2 with input shape [224,224,3]
MobileNetV2 with input shape [128,128,3]
Taking the following layer weights for encoder part in U-net architecture

1. block_1_expand_relu
2. block_3_expand_relu
3. block_6_expand_relu
4. block_13_expand_relu
5. block_16_project

D. Layers for upsample

Taking the following upsample layer for decoder part of U-net architecture

6. upsample (512, 3)
7. upsample (256, 3)
8. upsample (128, 3)
9. upsample (64, 3)

E. Output Channels = 3

X. TRAINING

The input images and their corresponding segmentation maps (Train labels) are used to train the network with Adam Optimizer and the loss function as the Sparse Categorical Cross Entropy loss.

Adam optimization is a stochastic gradient descent method that is based on adaptive estimation of first-order and second-order moments. According to the paper Adam: A Method for Stochastic Optimization. Kingma et al., 2014, the method is "*computationally efficient, has little memory requirement, invariant to diagonal rescaling of gradients, and is well suited for problems that are large in terms of data/parameters*". Adam is different to classical stochastic gradient descent. Stochastic gradient descent maintains a single learning rate (termed alpha) for all weight updates and the learning rate does not change during training.

A learning rate is maintained for each network weight (parameter) and separately adapted as learning unfolds.

Sparse Categorical Cross entropy loss is used when there are two or more label classes and it suits our needs as for semantic segmentation there are usually more than 2 classes

Below is the model trained for input size 224*224*3

Layer (type)	Output Shape	Param #	Connected to
input_2 (InputLayer)	[(None, 224, 224, 3)	0	
model (Model)	[(None, 112, 112, 96	1841984	input_2[0][0]
sequential (Sequential)	(None, 14, 14, 512)	1476608	model[1][4]
concatenate (Concatenate)	(None, 14, 14, 1088)	0	sequential[0][0] model[1][3]
sequential_1 (Sequential)	(None, 28, 28, 256)	2507776	concatenate[0][0]
concatenate_1 (Concatenate)	(None, 28, 28, 448)	0	sequential_1[0][0] model[1][2]
sequential_2 (Sequential)	(None, 56, 56, 128)	516608	concatenate_1[0][0]
concatenate_2 (Concatenate)	(None, 56, 56, 272)	0	sequential_2[0][0] model[1][1]
sequential_3 (Sequential)	(None, 112, 112, 64)	156928	concatenate_2[0][0]
concatenate_3 (Concatenate)	(None, 112, 112, 160	0	sequential_3[0][0] model[1][0]
conv2d_transpose_4 (Conv2DTrans	(None, 224, 224, 3)	4323	concatenate_3[0][0]

Total params: 6,504,227
Trainable params: 4,660,323
Non-trainable params: 1,843,904

Table 2 : Model summary for input size 224*224*3

XI. SEMANTIC SEGMENTATION PREDICTION

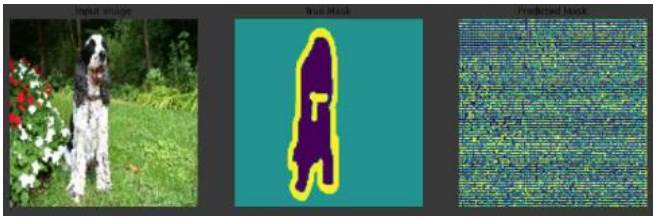

Fig.10 : The prediction without any training

Fig.11: Prediction after training for 20 epochs

XII. PORTRAIT MODE/DEPTH EFFECT

Portrait Mode on the Pixel smartphones lets you take professional-looking images that draw attention to a subject by blurring the background behind it. We will be using our trained network which is a U-net architecture based neural network to determine what pixels correspond to the object versus the background, and enable a depth-dependent blur, which is closer to what a professional camera does. We have blurred the background and sharpened the in-focus object.

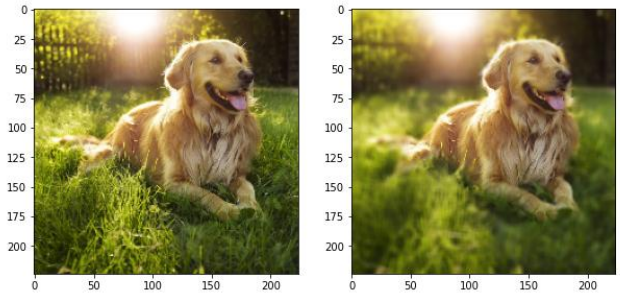

Fig.12 Left image is the original image and right image is the image with depth effect applied on it.

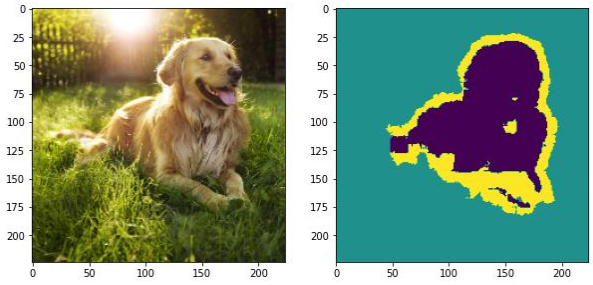

Fig.13 Left image is the original image and the right image shows the pixel level classification – focus area, boundary of focus area and the background

XIII. COMPARISON OF PERFORMANCE OF MODELS

Experiment 1

- Below are the results from the model that accepted the input image of resolution 128 x 128
- The output format would be an image with the classes laid out next to the true image
- The model has attained an accuracy of 88.6% on the Validation dataset, and as expected its higher in training dataset -93.5%
- Below is the performance improvement trend across epochs

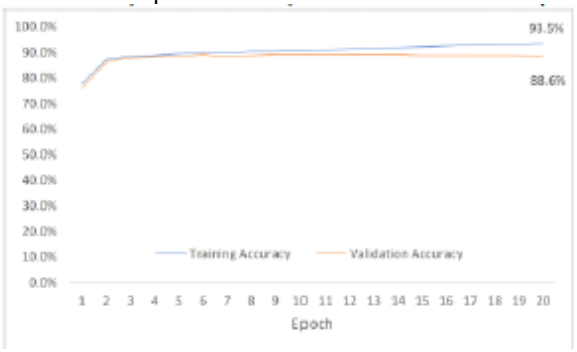

Fig.14 : Performance improvement trend across epoch for network trained with input image size 128*128

Experiment 2

- Below are the results from the model that accepted the input image of resolution 224*224

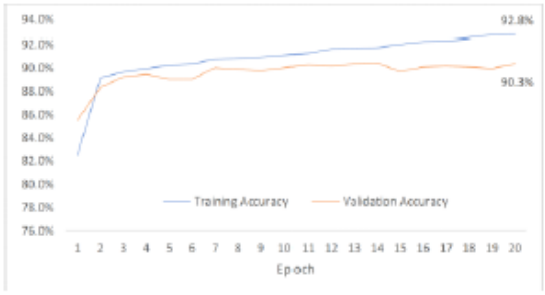

Fig.15 : Performance improvement trend across epoch for network trained with input image size 224*224

- The best training accuracy after 20 epochs has been observed as 92.8%, and the respective validation accuracy in 90.3%

INFERENCE:

- Enabling the architecture to process higher resolution images resulted in reduced overfitting
- With higher resolution, the training accuracy was lower compared to 128 resolution architecture, however on the validation dataset the model with higher resolution performed much better with an accuracy of 90.3%
- This is an indication of overfitting that has been avoided by introducing higher variance in the dataset with the help of higher resolution images, increasing the resolution also increased the number of parameters learned

REFERENCES

[1] Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS. pp. 1106–1114 (2012)

[2] LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., Jackel, L.D.: Backpropagation applied to handwritten zip code recognition. Neural Computation 1(4), 541–551 (1989)

[3] Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition (2014), arXiv:1409.1556 [cs.CV]

[4] Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2014)

[5] Hariharan, B., Arbelez, P., Girshick, R., Malik, J.: Hypercolumns for object segmentation and fine-grained localization (2014), arXiv:1411.5752 [cs.CV]

[6] He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing humanlevel performance on imagenet classification (2015), arXiv:1502.01852 [cs.CV]

[7] Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding (2014), arXiv:1408.5093 [cs.CV]

[8] *O. M. Parkhi, A. Vedaldi, A. Zisserman, C. V. Jawahar:* Cats and Dogs (2012)

[9] Olaf Ronneberger, Philipp Fischer, and Thomas Brox.: U-Net: Convolutional Networks for Biomedical Image Segmentation (2015), arXiv:1505.04597 [cs.CV]

[10] Xiaolong Liu · Zhidong Deng · Yuhan Yang.: Recent progress in semantic image segmentation (2018), arXiv:1809.10198 [cs.CV]