A project Report on

Intelligent security and information management system using video analysis

Submitted by

Ajeet Kumar Yadav	1805231005
Maneesh	1805231032
Prashant Singh	180523103
Pranjal Mittal	1900520319003

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE
AWARD OF THE DEGREE
BACHELOR OF TECHNOLOGY
In
ELECTRONICS AND COMMUNICATION ENGINEERING

Under the Guidance of

Dr. Rajiv K. Singh

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING INSTITUTE OF ENGINEERING & TECHNOLOGY

LUCKNOW – 226021

DECLARATION

We, hereby, declare that the work being presented in this dissertation entitled “Intelligent security and information management system using video analysis” towards the partial fulfillment for the award of degree of Bachelor of Technology in Electronics and Communication Engineering, is an authentic record of our own work carried out under the supervision and guidance of Dr. Rajiv Kumar Singh, Department of Electronics and Communication Engineering, IET Lucknow. The content of my dissertation, in full or in part, has not been submitted by me to any other university or institute for the award of any degree or diploma.

Date: 18/02/2022
Place: Lucknow

Ajeet Kumar Yadav(1805231005)
Maneesh (1805231032)
Prashant Singh (1805231037)
Pranjal Mittal (1900520319003)

CERTIFICATE

This is to certify that the project report entitled "**Intelligent security and information management system using video analysis**" submitted by **Mr. Ajeet Kumar Yadav, Mr. Maneesh , Mr. Prashant Singh, Mr. Pranjal Mittal** students of B.Tech. in Electronics and Communication Engineering, is a record of bonafide work carried out by the candidates under my supervision in the Department of Electronics and Communication Engineering at Institute of Engineering & Technology Lucknow. The project report is submitted in partial fulfillment of the requirements for the award of degree of Bachelor of Technology, in Electronics and Communication Engineering from Institute of Engineering and Technology, Lucknow.

To the best of my knowledge, the work carried out has not been submitted elsewhere for the award of any degree.

Dr. Rajiv K. Singh

Dr. Neelam Srivastava Professor
Head of Department Department
Electronics Communication Engineering

ACKNOWLEDGEMENT

Success is a sweet fruit, which everyone strives to taste. To achieve this goal, one has to put in a lot of physical and mental efforts. Each time we write this report, we gain stronger appreciation for the fact that we couldn't do it without the help of many talented and dedicated people. So we wish to express our appreciation to those whose help has been most valuable.

We wish to express our profound gratitude and indebtedness to **Dr. Rajiv K. Singh**, Professor Department of Electronics Engineering, IET Lucknow for introducing the present topic and for their inspiring guidance, constructive criticism and valuable suggestions throughout the project work.

We deeply express sincere thanks to our Head of Department **Dr. Neelam Srivastava** for encouraging and allowing us to present the project on the topic "Intelligent security and information management system using video analysis" at our department premises for the partial fulfillment of the requirements leading to the award of Btech degree.

Last but not least, our sincere thanks to all our friends who have patiently extended all sorts of help for accomplishing this undertaking

Ajeet Kumar Yadav (1805231005)
Maneesh (1805231032)
Prashant Singh (1805231037)
Pranjal Mittal (1900520319003)

Abstract

Nowadays, as computers are powerful enough for implementing complex algorithms, there are numerous applications that people utilize computers to run. In which, facial recognition is one of the most active fields of applications. In fact, computers can not only automatically identify who a person is, but also operate 24/7, which human beings cannot endure. This leads to the replacement of people by computers in some repetitive and real-time applications. In this work, we apply facial recognition into a video monitoring system that uses faces of registered people to check their presence and their frequency of activities.

This system has a GUI which allows easy user-to-system interaction. The core of the system is faster-RCNN technique. This model has five stages (e.g., Preprocessing of frames, detecting faces, removing non-frontal-view faces, recognizing, and database management). Particularly, in the recognition phase, we consider this stage as an open-set facial recognition problem, so the system is able to detect people who have not registered in the database before. Also, we boost the performance of the system by utilizing hardware resources of users' computers. Although the system is designed to run with a low-resolution webcam, its performance is reasonably accurate on our private dataset.

Introduction

Face recognition systems are being applied widely in real life such as tracking, managing employees, finding information about celebrities, and so forth. There are many approaches to design a security system using video analysis, but such systems are frequently affected by light, non-frontal faces, resolution of cameras, etc. Thus, each method has specific challenges. Overall, face recognition has two main stages which are face detection and face recognition.

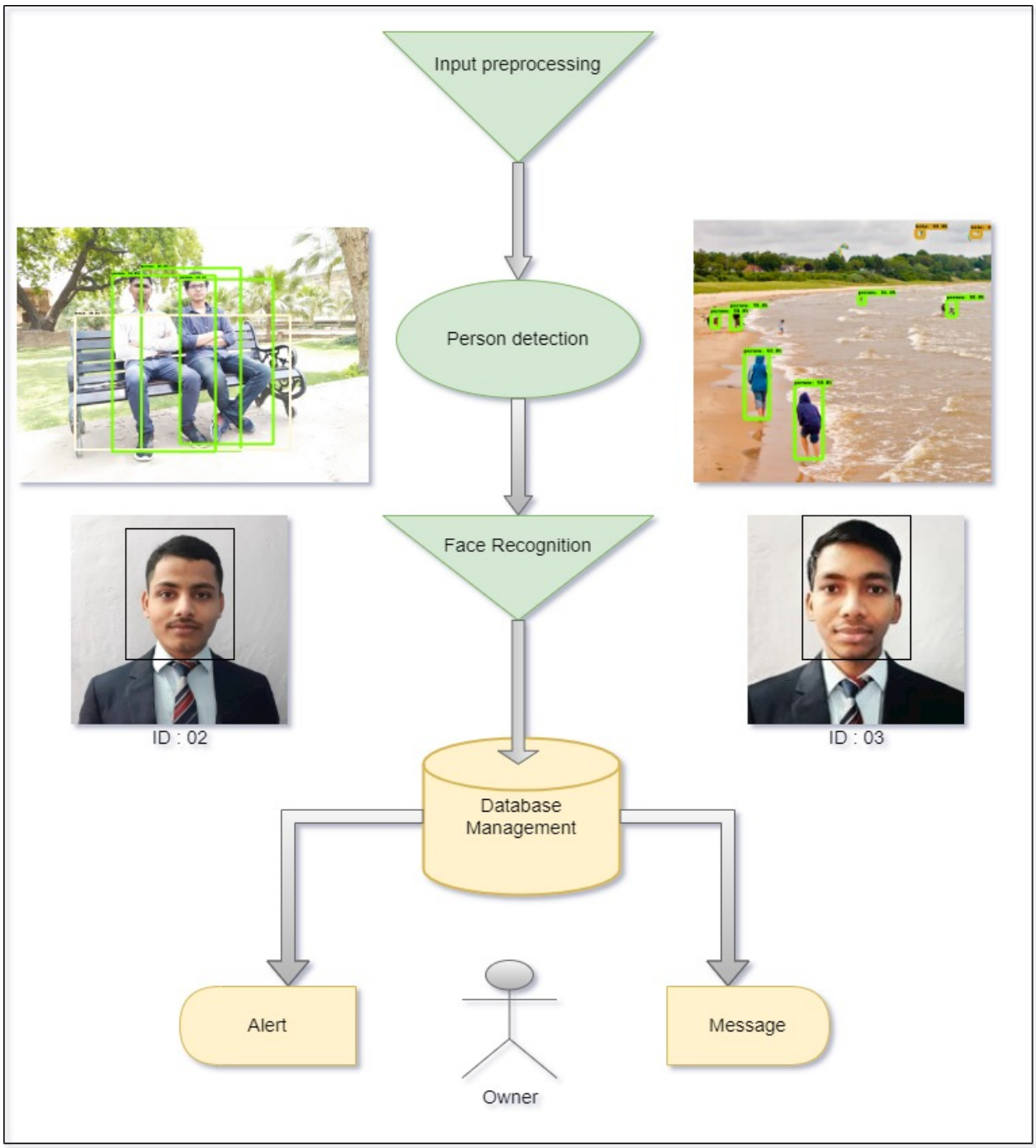
1. It is necessary to make use of automatic video analysis technologies for the development of smart surveillance systems which can aid the human operator in both detection and reaction to potential threats.
2. This project is based on the state-of-the-art object detection network, a faster-Region convolutional neural network. Faster R-CNN, is composed of two modules, where the first module is a deep fully convolutional network that proposes regions, and the second module is the Fast R-CNN detector that uses the proposed regions.
3. The entire system is a single, unified network for object detection.

Roadmap

A system is built in order to manage the appearance of a person in front of a camera. Normally, a facial recognition system is organized as a pipeline of typical stages such as face detection, landmark detection, and face recognition. However, to ensure input frames for underlying algorithms are high quality, we append an early filter (Input preprocessing stage) that is able to discard blur frames, which are caught by motions of people in front of a webcam. Then, clean frames are passed through the Face detection block for counting the number of faces existing inside these frames.

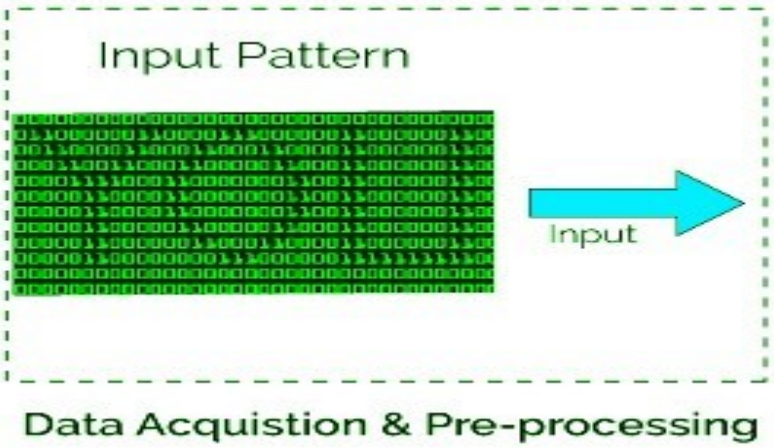
In our specific constraint, there is only one face per frame allowed to be processed. There are five stages, namely, input preprocessing, face detection/person detection, face recognition, Database management, and alert system. This points out that our system is tougher to input frames from the camera when such frames pass through the system. Therefore, the best frames are likely to be processed, which may improve the final recognition accuracy. that contain more than one face are rejected. Afterward, the Landmark detection is to localize statistic-salient points in the face in order to verify whether the face is in the frontal view of the camera. This frontal-view check will help improve the accuracy of the face recognition algorithm. Finally, the Face recognition uses blur-clean, one-face, and frontal-view frames from previous stages to extract relevant features and then perform a classification task to indicate which category the input most likely belongs to. If the person belongs to our database then simply it will count the frequency, but if the person does not belong to the database first it will send an alert to the owner and keep the frequency record.

In addition, to leverage the ease of use, we design a friendly graphic user interface (GUI) so that people who want to use the system to manage (owner) or check (home visitor) presence can interact with the application without any specialized knowledge. To make the system more robust, we carefully analyze the distribution outlier of features representing registered accounts. Therefore, the algorithm has the ability to detect people who have not registered in the application before, which is equivalent to the open-set problem in face recognition.


Input preprocessing
Person detection
Face Recognition
ID : 02
ID : 03
Database Management
Alert
Owner
Message

IMPLEMENTATION

Input Preprocessing: It may seem a bit fussy, but Keras has utilities to take over this whole algorithm and do the heavy lifting for you. Keras has a module with image-processing helping tools, located at keras.preprocessing.image. It contains the class *ImageDataGenerator*, which lets you quickly set up Python generators that can automatically turn image files on disk into batches of preprocessed tensors.



Algorithm:

1. Read the picture files (extracted from real time video)
2. Decode the JPEG content to RGB grids of pixels with channels.
3. Convert these into floating-point tensors for input to neural nets.
4. Rescale the pixel values (between 0 and 255) to the [0, 1] interval (as training neural networks with this range gets efficient).

Person Detection/Face Detection Our network consists of a batch input layer and a deep CNN followed by L2 normalization, which results in the face embedding. This is followed by the triplet loss during training. technique with the different sizes of the windows. Using the sliding window technique we could complete the calculation of HOG features, applied to detect and differentiate the face and the false face recognition using the SVM technique. All of the pre-processing steps are automatically implemented before using the Dlib library with the input of being given facial images and the output of localization of the identified faces. C. Frontal-view detection To check whether the shape of the faces has to be frontal, we implement these 3 following steps: Focusing on the center of the image. Only accept these faces that are located in the most central of the

images. ((120 − 360),(90, 360)) 2) Identify the skew of the image: calculate the coordinate of these eyes and the angle deviation between two eyes in the horizontal direction. If the angle deviation is larger than 10 degree, the image will be ignored. 3) Identify the rotation of the image: choose the point which is the midpoint of the right and the left eye. If the nose which is deviated from the selected point is greater than 10 pixels in the horizontal direction, the image is ignored.

Face Recognition- In this stage, faces in raw images are detected and aligned by Multi-task CNN, we use the convenient pre-trained faster-RCNN model to extract features and then feedforward them to an SVM classifier for recognition. Multi-task Convolution Network: The overall pipeline Multi-task CNN. An image is initially resized to different scales to build an image pyramid, which is the input of the following three-stage cascaded framework with CNN architectures. A fully convolutional network is exploited, called Proposal Network (P-Net), to obtain the proposed windows and their bounding box regression vectors. Then using the estimated bounding box regression vectors to calibrate the candidates. After that, employing non-maximum suppression (NMS) to merge highly overlapped candidates. For each candidate window, faster-RCNN predicts the offset between it and the nearest ground truth.

Faster RCNN

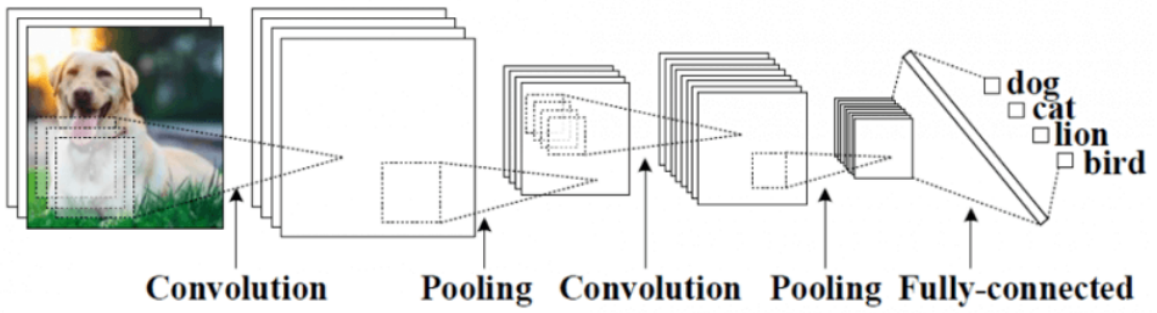
Faster R-CNN is proposed to shorten the time spent on region proposal steps during test time. Previous region proposal methods such as Selective Search (SS), is an order slower than state-of-the-art object detection network (Fast R-CNN). Hence the authors of Faster R-CNN came with the idea that the region proposal step can be combined with Fast R-CNN, so that it can take advantage of GPUs as well as sharing computations. Faster R-CNN is composed of two modules. The first one is Region Proposal Network (RPN), which shares convolutional layers with the second module, Fast R-CNN. The RPN first generates a set of bounding boxes with one or different size scales and ratios on each pixel. Since there will be many boxes with a relatively large overlap area, we will use Non-Maximum Suppression (NMS) technique to reduce the number of candidate boxes. Then, bounding boxes with Intersection-overUnion (IoU) score (with ground-truth box) higher than 0.7 are selected as foreground, and those lower than 0.3 as background. To optimize the RPN during the training, both classification loss and bounding box regression loss will be calculated. Classification Loss is the entropy loss on whether it's foreground or background. For Bounding Box Regression Loss, the difference between the regression of the foreground box and that of the ground truth box is plugged into the L1 function, and then sum all terms together. For the second module (or detection network), Fast R-CNN, there is a ”head” part which processes the original image using several convolutional layers and max pooling layers to produce a feature map. The ”tail” part takes object proposals (from RPN) to the region of interest (RoI) pooling layer and extracts a fixed-length feature vector from the feature map. Then, the feature vector is put into a sequence of fully connected layers which will branch into two output layers. One will produce softmax probability to estimate over 20 object classes and a “background” class. Another will output four real-valued numbers for each of the 20 object classes. Note that a set of 4 values will be used to calculate refined bounding box positions for a certain class. The author introduced three training methods. The first one is 4-Step Alternating Training: initialize the convolutional layers in RPN network by a pre-trained model and train the RPN network; initialize the Fast R-CNN convolutional layers with the pre-trained model; use the proposals generated from the trained RPN to train the Fast R-CNN; use the trained RPN to train the layers above the shared convolutional layers of Fast R-CNN. The second one is Approximate joint training, which is the one we use to train out model, and it will be introduced later. The third one is Non-approximate joint training. It involves the derivative w.r.t. the proposal boxes' coordinates that are ignored in the second method.

Implementation

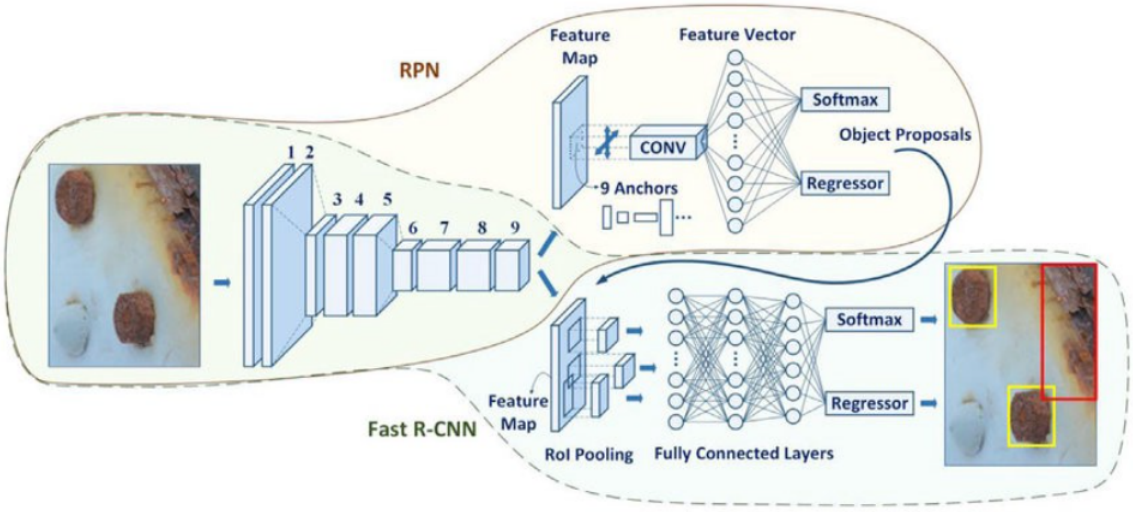
In this section, we will briefly introduce the main idea of the code our group developed for this project. We will explain the architecture of the code along with some important hyper-parameters as well as the training method we used for this network.

Architecture

The architecture of the network is explained in great detail in the original paper [1]. Thus, we mainly follow the code structure of Girshick, one of the paper authors, from his github repository [1]. The structure of the network is that a convolutional neural network is on the base and its feature map from the last layer is used to feed a region proposal network (RPN).



We used two kinds of base net in our 1 project: the VGG16 and Res101. On the other hand, we generate initial bounding boxes for each pixel through the image, (the initial bounding boxes are called anchors) and RPN learns to extract foreground anchors and adjust its shape to get bounding boxes. Then we apply non-maximum suppression (NMS) layer to the proposed bounding boxes to eliminate their numbers. The suppression is done to eliminate boxes that overlap greatly and only keeps those with the top foreground scores. So far, the network has proposed bounding boxes for the region of interest, then we will use them to find classes for the object int he bounding boxes as well as a class dependent regression for the bounding boxes. We use crop pooling to extract feature maps to a given shape, so that all bounding boxes will give feature maps with the same shape. As a result, we can pass the feature maps of standard shape from the bounding boxes to the last classifier layer to have class and bounding box regression coefficients for the bounding boxes. Both of the classifier and the bounding box regression layer are one layer linear fully connected networks.



Training Method

In this architecture, we have introduced four losses. The RPN network will output a foreground/background classification cross entropy loss as well as the bounding boxes regression loss. The last classifier layer will introduce the object classification cross entropy layer and the class dependent bounding box regression loss. We sum all the four losses and use backpropagation to decrease the loss. This is the method called Approximate joint training. We choose this method because it is easier to implement and more convenient to train. Note this solution is approximate since it ignores the derivative with respect to the proposal boxes' coordinates which are also network responses. Although it is not exact, it works pretty good in practice. One reason might be that the RPN gets 'direct' supervision from optimizing the total loss. With the main idea introduced, it is better to explain the bounding box regression loss we used since this loss is not a classical loss. We chose to use smooth L1 loss as suggested by the original paper [1]. In implementing this loss for RPN, we choose anchors that overlap greatly with the ground truth box as foreground anchors. The RPN net will give regression coefficients for the foreground anchors ui(predicted), with i = width, height, x and y. We will use the ground box which overlaps mostly with the anchor we are considering as our label and get its regression coefficient to the anchor ui(target). Then our loss is L = 1 N X foreground "X i S (ui(predicted) − ui(target))# , where S(x) = σ 2x 2 2 when kxk < 1/σ2 kxk − 0.5/σ2 otherwise (1) Here we choose σ = 3 and N is the total number of chosen foreground anchors. Since RPN and the last classifier layer all share the base convolution layers, the backpropagation will backpropagate to the base layers. To save computational resources, we fix the layer weights for the first few layers in the base network. In the VGG16 base case, we fixed the layer weights before the conv3 layer. On the other hand, in the Res101 case, we fixed the first convolutional layer and the first batch normalization layer. Our code allows ADAM optimizers as well as SGD optimizers. However, we mainly focus on the SGD optimizer with decreasing learning rate.

Hyper-parameters

Above is the basic structure of our network, and we will introduce some important hyper-parameters such as the anchor initialization, the maximum number of bounding boxes kept and learning rate decay. We initialize our anchors on each pixel of the feature map got from the base net. This corresponds to put anchors on every 16 pixels on each dimension. For each pixel of the feature map, there will be 9 anchors initialized. The 9 anchors are generated by three different scales and three different height-width ratios. Our scales are 8, 16 and 32. Our ratios are 1 : 2, 1 : 1 and 2 : 1. Also, in training, we need to use the target bounding boxes as labels to train our RPN. We choose the anchors whose overlap with ground truth box is greater than RPN POSITIVE OVERLAP = 0.7 as foreground label and whose overlap with any ground truth box is lower than RPN NEGATIVE OVERLAP = 0.3 as a negative label or background label. Those who do not fulfill any of the two conditions are not considered. Thus, we have chosen the appropriate number of anchors to train the RPN network. With the reduced number of anchors, we will have a reduced number of ROIs which are bounding boxes generated from the chosen anchors by applying transformations calculated by RPN. However, we will choose ROIs whose overlap with the foreground is higher than FG THRESH = 0.5 as foreground ROIs and use those ROIs to do crop pooling and then get the classifier's classification loss and bounding box regression loss. Although we have many other hyperparameters, we do not list all of them here. Table 1 shows the training hyperparameters for two nets we trained, VGG16 and Res101. Since we also did some other measurements and comparisons with the original paper[1], we changed the above-mentioned hyperparameters in those experiments and the particular choice will be mentioned when we introduce the experiment results.

Faster RCNN for Object Detection

In the R-CNN family of papers, the evolution between versions was usually in terms of computational efficiency (integrating the different training stages), reduction in test time, and improvement in performance (mAP). These networks usually consist of — a) A region proposal algorithm to generate “bounding boxes" or locations of possible objects in the image; b) A feature generation stage to obtain features of these objects, usually using a CNN; c) A classification layer to predict which class this object belongs to; and d) A regression layer to make the coordinates of the object bounding box more precise. The only stand-alone portion of the network left in Fast R-CNN was the region proposal algorithm. Both R-CNN and Fast R-CNN use CPU based region proposal algorithms, Eg- the Selective search algorithm which takes around 2 seconds per image and runs on CPU computation. The Faster R-CNN paper fixes this by using another convolutional network (the RPN) to generate the region proposals. This not only brings down the region proposal time from 2s to 10 ms per image but also allows the region proposal stage to share layers with the following detection stages, causing an overall improvement in feature representation. In the rest of the article, “Faster R-CNN " usually refers to a detection pipeline that uses the RPN as a region proposal algorithm, and Fast R-CNN as a detector network.

Region Proposal Network (RPN)

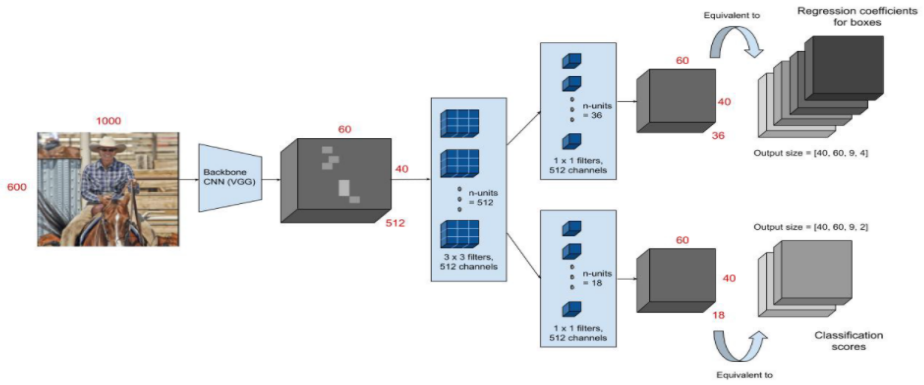
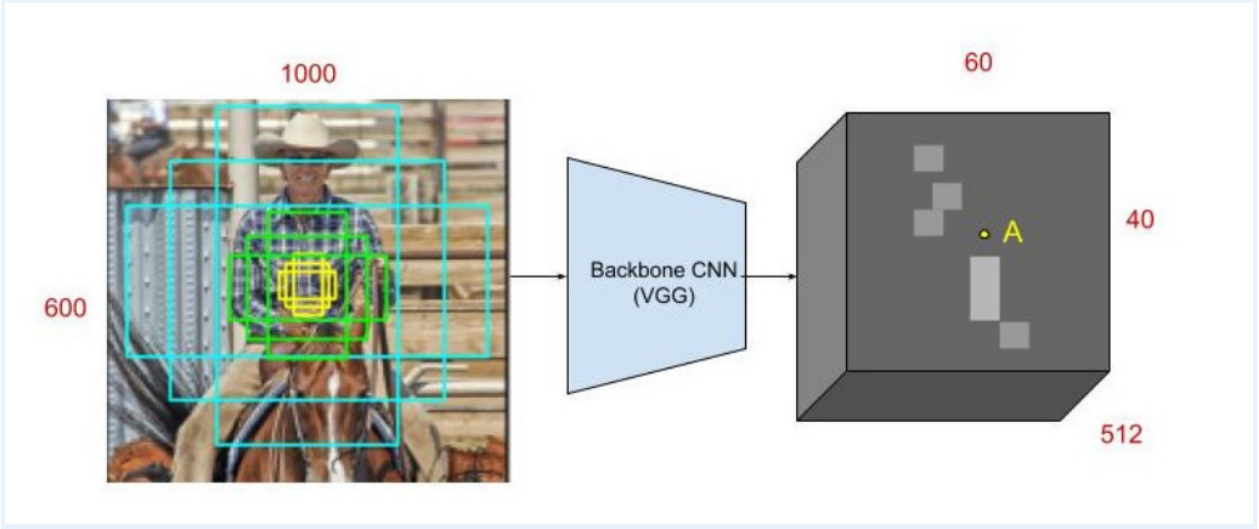


Figure 1: The architecture of the region proposal network or RPN

Architecture

- The region proposal network (RPN) starts with the input image being fed into the backbone convolutional neural network. The input image is first resized such that its shortest side is 600px with the longer side not exceeding 1000px.
- The output features of the backbone network (indicated by H x W) are usually much smaller than the input image depending on the stride of the backbone network. For both the possible backbone networks used in the paper (VGG, ZF-Net) the network stride is 16. This means that two consecutive pixels in the backbone output features correspond to two points 16 pixels apart in the input image.
- For every point in the output feature map, the network has to learn whether an object is present in the input image at its corresponding location and estimate its size. This is done by placing a set of “Anchors” on the input image for each location on the output feature map from the backbone network. These anchors indicate possible objects in various sizes and aspect ratios at this location. The figure below shows 9 possible anchors in 3 different aspect ratios and 3 different sizes placed on the input image for a point A on the output feature map. For the PASCAL challenge,

the anchors used have 3 scales of box area 128^2, 256^2, 512^2 and 3 aspect ratios of 1:1, 1:2 and 2:1.



- First, a 3 x 3 convolution with 512 units is applied to the backbone feature map as shown in Figure 1, to give a 512-d feature map for every location. This is followed by two sibling layers: a 1 x 1 convolution layer with 18 units for object classification, and a 1 x 1 convolution with 36 units for bounding box regression.
- The 18 units in the classification branch give an output of size (H, W, 18). This output is used to give probabilities of whether or not each point in the backbone feature map (size: H x W) **contains an object** within all 9 of the anchors at that point.
- The 36 units in the regression branch give an output of size (H, W, 36). This output is used to give the 4 regression coefficients of each of the 9 anchors for every point in the backbone feature map (size: H x W). These regression coefficients are used to improve the coordinates of the anchors that contain objects.

Training and Loss functions

- The output feature map consists of about 40 x 60 locations, corresponding to 40*60*9 ~ 20k anchors in total. At train time, all the anchors that cross the boundary are ignored so that they do not contribute to the loss. This leaves about 6k anchors per image.
- An anchor is considered to be a “positive” sample if it satisfies either of the two conditions — a) The anchor has the highest IoU (Intersection over Union, a measure of overlap) with a ground truth box; b) The anchor has an IoU greater than 0.7 with any ground truth box. The same ground truth box can cause multiple anchors to be assigned positive labels.
- An anchor is labeled “negative” if its IoU with all ground truth boxes is less than 0.3. The remaining anchors (neither positive nor negative) are disregarded for RPN training.
- Each mini-batch for training the RPN comes from a single image. Sampling all the anchors from this image would bias the learning process toward negative samples, and so 128 positive and 128 negative samples are randomly selected to form the batch, padding with additional negative samples if there are an insufficient number of positives.
- The training loss for the RPN is also a multi-task loss, given by:

$$L(\{p_i\}, \{t_i\}) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*).$$

- Here *i* is the index of the anchor in the mini-batch. The classification loss L_c (p_i, p_i^*) is the log loss over two classes (object vs not object). p_i is the output score from the classification branch for anchor *i*, and p_i^* is the ground-truth label (1 or 0).
- The regression loss $L_{re}(t_i, t_i^*)$ is activated only if the anchor actually contains an object i.e., the ground truth p_i^* is 1. The term t_i is the output prediction of the regression layer and consists of 4 variables [t_x, t_y, tw, t]. The regression target t_i^* is

calculated as —

$$t_x^* = (x^* - x_a)/w_a, \quad t_y^* = (y^* - y_a)/h_a, \quad t_w^* = \log(w^*/w_a), \quad t_h^* = \log(h^*/h_a)$$

- Here x, y, w, and h correspond to the *(x, y)* coordinates of the box centre and the height *h* and width *w* of the box. x_a, x* stand for the coordinates of the anchor box and its corresponding ground truth bounding box.
- Remember that all *k* (= 9) of the anchor boxes have different regressors that do not share weights. So the regression loss for an anchor *i* is applied to its corresponding regressor (if it is a positive sample).
- At test time, the learned regression output t_i can be applied to its corresponding anchor box (that is predicted positive), and the *x, y, w, h* parameters for the predicted object proposal bounding box can be back-calculated from —

$$t_x = (x - x_a)/w_a, \quad t_y = (y - y_a)/h_a, \quad t_w = \log(w/w_a), \quad t_h = \log(h/h_a)$$

Test time details

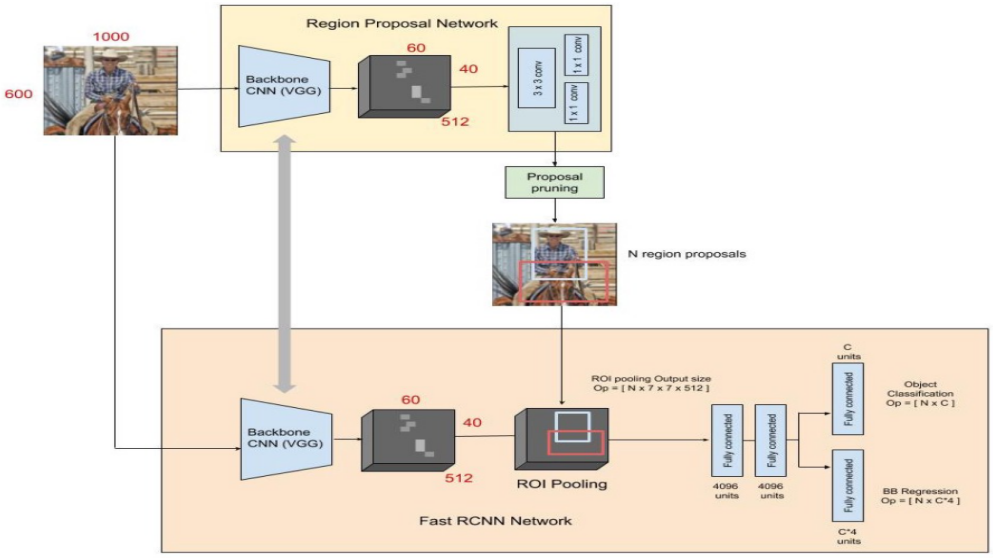
At test time, the 20k anchors from each image go through a series of post-processing steps to send in the object proposal bounding boxes.

- The regression coefficients are applied to the anchors for precise localization. This gives precise bounding boxes.
- All the boxes are arranged according to their *cls* scores. Then, a non-maximum suppression (NMS) is applied with a threshold of 0.7. From the top down, all of the bounding boxes which have an IoU of greater than 0.7 with another bounding box are discarded. Thus the highest-scoring bounding box is retained for a group of overlapping boxes.

- This gives about 2k proposals per image.
- The cross-boundary bounding boxes are retained and clipped to the image boundary.
- While using these object proposals to train the Fast R-CNN detection pipeline, all 2k proposals from the RPN are used. At test time for Fast R-CNN detection, only the Top N proposals from the RPN are chosen.

Object detection: Faster R-CNN (RPN + Fast R-CNN)

The Faster R-CNN architecture consists of the RPN as a region proposal algorithm and the Fast R-CNN as a detector network.



Fast R-CNN as a detector for Faster R-CNN

The Fast R-CNN detector also consists of a CNN backbone, an ROI pooling layer, and fully connected layers followed by two sibling branches for classification and bounding box regression as shown in Figure 3.

- The input image is first passed through the backbone CNN to get the feature map (Feature size: 60, 40, 512). Besides test time efficiency, another key reason using an RPN as a proposal generator makes sense is the advantages of weight sharing between the RPN backbone and the Fast R-CNN detector backbone.
- Next, the bounding box proposals from the RPN are used to pool features from the backbone feature map. This is done by the ROI pooling layer. The ROI pooling layer, in essence, works by a) Taking the region corresponding to a proposal from the backbone feature map; b) Dividing this region into a fixed number of sub-windows; c) Performing max-pooling over these sub-windows to give a fixed size output. To understand the details of the ROI pooling layer and it's advantages, read Fast R-CNN.
- The output from the ROI pooling layer has a size of (N, 7, 7, 512) where N is the number of proposals from the region proposal algorithm. After passing them through two fully connected layers, the features are fed into the sibling classification and regression branches.
- Note that these classification and detection branches are different from those of the RPN. Here the classification layer has C units for each of the classes in the detection task (including a catch-all background class). The features are passed through a softmax layer to get the classification scores — the probability of a proposal belonging to each class. The regression layer coefficients are used to improve the predicted bounding boxes. Here the regressor is size agnostic, (unlike the RPN) but is specific to each class. That is, all the classes have individual

regressors with 4 parameters each corresponding to C*4 output units in the regression layer.

- For more details on how the Faster R-CNN is trained and its loss functions refer to Fast R-CNN

Step Alternating training

In order to force the network to share the weights of the CNN backbone between the RPN and the detector, the authors use a 4 step training method:

a) The RPN is trained independently as described above. The backbone CNN for this task is initialized with weights from a network trained for an ImageNet classification task, and is then fine-tuned for the region proposal task.
b) The Fast R-CNN detector network is also trained independently. The backbone CNN for this task is initialized with weights from a network trained for an ImageNet classification task, and is then fine-tuned for the object detection task. The RPN weights are fixed and the proposals from the RPN are used to train the Faster R-CNN.
c) The RPN is now initialized with weights from this Faster R-CNN, and fine-tuned for the region proposal task. This time, weights in the common layers between the RPN and detector remain fixed, and only the layers unique to the RPN are fine-tuned. This is the final RPN.
d) Once again using the new RPN, the Fast R-CNN detector is fine-tuned. Again, only the layers unique to the detector network are fine-tuned and the common layer weights are fixed.

This gives a Faster R-CNN detection framework that has shared convolutional layers.

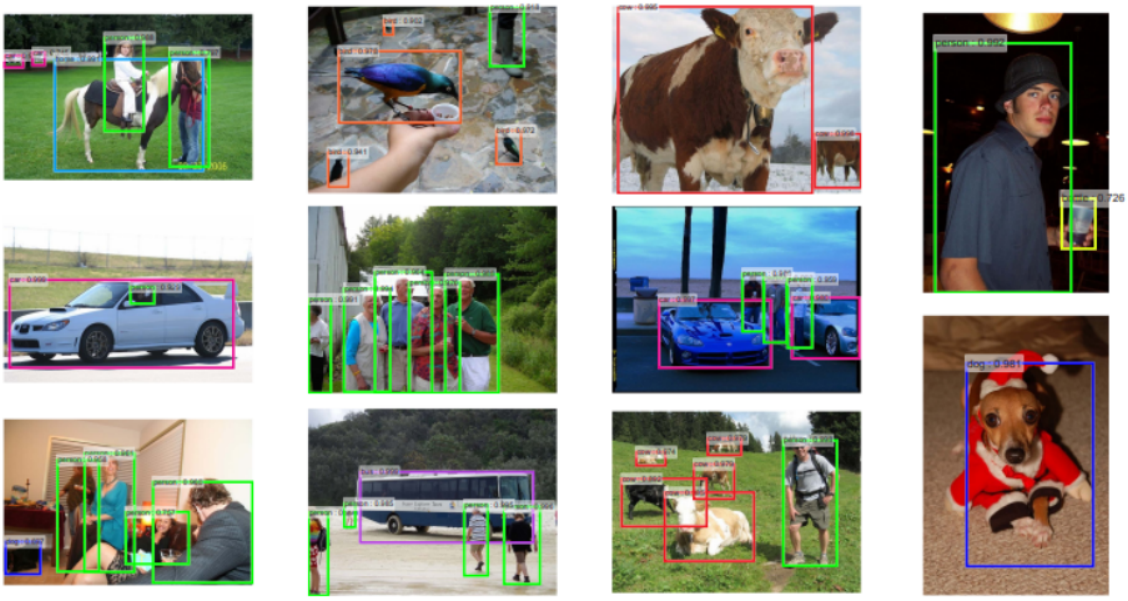


Figure 4: Results of the Faster R-CNN detection framework with a VGG backbone

Cloud Computing

Cloud computing is the on-demand delivery of compute power, database, storage, applications, and other IT resources through a cloud services platform via the Internet with pay-as-you-go pricing. Whether you are running applications that share photos to millions of mobile users or you're supporting the critical operations of your business, a cloud services platform provides rapid access to flexible and low-cost IT resources. With cloud computing, you don't need to make large upfront investments in hardware and spend a lot of time on the heavy lifting of managing that hardware. Instead, you can provision exactly the right type and size of computing resources you need to power your newest bright idea or operate your IT department. You can access as many resources as you need, almost instantly, and only pay for what you use. Cloud computing provides a simple way to access servers, storage, databases and a broad set of application services over the Internet. A cloud services platform such as Amazon Web Services owns and maintains the network-connected hardware required for these application services, while you provision and use what you need via a web application.

Advantages of Cloud Computing

• Trade capital expense for variable expense – Instead of having to invest heavily in data

centers and servers before you know how you're going to use them, you can pay only when you consume computing resources, and pay only for how much you consume.

• Benefit from massive economies of scale – By using cloud computing, you can achieve a lower variable cost than you can get on your own. Because usage from hundreds of thousands of customers is aggregated in the cloud, providers such as AWS can achieve higher economies of scale, which translates into lower pay as-you-go prices.

• Stop guessing capacity – Eliminate guessing on your infrastructure capacity needs. When you make a capacity decision prior to deploying an application, you often end up either sitting on expensive idle resources or dealing with limited capacity. With cloud computing, These problems go away. You can access as much or as little capacity as you need, and scale up and down as required with only a few minutes' notice.

• Increase speed and agility – In a cloud computing environment, new IT resources are only a click away, which means that you reduce the time to make those resources available to your developers from weeks to just minutes. This results in a dramatic increase in agility for the organization, since the cost and time it takes to experiment and develop is significantly lower.

• Stop spending money running and maintaining data centers – Focus on projects that differentiate your business, not the infrastructure. Cloud computing lets you focus on your own customers, rather than on the heavy lifting of racking, stacking, and powering servers.

• Go global in minutes – Easily deploy your application in multiple regions around the world with just a few clicks. This means you can provide lower latency and a better experience for your customers at a minimal cost.

Types of Cloud Computing

Cloud computing provides developers and IT departments with the ability to focus on what matters most and avoid undifferentiated work such as procurement, maintenance, and capacity planning. As cloud computing has grown in popularity, several different models and deployment strategies have emerged to help meet specific needs of different users. Each type of cloud service and deployment method provides you with different levels of control, flexibility, and management. Understanding the differences between Infrastructure as a Service, Platform as a Service, and Software as a Service, as well as what deployment strategies you can use, can help you decide what set of services is right for your needs.

Cloud Computing Models

Infrastructure as a Service (IaaS) : Infrastructure as a Service (IaaS) contains the basic building blocks for cloud IT and typically provides access to networking features, computers (virtual or on dedicated hardware), and data storage space. IaaS provides you

with the highest level of flexibility and management control over your IT resources and is most similar to existing IT resources that many IT departments and developers are familiar with today.

Platform as a Service (PaaS) : Platform as a Service (PaaS) removes the need for your organization to manage the underlying infrastructure (usually hardware and operating systems) and allows you to focus on the deployment and management of your applications. This helps you be more efficient as you don't need to worry about resource procurement, capacity planning, software maintenance, patching, or any of the other undifferentiated heavy lifting involved in running your application.

Software as a Service (SaaS) : Software as a Service (SaaS) provides you with a completed product that is run and managed by the service provider. In most cases, people referring to Software as a Service are referring to end-user applications. With a SaaS offering you do not have to think about how the service is maintained or how the underlying infrastructure is managed; you only need to think about how you will use that particular piece of software. A common example of a SaaS application is web-based email which you can use to send and receive email without having to manage feature additions to the email product or maintain the servers and operating systems that the email program is running on.

Global Infrastructure

AWS serves over a million active customers in more than 240 countries and territories. We are steadily expanding global infrastructure to help our customers achieve lower latency and higher throughput, and to ensure that their data resides only in the AWS Region they specify. As our customers grow their businesses, AWS will continue to provide infrastructure that meets their global requirements. The AWS Cloud infrastructure is built around AWS Regions and Availability Zones. An AWS Region is a physical location in the world where we have multiple Availability Zones. Availability Zones consist of one or more discrete data centers, each with redundant power, networking, and connectivity, housed in separate facilities. These Availability Zones offer you the ability to operate production applications and databases that are more highly available, fault tolerant, and scalable than would be possible from a single data center. The AWS Cloud operates in 80 Availability Zones within 25 geographic Regions around the world, with announced plans for more Availability Zones and Regions. For more information on the AWS Cloud Availability Zones and AWS Regions, see AWS Global Infrastructure. Each Amazon Region is designed to be completely isolated from the other Amazon Regions. This achieves the greatest possible fault tolerance and stability. Each Availability Zone is isolated, but the Availability Zones in a Region are connected through low-latency links. AWS provides you with the flexibility to place instances and store data within multiple geographic regions as well as across multiple Availability Zones within each AWS Region. Each Availability Zone is designed as an independent failure zone. This means that Availability Zones are physically separated within a typical metropolitan region and are located in lower risk flood plains (specific flood zone categorization varies by AWS Region). In addition to discrete uninterruptible power supply (UPS) and onsite backup generation facilities, data centers located in different Availability Zones are designed to be supplied by independent substations to reduce the risk of an event on the power grid impacting more than one Availability Zone. Availability Zones are all redundantly connected to multiple tier-1 transit providers.

Security and Compliance

Security

Cloud security at AWS is the highest priority. As an AWS customer, you will benefit from a data center and network architecture built to meet the requirements of the most security-sensitive organizations.

Security in the cloud is much like security in your on-premises data centers—only without the costs of maintaining facilities and hardware. In the cloud, you don’t have to manage physical servers or storage devices. Instead, you use software-based security tools to monitor and protect the flow of information into and out of your cloud resources.

An advantage of the AWS Cloud is that it allows you to scale and innovate, while maintaining a secure environment and paying only for the services you use. This means that you can have the security you need at a lower cost than in an on-premises environment.

As an AWS customer you inherit all the best practices of AWS policies, architecture, and operational processes built to satisfy the requirements of our most security-sensitive customers. Get the flexibility and agility you need in security controls.

The AWS Cloud enables a shared responsibility model. While AWS manages security of the cloud, you are responsible for security in the cloud. This means that you retain control of the security you choose to implement to protect your own content, platform, applications, systems, and networks no differently than you would in an on-site data center. AWS provides you with guidance and expertise through online resources, personnel, and partners.

AWS provides you with advisories for current issues, plus you have the opportunity to work with AWS when you encounter security issues. You get access to hundreds of tools and features to help you to meet your security objectives. AWS provides security-specific tools and features across network security, configuration management, access control, and data encryption.

Finally, AWS environments are continuously audited, with certifications from accreditation bodies across geographies and verticals. In the AWS environment, you can take advantage of automated tools for asset inventory and privileged access reporting.

Benefits of AWS Security

• **Keep Your Data Safe:** The AWS infrastructure puts strong safeguards in place to help protect your privacy. All data is stored in highly secure AWS data centers.

• **Meet Compliance Requirements:** AWS manages dozens of compliance programs in its infrastructure. This means that segments of your compliance have already been completed.
• **Save Money:** Cut costs by using AWS data centers. Maintain the highest standard of security without having to manage your own facility

• **Scale Quickly:** Security scales with your AWS Cloud usage. No matter the size of your business, the AWS infrastructure is designed to keep your data safe

Compliance

AWS Cloud Compliance enables you to understand the robust controls in place at AWS to maintain security and data protection in the cloud. As systems are built on top of AWS Cloud infrastructure,compliance responsibilities will be shared. By tying together governance-focused, audit-friendly service features with applicable compliance or audit standards, AWS Compliance enablers build on traditional programs. This helps customers to establish and operate in an AWS security control environment.

The IT infrastructure that AWS provides to its customers is designed and managed in alignment with best security practices and a variety of IT security standards. The following is a partial list of assurance programs with which AWS complies:
• SOC 1/ISAE 3402, SOC 2, SOC 3
• FISMA, DIACAP, and FedRAMP
• PCI DSS Level 1
• ISO 9001, ISO 27001, ISO 27017, ISO 27018
AWS provides customers a wide range of information on its IT control environment in whitepapers, reports, certifications, accreditations, and other third-party attestat

AWS Management Console

Access and manage Amazon Web Services through the AWS Management Console, a simple and intuitive user interface. You can also use the AWS Console Mobile Application to quickly view resources on the go.

AWS Command Line Interface

The AWS Command Line Interface (CLI) is a unified tool to manage your AWS services. With just one tool to download and configure, you can control multiple AWS services from the command line and automate them through scripts.

Software Development Kits

Aws Software Development Kits (SDKs) simplify using AWS services in your applications with an Application Program Interface (API) tailored to your programming language or platform

Amazon EC2

Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides secure, resizable compute capacity in the cloud. It is designed to make web-scale computing easier for developers. The simple web interface of Amazon EC2 allows you to obtain and configure capacity with minimal friction. It provides you with complete control of your computing resources and lets you run on Amazon's proven computing environment. Amazon EC2 reduces the time required to obtain and boot new server instances (called Amazon EC2 instances) to minutes, allowing you to quickly scale capacity, both up and down, as your computing requirements change. Amazon EC2 changes the economics of computing by allowing you to pay only for capacity that you actually use. Amazon EC2 provides developers and system administrators the tools to build failure resilient applications and isolate themselves from common failure scenarios.

Amazon EC2 Auto Scaling

Amazon EC2 Auto Scaling helps you maintain application availability and allows you to automatically add or remove EC2 instances according to conditions you define. You can use the fleet management features of Amazon EC2 Auto Scaling to maintain the health and

availability of your fleet. You can also use the dynamic and predictive scaling features of

AWS Lambda

AWS Lambda lets you run code without provisioning or managing servers. You pay only for the compute time you consume—there is no charge when your code is not running. With Lambda, you can run code for virtually any type of application or backend service—all with zero administration. Just upload your code, and Lambda takes care of everything required to run and scale your code with high availability. You can set up your code to automatically trigger from other AWS services, or you can call it directly from any web or mobile app

Storage on AWS

AWS storage services are grouped into three different categories: block storage, file storage, and object storage.

File Storage
You may be familiar with file storage if you've interacted with file storage systems like Windows File Explorer or Finder on MacOS. You place your files in a tree-like hierarchy that consists of folders and subfolders. For example, if you have hundreds of cat photos on your laptop, you may want to create a folder called Cat photos, and place those images inside that folder to organize them. Since you know these images will be used in an application, you may want to place the cat photos folder inside another folder called Application files.

Common use cases for file storage include:

- Large content repositories
- Development environments
- User home directories

Block Storage

While file storage treats files as a singular unit, block storage splits files into fixed-size chunks of data called blocks that have their own addresses. Since each block is addressable, blocks can be retrieved efficiently.

When data is requested, these addresses are used by the storage system to organize the blocks in the correct order to form a complete file to present back to the requestor. Outside of the address, there is no additional metadata associated with each block. So, when you want to change a character in a file, you just change the block, or the piece of the file that contains the character. This ease of access is why block storage solutions are fast and use less bandwidth.

Object Storage

Objects, much like files, are also treated as a single unit of data when stored. However, Unlike file storage, these objects are stored in a flat structure instead of a hierarchy. Each object is a file with a unique identifier. This identifier, along with any additional metadata, is bundled with the data and stored.
Changing just one character in an object is more difficult than with block storage. When you want to change one character in a file, the entire file must be updated.

Amazon EC2 Instance Store

Instance store is ephemeral block storage. This is preconfigured storage that exists on the same physical server that hosts the EC2 instance and cannot be detached from Amazon EC2. You can think of it as a built-in drive for your EC2 instance. Instance stores are generally well-suited for temporary storage of information that is constantly changing, such as buffers, caches, and scratch data. It is not meant for data that is persistent or long-lasting.

Amazon S3

If your data doesn't change that often, Amazon S3 might be a more cost-effective and scalable storage solution. S3 is ideal for storing static web content and media, backups and archiving, data for analytics, and can even be used to host entire static websites with custom domain names.

Here are a few important features of Amazon S3 to know about when comparing it to other services.

- It is object storage.
- You pay for what you use (you don't have to provision storage in advance).
- Amazon S3 replicates your objects across multiple Availability Zones in a Region.
- Amazon S3 is not storage attached to computers.

Monitoring on AWS

When operating a website like the Employee Directory Application on AWS you may have questions like:

- How many people are visiting my site day to day?
- How can I track the number of visitors over time?
- How will I know if the website is having performance or availability issues?
- What happens if my Amazon Elastic Compute Cloud (EC2) instance runs out of capacity?
- Will I be alerted if my website goes down

Amazon CloudWatch

How CloudWatch Works

The great thing about CloudWatch is that all you need to get started is an AWS account. It is a managed service, which enables you to focus on monitoring, without managing any underlying infrastructure. The employee directory app is built with various AWS services working together as building blocks. It would be difficult to monitor all of these different services independently, so CloudWatch acts as one centralized place where metrics are gathered and analyzed.
You already learned how EC2 instances post CPU utilization as a metric to CloudWatch. Different AWS resources post different metrics that you can monitor. You can view a list of services that send metrics to CloudWatch in the resources section of this unit.

Load Balancer

Load balancing refers to the process of distributing tasks across a set of resources. In the case of the corporate directory application, the resources are EC2 instances that host the application, and the tasks are the different requests being sent. It's time to distribute the requests across all the servers hosting the application using a load balancer.
To do this, you first need to enable the load balancer to take all of the traffic and redirect it to the backend servers are based on an algorithm. The most popular algorithm is round-robin, which
sends the traffic to each server one after the other.
A typical request for the application would start from the browser of the client. It's sent to a load balancer. Then, it's sent to one of the EC2 instances that hosts the application. The return traffic would go back through the load balancer and back to the client browser. Thus, The load balancer is directly in the path of the traffic.

Conclusion

Analysis of video and activity prediction allow the system to give proper recommendations based on the proper management of the field.
The real time surveillance along with provided activity data ensure proper working environment according to the application.

References

[1] R. Girshick, Fast R-CNN, arXiv:1504.08083v2, 27 Sep 2015.

[2] S. Ren, K. He, R. Girshick, and J. Sun, Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, arXiv:1506.01497v3, 6 Jan 2016.

[3] T. Karmarkar(2018, Aug 19), Region Proposal Network (RPN) — Backbone of Faster R-CNN, [Online]. Available: https://medium.com/egen/region-proposal-network-rpn-backbone-of-faster-r-cnn-4a744a38d7f9