A Survey of Popular Image and Text analysis Techniques

Rahul Suresh[#1], Keshava N[#2]

[#]*Department of Computer Science and Engineering, RV College of Engineering*
R V Vidyanikethan Post, Mysuru Road, Bengaluru - 560 059, Karnataka, India
[1]`rahuls.cs15@rvce.edu.in`
[2]`keshavankeshavn.cs15@rvce.edu.in`

***Abstract*—Image processing has made huge progress in recent times and has captured the attention of the international research community in recent times. In this survey paper the winning Convolutional Neural Network (CNN) architectures of the popular ImageNet Large Scale Visual Recognition Competition (ILSVRC) competition along with the innovations they introduced are discussed in detail. CNNs to the likes of the ZFNet, Inception V1, V2, V3 and V4 versions, ResNet versions, Inception ResNet versions, VGG versions are all covered. Also object detection is looked at as a preeminent task and popular models like RCNN, Fast RCNN, Faster RCNN and the YOLO versions along with their award winning architectures are discussed. Text classification is another major practice that is followed before making business decisions. In this survey paper section III explains the text classification techniques. preprocessing techniques such as removing stop words, stemming, lemmatization. Two variants of vectorization method such as vectorization-delta, vectorization-center. paper also explains the hyperparameter tuning for optimization of support vector machine(SVM), Logistic regression(LR), random forest(RF) and Boosted Regression Tree (BRT). Explored fine tuning parameter which enhance the performance of machine learning models.**

***Keywords*—Image processing; Image Analysis; Image Classification; Convolutional Neural Network; CNN; ILSVRC; ZFNet; Inception V1; Inception V2; Inception V3; GoogLeNet; Inception V4; Inception ResNet V1; Inception ResNet V2; VGG 11; VGG 11 LRN; VGG 13; VGG 16; VGG 19; ResNet; ResNeXT; Object detection; RCNN; Fast RCNN; Faster RCNN; YOLO; YOLO V2; YOLO 9000; YOLO V3; Text Analysis; Document - classDistance; DCD; Support Vector Machine; SVM; Logistic Regression; LR; Random Forest; RF; Vectorization;**

I. INTRODUCTION

A. Image Processing:

The task of image processing has garnered global interest recently and the field has seen groundbreaking innovations. This large scale attention was partially due to the resurgent interest in deep learning based architectures applied to image processing tasks with the success of AlexNet at ILSVRC being an example. With the advent of affordable smartphone cameras and public sharing platforms, visual content, especially images being generated at a massive rate in today's world. Thus processing them has massive value and in this paper popular models that do the same are discussed.

First image classification is looked on at a task and popular deep learning architectures in the field are discussed. The convolutional neural networks discussed are all award winning submissions to ILSVRC. The innovations, ideas and architectures that enabled them to do so are discussed in detail. CNNs discussed encompass the ZFNet, Inception V1, V2, V3/ GoogLeNet, V4 versions, ResNet, ResNeXt , Inception ResNet V1, V2 versions and the VGG 11, 11 LRN, 13, 15, 19 versions. Next, object detection is looked at as a task and popular architectures with the innovations they ushered in are discussed. Models discussed include Region CNN (RCNN), Fast RCNN, Faster RCNN, YOLO V1, YOLO V2, YOLO 9000 and YOLO V3.

B. Text Processing:

Information available for text attribute extraction is in the form of url, description and basic attribute-value pairs like color, size. This information is given either from the delivery or crawling team. From this given data which may contain more thousands of products description which contains both structured and unstructured data is fed to the machine learning technique called natural language processing(NLP), i.e for the preprocessing of data to clean.

Preprocessed text data is then converted into vectorized form using count-vectorizer/ TfidfVectorizer methods. Then it is fed to different machine learning models such as, support vector machine, logistic regression, Multinomial-NB and KneighborsClassifier to extract the relevant target attributes from a given set of classes. For attribute extraction from store shelf images/videos, Machine Learning (ML) models would be used. A preprocessing step for videos would be to get the most optically stable frame. Object detection can be done using transfer learning on existing ML models. The detected tags would be pre-processed to normalize pixel values. Then, Optical Character Recognition would be performed to obtain item name, price and any discount offered.

II. Survey of Image Processing Techniques

The task of object detection in simple terms is identifying objects in an image as well as identifying where they are present in the image by drawing a bounding box around images. In the field of object detection RCNN [1] is a popular model. RCNN uses a method called selective search [2] to extract around 2000 region proposals per image. Using selective search is much better than using naive localization algorithms that generate a computationally impractical number of regions. Selective search proceeds by an over segmented image as the starting seed. Then iteratively it adds bounding boxes of image segments to the proposal list, before grouping similar images based on pixel color, intensity, etc. The result of running the selective search algorithm is around 2000 regions of interest. These regions of interest are then individually warped and resized and are fed to the CNN feature extractor. The CNN like say Alexnet [3] or inception versions - V1 [4], V2 [5], V3 [5], V4[6] could be pre-trained on the Imagent [7] dataset. The inception architectures, themselves are a departure form the earlier approach of making a network deeper to get better performance. The problem plaguing CNNs are the nature of the object in the image itself. In other words the as the size of the object varies the salient area in the image also varies. For smaller objects a smaller kernel is preferred and for larger objects a larger kernel is preferred. Very deep networks are prone to overfitting and it also becomes harder to pass gradient updates through the layers apart from making it computationally expensive.

Inception aims to solve this problem by making the network wider than deeper, i,e, in the same layer multiple convolutions with different kernel sizes are performed and all the outputs are concatenated. The logic behind this being that by replacing densely connected layers with sparse ones computational complexity is reduced. The is especially seen in GoogleLeNet or Inception V1 [4]. In the naive inception layer the output of the previous layer is connected to 1x1, 3x3, 5x5, 3x3 max pooling layer in parallel. The output of these layers is concatenated and reduced to a single vector, to be fed into the next layer. Given the naive inception later, a 1x1 convolution filter is added to reduce input dimensionality. Its is added before the 3x3, 5x5 conv layer and then after the max pooling layer. A notion that the current layer should pay attention to the previous layer and its learnings is used. So as the saliency region in the image changes the model automatically chooses a larger filter size for a larger saliency region and a smaller filter for a smaller saliency size. Among the other unique features of the network is the auxiliary classifiers in the middle of the network. This forces the middle layers of the network to be active and learn important features. The final loss metric was calculated as output of the overall model + 0.3 x auxiliary loss. Therefore inception V1 has 9 inception modules which brings the layer count to 22 and then 5 pooling layers, so 27 layers all in all.

Inception V2 [5] had a couple of modifications to bring down the computational complexity and improve the accuracy. Representation of the images through the CNN tended to be a bottleneck with convolutions drastically reducing the dimensions of the image, causing a loss of information aptly termed "Representational bottleneck" in Inception V1. V2 first off, reduced the computational complexity of the 5x5 convolution by stacking two 3x3 convolutions. The next improvement was converting a nxn convolution operation into a 1xn and nx1 convolution, which one again worked in favour of reducing computational complexity. The next upgrade was to make the network even wider by running the 1xn and nx1 in parallel and concatenating the output together. For Inception V3 [5] also released in the same paper, the auxiliary classifiers were thought of as regularizers as it was observed that they didn't contribute much to the training process other than at the very end when accuracy maxed out and flattened. So the newly created V3 had all the pros of the V2 while additionally incorporating RMSprop as the optimizer and 7x7 convolutions which were factorized. In inception V4 [6] the modules were made more uniform, removing unnecessary complications and making it faster to train. The initial convolutions applied before introducing the inception layers, also known as the stem of the model, was modified. Additionally, the inception modules were divided into 3 classes - A, B, C depending on the sub modules. Furthermore specialized reduction blocks were introduced, again of 3 subclasses - A, B, C, to change the height, width of the grid.

Inception ResNet [6] is a hybrid of ResNet [8] and the inception modules. ResNet introduced the concept of skip connections - similar to gated recurrent units, and heavy batch normalization. Thus despite having 152 layers the time it takes to train is still lesser than the time taken by VGG [9]. ResNet itself won first place in ICLR 2015, with a top 5 accuracy metric of 3.57% surpassing human performance. ResNet mainly tackled the problem of degradation of CNNs after initial convergence. If h(x) represents the mapping from x to y or inputs to output that a CNN is ideally supposed to learn, the authors defined a function called residual function. $f(x) = h(x) - x$. The authors propose that optimizing f(x) is simpler than optimizing h(x). This is what gives wind to residual blocks where the input is fed into the output, unlike a plain block and thus for the stacked neural network layers between the input and the output in case of residual blocks they learn to predict 0 much easier than they learn to predict x for plain blocks. All this is under the assumption that identity mapping is ideal. The authors made a couple of example cases to test the assumption. A plain 18 layer VGG 19 was compared side by side along with two 34 layer deeper versions - one plain and the other with residual layers. The results as expected showed that the model using residual layers performed much better than the one without it. The residual layers were so effective that its benefit was felt even at the level of the original VGG 19. Two types of residual connections were mainly used in the network. One is a direct identity shortcut [10] when input and output are of the same dimensions. The other kind is used when there is a change of dimensions, zero padding in case of increased dimensions, downscaled using 1x1 convolutions in case of reduced size.

For further studies, ResNet 2 layer and 3 layer blocks were studied all pointing towards the undoubted superiority of ResNet. More formally, ResNet converges quicker than a plain network of the same size. Also in case of the projection versus identity shortcuts, only identity shortcuts were found to give a significant gain on the performance and thus as a result projection shortcuts were only used there was a absolute need to use them. Furthermore the problem of vanishing gradient which has been present in all deep CNN architectures is not present in ResNet [10]. ResNeXt [11] is a further improvement on the ResNet architecture. ResNeXt ended up taking the runner up position in ICLR 2016. This paper introduces the concept of cardinality. Apart from revealing the complications if the Inception networks this paper showcases the simplicity of ResNet. ResNeXt tests a strategy of split-aggregate while performing the convolutions. The cardinality of a ResNet Block is the number of split paths in it. All paths consist of the same sub modules. The idea behind this entire concept being that instead of having the more width or depth, having higher cardinality helps reduce the validation error. A bottleneck is the term given to such a unit with cardinality. Popular variations of the network include one with four bottlenecks each with 32 cardinality, called resnext_32*4d. Further studies indicated the in the question of width versus cardinality, decreasing the width to increase the cardinality has a favourable effect on the overall accuracy of the model.

VGG [9] is the runner up of the 2014 ILSVRC competition, beaten only by GoogLeNet [4]. Different VGG versions including VGG 16 and VGG 19 exist. The complete ImageNet dataset consists of 15 million labeled images with categories above 22,0000. ILSVRC uses a subset of around 1000 images in each of a 1000 classes. VGG successfully used multiple 3x3 filters to represent a filter with higher receptive field like say 5x5 or 7x7. Thus the need to use a larger kernel like say 11x11 in AlexNet [2] or a 7x7 kernel in ZFNet [12] which are computationally much more expensive is avoided. Therefore, quicker convergence is brought about with lowered probability of overfitting due to reduced number of parameters. VGG 11 straight off achieves a top 5 error of 10.4 % which is similar to the score of ZFNet. VGG 11 LRN version, with LRN standing for Local Response Normalization as suggested by AlexNet, the error in fact rises by 0.1%. Therefore in other versions batch normalization is used instead of LRN. Batch normalization with its effects of reducing covariance shift and slight regularization effects is prefered. With VGG 13 the error drops to 9.9% showing the positive effect of adding extra conv layers. In VGG 16 conv1, adding conv1 performs projection in the same high dimension a technique aptly called network in network [13]. Reinforcing the benefits of still adding more layers VGG 16 without conv1 gets an error rate of only 8.8%, but a limit is reached in VGG 19 with error reaching 9.0%. Taking note of the fact that typically in an image an object may be present at various scales, multi scale training is used. Normally in single scale training, the training image is resized to around 256 or 384 pixels per side. Then an input size of 224x224 is cropped from the centre of the image and fed to the network. For multi scale training the image is scaled into a number of image with sides ranging from 256 to 512 px and then all are cropped into 224x224 and fed into the network. Thus the network sees different scales of the same image, leading to better performance during training.

During testing as well this strategy can be implemented to scale the images into a range before running VGG to obtain inferences. In this manner the error is reduced and using both multiscale testing and training the best results are obtained. For example in VGG 16 the error is dropped from 8.6% using plain approach to around 7.5% using multiscale testing and training. Further modifications [14] such as replacing the fully connected layer with convolutional layers too is purported to have some benefits. Also while testing, including the horizontal flips, corners etc, can decrease the slightly. Overall applying all the above said modifications and combining VGG 16 and VGG 19 gives the lowest error metric of around 6.8%. Now, VGG achieves an accuracy of 6.8% using just 2 networks as compared to GoogLeNet which uses 7 networks to obtain an error of 6.7%. If only a single net is used Inception V1 / GoogLeNet gives an error of 7.9% while VGG gives around 7.2%. ZFNet [12], became the winner of ILSVRC, by visualizing the convnet and based on inferences drawn from such visualization, upgraded AlexNet to achieve lower error rates. The first step to visualize a conv layer is by applying a deconv operation. In normal convolution the series of operations applied are convolution, non linearity application and pooling. To visualize a feature map in pixel space, the inverse of the above operations in the reverse order has to be applied. To make a CNN fit for visualization, extra code has to be added to state of pixels, before operation is applied. Unpooling is one such step in which the positions of maxima in the input is recorded and recreated to create an approximate inverse.

The case of reversing the non-linearity is simplified for Relu since it keeps positive values unchanged and reduces negative value to 0. In other words, simply applying another Relu operation, will reverse the operation. The reverse of convolution, deconvolution is performed by zero padding the conv output to a size greater than the original input and applying the conv filters again. Now, using the above set of processes given a layer the set of images that activate it the most i.e produce the most active feature maps is reconstructed. Layer 1 of AlexNet was found to learn only extremely low frequency and high frequency information missing on the middle frequencies. As a result the layers more deeper only see information as seen by layer 1 and thus also miss out on the middle frequencies. Layer 2 suffers from aliasing which occurs due to the fact that the sampling frequency is too low because of the large stride of 4 used in layer 1. Layer 3 is found to learn patterns and texture, while layer 4 is more object specific features like faces of cats etc. Only in layer 5 is the full object looked upon. ZFNet was therefore changed based on insights from AlexNet. The size of the kernel/ filter in the first layer was reduced from 11x11 to 7X7. Also the stride in the first layer was reduced from 4 to 2. As a result of all these modifications. ZFNet got an top 5 error of around 16% which as of 2013 won it ILSVRC.

Inception Resnet [6] has two versions V1 and V2 and is a hybrid of the ResNet and Inception architectures. Between them, they have different stems - initial conv layers before the inception blocks and different hyperparameter settings. Also in terms of cost of computation and training time Inception ResNet V1 is similar to Inception V3 and Inception ResNet V2 is similar to Inception V4. As discussed earlier the reduction blocks A, B, C remain the same. The identity mapping scheme is used and 1x1 convolutions are introduced as necessary to match the depth and therefore the shape of the input and output of the skip connections. The pooling connections remained in the skip connections took over and replaced the pooling connection in the inception block. The residual activations were scaled by a range of 0.1 to 0.3 to keep the network in the middle from dying, this tended to happen when the number of filters exceeded a 1000. Overall the effect of this hybrid architecture was felt in the fact that the model attained higher accuracies at a lower epoch.

Coming back to RCNN, like said earlier, after running selective search to extract the region proposal all the (resized) region crops are fed to the pretrained CNN - any one of the many discussed above - to extract all the features. The last layer of the CNN must be retrained with the objects that are to be detected along with an class that represents no object being detected. Alternatively a SVM for each class of objects to be detected can be trained on the features obtained from the CNN. A popular approach is to train one binary SVM for each class with the outputs being either object present or not present. The features are also used to train a bounding box regressor that outputs a correction factor (as the input itself is a region of interest and not the entire image). The biggest drawback of the RCNN is the time it takes to train and predict. Each step in the above described process is slow. Thus while RCNN achieved a high accuracy on object detection tasks along with a high IOU it isn't widely used. IOU which stands for intersection over union is a metric where the intersection between the correct bounding box and the predicted bounding box is taken. The IOU value is set as the minimum threshold value that all predictions are greater than. To improve upon RCNN, fast RCNN [15] was released. The improvement was that all the predictions of the selective search algorithm are fed to a region of interest (ROI) pooling layer. This layer is attached after the Convolutional feature extractor and takes in the regions. It then extracts the corresponding features from the feature map obtained by running the convolutional feature extractor on the entire image, resizes them before passing them one by one to the classification and bounding box regression heads. Thus in this manner the convolutional feature extractor is run only once per image rather than 2000 times per image as in RCNN. The test time for example per image is reduced to about lesser than 1/25 of the time taken in naive RCNN. Though a massive improvement in training and prediction time, this model still uses a external region proposal model.

Faster RCNN [16] is a further improvement that addresses the issue of an external region proposal algorithm. The compute intensive and slow selective search algorithm was replaced with a network which learnt the regions on its own. This network called the Region Proposal Network (RPN) replaces the selective search algorithm and forwards all the region proposal to the ROI pooling layer which functions just the same as in Fast RCNN. In concept the last few layers of the convolutional feature extractor is used to extract the region proposals. So the training process of the Faster RCNN networks is to, first, use a pretrained CNN specifically the last convolutional layer to extract features from an image. Then to train a RPN to determine the bounding box of an object in the image, if an object is present. The outputs of the RPN are fed to a custom layer which creates object proposals. Specifically it clips the image using the boxes, runs sanity checks. Such checks include removing boxes with height or width greater than a threshold, sorting proposals by confidence values in the descending order. It then selects top N proposals, applies Non Maximum Suppression (NMS) then choose top N again and then sends both sets to the region proposal network. The RPN can be separately trained by applying a classification loss on whether an object is present or not and a bounding box regression loss for the bounding box region proposals. More intuitively the RPN slides a 3x3 window on the feature map, and generates K anchor boxes from the center of the window, classifies those boxes as objects or not along with giving bounding box corrections. In addition to training the RPN the main network is trained using the classification loss for each class and bounding box regression loss for the predicted bounding boxes. The two subnetworks, the RPN and the main network can be trained either separately or together. The time improvement is massive, slashing the training time as compared to RCNN to 1/250 times it. The accuracy of this model is similar to Fast RCNN and higher than the naive RCNN. The Mean Average Precision (mAP) on the PASCAL VOC dataset [17] for example for RCNN is 66.0, whereas for Fast RCNN and Faster RCNN it is around 66.9.

YOLO [18] is an other popular object detection model. YOLO works by creating a NxN grid from the input image. Each cell in the grid predicts a single image. The cell then makes a certain fixed number of bounding box predictions. As each cell can predict only one object YOLO can not detect objects that are very close to each other. Specifically for each cell YOLO predicts 'K' number of bounding boxes, detects one object and predicts 'C' number of conditional classes. Each bounding box prediction consists of a confidence score reflecting the likelihood of the box containing an object and the accurateness of the bounding box and the starting x, y coordinates followed by the height h and width w of the bounding box. Since the entire image is normalized with its own width and height, x,y,w and h are in the scale of 0 to 1. YOLO on the whole uses a CNN to predict a tensor in the shape of (N,N,Kx5,C). Once the tensor is obtained two fully connected layers are used to predict NxNxK bounding boxes. The final predictions are made only by keeping the predictions where the confidence values are greater than a particular threshold. The class confidence score for each prediction box is predicted as the product of box confidence score and conditional class probability. It represents the confidence of classification and localization. In terms of architecture YOLO has 24 convolutional layers and two fully connected layers. To decrease the depth of the feature map a 1x1 reduction layer is utilized. Fast YOLO is an other spinoff that uses only 9 convolutional layers for a much faster performance but at reduced accuracy . Many boxes are predicted for each cell of the grid, to compute loss for true positives, the prediction with the highest IOU with the ground truth is selected. The final loss for YOLO is the sum of squared error difference between the predictions and the ground truth. The final loss function comprises of first, a classification loss, which is the squared error of the predicted conditional class probabilities for each class, assuming the presence of an object. Second a localization loss is used which represents the error in the bounding box prediction locations and sizes. To avoid loss being affected by the size of the bounding box i.e a 1 pixel difference is penalized by the same loss in case of a small or large bounding box, YOLO is designed to predict the square root of the bounding box height and width.

The confidence loss varies by whether an object is present in a bounding box or not. To avoid an imbalanced problem due to the presence of more object absences than presences a weight factor is used on this loss, the default being 0.5. To avoid duplicate detections of the same object non maximal suppression is applied to weed out duplicates with lower values of confidence. This also adds to the mAP of the model in tests by around 2-3%. The advantages of using yolo are the speed and end to end nature of the network . The end to end nature removes external dependencies making the network fast which in turn enables it to be used in real-time. Another important advantage is that in YOLO the network has access to the entire image leading to fewer false positives in the background area of the image. In external region proposal systems the network is limited to the regions and thus can't use the context of the object in the complete image. YOLO V2 [19] is the successor to YOLO with the aim of making the model quicker while increasing the accuracy of the predictions. The main drawback of YOLO was that it had a higher localization error compared to similar region based object detectors. The main steps for accuracy improvement taken included - addition of batch normalization which enables dropout to be removed boosting the mAP by 2% and the use of a higher resolution classifier. In YOLO a classifier CNN was taken trained with 244x244 images. Then the last fully connected layers were replaced with convolutional layers and the network was trained end to end with images of size 448x448. In YOLO V2 the classifier is trained with 244x244 images initially and then retrained with images of size 448x448 for a fewer number of epochs.

This simplified the object detection training that followed later and increased the mAP by an entire 4%. In YOLO the network initially makes predictions of the bounding box that are all around the place. This leads to unstable gradients in the initial stages of training. For YOLO V2 it was observed that in real life the ratios of sides of bounding boxes wasn't random as seen in the case of cars and even pedestrians. Therefore specific shapes of bounding boxes are used from the beginning with the network predicting offsets to each of the boxes. Also in the network the offsets are constrained to have each prediction target a specific shape. Multiscale training is also used to improve accuracy. Overall as a result of including all of these improvements the mAP in the VOC dataset is improved to 78.6%. YOLO 9000 [19] is an augmentation to YOLO to detect north of 9000 object classes by adding top 9000 classes from ImageNet with COCO [20] classes. YOLO V3 [21] is a further improvement based on the YOLO platform. YOLO like the other object detection models applies softmax to convert class scores into probabilities totalling 1. YOLO V3 on the other hand makes no assumption that the objects in the should be mutually exclusive and instead applies multilabel classification and uses independent logistic classifiers to calculate the score of an object belonging to each class. The squared error loss function used earlier is replaced with a class wise binary cross-entropy loss function. YOLO V3 overall has higher localization error but is better at detecting small objects. YOLO V3 is mainly built for speed and uses Darknet - an open source neural network - as the classifier.

III. Survey of Text processing techniques

Text mining is the field of extracting the information from the text data for given class labels. One of the supervised learning technique called DC Distance[22]. Document Class Distance algorithm is a supervised feature extraction and additionally proposed a reduction algorithm. Algorithm preprocess the unstructured data followed by the vectorization[22]. Vectorization converts the human readable data into the machine readable format. DCD algorithm outperforms in reducing the feature extraction to class labels and maximizing the accuracy of the model. One of the classification model Naive Bayes [23] [26] is a supervised and probabilistic way of classifying the text data based on the trained data. Naive Bayes algorithm is based on Bayes theorem, which is robust, accurate and fast which is used in the classification or filtering of spam-emails [24], diagnosis of treatment and decision making based on the process output [25]. Naive Bayes model is performed with a sequence of steps as follows, Feature Engineering [23] [26], calculating the probability for each word [23]. A technique called laplace smoothing [26] is used for regularizing the Naive Bayes classification which maintains a Pseudo-Count [26] for each probability therefore there will be no zero probability. Most of the time data scientist spend 80% of time on collecting, cleaning and organizing the data from the raw or unstructured data. Preprocessing [27] is the most important process for classifying the categorical data. FastText [27] is the open source word-vectorization representation. Different techniques of preprocessing is tested on four different models such as Multinomial Naive Bayes, Extreme Gradient Boost and LSTM models, which has resulted with the variant accuracies w.r.t to preprocessing.

Proposed that training of a model without any transformation, produced a decent accuracy. By following the preprocessing [27] steps produced worse accuracy. Technically, preprocessing place an important role for classification, decision making for business problem. One of another efficient method other than preprocessing [28] is proposed that converts raw data into low-level feature dimension called "Binary unique number of words" also called "BUNOW". This is dictionary based approach where each unique word will store unique Id in as dictionary that is represented as k-dimensional vector in binary encoding. Generated vector is fed into CNN [29] for classification. AG's news dataset is used to test the accuracy of the CNN module. A decent accuracy of 91.99% is achieved and an error of 8.01% is observed. After all the preprocessing of raw data next step is the vectorization [30], Relevant word order vectorization will determine the structure of the most relevant words and predict the class of text data. Concept of the vectorization is to convert the human readable text into machine readable format, convert data into matrix format i.e in mxn dimensionality. Data used for this Relevant word order vectorization (RWOV) was Electronic health Record (EHR) [30] which is an unstructured raw data which describes the characteristic of a patients. Therefore, focus is made on the terms which occurs repeatedly and describe the symptoms or disease of a patient. It has been observed that, only a fraction of terms indicate the meaning in relation to the term of Interest. RWOV creates a matrix, where each indicates a subject and each column has words. Entire matrix is stored with null values. Frequently occuring word will have the highest score and inverse of that word count. It is observed that, high accuracy is observed with ER,PR and HER2 parameter [30] or terms of an electronic health record status compared to the SVM or Neural Network vectorization methods. Going back to the vectorization, text label Extraction from the unstructured data is resulted a decent accuracy for the models that are in this project. Term Frequency - Inverse Document Frequency [31] vectorization method calculates the frequency of each word in a document. In a layman term, It is the number of times a word appears in document.

Problem with the term frequency, some of the frequently occuring word like"the" will have the highest score, this problem is overcome by the Inverse document frequency (IDF). It is total number of documents to the total number of times word appears in that document. TF-IDF vectorizer [31] used two methods, smooth inverse frequency and sub-sampling function to improve the vectorization method by embedding. Smooth Inverse frequency method is lagged on considering the more than 300 words. Two variants of IDF such idf-center and idf-delta, idf-center has performed for the smaller length of words and the idf-delta has performed better for larger length of words. Coming to the classification and regression method support vector machine which is a supervised machine learning algorithm method has given decent accuracy on label extraction. SVM is the one of the most commonly used for classification problem.when it comes complex data choosing a hyperplane becomes a challenge. To tune the hyper parameters [32] specific to the SVM, [32] explains how choose the kernels or address the spatial autocorrelation, hyper parameter tuning for the optimization of algorithm. These are addressed for the models such as support vector machine, K-Nearest Neighbour (KNN), Random forest (RF), Logistic Regression(GLR) and Boosted Regression Tree (BRT).

IV. Conclusions

Various image classification models and the novelties, improvements that they brought to the field were discussed in detail. The performance gains with each new model was also looked at. Additionally popular object detection models and their new methods and approaches were also explicated. explored the various techniques of preprocessing of raw unstructured data. explored the some of the fine hyperparameter tuning which enhance the performance of the models.

Acknowledgment

The guidance of Prof. Chaitra B.H., Assistant Professor, RV College of Engineering, towards completing this survey is greatly appreciated.

References

[1] Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik,"Rich feature hierarchies for accurate object detection and semantic segmentation", *IEEE Conference on Computer Vision and Pattern Recognition, Columbus*, 2014, pp. 580-587

[2] J.R.R. Uijlings, K.E.A. van de Sande, T. Gevers, A.W.M. Smeulders, "Selective Search for Object Recognition", *International Journal of Computer Vision,* 2013, Volume 104, pp. 154-171

[3] Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks", *Proceedings of the 25th International Conference on Neural Information Processing Systems, Lake Tahoe*, 2012, pp. 1097-1105

[4] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, Zbigniew Wojna, "Going deeper with convolutions", *IEEE Conference on Computer Vision and Pattern Recognition, Boston,* 2015, pp. 1-9

[5] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, Zbigniew Wojna, "Rethinking the Inception Architecture for Computer Vision", *IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas,* 2016, pp. 2818-2826

[6] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, Alex Alemi, "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning", *Proceedings of the Thirty-First Conference on Artificial Intelligence, San Francisco*, 2017, pp. 4278-4284

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li and Li Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database", *IEEE Conference on Computer Vision and Pattern Recognition, Miami Beach*, 2009, pp. 248-255

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, "Deep Residual Learning for Image Recognition", *IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas*, 2016, pp. 770-778

[9] Karen Simonyan, Andrew Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition", *International Conference on Learning Representations, San Diego*, 2015

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Identity Mappings in Deep Residual Networks", *European Conference on Computer Vision, Amsterdam,* 2016, pp. 630-645

[11] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, Kaiming He, "Aggregated Residual Transformations for Deep Neural Networks", *IEEE Conference on Computer Vision and Pattern Recognition, Honolulu*, 2017, pp. 5987-5995

[12] Matthew D. Zeiler, Rob Fergus, "Visualizing and Understanding Convolutional Networks", *European Conference on Computer Vision, Zurich,* 2014, pp. 818-833

[13] Min Lin, Qiang Chen, Shuicheng Yan, "Network in Network", *International Conference on Learning Representations,* Banff, 2014

[14] Jonathan Long, Evan Shelhamer, Trevor Darrell,"Fully Convolutional Networks for Semantic Segmentation", *IEEE Conference on Computer Vision and Pattern Recognition, Boston,* 2015, pp. 3431-3440

[15] Ross Girshick, "Fast R-CNN", *IEEE International Conference on Computer Vision, Araucano Park,* 2015, pp. 1440-1448

[16] Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", *Proceedings of the 28th International Conference on Neural Information Processing Systems, Montreal,* 2015, pp. 91-99

[17] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, Andrew Zisserman, "The Pascal Visual Object Classes (VOC) Challenge", *International Journal of Computer Vision*, 2010, Volume 88 Issue 2, pp. 303-338

[18] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi, "You Only Look Once: Unified, Real-Time Object Detection", *IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas*, 2016, pp. 779-788

[19] Joseph Redmon, Ali Farhadi, "YOLO9000: Better, Faster, Stronger", *IEEE Conference on Computer Vision and Pattern Recognition, Honolulu*, 2017, pp. 6517-6525

[20] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, Piotr Dollár, "Microsoft COCO: Common Objects in Context", *European Conference on Computer Vision*, 2014, vol 8693. Springer

[21] Joseph Redmon, Ali Farhadi, "YOLOv3: An Incremental Improvement", *CoRR*, 2018, abs/1804.02767

[22] Charles Henrique Porto Ferreira, Debora Maria Rossi de Medeiros, Fabricio Olivetti de Franca, "DC Distance: A Supervised Text Document Feature extraction based on class labels", *CoRR*, 2018, abs/1801.04554

[23] Sebastian Raschka, "Naive Bayes and Text Classification I Introduction and Theory", *CoRR*, 2014, abs/1410.5329

[24] Joanna Kazmierska, Julian Malicki, "Application of the na¨ıve bayesian classifier to optimize treatment decisions", *Radiotherapy and oncology : journal of the European Society for Therapeutic Radiology and Oncology*, 2008, volume 86, pp. 6-211

[25] Mehran Sahami, Susan Dumais, David Heckerman, and Eric Horvitz. A bayesian approach to filtering junk e-mail. In Learning for Text Categorization: *Papers from the 1998 workshop,* 1998, volume 62, pp. 98–105.

[26] Bo Tang, Steven Kay and Haibo He, "Toward Optimal Feature Selection in Naive Bayes for Text Categorization", *IEEE Transactions on Knowledge and Data Engineering,* 2016, Volume 28 Issue 9, pp. 2508-2521

[27] Fahim Mohammad, "Is preprocessing of text really worth your time for online comment classification?", *CoRR*, 2018, abs/1806.02908

[28] Amr Adel Helmy, Yasser M.K. Omar, Rania Hodhod, "An Innovative Word Encoding Method For Text Classification Using Convolutional Neural Network", 2018, *14th International Computer Engineering Conference (ICENCO)*, 2018, pp. 42-47

[29] Asifullah Khan, Anabia Sohail, Umme Zahoora , Aqsa Saeed Qureshi, "A Survey of the Recent Architectures of Deep Convolutional Neural Networks", *CoRR*, 2019, abs/1901.06032

[30] Jeffrey Thompson, Jinxiang Hu, Dinesh Pal Mudaranthakam, David Streeter, Lisa Neums, Michele Park, Devin C. Koestler, Byron Gajewski, Matthew S.Mayo, "Relevant Word Order Vectorization for Improved Natural Language Processing in Electronic Healthcare Records", *CoRR*, 2018, abs/1812.02627

[31] Craig W. Schmidt TripAdvisor, Inc. "Improving a tf-idf weighted document vector embedding", *CoRR*, 2019, abs/1902.09875

[32] Patrick Schratz, Jannes Muenchow, Eugenia Iturritx, Jakob Richter, Alexander Brenning, "Performance evaluation and hyperparameter tuning of statistical and machine-learning models using spatial data", *CoRR*, 2018, abs/1803.11266

[33] Akshay Agrawal, Akshay Naresh Modi, Alexandre Passos,

Allen Lavoie, Ashish Agarwal, Asim Shankar, Igor Ganichev, Josh Levenberg, Mingsheng Hong, Rajat Monga, Shanqing Cai, “ Tensorflow Eager: a multi-stage, python embedded dsl for machine learning”, *CoRR*, 2019, abs/1903.01855

[34] Hans Fangohr, Neil O’Brien, “Teaching Python programming with automatic assessment and feedback provision”, CoRR, 2015, abs/1509.03556