Convex Hull Object Detection

Kaamraan Khan
M.Tech (CSE) - IIITD
New Delhi, India
kaamraan19064@iiitd.ac.in

Ishita Agarwal
M.Tech (CSE) - IIITD
New Delhi, India
ishita19013@iiitd.ac.in

***Abstract*—We have multiple efficient methods for detecting objects within rectangular bounding boxes, but these rectangular boxes have large portions of background information. This leads to noise in the image. Also, for some cases like circular objects, oval-shaped objects, having a polygon encompassing their shape, may lead to 3-d interpretation of the object. With this view in mind, we have introduced the first Convex-hull object detector, based on Residual-Dense USegNet. Our architecture can detect objects of any arbitrary shape and place it inside the object's convex hull. We have done experiments on baseline models and also on various variants of USegNet, but our model performs the best. The experiments are performed on the entire cityscapes-dataset, with the semantic segmentation as the ground truth.**

***Index Terms*—USegNet, convex hull, semantic segmentation, object detection, residual skip connections, dense skip connections**

I. Introduction

With the advances in machine learning and deep learning methods, we have seen tremendous improvements in the fields of object detection as well. Various CNN algorithms and their variants have been applied over the years, which has made object detection and localization one of the most promising research areas.The networks present today are capable of recognizing the objects even in a very complicated environment. By complicated, we mean extremely cluttered, with a large number of objects, or objects being vague.

Currently, the methods like RCNN, Fast RCNN, Faster RCNN, YOLO, SSD etc, are used for object detection. They efficiently detect objects, and the detected objects are placed inside anchor-boxes, which are matched against the ground truth. Efficient implementations have been done in this field which have led to optimized detections at a faster rate. However, these boxes tend to have some background or noise, which may be unnecessary. Also, if the object is not rectangular shaped, we can never find the true shape of the object from the image. This idea formed the basis of our project. Instead of detecting objects of fixed shape (rectangle), why not detect objects of arbitrary shape, which will be closer to their real shape than the rectangular bounding boxes. Motivated from USegNet architecture, we have created Residual Dense USegNet for detecting the objects, which has not been explored much, which may form the basis to many similar and improved versions to be explored in the future.

The main challenge at hand was to take an image and efficiently draw a convex hull out of it. Now the convex hull algorithm works in cases where the foreground(object) and background are quite distinguishable. We tried various datasets, with various preprocessing, but the results were not very promising. So, we switched to datasets with semantic segmentation, where we created different channels for each type of object. These images with different channels formed our ground truth images. Next challenge we faced was to separate out the objects in a cluttered scenario, where some objects are very close to each other, they become a part of a single convex hull, instead of being detected as single objects.

Our contribution are as follows:

1) We have developed the first architecture, which can detect objects not just within rectangular boundaries, but any arbitrary shaped polygon. We don't need to have objects contained in bounding boxes having a certain aspect ratio.
2) We have successfully bounded objects in their convex hull maximising the signal to noise ratio.
3) Our architecture has less number of parameters, as compared to the parent architectures, i.e. Segnet and UNet.
4) Not much work has been done in this field. This research can form the basis for many new developments in the future.

II. Literature Review

With the advancement of convolutional neural networks, object detection and classification has become one of the prominent research fields. Earlier methods like Regional R-CNN [11], and it's variants like Fast R-CNN[12] and Faster R-CNN[13] were used for detecting objects within a bounding box. In these methods, there are 2 stages. In the first stage, there are regions of multiple scales containing the object, and only those regions are selected for next stage if the Intersection over Union (IoU) is greater than 0.5. After this step, the regions are passed to Convolutional Neural Network (CNN) which predicts the object of interest and it's location. Other methods like YoloV1[14], YoloV2[15], YoloV3[16], YoloV4[17], SSD[18], uses a single stage (faster than R-CNN and it's variants, but less accurate), to find the object of interest. It uses anchor-boxes to match predicted and ground truth results. All these methods focus on rectangular boxes to detect objects. Sometimes the object of interest might be very small as compared to the bounding box, which leads to unnecessary noise. Thus we decided to look for convex hulls to bound the object.

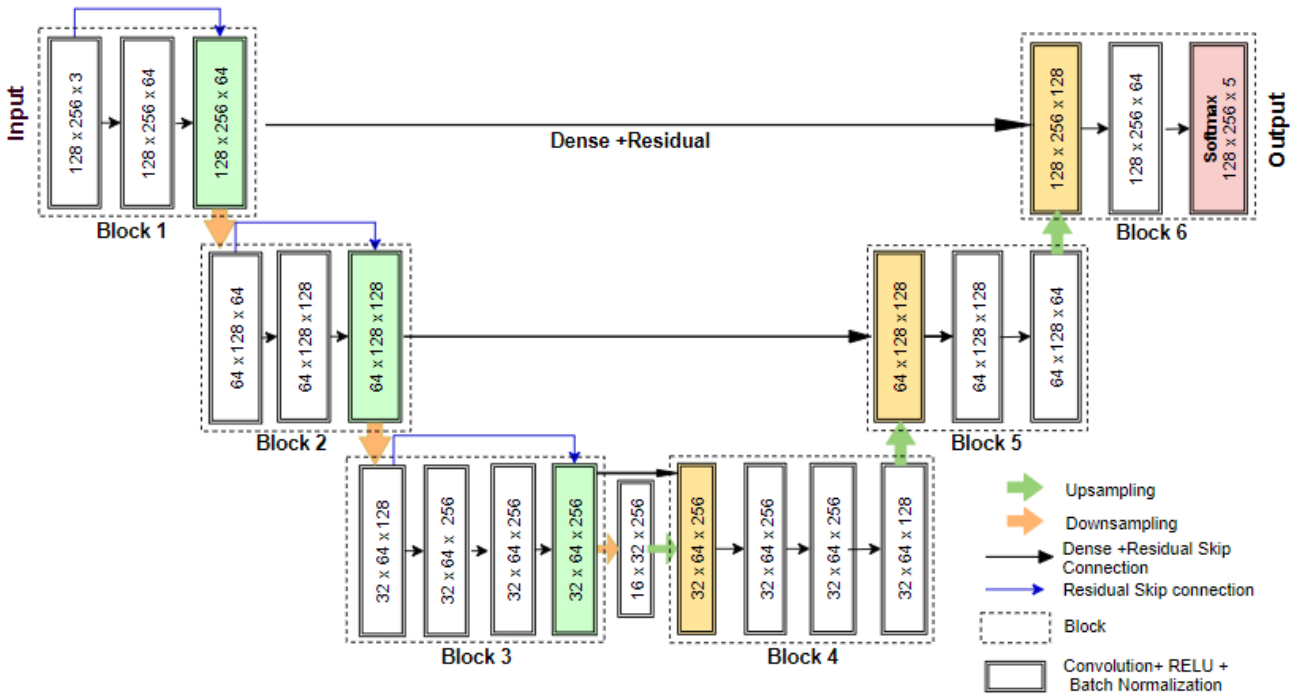

Fig. 1. USegNet Architecture, using residual dense skip connection.

Now, we will discuss USegNet[5]. But before that, one should know about SegNet[6] and UNet[4], as USegNet is the hybrid of SegNet and UNet. UNet follows a U-shaped architecture, consisting of a number of pooling and convolution layers. UNet follows a procedure of downsampling and upsampling and uses deconvolution layers for upsampling. During the upsampling path, feature maps of the same resolution are concatenated [5]. SegNet uses VGG encoder for downsampling and inverse VGG for upsampling.[5] It uses pooling layers for doing the upsampling and downsampling. Further details of our architecture USegNet (being the base), with residual and dense skip connections are described in proposed architecture and figure 1.

Convex hull is the box or space enclosing certain number of points, such that if a line is drawn anywhere within the space, it will be completely enclosed in that space. There won't be any concavities in the line forming the hull. It is a polygon which can be of any arbitrary shape.

Now let us discuss the concept of skip connections. We know that accuracy increases with layers, but there is a saturation to it due to vanishing gradient issues. So, in order to tackle the degradation issue, we have residual and dense skip connections, which builds shorter paths from one layer to another, rather than the traditional concept each layer feeding into the next layer. It reduces the issue of vanishing gradient and curse of dimensionality of very large neural networks. For residual blocks, apart from the layers feeding into the next layer,layers also feed into the layer about 2-3 jumps away as shown in blue arrow in figure 1.

DenseNet skip connection works on the same ground as residual skip connection. In residual connection output from the convolution layer is added to the input of the layer 2-3 layers away. In Dense connection, output from the convolution layer is concatenated. Since concatenation is done, final output is comparatively larger, making them highly expensive in terms of memory. It is generally preferred in shallow networks.

III. Methodology

A. Proposed Architecture

We have used USegNet as our base model, which is a combination of U-Net and Segnet architecture. We have added residual connection within a block and residual-dense skip connections across a block as a modification for better results.Residual and dense skip connections in the network enable short paths to be built directly from the output to each layer, alleviating the vanishing-gradient problem of a very deep network.They also add multiscale information along with it, also help us in incorporating both coarser and finer information. We call it Residual dense USegnet. USegNet is a hybrid architecture of U-Net and USegnet. The architecture is a U-shaped model, having different layers of convolutional, relu, batch normalization and pooling layers, as shown in the fig. 2. The initial feature map size is 128 x 256 x 3 i.e. the feature map has 3 channels, and the input is passed through 2 layers (1 layer consists of a convolution layer with kernel size =3, batch-normalization and RELU activation). The input of the first layer of one block is passed as an input to the first layer of the next block through residual skip connection.

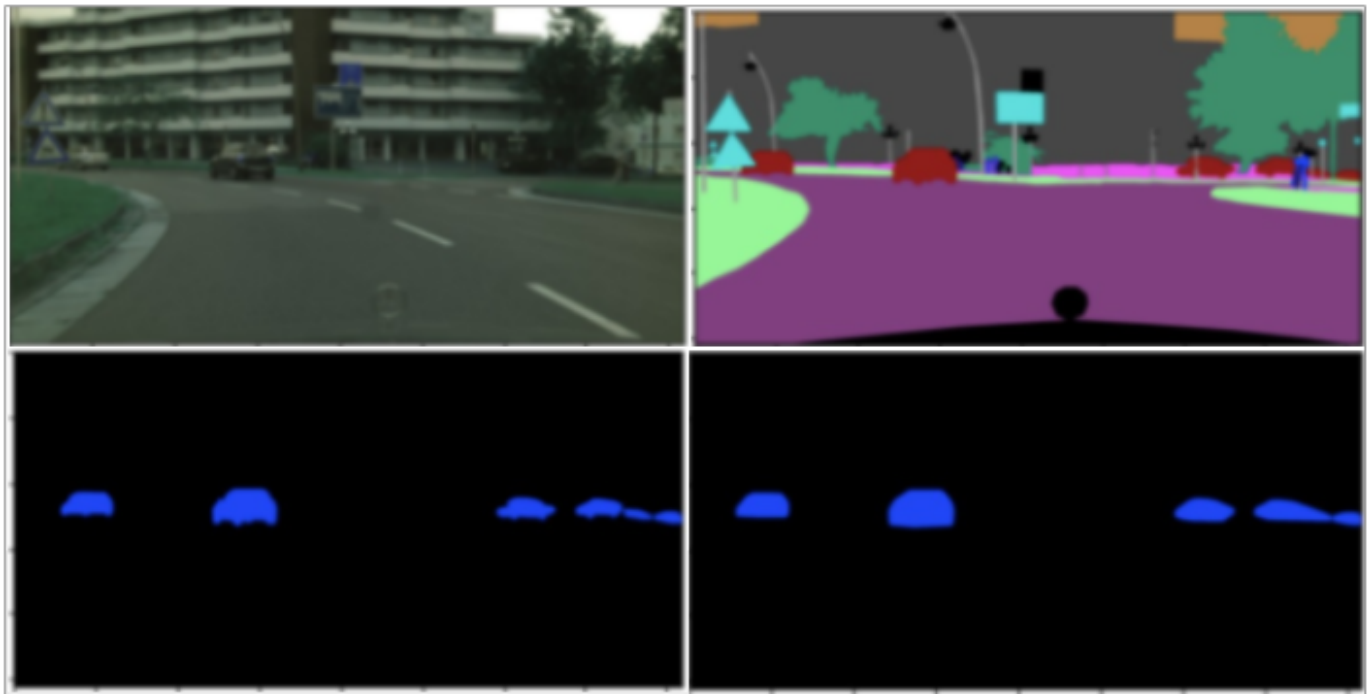

Fig. 2. The first image is the original image. The image to the right is semantic segmentation of the original image. The image in the 3rd quadrant or one below the original image is the one where foreground(cars) and background is seperated. Finally we apply a convex hull algorithm on this image and we finally get the image on right to it (in the fourth quadrant). All the pixels which come inside the convex hull, become a part of the foreground and we get our ground truth image.

The feature maps obtained from the first block are down-sampled using a max-pooling layer of kernel-size=2, and stride=2, to become the input of the next block. The above procedure is repeated thrice (i.e. for encoder part), till the feature map size is reduced to 16x32. After this the feature maps are upsampled using a max-unpooling layer with similar dimension as the max-pooling layer. The upsampled feature map is passed to the convolutional, relu, batch-normalization layer, and then to the max unpooling layer thrice, till the feature maps are of the same resolution as the initial feature map.Feature maps of same resolution from down-sampling and up-sampling layers are connected through residual dense skip connection in the up-sampling (decoder) path to incorporate both coarser and finer information. Finally the output is passed on to the softmax layer with 5 channels in order to implement 5 label classification (with 5 channels).

B. Implementation details

The input images and their corresponding ground truth (convex hulls of the semantic segmentation) are used to train the network with the stochastic gradient descent implementation of PyTorch. We have trained the network from the scratch (without using pre-trained weights) using Stochastic gradient descent (SGD) optimization with a learning rate of 0.01, batch size of 32, momentum of 0.9 and a weight decay of 0.005 for a maximum epoch of 500 during the training and used mean-squared error as the loss function. MSE is the mean of difference squared between the predicted value and the ground truth, represented as,

$$\frac{\sum_i^n (y_i - \hat{y}_i)^2}{n}$$

The number of training parameters in our model was 3660293, whereas the U-SegNet has 3483652 learnable parameters and UNet has 7763041 learnable parameters. Our model though has less number of parameters as compared to UNet and almost equivalent to USegNet, but gives better results compared to both the latter models.

C. Evaluation Metrics

For evaluation we have used 3 metrics.

1) Accuracy: We measure the training and testing accuracies based on the number of pixels correctly predicted by the model.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{1}$$

2) Dice ratio: It is one of the common metrics used to evaluate the performance of segmentation methods. The formula is:

$$Diceration = \frac{2 * TP}{2 * TP + FP + FN} \tag{2}$$

Where, DC is the dice ratio, TP is true positive, FP is false positive and FN is false negative.

3) Confusion matrix: It is used for measuring the performance of machine learning classification problems

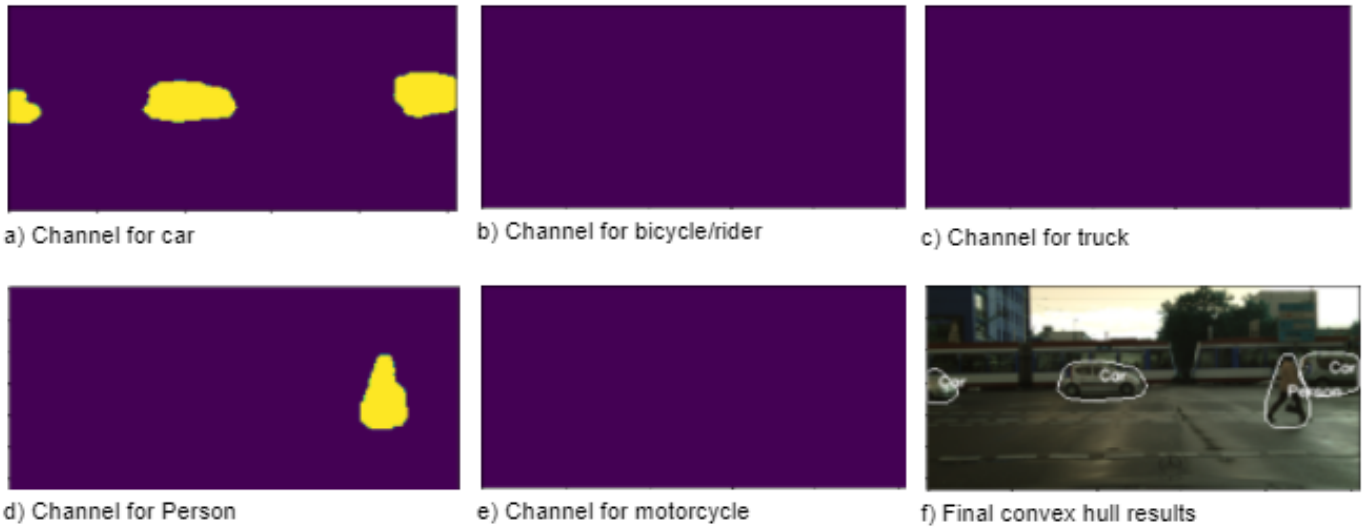

Fig. 3. a, b, c, d, e are the channels that come from our model. The images has car and person, but not bicycle, truck, motorcycle, hence no hull. Final results are shown in image f.

where output is generally 2 or more classes. It looks like:

Using confusion metrics we get to know the exact number of pixels of foreground and background correctly predicted by the model.

Confusion matrix and dice ratio are the main metrics used to evaluate the model performance as the number of foreground pixels are quite less than the number of background pixels. So, accuracy alone as an evaluation metric can misguide us.

IV. Experiments

A. Dataset Preparation

We have taken cityscapes-dataset comprising real-life images as well as semantic understanding of urban street scenes. The semantic segmentation consists of fine annotations of about 50 cities. The ground truth is provided by taking a semantic segmentation and finding the corresponding pixels with our object. We have taken 5 objects into focus and the part where the object is present is assigned as 1, others are 0 (background). After getting semantic segmentation of each object in one channel, we applied a convex hull algorithm on that channel and got a convex hull shape on that object. After getting the convex hull around our object, we further assigned the pixels which were a part of the background but a part of the convex hull as 1. So, we got a convex hull of objects as ground truth for training and real-life images as input. So, the shape of ground truth image is 128*256*5 (where 5 are no of channels and each channel is corresponding to 1 object) and shape of input image is 128*256*3. Original size of the image was 1024 x 2048, it's size was reduced for faster computation.

B. Experimental Details

We have used Residual-Dense USegNet, and have compared it with baseline models like SegNet, USegNet and Unet. We have also applied variants to USegnet, like Dense USegNet (In this, we have applied dense connections between each block i.e. 3 dense connections though in USegnet, we have only 1 dense skip connection) , Residual USegNet (In this instead of dense skip connection, we have applied residual skip connection.), Residual-Residual USegNet(In this we have applied residual connections between each block i.e. 3 residual skip connections though in USegnet, we have only 1 dense skip connection) ,USegNet with transConvolution(In this we have applied trans convolution for upsampling though in Usegnet we use unpooling for upsampling) , Residual-Residual USegNet with transConvolution(In this we have applied residual connections between each block i.e. 3 residual skip connections with trans convolution for upsampling). USegNet without skipConnection and with transConvolution(In this, we haven't applied any skip connection but used transconvolution for upsampling). For all the models we have applied the same parameters and our model outperforms all.

Dice ratio has been the major metric for comparison among these models. Another important parameter was the confusion matrix for test images. Larger the number of correct foreground pixels, better are the results. Our dataset had original images, and also semantic segmentation of those images. We have done the preprocessing on those images, to form the ground truth. The implementation details remain the same as described above.

In our experiments, the test ac-curacies of almost all the models were similar, given the large number of background pixels correctly predicted. But our main focus is correctly predicting the foreground pixels, i.e. the dice ratio should be higher. For that we can refer the table and see that, our model USegNet Residual Dense outperforms all.

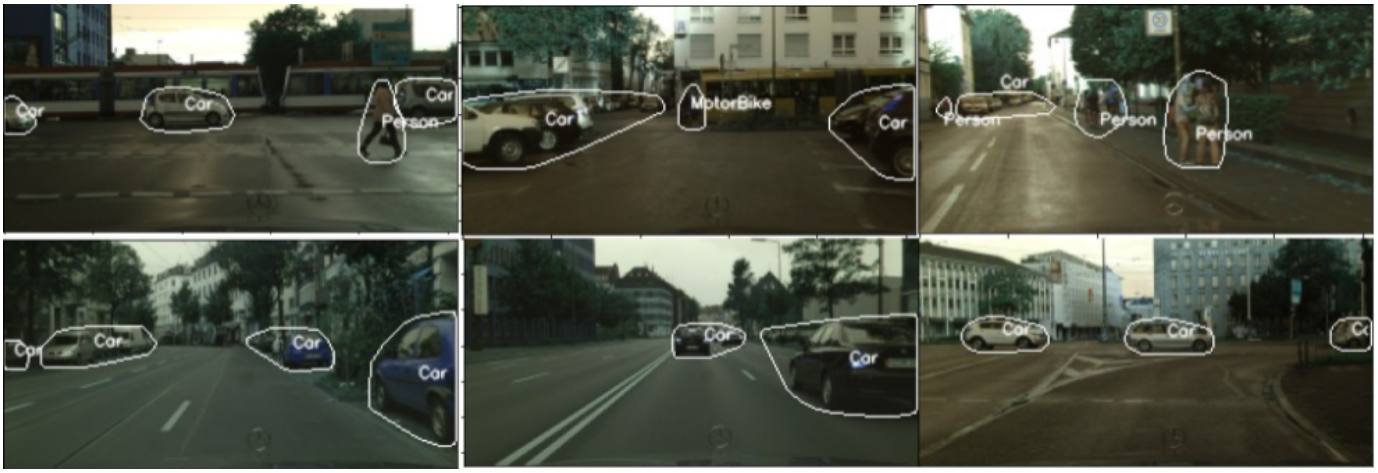

Fig. 4. Convex hull and classification on results generated by Residual Dense USegNet architecture

Model	Dice Ratio
SegNet	0.784
USegNet	0.781
UNet	0.776
USegNet with transConvolution	0.777
Dense USegNet	0.7880
Residual USegNet	0.7915
USegNet without skipConnection and with transConvolution	0.779
Residual-Residual USegNet	0.8016
Residual-Residual USegNet with transConvolution	0.7949
USegNet Residual Dense	0.814

TABLE I
PROPOSED MODEL RESULTS COMPARED WITH BASELINES AND OTHER VARIANTS OF USEGNET

C. Post-processing

From our architecture, we obtain the results which look like the one given in the figure 3. There are 5 channels corresponding to car, bicycle or rider, truck, person, motorcycle and are in this order only. If there is car in the image, the channel 1 will have convex hull results. If there is no car, then the 1st channel will be empty. See the figure 3. Now these channels are passed to convex hull algorithm, but before that we are doing gray scale conversion, image blurring, thresholding and are passing the converted images to find the contours using Chain Approx Simple method. If the area of contour is greater than certain value, we are bounding the real life image corresponding to the contour within the contour lines.Part F of Fig 3 shows the final image that we got after post-processing.

V. RESULTS

The figure 3 shows convex hull results from Residual dense USegNet architecture. There are objects of varying shapes in the image. Our model finds their shape and after getting the image shape in the form of semantic segmentation, we apply the convex hull algorithm to get the result.

From table 1, we also observe that Some models like Residual-Residual USegNet have performance comparable to our model. Still our model performs better. Our model clearly outperforms the baseline architectures, i.e. UNet, USegNet, SegNet. We also know that, the number of training parameters in our model was 3660293, whereas the U-SegNet has 3483652 learnable parameters and UNet has 7763041 learnable parameters. So, with moderate number of parameters, our models shows fairly great results. We have shown the results of our experiment, in the appendix, due to large number of results.

VI. CONCLUSION

We have worked upon an efficient way, to get a polygon around a device, rather than the traditional approaches of bounding the object within a rectangular boundry. We have worked on various architectures, but obtained the best results for our model (USegNet Residual Dense). We have obtained a dice ratio of 81.4. This type architecture helps in reducing the background or higher signal-to-noise ratio. Not much work has been done in this regard, making it one of the problems to be further explored in the coming future.

Acknowledgenemt: We would like to express immense gratitude to our guide, Prof. Koteswar Rao Jerripothula. He helped us, whenever we were stuck and explained us everything which were hurdle in the path of this research.

REFERENCES

[1] Lu, K., Pavlidis, T. (2007). Detecting textured objects using convex hull. Machine Vision and Applications, 18(2), 123–133. doi:10.1007/s00138-006-0060-0.

[2] Dong, Wenbo Roy, Pravakar Peng, Cheng Isler, Volkan. (2020). Ellipse R-CNN: Learning to Infer Elliptical Object from Clustering and Occlusion.

[3] Cupec, R., Vidović, I., Filko, D., urović, P. (2020). Object Recognition Based on Convex Hull Alignment. Pattern Recognition, 107199. doi:10.1016/j.patcog.2020.107199.

[4] Ronneberger, O., Fischer, P., Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, 234–241. doi:10.1007/978-3-319-24574-4-28.

[5] Kumar, P., Nagar, P., Arora, C., Gupta, A. (2018). U-Segnet: Fully Convolutional Neural Network Based Automated Brain Tissue Segmentation Tool. 2018 25th IEEE International Conference on Image Processing (ICIP). doi:10.1109/icip.2018.8451295.

[6] Badrinarayanan, V., Kendall, A., Cipolla, R. (2017). SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(12), 2481–2495. doi:10.1109/tpami.2016.2644615.

[7] M.A, Jayaram Fleyeh, Hasan. (2016). Convex Hulls in Image Processing: A Scoping Review. American Journal of Intelligent Systems. 2016. 48-58. 10.5923/j.ajis.20160602.03.

[8] Wang, X., Yu, K., Wu, S., Gu, J., Liu, Y., Dong, C., … Loy, C. C. (2019). ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks. Computer Vision – ECCV 2018 Workshops, 63–79. doi:10.1007/978-3-030-11021-5-5.

[9] He, K., Zhang, X., Ren, S., Sun, J. (2016). Deep Residual Learning for Image Recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). doi:10.1109/cvpr.2016.90.

[10] T. Tong, G. Li, X. Liu and Q. Gao, "Image Super-Resolution Using Dense Skip Connections," 2017 IEEE International Conference on Computer Vision (ICCV), 2017, pp. 4809-4817, doi: 10.1109/ICCV.2017.514.

[11] Ross Girschick, Jeff Donahue, Trevor Darell and Jitendra Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation", https://arxiv.org/abs/1311.2524

[12] Ross Girschick, "Fast R-CNN", In IEEE Conference on Computer Vision (ICCV) 2015.

[13] Shaoqing Ren, Kaiming He, Ross Girshick, Jian Shun, "Faster R-CNN: Towards RealTime Object detection with Region Proposal Networks", https://arxiv.org/abs/1506.01497.

[14] Joseph Redmon, Santosh Divvala, Ross Girshick and Ali Farhadi, "You Only Look Once: Unified Real-Time Object Detection", in Conference on Computer Vision and Pattern Recognition (CVPR) 2016.

[15] Joseph Redmon, Ali Farhadi, "YOLO9000: Better, Faster, Stronger", https://arxiv.org/abs/1612.08242.

[16] Joseph Redmon, Ali Farhadi, "YOLOv3: An Incremental Improvement", https://arxiv.org/abs/1804.02767.

[17] Bochkovskiy, Alexey Wang, Chien-Yao Liao, Hong-yuan. (2020), "YOLOv4: Optimal Speed and Accuracy of Object Detection".

[18] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, ChengYang Fu, Alexander C. Berg, "SSD: Single Shot MultiBox Detector", in European Conference on Computer Vision (ECCV) 2016.