



e-ISSN: 2319-8753 | p-ISSN: 2320-6710

# IJIRSET

International Journal of Innovative Research in  
**SCIENCE | ENGINEERING | TECHNOLOGY**



# INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

Volume 10, Issue 4, April 2021

**ISSN** INTERNATIONAL  
STANDARD  
SERIAL  
NUMBER  
INDIA

**Impact Factor: 7.512**



9940 572 462



6381 907 438



ijirset@gmail.com



www.ijirset.com

# Superiority of YOLOv4 in terms of Speed and Accuracy over YOLOv3

**Prof. Dr.Kamal Alaskar**

Professor, Department of Computer Applications Bharati Vidyapeeth (Deemed to be University) Institute of Management Kadamwadi, Kolhapur, Maharashtra, India

**ABSTRACT:** The YOLOv4 released by Alexey Bochkovskiy and there are a huge number of features which are said to improve Convolutional Neural Network (CNN) accuracy. Practical testing of combinations of such features on large datasets, and theoretical justification of the result, is required. Some features operate on certain models exclusively and for certain problems exclusively, or only for small-scale datasets; while some features, such as batch-normalization and residual-connections, are applicable to the majority of models, tasks, and datasets.

There are lots of ways used to pick or grasp object through robotic hand but there are some hardly worked done with help the of Deep Learning approaches. To solve this issue, a solution is proposed which involves human strategies of picking up an object using Neural Network classifier. Classifier uses help of object detection model to detect object in environment and classifier classifies into picking strategy as per objects shape and orientation. Strategy detected by classifier can be used by soft hand as anticipatory action and reactive grasp. To increase accuracy number of primitive measures taken into consideration, our bounded and some of limitation are taken into mind while proposing architecture.

**KEYWORDS:** Convolutional Neural Network (CNN), Speed, Accuracy, Deep learning, Grasping Strategies, Soft Hand.

## I. INTRODUCTION

### New Features of YOLOv4

1. Weighted-Residual-Connections (WRC)
2. Cross-Stage-Partial-connections (CSP)
3. Cross mini-Batch Normalization (CmBN)
4. Self-adversarial-training (SAT)
5. Mish activation
6. Mosaic data augmentation
7. DropBlock regularization
8. CIOU loss

### YOLO v4 uses:

- Bag of Freebies (BoF) for backbone: CutMix and Mosaic data augmentation, DropBlock regularization, Class label smoothing
- Bag of Specials (BoS) for backbone: Mish activation, Cross-stage partial connections (CSP), Multiinput weighted residual connections (MiWRC).
- Bag of Freebies (BoF) for detector: CIOU-loss, CmBN, DropBlock regularization, Mosaic data augmentation, Self-Adversarial Training, Eliminate grid sensitivity, Using multiple anchors for a single ground truth, Cosine annealing scheduler [52], Optimal hyperparameters, Random training shapes.
- Bag of Specials (BoS) for detector: Mish activation, SPP-block, SAM-block, PAN path-aggregation block, DIOU-NMS.

Soft hand as proven to be more efficient when used in supervision of human[1][2]. But such approach is still lagging in performance hence Data Driven approach can be used to improve the performance (see [3]).For detection of object on which grasping is done YOLO is embedded (You Only Look Once) technology YOLOv4, unlike used in [5] YOLOv3. YOLOv4 has shown good accuracy with respect to Single Shot MultiBox Detector (SSD). The code is online at

<https://pjreddie.com/yolo/>. [4]. YOLOv4 replaces soft max function with binary classification which helps in reducing complexity and output is same as multiclass classification only [6].

Machine Learning approach had proved positive results in detection of the object or say grasping details of object [7][8][9]. In [10] Neural network helps in predicting unseen object and action to be performed on that object with the help of learning from pre-labeled data. In [11] convolution neural network had played good role in detecting objects strategy output of whose network can be used to determine controlling robotic soft hand. Using such network objects are trained on 45 objects and tested with 10 objects which gives approximately accuracy of 84% [3]. Especially focus on updating object detection technique used in [3] from YOLO9000 to YOLOv4 which are faster than as used in [3].

## II. LITERATURE SURVEY

As a main reference to project [3] whose system is used for further increment in approach is used in Data Driven approach to control Anthropomorphic soft hand Which uses Deep learning technology significance for increasing performance of control strategy of soft hand. It uses InceptionV3 module for grasping object and concluding strategy to be used by [11] reactive strategies have become significant in human robot exchanging of objects. It is tested over 10 objects by training on 45 objects which corresponds to accuracy of 84% on test objects. Where it is decided to update object detection method input given to model of Inceptionv3.

Paper achieves these goals by:

- Using Deep Neural Network model Inceptionv3 model predicts or decides which action human would take to pick certain type of objects.
- Understanding through which action would be performed by human using robotic hand containing soft hand capability.
- Testing over on 10 objects which are not used in model training model would be able to predict action to perform on object by soft hand or robotic arm with accuracy of prediction 84%.

### A. DIFFERENT MODELS

For Deep Learning InceptionV3 module will be used which is pre-trained model for object grasping and third version of inception module (see [12]). Module is trained over ImageNet Dataset. InceptionV3 model with 144 crops gained top-5 error rate is 4.2%, which pullback PReLU-Net and Inception-v2 which were used in 2015. With 42 layers deep, the parameter complexity increases by only 2.5% Google Net [12]. It consists of 313 layers of neuron where some of layers will be retrained to get our performance output. PyTorch version Inception-v3: [14] <https://github.com/pytorch/vision/blob/master/torchvision/models/inception.py>

### B. OBJECT DETECTION MODELS

#### Object Detector:

The object detector is composed of several parts. The parts are

- **Input:** Image, Patches, Image Pyramid
- **Backbones:** VGG16 [68], ResNet-50 [26], SpineNet [12], EfficientNet-B0/B7 [75], CSPResNeXt50 [81], CSPDarknet53 [81]
- **Neck:**
  - **Additional blocks:** SPP [25], ASPP [5], RFB [47], SAM [85]
  - **Path-aggregation blocks:** FPN [44], PAN [49], NAS-FPN [17], Fully-connected FPN, BiFPN [77], ASFF [48], SFAM [98]
- **Heads::**
  - **Dense Prediction (one-stage):**
    - RPN [64], SSD [50], YOLO [61], RetinaNet [45] (anchor based)
    - CornerNet [37], CenterNet [13], MatrixNet [60], FCOS [78] (anchor free)
  - **Sparse Prediction (two-stage):**
    - Faster R-CNN [64], R-FCN [9], Mask R-CNN [23] (anchor based)
    - RepPoints [87] (anchor free)

The difference of YOLOv4 from YOLOv3 is only the backbone. Because the YOLOv3 has Darknet53 backbone. But the yolov4 has CSPDarknet53.

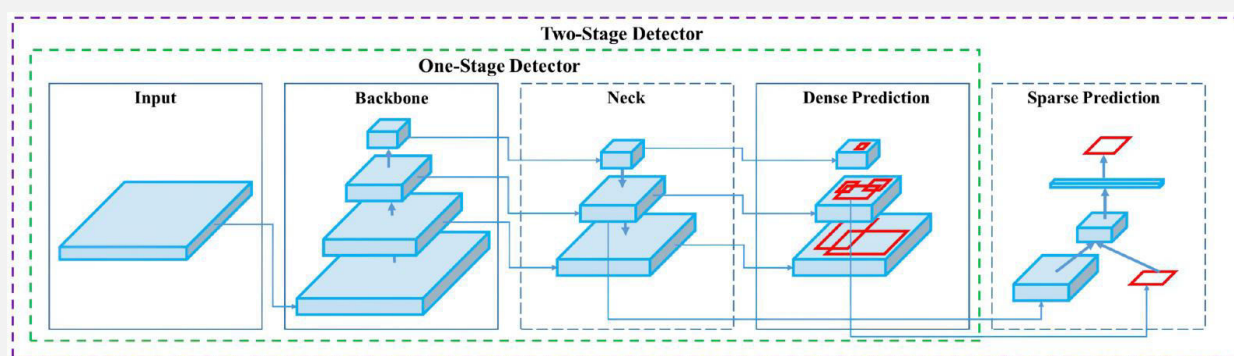


Table 1: Parameters of neural networks for image classification.

| Backbone model         | Input network resolution | Receptive field size | Parameters    | Average size of layer output (WxHxC) | BFLOPs (512x512 network resolution) | FPS (GPU RTX 2070) |
|------------------------|--------------------------|----------------------|---------------|--------------------------------------|-------------------------------------|--------------------|
| CSPResNext50           | 512x512                  | 425x425              | 20.6 M        | <b>1058 K</b>                        | 31 (15.5 FMA)                       | 62                 |
| CSPDarknet53           | 512x512                  | 725x725              | <b>27.6 M</b> | 950 K                                | 52 (26.0 FMA)                       | <b>66</b>          |
| EfficientNet-B3 (ours) | 512x512                  | <b>1311x1311</b>     | 12.0 M        | 668 K                                | 11 (5.5 FMA)                        | 26                 |

## CSPDarknet53

All other things are same compare with yolov3. The Yolov4 heads is same as yolov3. The Heads is prediction part and it has two types. One is Dense Prediction (One-stage detector) and another one is Sparse Prediction (Two stage detector).



detector types

|                           | backbone                 | AP          | AP <sub>50</sub> | AP <sub>75</sub> | AP <sub>S</sub> | AP <sub>M</sub> | AP <sub>L</sub> |
|---------------------------|--------------------------|-------------|------------------|------------------|-----------------|-----------------|-----------------|
| <i>Two-stage methods</i>  |                          |             |                  |                  |                 |                 |                 |
| Faster R-CNN+++ [5]       | ResNet-101-C4            | 34.9        | 55.7             | 37.4             | 15.6            | 38.7            | 50.9            |
| Faster R-CNN w FPN [8]    | ResNet-101-FPN           | 36.2        | 59.1             | 39.0             | 18.2            | 39.0            | 48.2            |
| Faster R-CNN by G-RMI [6] | Inception-ResNet-v2 [21] | 34.7        | 55.5             | 36.7             | 13.5            | 38.1            | 52.0            |
| Faster R-CNN w TDM [20]   | Inception-ResNet-v2-TDM  | 36.8        | 57.7             | 39.2             | 16.2            | 39.8            | <b>52.1</b>     |
| <i>One-stage methods</i>  |                          |             |                  |                  |                 |                 |                 |
| YOLOv2 [15]               | DarkNet-19 [15]          | 21.6        | 44.0             | 19.2             | 5.0             | 22.4            | 35.5            |
| SSD513 [11, 3]            | ResNet-101-SSD           | 31.2        | 50.4             | 33.3             | 10.2            | 34.5            | 49.8            |
| DSSD513 [3]               | ResNet-101-DSSD          | 33.2        | 53.3             | 35.2             | 13.0            | 35.4            | 51.1            |
| RetinaNet [9]             | ResNet-101-FPN           | 39.1        | 59.1             | 42.3             | 21.8            | 42.7            | 50.2            |
| RetinaNet [9]             | ResNeXt-101-FPN          | <b>40.8</b> | <b>61.1</b>      | <b>44.1</b>      | <b>24.1</b>     | <b>44.2</b>     | 51.2            |
| YOLOv3 608 × 608          | Darknet-53               | 33.0        | 57.9             | 34.4             | 18.3            | 35.4            | 41.9            |

Table I Object Detection score working of block in detail for figure shown above (Figure) .

## III. METHODOLOGY

- Training Phase
- Test Phase

## A. TRAINING PHASE

To reduce training time, pre trained model is used to get more accuracy in less time YOLOv4 pre trained model weights and model to detect object which is been trained on COCO datasets which detects 80 classes if required to train can train using link to code (<https://github.com/qgwweee/keras-yolo3>). Training InceptionV3 module which is best for classification which can perform better than human vision also.

## • DATASET

For creation of dataset as suggested in paper [3] objects are trained on 45 objects that are mostly different than used in [3] . Objects are kept on table at center position, different people are asked to pick up object and keep it onside and

record this video. Each video is labeled as per the primitive taken by human in that video and identified six actions namely unlike of [4] to increase accuracy of model as top, left and right pinches can be replaced

**Top:** In top grasp approach, hand is approached from top with palm parallel to object on top it. This approach is used to pick object which are big in size like box used in following example.

**Bottom:** In this primitive hand are approach from right side and picked object with 4 fingers and thumb in opposite of 4 fingers providing negative force to hold object e.g. bowl.

**Pinch:** This type of approach is like top approach only, but have some difference which is commonly used for small objects like match stick.

**Slide:** In this approach object which are thin in size like CD are slide to corner of table and flipped and picked up

**Flip:** This is primitive use for small object with thin in size like coin where slide primitive cannot be used here and just flip object perpendicular to table.

**Lateral:** In this primitive hand approaches towards objects from right side with palm perpendicular to table and parallel to object like picking up bottle  
These are 45 objects that are used here

**Note:** Objects which are used are also orientation dependent.

#### • TRANSFER LEARNING

Transfer Learning approach is where used weights of pre trained model to predict our aim can add some of more layers at the end of model that are fully connected or expand model and train concatenate part can also retrain some of the layers of models.  
Adam as our optimizer.

#### • STEPS IN TRAINING

Initially it is set are some of the parameters of our build model taking from [3] paper outcomes

BATCH\_SIZE = 10

EPOCHS = 10

Regularizer = 0.01

Pdrop = not used in our case

Learning rate = Used Adam doesn't require learning rate.

Learning rate for fine tuning = Used Adam doesn't require learning rate.

Optimizer = Adam.

Activation function for last FFC Layers = relu.

Prediction layer activation function = SoftMax.

Next freeze all layers in model except the last three layers in model as shown in figure and train last three layers with learning rate initialized in step1.

After training of last layers freeze all layers in model except unfreezing middle layers from 173-249 This layers are used for fine tuning to get inner attributes.

Using less amount of time using Kaggle platform to run our network on GPU which is freely provided to us by Kaggle of 12 GB ram of NVIDIA graphics.

#### TEST PHASE

Once our model is trained, it can now integrate in our system to test new objects.

#### FLOWCHART

For implementation two flow charts are suggested one for Training and other for Real Time implementation. Dataset is created using Videos generated by human strategy for picking up objects Instead of generating videos we have used objects image directly images. Model is created using pre trained model of InceptionV3 popping last layer and adding 2 fully connected layers and SoftMax layer for multi class classification. Dataset is resized to input require size of model i.e. 416\*416 and then split into 80% as training data and 20% as validation data.

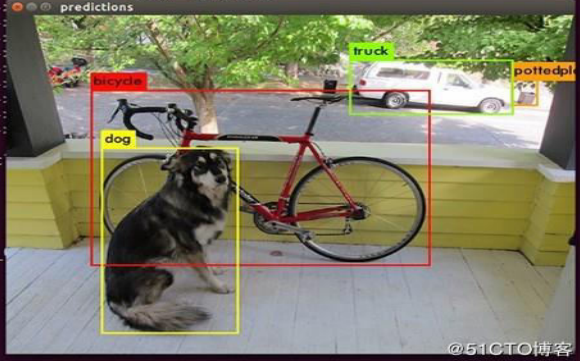
Freeze all layers except added layers and compile model with batch size of 10 and learning rate of 0.001 Adam as optimizer and train model with 30 epochs on gpu. Freeze all layers except layers from 72-249 layers and compile model with the same parameters as mentioned above except learning rate of 0.00001 and train model. Model is trained and ready for prediction.

#### SUPERIORITY OF YOLOv4 PREDICTION YOLOv3 COMPARISION:

```

bal@bal-MS-7A94: ~/darknet
148 conv 512 3 x 3/ 1 38 x 38 x 256 -> 38 x 38 x 512 3.407 BF
149 conv 255 1 x 1/ 1 38 x 38 x 512 -> 38 x 38 x 255 0.377 BF
150 yolo
[yolo] params: iou loss: ciou (4), iou norm: 0.07, cls_norm: 1.00, scale_x_y: 1.10
nms kind: greedy (1), beta = 0.600000
151 route 147 -> 38 x 38 x 256
152 conv 512 3 x 3/ 2 38 x 38 x 256 -> 38 x 38 x 512 3.407 BF
153 route 152 116 -> 38 x 38 x 512
154 conv 512 1 x 1/ 1 19 x 19 x 1024 -> 19 x 19 x 512 0.377 BF
155 conv 1024 3 x 3/ 1 19 x 19 x 512 -> 19 x 19 x 1024 3.407 BF
156 conv 512 1 x 1/ 1 19 x 19 x 1024 -> 19 x 19 x 512 0.377 BF
157 conv 1024 3 x 3/ 1 19 x 19 x 512 -> 19 x 19 x 1024 3.407 BF
158 conv 512 1 x 1/ 1 19 x 19 x 1024 -> 19 x 19 x 512 0.377 BF
159 conv 1024 3 x 3/ 1 19 x 19 x 512 -> 19 x 19 x 1024 3.407 BF
160 conv 255 1 x 1/ 1 19 x 19 x 1024 -> 19 x 19 x 255 0.377 BF
161 yolo
[yolo] params: iou loss: ciou (4), iou norm: 0.07, cls_norm: 1.00, scale_x_y: 1.10
nms kind: greedy (1), beta = 0.600000
Total BFLOPS 128.459
avg outputs = 1068395
Allocate additional workspace size = 52.43 MB
Loading weights from ./yolov4.weights...
seen 64, trained: 32032 K-images (500 Kilo-batches)
Done! Loaded 162 layers from weights-file
data/dog.jpg: Predicted in 29.073000 milli-seconds.
bicycle: 92%
dog: 98%
truck: 92%
pottedplant: 33%

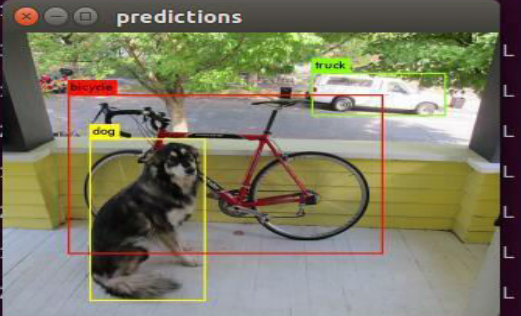
```



```

f@f-Lenovo-G405: ~/darknet
OPS
97 upsample 2x 38 x 38 x 256 -> 38 x 38 x 512 3.407 BF
98 route 97 36 -> 38 x 38 x 512
99 conv 128 1 x 1 / 1 19 x 19 x 1024 -> 19 x 19 x 512 0.377 BF
OPS
100 conv 256 3 x 3 / 1 19 x 19 x 512 -> 19 x 19 x 1024 3.407 BF
OPS
101 conv 128 1 x 1 / 1 19 x 19 x 1024 -> 19 x 19 x 512 0.377 BF
OPS
102 conv 256 3 x 3 / 1 19 x 19 x 512 -> 19 x 19 x 1024 3.407 BF
OPS
103 conv 128 1 x 1 / 1 19 x 19 x 1024 -> 19 x 19 x 512 0.377 BF
OPS
104 conv 256 3 x 3 / 1 19 x 19 x 512 -> 19 x 19 x 1024 3.407 BF
OPS
105 conv 255 1 x 1 / 1 19 x 19 x 1024 -> 19 x 19 x 255 0.377 BF
OPS
106 yolo
Loading weights from yolov3.weights...Done!
data/dog.jpg: Predicted in 119.732267 seconds.
dog: 100%
truck: 92%
bicycle: 99%

```



[https://blog.csdn.net/FJY\\_sunshine](https://blog.csdn.net/FJY_sunshine)

#### IV. RESULTS

Results of the new features are FPS, Accuracy with V100 GPU and the results shown in below. They are tested with multiple GPUs and the results here.

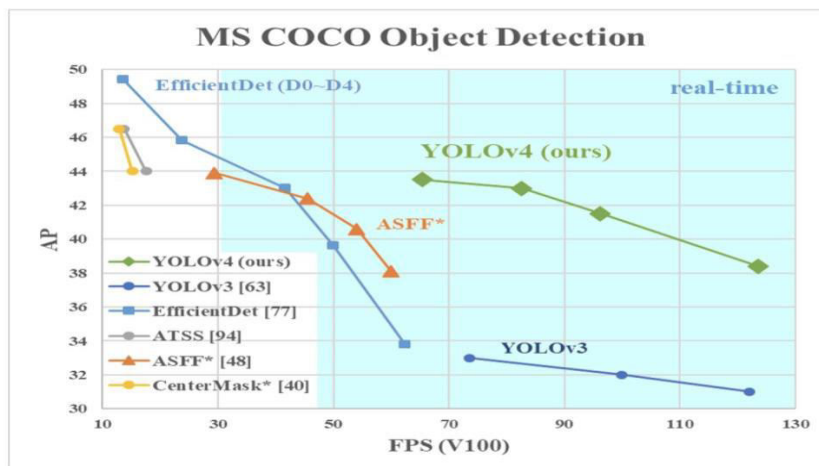


Figure 1: Comparison of the proposed YOLOv4 and other state-of-the-art object detectors. YOLOv4 runs twice faster than EfficientDet with comparable performance. Improves YOLOv3's AP and FPS by 10% and 12%, respectively.

#### V. CONCLUSION

Our work includes use of YOLOv4 instead of YOLOv3 which detects objects more accurately. Proposed and validated a Primitive detection using deep learning and performing using soft hand Goal can be gained by:

- Creating new model by using transfer learning methodology and using new Adam optimizer.
- Creating new primitive to increase accuracy of model to predict actions and reducing primitive used in previous work,
- Using Neural Network methodology in robotic system to make it more intelligent.
- Testing on wide range of object that are not used in training data.

Further in the next approach or future work stereo camera will be used to get more accuracy.

#### REFERENCES

- C. Erdogan, A. Schroder, and O. Brock, "Coordination of intrinsic and extrinsic degrees of freedom in soft robotic grasping," in 2018 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2018, pp. 1–6.
- M. Haas, W. Friedl, G. Stillfried, and H. Hoppner, "Human-robotic variable-stiffness grasps of small-fruit containers are successful even under severely impaired sensory feedback," *Frontiers in neurorobotics*, vol. 12, p. 70, 2018.
- J. Zhou *et al.*, "A Soft-Robotic Approach to Anthropomorphic Robotic Hand Dexterity," in *IEEE Access*, vol. 7, pp. 101483-101495, 2019.
- YOLOv3: An Incremental Improvement Joseph Redmon, Ali Farhadi University of Washington
- J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," arXiv preprint, 2017.
- [https://medium.com/@jonathan\\_hui/real-time-object-detection-with-yolo-volov2-28b1b93e2088](https://medium.com/@jonathan_hui/real-time-object-detection-with-yolo-volov2-28b1b93e2088)
- X. Yan, J. Hsu, M. Khansari, Y. Bai, A. Pathak, A. Gupta, J. Davidson, and H. Lee, "Learning 6-dof grasping interaction via deep geometryaware 3d representations," in 2018 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2018, pp. 1–9.





8. P. Schmidt, N. Vahrenkamp, M. Wächter, and T. Asfour, "Grasping of unknown objects using deep convolutional neural networks based on depth images," in 2018 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2018, pp. 6831–6838.
9. A. Zeng, S. Song, K.-T. Yu, E. Donlon, F. R. Hogan, M. Bauza, D. Ma, O. Taylor, M. Liu, E. Romo et al., "Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching," in 2018 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2018, pp. 1–8.
10. T. Nishimura, K. Mizushima, Y. Suzuki, T. Tsuji, and T. Watanabe, "Thin plate manipulation by an under-actuated robotic soft gripper utilizing the environment," in Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on. IEEE, 2017, pp. 1236–1243.
11. M. Bianchi, G. Averta, E. Battaglia, C. Rosales, A. Tondo, M. Poggiani, G. Santaera, S. Ciotti, M. G. Catalano, and A. Bicchi, "Tactilebased grasp primitives for soft hands: Applications to human-to-robot handover tasks and beyond," in Robotics and Automation (ICRA), 2018 IEEE International Conference on. IEEE, 2019.
12. [2015 CVPR] [GoogLeNet / Inception-v1] Going Deeper with Convolutions
13. C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2818–2826.
14. <https://medium.com/@sh.tsang/review-inception-v3-1st-runner-up-image-classification-in-ilsvrc-2015-17915421f77c>





**Impact Factor:  
7.512**



# **INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH**

**IN SCIENCE | ENGINEERING | TECHNOLOGY**

**9940 572 462 6381 907 438 ijirset@gmail.com**



**www.ijirset.com**

Scan to save the contact details