Performance Metric Estimation of Fast RCNN with VGG-16 Architecture for Emotional Recognition

SAMSON IMMANUEL J[1], MANOJ G[2], DIVYA.P. S[3]
Department of ECE[1,2], Department of Mathematics[3]
Karunya Institute of Technology and Sciences
Karunya Nagar, Coimbatore, Tamil Nadu
INDIA-641114

Abstract: - Faster R-CNN is a state-of-the-art universal object detection approach based on a convolutional neural network that offers object limits and objectness scores at each location in an image at the same time. To hypothesis object locations, state-of-the-art object detection networks rely on region proposal techniques. The accuracy of ML/DL models has been shown to be limited in the past due to a range of issues, including wavelength selection, spatial resolution, and hyper parameter selection and tuning. The goal of this study is to create a new automated emotional detection system based on the CK+ database. Fast R-CNN has lowered the detection network's operating time, revealing region proposal computation as a bottleneck. We develop a Region Proposal Network (RPN) in this paper that shares full-image convolutional features with the detection network, allowing for almost cost-free region suggestions. The suggested VGG-16 Fast RCNN model obtained user accuracy close to 100 percent in the emotion class, followed by VGG-16 (99.79 percent), Alexnet (98.58 percent), and Googlenet (98.58 percent) (98.32 percent). After extensive hyper parameter tuning for emotional recognition, the generated Fast RCNN VGG-16 model showed an overall accuracy of 99.79 percent, far higher than previously published results.

Key-Words: - Fast RCNN, VGG-16, GoogleNet, CK+Database, Alexnet.

Received: May 17, 2021. Revised: April 13, 2022. Accepted: May 12, 2022. Published: June 25, 2022.

1 Introduction

A R-CNNs (Region-based Convolutional Neural Networks) are a family of machine learning models used in computer vision and image processing. In most cases, these networks have an output, an input, and numerous layers in hidden. Hidden layers such as convolutional, pooling, fully connected, and normalization are common. Object Detection, Image classification, text detection, object tracking, object detection and voice recognition and natural language processing, and other tasks can all be done with convolutional neural networks. In deep learning architectures, an input image given to the R-CNN model goes through a mechanism called selective search to extract information about the region of interest. Region of interest can be represented by the rectangle boundaries. The central component of CNN is the convolution operation employing tiny filter patches (kernels). he resulting method can train a very deep detection network (VGG16 [1 0]) 9× faster than R-CNN [9] and 3× faster than SPPnet [11].

The convolution layer after the pooling layer might work on a different scale than the layers before it to subsampling. These CNN-learned characteristics can be input into different network structures to accomplish complex Object detection and recognition are two examples of tasks, semantic segmentation [1].

2 FAST RCNN

Faster R-CNN builds on previous work to efficiently classify object proposals using deep convolutional networks. Compared to previous work, Faster R-CNN employs a region proposal network and does not require an external method for candidate region proposals. Fast R-CNN [2] is a complex technology that uses deep convolutional networks to accelerate the training and testing phases while enhancing accuracy and effectively classifying object proposals. (A multitask loss is used to train the Fast R-CNN architecture is shown in the Fig 3.1 from start to finish. The Region-based Convolutional Network method (RCNN) [9] achieves excellent object detection accuracy by using a deep ConvNet to classify object proposals.

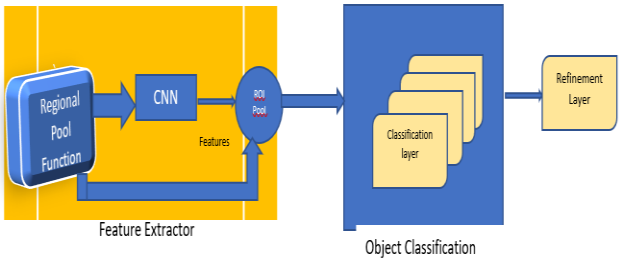


Fig 1 Fast RCNN Architecture

R-CNN is slow because it performs a ConvNet forward pass for each object proposal, without sharing computation. Features are extracted for a proposal by maxpooling the portion of the feature map inside the proposal into a fixed-size output (e.g., 6 × 6). We execute glomerular identification as a binary classification task to see if Faster R-CNN can correctly detect bounding boxes surrounding the glomeruli using ground truth. As a result, the evaluation measures were micro-averages of recall, precision, and their harmonic mean (F-measure). We used Faster R-CNN implementation provided by Tensorflow Object Detection API, which is a deep learning framework with Python language. We applied a technique called fine tuning that takes a pretrained model and transfers the connection weights to our own model, retraining the model to be adapted to our task.

3 Types of architectures

Fast RCNNs are available in many different forms and layer configurations. This section gives an overview of the most prevalent Fast RCNN models and discusses their advantages.

3.1 AlexNet Architecture

On the other hand, AlexNetLeNet [10] established the history of deep CNNs, however Fast RCNN was restricted to digit recognition tasks on the hand at the time then did not achievefine across all image modules. AlexNet [11] is usually recognized as the initial deep Fast RCNN construction, with breakthrough outcomes in AlexNet is an image categorization and identification system [12] was suggested who increased the RCNN's learning capacity by making it richer and more complex utilizing multiple constraint optimization procedures is shown in Fig 2.

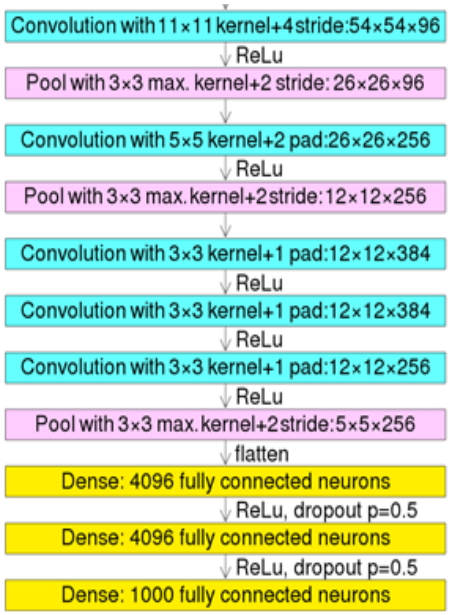


Fig 2 Fast RCNN Architecture of AlexNet

Fig 2 depicts AlexNet's core architectural design. Deep CNN's learning power was limited due to hardware limitations designs in the early 2000s, limiting them to tiny sizes. To circumvent hardware limitations, Alexnet was trained in parallel on two NVIDIA GTX 580 GPUs limitations and take advantage of deep RCNNs' representational capability. To make fast RCNN suitable to a larger range of image formats, AlexNet's the depth was increased from 5 to 8 layers (LeNet). Despite the fact that the depth of the ocean is growing, increases generalization for various image resolutions, overfitting is the main downside. Krizhevsky [12] concept to address this problem, in which their method during training, some transformational units are skipped at random to drive the model to learn more durable characteristics.

Furthermore, ReLU was used as a non-saturating activation function to speed up convergence by alleviating the vanishing gradient issue. To boost generalization by decreasing overfitting, local response and overlapping subsampling normalization were also used. Furthermore, when compared to prior proposed networks, big size filters (11x11 and 5x5) were used in the early layers. AlexNet is important in the new generation of CNNs because of its It's a time-saving technique to learning, and it's proven to be effective ushered in a new era of CNN architecture research.

The true emergence of CNNs as an important technique in computer vision occurred when [12] submitted AlexNet, the first success of using CNNs in classifying the ImageNet data-set. AlexNet was the first network to use a conv topology with five layers, three-layer maxpool topology, and three-layer FC topology. AlexNet also added the ReLU activation tool, which substantially sped up the the training procedure AlexNet extracts 96 FMs in the first layer, 256 FMs in the second layer, and 384 FMs in the levels conv3, conv4, and conv5.

3.2 GoogleNet Architecture

The 2014-ILSVRC competition was won by GoogleNet, commonly referred to as Inception-V1. The fundamental goal of the GoogleNet design was to accomplish what has been dubbed Inception-V1. The great accurateness at a low cost of computation [4]. It was the first CNN to use an inception block, including the divide, transform, and merge method is used to make multi-scale convolutional alterations. Fig 6 depicts the conception block's architecture. GoogleNet replaces traditional convolutional layers in small blocks, similar to the idea of replacing each layer with a micro NN in the Network in Network (NIN) architecture [13]. This block consists of several size filters (1x1, 3x3, and 5x5) that collect spatial data at different scales, including fine and coarse grain levels. For Fast RCNN GoogleNet'sis shown in the Fig 3 which use of the split, transform, and merge techniques idea helps in the resolution of a learning difficulty.

In GoogLeNet, [4] to outperform VGG accuracy by 3.2 percent, utilize a CNN with 22 learned layers. To capture visual patterns at different scales, this network employs 9 micro-networks, each with a different filter size. 6. Each individual Inception module is a micro-network with a (3×3) max pool and parallelized (11), (33), and (55) convolution layers. Before the three-dimensional (1×1) convolutions, also known as bottleneck filters, are included in GoogleNet's (3×3) and (5×5) dimension reduction filters. Their job is to increase the depth of the data while reducing the processing complexity each layer and, as a result, its modelling power.

3.3 VGG-16 Architecture

VGG Convolutional neural networks (CNNs) have been successfully applied to image processing identification tasks has advanced architectural design research. In this regard, developed CNN architectures can benefit from a simple and effective design guideline. Their architecture, VGG, was layered and modular [14].

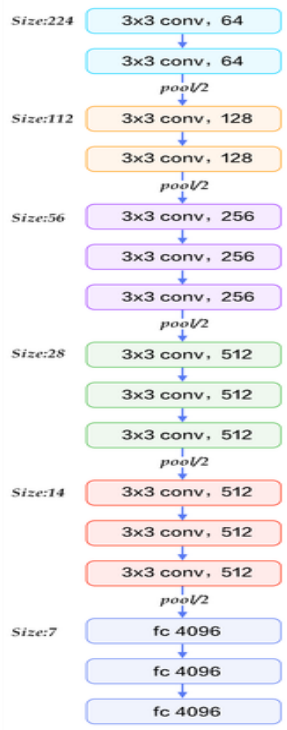


Fig 3 Fast RCNN Architecture of GoogleNet

In comparison to AlexNet and ZfNet, VGG was built 19 layers deep to reflect the depth-representational capacity link [12]. Small size filters can increase CNN performance, according to ZfNet, In the 2013-ILSVRC competition, a frontline network was created. The 11x11 and 5x5 filters were replaced by a layer stack of 3x3 filters by VGG based on these findings, and demonstrated experimentally that the simultaneous the usage of small (3x3) filters could amplify the effect of a large filter (5x5 and 7x7).

The usage of small size filters has the added benefit of minimizing the number of parameters, which reduces computing complexity. These findings are part of a new CNN study trend that employs lower size filters. VGG decreases a network's complexity by introducing 1x1 convolutions between convolutional layers, which also learn a linear combination of the feature-maps created. After the convolutional layer, max-pooling is introduced to adjust the network, while padding is utilized to maintain spatial resolution. In both picture classification and localization challenges, VGG performed admirably. In the 2014-ILSVRC competition, VGG came in second

place, although it quickly climbed to prominence because to its ease of use, homogeneous topology, and improved depth. The inclusion of 138 million parameters in VGG's major constraint rendered It is computationally intensive and impossible to implement for systems with limited resources is shown in the Fig 4.

The VGG-16 network in Fig 4 has the following precise structure: the first and second convolutional layers are made up of 64 feature kernel filters with a filter size of 33. As the input picture (RGB image with depth 3) travels through the first and second convolutional layers, its dimensions change to 224x224x64. With a stride of 2, the output is subsequently transferred to the max pooling layer.

The 124 feature kernel filters with a filter size of 33 make up the third and fourth convolutional layers. Following these two layers is a stride 2 max pooling layer, resulting in a 56x56x128 output.

The fifth, sixth, and seventh layers use convolutional layers with a kernel size of 33×256 feature maps are used in all three. A max pooling layer with stride 2 follows these layers.

Two sets of convolutional layers with kernel sizes of 33 are used in the eighth through thirteenth layers. There are 512 kernel filters in each of these convolutional layer sets. Following these layers is a max pooling layer with a stride of 1.

the VGG-16 ConvNet is also utilized to train the Fast R-CNN object detector. The VGG-16 ConvNet is also subjected to the transfer learning approach. The previous network's picture datastore was loaded into image labeler, and each emotion was given its own label. Each image's rectangular area of interest (ROI) label must be given. Because each image in this image datastore is a cropped image of a face expressing an emotion, with the region for each image labelled with the relevant emotion label was set to [1,1,224,224].

In 2014, VGG [14] improved the CNN architecture by adding 13 conv and 3 FC layers. The filters (5×5) and (11×11) are replaced by consecutive levels of filters (3×3) in this VGG model. As seen in Fig 5, this strategy preserves the size of a filter's receptive field while needing fewer computations.

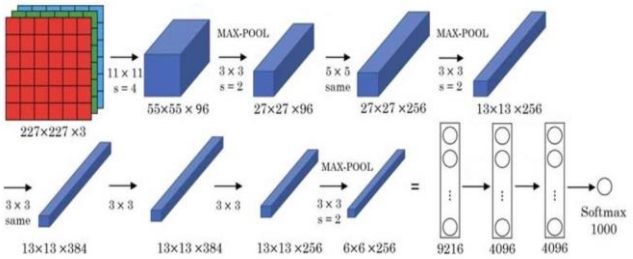


Fig 5 Decomposing (5 × 5) filters into two stages of (3 ×3) filters

By minimizing the computational cost Simonyan et al. may deepen the CNN in the first layers, improving classification performance on ImageNet by 14.8 percent. This network is offered in three depth-varying forms. It's worth noting VGG19, the deepest model, requires 27% more calculations than VGG16, the shallowest model achieve a 1% improvement in accuracy. The use of smaller filter kernels and hence deeper networks (up to 19 layers for VGG19 vs. 7 for AlexNet and ZFNet) as well as pre-training on shallower versions of the deeper networks gained popular.

4. Result and Discussion

4.1. Implementation of Alex Net with Fast RCNN

The architecture consists of Alex net are as follows. There are 5 convolutional layers and a fully connected layers in this image. These eight layers were merged with two novel ideas at the time. Their model gained an advantage thanks to MaxPooling and ReLU activation.

The Alex net architecture for Fast RCNN is trained to 120 epoch and the value for optimization is done with the for 50 Epoch. The plot using Tensor flow is shown in the Fig 6. The validation of the system is done with for the accuracy of the data for an epoch of 50.

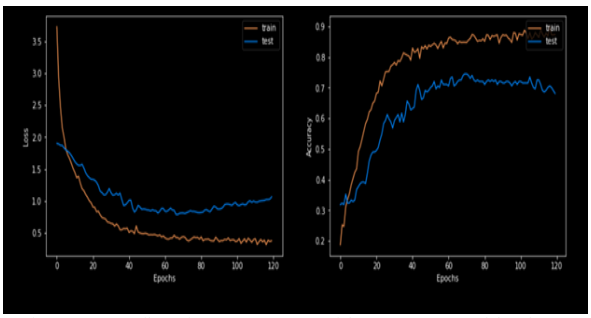


Fig 6 Tensor flow for an accuracy with Fast RCNN with AlexNet

The value of the training of the Alexnet architecture for Fast RCNN is shown for the optimized value of 50 epoach is shown in the Fig 7.The value of smoothed to 0.6596 for training of Alexnet, the validation value of the Fast RCNN is 0.9590 is shown in the

Name	Smoothed	Value	Step	Time	Relative
fit\run_2020_08_05-04_42_15\train	0.6596	0.6596	49	Wed Aug 5, 05:15:52	32m 53s
fit\run_2020_08_05-04_42_15\validation	0.9590	0.9590	49	Wed Aug 5, 05:15:53	32m 53s

Fig 7.

Fig 7 Tensor flow for smoothed values of validation with Fast RCNN AlexNet

4.2 Implementation of GoogleNet with Fast RCNN

GoogLeNe with Fast RCNN is shown in the Fig 8 which differs significantly from earlier state-of-the-art architectures like AlexNet and ZF-Net. It creates deeper architecture by employing a variety of techniques such as 11 convolution and global average pooling. The following are the architectural details of auxiliary classifiers: A pooling layer with an average filter size of 55 and a stride of 3. For dimension reduction and ReLU activation, an 11 convolution with 128 filters was used. With 1025 outputs and ReLU activation, this layer is fully connected. Dropout With a dropout ratio of 0.7, regularization is possible. The output of a softmax classifier with 1000 classes is comparable to the main softmax classsifier.

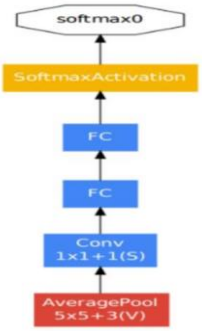
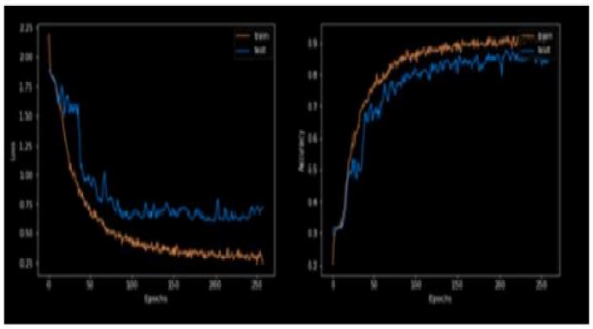


Fig 8 GoogleNet Architecture

The Google Net architecture for Fast RCNN is trained to 120 epoch and the value for optimization is done with the for 50 Epoch. The plot using Tensor flow is shown



in the Fig 9. The validation of the system is done with for the accuracy of the data for an epoch of 50.

Fig 9 Tensor flow for an accuracy with Fast RCNN with GoogleNet

The Googlenet architecture for Fast RCNN is shown for the optimised value of 50 epoach.The value of smoothed to 0.6596

Name	Smoothed	Value	Step	Time	Relative
fit\run_2020_08_05-04_42_15\train	0.6596	0.6596	49	Wed Aug 5, 05:15:52	32m 53s
fit\run_2020_08_05-04_42_15\validation	0.8976	0.8976	49	Wed Aug 5, 05:15:53	32m 53s

for training of Googlenet , the validation value of the Fast RCNN is 0.89876i shown in the Fig 10.

Fig. 10 Tensor flow for smoothed values of validation with Fast RCNN GoogleNet

4.3 Implementation of VGG-16 with Fast RCNN

The VGG16 network model is a convolutional network model. It is highly well-liked in for computer vision techniques,

it has already received the ImageNet 2014 prize competition. The top layers of this model have been removed, and new layers have been created in their place. Use the Flatten, Dense, Drop, and dense-SoftMax layers for classification. It is conceivable. The architecture of our model exemplifies this. The purpose of the drop layer is to Drop certain numbers at random to avoid overfitting. For this, the SoftMax layer is used. Emotions can be categorized in a number of ways. It was decided to use the ReLu activation function except for the last layer is shown in the Fig.11.

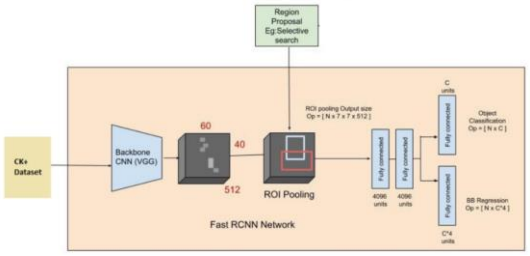
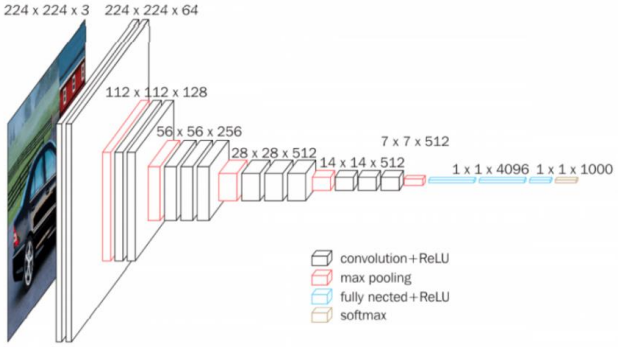


Fig.11 VGG-16 Architecture of Fast RCNN

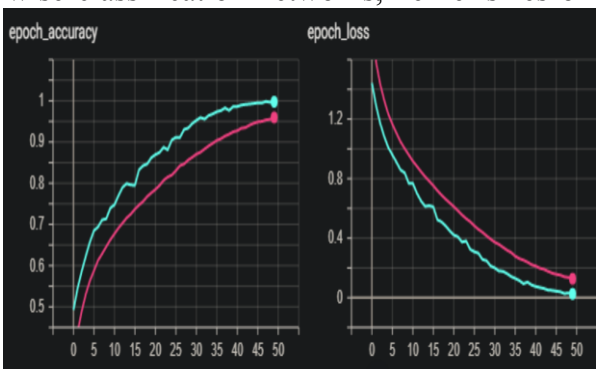
The VGG-Net architecture was created to see what effect increasing the size of the network will have. You can enhance the depth of a network by adding tiny convolution filters. As a result of the research, the VGG was created features cutting-edge performance in a network made up of 16-19 weight layers Large-scale picture classification was utilized at the time. The significance of this investigation was confirmed. Visual representations with a sense of depth. The initial design of the 16-



layer model is illustrated. All of these models have the extending route of the end of the model is shown in Fig. 12, which maps the output to the

Fig. 12 The dimensions of the VGG-16 filters and layers Configuration of Fast RCNN(Chen Zhang et al., 2015)

Before a network can be built, a large number of hyperparameters must be defined can be trained. There are various parameters that are required for setting up and training a network, in addition to those that define the training data set.The parameters were identified mostly by examining typical segmentation networks, such as, in pixel-wise classification networks, kernel sizes of



(3×3) and strides of 1 are typically utilized. The bulk of by trial and error, the parameters have also been fine-tuned which shown in the Fig. 13.

Fig. 13 Tensor flow for an accuracy with Fast RCNN with VGG-16

Larger batches yield a more precise gradient estimate, but when all 20 batches are processed at the same time, the gradient is overestimated. Batch size is typically limited by the amount of memory and computational power available. Small batch

Name	Smoothed	Value	Step	Time	Relative
fit\run_2020_08_05-04_42_15\train	0.9596	0.9596	49	Wed Aug 5, 05:15:52	32m 53s
fit\run_2020_08_05-04_42_15\validation	0.9976	0.9976	49	Wed Aug 5, 05:15:53	32m 53s

sizes can also help with regularization, potentially because to increased noise during the training process. It's also crucial that the mini-batches are selected at random [18].

Fig. 14 Tensor Flow for smoothed values of validation with Fast RCNN VGG-16

The value of the training of the VGG16 architecture for Fast RCNN is shown for the optimised value of 50 epoach is shown in the Fig 3.14.The value of smoothed to

0.9596 for training of VGG-16 , the validation value of the Fast RCNN is 0.9976.

5. Performance estimation of Fast CNN

1) Confusion Matrix

The Confusion Matrix consists of Normalized shape (SPTS) features for the emotional recognition of the CK+ Database is given in the table given below.

Table I Normalized Shape (SPTS) Features

	An	Di	Fe	Ha	Sa	Su	Co
An	35.0	40.0	0.0	5.0	5.0	15.0	0.0
Di	7.9	68.4	0.0	15.8	5.3	0.0	2.6
Fe	8.7	0.0	21.7	21.7	8.7	26.1	13.0
Ha	0.0	0.0	0.0	98.4	1.60	0.0	0.0
Sa	28.0	4.0	4.0	0.0	28.0	4.0	24.0
Su	0.0	0.0	0.0	0.0	4.0	100	0.0
Co	3.1	3.1	0.0	6.3	3.1	0.0	25.4

The Confusion Matrix consists of Canonical appearance (CAPP) features for the emotional recognition of the CK+ Database is given in the table given below.

Table II Canonical Appearance (CAPP) Features

	An	Di	Fe	Ha	Sa	Su	Co
An	70.0	7.5	5.0	0.0	5.0	2.5	5.0
Di	5.3	94.7	0.0	0.0	0.0	0.0	0.0
Fe	4.4	0.0	21.9	8.7	0.0	13.0	8.7
Ha	0.0	0.0	0.0	100	0.0	0.0	0.0
Sa	12.0	4.0	4.0	0.0	68.0	4.0	8.0
Su	0.0	0.0	0.0	0.0	4.0	96.0	0.0
Co	3.1	3.1	0.0	6.3	3.1	0.0	21.7

The Confusion Matrix consists for the combination of SPTS and CAPP characteristics emotional recognition of the CK+ Database is given in the table given below. The value of the best fit is given in the diagonal value and the value is the best fit for the emotional recognition of the system.

Table III Combination of Features (SPTS+CAPP)

	An	Di	Fe	Ha	Sa	Su	Co
An	75.0	7.5	5.0	0.0	5.0	2.5	5.0
Di	5.3	94.7	0.0	0.0	0.0	0.0	0.0
Fe	4.4	0.0	65.2	8.7	0.0	13.0	8.7
Ha	0.0	0.0	0.0	100	0.0	0.0	0.0
Sa	12.0	4.0	4.0	0.0	68.0	4.0	8.0
Su	0.0	0.0	0.0	0.0	4.0	96.0	0.0
Co	3.1	3.1	0.0	6.3	3.1	0.0	84.4

2) Accuracy

The percentage of all correctly predicted pixels divided by the total number of pixels predicted is the overall accuracy of an image segmentation prediction “(2)”. It's important to note that pixels with no annotations aren't counted. As a result, these are all values from the confusion matrix's diagonal, with the exception of the first item, which is ignored:

$$Overall\ accuracy = \frac{correctly\ prdicted\ pixels}{all\ predicted\ pixels} \qquad (1)$$

3) Precision

Precision and recall of each class have been determined for each split image from the test data set. The fraction of true positives (TP) among true and false positives (TP + FP) is known as precision.

$$precision = \frac{TP}{TP + FP} \qquad (2)$$

4) Recall

The fraction of true positives (TP) among true positives and false negatives (TP+ FN) is known as recall.

$$recall = \frac{TP}{TP + FN} \quad (3)$$

5) F-Score

The F-score is a statistic used to determine how accurate a forecast is equation-4. An F-score will be assigned to each class. The F-score, which has a perfect score of 1, combines precision (equation 3) with recall (equation 4).

$$F - score = \frac{2.precision.recall}{precision + recall} \quad (4)$$

The fraction of true positives (TP) among true positives and false negatives (TP+ FN) is known as recall.

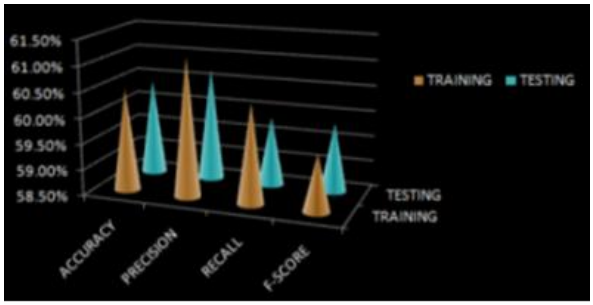


Fig 15 Performance Estimation of Fast RCNN with VGG-16

The Fig 15 is shows the performance of Fast RCNN with the VGG16 architecture model which consist of the accuracy, precision, recall, F-Score of the training and testing of the system using CK+ data base.

5. Conclusion

Design of the Structure of the Emotional Recognition system is done with hardware design architecture. The Fast RCNN uses various architectures such as AlexNet, GoogleNet and VGG-16. The performance of VGG-16 has accuracy of 99.79%. The performance of Fast RCNN is estimated accuracy, FSCORE, Recall and precision of the training with VGG-16. The CK+ database for real time emotional recognition for VGG-16 architecture is gives better performance.

Acknowledgement

This research was supported supported by Karunya Instittue of Technology and Sciences. We thank our colleagues from Karunya Institute of Technology, Department of ECE faculty supporting institution who provided insight and expertise that greatly assisted the research of this paper.

We thank Dr. Samson Immanuel for assistance with Fast RCNN technique, methodology], and Centre for Research in VLSI comments that greatly improved the manuscript.

References:

[1] Girshick, R., Donahue, J., Darrell, T., & Malik, J. "Rich feature hierarchies for accurate object detection and semantic segmentation". CVPR 2019, 1–8. http://arxiv.

[2] Wang, K., Dong, Y., Bai, H., Zhao, Y., & KunHu. "Use Fast R-CNN and Cascade Structure for Face Detection". VCIP 2016.Author, *Title of the Book*, Publishing House, 200X.

[3] Girshick, R. "Fast R-CNN". *Microsoft Research*. 2015, http://arxiv.org/abs/1504.08083

[4] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. "Going Deeper with Convolutions". *CVPR 2015*, 1–9.

[5] Pang, J., Chen, K., Shi, J., Feng, H., Ouyang, W., & Lin, D. "Libra R-CNN: Towards balanced learning for object detection". *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2019-June*, 821–830. https://doi.org/10.1109/CVPR.2019.00091

[6] Yamada, Y., Iwamura, M., & Kise, K. "Deep Pyramidal Residual Networks with Stochastic Depth". *Workshop Track - ICLR 2017*, 1–4. https://github.com/facebook/fb.resnet.torch.

[7] Lv, E., Wang, X., Cheng, Y., & Yu, Q. "Deep ensemble network based on multi-path fusion". *Artificial Intelligence Review*, *2019*, *52*(1), 151–168. https://doi.org/10.1007/s10462-019-09708-5

[8] Han, S., Mao, H., & Dally, W. J. "Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman

Coding". *ICLR 2016*, 1–14. http://arxiv.org/abs/1510.00149

[9] Xiong, Y., Kim, H. J., & Hedau, V. "ANTNets: Mobile Convolutional Neural Networks for Resource Efficient Image Classification". *Amazon Lab126*, 2019, 1–9. http://arxiv.org/abs/1904.03775

[10] Leon, V., Mouselinos, S., Koliogeorgi, K., Xydis, S., Soudris, D., & Pekmestzi, K. "A TensorFlow Extension Framework for Optimized Generation of Hardware CNN Inference Engines". *Technologies*, 2020, *8*(1), 1–15. https://doi.org/10.3390/technologies8010006

[11] Aydonat, U., O'Connell, S., Capalija, D., Ling, A. C., & Chiu, G. R. "An OpenCL™ deep learning accelerator on Arria 10". *FPGA 2017 - Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, *2017, 55–64.* https://doi.org/10.1145/3020078.3021738

[12] Krizhevsky, A., Sutskever, I., & Hinton, G. E. "ImageNet classification with deep convolutional neural networks". *Communications of the ACM*, *60*(6), *2017,* 84–90. https://doi.org/10.1145/3065386

[13] Ma, Y., Cao, Y., Vrudhula, S., & Seo, J. S. "Optimizing the Convolution Operation to Accelerate Deep Neural Networks on FPGA". *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, *26*(7), 2018, 1354–1367. https://doi.org/10.1109/TVLSI.2018.2815603

[14] Simonyan, K., & Zisserman, A. "Very Deep Convolutional Networks for Large-Scale Image Recognition". *ICLR 2015*, *2015, 1–14.* http://arxiv.org/abs/1409.1556

[15] Peemen, M., Setio, A. A. A., Mesman, B., & Corporaal, H. "Memory-centric accelerator design for convolutional neural networks". *IEEE 31st International Conference on Computer Design, ICCD 2013*, 13–19. https://doi.org/10.1109/ICCD.2013.6657019

[16] Yan, W. J., Li, X., Wang, S. J., Zhao, G., Liu, Y. J., Chen, Y. H., & Fu, X. "CASME II: An improved spontaneous micro-expression database and the baseline evaluation". *PLoS ONE*, *2014, 9*(1), 1–8. https://doi.org/10.1371/journal.pone.0086041

[17] Jouppi, N. P., Young, C., Patil, N., Patterson, D., Agrawal, G., Bajwa, R., Bates, S., Bhatia, S., Boden, N., Borchers, A., Boyle, R., Cantin, P. L., Chao, C., Clark, C., Coriell, J., Daley, M., Dau, M., Dean, J., Gelb, B., … Yoon, D. H. "In-datacenter performance analysis of a tensor processing unit". *Proceedings - International Symposium on Computer Architecture*, *Part F1286*, *2017, 1–12.* https://doi.org/10.1145/3079856.3080246

[18] Goodfellow, I. J., Erhan, D., Carrier, P. L., Bengio, Y. "Challenges in Representation Learning: A Report on Three Machine Learning Contests". *ICONIP 2013, Part III, LNCS 8228*, 117–124.