International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 10 Issue 5 ǁ 2022 ǁ PP. 115-121

Image Recognition using Deep Residual Learning

PUNDRU CHANDRA SHAKER REDDY[1*], G RAVI KUMAR[2], SUCHARITHA YADALA[3]
[1,2]*Associate Professor, Department of Computer Science and Engineering, CMR College of Engineering & Technology, Hyderabad, India*
[3]*Assistant Professor, Department of Computer Science and Engineering, CMR Institute of Technology, Hyderabad, India*

Abstract: *Deeper neural networks are more difficult to train. We present a residual learning framework to ease the training of networks that are substantially deeper than those used previously. We explicitly reformulate the layers as learning residual functions with reference to the layer inputs, instead of learning unreferenced functions. We provide comprehensive empirical evidence showing that these residual networks are easier to optimize, and can gain accuracy from considerably increased depth. On the ImageNet dataset we evaluate residual nets with a depth of up to 152 layers—8× deeper than VGG nets but still having lower complexity. An ensemble of these residual nets achieves 3.57% error on the ImageNet test set. This result won the 1st place on the ILSVRC 2015 classification task. We also present analysis on CIFAR-10 with 100 and 1000 layers. The depth of representations is of central importance for many visual recognition tasks. Solely due to our extremely deep representations, we obtain a 28% relative improvement on the COCO object detection dataset. Deep residual nets are foundations of our submissions to ILSVRC & COCO 2015 competitions1, where we also won the 1st places on the tasks of ImageNet detection, ImageNet localization, COCO detection, and COCO segmentation.*

Key Terms: *Deep residual learning, deep learning, image recognition, neural networks*

Date of Submission: 15-05-2022 Date of acceptance: 30-05-2022

I. INTRODUCTION

Deep convolutional neural networks have led to a series of breakthroughs for image classification. Deep networks naturally integrate low/mid/highlevel features [1] and classifiers in an end-to-end multilayer fashion, and the "levels" of features can be enriched by the number of stacked layers (depth). Recent evidence [2] reveals that network depth is of crucial importance, and the leading results [3] on the challenging ImageNet dataset all exploit "very deep" models, with a depth of sixteen to thirty. Many other nontrivial visual recognition tasks have also greatly benefited from very deep models.

Driven by the significance of depth, a question arises: Is learning better networks as easy as stacking more layers? An obstacle to answering this question was the notorious problem of vanishing/exploding gradients [4], which hamper convergence from the beginning. This problem, however, has been largely addressed by normalized initialization and intermediate normalization layers, which enable networks with tens of layers to start converging for stochastic gradient descent (SGD) with backpropagation [5].

When deeper networks are able to start converging, a degradation problem has been exposed: with the network depth increasing, accuracy gets saturated (which might be unsurprising) and then degrades rapidly. Unexpectedly, such degradation is not caused by overfitting, and adding more layers to a suitably deep model leads to higher training error, as reported in [6] and thoroughly verified by our experiments. Fig. 1 shows a typical example.

The degradation (of training accuracy) indicates that not all systems are similarly easy to optimize. Let us consider a shallower architecture and its deeper counterpart that adds more layers onto it [7]. There exists a solution by construction to the deeper model: the added layers are identity mapping, and the other layers are copied from the learned shallower model. The existence of this constructed solution indicates that a deeper model should produce no higher training error than its shallower counterpart [8]. But experiments show that our current solvers on hand are unable to find solutions that are comparably good or better than the constructed solution (or unable to do so in feasible time).

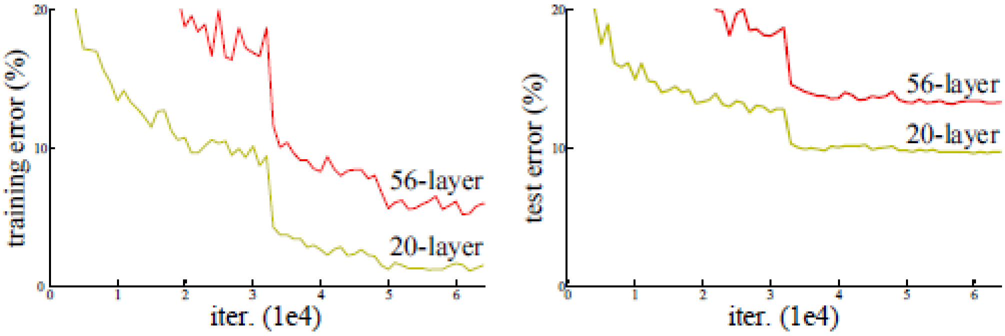


Figure 1. Training error (left) and test error (right) on CIFAR-10 with 20-layer and 56-layer "plain" networks.

In this paper, we address the degradation problem by introducing a deep residual learning framework. Instead of hoping each few stacked layers directly fit a desired underlying mapping, we explicitly let these layers fit a residual mapping. We present comprehensive experiments on ImageNet [9] to show the degradation problem and evaluate our method. We show that: 1) Our extremely deep residual nets are easy to optimize, but the counterpart "plain" nets (that simply stack layers) exhibit higher training error when the depth increases; 2) Our deep residual nets can easily enjoy accuracy gains from greatly increased depth, producing results substantially better than previous networks [10].

Similar phenomena are also shown on the CIFAR-10 set, suggesting that the optimization difficulties and the effects of our method are not just akin to a particular dataset. We present successfully trained models on this dataset with over 100 layers, and explore models with over 1000 layers. On the ImageNet classification dataset [11], we obtain excellent results by extremely deep residual nets. Our 152- layer residual net is the deepest network ever presented on ImageNet, while still having lower complexity than VGG nets [12]. Our ensemble has 3.57% top-5 error on the ImageNet test set, and won the 1st place in the ILSVRC 2015 classification competition. The extremely deep representations also have excellent generalization performance on other recognition tasks, and lead us to further win the 1st places on: ImageNet detection, ImageNet localization, COCO detection, and COCO segmentation in ILSVRC & COCO 2015 competitions. This strong evidence shows that the residual learning principle is generic, and we expect that it is applicable in other vision and non-vision problems.

II. RELATED WORKS

In image recognition, VLAD [13] is a representation that encodes by the residual vectors with respect to a dictionary, and Fisher Vector can bebformulated as a probabilistic version of VLAD. Bothbof them are powerful shallow representations for image retrievalband classification. For vector quantization, encoding residual vectors [14] is shown to be more effective than encoding original vectors. In low-level vision and computer graphics, for solving Partial Differential Equations (PDEs), the widely used Multigrid method reformulates the system as subproblems at multiple scales, where each subproblem is responsible for the residual solution between a coarser and a finer scale. An alternative to Multigrid is hierarchical basis preconditioning, which relies on variables that represent residual vectors between two scales. It has been shown [15] that these solvers converge much faster than standard solvers that are unaware of the residual nature of the solutions. These methods suggest that a good reformulation or preconditioning can simplify the optimization.

Practices and theories that lead to shortcut connections [16] have been studied for a long time. An early practice of training multi-layer perceptrons (MLPs) is to add a linear layer connected from the network input to the output. In [17], a few intermediate layers are directly connected to auxiliary classifiers for addressing vanishing/exploding gradients. The papers of [15, 16] propose methods for centering layer responses, gradients, and propagated errors, implemented by shortcut connections. In [18], an "inception" layer is composed of a shortcut branch and a few deeper branches. Concurrent with our work, "highway networks" present shortcut connections with gating functions [19]. These gates are data-dependent and have parameters, in contrast to our identity shortcuts that are parameter-free. When a gated shortcut is "closed" (approaching zero), the layers in highway networks represent non-residual functions. On the contrary, our formulation always learns residual functions; our identity shortcuts are never closed [20, 21], and all information is always passed through, with additional residual functions to be learned. In addition, high-way networks have not demonstrated accuracy gains with extremely increased depth (e.g., over 100 layers) [22, 23].

III. PROPOSED METHODOLOGY

3.1. Residual Learning

Let us consider H(x) as an underlying mapping to be fit by a few stacked layers (not necessarily the entire net), with x denoting the inputs to the first of these layers. If one hypothesizes that multiple nonlinear layers can asymptotically approximate complicated functions2, then it is equivalent to hypothesize that they can asymptotically approximate the residual functions, i.e., $H(x) - x$ (assuming that the input and output are of the same dimensions). So rather than expect stacked layers to approximate H(x), we explicitly let these layers approximate a residual function $F(x): = H(x) - x$. The original function thus becomes $F(x) +x$. Although both forms should be able to asymptotically approximate the desired functions (as hypothesized), the ease of learning might be different.

This reformulation is motivated by the counterintuitive phenomena about the degradation problem (Fig. 1, left). As we discussed in the introduction, if the added layers can be constructed as identity mappings, a deeper model should have training error no greater than its shallower counterpart. The degradation problem suggests that the solvers might have difficulties in approximating identity mappings by multiple nonlinear layers. With the residual learning reformulation, if identity mappings are optimal, the solvers may simply drive the weights of the multiple nonlinear layers toward zero to approach identity mappings. In real cases, it is unlikely that identity mappings are optimal, but our reformulation may help to precondition the problem. If the optimal function is closer to an identity mapping than to a zero mapping, it should be easier for the solver to find the perturbations with reference to an identity mapping, than to learn the function as a new one. We show by experiments that the learned residual functions in general have small responses, suggesting that identity mappings provide reasonable preconditioning.

3.2 Identity Mapping by Shortcuts

We adopt residual learning to every few stacked layers. A building block is shown in Fig. 2. Formally, in this paper we consider a building block defined as:

$$y = F(x, \{Wi\}) + x \quad (1)$$

Here x and y are the input and output vectors of the layers considered. The function $F(x, \{Wi\})$ represents the residual mapping to be learned. For the example in Fig. 2 that has two layers, $F = W2\sigma\ (W1x)$ in which σ denotes ReLU [29] and the biases are omitted for simplifying notations.

The operation F + x is performed by a shortcut connection and element-wise addition. We adopt the second nonlinearity after the addition.

The shortcut connections in Eq. (1) introduce neither extra parameter nor computation complexity. This is not only attractive in practice but also important in our comparisons between plain and residual networks. We can fairly compare plain/residual networks that simultaneously have the same number of parameters, depth, width, and computational cost (except for the negligible element-wise addition). The dimensions of x and F must be equal in Eqn.(1). If this is not the case (e.g., when changing the input/output channels), we can perform a linear projection Ws by the shortcut connections to match the dimensions:

$$y = F(x, \{Wi\}) +Wsx \qquad (2)$$

We can also use a square matrixWs in Eqn.(1). But we will show by experiments that the identity mapping is sufficient for addressing the degradation problem and is economical, and thus Ws is only used when matching dimensions. The form of the residual function F is flexible. Experiments in this paper involve a function F that has two or three layers (Fig. 5), while more layers are possible. But if F has only a single layer, Eqn.(1) is similar to a linear layer: $y = W1x+x$, for which we have not observed advantages. We also note that although the above notations are about fully-connected layers for simplicity, they are applicable to convolutional layers. The function $F(x, \{Wi\})$ can represent multiple convolutional layers. The element-wise addition is performed on two feature maps, channel by channel.

3.3. Network Architectures

We have tested various plain/residual nets, and have observed consistent phenomena. To provide instances for discussion, we describe two models for ImageNet as follows.

Plain Network. Our plain baselines are mainly inspired by the philosophy of VGG nets [40]. The convolutional layers mostly have 3×3 filters and follow two simple design rules: (i) for the same output feature map size, the layers have the same number of filters; and (ii) if the feature map size is halved, the number of filters is doubled so as to preserve the time complexity per layer.

We perform downsampling directly by convolutional layers that have a stride of 2. The network ends with a global average pooling layer and a 1000-way fully-connected layer with softmax. The total number of weighted layers is 34. It is worth noticing that our model has fewer filters and lower complexity than VGG nets [40] (Fig. 3, left). Our 34- layer baseline has 3.6 billion FLOPs (multiply-adds), which is only 18% of VGG-19 (19.6 billion FLOPs).

Residual Network. Based on the above plain network, we insert shortcut connections (Fig. 3, right) which turn the network into its counterpart residual version. The identity shortcuts (Eqn.(1)) can be directly used when the input and output are of the same dimensions. When the dimensions increase (dotted line shortcuts in Fig. 3), we consider two options: (A) The shortcut still performs identity mapping, with extra zero entries padded for increasing dimensions. This option introduces no extra parameter; (B) The projection shortcut in Eqn.(2) is used to match dimensions (done by 1×1 convolutions). For both options, when the shortcuts go across feature maps of two sizes, they are performed with a stride of 2.

3.4 Implementation

Our implementation for ImageNet follows the practice in [21, 40]. The image is resized with its shorter side randomly sampled in [256, 480] for scale augmentation [40]. A 224×224 crop is randomly sampled from an image or its horizontal flip, with the per-pixel mean subtracted [21]. The standard color augmentation in [21] is used. We adopt batch normalization (BN) [16] right after each convolution and before activation, following [16]. We initialize the weights as in [12] and train all plain/residual nets from scratch. We use SGD with a mini-batch size of 256. The learning rate starts from 0.1 and is divided by 10 when the error plateaus, and the models are trained for up to 60×104 iterations. We use a weight decay of 0.0001 and a momentum of 0.9. We do not use dropout [13], following the practice in [16]. In testing, for comparison studies we adopt the standard 10-crop testing [21]. For best results, we adopt the fully convolutional form as in [40, 12], and average the scores at multiple scales (images are resized such that the shorter side is in {224, 256, 384, 480, 640}).

IV. EXPERIMENTAL RESULTS AND DISCUSSIONS

4.1. ImageNet Classification

We evaluate our method on the ImageNet 2012 classification dataset [35] that consists of 1000 classes. The models are trained on the 1.28 million training images, and evaluated on the 50k validation images. We also obtain a final result on the 100k test images, reported by the test server. We evaluate both top-1 and top-5 error rates.

Plain Networks. We first evaluate 18-layer and 34-layer plain nets. The 18-layer plain net is of a similar form. The results in Table 1 show that the deeper 34-layer plain net has higher validation error than the shallower 18-layer plain net. To reveal the reasons, in Fig. 4 (left) we compare their training/validation errors during the training procedure. We have observed the degradation problem – the 34-layer plain net has higher training error throughout the whole training procedure, even though the solution space of the 18-layer plain network is a subspace of that of the 34-layer one.

Table 1. Top-1 error (%, 10-crop testing) on ImageNet validation.

	Plain	ResNet
18-Layers	27.94	27.88
34-Layers	28.54	25.03

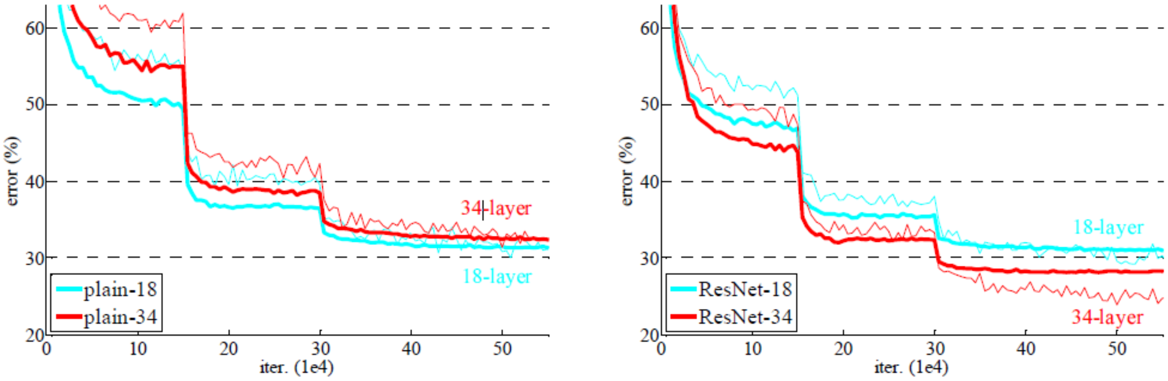


Figure 2. Training on ImageNet

We argue that this optimization difficulty is unlikely to be caused by vanishing gradients. These plain networks are trained with BN [16], which ensures forward propagated signals to have non-zero variances. We also verify that the backward propagated gradients exhibit healthy norms with BN. So neither forward nor backward signals vanish. In fact, the 34-layer plain net is still able to achieve competitive accuracy (Table 3), suggesting that the solver works to some extent. We conjecture that the deep plain nets may have exponentially low convergence rates, which impact the reducing of the training error3. The reason for such optimization difficulties will be studied in the future.

Residual Networks. Next we evaluate 18-layer and 34- layer residual nets (ResNets). The baseline architectures are the same as the above plain nets, expect that a shortcut connection is added to each pair of 3×3 filters as in Fig. 3 (right). In the first comparison (Table 1 and Fig. 2 right), we use identity mapping for all shortcuts and zero-padding for increasing dimensions (option A). So they have no extra parameter compared to the plain counterparts.

We have three major observations from Table 1 and Fig. 2. First, the situation is reversed with residual learning– the 34-layer ResNet is better than the 18-layer ResNet (by 2.8%). More importantly, the 34-layer ResNet exhibits considerably lower training error and is generalizable to the validation data. This indicates that the degradation problem is well addressed in this setting and we manage to obtain accuracy gains from increased depth.

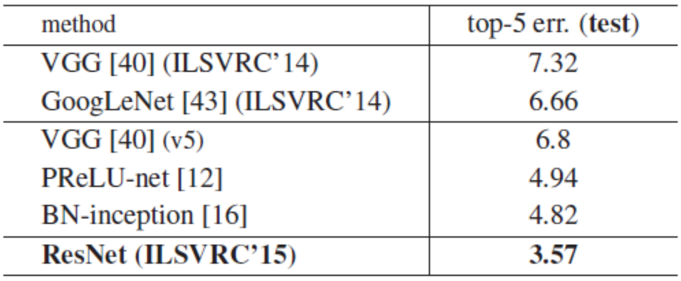
Second, compared to its plain counterpart, the 34-layer ResNet reduces the top-1 error by 3.5% (Table 1), resulting from the successfully reduced training error (Fig. 2 right vs. left). This comparison verifies the effectiveness of residual learning on extremely deep systems. Last, we also note that the 18-layer plain/residual nets are comparably accurate (Table 2), but the 18-layer ResNet converges faster. When the net is “not overly deep” (18 layers here), the current SGD solver is still able to find good solutions to the plain net. In this case, the ResNet eases the optimization by providing faster convergence at the early stage.

Comparisons with State-of-the-art Methods. In Table 2 we compare with the previous best single-model results. Our baseline 34-layer ResNets have achieved very competitive accuracy. Our 152-layer ResNet has a single-model top-5 validation error of 4.49%. This single-model result outperforms all previous ensemble results (Table 3). We combine six models of different depth to form an ensemble (only with two 152-layer ones at the time of submitting). This leads to 3.57% top-5 error on the test set (Table 3). This entry won the 1st place in ILSVRC 2015.

Table 2. Error rates (%) of single-model results on the ImageNet Valodation set

method	top-1 err.	top-5 err.
VGG [40] (ILSVRC’14)	-	8.43†
GoogLeNet [43] (ILSVRC’14)	-	7.89
VGG [40] (v5)	24.4	7.1
PReLU-net [12]	21.59	5.71
BN-inception [16]	21.99	5.81
ResNet-34 B	21.84	5.71
ResNet-34 C	21.53	5.60
ResNet-50	20.74	5.25
ResNet-101	19.87	4.60
ResNet-152	**19.38**	**4.49**

Table 3. Error rates (%) of ensembles.



method	top-5 err. (**test**)
VGG [40] (ILSVRC'14)	7.32
GoogLeNet [43] (ILSVRC'14)	6.66
VGG [40] (v5)	6.8
PReLU-net [12]	4.94
BN-inception [16]	4.82
ResNet (ILSVRC'15)	**3.57**

V. CONCLUSION

Our method has good generalization performance on other recognition tasks. Table 7 and 8 show the object detection baseline results on PASCAL VOC 2007 and 2012 and COCO. We adopt Faster R-CNN as the detection method. Here we are interested in the improvements of replacing VGG-16 [40] with ResNet-101. The detection implementation (see appendix) of using both models is the same, so the gains can only be attributed to better networks. Most remarkably, on the challenging COCO dataset we obtain a 6.0% increase in COCO's standard metric, which is a 28% relative improvement. This gain is solely due to the learned representations. Based on deep residual nets, we won the 1st places in several tracks in ILSVRC & COCO 2015 competitions: ImageNet detection, ImageNet localization, COCO detection, and COCO segmentation.

References

[1]. Reddy, P.C.S., Sucharitha, Y. and Narayana, G.S., 2021. Forecasting of Covid-19 Virus Spread Using Machine Learning Algorithm. International Journal of Biology and Biomedicine, 6.

[2]. Shaker Reddy, P.C. and Sureshbabu, A., 2020. An enhanced multiple linear regression model for seasonal rainfall prediction. International Journal of Sensors Wireless Communications and Control, 10(4), pp.473-483.

[3]. Sucharitha, Y., Vijayalata, Y. and Prasad, V.K., 2021. Predicting election results from twitter using machine learning algorithms. Recent Advances in Computer Science and Communications (Formerly: Recent Patents on Computer Science), 14(1), pp.246-256.

[4]. Reddy, P.C.S., Nachiyappan, S., Ramakrishna, V., Senthil, R. and Sajid Anwer, M.D., 2021. Hybrid Model Using Scrum Methodology for Softwar Development System. J Nucl Ene Sci Power Generat Techno 10, 9, p.2.

[5]. Reddy, P.C. and Babu, A.S., 2018. Usage of co-event pattern mining with optimal fuzzy rule-based classifier for effective web page retrieval. International Journal of Engineering & Technology, 7(3.29), pp.275-279.

[6]. Sucharitha, Y., Prasad, V.K. and Vijayalatha, Y., 2019. Emergent events identification in micro-blogging networks using location sensitivity. J. Adv. Res. Dynam. Cont. Syst, 11, pp.596-607.

[7]. Reddy, P.C.S., Pradeepa, M., Venkatakiran, S., Walia, R. and Saravanan, M., 2021. Image and Signal Processing in the Underwater Environment. J Nucl Ene Sci Power Generat Techno 10, 9, p.2.

[8]. Sucharitha, Y., Vinothkumar, S., Rao Vadi, V., Abidin, S. and Kumar, N., 2021. Wireless Communication without the Need for Pre-Shared Secrets is Consummate Via the Use of Spread Spectrum Technology. J Nucl Ene Sci Power Generat Techno 10, 9, p.2.

[9]. Balamurugan, D., Aravinth, S.S., Reddy, P., Rupani, A. and Manikandan, A., 2022. Multiview Objects Recognition Using Deep Learning-Based Wrap-CNN with Voting Scheme. Neural Processing Letters, pp.1-27.

[10]. Reddy, P.C. and Babu, A.S., 2017. A Novel Approach to Analysis District Level Long Scale Seasonal Forecasting of Monsoon Rainfall in Andhra Pradesh and Telangana. International Journal of Advanced Research in Computer Science, 8(9).

[11]. Reddy PC, Sucharitha Y, Narayana GS. Development of Rainfall Forecasting Model Using Machine Learning With Singular Spectrum Analysis. IIUM Journal of Engineering, 23(1), 172–186. https://doi.org/10.31436/iiumej.v23i1.1822

[12]. Reddy PC, Nachiyappan S, Ramakrishna V, Senthil R, Sajid Anwer MD. Hybrid Model Using Scrum Methodology for Softwar Development System. J Nucl Ene Sci Power Generat Techno 10. 2021;9:2.

[13]. Vemuri, R.K., Reddy, P.C.S., Kumar, P., Ravi, J., Sharma, S. and Ponnusamy, S., 2021. Deep learning based remote sensing technique for environmental parameter retrieval and data fusion from physical models. Arabian Journal of Geosciences, 14(13), pp.1-10.

[14]. Reddy, P. and Sureshbabu, A., 2019, June. An applied time series forecasting model for yield prediction of agricultural crop. In International Conference on Soft Computing and Signal Processing (pp. 177-187). Springer, Singapore.

[15]. Reddy, P.C. and Sureshbabu, A., 2019. An adaptive model for forecasting seasonal rainfall using predictive analytics. International Journal of Intelligent Engineering and Systems, pp.22-32.

[16]. Reddy, P.C. and Babu, A.S., 2017, February. Survey on weather prediction using big data analystics. In 2017 Second international conference on electrical, computer and communication technologies (ICECCT) (pp. 1-6). IEEE.

[17]. Sucharitha, Y., Vijayalata, Y. and Kamakshi Prasad, V., 2019. Analysis of early detection of emerging patterns from social media networks: a data mining techniques perspective. In Soft Computing and Signal Processing (pp. 15-25). Springer, Singapore.

[18]. Sucharitha, Y., Anantha Christu Raj P, Karthik TS, Kapila D, Mathiazhagan V, et al.,(2021) Implementing an Effective and Secure Resource Architecture for vlsi Block Encryption. J Nucl Ene Sci Power Generat Techno 10, 9, p.2.

[19]. Vasumathi, D., Chandrashekar-Reddy, P. and Sucharitha, Y., 2012. Analysis of clustering algorithms. International Journal of Emerging Trends in Engineering and Development, 4(2), p.7.

[20]. Ashreetha, B., Devi, M. R., Kumar, U. P., Mani, M. K., Sahu, D. N. and Reddy, P. C. S. (2022) "Soft optimization techniques for automatic liver cancer detection in abdominal liver images", International journal of health sciences, 6(S1).

[21]. Prasath, A. S. S., Lokesh, S., Krishnakumar, N. J., Vandarkuzhali, T., Sahu, D. N. and Reddy, P. C. S. (2022) "Classification of EEG signals using machine learning and deep learning techniques", International journal of health sciences, 6(S1)

[22]. Lokesh, S., Priya , A., Sakhare, D. T., Devi, R. M., Sahu, D. N. and Reddy, P. C. S. (2022) "CNN based deep learning methods for precise analysis of cardiac arrhythmias", International journal of health sciences, 6(S1)

[23]. Chary, B.D., Sucharitha, Y. and shaker Reddy, P.C., Way to Learning Dropouts in Distance Education-A Study using Data mining Techniques.