Estimating the Scalability of Various Modern Deep Learning Models on a Single Data Set: A Theoretical Analysis

Sai Sathwik Kosuru
Department of Computer Science and Engineering
B V Raju Institute of Technology,
Narsapur, Medak District,
Telangana State 502313, India.
saisathwik363@gmail.com

Padmanabha Reddy Y C A*
Department of Artificial Intelligence and Machine Learning
Chaitanya Bharathi Institute of Technology,
Gandipet, Ranga Reddy District,
Telangana State 500075, India.
ananthayc.reddy@gmail.com

Shreya Akiti
Department of Computer Science and Data Science
B V Raju Institute of Technology,
Narsapur, Medak District,
Telangana State 502313, India.
shreyareddy072@gmail.com

Rishanka Madamshetti
Department of Computer Science and Engineering
B V Raju Institute of Technology,
Narsapur, Medak District,
Telangana State 502313, India.
rishanka123@gmail.com

***Abstract*—Diversification of Learning algorithms is critical to the functioning of smart computing systems wherein there is a need for deriving insights, abstracting all the complex architectures beneath. Modeling the human brain, these networks/algorithms work on inputs through suitable weight and bias assignments undergoing multiple iterations of learning after that, and ultimately ending in the learning of the input patterns. This research focuses on the scalabilities of such heavy applications on a diverse range of existing data sets. Since these models are complex, they may easily overfit over smaller and simpler data sets. So, we have considered a multi-class small-sized data source to perform apriori and posteriori analyses of the networks. The research gives a clear picture of these algorithms' scalability and performance wherein a few algorithms failed in the initial stages, some of them overfitted, and others could not run when the hardware was limited. Various ideas were also considered in the process of selecting the most suitable network. In the end, we proposed the critical aspects of considering system-cum-environmental scalabilities across Compilation& Platform Suitabilities, Architectural Dependencies, & Network Analyses.**

***Index Terms*—Lightweight networks, Scalability, Overfitting, Learning Rate, Accuracy**

I. INTRODUCTION

Images are composed of a collection of pixels, their semantic meanings, and color distributions, with alternatives as gradient distributions. Identifying the overall image is an attribute of Multi-class learning, and segmenting the contours within the color distributions to classify labeling information in images, is termed Multi-label learning. With perspectives from the human consciousness, it may be viable to recognize the patterns of nature owing to the adaptive synapses of human neurons. It may be difficult to entirely mimic human brain behavior for implementing such techniques in a computer. So, various frameworks are being proposed in this direction. As a result of the complexity of such requirements, modern frameworks are much deeper and very complex to implement.

The problem with deep learning models is that they are computationally intense and architecturally complex though their functioning abstracts all the underlying things. Obscured under this abstraction, there is a much-needed aspect of scalability. Not every model may be used in every application thereby raising a need for testing the scalability of models. Though lightweight networks have been implemented to adapt themselves to simple devices, there are still some backlogs in terms of performance and platform requirements. Additionally, there may be a problem in the purview of representation and observation where the recent additions of these lightweight networks to the existing systems complicate the process as there are only latent observations for all of them. This is the reason why Research in these aspects needs to be considered and considerably expanded.

Most of the existing works focus on the aspect of performance of models quantized in accuracy calculations but do not analyze the empirical architectural characteristics and usage capabilities. Existing systems propose separate architectures for separate requirements, data sets, and data types which enforce the avoidance of one-for-many models. It is theoretically possible to introduce such types of models which can be modified with architectural parametrical variations and dynamic modeling of the sequential structure of these models.

*Corresponding author.

The current study differs from the existing ones in the order of phase where the existing systems focus on the process before and after designing of models while the current system proposes the process of modifying the model in situ while it is being designed.

Behavioral dynamics of Deep learning models help organizations and researchers understand the models' performance on an architectural and performance basis and serve to explain the cornerstones in dataset-based approaches of models. Error gradients are the most common but accuracy gradients are the least considered. Though accuracy gradients are highly subjective, they can be considered to apply early stopping to gain sufficient accuracy. Subsequent iterations are not required after desirable accuracy is reached. This attribute is the primary consideration of the current research.

II. Literature Survey

A. Existing Studies

Accuracy	Learning Rate	Performance	Fixed Architecture

TABLE I
: Focus of Existing Works

Deep Learning Architecture, in limited computing resource region [7]s, behaves neutrally concerning the model when we consider their performances. The main challenge in this context is the problem of computational requirements for the network training which is way too complex and resource-consuming. Though new architectural designs and dimensions have been introduced dynamically throughout recent years, architectural parameters need to be tuned to the application and the data set. The Architectural modifications in the existing systems include different numbers of nodes, different interconnections, distinct layer arrangements, diverse layer types, and various values of depth [2] [4]. Existing networks proposed a directed approach to expected predictions and optimal accuracies. The existing architectural designs work towards layer-by-layer successive feature extractions, dimensionality reductions, and machine learning to determine the final prediction which may itself involve recursive brute force implementations [8].

During previous studies, feature space is identified, constituent elements traced and their types used to cluster the features into designated feature groups. The feature extraction is followed by feature engineering to select only the suitable features that can benefit the model/network with sufficient outcome-oriented accuracies where these features aid the developer in synthesizing the required network(table I). All of the existing networks and models follow this same approach [8] [9] where we employ reverse engineering that has never been done before. The authors performed a similar study on Machine Learning Models previously and recorded their performances in a research paper where the work was presented at a research venue and is yet to be published [11]. This is a modification and an extension of that work.

B. Literature Review

Previous surveys are critical to leveraging the impact of referential and technology-oriented pessimal views of the technical requirements for a model functioning [1]. Existing works propose scalability in terms of deployment standards but contain limited information about how well they are suited to a diverse range of datasets. It is a fact that deep learning models' behavioral statistics are not consistent and are predictable universally. Despite having a lot of attempts in the past, there is little research on this particular aspect when Deep learning models are considered [2] [3] [7]. There is a need for reverse engineering to improve the efficiency of the existing models thereby amplifying the impact of the current work. Existing works contain limited discussion on several aspects concerning model functioning since most of the researches focus implementing on real-time use cases. In this context, past research enables the usage of a diverse array of model behaviors and framework architectures in solving high-stake real-world problems [5] [7] [9]. Though we have not focussed on this particular aspect in our current work, part of our research contributes to this.

Image segmentation, contour distribution-based semantics, and segment anything models [6] have been of major focus in recent decades. As new techniques, algorithms, and models are being introduced, there has been a need for enhancing application-oriented research to produce more of such kind. Most of the current models are built on an "IS A" relationship pattern [8], much similar to the inheritance concept of OOPs. Segmentation considerations [6] and comparative analysis of models took place in the previous findings. As such, models with the best accuracy are chosen and the generalization capabilities are tested to a limited extent. There is a need to leverage investigations in the latter phase which is a primary cause of motivation in this work [10]. Architectural-scalability relations are not seen in many of the works [10] while they just underscore the accuracy improvements [8] and static test results which happen to show incomplete results. To accomplish the dynamic nature of evolutional computing which extensively involves new deep learning models and novel frameworks, it has been imperative to take architectural analysis and evaluate the scalability of models across varied data sets. Owing to the continued usage of lightweight networks, we can currently focus our work completely on the scalability of lightweight networks.

III. Methodology

It is to be noted that the findings here are indicative of an extensive and time-consuming process performed beforehand followed by a trial-and-error format of observations in the direction of insightful decisions in the end. The Purposive or judgment sampling technique may not be the appropriate technique when considering the generalization capabilities of models' performances. Real-time analysis of situations and problems may not be consistent with universal situations.

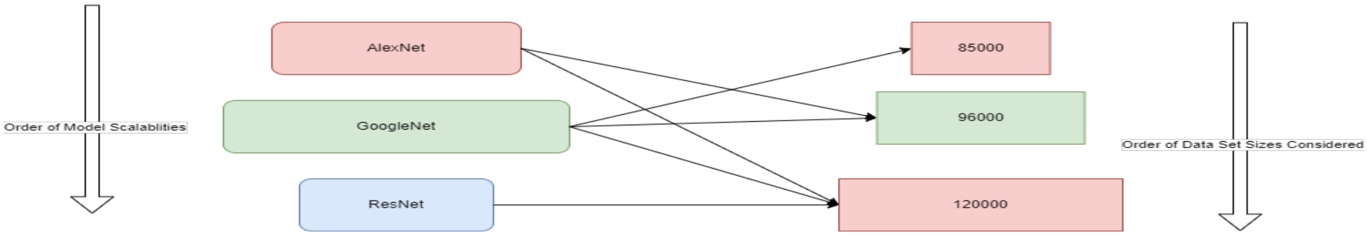

Fig. 1. Color-Size Matching for Diagrammatic Representation of Respective Model Scalabilities

Surprisingly, Environmental constraints also played an important role in deciding the models' efficiency. Considering all of these constraints, the research has progressed on three different fronts- Apriori analysis of Lightweight networks and profiling of the Networks' behavior across all the epochs of training. The technical framework of the methodology is seen in Figure 2 where we use 3 primary elements- data sets, lightweight networks, and the reported results.

A. Data set

A multi-class large-sized data set was taken to analyze and trace the performance of algorithms. Biological phyla and disease classifications are diverse when compared to other existent data sets. Though genetic data is vastly diverse, it is too complex to perform the primary investigation upon. Hence, the current work has considered a plant leaf disease data set. It has 1,20,000 annotated images, arranged into 10 different directories with 4 different sub-directories, and the bottommost level directory is train and valid directories for different differentiators to be considered for training and testing purposes. The considered data set is a Kaggle source inserted for research purposes.

B. Network Analysis

Lightweight networks are modern frameworks/architectures explicitly designed to reduce computational complexity and resource expenditures. These are relatively deeper compared to the traditional algorithms, with deeper layers and three-dimensional considerations of training added as specialization layers to basic frameworks. Basic frameworks follow the blueprint of all the traditional networks but employ additional parameters of a deeper analysis of patterns and more accurate predictions. General intuitions will be that these will be computationally expensive thus underscoring the need to make them less architecture-dependent and more function-efficient. Hence, this has been the basis for the motivation of designing lightweight networks. The current field of interest is advancing an apriori analysis of these models to reciprocate the apriori analysis in terms of scalability perspectives. It is worth noting that the scalability here does not refer to deployment factors but entirely depends on the functioning patterns of the model.

*1) **ResNet**:* ResNet is the acronym for Residual Network and is the network proposed by Microsoft. Forward and backward informational flows are of significance here since they help the network resolve errors and defects on an iterative basis in an epoch-based approach compounded with layer-wise computations. Residual means 'in-situ' behavior associated with things and attributes and may call for novel storage capabilities and innovative procedures to simplify the process and amplify the use of such approaches. Vanishing or exploding gradients are the major things to be considered for this model performance. Vanishing gradient explores the possibility of swiftly falling accuracy in the training cycles and the exploding suggests that there is some suspicious anomaly in the behavior of the network. As a result, this network consists of Residual Blocks which are skipped in the layer-wise flow of information throughout the network. Explicit and automatic selections of these blocks which may happen due to discretion or in a random manner may impact the performance characteristics of the network and there is no predefined procedure to assess the impact of a selection. Judgment-based selections also result in surprisingly low accuracy, thus far enabling researchers in this field to explore more of this architecture. Moreover, regularization in the traditional techniques disables overfitting whereas in the case of lightweight networks focus on reducing error to the minimum possible extent.

Exploding/Vanishing gradient is a problem that greatly impedes the learning abilities and the working capabilities of a network where it is critical to align the gradient descent to our needs. Therefore, Resnets employ the use of Residual Blocks where we skip some layers while connecting between layers of the network where the architecture is dynamic according to the problem. The implementation of shortcuts makes the networks robust and hence, the scalability of the network can be mapped accordingly.

*2) **GoogleNet**:* GoogleNet, with support from Google, having been focussed on going deeper with convolutions, uses 1x1 convolutions and average pooling in its architecture. The main differentiator for GoogleNet is its ability to be designed deeper and deeper with a primary focus on reducing parameters and a secondary focus on parameter downscaling. With inception layers, the model architecture invokes as many as 22 Layers where computational efficiency is also considered. Auxiliary classifiers present near the output layers amplify the network's performance by 30%-50%. While the network is used for both Detection and Classification tasks, It is imperative to know its scalability to enable the model's computational efficiency in all the concerned applications.

*3) **AlexNet**:* AlexNet is relatively simple in architecture where the common nodal depth is 5 where we can make modifications by adding suitable strides to a normal CNN. The Maxpooling layers are made to prevent them from overlapping with variations in their padding ratios.

C. Architectural Designs

System	Computational Complexity	Crash and Recovery
Platform	Framework Support	Backend tools
Time	Epoch-wise Analysis	Parametric Tuning
Performance	Accuracy	Training Time
Results	Subjective Analysis	Quantitative Analysis

TABLE II
: SCALABILITY FACTORS CONSIDERED IN THE STUDY

The change rate of 5-10 layered differences was added to modify the existing architectures with 50% focus on changing the number of layers, 30% on hyperparameter tuning, and 20% on the number of hyperparameters used. While these architectural changes do not pose any functional differences to the network, we can further decide the dynamics of architectural changes with the result parameters such as accuracy, learning rate, and training time.

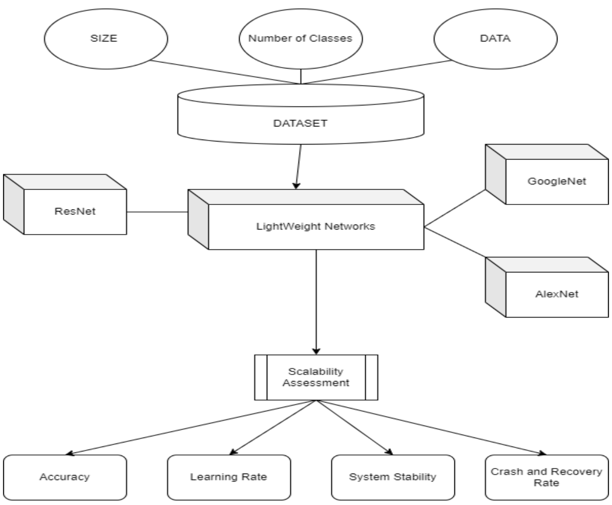

Fig. 2. Process flow of the working system

*1) **GoogleNet**:* We followed a recursive methodology to modify the network layer by layer with a simultaneous focus on the upper and lower layers, which we term a nested loop of changes. Owing to the complexity of the architecture with its optimal depth of 22 layers, there is a need for the use of OOP principles like abstraction, object notations, and class-based architectures to implement modularity among the layer components. We tuned the Inception V3 layers where the weight parameter is set to an ImageNet-based weighting mechanism. A multi-class data set with 1 lakh images is considered and hence, there is a need for a robust pre-trained model to serve the purpose of saving time through transfer learning procedures. Owing to the architectural complexity, data augmentation and preprocessing of the image data have to be done well in advance to perform pure tests on the network architecture.

*2) **ResNet**:* We used the torchnet implementation and class concept to modularize the architecture of the residual blocks present in the ResNets. We need to simulate skip connections and demonstrate highway connections while implementing the architectural decisions for the network. To demonstrate filter significance, a different number of filter combinations with varied filter limits is applied to modulate the learning depth of the model. ReLu was the most prominent optimizer employed where an appropriate kernel regularizer is used for regularizations in skip connections of the network. A sufficient amount of dropout layer quantity, about 30% of the total portion of layers, is implemented.

*3) **AlexNet**:* The architectural variations of the AlexNet module were not implemented and we just considered the scalability of the model. The ReLu activation function was implemented and appropriate strides were added in every network layer. While the dropout usage is less, it is still used in the last 4 or 5 layers.

D. Compilation and Platform Scalability of the Lightweight Neural Networks

We found that Google Colab was not suitable for the implementation of such computationally complex networks where insufficient GPU is provided and we used Jupyter as the runtime platform for all our investigations. In the cases of the minute scale of the dataset sizes, we employed Kaggle IDE to test compilation scalabilities with different hyperparameters shown in Table III.

*1) **GoogleNet**:* **System Overloads:** System has seldom been overwhelmed with the training process of GoogleNet where Google Colab also performed reasonably well. The time taken for compilation and training was also

less than 7.5 hours. No system crash was reported.
Epoch Consistency: Accuracy was steady throughout the investigation with variations of 0.15-0.245% accuracy differences between consecutive epochs and the final accuracy has no significant offset from the initial value.
Performance over smaller datasets: Conclusively, there are notable numerical or qualitative differences/margins between the scalability parameter and the dataset size for a network. The equation 1 shows how inception parameter Inc[x,y] adds over CNN parameters like H[x,y] and F[x,y] in analysis.

$$G[i,j] = \sum_{n=-k}^{k} \sum_{n=-k}^{k} H[u,v](F[i+u,j+v]+Inc[u,v]) \quad (1)$$

2) ResNet: **System Overloads:** ResNet implementation over smaller datasets has proven to be computationally expensive with observed 2-hour interval system crashes and heat consumption of the physical hardware.
Epoch Consistency: We noted a remarkable variation of accuracies with differences in epochs following a trapezoidal relation.
Performance over Smaller Datasets: ResNets proved moderate for smaller datasets where the accuracy-dataset size curve was a smooth negative slope curve. Hence, ResNets have almost poor performance in this case. The numerical explanation is in 2 where Residual layers' behavioral numerical equivalents are added as Res[x,y] to the existent parameters of CNN.

$$G[i,j] = \sum_{n=-k}^{k} \sum_{n=-k}^{k} H[u,v](F[i+u,j+v] + Res[u-1,v]) \tag{2}$$

l1	l2	l3	Additional layer
32	32	64	64(dropout)
32	64	64	32(dropout)
32	64	64	128(dropout)
32	64	128	128(dropout)
32	64	64	32(dropout)

TABLE III
CONSIDERED HYPERPARAMETERS: ARCHITECTURAL MODIFICATIONS

3) AlexNet: **System Overloads:** AlexNet training process followed the alternating sequences of system failures and training progresses. In certain cases, the whole system was corrupted with long-running processes thus implying the high complexity of the network.
Epoch Consistency: AlexNet was the simplification of the trapezoidal meshes observed in the case of ResNets. Though both have different functionality and purposes, remarkable similarities have been solicited.
Perforance over Smaller Datasets: AlexNet has straightforward moderate performance overall dataset sizes.

IV. RESULTS AND DISCUSSIONS

The scalability has been numerically evaluated over the considered dataset with selected factors/attributes as seen in Table IV with contour representation displayed in the Figure. 1.

1) GoogleNet : **Experimental Observations:** The total computations performed over 10+ epochs produced sufficient accuracies for a robust implementation whereas an inferior epoch count suggests the need for a deeper-layered approach by applying multiple layers. 80% of epochs are covered under a 0.5-1.0 accuracy regime and a reasonable number of layers came under lower accuracies.

Generalized Inference: The difference among the sizes of the data set has little impact on GoogleNet performance with 60% performance elasticity considerations and 40% infrastructure requirement scalability. Intuitively, the depth of the network decides the scalability of the model over various data sets of various sizes where the network analyzes the features in depth. While GoogleNet is of higher accuracy on large datasets, it even shows an accuracy of 0.924 in all possible hyperparameter combinations.

2) ResNet : **Experimental Observations:** While the residual blocks and skip connections enable effective computational power use, the process is tedious enough to sustain a higher number of layers where though more layers facilitate higher power of the resulting model, we need to waiver the usage of the network at a wider scale for training ResNets. Due to the above factor, the accuracy for the situation was found to be lying in the 0.5-0.7 accuracy region.
Generalized Inference: Residual Network follows trapezoidal relations over 95% of data set sizes with triangular peaks and troughs at various intervals. Epoch-wise consistency of accuracy proves the same for the data sets of any size and hyperparameter combinations produce behavior similar to CNNs with modified ranges. Seemingly simple, the accuracy variation is actually complex and differs greatly from Ground Truth, and hence, improvisation of hyperparameter combinations is a mandatory requirement for such networks.

3) AlexNet : **Experimental Observations:** AlexNets and ResNets follow similar behaviors over datasets of various sizes with minute variations in their values. AlexNets is a power-intensive algorithm that consumes higher computational power and we observed a rise in temperatures of the CPU and frequent system failures in the process. The network follows a trapezoidal accuracy range with less peaks/troughs implying almost consistent accuracy throughout the training process. The accuracy was reported to be at 75%.
Generalized Inference: AlexNets may not be feasible for smaller data sets since no significantly better accuracies were reported. In the current trends, 0.75 accuracy is still a meager approximation of a perfectly functioning network. AlexNets are therefore suitable for larger datasets and show

poor scalability towards smaller data sets. The accuracies fall from 90% over larger data sets to 60%-70% over smaller data sets. Owing to this variation, data augmentation techniques and synthetic techniques required for increasing the size of small data sets seem necessary for the AlexNets. Epoch-wise accuracy shows no significant offsets between consecutive layers and the overall network has no remarkable performance on small data sets.

4) Discussion of the above cases: GoogleNet performs much better than lightweight networks in terms of accuracy spanning over diverse data set sizes. AlexNets failed too early in their architectural implementations, failing to complete training even for the data set size of 96000 with poor performances noted for smaller sizes. As discussed earlier, performance metrics follow trapezoidal patterns wherein we could find occasional peaks and troughs with quantitative extreme numbers ranging from 1 and 3. Though we cannot generalize results to the lightweight networks universally, we can assess the significance of the observations to predict sparse scalability at the minimum. GoogleNets can be used for all computational requirements and their use is feasible with all data sizes and computational hardware available. ResNets and AlexNets are resource-intensive and they require subtle technical implementations to minimize their resource consumption and optimize their training and testing processes. ResNets can be efficiently used over data with some distinguishing features and patterns among them. Conclusively, GoogleNet has 3 parts with AlexNet and ResNet having a part each over a scale of 5 in terms of performance thus enabling the allocation of more scalability for GoogleNet compared to other lightweight networks as demonstrated in table IV.

Factor	GoogleNet	AlexNet	ResNet
Timing	12 Hours	12-18 Hours	24-48 Hours
Accuracy	0.7-0.98	0.5-0.6	0.5
Suitable Size	12000	75000	75000
Scalability Percent	92%	72%	41%
Feasible Range	96000-120000	50000-90000	40000-45000
Optimal Range	96000-100000	80000-85000	40000-40100

TABLE IV
: SCALABILITY OF THE SELECTED MODELS

V. CONCLUSION AND FUTURE WORK

Lightweight Networks are particularly designed for larger and more diverse data sets with a reasonable amount of variations among the data elements. They are also designed to reduce computational complexity and smoothen the learning processes of neural network training. In the current study, we detailed the scalability of neural networks across various sizes

of data sets with a particular focus on smaller data sets. To make this study novel, it is to be noted that We presented our results in the form of case studies, and the methodologies involved are performed in the form of systematic investigations which are performed uniquely. Out of the three networks

considered, GoogleNet performed better with two other networks showing huge similarities in their functioning. This work follows our earlier apriori study on general Machine Learning Models whose results are already presented. The future task of this work is to ensure the synthesis of reliable blueprints and improvise existing architectures to facilitate higher accuracies and optimal computational resource usage.

REFERENCES

[1] Calderon-Ramirez S, Yang S, Moemeni A, Colreavy-Donnelly S, Elizondo DA, Oala L, Rodriguez-Capitan J, Jimenez-Navarro M, Lopez-Rubio E, Molina-Cabello MA. Improving Uncertainty Estimation With Semi-Supervised Deep Learning for COVID-19 Detection Using Chest X-Ray Images. IEEE Access. 2021 Jun 2;9:85442-85454. doi: 10.1109/ACCESS.2021.3085418. PMID: 34812397; PMCID: PMC8545186.

[2] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. 2017. Simple and scalable predictive uncertainty estimation using deep ensembles. In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17). Curran Associates Inc., Red Hook, NY, USA, 6405–6416.

[3] Cui P, Pan H, Li Z, Zhang P, Miao T, Zhou J, Guan H and Xie G. Enabling In-Network Floating-Point Arithmetic for Efficient Computation Offloading. IEEE Transactions on Parallel and Distributed Systems. 10.1109/TPDS.2022.3208425. 33:12. (4918-4934).

[4] Choudhary, K., DeCost, B., Chen, C. et al. Recent advances and applications of deep learning methods in materials science. npj Comput Mater 8, 59 (2022). https://doi.org/10.1038/s41524-022-00734-6

[5] Liu Y, Xing T, Zhou Y, Li N, Ma L, Wen Y, Liu C and Shi H. A Large-Scale Multi-objective Brain Storm Optimization Algorithm Based on Direction Vectors and Variance Analysis. Advances in Swarm Intelligence. (413-424).

[6] F. K. Gustafsson, M. Danelljan and T. B. Schon, "Evaluating Scalable Bayesian Deep Learning Methods for Robust Computer Vision," 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Seattle, WA, USA, 2020, pp. 1289-1298, doi: 10.1109/CVPRW50498.2020.00167.

[7] Chen, C., Zhang, P., Zhang, H., Dai, J., Yi, Y., Zhang, H., Zhang, Y. (2020). Deep Learning on Computational-Resource-Limited Platforms: A Survey. Mobile Information Systems, 2020, 8454327. https://doi.org/10.1155/2020/8454327

[8] Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., Farhan, L. (2021). Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions. Journal of Big Data, 8(1), 53. https://doi.org/10.1186/s40537-021-00444-8

[9] Ahmed, S. F., Alam, M. S. B., Hassan, M., Rozbu, M. R., Ishtiak, T., Rafa, N., Mofijur, M., Shawkat Ali, A. B. M., Gandomi, A. H. (2023). Deep learning modeling techniques: Current progress, applications, advantages, and challenges. Artificial Intelligence Review, 56(11), 13521–13617. https://doi.org/10.1007/s10462-023-10466-8

[10] Tripathi, Diwakar et al. 'BAT Algorithm Based Feature Selection: Application in Credit Scoring'. 1 Jan. 2021 : 5561 – 5570.

[11] Sathwik, K., Padhmanabha, Y., Akanksha, P., Chaitanya, K., Snehaja, P., & Madhu, C.(2023, October 26-27). Apriori Analysis of Deep Learning Models on a Multi-class Data Set [paper presentation]. SSIC 2023, Jaipur, Rajasthan, India