Boundary-Based BenchMark Analysis For Diverse Models & Networks On A Generic Data Set

Sai Sathwik Kosuru[1,2][0009-0008-8057-8799], Perala Venkata Akanksha[1,3], Snehaja Parsi[1,4], Lanke Pallavi[1,5][0000-0002-4335-2300], and Madhu Babu Ch[1,6][0000-0003-1742-6473]

[1] B V Raju Institute of Technology, Narsapur 502313, India
[2] `21211a05r6@bvrit.ac.in`
[3] `21211a05n3@bvrit.ac.in`
[4] `21211a05m1@bvrit.ac.in`
[5] `pallavi503@gmail.com`
[6] `madhubabu6585@gmail.com`

Abstract. Stereotypical Intuitions for ML Research lie in the evolution of newer, simpler, and more innovative architectural designs that succeed in improving learning rates & enhancing the predictability potential of respective models. While the current study revolves around similar concepts, it delves into the architectural & learning pattern differentiators across various models. In this work, we consider a One-fits-all Multiclass data set of 1,20,000 coarse image representations on which we performed multiple rounds of hit-and-trial experiments to assess actual differences among their functioning, performances, and applications. The way the experiments were conducted is relatively novel owing to the involvement of aspects like discrete partitioning of behaviors, functionality dispersion studies, etc. We could observe a significant relevance of our study in ML Research which has 90% more impact than other studies conducted on a similar scale. Our results produced 88%-92% accuracy in the field of differentiating & categorizing the considered models.

Keywords: Benchmark · Lightweight Networks · Machine Learning · One-Fits-All · Differentiators.

1 Introduction

There is no one-on-one relationship between the network and the provided data set. Data Repositories often tend to be diverse in the requirements of the model being designed where they cover every possible test case during the prediction phase. While the data set considered for model training is sufficiently large & is completely not associated with the model, we can consider the depth & architectural arrangements. Each node within the learning graph of a model acts as an intermediate point for disbursal of learn't & input data. A node at a particular level, which in the ML glossary is a layer, constitutes some learning milestone for the whole process. In a conventional network like a Convolutional Neural

Network (CNNs), every node is connected to other nodes, thus constituting a complete graph. The Learning Paradigm in such architecture is purely based on data flow & data path defined purely by a particular model structure which is in its whole, a differentiating theoretical aspect for a model. Data-Model Relationship is a many-many relationship, if not a one-many relationship where it clearly indicates that there is no mandate to fix a particular data set for the model once its training process starts. This is evident in the case of fine-tuning images where we add pre-deployment learning procedures to optimize model functionality & parameters.

Existing implementations particularly focus on the aspect of architectural designs & deployment dependencies for a considered model. Though Machine Learning Operations(MLOps) have been rarely discussed about in the research, Industrial Researchers/ experts often provide details into such aspects. Reinforcement Learning Through Human FeedBack(RLHF) which we can see in the applications of Large Language Models(LLMs) is a clear manifestation of ever-expanding deployment landscapes for ML Models. While all of these parameters are of little concern for the current study, the relevance of them lies in the consideration of Lightweight Networks like GoogleNet, AlexNet, etc., which have relatively less computational load & resource requirements. Benchmark Analysis is often limited to the evaluation of real-world products rather than ML Models in earlier studies but as evident in the consideration for mobile-based ML/DL training/deployments, we need to consider the necessary benchmarks to include in such studies requiring detailed insights.

The current Boundary-Based Benchmark Analysis involves the aspects of model functionality, predictability potential, architectural differences, and application-oriented observations for different models & networks. This work takes a different approach far from the conventional approach of directly evaluating by coding & testing where it involves systematic concerted manual-cum-computational efforts to gain insights into the diversity of models & networks. The findings serve as contributions for

- Designing Model Pipelines for future networks to differentiate their learning approach toward diverse data sets.
- Although the study doesn't explicitly contribute any new data set, we give necessary prospects for selecting a suitable data repository for any further studies.
- We propose novel pointers called benchmarks which can be flexibly used for marking different milestones in ML Model Synthesis which is helpful for Developers & Scientists.
- The results can be used for the enhancement of the considered models & networks by substantially decreasing the commonality of similar properties among them.

2 Literature Survey

2.1 Background

Existential Scenarios for BenchMark Considerations for ML Practical Applications include a scenario where 25 ML techniques & OLS(Ordinary Least Square) regression was applied in U.S yearly inflation metrics calculated a year ahead [1]. In this work, we can observe the significant relevance of such benchmarking processes in ML over the rest of the techniques. To further justify this relevance of benchmarking applications in the existing systems over inter-domain applicability of ML models, we can observe domains such as Functional Near-Infrared Spectroscopy [2], Outlier Detection in process data sets more relevant to our one-fits-all data repository approach [3], Tabular Regression calculations [4], etc., that involve benchmark considerations for evaluating their performances in their applications. OpenML-CTR23 suite, which is available for disposal on OpenML, with 35 regression problems is an additional manifestation of benchmarking procedures over data collection processes. [4]. The ConceptARC benchmark, the other interdisciplinary benchmarking theory earmarked for distinguishing ML applications, is focused on concept abstraction from where logical reasoning follows. This highlights the relevance of pattern learning paradigms of various AI systems [5].

One application that requires thorough data quality checking & complex data pre-processing procedures is the Remote Sensing Technology. BenchCloudVision [6] is a benchmark analysis that is focused on this technology and aims to decipher the image spectra logics for training a model. Similarly, Image Dehazing which is useful for enhancing image quality uses Lightweight Networks such as PriorNet [7] where benchmarks are set on the models' capability to provide valid insights. If we consider the quantitative aspects of the earlier studies, we can find a minimum of 21 data sets, 32 SOTA(State-Of-The-Art) techniques, 3 modules, & 8 tasks which is evident in the DANCE model that is applied for genetic disorder analysis [8]. Code Base Summarization, Searching, Completion, and Type Inference are the tasks that are often benchmarked over leading models when we consider benchmark analysis of deep learning source codes, which are all exemplified with the introduction of Novel Large Language Models [9]. Benchmarking on progressive training toward desired learning functions [10], Multi-Modal Training considering complex vision technologies, text, sound, etc. [11], Extension of Deep Learning Approaches toward Network-enabled Human Sensor Technologies [12], Reference Metrics for Vision & Perception [13], Complex Biological Specimen Analyses like in-vitrio fertilization [14], pixel-based modeling such as next frame prediction [15], etc.

2.2 Comparisons: Existing Studies

The Comparisons with the existing studies and their corresponding impact of improvement are shown in the table 1.
Distinction 1 One-Fits-All Data Set Collection is not a simple task since the

requirement is for a universal data representation & possible repository location for such elements. Earlier studies considered domain-specific data sets such as unmanned marine vessels [16], social media-based disaster-focused image repositories [17], Generative Deepfake oriented data set [18], biology-related massive antibody sources [19], UAV captured human pose image collections [20], etc.
Impact Certain revelations of studies such as lightweight deep reinforcement learning for flow control [21], low availability of data sets for food benchmarks [22], etc., suggest 99.8% improvement over standard ML Projects if there is a presence of universal data sets. Although we are not contributing any specific data set here, we considered how a universal data set can be chosen for the current study.

Distinction 2 Lightweight Network Benchmarking is still an untouched topic with rare research explorations such as for Enhanced YOLO8n for Cloth Image Detection [23], YOLOV5 for Vehicle Detection [24], Determination of Scratch Based LightWeight Network Synthesis [25], etc. Here, we propose a boundary-based benchmarking mechanism that discretely partitions lightweight networks based on their observed functionality & performance.
Impact Li et al [26], in their work mention the transformative impact the lightweight networks bring with them in the data-driven landscape. While Methodologies like Architectural Distillation [27] are being laid towards developing networks that are light both in development & deployment, the current work rightly relays its approach towards distinctly partitioning them using benchmark metrics marking 30% increase in existing research of the field.

Distinction 3 Benchmarking is majorly regarded as a contribution towards quality improvement [28], industry-level comparisons & required improvements [29], quantum machine learning before availability of noiseless hardware [30], etc. We take an innovative form of benchmarking mechanism which we specifically use for identifying apriori differences among the set of considered lightweight networks by us.
Impact Unique ways/attributes for studying lightweight networks in different ways significantly improve the ever-changing data domain. We can observe tight dominance competition between Large Language Models(LLM) & Computer Vision(CV) in recent days [31]. Advanced versions of partitioning these models still exist where language-vision models are being developed [32], hence highlighting the need for different studies to properly & systematically partition models' behaviors & performances.

3 Methodology

3.1 Experimental Benchmarks

Discrete Phases have been systematically designed for 6 machine learning networks namely GoogleNet, AlexNet, ResNet, Convolutional Neural Network(CNN),

Table 1: Existing Systems' comparison with the current study

Current Study	**Existing Implementation**	**Implication**	**Impact**
Generic Repository	Problem-Specific Repository	Generalization of Models' Benchmarking Mechanism	20%
Training Data Set Type			
Interdisciplinary	Domain-Specific	Vast Validity for the Current Study	35%
Considers Lightweight Networks	Mostly dealt with Conventional Models	Study in Line With Latest Innovations	55%
Considered Model Type			
Functionality& Performance-Based Partitions	No Partition BenchMarks Available	Paves Way For Further Model Designs	82%
Assess Apriori Differences	Assess Quality& Performance	Relation between Theoretical Description & Practical Implementation is assessed	75%
Study Type			
Contribution to Theoretical Comparisons	Contribution to Quality Improvement	Study Near To Theoretical Aspects	15%

Feedforward Neural Network(FNN), and Autoencoders. We have considered performing 3 benchmarking experiments with each outcome aimed toward a numerical calculation as its summative milestone.

Experiment 1: ***Timing Metrics*** Post-Deployment Statistics and Comprehensive Analysis of Timings pose complex calculations that do not produce one-shot significant numerical output. The major part of the Model Development Process is covered with the learning phase, which is different from the training phase, which deals with error rates & calculations. We focus on the time complexities of architectural parameters & hyperparameters used for training, particularly during each epoch. The query timing function is tabulated in

Equation 1 derived into Equation 2 derivable into Equation 3. Hyperparameter tuning is generally taken as a higher precedence over other architectural tuning methods with successive differentiations from equation to equation.

$$\text{time-factor} = \text{f}(\text{architecture*}, \text{features}, \text{bias}) \tag{1}$$

$$\text{code-time} = \frac{\text{d}}{\text{dx}}(\text{f}) * \text{calc. benchmark} + \frac{\text{Hyperparameter Tuning}}{\text{Architectural Pattern}} \tag{2}$$

$$\text{Layer-time} = \frac{\text{d}}{\text{dx}}(\text{code time}) * \text{calc. benchmark} + \frac{\text{Hyperparameter Tuning}}{\text{Special Architecture}} \tag{3}$$

Experiment 2: *Epoch Metrics* Starting from the minimum possible epoch count of 10, we started incrementing benchmark epoch counts incrementally with a corresponding recording of their accuracy, performance metrics & loss computations which resulted in the equation 4 that is further expanded as equations 5 & 6 where P(x) is Performance Function. All the involved benchmarks are specific to a project's requirements.

$$\text{Epoch-factor} = \text{f}(\text{Batch Size}, \text{Learning Rate}, \text{Progression Benchmark}) \tag{4}$$

$$\text{Progression-Benchmark} = \left[\oint_{\text{Epoch}} \text{P}(\text{architecture}) d(\text{architecture})\right] + \text{Accuracy-Benchmark} \tag{5}$$

$$\text{g(layer,epoch)} = \frac{\text{epoch} * \text{batch-size}}{\text{Architectural-Parameter}} + \text{Epoch-BenchMark} \tag{6}$$

Experiment 3: *Train & Test Metrics* Training involves the generation of latent representations of the given data collection, synthesizing patterns out of it on an experience basis which ultimately leads to the production of a model that is subjected to varied testing phases. While this is the most common conventional approach, it is legitimate to evaluate the benchmark-based procedures that result in insights into the whole process, as demonstrated in the equation 7 which is derived as the equation 8 and further into the equation 9.

$$\text{Train-factor} = \text{f}(\text{Training Benchmark}, \text{Recursion Benchmark}, \text{Test Benchmark}) \tag{7}$$

$$\text{Training-Benchmark} = \text{Architectural-Benchmark} * \text{Accuracy} \tag{8}$$

$$\text{Test-Benchmark} = \text{g}(\text{Accuracy}, \text{Hyperparameter Benchmark}) \tag{9}$$

3.2 Test-Based Benchmarks

Test 1: *Architectural Test* We discuss the architectural benchmarks for testing 6 discrete Neural Networks & ML Models in this study where we have considered the addition of inception & dropout layers to consider the universal scale of the current study parameters as demonstrated in the figure 1.

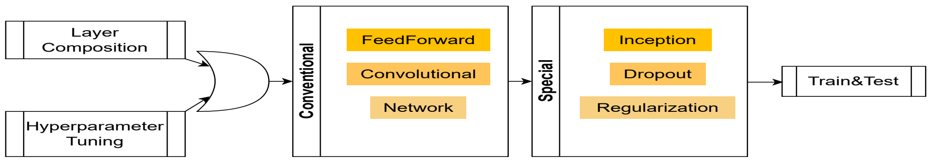

Fig. 1: Process Flow in Architectural Test

Test 2: *Training Test* With the precursor being Architectural Tuning, we add an extra dimension for benchmark testing, Hyperparameter Tuning, which is used as a pretext for improving model performance. Consecutive Tests were performed to evaluate trade-offs between the requirements for Hyperparameter Tuning & Learning Rate Calculations based on the optimality of the results. The process is demonstrated in the figure 2

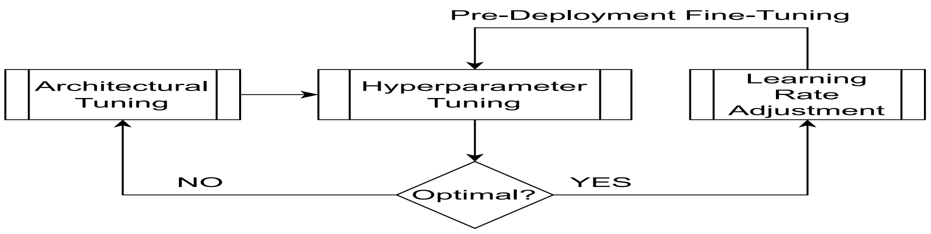

Fig. 2: Process Flow in The Training Test

3.3 Performance Benchmarks

The Performance is benchmarked on the grounds of Scalability, Accuracy, & Test Success.
As Reflected in the Table 2

Scalability considers Data Repository sizes which are modeled based on Models and Standards with empirical function of size&accuracy where the auxiliary factors used are optimality, feasibility, prediction capabilities& Fine-Tuning Procedures to improve performance.

Table 2: Considered Performance BenchMarks across Parameters(Param.)

Scalability	**Param.**	**Accuracy**	**Param.**	**Success**	**Param.**
Data Set Size	Model	Optimal	Prediction	Positives	Classification
Empirical Function	f(size, accuracy)	Threshold	Required	Deployment	Statistics
Optimality	Feasibility	Epoch	Accuracy	Rate	Percentage
Diversity	Score			Dynamics	Function
Prediction	Capability	Error	Rate	Labels	Association
Test Set Size	Standard	Readiness	Deployment	Precision	Rate
Fine Tuning	Performance	Generalization	Diversity		

Accuracy A threshold is set based on requirements with epoch accuracy considerations& preferences for optimal predictions along with associated error-rate relations, deployment-ready characteristics, and Diversity Generalizations.

Success Rate The constituent parameters are Classification positives, Deployment Statistics, Rate in percentage, Function-Modelled Dynamics, Labelled Associations & Precision rates.

4 Results And Discussions

4.1 BenchMark-Network[7] Classification

Timing Records GoogleNet & FeedForward Neural Network takes 5-6 hours of training time & 0.5-1 hours of testing time. We could observe significant upthrust to 10-12 hour training time & 2-4 hour testing time for complex networks like ResNet, AlexNet, and Convolutional Neural Network(though conventional & simple). The timings are detailed in the table 3.

Epoch Trends GoogleNet & Autoencoders consist of majorly stagnant epoch accuracy rates with AlexNet, FeedForward Neural Network, Convolutional Neural Networks, & ResNets forming skewed epoch accuracy distributions. In the skewed distributed networks, we can find negative origins for Convolutional Neural Network, FeedForward Neural Network, & ResNet with opposite trends for the others, all of which are demonstrated in the figure 3.

[7] Note: Terms **Network** & **Model** are often used interchangeably.

Table 3: Obtained Timing Records-Training(Tr.)&Testing(Ts.) Timings

Network	**Trial 1**		**Trial 2**		**Trial 3**		**Trial 4**		**Trial 5**		**Trial 6**	
	Trn.	Tst.	Trn.	Tst.	Trn.	Tst.	Trn.	Tst.	Trn.	Tst.	Trn.	Tst.
GoogleNet	5hr	1.2hr	6.4hr	1hr	8.2hr	0.2hr	6.4hr	0.5hr	7.8hr	0.9hr	8.2hr	0.7hr
AlexNet	8hr	3.2hr	8.1hr	3.8hr	9.2hr	3.5hr	9.2hr	3.7hr	8.5hr	3.6hr	8.7hr	2.1hr
ResNet	5hr	1.2hr	4.2hr	0.5hr	5.6hr	0.8hr	5.1hr	1.5hr	5hr	1.2hr	5hr	1.0hr
CNN	10hr	2hr	8hr	1hr	10hr	1.2hr	9hr	1.1hr	8hr	3.5hr	10hr	2hr
FNN	3hr	0.2hr	4hr	0.1hr	3hr	0.6hr	3.2hr	0.8hr	3.1hr	0.5hr	3.0hr	0.2hr
AutoEncoder	7hr	2hr	7hr	2hr	7hr	1hr	3hr	2hr	9hr	3.2hr	4hr	3.1hr

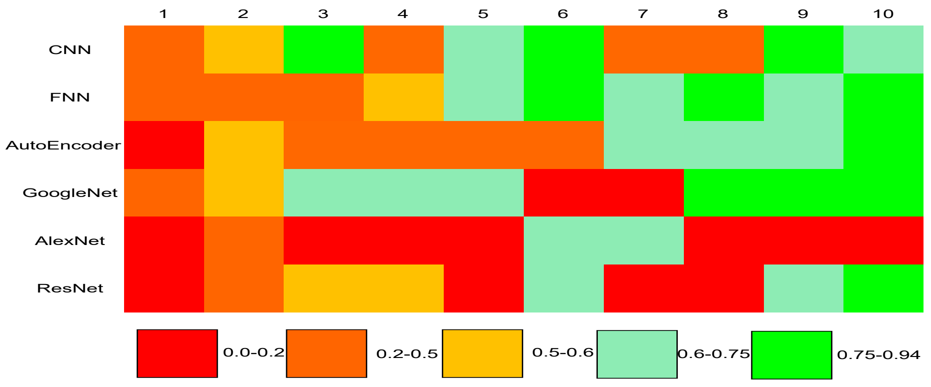

Fig. 3: Epoch Trends Along Networks

Train & Test Observations Conventional Models like CNN, FNN, etc., suffer from early over fittings, thus necessitating multiple dropout layers in their outermost layers. Although there is no observed overfitting characteristic in GoogleNet, it is most likely to necessarily introduce a minimum of 5-10 optional dropout layers in the inception layers. For other lightweight Networks, ResNet requires the mandatory addition of 3-5 dropout layers in each & every conventional model layer because of the complex nature of the block structure of the model. The same applies to AlexNet and we could observe a special behavior for autoencoders with less probability of overfitting. Moreover, it is not feasible to add dropouts to smaller autoencoder models when considering smaller data sets. We can take the number of iterations as specified in the table 4 to obtain the above findings.

Table 4: Train&Test Observations on Error, Learning Rate(Learn.), Hyperparameter Tuning(Hyp.)& Deployment(Depl.)

Network	**Train Iteration For**				**Test Iteration For**			
	Error	Learn.	Hyp.	Depl.	Error	Learn.	Hyp.	Depl.
CNN	3	8	2	1	2	1	1	1
FNN	2	20	42	10	10	10	5	5
AutoEncoders	10	20	0	0	0	0	5	10
GoogleNet	0	20	10	1	0	0	0	10
AlexNet	30	15	0	0	0	0	0	10
ResNet	5	5	15	15	1	1	0	0

4.2 BenchMark Fixtures Across Networks

Architectural Test Results Conventional Layers add to the functioning of every neural network considered during the study where special layers like dropout prevent model particularization. GoogleNet's inception networks are generally numbered 3-4 for an accuracy of 0.5, and 5-6 for an accuracy above 0.6. The number of Residual Blocks in a good ResNet model is benchmarked at 5/6 based on accuracy requirements with the minimum accuracy threshold at 0.56. The Architectures of CNN, FNN, and Autoencoders have peak layer count of 6-10. The relevance for benchmarks on the addition of dropout layers is available in the table 5.

Table 5: Dropout Layers(m-mandatory,o-optional) Along Networks

CNN	FNN	AutoEncoder	GoogleNet	AlexNet	ResNet
2	0	5	10	2	5
m	o	m	m	o	m

Training Test Results We can observe from the graphs in figure 4 that the predictable training sequences for CNN, FNN, & ResNet(with some exceptions) with a non-conventional learning curve observed for ResNet where there are 0 learning peaks which is unusual. AlexNet & GoogleNet produced a horizontal linear learning curve which indicates early overfitting & a lack of generalization. Hyperparameter tuning is required for GoogleNet, CNN, FNN, & Autoencoder where pre-deployment fine-tuning is mandatory for GoogleNet. Architectural Tuning is mostly found to be irrelevant for many of the models considered.

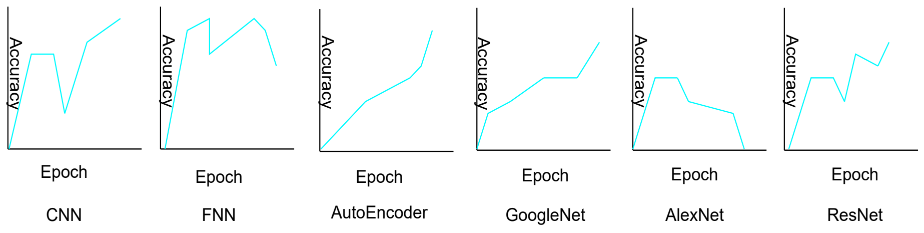

Fig. 4: Training Trends Along Networks

4.3 Cases for Performance BenchMarks

Data Set Size is available with a feasible range of 75000-79000 for conventional models, 1,10,000-1,20,000 for lightweight networks, and an optimal range of 75000-75100 and 1,15,000-1,20,000. The accuracy Threshold for a conventional model is 0.52 & a lightweight network is 0.825 where epoch accuracies generally vary between 0.521 & 0.988 for all the models considered. The test set size used for the study is 35000 which is found to be sufficient for all the models considered while fine-tuning the repository of 10 images is considered for improving complex models like Autoencoders, ResNets, and GoogleNet. The results are tabulated in the table 6

4.4 Benchmark Partitions: Discussion

Heavy-Weight Network Partitions A Functional Benchmark Partition can be found between CNN, & FNN as one block & Autoencoders as the other. Though Autoencoders are majorly used as constituents of advanced models like Artificial General Intelligence(AGI) models, they can be considered for benchmark partition because their behavior reflects most of the other heavyweight model functionalities.

Lightweight Network Partitions GoogleNet has a unique benchmark specific to it with innovative variations in special layer composition like inception architecture variations. The Dropout Layer mandate adds another partition benchmark of overfitting probability between GoogleNet, & AlexNet as one set and

Table 6: Performance Benchmarks on Diversity(Div.), Accuracy& Success Rate

Network	**Scalability**		**Accuracy**		**Success**	
	Size	Div.(Classes)	Accuracy	Error	Rate(%)	Score(/10)
CNN	35000	5	0.75	34%	92%	8
FNN	25000	4	0.62	72%	96.8%	6
Autoencoder	75000	8	0.52	23%	98%	9.1
GoogleNet	1,20,000	10	0.92	12%	99.2%	9.8
AlexNet	1,10,000	10	0.32	85%	65%	5.2
ResNet	1,10,000	10	0.65	15%	85%	7.2

ResNet as the other. Architectural Tuning differentiates as another benchmark with GoogleNet, AlexNet, and ResNet as distinct groups due to different individual properties. GoogleNet is the only one in the set benchmarked for fine-tuning required cases and invalid learning rate cases.

Generic Network Partitions The line of boundary is insignificant and highly narrow between heavyweight & lightweight networks except for ResNets with specialized architectures in the form of residual blocks. Lightweight Networks are generally more efficient than other networks but the above observations suggest narrowing gaps between both types with the substantial increase in the number of Architectural Layers of Heavyweight Networks. Accuracy Thresholds are significantly high for lightweight networks with a significant rise in performance benchmark requirements where epoch accuracies are also on the positive side while the positive benchmark for a conventional model lies in initial epochs, classification labels, & true prediction rates. The partitions have been visually demonstrated in the figure 5.

5 Conclusion And Future Work

The study can be inferred as a comprehensive & abstract evaluation of all the 6 models-GoogleNet, AlexNet, ResNet, CNN, FNN, and Autoencoders which is performed on a 10-class image data set of 1,20,000 diversely coarse pixellated representations. While the provided benchmarks are excessively numerical, it is equally crucial to determine theoretic variations to partition model behaviors. 1-5 Classes are suitable for training conventional models like CNN, FNN, and Autoencoders while Lightweight Networks like GoogleNet, ResNet, and AlexNet require a minimum class count of 10. Out of all the models considered, GoogleNet is found to be the most vulnerable to overfitting, AlexNet & ResNet the most vulnerable to Error & Training Failure while Conventional Models like CNN,

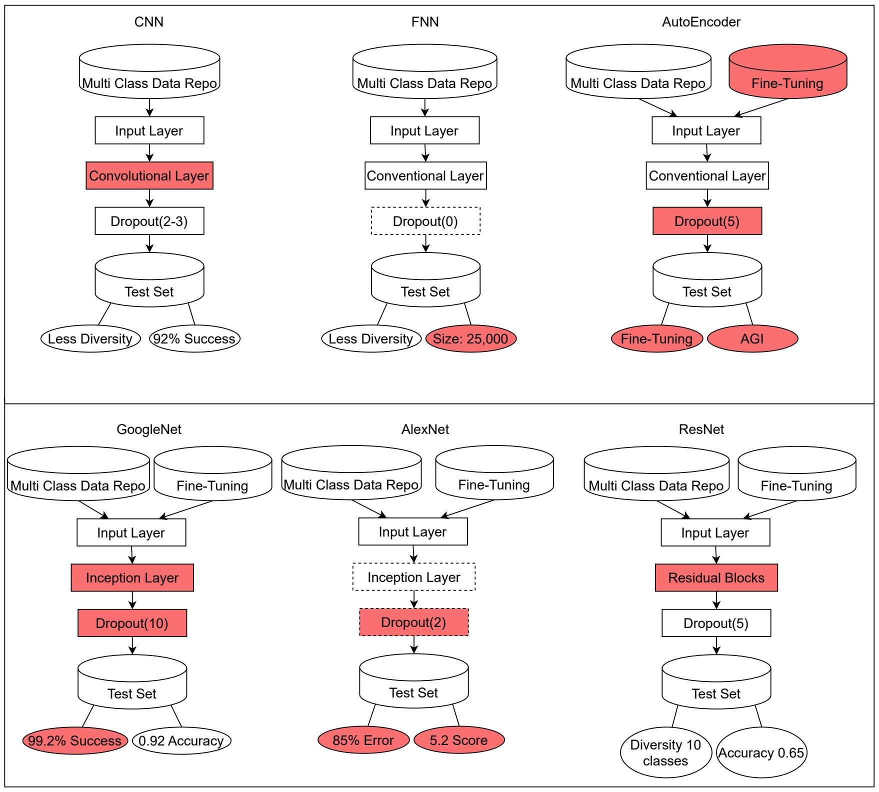

Fig. 5: Partitions With The BenchMarkable Features Highlighted

FNN, & Autoencoders are the most vulnerable to underfitting. The future directions in the current work are to analyze architectural benchmarks in detail & expand the current study to more networks & models in line with continuously evolving data-driven landscape.

6 Declarations

Data Availability: Data Will Be Made Available Soon On Request. Currently, It is Being Processed.

Funding The authors did not receive support from any organization for the submitted work.

Conflict of Interest: The authors declare that they have no conflict of interest.

References

1. Malladi, R.K. Benchmark Analysis of Machine Learning Methods to Forecast the U.S. Annual Inflation Rate During a High-Decile Inflation Period. Comput Econ (2023). https://doi.org/10.1007/s10614-023-10436-w
2. Benerradi J, Clos J, Landowska A, Valstar MF and Wilson ML (2023) Benchmarking framework for machine learning classification from fNIRS data. Front. Neurosurgeon. 4:994969. doi: 10.3389/fnrgo.2023.994969
3. Schindler TF, Schlicht S, Thoben K-D. Towards Benchmarking for Evaluating Machine Learning Methods in Detecting Outliers in Process Datasets. Computers. 2023; 12(12):253. https://doi.org/10.3390/computers12120253
4. Fischer, S. F., Feurer, M., & Bischl, B. (2023). OpenML-CTR23 – A curated tabular regression benchmarking suite. In AutoML Conference 2023 (Workshop). Retrieved from https://openreview.net/forum?id=HebAOoMm94
5. Moskvichev, A., Odouard, V. V., & Mitchell, M. (2023). The ConceptARC Benchmark: Evaluating Understanding and Generalization in the ARC Domain. arXiv preprint arXiv:2305.07141.
6. Fabio, L., Piga, D., Michelucci, U., & El Ghazouali, S. (2024). BenchCloudVision: A Benchmark Analysis of Deep Learning Approaches for Cloud Detection and Segmentation in Remote Sensing Imagery. arXiv preprint arXiv:2402.13918.
7. Chen, Y., Wen, Z., Wang, C., Gong, L., & Yi, Z. (2024). PriorNet: A Novel Lightweight Network with Multidimensional Interactive Attention for Efficient Image Dehazing. arXiv preprint arXiv:2404.15638.
8. Ding, J., Liu, R., Wen, H. et al. DANCE: a deep learning library and benchmark platform for single-cell analysis. Genome Biol 25, 72 (2024). https://doi.org/10.1186/s13059-024-03211-z
9. Wan, Y., He, Y., Bi, Z., Zhang, J., Zhang, H., Sui, Y., Xu, G., Jin, H., & Yu, P. S. (2023). Deep Learning for Code Intelligence: Survey, Benchmark and Toolkit. arXiv preprint arXiv:2401.00288.
10. Ahmad, R., Alsmadi, I. & Al-Ramahi, M. Optimization of deep learning models: benchmark and analysis. Adv. in Comp. Int. 3, 7 (2023). https://doi.org/10.1007/s43674-023-00055-1
11. Chai W, Wang G. Deep Vision Multimodal Learning: Methodology, Benchmark, and Trend. Applied Sciences. 2022; 12(13):6588. https://doi.org/10.3390/app12136588
12. Yang, J., Chen, X., Wang, D., Zou, H., Lu, C. X., Sun, S., & Xie, L. (2023). SenseFi: A Library and Benchmark on Deep-Learning-Empowered WiFi Human Sensing. arXiv preprint arXiv:2207.07859.
13. O. Soufi, Z. Aarab and F. -Z. Belouadha, "Benchmark of deep learning models for single image super-resolution (SISR)," 2022 2nd International Conference on Innovative Research in Applied Science, Engineering and Technology (IRASET), Meknes, Morocco, 2022, pp. 1-8, doi: 10.1109/IRASET52964.2022.9738274.
14. Kromp, F., Wagner, R., Balaban, B. et al. An annotated human blastocyst dataset to benchmark deep learning architectures for in vitro fertilization. Sci Data 10, 271 (2023). https://doi.org/10.1038/s41597-023-02182-3
15. Zhou, Yufan & Dong, Haiwei & El Saddik, Abdulmotaleb. (2020). Deep Learning in Next-Frame Prediction: A Benchmark Review. IEEE Access. PP. 1-1. 10.1109/ACCESS.2020.2987281.

16. Ning Wang, Yuanyuan Wang, Yi Wei, Bing Han, Yuan Feng, "Marine vessel detection dataset and benchmark for unmanned surface vehicles," Applied Ocean Research, Volume 142, 2024, Article 103835, ISSN 0141-1187, https://doi.org/10.1016/j.apor.2023.103835.
17. F. Alam, F. Ofli, M. Imran, T. Alam and U. Qazi, "Deep Learning Benchmarks and Datasets for Social Media Image Classification for Disaster Response," 2020 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), The Hague, Netherlands, 2020, pp. 151-158, doi: 10.1109/ASONAM49781.2020.9381294.
18. Pei, G., Zhang, J., Hu, M., Zhang, Z., Wang, C., Wu, Y., Zhai, G., Yang, J., Shen, C., & Tao, D. (2024). Deepfake Generation and Detection: A Benchmark and Survey. arXiv preprint arXiv:2403.17881.
19. Ruffolo, J.A., Chu, LS., Mahajan, S.P. et al. Fast, accurate antibody structure prediction from deep learning on massive set of natural antibodies. Nat Commun 14, 2389 (2023). https://doi.org/10.1038/s41467-023-38063-x
20. Kalampokas, T., Krinidis, S., Chatzis, V. et al. Performance benchmark of deep learning human pose estimation for UAVs. Machine Vision and Applications 34, 97 (2023). https://doi.org/10.1007/s00138-023-01448-5
21. Viquerat, J., Meliga, P., Jeken, P., & Hachem, E. (2024). Beacon, a lightweight deep reinforcement learning benchmark library for flow control. arXiv preprint arXiv:2402.17402.
22. Mohanty SP, Singhal G, Scuccimarra EA, Kebaili D, Héritier H, Boulanger V and Salathé M (2022) The Food Recognition Benchmark: Using Deep Learning to Recognize Food in Images. Front. Nutr. 9:875143. doi: 10.3389/fnut.2022.875143
23. Gong, Haowei and Liao, Haibin and Ai, Zhe and Deng, Yizhou and Zhe, Wei, Lightweight Network Based on Improved Yolov8n for Clothing Image Detection. Available at SSRN: https://ssrn.com/abstract=4740499 or http://dx.doi.org/10.2139/ssrn.4740499
24. Xudong Dong, Shuai Yan, Chaoqun Duan, "A lightweight vehicles detection network model based on YOLOv5," Engineering Applications of Artificial Intelligence, Volume 113, 2022, Article 104914, ISSN 0952-1976, https://doi.org/10.1016/j.engappai.2022.104914.
25. Li, H., Yue, X., Zhao, C. et al. Lightweight deep neural network from scratch. Appl Intell 53, 18868–18886 (2023). https://doi.org/10.1007/s10489-022-04394-3
26. Y. Li, J. Liu and L. Wang, "Lightweight Network Research Based on Deep Learning: A Review," 2018 37th Chinese Control Conference (CCC), Wuhan, China, 2018, pp. 9021-9026, doi: 10.23919/ChiCC.2018.8483963.
27. Ting-Bing Xu, Peipei Yang, Xu-Yao Zhang, Cheng-Lin Liu, "LightweightNet: Toward fast and lightweight convolutional neural networks via architecture distillation," Pattern Recognition, Volume 88, 2019, Pages 272-284, ISSN 0031-3203, https://doi.org/10.1016/j.patcog.2018.10.029.
28. Willmington C, Belardi P, Murante AM, Vainieri M. The contribution of benchmarking to quality improvement in healthcare. A systematic literature review. BMC Health Serv Res. 2022;22(1):139. Published 2022 Feb 2. doi:10.1186/s12913-022-07467-8
29. Dattakumar, R. & Rajashekharaiah, Jagadeesh. (2003). A review of literature on benchmarking. Benchmarking: An International Journal. 10. 176-209. 10.1108/14635770310477744.
30. Bowles, J., Ahmed, S., & Schuld, M. (2024). Better than classical? The subtle art of benchmarking quantum machine learning models. arXiv preprint arXiv:2403.07059.

31. H. Chen, "Comparison of Large Language And Vision Models on Representative Downstream Tasks," 2023 International Conference on Image Processing, Computer Vision and Machine Learning (ICICML), Chengdu, China, 2023, pp. 307-311, doi: 10.1109/ICICML60161.2023.10424913.
32. Ghosh, A., Acharya, A., Saha, S., Jain, V., & Chadha, A. (2024). Exploring the Frontier of Vision-Language Models: A Survey of Current Methodologies and Future Directions. arXiv preprint arXiv:2404.07214.