Image Classification using ImageNet Classifiers in Environments with Limited Data

Anirvin Sharma
Amity School of Engineering & Technology, Noida UP 201301, India
Abhinav Singh
Amity School of Engineering & Technology, Noida UP 201301, India
Tanupriya Choudhury (tanupriya1986@gmail.com)
University of Petroleum & Energy Studies, Dehradun, India
Tanmay Sarkar (tanmays468@gmail.com)
Malda Polytechnic, West Bengal State Council of Technical Education, Govt. of West Bengal, Malda-732 102, India

Research Article

Keywords: Image Classification, Transfer Learning, Convolutional Neural Networks, Computer Vision

Posted Date: April 16th, 2021

DOI: https://doi.org/10.21203/rs.3.rs-428416/v1

Abstract

In this research, we compare and contrast various image classification algorithms and how effective they are in specific problem sets where data might be scarce such as prediction of rare phenomena (for example, natural calamities), enterprise solutions etc. We have employed various state-of-the-art algorithms in this study credited to have been some of the best classifiers at the time of their inception. These classifiers have also been suspected to fall prey to overfitting on the datasets they were initially tested on viz. ImageNet and Common Objects in Context (COCO); we test to what extent these classifiers tend to generalize to the new data provided by us in a transfer learning framework. We utilize transfer learning on the ImageNet classifiers to adapt to our smaller dataset and examine various techniques such as data augmentation, batch normalization, dropout etc. to mitigate overfitting. All the classifiers follow a standard fully connected architecture. The end result should provide the reader with an overall analysis of which algorithm or approach to use in conditions where data might be limited while also giving a brief overview of the progress of image classification algorithms since their advent. We also provide an analysis on the effectiveness of data augmentation in limited datasets by providing results achieved with and without utilizing data augmentation. In our case, we found the MobileNet (with its lightweight nature contributing to low computational costs) and InceptionV3 (owing to its lower training time) to be the best performing classifiers for applying transfer learning to limited datasets out of the classifiers we have used for our study. This paper aims to establish preemptive standards that can be used to evaluate the models which can be used in object recognition, and image classification for problems containing limited amounts of data.

1 Introduction

Image Classification has always been one of the most widely used application of Computer Vision. There is no “perfect” way to classify images, just better ones. With increasing research and implementation in this area, we perennially see new approaches evolving, taking inspiration out of older methods and transforming them into neoteric standards. However, most of these aforementioned studies focus on classification of humongous datasets while establishing themselves as some of the best classifiers.

We found that there is a severe lack of reference and studies when it comes to the performance of these classifiers when generalizing to a dataset which has limited data for learning (or adaptation, in the case of Transfer Learning).

The aim of this research is to answer the following questions at the onset of this study:-

1. The ILSVRC (ImageNet Large Scale Visual Recognition Challenge) entries have been accused of falling prey to overfitting on the dataset. A concept which seems far-fetched at the get go, considering the enormous size of the ImageNet dataset, but something that has made its way around the research circle causing speculation and giving rise to multiple studies and experiments including one that tested how well the ImageNet classifiers would generalize to a dataset created to

closely mimic ImageNet [1] which proves that the accuracy results are not replicated in a dataset created in a similar fashion to ImageNet; However, as Recht et al. 2019 [1] observe and document, the models which performed better on ImageNet than their predecessors, performed even better on this similar dataset as compared to the same predecessors, hence, cementing the performance of these classifiers relative to each other. We aim to test this notion in our study on a low-scale dataset and set parameters by using these classifiers in our study to not only test the top-1 accuracy in absolute terms, but also, the relative terms.

2. The ImageNet is a humongous dataset with a whopping 14 million images encapsulated within itself [2]. The results of the ILSVRC are geared towards providing solutions that specifically would only cater to industrial or academic problems in which access to large amounts of data is available and the only pertinent issue is utilizing all this data to form accurate predictions for classification and detection.

   However, this doesn't account for situations where only a scarce amount of data might be available for you to work with; these situations can also frequently occur in the industry and might leave some of the most prepared organizations with their big data suites and weathered data scientists in a rut. These situations can range from enterprise solutions to time series to aggregating modelling to prediction of rare phenomena.

   Through our study, we propose to view how well these state-of-the-art classifiers would function exclusively in a custom environment for situations where data might be limited. The main thing to note would be whether the same algorithms would function with better accuracy and speed as compared to their conventional counterparts when considering a smaller dataset for different utility.

We have divided the paper in 5 segments. Section 2 covers the background of our research in the form of a literature survey. Section 3 delineates the methodology applied by us while preparing our study. Section 4 reports the results, observations, and inferences that we generate and, thereof, present after completing our research. Section 5 and Sect. 6 close with the Conclusion and Future Scope respectively.

2. Background & Related Work

2.1 The Re-Rise of CNNs

When you talk about computer vision and image classification, ImageNet is a name which is bound to make an appearance. ImageNet is a large-scale, state-of-the-art dataset full of human-annotated images (courtesy of The Amazon Mechanical Turk).

ImageNet is used worldwide by leading researchers, academicians, and students to test their algorithmic modifications on a large scale dataset. The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) was a platform provided to researchers from 2010 to 2017, aimed at comparing and presenting the best

results achieved by the highest performing algorithms on the ImageNet dataset. The challenge was to achieve the highest efficiency on the ImageNet database, an exorbitantly humongous dataset which had a comprehensively categorized set of images; it had categorized many entities to an unprecedented depth, example given, 50 breeds of dogs!

This challenge has given birth to a lot of algorithms and classifiers that have been glorified in the recent history of Image Classification and Object Detection. We also know from various studies that deep learning and, in particular, CNNs perform significantly better than other machine learning techniques such as SVM etc. [3].

Wiatowski et al., 20118 [4] made waves in the sphere of image classification through the creation of AlexNet, a convolutional neural network that blew the entire competition in ILVRSC out of the water (15.3% top-5 classification error as compared to 26.1% of the closest competitor). They managed to circumvent overfitting and the humongous training time by utilising dropout [5] and non-saturating nonlinearities by virtue of Rectified Linear Unit viz. ReLU [4] respectively. This led to a widespread resurgence of CNNs as more individuals realised how the power of these neural nets allowed the creators to clinch the highly-coveted top spot in the 2012 edition of the prestigious ILSVRC.

Only a couple of years later, the VGG16 and VGG19 classifiers (where the numeric stood for the number of layers each network possessed), named after the Visual Geometry Group of Oxford University where it was incepted, placed second in the classification track of the ILSVRC 2014 (achieving a top-5 test error of just 7.32% becoming one of the first networks to go below the 10% mark) and first in the localisation track. One of the major changes that Karen Simonyan & Andrew Zisserman implemented in their convolutional network was the inclusion of multiple smaller filters to cover the same amount of area as done by a large filter; this change helped not only introduce increased non-linearity but also helped decrease the number of parameters which, in turn, helped reduce over-fitting and reduce the time it would take for the model to converge.

In the very same year, Szegedy et al. designed a convolutional neural network, named GoogLeNet, that deviated from the basic architecture that could be found in CNNs at that time to significantly decrease the number of parameters while vastly increasing the number of layers. The network, codenamed Inception (owing to the authors' cinephilic tendencies), incorporated 22 layers and closely beat VGG in the ImageNet Classification Challenge with a top-5 classification error of just 6.67%. Taking inspiration from the Network-in-Network approach [6], GoogLeNet incorporated a lot of 1x1 convolutions to overcome computational bottlenecks via dimensionality reduction which is similar to the approach adopted by the VGG networks as well. What truly set GoogLeNet apart and allowed them to place above VGG in the ImageNet challenge, though, was their implementation of convolutions, where the stacking of convolutions was done not only in a sequential manner but also on the same level. This allowed them to capture both global and local convolutions more effectively due to different sized filters being employed at the very same level. This idea of going deeper not only in laterally but also horizontally was inspired by the theoretical work met out by Arora et al. regarding sparse structure and how modules with high

correlations should be grouped together to connect the previous layer to the next layer by forming filter banks [7].
Eventually, more versions of the Inception model were released which used, more frequently, the concept of factorised convolutions (as seen in the VGGnets), and enhanced it to break up square filters into pairs of one-dimensional filters (a filter of size NxN would be now factorised to produce two filters viz. Nx1 and 1xN). The Inception Version 3 (InceptionV3) extended the same concepts from previous versions, further reducing the scope of representational bottlenecks, while adding various new approaches such as Batch Normalisation, Label Smoothing, utilisation of the RMSProp Optimizer.

The year right after the Inception and VGG networks got introduced, researchers engineered a neural network [8] that could incorporate an exponential amount of layers while continually increasing the performance of the model, something which had been elusive to the entire deep learning community prior to this (successful efforts were made to mitigate this, but none came close to the breakthroughs achieved by this network).

The two major issues that had served as the hurdles in the quest for going deeper in CNNs were- the vanishing/exploding gradient problem and the computational time (and power) to train these networks.

The Residual Networks (or ResNets) designed by researcher mitigated both of these issues with the introduction of skip-connections, revolutionary concept that helped it achieve great heights through treading in the depths unvisited by any other CNN till 2015 (at least, effectively enough to be documented) [8]. Their deepest network comprised of a whopping 152 layers, which was almost 8 times the number of layers which were boasted by both VGG and Inception the very last year! It achieved an error of 3.57% (less than half of what was achieved by VGG last year) and it surprised no one when He et al. clinched the top spot in both the COCO and ImageNet Challenges.
The skip-connections are simply identity shortcut connections which stack identity mappings while skipping over one or more layers to ensure that the vanishing gradient problem doesn't occur. He et al. theorised that the introduction of these residual blocks should allow these networks to achieve a training error lower than that of their shallow counterparts. Many other networks such as ResNeXt- a blend of ResNet and Inception which used a hybrid architecture based off both these networks which introduced a hyper-parameter called cardinality corresponding to the number of independent paths that could be taken while traversing the network [9], DenseNet- a network which uses skip-connections while exploiting feature reuse by aggregating the feature maps with depth concatenations [10], [11].

MobileNet is another recent model, which employs the use of factorised convolutions to drastically reduce training time and the model size. It utilizes the concept of depthwise separable convolutions [12], similar to the one applied in Inception models, to a larger extent to create an extremely light-weight deep neural architecture with highly optimised latency to account for a more accessible model with low computational requirements. MobileNet also features two novel hyper-parameters viz. the width multiplier- α, and the resolution multiplier- ρ, which allow it to transition into being even more light-weight) [13].

Table 1: Summary of ImageNet Classifiers.

Classifier	Year of Conception	Recognitions	Model Error	Defining Features
AlexNet (CaffeNet)	2012	Winner, ILSVRC Classification 2012	15.3% Top-5 Error	First Deep CNN to win ILSVRC, utilised dropout and ReLu. Incorporated 8 learning layers.
VGGNet	2014	Runners Up, ILSVRC Classification 2014 and Winner, ILSVRC Localisation 2014	7.32% Top-5 Error	Implemented smaller filters to reduce overfitting, the number of parameters, and time taken to converge. Incorporated 19 learning layers.
InceptionNet	2014	Winner, ILSVRC Classification 2014	6.67% Top-5 Error	Implemented horizontal stacking of convolutions along with the characteristic vertical (sequential) stacking, and 1x1 filters to reduce dimensionality and, hence, number of parameters. Incorporated 22 learning layers.
ResNet	2015	Winner, ILSVRC Classification 2015 and the COCO Classification Challenge 2015	3.57% Top-5 Error	Implemented skip-connections to mitigate the vanishing gradient problem. Incorporated 152 learning layers.
MobileNet	2017	Achieved similar results to state-of-the-art with reduced parameters	29.4% Top-1 Error (outperforms VGGNet and InceptionNet)	Introduced width and resolution multipliers as parameters to alter the weight of the model. Introduced depthwise separable convolutions to reduce the model size and complexity.

We now briefly discuss the related work in the field of Image classification models and its evolution in recent years. Datasets are a vital part of any image recognition research. One might hope that a given classifier would perform well on a new data set assembled from the same source following the same protocols. Researcher showed that early datasets resulted in overfitting leaving classifiers with minimal accuracy on other datasets [14]. Studies showed that deeper networks showed higher accuracy across various transfer tasks whereas wider networks showed lower accuracy [2], [14], [15].

Researchers have replicated a new dataset based on two prominent benchmark datasets- CIFAR10 & ImageNet and demonstrated that even with small variations, the current classification models generalize the new data with accuracy drops ranging from 3%-15% on CIFAR 10 and 11%-14% on ImageNet [13], [15],

[16]. Their studies show that adaptivity is an unlikely explanation for the accuracy drops and that the differences in accuracy stem from a larger distribution gap between the datasets, while Adaptivity and Generalisation gaps are also listed as additional possible causes.

In a substantial research presents a comparative study of classifiers using popular datasets based on relative data bias and cross data generalisation showing how a given classifier trained on one dataset performs on a different dataset. The authors have tried to raise awareness about an important issue i.e. bias in datasets. The results of the in-depth study of cross data generalisation were rather disheartening, since almost all the datasets are assembled from one source- the Internet which raises too many red flags [4], [6], [17]. A simple explanation to this was bias in data datasets such as selection, label or even capture bias but most importantly, the negative set bias.

The major difference in our research is that we tried to study to what extent does these classifiers generalise on different datasets, thereby showing the extent of overfitting and how effectively these classifiers generalise when used across various target sets with limited data which in turn provides a substantial base for using a specific approach/algorithm under specific conditions.

2.2 Learning with the Training Wheels On

Transfer Learning is an investigating technique utilized in Machine Learning to use data learnt in previous problems by applying it in related existing problems. This concept that vies for the cross-utilization of knowledge to tackle related, novel problems is inspired from the intrinsic ability of the human brain to transfer knowledge across different tasks [21]. Transfer Learning is touted to be the next big driver of the professional success of Machine Learning by many eminent individuals in the deep learning community [10], [11], [18]. There have been various comprehensive reviews of transfer learning that have incorporated recent advancements in the various types of transfer learning that exist [9][1], [19].

It can be interesting to see the situations in which Transfer Learning is adopted when we focus on the formal definition used for it. We can look at the framework provided employing the usage of domains, tasks, and marginal probabilities [1], [19].

The framework, concisely put, consists of a domain, D, defined as a two-element tuple encapsulating feature space, 𝒳, and marginal probability, P(X), where X represents a sample data point ($X=\{x_i\}_{i=1}^{n}$, where i=1,...,n with n sample points) and a Task, T, defined as a two-element tuple encapsulating the label space, γ, and objective (predictive) function, η, where η can also be represented by P(γ|X) employing conditional probability. η is learned from feature-label pairs- (x_i, y_i), such that $\eta(x_i)=y_i$.

D can therefore be mathematically be represented as, D = {𝒳, P(X)}, whereas T can be mathematically represented as, T = {γ, P(γ|X)}.

With this framework in mind, Transfer Learning may be defined as the process of learning the target conditional probability, $P(Y_T|X_T)$ in D_T with the help of the information gained from D_S and T_S (where,

$D_S \neq D_T$ and $T_S \neq T_T$), where, D_S and T_S are the source domain and corresponding source task and D_T and T_T are the target domain and corresponding target task, respectively.

Now, this leaves us with 4 typical situations in which Transfer Learning may be utilised:-

i. $\mathcal{X}_S \neq \mathcal{X}_T$ - In this situation, the feature space of the source domain and the target domain are different.
ii. $P(X_S) \neq P(X_T)$ - In this situation, the conditional probability distributions of the source domain and target domain are different.
iii. $\gamma_S \neq \gamma_T$ - In this situation, the label space of the source domain and target domain are different.
iv. $P(Y_S|X_S) \neq P(Y_T|X_T)$ - In this situation, the conditional probabilities of the source task and target task are different.

Even though almost every Transfer Learning implementation would generally employ all of the four situations partly, our work and experimentation primarily focuses on the third and fourth situation, wherein the size and manner of the label space and conditional probabilities of the initial and final tasks differ a lot from the ImageNet winners' work.

Transfer Learning has wide reaching applications which pervade all fields of Artificial Intelligence and Machine Learning but become extremely useful when we take into consideration our own problem- Image Classification. Since, Image Classification depends heavily on Feature Extraction and most of the layers while training contribute to the same concept, using a model already trained to extract features makes sense. But, transfer learning doesn't simply use a pre-trained model on a new dataset. In our case, it takes in the necessary layers required for feature extraction which are already trained and connects them with some Fully Connected (FC) layers with some dropout to make sure that the pre-trained model "learns" to use the pre-trained models' learned characteristics to learn the characteristics of the new problem set, hence, creating a new model catered to our problem.

3. Methodology

In our work, we use 8 different classifiers to derive the needed results. Five of the aforementioned classifiers are used in a transfer learning framework with the ImageNet trained models (without the dense layers) used as the base models, whereas, the rest of the models (viz. the DNN, CNN, and AlexNet) are trained and validated purely on our dataset. Even though the expectations for these (non-transfer learning) primitive networks were minimal, we included these in our study to illustrate the contrast in performance and delineate the relative superiority of the models employing transfer learning.

Our dataset consists of four basic classes, having only around 800 images. These classes are generalized supersets of the many specific classes found in ImageNet and, thus, more than viable for Transfer Learning.

Our approach algorithmically, outlined briefly, is as follows:

1: Preprocessing the data & generating augmentations through data generators.

2: Downloading the base model weights and architecture (wherever applicable)

3: Applying transfer learning by adding custom dense layers to the model (wherever applicable)

4: Compiling the model & training it on the augmented data.

5: Checking for & mitigating over-fitting through the use of dropout, batch normalization (wherever applicable)

6: Getting the validation results & plotting them to compare performances.

7: Calculating the relevant scores on the test set.

8: Checking for cases where early stopping might be utilised to save the best parameters.

9: Comparing all these results to obtain the best performing classifier.

To ensure a standard for comparison, we utilized the same amount of Fully Connected Dense Layers (hereon, referred to as FC) with the same amounts of nodes after the convolutional layers of the base model used (as seen in Table 2 (a) below). The exception to this was our Basic Deep Neural Network as we felt that to ensure a fair comparison and to make up for the absence of any feature extraction (due to the omission of convolution layers), the DNN needed extra layers (the composition can be seen in Table 2 (b) below).

Standard Network Architecture
Image Input
Base Model
Flatten
FC 1024
FC 512
FC 256
FC 128
FC Softmax

DNN Network Architecture
Image Input
Flatten
FC 2048
FC 1024
FC 512
FC 512
FC 256
FC 256
FC 128
FC 128
FC 128
FC 64
FC Softmax

Table 2: (a) Standardised Network Architecture used for all models except DNN. (b) Network Architecture used for DNN.

Each layer in all our models are accompanied by ReLU non-linearity [3] barring the last Fully Connected layer which is followed by a Softmax layer to realize classification. Since parameter tuning is the main oil that makes the cogs turn in a neural network, it is desirable to have a lot of parameters available for tuning and tweaking. But, in order to have a lot of parameters, we need to have a lot of training examples. All our networks were very prone to overfitting due to the limited amount of data that was available to train with. To reduce overfitting, we've also employed Data Augmentation [20],[6], [21], [22] to artificially increase the size of the database by using label-preserving image transformations [14]. Data Augmentation works on the principle of mathematical transformations such as scaling, rotation, translation, cropping, flipping the image on both the horizontal and the vertical axis, off-center randomized zoom. We use three prominent transformations, namely, horizontal flip, shear, and zoom. All of these are applied with randomness in each iteration of the epoch so that the "same" image is never visited twice by the classifier during training to increase the model's ability to generalize [6], [12], [17]. The value we used for the scope of randomness in the zoom transformations was set to 0.2 (this corresponded to a range of [0.8, 1.2]). We have given a representation of the impact created by data augmentation by providing results with and without data augmentation.

Dropout was also considered but, after considerable testing, proved to be superfluous (quite ironically), potentially, because the training dataset, being small, didn't necessarily require random co-adaptation of

neurons to avoid overfitting [5], [15].

4. Results And Discussion:

The authors have completed a comprehensive survey of all the milestone image classification algorithms used and tested them in the aforementioned conditions. The major project includes the implementation and outlines the explicit and implicit comparison of the aforementioned algorithms. Various combinations of regularisation, hypertuning, activation functions were used and hence, the best outcomes (to the best of our humble abilities, that is) determined. The authors have also examined various techniques to reduce loss while comprehending neoteric discoveries in the sphere of Deep Learning. We hope that this comparison serves as a beginning point for others to choose and compare algorithms for their limited-data image classification nets. All hyper-parameters have been set after a lot of trial and error to provide a fair and standardised comparison. We notice many unintuitive results after conducting our work which were shocking when considered to their ImageNet performance. We have employed many measures to reduce overfitting and loss and if we glance at the graphs for accuracy above (comparing the training and validation accuracies), we realise that even after these measures, there is still a lot of overfitting taking place owing to the small amount of data available to us.

It is fair to assume that this is the result that would be replicated in any other such similar circumstance while using these algorithms with limited data. These results, thereof, are not and should not be considered endemic and specific to our dataset. These results might be expected to be replicated as long as the classes to be classified are similar to the ones already present in the ImageNet database, barring which, only the results of DNN, CNN, and AlexNet would be of insight to the user. But, taking the more probable (and certainly more hopeful and convenient) scenario of the target classes (the ones you wish to classify) being similar to the source classes (the ones present in ImageNet), this conclusion should serve as a starting point for the readers to discern which approach is the best to utilise and, hence, spend resources like time for optimisation and computation on.

Without further ado, these are the results that we received with our implementation:-

Table 3
Accuracy, Loss, and Runtime Results achieved by our fully trained models.

Model/ Algorithm	Training Accuracy (%)	Validation Accuracy (%)	Training Loss	Validation Loss	Total Runtime+ (Minutes)
Deep Neural Network	61.88	63.75	0.8592	0.8186	19.90
Convolutional Neural Network	97.07	76.88	0.0835	0.8429	27.41
ResNet-50*	98.30	98.12	0.3368	0.4644	23.18
VGG-16*	98.77	90.00	0.0606	0.8151	27.51
VGG-19*	99.69	95.00	0.0036	0.1068	33.40
MobileNet*	99.85	96.25	0.0165	0.0245	14.48
Inception-V3*	100	98.12	0.0015	0.0891	13.08
AlexNet (CaffeNet)	91.05	68.75	0.2629	1.1898	14.16

*Transfer Learning applied to models pre-trained on ImageNet

+ Using a Google Collaboratories TPU backend

We also calculate the scores corresponding to our models based on their implementation calculated from individual confusion matrices as shown in Table 4 below.

Table 4
Scores achieved by our fully trained models based on their classification.

S.No.	Classifier	Precision	Sensitivity	Specificity	F1-score
1.	DNN	0.64	0.64	0.8775	0.64
2.	CNN	0.77	0.71	0.92	0.74
3.	AlexNet	0.68	0.60	0.88	0.64
4.	VGG16	0.90	0.97	0.99	0.93
5.	VGG19	0.95	0.89	0.97	0.92
6.	ResNet	0.98	0.98	0.91	0.98
7.	InceptionNet	0.98	0.98	0.99	0.98
8.	MobileNet	0.96	0.98	0.99	0.97

We, as mentioned earlier, see certain results which digress from the normal (and ImageNet results).

As expected, the DNN is the one achieving the lowest accuracy owing to its inability to efficiently extract features and, thereby, stunting it from achieving a high accuracy. It's accuracy of 63.75 is still impressive considering that the time taken for it to train and validate is just around 20 minutes. One thing to note here, though, is that training this DNN required very high computational requirements where it even made GoogleColab's Tensor Processing Unit crash while we implemented it owing to the high number of nodes it took to train the said network. Suffice to say, this is not a network that should be used for image classification of any type and it is probably wise to steer away from this model while trying to achieve valuable results.

Next on the list, shockingly enough, coming second to last if we order our implementation by the accuracy achieved is AlexNet (the CaffeNet version). This might come as a shock to some that the prodigal model which shook up the entire academia by winning the ILSVRC 2012 loses to a normally trained one-layer CNN (68.75% as compared to 76.88%). However, upon a closer look, there are three things to note that could possibly explain this anomaly. Firstly, this is a CaffeNet version of AlexNet. The difference, as explained in the literature survey of AlexNet, is that while the actual AlexNet is trained on two GPUs which significantly increases its accuracy, the CaffeNet model is just implemented on a singular GPU. Secondly, unlike all other algorithms (whose genesis came from the ILSVRC) we implemented in our work, our version of AlexNet (or rather, CaffeNet) is not pre-trained on ImageNet. That is to say that the weights assigned are purely trained from scratch and there is no transfer learning applied. Thirdly, and perhaps most importantly, the implementation of CaffeNet which we used was one in which we stuck to the archaic norm of having large filter sizes (as defined in the AlexNet architecture) of the order 11x11, 7x7, and 5x5, as compared to the recent norm of using simple, small 3x3 filters with deeper CNNs; also, the filters used were also massive (starting from 96 filters in the first convolutional layer and reaching up to a whopping 384 filters in the third and fourth convolutional layers) to account for the vast size of ImageNet as compared to the mere 32 filters used for CNN. These three points and the two differences (crystalised in the last point) make a huge difference and, hence, affect our accuracy to a large degree especially when the limited size of the dataset is taken into consideration. This to an extent, possibly, explains why our AlexNet with 5 Convolutional Layers was one-upped by our CNN with just 1 Convolutional Layer.

However, another thing to note is that while our CNN takes 32.9 seconds to train on one epoch, AlexNet does the same in just 17 seconds (which is almost half the time). Therefore despite the archaic and, now, outdated architecture our AlexNet makes up for more than what it lost on accuracy by the time it saved in training (which further opens up avenues for optimization without drastically increasing the time).

The rest of the results are fairly in tandem with what the readers might expect out of them. After CNN, we have VGG-16 which provides an accuracy of 90% (17% more than what was achieved by the CNN we trained). It is followed by VGG-19 which gives us an accuracy of 95%.

Both these networks take a lot of time to train (33 s/epoch for VGG-16 and 40 s/epoch for VGG-19) owing to their large size and lack of options to reduce computational requirements and are therefore not highly

feasible for industry implementation.

ResNet50 and Inception-V3 share the highest accuracy of 98.12 between each other, closely beating MobileNet which ended its run with a 96.25% top-1 accuracy. However, ResNet50 falls short because of the massive amount of time it takes to train (27.8 s/epoch) as compared to Inception-V3 and the light-weight MobileNet (15.7 s/epoch and 17.4 s/epoch respectively).

Table 5
Accuracy, Loss, and Runtime Results achieved by our fully trained models without data augmentation.

Model/ Algorithm	Training Accuracy (%)	Validation Accuracy (%)	Training Loss	Validation Loss	Total Runtime+ (Secondss)
Deep Neural Network	86.68	58.75	0.3477	1.4646	497
Convolutional Neural Network	100	66.25	8.16e-7	3.3457	2032
ResNet-50*	81.54	71.25	0.4359	0.7303	984
VGG-16*	100	94.38	3.49e-6	0.1761	1688
VGG-19*	99.69	95.00	0.0036	0.1068	1993
MobileNet*	100	98.75	1.72e-8	0.0562	302
Inception-V3*	100	99.37	2.49e-7	0.0113	502
AlexNet (CaffeNet)	90.20	60.00	0.2531	2.0627	666

We also look at the effects of data augmentation and the impact it has on the accuracy and loss of our classifiers on the dataset in Table 5. As preempted, we can see a lot of overfitting taking place as compared to the case where we used data augmentation. We can also observe a drop in accuracy for most of the networks barring VGG16, VGG19, MobileNet and InceptionV3 which perform similar to their original performance (with augmentation). We see an astounding drop in the accuracy of ResNet from 98.12 to 71.25, however other networks display increased robustness. DNN, CNN, and CaffeNet also see a drop in accuracies, albeit much less severe than that of ResNet.

Surely this analysis would then make the readers believe that Inception-3 is hands-down the best algorithm to use in terms of both, the time taken to train and the accuracy received. This analysis can only be deemed partially true, however, because there is still a facet that we need to analyse in order to do justice to this comparative analysis, that is, accounting for the standardization.

We have used 50 epochs and 4 dense layers with the exact same number of nodes (along with adams optimizer with default learning rate) to standardize our comparison and provide the readers with an efficient comparison and guide to utilise models in small datasets. However, one cannot help but wonder

what if, with adequate tweaking, other algorithms might perform better than the current top-spot position holder- InceptionV3.

While there is no way to conclusively answer this question, there is a way to pointedly tackle one facet of it through our work. We can do so by looking at the highest validation accuracy that was achieved throughout the epochs and their corresponding training accuracies.

One of the run-of-the-mill ways to counter over-fitting is “early stopping”. However, it is imperative to realise whether the validation accuracies peaked due to natural circumstances of good training or whether it was just an anomaly (brought out by flukes) which later got rectified as the epochs continued. In order to verify this, it’s very important to take note of the training accuracies in order to definitively (to an extent) tell whether a model actually would perform better with early stopping.

Now, let’s take a look at Table 5 (which does just that) providing you with the Highest Validation Accuracy Received along with the Corresponding Training Accuracy and the Epoch on which it achieved this.

Table 6
Analysis of Results considering Early Stopping.

Model/ Algorithm	Highest Validation Accuracy Received (%)	Corresponding Training Accuracy (%)	Epoch
Deep Neural Network	65.62	60.8	47
Convolutional Neural Network	83.13	96.91	49
AlexNet (CaffeNet)	72.5	83.49	49
ResNet-50*	99.37 [R]	98.61	7
		99.23	28
		99.23	30
		100	44
VGG-16*	96.25 [R]	99.85	21
		96.14	28
		100	49
VGG-19*	96.88	100	49
MobileNet*	100 [R]	98.3	4
		99.69	10
		100	17
		99.54	22
		100	32
		100	34
		99.85	38
		100	41
Inception-V3*	99.37 [R]	98.46	25
		99.38	31
		99.69	36
		98.77	40

*Transfer Learning applied to models pre-trained on ImageNet

[R] Received on Multiple Epochs

We see that almost all algorithms (apart from DNN because of its lower training accuracy) performed better before hitting the 50th epoch. The 5% accuracy difference between both the VGG models gets reduced to a measly 0.63% difference whereas ResNet50 and Inception-V3 still enjoy extremely high matching accuracies of 99.37%.

What is a bit out-of-the-blue, though, is the accuracy achieved by MobileNet. With a whopping 100% validation accuracy- a perfect score, if there ever were one- the MobileNet blows all of its competitors out of the water. To concretize that it was not just a fluke or an anomaly, we see it repeating this glorious accolade an astounding 8 times (twice the amount of times it was done by our previous (now dethroned) top-spotters- ResNet50 and InceptionV3. To further establish this point, we see that the corresponding train accuracies are also in tandem with the results achieved in Validation Accuracies.

With its low computation time of just 17.4 seconds per epoch and the high validation accuracy that may be achieved through early stopping, the MobileNet is a neck to neck competitor to the Inception-V3 with its similar accuracies and train time.

6. Future Scope

The Future Scope and Implications are pretty clear in how there needs to be a rising focus on achieving higher state-of-the-art efficiencies in smaller datasets rather than just trying to build a model that predicts after being fed astronomical amounts of data. There have been many recent strides towards that direction through Tiny ImageNet, however, we feel there is still scope for more focus on this endeavor.

However, some improvements that readers might implement include delving into Generative Adversarial Neural (GAN) Networks (Dueling Neural Networks) and various other approaches and techniques that have emerged recently and require a high level of expertise to implement and possesses high computational requirements. GAN, specifically, an innovative and disruptive technique (network) to expand and artificially create a larger dataset with the limited data available which may be utilized by the readers for making their classification problem more approachable via networks which work better with more data available for training, viz., ResNet in contrast with MobileNet.

7. Conclusion

In conclusion, we feel that the two questions that we posed at the start of the introduction setting out the basis and need of this research work are sufficiently tackled.

The first point, which questioned the authenticity of the results of the ILSVRC (shrouded by allegations of overfitting) stands answered when we look at how all the algorithms performed in the same order we expected them to (barring AlexNet, for which a detailed possible explanation is outlined above). We also tested out pre-trained models to see if they would work on a target dataset with similar expectations, and that too was fulfilled and verified in the same breadth.

The second point which questioned the efficacy of these models and algorithms on a smaller dataset and asked for a comparative analysis has also been fulfilled by authors as outlined above.

Lastly, to summarise, we'd like to suggest our readers to employ either Inception-V3 or MobileNet based on the computational apparatus available and the task to be completed when faced with a choice between these algorithms. The MobileNet while it's incredibly viable because of its light-weight nature, low computational requirements and high validation accuracy is then given competition and sufficiently rivaled by Inception-V3 which has (albeit, higher computational requirements but also) lower train time, and equally high validation accuracies (if not higher).

Declarations

Conflict of Interest: authors declare no conflict of interest.

References

1. B. Recht, R. Roelofs, L. Schmidt, and V. Shankar, "Do {I}mage{N}et Classifiers Generalize to {I}mage{N}et?," in *Proceedings of the 36th International Conference on Machine Learning*, 2019, vol. 97, pp. 5389–5400, [Online]. Available: http://proceedings.mlr.press/v97/recht19a.html.
2. J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," in 2009 *IEEE Conference on Computer Vision and Pattern* Recognition, Jun. 2009, pp. 248–255, doi: 10.1109/CVPR.2009.5206848.
3. S. Y. Chaganti, I. Nanda, K. R. Pandi, T. G. N. R. S. N. Prudhvith, and N. Kumar, "Image Classification using SVM and CNN," in 2020 *International Conference on Computer Science, Engineering and* Applications *(ICCSEA)*, Mar. 2020, pp. 1–5, doi: 10.1109/ICCSEA49143.2020.9132851.
4. T. Wiatowski and H. Bölcskei, "A Mathematical Theory of Deep Convolutional Neural Networks for Feature Extraction," *IEEE Trans. Inf. Theory*, vol. 64, no. 3, pp. 1845–1866, Mar. 2018, doi: 10.1109/TIT.2017.2776228.
5. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014.
6. Z. Zhao, P. Zheng, S. Xu, and X. Wu, "Object Detection With Deep Learning: A Review," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 30, no. 11, pp. 3212–3232, Nov. 2019, doi: 10.1109/TNNLS.2018.2876865.
7. M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald, and E. Muharemagic, "Deep learning applications and challenges in big data analytics," *J. Big Data*, vol. 2, no. 1, p. 1, 2015, doi: 10.1186/s40537-014-0007-7.
8. K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in 2016 *IEEE Conference on Computer Vision and Pattern* Recognition *(CVPR)*, Jun. 2016, pp. 770–778, doi: 10.1109/CVPR.2016.90.

9. C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, “A survey on deep transfer learning,” in *In International conference on artificial neural networks*, 2018, pp. 270–279.
10. Y. Pang, M. Sun, X. Jiang, and X. Li, “Convolution in Convolution for Network in Network,” *IEEE Trans. Neural Networks Learn. Syst.*, vol. 29, no. 5, pp. 1587–1597, May 2018, doi: 10.1109/TNNLS.2017.2676130.
11. R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Region-Based Convolutional Networks for Accurate Object Detection and Segmentation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 1, pp. 142–158, Jan. 2016, doi: 10.1109/TPAMI.2015.2437384.
12. Z. Qin, Z. Zhang, S. Zhang, H. Yu, and Y. Peng, “Merging-and-Evolution Networks for Mobile Vision Applications,” *IEEE Access*, vol. 6, pp. 31294–31306, 2018, doi: 10.1109/ACCESS.2018.2843341.
13. G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely Connected Convolutional Networks,” in 2017 *IEEE Conference on Computer Vision and Pattern* Recognition *(CVPR)*, Jul. 2017, pp. 2261–2269, doi: 10.1109/CVPR.2017.243.
14. A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017, doi: 10.1145/3065386.
15. S. H. Khan, M. Hayat, and F. Porikli, “Regularization of deep neural networks with spectral dropout,” *Neural Networks*, vol. 110, pp. 82–90, 2019, doi: https://doi.org/10.1016/j.neunet.2018.09.009.
16. Https://, “No Title.”.
17. S. S. Yadav and S. M. Jadhav, “Deep convolutional neural network based medical image classification for disease diagnosis,” *J. Big Data*, vol. 6, no. 1, p. 113, 2019, doi: 10.1186/s40537-019-0276-2.
18. L. Hao, L. Gao, X. Yi, and Z. Tang, “A Table Detection Method for PDF Documents Based on Convolutional Neural Networks,” in 2016 *12th IAPR Workshop on Document Analysis Systems (DAS)*, Apr. 2016, pp. 287–292, doi: 10.1109/DAS.2016.23.
19. K. Weiss, T. M. Khoshgoftaar, and D. Wang, “A survey of transfer learning,” *J. Big Data*, vol. 3, no. 1, p. 9, 2016, doi: 10.1186/s40537-016-0043-6.
20. Anfeng He and Xinmei Tian, “Multi-organ plant identification with multi-column deep convolutional neural networks,” in 2016 *IEEE International Conference on Systems, Man, and* Cybernetics *(SMC)*, Oct. 2016, pp. 2020–2025, doi: 10.1109/SMC.2016.7844537.
21. Z. Abbas, H. Tayara, and K. t. Chong, “SpineNet-6mA: A Novel Deep Learning Tool for Predicting DNA N6-Methyladenine Sites in Genomes,” *IEEE Access*, vol. 8, pp. 201450–201457, 2020, doi: 10.1109/ACCESS.2020.3036090.
22. T. Sarkar *et al.*, “Spatial optimisation of mango leather production and colour estimation through conventional and novel digital image analysis technique,” *Spat. Inf. Res.*, 2021, doi: 10.1007/s41324-020-00377-z.

Figures

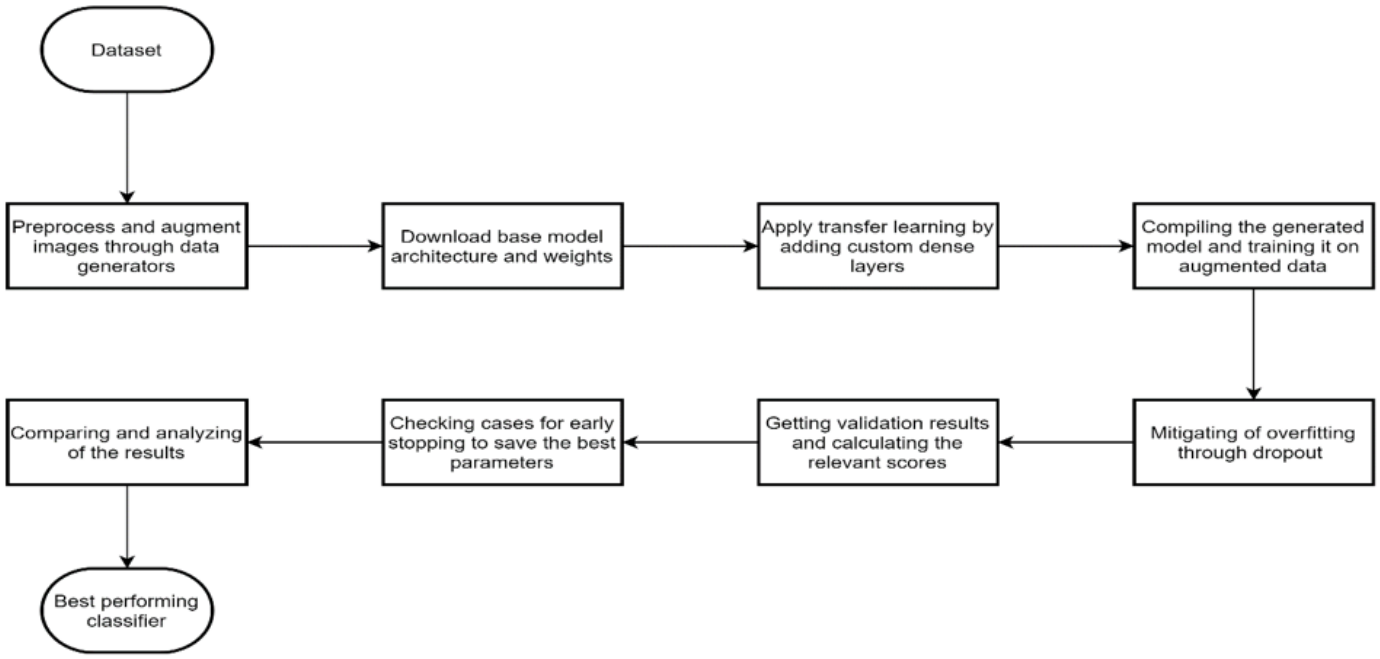

Figure 1

Representation of our methodology via a flow chart

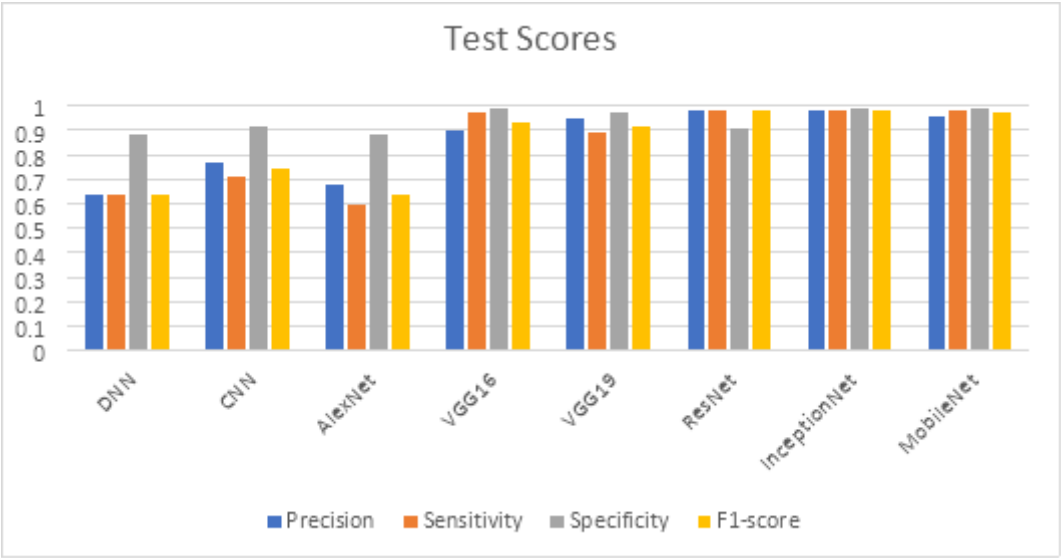

Figure 2

Graphical Representation of the scores achieved by the classifiers

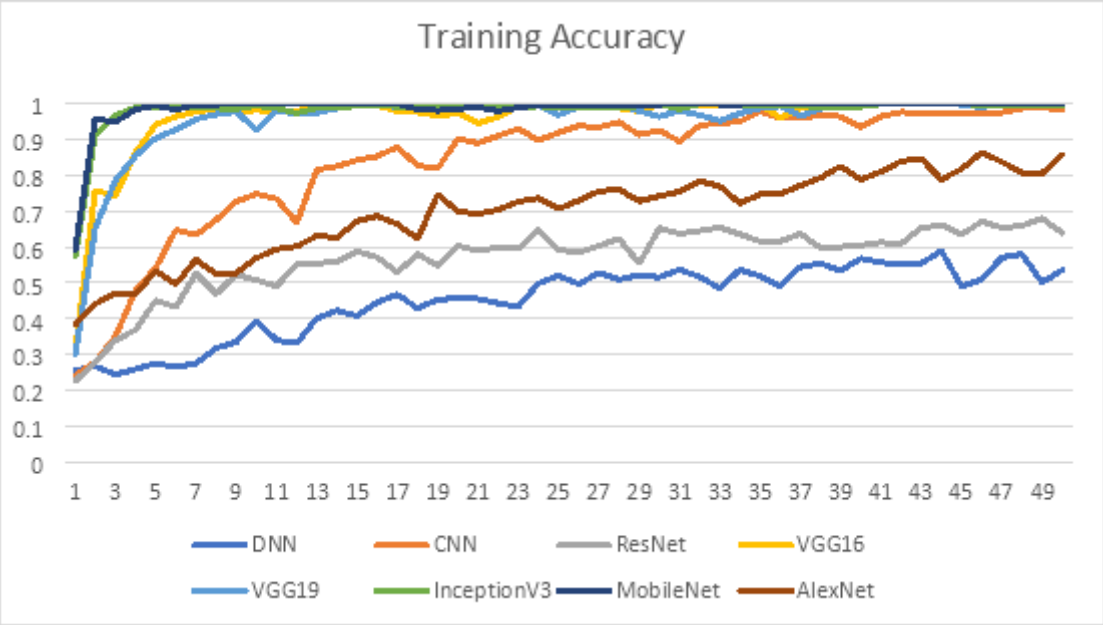

Figure 3

Graphical Representation of Training Accuracy achieved by classifiers

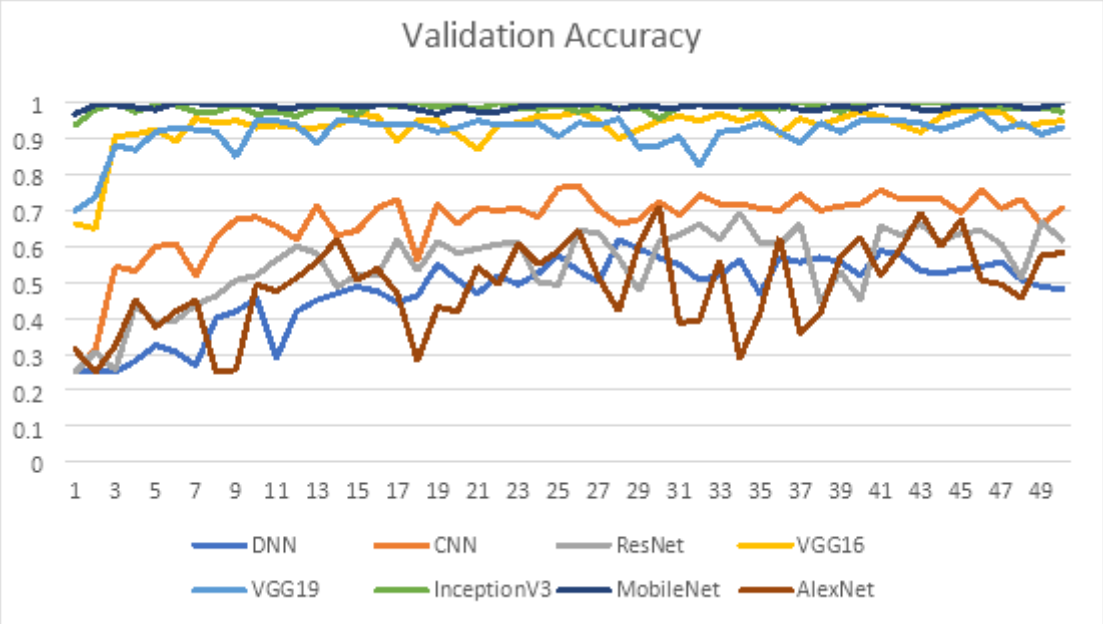

Figure 4

Graphical Representation of Validation Accuracy achieved by classifiers