Image Classification with CIFAR-10

Snithika Kalakoti, Hsi-Sheng Wei, Andrew Abrahamian
UC Berkeley, DATASCI 207 Applied Machine Learning, Section 3
Published April 18, 2023

Problem Motivation
We are a startup that builds machine learning models to help classify images. Specifically, we specialize in CNN models used to cluster and classify objects in aerial photographs and satellite images into one of the 10 labels: (0: airplane, 1: automobile, 2: bird, 3: cat, 4: deer, 5: dog, 6: frog, 7: horse, 8: ship, 9: truck). All 10 labels fall under 2 categories which are animal vs non animal. So our intention is to build a model that not only has a high accuracy but also has high categorical (animal vs non-animal) precision.

Our clients have use cases ranging from object detection in aerial photography shot via drones to object classification in traffic control systems that improve driver and pedestrian safety. For example, drone companies have seen an increasing demand for object detection software to enable users to easily identify different objects in their captured aerial photographs. Similarly, Object detection is a key technology behind advanced driver assistance systems (ADAS) that use real time satellite imagery to enable cars in detecting driving lanes or perform pedestrian detection to improve road safety.

The dataset chosen for this project is the CIFAR-10 dataset. Images in this dataset are relatively small (32x32) and have low resolution which fits our use case as compute resources on small devices (such as drone cameras) are limited and satellite imagery often has low resolution.

Dataset Description
For our project, we will use CIFAR-10 (Canadian Institute for Advanced Research, 10 classes, named after the funder of the project) by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton as our dataset for analysis.[1] The images in CIFAR-10 are from a much larger 80 million image dataset originally collected by several researchers at MIT and NYU over a 6-month span. Each image in the original dataset was associated with an initial label, which was the search term used to find that specific image on the internet. The authors of CIFAR-10 paid their students to double check and filter out the mislabeled images. CIFAR-10 consists of 60,000 labeled color images in 10 classes (airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck), with 6,000 images per class. Each image is 32x32 pixels, which is small in size when compared to some of the more recent image databases. The dataset is divided into a training subset of 50,000 images and another test subset of 10,000. The 10 classes are mutually exclusive, and there is no overlap between images.

Solution Approach & Experiments
Convolutional Neural Network Model Development

[1] Krizhevsky, A., & Hinton, G. (2009). Learning multiple layers of features from tiny images.

Our goal is to achieve a test accuracy above 80% in training an image classification model using the CIFAR-10 dataset. We adopted a step-by-step approach, progressing from multi-class logistic regression to more sophisticated models with convolutional layers in Keras. The CIFAR-10 dataset was split into a 50,000-image training set and a 10,000-image test set, with label arrays in both sets flattened. Within the training set, 5,000 images were designated as the validation subset, and the entire training set was shuffled. We compiled a basic logistic regression model with a single dense layer containing 10 output units and a Softmax activation function. The optimizer used was SGD, and the loss function was categorical cross-entropy. Training took place over 10 epochs with a batch size of 64 and a learning rate of 0.01. Results were underwhelming (40% accuracy for training data, 34% for validation data). A confusion matrix analysis indicated significant confusion between deer/frog and dog/cat, as well as frequent misclassification of airplanes, automobiles, and trucks as ships. Efforts to fine-tune the model by adjusting hyperparameter values to optimize performance were made, such as increasing the number of epochs to 30 and reducing the batch size to 20. However, these changes yielded minimal improvement (41% accuracy for training data, 35% for validation data).

For the next step, we constructed a straightforward feedforward neural network model featuring a single hidden layer with 1,024 units. The model's architecture was implemented using a learning rate of 0.01, sparse categorical cross-entropy loss, the Adam optimizer, and the Softmax activation function. We trained the model for 20 epochs with a batch size of 64. With a total of 3,157,002 parameters, the model achieved a training accuracy of 54%, a validation accuracy of 49%, and a test accuracy of 49%.

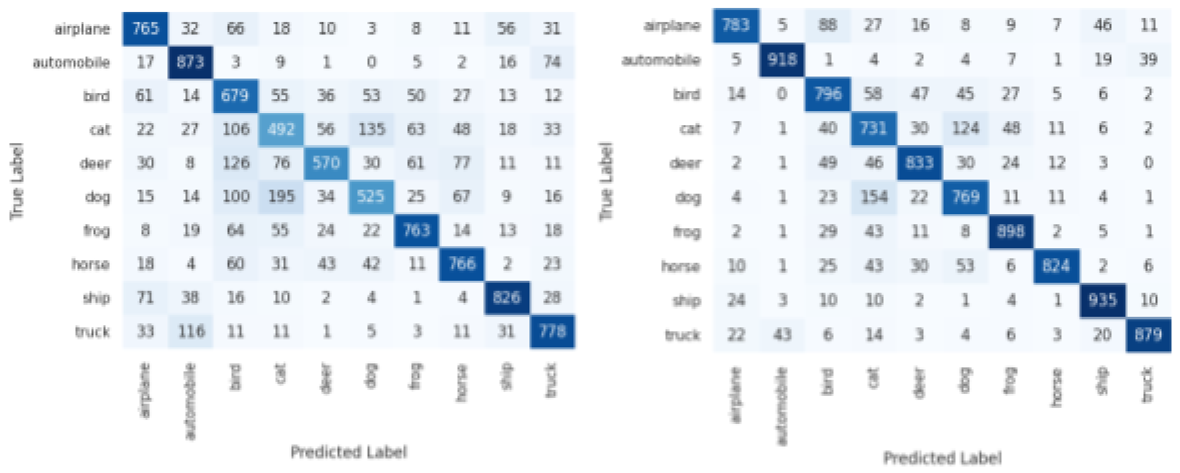
The confusion matrix of the FNN model for the test data revealed that the most significant misclassifications occurred between dogs/cats, deer/birds, cats/frogs, airplanes/ships, and trucks/automobiles. In particular, the predicted labels for birds and cats had the lowest precision (30% and 36%, respectively), while the true labels for deer, cats, and dogs had the lowest recall (40%, 37%, 23%). In summary, our feedforward neural network model demonstrated better performance with non-animals than with animals, but the overall accuracy still fell short of expectations.

During the third phase of our experiment, we constructed a series of convolutional neural network (CNN) models in succession. The CNN-1 architecture started with an input layer, followed by two convolutional blocks. Each block contained a Conv2D layer and a MaxPool2D layer. Both Conv2D layers utilized 64 filters of size (5, 5) and a ReLU activation function. The MaxPool2D layers in both blocks employed a pool size of (2, 2). We then added a dense layer with 1,024 units, followed by a dropout layer (rate=0.5) and an output layer with a Softmax activation function. The dropout layer helped minimize overfitting and enhance the model's generalizability. We trained the model for 20 epochs using a batch size of 64 and a learning rate of 0.01. The model had a total of 1,757,002 parameters and demonstrated significant improvement in performance (training accuracy=88%, validation accuracy=71%, test accuracy=70%). The primary confusions were between dog/cat, deer/bird, and truck/automobile.

In our improved CNN-2 model, we expanded each of the two convolutional blocks to include two Conv2D layers with 32 and 64 filters, and a kernel size of (5, 5). We also added a MaxPool2D layer with a pool size of (2, 2), and a dropout layer with a rate of 0.5. After these, we included two hidden layers with 1,024 units each, another dropout layer, and an output layer. This model, which had 5,411,370 parameters, achieved better test accuracy (73%) than CNN-1 (70%), with the most significant confusion being between dogs and cats.

We further developed a deeper CNN-3 model, consisting of three convolutional blocks and a BatchNormalization layer following each Conv2D layer. This model had 8,438,858 parameters and significantly higher accuracy compared to CNN-2 (training accuracy=94%, validation accuracy=83%, test accuracy=83%). Upon closer examination, we found that the model's lowest precision predictions were for cats and dogs (70% and 71%), while the highest precision predictions were for automobiles, horses, and frogs (91%, 90%, and 89%).

Since our goal of building a machine learning image classification model with a test accuracy over 80% was achieved, we chose CNN-3 as our final model. We then started another line of experiments, focusing on data pre-processing techniques for raw images. We hope that enhancing the quality of the images will further improve our model's performance.



	airplane	automobile	bird	cat	deer	dog	frog	horse	ship	truck
airplane	765	32	66	18	10	3	8	11	56	31
automobile	17	873	3	9	1	0	5	2	16	74
bird	61	14	679	55	36	53	50	27	13	12
cat	22	27	106	492	56	135	63	48	18	33
deer	30	8	126	76	570	30	61	77	11	11
dog	15	14	100	195	34	525	25	67	9	16
frog	8	19	64	55	24	22	763	14	13	18
horse	18	4	60	31	43	42	11	766	2	23
ship	71	38	16	10	2	4	1	4	826	28
truck	33	116	11	11	1	5	3	11	31	778

	airplane	automobile	bird	cat	deer	dog	frog	horse	ship	truck
airplane	783	5	88	27	16	8	9	7	46	11
automobile	5	918	1	4	2	4	7	1	19	39
bird	14	0	796	58	47	45	27	5	6	2
cat	7	1	40	731	30	124	48	11	6	2
deer	2	1	49	46	833	30	24	12	3	0
dog	4	1	23	154	22	769	11	11	4	1
frog	2	1	29	43	11	8	898	2	5	1
horse	10	1	25	43	30	53	6	824	2	6
ship	24	3	10	10	2	1	4	1	935	10
truck	22	43	6	14	3	4	6	3	20	879

Confusion Matrix Compare between CNN v1 (Left) and CNN v3 (Right

<u>Analyzing Impact of Data Pre Processing</u>

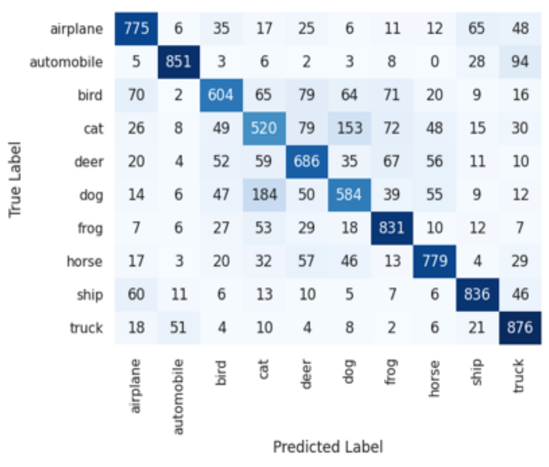
The quality of the data determines how well a machine learning algorithms can learn. The aim of pre-processing is an improvement of the image data that suppresses unwilling distortions or enhances Some image features can improve the overall quality of the data. This includes, but is not limited to, resizing, orienting, and color corrections. Image preprocessing may also decrease model training time and increase model inference speed.

Although enhancing images and data preprocessing can have a significant impact on accuracy of a machine learning algorithm, it does have some disadvantages. In addition to large power consumption, data preprocessing requires very large memory which can make it more expensive for you to allocate required RAM and run your models. Furthermore, image transformation does not always improve the model's accuracy, it heavily depends on the type of image (satellite, x-ray, animal, objects, etc.) and model structure. Therefore, a step by step analysis of each data transformation step is required in order to verify that the transformation is helping the accuracy of your model.

<u>Step-by-Step Analysis of Data Preprocessing</u>

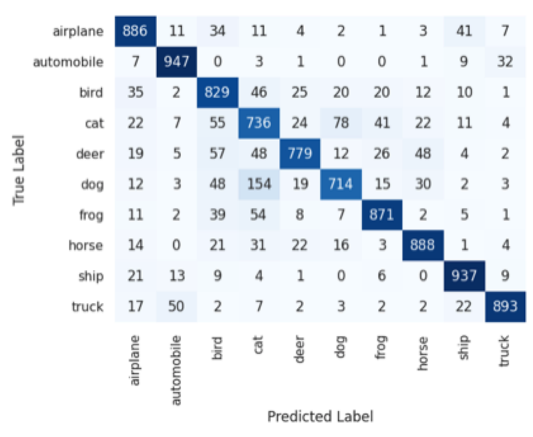
1. **Image Resolution/Size:**
   We noticed that Images in the CIFAR-10 dataset are relatively small in size (32 x 32). Therefore, our first step was to increase resolution and clarity of the image to 64 x 64. The next preprocessing. This was applied on both the training and validation set
2. **Increase Brightness, Contrast and add Random Flips from Left to Right:**
   The second step was to improve and emphasize contours of the object in an image. We saw most improvement in accuracy when brightness was set between a range of 2.5 and 3.5 with a contrast of 2. We also noticed that anything with a contrast of more than 2 decreased the testing accuracy. This was applied only on the training set.
3. **Data Augmentation:**
   At this point in testing, our confusion matrix showed that our model greatly confused predicting closely related objects such as a dog vs a cat. To solve this issue, we decided to increase the diversity of data gathered. We did so by applying augmentations to artificially inflate the training dataset by warping the original data such that their label does not change. This significantly decreased incorrect predictions among closely related objects i.e. dog vs cat as seen in the figure below. False Predictions for dog decreased from 153 to 78.

Without Pre-processing:



	airplane	automobile	bird	cat	deer	dog	frog	horse	ship	truck
airplane	775	6	35	17	25	6	11	12	65	48
automobile	5	851	3	6	2	3	8	0	28	94
bird	70	2	604	65	79	64	71	20	9	16
cat	26	8	49	520	79	153	72	48	15	30
deer	20	4	52	59	686	35	67	56	11	10
dog	14	6	47	184	50	584	39	55	9	12
frog	7	6	27	53	29	18	831	10	12	7
horse	17	3	20	32	57	46	13	779	4	29
ship	60	11	6	13	10	5	7	6	836	46
truck	18	51	4	10	4	8	2	6	21	876

With Pre-Processing



	airplane	automobile	bird	cat	deer	dog	frog	horse	ship	truck
airplane	886	11	34	11	4	2	1	3	41	7
automobile	7	947	0	3	1	0	0	1	9	32
bird	35	2	829	46	25	20	20	12	10	1
cat	22	7	55	736	24	78	41	22	11	4
deer	19	5	57	48	779	12	26	48	4	2
dog	12	3	48	154	19	714	15	30	2	3
frog	11	2	39	54	8	7	871	2	5	1
horse	14	0	21	31	22	16	3	888	1	4
ship	21	13	9	4	1	0	6	0	937	9
truck	17	50	2	7	2	3	2	2	22	893

<u>Analyzing Impact of Transfer Learning with Pre-Trained Models</u>

In image classification, transfer learning follows the logic that if a model is trained on a large enough dataset of images, then it will be able to generalize well to classify new images accurately. The features learned by these pretrained models can effectively act as a generic model of the visual world. This type of portability would be very useful for our use case, the classification of aerial and satellite images, because it allows us to save time and compute resources required to build out our own proprietary neural network. We used two different pre-trained models on our dataset, VGG16 and ResNet50.

The VGG16 network architecture, developed by Karen Simonyan and Andrew Zisserman in 2014[2], is a convolutional neural network that is 16 layers deep. The model is an older pretrained model, having been proposed and published within 2 years of AlexNet. However, it introduced meaningful innovations like shrinking both the kernel field and pixel strides within each layer and increasing the number of total layers. The advantage of using multiple smaller layers is that more non-linear activation functions accompany these layers, which improve the speed at which the model learns the data's features and allows for faster convergence. Second, the 3x3 convolutional filters are the smallest possible size to learn an image's spatial features which improve the efficiency of the network. However, the researchers did not solve the problem of *vanishing gradients* and could not typically increase the depth of their network beyond 20 layers.

We implemented a pre-trained VGG16 model with the following data augmentation procedures to our training data that were fixed across all model runs:

1. Flipping random images on the horizontal plane
2. Rotating random rotating images 10% counterclockwise
3. Zooming out of a random image by 20%

The first model run of only the VGG16 neural network architecture resulted in 80.3% training accuracy and 75.8% validation accuracy, with the model reaching 75% accuracy on the testing set of images.

To improve prediction accuracy, we added three densely connected layers [512, 256, 128] with accompanying dropout layers (0.4). These changes did not improve training accuracy and in fact, reduced both validation and testing accuracy by 3% and 2% respectively. The inclusion of too many layers likely caused the model to overfit to the training data resulting in poor generalization.

We shrank the size and number of densely connected layers [128, 32] and eliminated dropout layers in the third run of the model. We also **froze** all layers of the model until the fourth from the last. By freezing these model layers, we are able to more closely fine-tune the top two layers in the convolutional base. This dramatically improved training and validation accuracy to 85.9% and 82.4% respectively. The model learned in a much more linear fashion, with very little noise in either loss or accuracy measures as it progressed through training. This model reached 82.9% accuracy on the testing data, highlighting much stronger generalization capability versus the first two versions of the model.

[2] Karen Simonyan and Andrew Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," arXiv (2014), https://arxiv.org/abs/1409.1556.

Our fourth implementation of the VGG16 pre-trained model added two sizable dense layers ([1024, 1024]) with accompanying dropout layers (0.4) with the hopes of improving both the prediction accuracy and generalization ability of the model. Data augmentation procedures were removed to test the model's ability to predict without them. This model improved prediction accuracy to 99.4% on the training set and 86.1% on the validation set. It improved accuracy on the testing set, reaching 85.6% accuracy, a 2.5% improvement in prediction power.

ResNet50, introduced in 2015 with the ResNet family of models developed by He et al. at Microsoft Research[3], is a convolutional neural network that is 50 layers deep. The primary innovation introduced by the Microsoft team was an innovation they termed "*deep residual learning*" which solved for the problem of vanishing gradients. This problem can be explained as follows: As the backpropagation function chain increases in length for **very deep neural networks**, the error term used to adjust the parameters of each function in the chain would amplify and overwhelm the gradient information, causing backpropagation itself to stop working. The Microsoft team implemented *residual connections* adding the input of a layer back to its output such that it acts as an information shortcut around noisy blocks, enabling gradient information from early layers to propagate noiselessly through the many layers of a very deep network.[4]

To implement ResNet50, we had to upsample our training images (32x32) by a factor of 7 to reach the required (224x224) resolution of the pre-trained models input specifications. This procedure was fixed across all model runs.

Our first implementation of ResNet50 resulted in 98.4% training accuracy and 78.5% validation accuracy, with the model reaching 78.9% accuracy on the testing set. These results highlight the model's tendency to overfit the training data due to its size (24.6 million parameters).

We added two dense layers [1024, 512] with accompanying dropout layers (0.2) in the second implementation of ResNet50. This dramatically improved the model's performance on the validation set (83.9%, +5.4%) and testing set (83.6%, +4.7%). These improvements came at the expense of the model's size, which exploded to 126.9 million parameters.

The third implementation of ResNet50 was similar in size at 127.3 million parameters. The only difference was the inclusion of batch normalization layers that adaptively normalize samples within each layer using the mean and variance of the current batch of data. Despite limited explanations of why this occurs, the consensus seems to highlight that batch normalization helps with gradient propagation and allows for deeper networks without running into the vanishing gradients problem mentioned above.[5] Despite this change and the improvement in training accuracy (99.4%, +15.5%), the model did not generalize well and only reached validation and testing accuracy of 81.8%, meaningful performance deteriorations.

---

[3] Kaiming He et al., "Deep Residual Learning for Image Recognition," Conference on Computer Vision and Pattern Recognition (2015), https://arxiv.org/abs/1512.03385.
[4] Chollet, F. Chapter 9: Advanced Deep Learning for Computer Vision. Deep Learning with Python.
[5] Chollet, F. Chapter 9: Advanced Deep Learning for Computer Vision. Deep Learning with Python.

To offset this performance decline, we implemented a global average pooling layer and two dense layers [1024, 1024] with accompanying dropout layers (0.4) in the final ResNet50 model. We also opted to change the optimizer to ‘Adam’ to take advantage of its ability to adaptively optimize its gradient to improve training speed. This model reached 96.3% training accuracy, 88.2% validation accuracy, and 87.4% testing accuracy despite shrinking in size to 26.7 million parameters.

Conclusions and Results

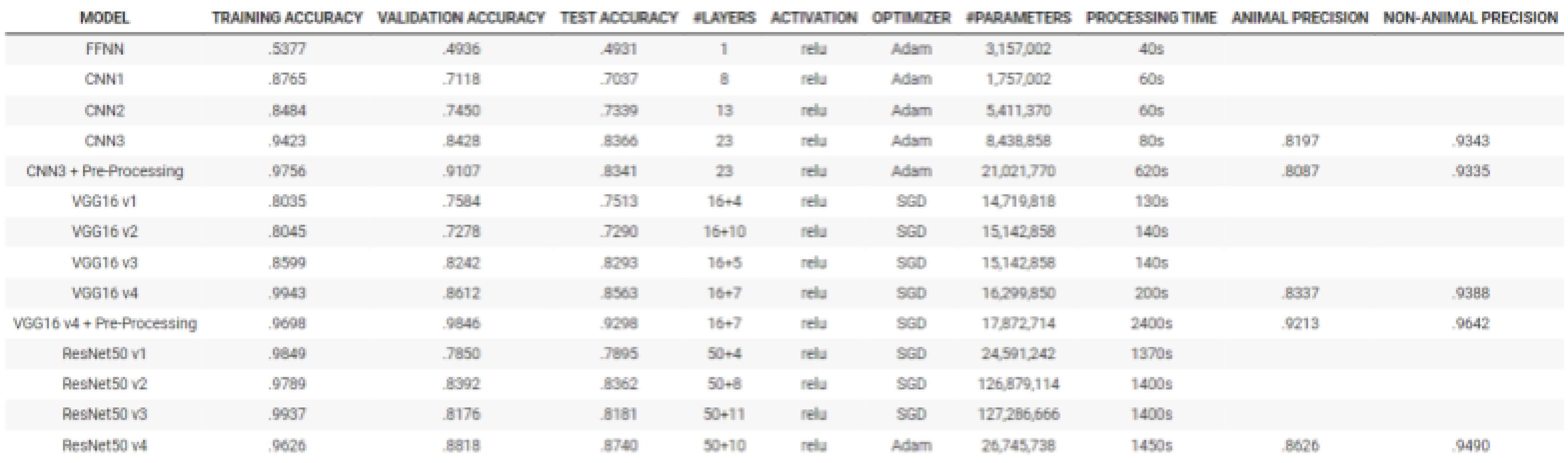
Our model results highlight the ability of convolutional neural networks to both predict features of images and generalize well to new, unseen images, as seen by our team’s best models routinely reaching >80% accuracy on the testing set of the CIFAR-10 dataset.

Generally and with few exceptions, this improved performance across model versions is accompanied by increased parameter counts, layer depth, and model size. However, this improved accuracy is not accompanied by improved precision across classes, as we see in the summary table below. In fact, our third CNN is more precise to the most performant VGG16 network architecture and within 5% of the most performant ResNet50 architecture for predicting animal classes. The results are even better for non-animal classes, with our third CNN performing within 1.5% of ResNet50 v4, a model that has >3x the parameters and takes >18x more time to train!

Data pre-processing steps had a significant impact on our models. Prior to pre-processing images, our models often confused similar objects such as cats and dogs. When preprocessed images were applied to models, we saw that it not only increased testing accuracy but also increased precision on both CNN v3 and VGG16. Particularly when image pre-processing was applied to VGG16, testing accuracy went up by 8%, animal precision went up by 10% and non animal precision went up by 4%. This clearly shows that resolution on our initial dataset is relatively low and increasing resolution and clarity results in better accuracy and precision on our models.

Despite these improvements, these data-preprocessing and augmentation procedures increased our compute requirements dramatically, often surpassing our GPU RAM capacity and increasing the training times of our models by as much as 10x. This held true even with the use of our pre-trained VGG16 and ResNet50 model architectures. This dynamic highlights that even our large and deep neural network architectures struggle to classify and differentiate the relatively low-resolution images (32x32) in the CIFAR-10 dataset. Though their performance improvements outpace our CNN v3 model accuracy and class precision, it is not worth the >2x increases in model size and the >10x increase in model training time.

Given our model results, CNN v3 not only meets our business use case needs (small model size and short processing time), it also meets our accuracy, generalization, and precision thresholds. We believe that given our constraints, this model is most appropriate for our business requirements.



MODEL	TRAINING ACCURACY	VALIDATION ACCURACY	TEST ACCURACY	#LAYERS	ACTIVATION	OPTIMIZER	#PARAMETERS	PROCESSING TIME	ANIMAL PRECISION	NON-ANIMAL PRECISION
FFNN	.5377	.4936	.4931	1	relu	Adam	3,157,002	40s		
CNN1	.8765	.7118	.7037	8	relu	Adam	1,757,002	60s		
CNN2	.8484	.7450	.7339	13	relu	Adam	5,411,370	60s		
CNN3	.9423	.8428	.8366	23	relu	Adam	8,438,858	80s	.8197	.9343
CNN3 + Pre-Processing	.9756	.9107	.8341	23	relu	Adam	21,021,770	620s	.8087	.9335
VGG16 v1	.8035	.7584	.7513	16+4	relu	SGD	14,719,818	130s		
VGG16 v2	.8045	.7278	.7290	16+10	relu	SGD	15,142,858	140s		
VGG16 v3	.8599	.8242	.8293	16+5	relu	SGD	15,142,858	140s		
VGG16 v4	.9943	.8612	.8563	16+7	relu	SGD	16,299,850	200s	.8337	.9388
VGG16 v4 + Pre-Processing	.9698	.9846	.9298	16+7	relu	SGD	17,872,714	2400s	.9213	.9642
ResNet50 v1	.9849	.7850	.7895	50+4	relu	SGD	24,591,242	1370s		
ResNet50 v2	.9789	.8392	.8362	50+8	relu	SGD	126,879,114	1400s		
ResNet50 v3	.9937	.8176	.8181	50+11	relu	SGD	127,286,666	1400s		
ResNet50 v4	.9626	.8818	.8740	50+10	relu	Adam	26,745,738	1450s	.8626	.9490

Table: Summary Model Results

Note: Data pre-processing steps includes resized training images to 64x64 resolution, adjusted image brightness (0.3), adjusted contrast factor (2), and randomly flipped training images.

Ethical Considerations

Our project, dataset and data follow all general ethical conduct guidelines. The CIFAR-10 data set is a publicly available dataset that does not contain any private or human derived data. Furthermore, our data does not consist of human images (Face, thumbprints etc).

Our team's only concern with the model being used in production systems is the potential impact of misprediction of Image classification. For example, wrong Image classification used in real time driver assistance systems may lead to road accidents (For example, a deer classified as a frog).

However, our business use case heavily relies on classification between object vs no object - detecting if an animal is present vs absent. Hence, we believe the above mentioned concern may be inconsequential.

Limitations

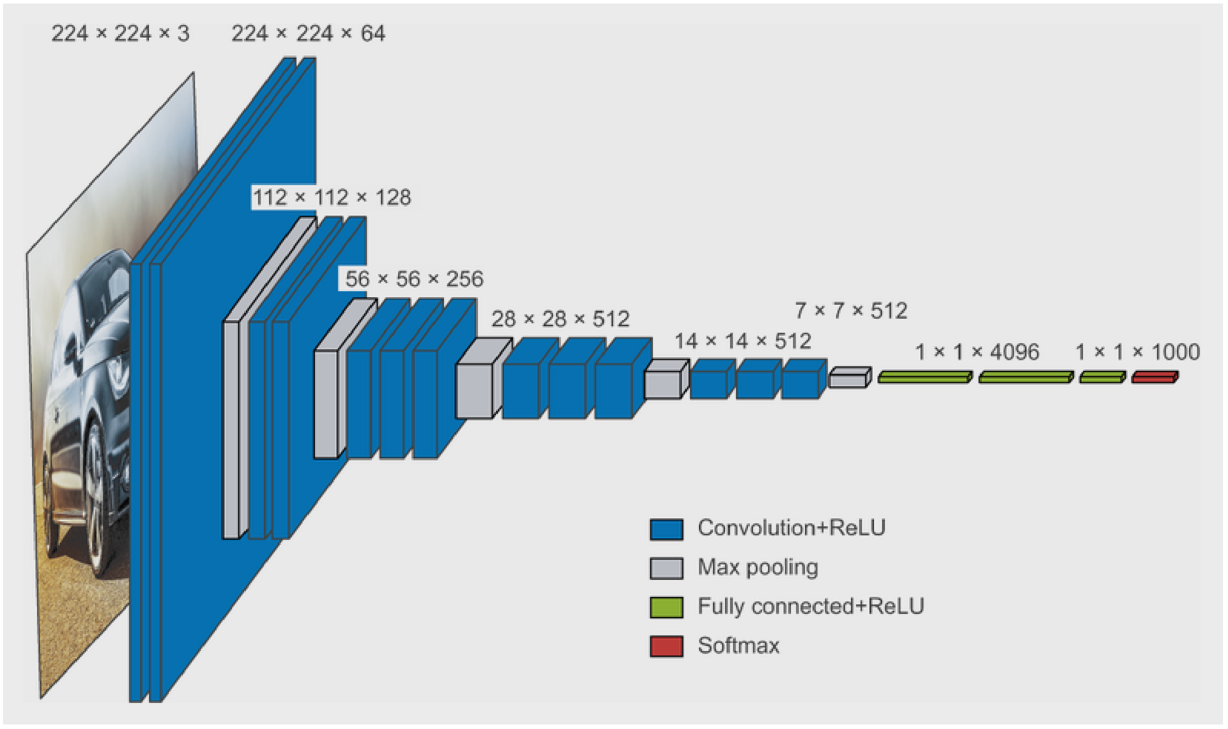
Compute resources for our project are heavily limited and restricted per our business use case. Therefore, we may not be able to use the most accurate model to accommodate our business use case criteria and conditions. Additionally, our models were trained on low resolution images, as camera tech in drones and satellite image resolution improves, we may have to re-train or update our model.

Appendix

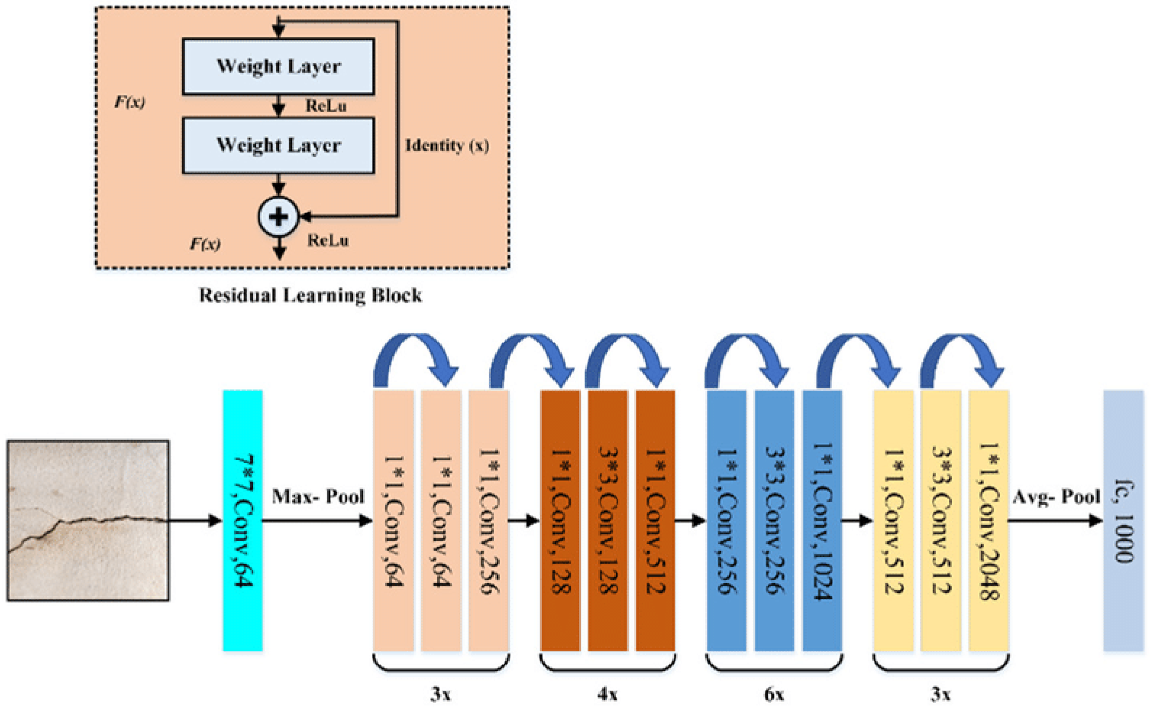
Item 1: NeurIPS 2021 Paper Checklist

1. For all authors...
    - (a) Do the **main claims** made in the abstract and introduction accurately reflect the paper's contributions and scope? **Yes**
    - (b) Have you read the **ethics review guidelines** and ensured that your paper conforms to them? **Yes**
    - (c) Did you discuss any potential **negative societal impacts** of your work? **Yes**
    - (d) Did you describe the **limitations** of your work? **Yes**
2. If you are including theoretical results...
    - (a) Did you state the full set of **assumptions** of all theoretical results? **N/A**
    - (b) Did you include complete **proofs** of all theoretical results? **N/A**
3. If you ran experiments...
    - (a) Did you include the code, data, and instructions needed to **reproduce** the main experimental results (either in the supplemental material or as a URL)? **Yes**
    - (b) Did you specify all the **training details** (e.g., data splits, hyperparameters, how they were chosen)? **Yes**
    - (c) Did you report **error bars** (e.g., with respect to the random seed after running experiments multiple times)? **N/A**
    - (d) Did you include the amount of **compute** and the type of **resources** used (e.g., type of GPUs, internal cluster, or cloud provider)? **Yes**
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
    - (a) If your work uses existing assets, did you **cite** the creators? **Yes**
    - (b) Did you mention the **license** of the assets? **N/A**
    - (c) Did you include any **new assets** either in the supplemental material or as a URL? **N/A**
    - (d) Did you discuss whether and how **consent** was obtained from people whose data you're using/curating? **N/A**
    - (e) Did you discuss whether the data you are using/curating contains **personally identifiable information** or **offensive content**? **N/A**
5. If you used crowdsourcing or conducted research with human subjects… **N/A**
    - (a) Did you include the full text of instructions given to participants and screenshots, if applicable? **N/A**
        - Including this information in the supplemental material is fine, but if the main contribution of your paper involves human subjects, then we strongly encourage you to include as much detail as possible in the main paper.
    - (b) Did you describe any potential participant **risks**, with links to Institutional Review Board (IRB) approvals, if applicable? **N/A**
    - (c) Did you include the estimated hourly wage paid to participants and the **total amount spent** on participant compensation? **N/A**

Item 2: VGG16 Neural Network Architecture
Karen Simonyan and Andrew Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," arXiv (2014), https://arxiv.org/abs/1409.1556.



Item 3: ResNet50 Neural Network Architecture
Kaiming He et al., "Deep Residual Learning for Image Recognition," Conference on Computer Vision and Pattern Recognition (2015), https://arxiv.org/abs/1512.03385.


Weight Layer
ReLu
Weight Layer
F(x)
Identity (x)
F(x)
ReLu
Residual Learning Block
7*7,Conv,64
Max- Pool
1*1,Conv,64
1*1,Conv,64
1*1,Conv,256
1*1,Conv,128
3*3,Conv,128
1*1,Conv,512
1*1,Conv,256
3*3,Conv,256
1*1,Conv,1024
1*1,Conv,512
3*3,Conv,512
1*1,Conv,2048
Avg- Pool
fc, 1000
3x
4x
6x
3x