CA1 – Image Classification on CIFAR100

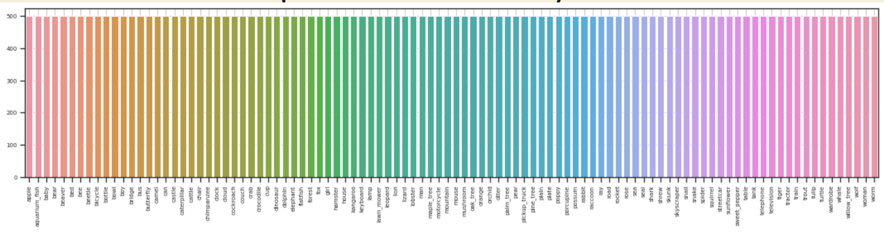
Name: Quah Johnnie
Class: DAAA/FT/2B/04
Admin No: 2007476

Contents (Jupyter Highlights)

1.0) CIFAR100 at a glance (EDA)

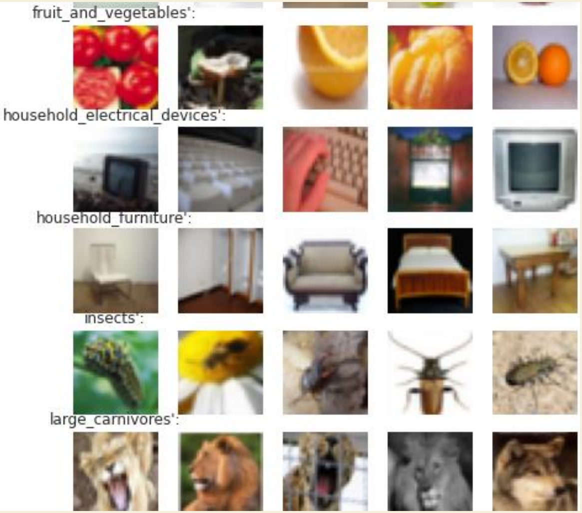
Current Highest Validation Accuracy (Fine) : NIL
Current Highest Validation Accuracy (Coarse) : NIL

Training data is uniformly distributed



Fine Labelled

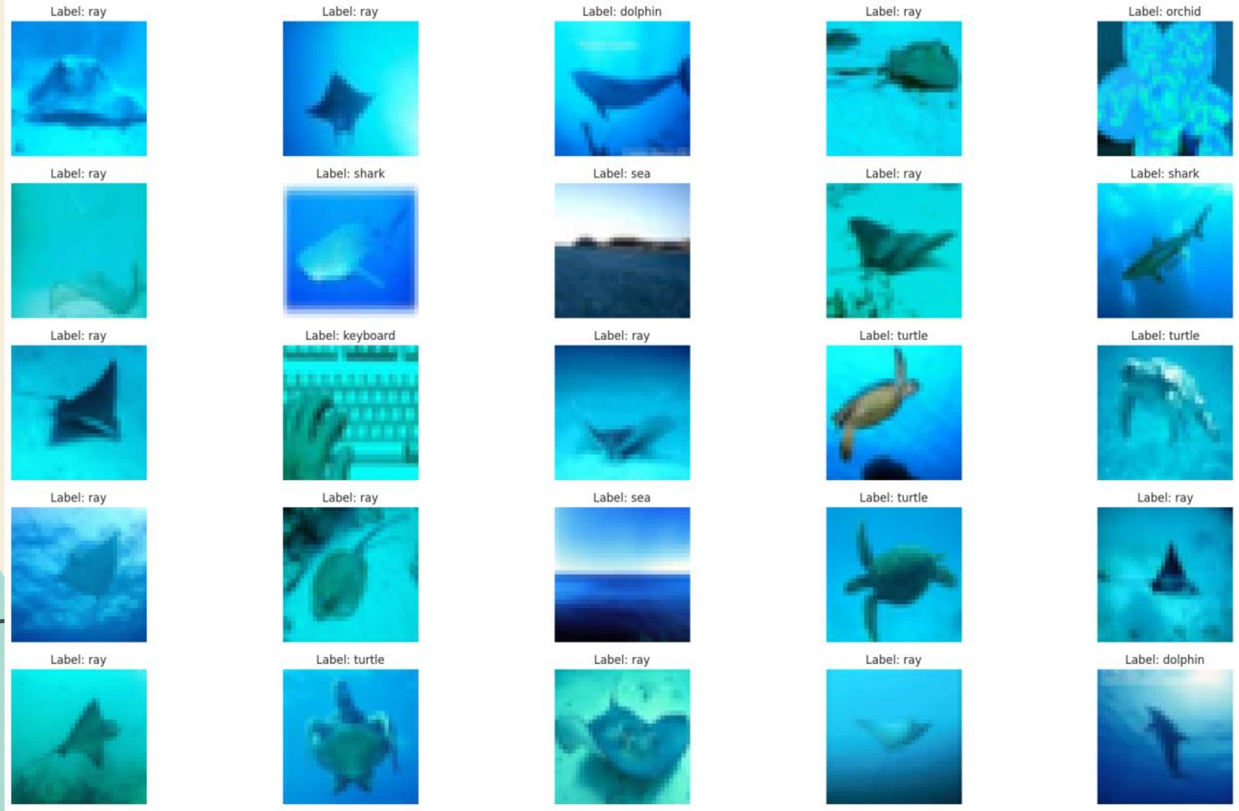
Coarse Labelled



Average of These Labels

1.0) Outliers Analysis - Autoencoder

Current Highest Validation Accuracy (Fine) : NIL
Current Highest Validation Accuracy (Coarse) : NIL



Are these outliers?
Well yes and no…

How to improve outlier analysis and look into more fine-grained outliers?

2.0) Data Augmentation/Splitting

Current Highest Validation Accuracy (Fine) : NIL
Current Highest Validation Accuracy (Coarse) : NIL

DATA SPLIT

Training Data split into:

- **40K Training**
- **10K Validation**
- **10K Testing**

DATA AUGMENTATION

Composing Augmentation Sets With Different Combinations Of:

- **ColorJitter**
- **CutMix**
- **RandomRotation**
- **RandomPerspective**
- **RandomSolarize**
- **HorizontalFlip**
- **AutoAugment (CIFAR10)**
- **AutoAugment (ImageNet)**
- **AugMix**

2.0) Visualising Augmentation

Current Highest Validation Accuracy (Fine) : NIL
Current Highest Validation Accuracy (Coarse) : NIL

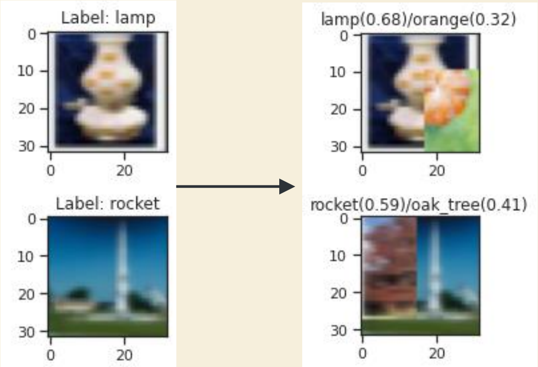
Custom: (ColorJitter, RandomRotation, RandomPerspective, RandomSolarize, HorizontalFlip)

AutoAugment (ImageNet Policy)

AutoAugment (CIFAR10 Policy)

AugMix

CutMix



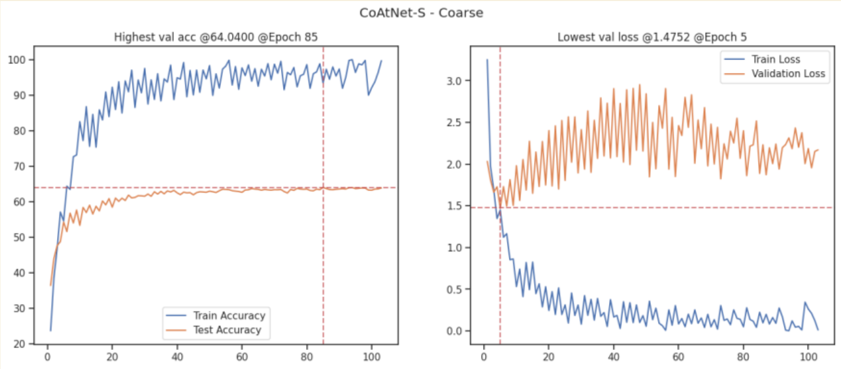
3.0) Model & Evaluation

Current Highest Validation Accuracy (Fine) : 54.66%
Current Highest Validation Accuracy (Coarse) : 68.80%

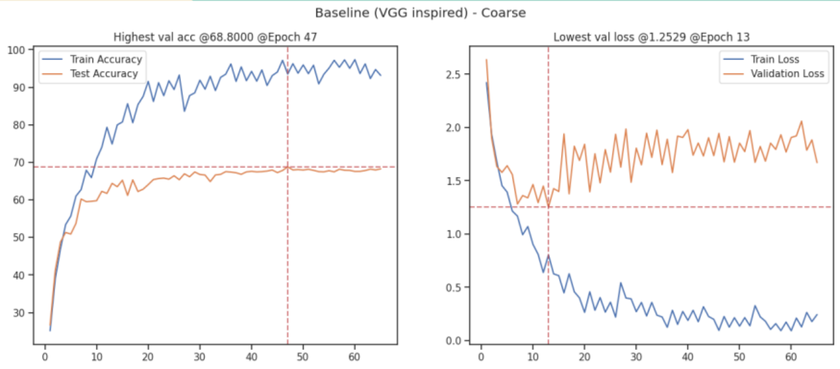
Lowest Val Loss	Highest Val Acc	Lowest Train Loss	Highest Train Acc	Epoch (Highest Val Acc)	Model Description	Parameter
1.981114	53.37	0.095882	97.58750	71	Baseline (VGG inspired)	9,198,276
2.512835	48.86	0.045856	99.10125	54	ResNet34	23,712,932
2.481324	48.89	0.034021	99.35750	49	ResNeXt-M	23,184,804
2.202396	54.66	0.004562	99.94875	80	CoAtNet-S	32,396,096
2.366959	49.75	0.030436	99.52000	72	DenseNet-M	20,013,928
1.252915	68.80	0.092217	97.36000	47	Baseline (VGG inspired) - Coarse	9,198,276
1.560987	60.92	0.026511	99.34250	46	ResNeXt-M - Coarse	23,184,804
1.475237	64.04	0.001255	99.99500	85	CoAtNet-S - Coarse	32,396,096

***Extreme overfitting… Perhaps, stronger augmentation has to be done before I see better results on more complex models…**

Best coarse labelled model – Learning curve



Best fine labelled model – Learning curve



All models trained with simple augmentation done every 2 epochs.

4.0) Model Improvements 1&2

Current Highest Validation Accuracy (Fine) : 63.85% (+9.19%)
Current Highest Validation Accuracy (Coarse) : 75.03% (6.23%

Comparing different augmentation compositions

Lowest Val Loss	Highest Val Acc	Lowest Train Loss	Epoch (Highest Val Acc)	Model Description	Parameter	Highest Train Acc
2.152134	49.41	0.077599	98	CoAtNet-AutoAugment sample:0	32,396,096	97.837500
2.839644	36.12	0.072459	90	CoAtNet-AugMix sample:0	32,396,096	96.557500
2.238827	46.17	0.626822	68	CoAtNet-ImageNet sample:0	32,396,096	79.947500
2.335603	44.15	0.113020	69	CoAtNet-CustomAugment sample:0	32,396,096	94.617500
1.687326	57.16	0.859597	67	CoAtNet-AutoAugment+CutMix sample:1	32,396,096	83.082500
1.718855	55.71	1.011111	55	CoAtNet-ImageNet+CutMix sample:1	32,396,096	79.366667
1.748327	55.29	0.916276	61	CoAtNet-AutoAugment+ImageNet sample:3	32,396,096	80.396667
1.853471	53.67	0.602639	67	CoAtNet-AutoAugment+ImageNet sample:4	32,396,096	80.353750
1.766168	56.50	0.301244	88	CoAtNet-AutoAugment sample:5	32,396,096	88.788750
1.746079	56.72	0.435966	102	CoAtNet-ImageNet sample:5	32,396,096	85.050000
2.358972	45.45	0.089996	53	CoAtNet-AugMix sample:5	32,396,096	95.651250

Lowest Val Loss	Highest Val Acc	Lowest Train Loss	Epoch (Highest Val Acc)	Model Description	Parameter	Highest Train Acc
0.939657	74.06	0.378762	103	SimpleNet-AutoAugment+CutMix sample:1 (Coarse)	9,198,276	90.644167
0.938005	73.76	0.535824	98	SimpleNet-ImageNet+CutMix sample:1 (Coarse)	9,198,276	88.567500
0.985341	72.69	0.173510	78	SimpleNet-AutoAugment sample:5 (Coarse)	9,198,276	92.812500
1.007861	71.22	0.152900	81	SimpleNet-ImageNet sample:5 (Coarse)	9,198,276	92.907500

ImageNet Augmentation + CutMix will be used as they provide the lowest variance for both fine and coarse labelled models.

Comparing different complexity of models

Lowest Val Loss	Highest Val Acc	Lowest Train Loss	Epoch (Highest Val Acc)	Model Description	Parameter	Highest Train Acc	
1.604467	59.95	0.842774	87	Simpler Custom CoAtNet	-	84.537500	~17M Parms
1.470690	63.85	0.607647	123	Simpler Custom CoAtNet2	-	90.615833	~22M Parms
1.676687	56.92	0.944950	75	Complex Custom CoAtNet	-	80.855000	~50M Parms

Lowest Val Loss	Highest Val Acc	Lowest Train Loss	Epoch (Highest Val Acc)	Model Description	Parameter	Highest Train Acc	
0.913700	75.03	0.536745	119	Simpler SimpleNet	-	87.925000	~7M Parms
0.918515	74.26	0.592309	64	Complex SimpleNet	-	84.976667	~13M Parms

4.0) Model Improvements 3&4

Current Highest Validation Accuracy (Fine) : 69.52% (+5.77%)
Current Highest Validation Accuracy (Coarse) : 79.57% (4.54%

Improvement for fine labelled model

Hyperparameter tuning from a checkpointed model:

- The checkpointed model was trained for 200 epochs with a high weight decay
- Checkpointed model reached 63.94% Val Accuracy.
- Augmentation have to be done every epoch during hyperparameter tuning.
- Heavily cut down number of epochs to train.

```
Trial #32 Finished - Search Time 27.56 Mins
Total Time Elapsed: 1002.02 Mins

Hyperparameters         |Trial Values: #32      |Best Trial Values: #20

Learning Rate           |0.000316               |0.000100
Weight Decay (L2)       |0.000000               |0.000006
Momentum                |0.920000               |0.920000
Highest Val Acc         |67.26                  |69.52
Epoch (Highest Val)     |40                     |46


Next trial:  [0.00021544346900318823, 4.6415888336127773e-07, 0.9199999999999999]


Trial ended at trial #32
```

Improvement for coarse labelled model

Transfer learning from our fine labelled model:

- Model is copied from our own hyperparameter tuned fine labelled model.
- Same X_train image data, different y_train.
- Change classification layer. Output as 20.
- Val accuracy already hit 75+% during the first 5 epochs
- Hyperparameter tuning can also be done. However, it is not done here due to time limitations. Hence, improvement not as drastic as fine labelled model.

```
- [Epoch 5/70] | Train Loss: 0.505| Train Accuracy: 88.33583333333333 | Val Loss: 0.857 | Val Accuracy: 75.79 | Est: 380.75s
- [Epoch 10/70] | Train Loss: 0.492| Train Accuracy: 90.48 | Val Loss: 0.764 | Val Accuracy: 78.96 | Est: 371.58s
- [Epoch 15/70] | Train Loss: 0.475| Train Accuracy: 90.96333333333334 | Val Loss: 0.729 | Val Accuracy: 79.09 | Est: 358.75s
- [Epoch 20/70] | Train Loss: 0.455| Train Accuracy: 91.21166666666667 | Val Loss: 0.730 | Val Accuracy: 79.33 | Est: 357.70s
- [Epoch 25/70] | Train Loss: 0.464| Train Accuracy: 90.43166666666667 | Val Loss: 0.722 | Val Accuracy: 79.41 | Est: 364.54s
- [Epoch 30/70] | Train Loss: 0.388| Train Accuracy: 92.63166666666666 | Val Loss: 0.730 | Val Accuracy: 79.11 | Est: 358.24s
- [Epoch 35/70] | Train Loss: 0.435| Train Accuracy: 91.93916666666667 | Val Loss: 0.741 | Val Accuracy: 79.01 | Est: 344.52s
- [Epoch 40/70] | Train Loss: 0.499| Train Accuracy: 89.32333333333334 | Val Loss: 0.726 | Val Accuracy: 79.21 | Est: 365.33s
EarlyStopper triggered at epochs: 42
*No improvement to validation loss and accuracy could be seen for the past 18 epochs
Highest Val Accuracy: 79.57  @ epoch 24 | Lowest Val Loss: 0.7164515900611877  @ epoch 16
```

5.0) Final Evaluation

Test Accuracy (Fine) : 69.64% (+0.12%)
Test Accuracy (Coarse) : 79.65% (+0.08%)

All training data is now used for our final model training. Evaluated done once on test data.

The 200 epochs checkpointed model is train with the best hyperparameter for 46 epochs. However, it is better to train from scratch as we have slightly more data for our final training.

Same procedure is done as before. Transfer learning from our fine labelled model, change last layer and trained for 25 epochs.

Fine labelled model

```
test_loss, test_accuracy, wrong_samples, wrong_preds, actual_preds, class_dict = eval(finalmodel_fine,criter
print('\nFinal model (fine) test_accuracy (Top-1 accuracy): ',test_accuracy)
print('\nFinal model (fine) test_loss: ',test_loss)
```

```
Final Model

Final model (fine) test_accuracy (Top-1 accuracy):  69.64

Final model (fine) test_loss:  1.30165576338768
```

Display only top 10 lowest f1 scores

```
displayLowestF1(class_dict)
```

```
otter {'precision': 0.3568181818181818, 'recall': 0.32, 'f1-score': 0.3372340425531915, 'support': 100}
lizard {'precision': 0.40082474226804123, 'recall': 0.39, 'f1-score': 0.3953299492385787, 'support': 100}
seal {'precision': 0.46025641025641024, 'recall': 0.36, 'f1-score': 0.39955056179775285, 'support': 100}
girl {'precision': 0.4657303370786517, 'recall': 0.41, 'f1-score': 0.4315343915343915, 'support': 100}
man {'precision': 0.4575824175824176, 'recall': 0.42, 'f1-score': 0.4379057591623037, 'support': 100}
woman {'precision': 0.44, 'recall': 0.44, 'f1-score': 0.4400000000000001, 'support': 100}
shrew {'precision': 0.43622641509433965, 'recall': 0.46, 'f1-score': 0.4477669902912621, 'support': 100}
mouse {'precision': 0.4456603773584906, 'recall': 0.47, 'f1-score': 0.45747572815533984, 'support': 100}
boy {'precision': 0.4670833333333333, 'recall': 0.45, 'f1-score': 0.4583673469387755, 'support': 100}
bear {'precision': 0.50987951807228917, 'recall': 0.43, 'f1-score': 0.46622950819672134, 'support': 100}
```

Coarse labelled model

```
Final model (coarse) test_accuracy (Top-1 accuracy):  79.65

Final model (coarse) test_loss:  0.7096987730026245
```

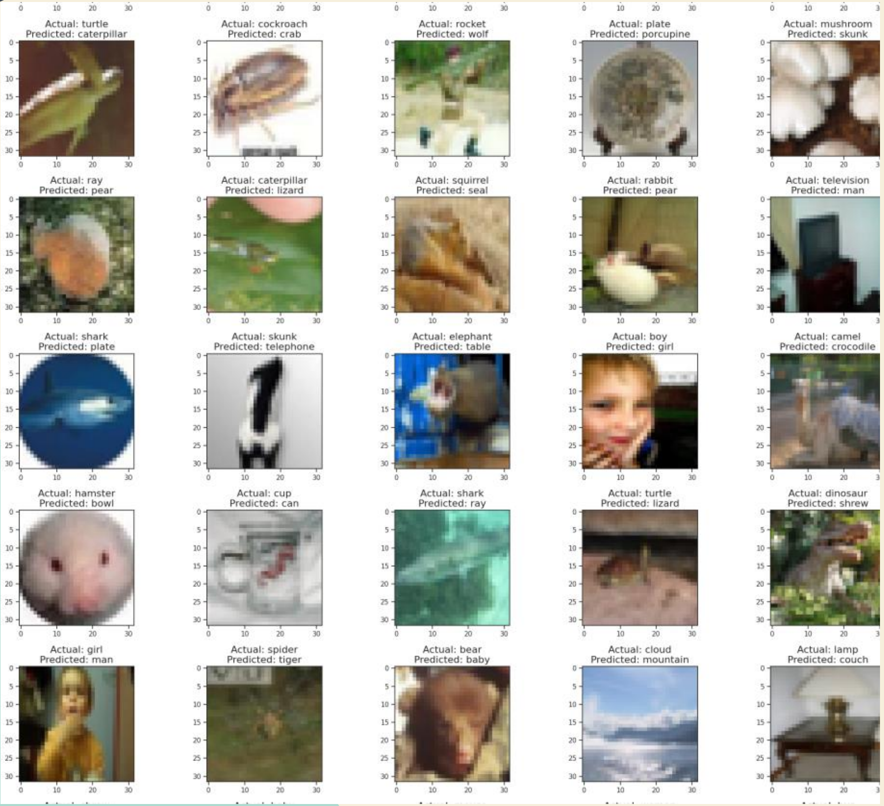
```
displayLowestF1(class_dict2,classes=20)
```

```
b'reptiles' {'precision': 0.644887983706721, 'recall': 0.634, 'f1-score': 0.6393945509586277, 'support': 500}
b'small_mammals' {'precision': 0.6481226053639846, 'recall': 0.674, 'f1-score': 0.6607827788649706, 'support': 500}
b'aquatic_mammals' {'precision': 0.671980198019802, 'recall': 0.678, 'f1-score': 0.6749751243781095, 'support': 500}
b'medium_mammals' {'precision': 0.7826696832579186, 'recall': 0.71, 'f1-score': 0.7399898089171974, 'support': 500}
b'large_carnivores' {'precision': 0.7027586206896552, 'recall': 0.804, 'f1-score': 0.7496296296296296, 'support': 500}
b'non-insect_invertebrates' {'precision': 0.8040909090909091, 'recall': 0.716, 'f1-score': 0.7572340425531914, 'support': 500}
b'large_omnivores_and_herbivores' {'precision': 0.7487072243346007, 'recall': 0.784, 'f1-score': 0.7659064327485379, 'support':
500}
b'household_electrical_devices' {'precision': 0.8704866180048662, 'recall': 0.728, 'f1-score': 0.792283205268935З, 'support': 5
00}
b'fish' {'precision': 0.8202061855670103, 'recall': 0.798, 'f1-score': 0.8089340101522843, 'support': 500}
b'insects' {'precision': 0.8826905829596412, 'recall': 0.796, 'f1-score': 0.8368710359408033, 'support': 500}
```

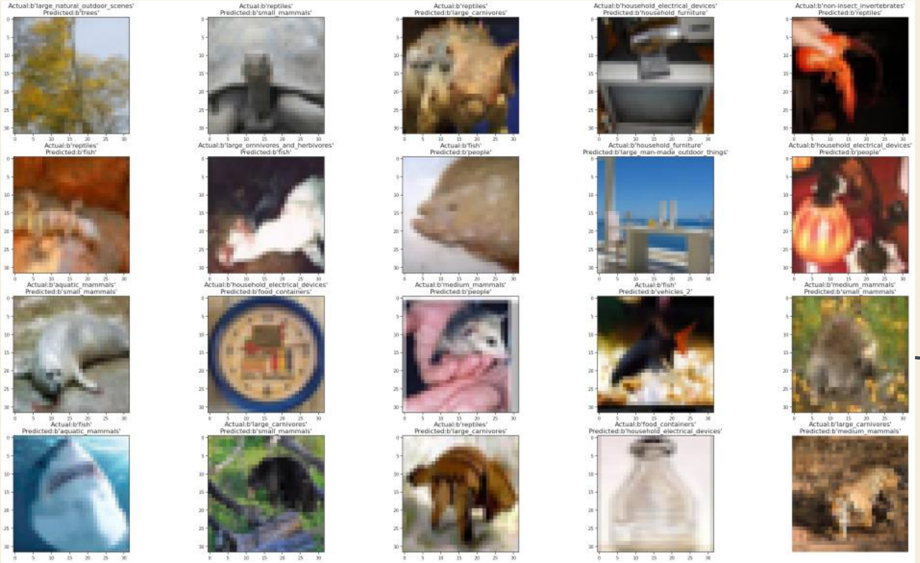
5.0) Final Evaluation – Error Analysis

Test Accuracy (Fine) : <u>69.64%</u>
Test Accuracy (Coarse) : <u>79.65%</u>

Fine label errors



Coarse label errors

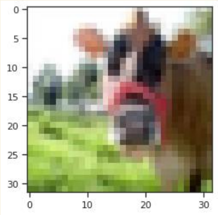


Test Accuracy (Fine) : 69.64%
Test Accuracy (Coarse) : 79.65%

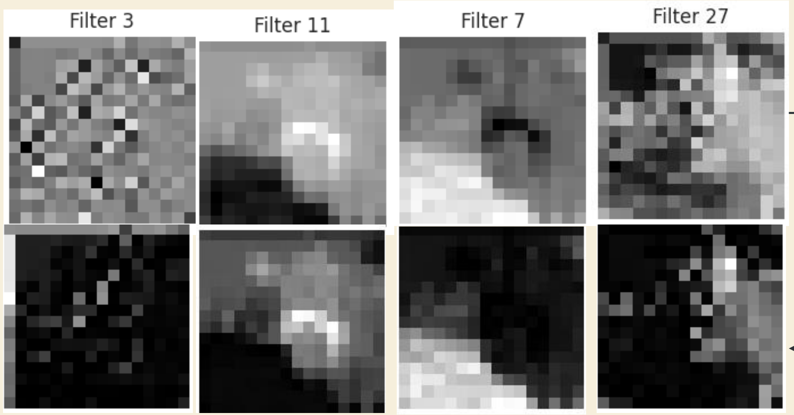
5.0) Final Evaluation – Feature Map (1st few layers)

Original (3 Channels):

Cattle



3 -> 64 Filters



Batch Normalized + Activation Function

5.0) Conclusion

Quick Conclusion:
We did EDA, then splitted our training data into train and validation data and did numerous augmentation compositions. Tried many models before settling on a custom made CoAtNet & VGGNet inspired model. We tested multiple augmentation compositions & also different data replication with augmentation methods, before doing hyperparameter tuning with model checkpoint & transfer learning to improve our model and finally did our final evaluation and analysis of our models.

Rooms for possible improvement:
Train the model from scratch (instead of checkpoint) since I have the whole training data to work with. Hyperparameter tune coarse labelled model as well. Explore other model scaling methods, most notably EfficientNet compound scaling. Explore more complex augmentation methods, FMix etc. Smaller batch size might help.
*No matter what. I feel that I have already tried and experiment a lot of ideas during this CA1 but there are still countless ways to improve a model, which is the beauty of deep learning :)

Essential lame memes

THANKS!

Fin. 完 。

CREDITS: This presentation template was created by **Slidesgo**, including icons by **Flaticon**, and infographics & images by **Freepik**

Please keep this slide for attribution