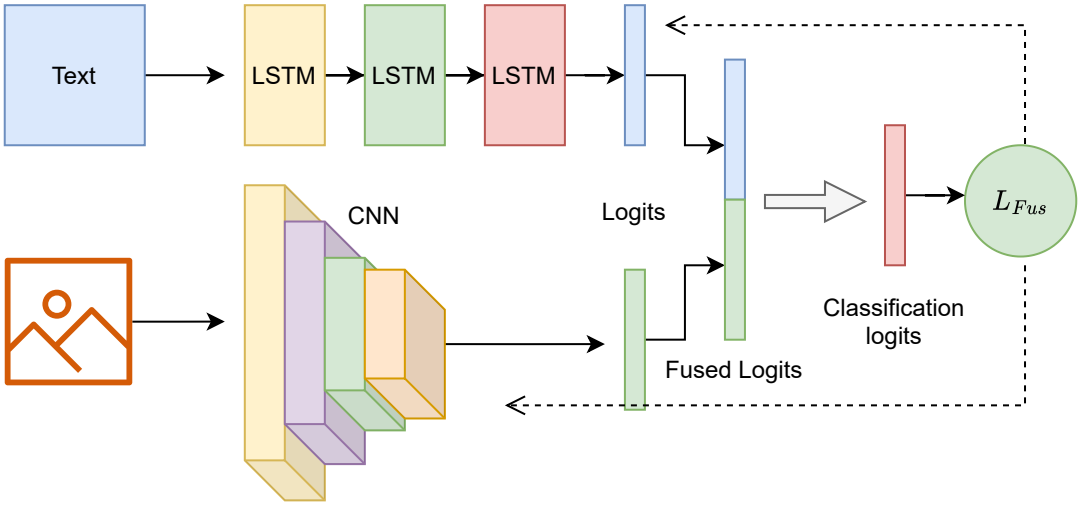
Nirbhay Sharma (B19CSE114)

PA-2 Dependable AI

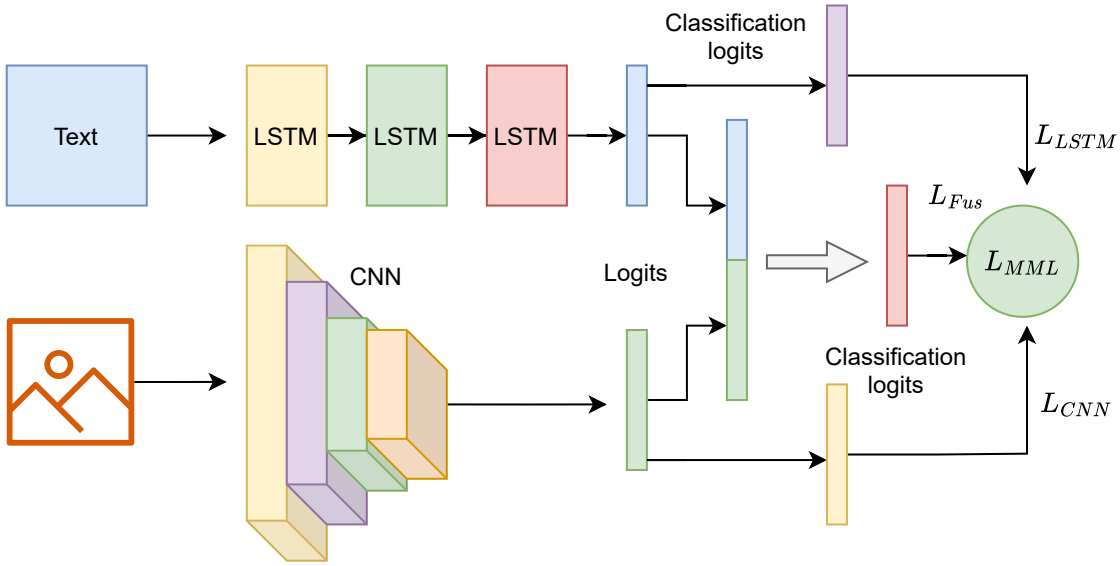
Que-1

Network Architecture (LSTMCNN)



(a) Figure representing LSTMCNN architecture for multimodal classification. The text is first passed to the LSTM network. Correspondingly the image is passed to the CNN network. The final embeddings from both the embeddings are concatenated to pass to the final classification layer.

Network Architecture (LSTMCNN + MTL)



(a) Figure representing LSTMCNN architecture for multimodal classification. The text is first passed to the LSTM network. Correspondingly the image is passed to the CNN network. The final embeddings from both the embeddings are concatenated to pass to the final classification layer. Simlarly LSTM and CNN embeddings are passed on to the classification layer to get their respective logits. The combination of all the three losses is backpropagated through the network.

We use two different networks having two different types of CNN backbones. One backbone is Resnet and another is Shufflenet. We perform variety of experiments to analyze the bias associated with each of the network. The first network is represented in Fig 1. which has only one head and the features from CNN and LSTM are concatenated and passed on to the classification layer where L_{Fus} loss is there which is cross entropy loss. The second network is a MTL network which also utilizes the individual features from CNN and LSTM apart from the fusion of their features. There are three loss in this network. First is loss from LSTM features i.e. L_{LSTM}. Second is loss from CNN features i.e. L_{CNN} and last one is loss for fusion of features L_{Fus}. All the loss functions are cross entropy loss function and the combination of all the three losses are backpropagated through the network. The final loss function L_{MML} of the MTL network is represented as follows:

$$L_{Fus} = CE(Z_{Fus}, y)$$

$$L_{LSTM} = CE(Z_{LSTM}, y)$$

$$L_{CNN} = CE(Z_{CNN}, y)$$

$$L_{MML} = L_{Fus} + \alpha_1 L_{CNN} + \alpha_2 L_{LSTM}$$

where Z_{Fus}, Z_{LSTM}, Z_{CNN} represents the final logits from Fusion layer, LSTM layer, and CNN layer. y represents the true labels of the data points.

Average metrics

To evaluate the network we analyze different networks comprising of different CNN backbones. We use two backbones Resnet and Shufflenet. We report the performance of the models in terms of Average accuracy, Precision, Recall, F1-score, AUROC, DOB (Degree of Bias).

Network	Acc	Precision	Recall	F1	AUROC	DOB
LSTMCNN + ShuffleNet	0.87	0.79	0.81	0.80	0.97	0.096
LSTMCNN + ResNet	0.84	0.80	0.77	0.78	0.96	0.128
LSTMCNN + ShflNet + MTL	0.86	0.66	0.72	0.69	0.97	0.364
LSTMCNN + ResNet + MTL	0.90	0.82	0.82	0.82	0.98	0.088

As presented above the models perform well on the dataset. However a comparison in terms of bias can be observed by using DOB bias metrics. The model having Resnet and trained using MTL learning is showing a low bias as compared to the other. On the other hand the Shufflenet + MTL does not seems to work well in terms of bias in the network. The networks without MTL are also showing promising results. Moreover the LSTMCNN + shufflenet is showing good performance in terms of bias. The networks performs well in terms of other metrics as well. The LSTMCNN + Resnet + MTL is showing a better performance in comparison to other methods in terms of Precision, Recall, F1-Score, and AUROC. Therefore, the least biased among them is LSTMCNN + Resnet + MTL in terms of DOB and other meaningful metrics.

Accuracy of multiple heads in LSTMCNN

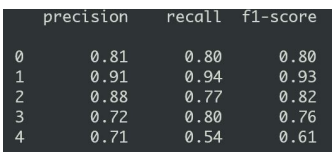
Next, for MTL network we evaluate the individual heads performance. Additionally, we evaluate the performance in terms of ensemble of the heads. We take the ensemble by taking the average of the logits from each head.

Network	Fusion	CNN	LSTM	Ensemble
LSTMCNN + ShflNet + MTL	0.856	0.566	0.856	0.858
LSTMCNN + ResNet + MTL	0.891	0.689	0.902	0.901

All the heads are performing individually good. The ensemble of them is also showing promising results and the networks learns better features by backpropagating the combination of three loss values which are represented by L_{MML}.

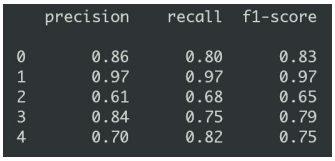
class wise Precision, Recall, Fscore

To evaluate more upon the bias. We evaluate class wise precision, recall, F1-score for each of the networks.



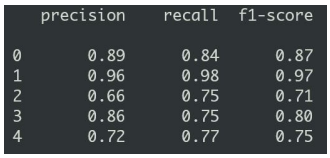
	precision	recall	f1-score
0	0.81	0.80	0.80
1	0.91	0.94	0.93
2	0.88	0.77	0.82
3	0.72	0.80	0.76
4	0.71	0.54	0.61

(a) LSTMCNN_Res



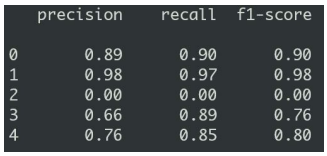
	precision	recall	f1-score
0	0.86	0.80	0.83
1	0.97	0.97	0.97
2	0.61	0.68	0.65
3	0.84	0.75	0.79
4	0.70	0.82	0.75

(b) LSTMCNN_Shfl



	precision	recall	f1-score
0	0.89	0.84	0.87
1	0.96	0.98	0.97
2	0.66	0.75	0.71
3	0.86	0.75	0.80
4	0.72	0.77	0.75

(c) LSTMCNN_Res_MTL



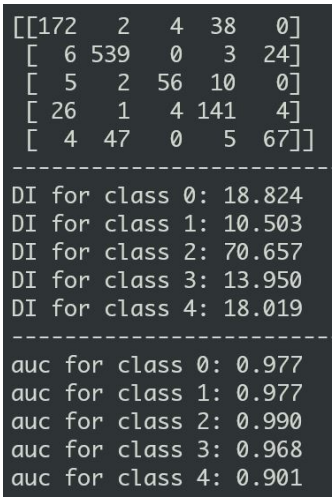
	precision	recall	f1-score
0	0.89	0.90	0.90
1	0.98	0.97	0.98
2	0.00	0.00	0.00
3	0.66	0.89	0.76
4	0.76	0.85	0.80

(d) LSTMCNN_Shfl_MTL

As shown in the above illustrations, each model is biased towards different classes. For example considering precision, each models shows different values of precision for each class which implies that they have different kind of learned representation and correspondingly they have different kind of biases in their network.

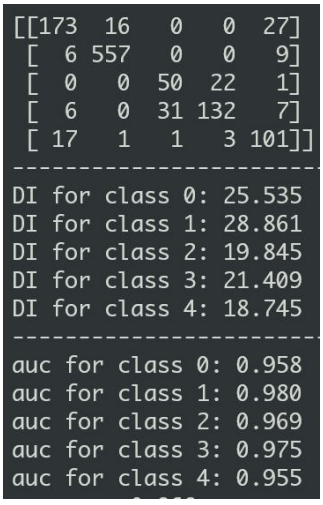
Class wise AUROC, DI, and confusion matrix

We also evaluate bias in terms of class wise AUROC, class wise DI (Disparate Impact), and confusion matrix.



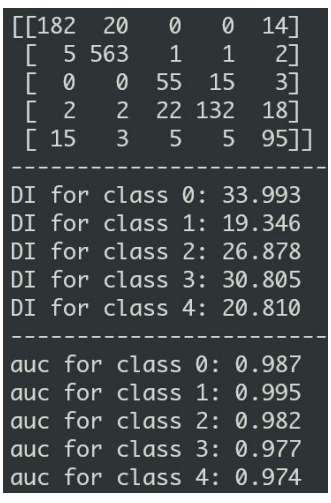
```
[[172   2   4  38   0]
 [  6 539   0   3  24]
 [  5   2  56  10   0]
 [ 26   1   4 141   4]
 [  4  47   0   5  67]]
------------------------
DI for class 0: 18.824
DI for class 1: 10.503
DI for class 2: 70.657
DI for class 3: 13.950
DI for class 4: 18.019
------------------------
auc for class 0: 0.977
auc for class 1: 0.977
auc for class 2: 0.990
auc for class 3: 0.968
auc for class 4: 0.901
```

(a) LSTMCNN_Res



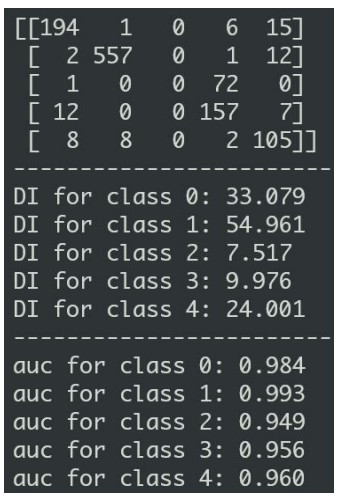
```
[[173  16   0   0  27]
 [  6 557   0   0   9]
 [  0   0  50  22   1]
 [  6   0  31 132   7]
 [ 17   1   1   3 101]]
------------------------
DI for class 0: 25.535
DI for class 1: 28.861
DI for class 2: 19.845
DI for class 3: 21.409
DI for class 4: 18.745
------------------------
auc for class 0: 0.958
auc for class 1: 0.980
auc for class 2: 0.969
auc for class 3: 0.975
auc for class 4: 0.955
```

(b) LSTMCNN_Shfl



```
[[182  20   0   0  14]
 [  5 563   1   1   2]
 [  0   0  55  15   3]
 [  2   2  22 132  18]
 [ 15   3   5   5  95]]
------------------------
DI for class 0: 33.993
DI for class 1: 19.346
DI for class 2: 26.878
DI for class 3: 30.805
DI for class 4: 20.810
------------------------
auc for class 0: 0.987
auc for class 1: 0.995
auc for class 2: 0.982
auc for class 3: 0.977
auc for class 4: 0.974
```

(c) LSTMCNN_Res_MTL



```
[[194   1   0   6  15]
 [  2 557   0   1  12]
 [  1   0   0  72   0]
 [ 12   0   0 157   7]
 [  8   8   0   2 105]]
------------------------
DI for class 0: 33.079
DI for class 1: 54.961
DI for class 2: 7.517
DI for class 3: 9.976
DI for class 4: 24.001
------------------------
auc for class 0: 0.984
auc for class 1: 0.993
auc for class 2: 0.949
auc for class 3: 0.956
auc for class 4: 0.960
```

(d) LSTMCNN_Shfl_MTL

Here also different models shows different values for each class. For example we can analyze DI metrics for each of the models class wise.

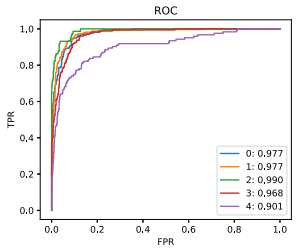
- For class 0 LSTMCNN_Res_MTL is showing higher value which implies that among all the models it has least bias towards class 0.

- For class 1 LSTMCNN_Shfl_MTL is showing higher value which implies that among all the models it has least bias towards class 1.
- For class 2 LSTMCNN_Res is showing higher value which implies that among all the models it has least bias towards class 2.
- For class 3 LSTMCNN_Res_MTL is showing higher value which implies that among all the models it has least bias towards class 3.
- For class 4 LSTMCNN_Shfl_MTL is showing higher value which implies that among all the models it has least bias towards class 4.

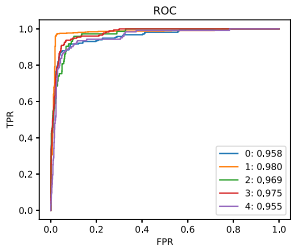
This shows that different models shows their bias towards different classes.

Roc_curves

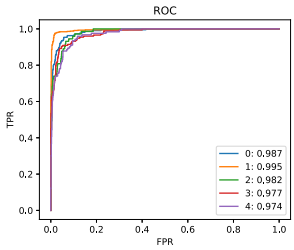
We also evaluate the performance using ROC curves. ROC curves are the plots between TPR (True Positive Rate) & FPR (Flase Positive Rate) values and are shown below.



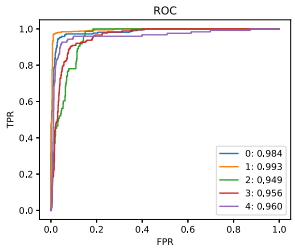
(a) LSTMCNN_Res



(b) LSTMCNN_Shfl

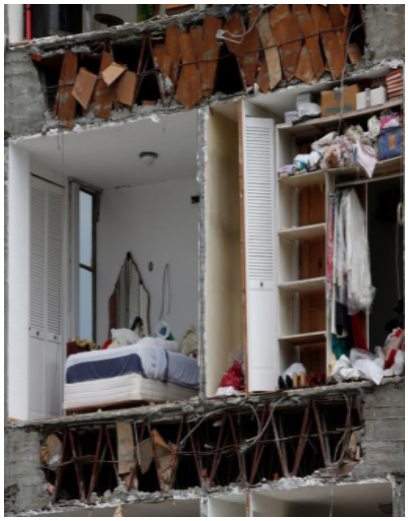


(c) LSTMCNN_Res_MTL



(d) LSTMCNN_Shfl_MTL

GradCAM & GradCAM++ Analysis




"But as for me and my house,
we will serve the Lord."


"But as for me and my house,
we will serve the Lord."


"But as for me and my house,
we will serve the Lord."


CAL FIRE


CAL FIRE


CAL FIRE

(a) Original Images (b) GradCAM (c) GradCAM++

Dependency of bias on Network

Yes, We agree that the bias towards a particular group or class is dependent on the network architecture. As the above experiments depicts that there are four different architectures LSTMCNN+Resnet, LSTMCNN+Shufflenet, LSTMCNN+Resnet+MTL, LSTMCNN+Shufflenet+MTL. Each models has their own specific bias towards a particular class. We can observe it from the metrics values reported above in terms of Precision, Recall, F1-score, AUROC, DI etc. Each network learns different features from the dataset and based on that it tries to develop its understanding of the features which in return generates a variety of bias towards a specific group or a specific class.

Can explainability be measured in a quantitative way ?

Yes, we can quantify explainability of a neural network to some extend. Some of the metrics are also proposed in the literature for example SHAP value. This assigns a score based on which feature is important for prediction. Therefore, we can quantify importance of the different features that are being used for output. Similar scores we can provide to each layers via Layer-Wise Relevance Propagation score. Therefore, we can also get a good idea to what layers are most important in prediction of the certain output.

Another metrics for bias measurement

We propose another metrics that can be used to measure bias in the model is standard deviation of the precision values. This metrics will tell how far it can deviate in terms of precision. The results of each model on this metrics is as follows:

Network	Precision std
LSTMCNN + ShuffleNet	0.127
LSTMCNN + ResNet	0.083
LSTMCNN + ShflNet + MTL	0.348
LSTMCNN + ResNet + MTL	0.110

As analyzed using the above metrics LSTMCNN + Resnet shows less bias than others but LSTMCNN + Resnet + MTL is not very far away from the LSTMCNN + Resnet model. But as it is also clear that LSTMCNN + Shfl+ MTL shows a very high bias towards a specific group.

Que-2

Chossing the paper for Image classification

For this task we choose Pyramidnet paper [1]. This paper talks about pyramidnet architecture and its implementation. We run it for CIFAR10 dataset which is one the datasets used in the paper.

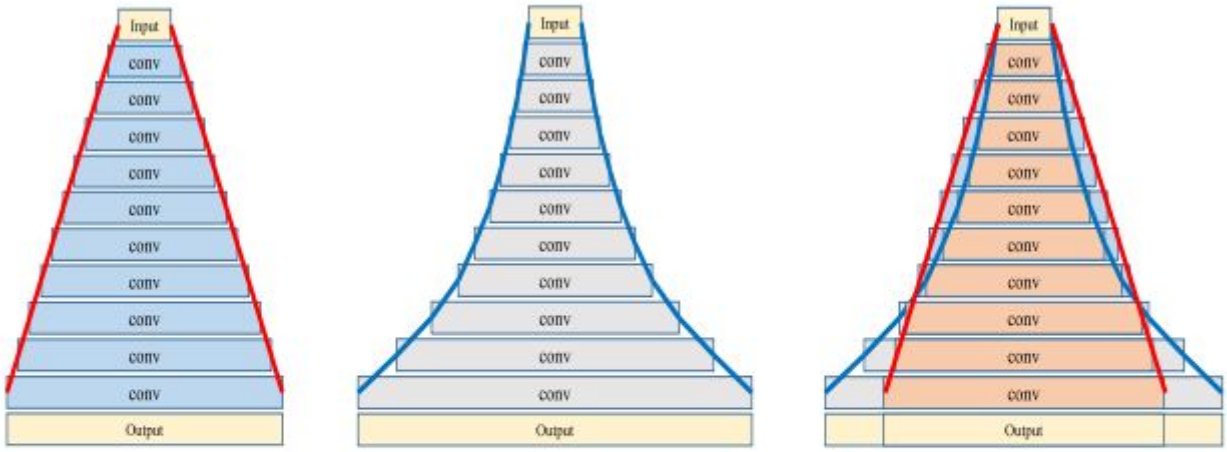


Figure 2. Visual illustrations of (a) additive PyramidNet, (b) multiplicative PyramidNet, and (c) a comparison of (a) and (b).

Chossing paper for Bias Mitigation

For bias mitigation we use a paper [2] titled as: **"Permuted AdaIN: Reducing the Bias Towards Global Statistics in Image Classification"**. The paper talks about textural bias and proposed a normalization layer to mitigate the bias in the network.

The algorithm (PAdaIN) is represented mathematically as follows.

First defining Instance Normalization (IN). Consider a feature map $x \in R^{N \times C \times H \times W}$. The Instance Norm is defined as:

$$IN(x) = \gamma(\frac{x - \mu(x)}{\sigma(x)}) + \beta$$

where $\mu(x)$ and $\sigma(x)$ denotes the mean and standard deviation in the space $R^{N \times C}$. Therefore, defined as:

$$\mu(x) = \frac{1}{HW} \sum_{h \in [H]} \sum_{w \in [W]} x$$

$$\sigma(x) = \sqrt{\frac{1}{HW} \sum_{h \in [H]} \sum_{w \in [W]} (x - \mu(x))^2 + \epsilon}$$

where we define $[W]$ as an interval of integers ranging from one to W, i.e. $[1, W]$

Then AdaIN is defined over two feature maps. It first tries to normalize a then it tries to scale it up by the statistics of b. Therewore it is defined as:

$$AdaIN(a, b) = \sigma(b)(\frac{a - \mu(a)}{\sigma(a)}) + \mu(b)$$

where $\mu(a)$ and $\sigma(a)$ are mean and standard deviation of a respectively.

Finally we define permuted AdaIN as follows. Let $x \in R^{N \times C \times H \times W}$ be a feature activation map having components as $x = x_1, x_2, \ldots, x_N$ and let $\pi(x) = [x_{\pi(1)}, x_{\pi(2)}, \ldots, x_{\pi(N)}]$ be the permuted feature map after applying a permutation π. On this permutation and the original feature map we apply $PIN^{\pi}(x)$ operation as follows

$$PIN^{\pi}(x) = AdaIN(x_i, x_{\pi(i)})$$

Finally the permuted AdaIN operation on a feature map x is as follows

$$PAdaIN(x) = (PIN^{\pi}(x_1), PIN^{\pi}(x_2), \ldots, PIN^{\pi}(x_N))$$

The authors have already mentioned in the paper that the PAdaIN is applied after the convolution layer and before the batch-norm layer. The batch-norm layers after the application of PAdaIN does not undo the effect of it. Rather it scales up the output using batch-wise statistics.

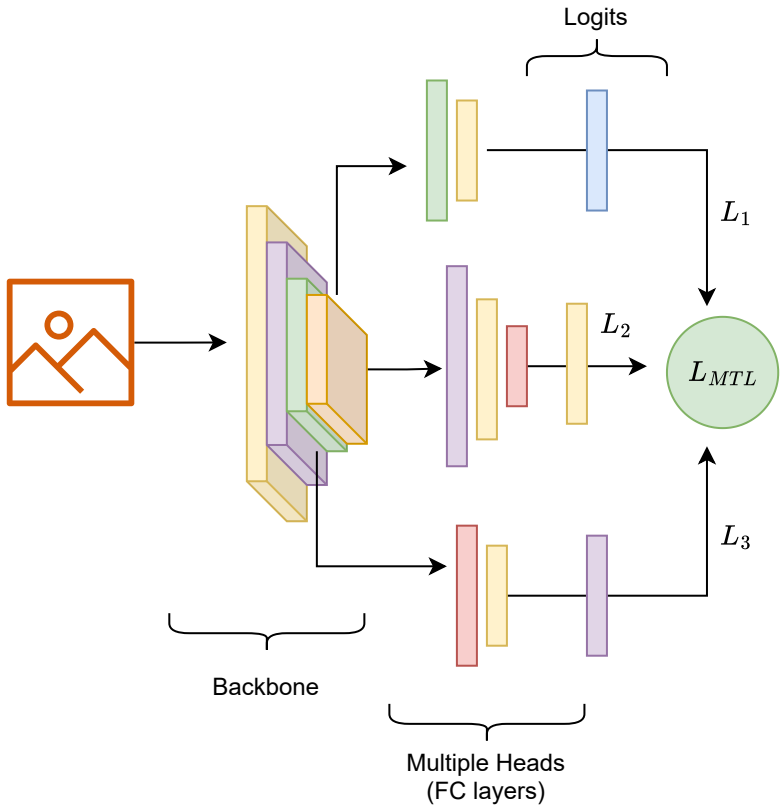
Data techniques for bias mitigation

Here we apply data augmentations for improving the performance of the architecture. We apply RandomHorizontalFlip and RandomRotation to the data to make it more robust and bias free.

Model and Loss techniques for bias mitigation

as represented below we apply MTL approach to mitigate the bias. We use three classification heads having three different loss functions at each head. Each head has different number of neurons to incorporate variations and learn different features. The three losses L_1, L_2, and L_3 are CrossEntropyLoss, KLDivergence Loss, L2Loss functions. The entire loss function is the combination of all these individual heads losses i.e. $L_{MTL} = L_1 + L_2 + L_3$. The combined loss L_{MTL} is backpropagated and trains the network.

Network Architecture (MTL)



(a) Figure representing MTL architecture for image classification. The image is first passed through the backbone network which is commom for all the heads. The output features are then passed on to the different head continaing different number of neurons to get their respective logits. Three different losses are used at each head and the combination of the losses is backpropagated.

Average metrics

For experiments, we analyze different combination of techniques with pyramidnet. We see the performance of pyramidnet alone, pyramidnet combined with augmentation, pyramidnet combined with PAdaIN, pyramidnet MTL network. We report the performance of the models in terms of Average accuracy, Precision, Recall, F1-score, AUROC, DOB (Degree of Bias).

Network	Avg Acc	Avg Precision	Avg Recall	Avg F1	Avg AUROC	DOB
Pyr	0.782	0.79	0.78	0.78	0.972	0.115
Pyr + Augment	0.864	0.86	0.86	0.86	0.989	0.087
Pyr + PAdaIN + Augment	0.868	0.87	0.87	0.87	0.989	0.075
Pyr + MTL + Augment	0.867	0.87	0.87	0.87	0.990	0.069

As presented above the different approachs works differently on the pyramidnet architecture. The accuracy is highest with Pyr+ PAdaIN approach. However, pyr + MTL is also showing comparable performance to PAdaIN technique. The data augmentation technique is also showing good improvement as compared to the one on which the augmentation is not applied (that shows least accuracy). Similar trends are visible with other metrics as well such as precision, recall, F1-score, AUROC etc. For AUROC pyr + MTL works slightly better than pyr + PAdaIN technique.

comparison of methods in terms of bias

In terms of bias metrics (DOB) we observe that bias is least for Pyr + MTL network and is comparable to pyr + PAdaIN technique. The bias for isolated pyramidnet is the most since it is not equipped with techniques like augmentation, MTL etc.

performance before and after bias mitigation

The bias mitigation techniques for sure helps in mitigation of bias in the network. We apply three different techniques to mitigate bias. Augmentation, PAdaIN, MTL techniques are used. after application of each technique the bias is reduced in the network as shown in DOB metrics. PAdaIN also works very well in reducing the bias. Apart from that MTL approach is showing highly comparable performance to PAdaIN approach. So the general trend is decrease in bias as per using different bias mitigation techniques.

Accuracy of multiple heads in Pyr + MTL + Augment

Next, we evaluate the accuracy of the individual heads and their ensemble by taking the average logits and applying softmax to that. The reports are available below. All the heads performs equally well and their ensemble is also showing a descent performance and even greater performance than one of the heads.

Network	Head1	Head2	Head3	Ensemble
Pyr + MTL + Augment	0.868	0.868	0.862	0.867

class wise Precision, Recall, Fscore

We also compute bias using class wise metrics such as precision, recall, and F1-score. different techniques combined with pyramidnet shows different values for these metrics for each class. Hence we can compare the models relative to each other by measuring their values classwise only.

	precision	recall	f1-score
0	0.71	0.90	0.79
1	0.87	0.93	0.90
2	0.79	0.59	0.67
3	0.61	0.59	0.60
4	0.75	0.74	0.75
5	0.64	0.74	0.69
6	0.75	0.89	0.81
7	0.90	0.73	0.81
8	0.92	0.83	0.87
9	0.93	0.82	0.87

(a) Pyr

	precision	recall	f1-score
0	0.87	0.89	0.88
1	0.94	0.94	0.94
2	0.77	0.83	0.80
3	0.81	0.63	0.71
4	0.81	0.90	0.85
5	0.78	0.81	0.79
6	0.87	0.91	0.89
7	0.92	0.88	0.90
8	0.94	0.92	0.93
9	0.93	0.93	0.93

(b) Pyr + Aug

	precision	recall	f1-score
0	0.88	0.86	0.87
1	0.94	0.94	0.94
2	0.82	0.82	0.82
3	0.77	0.69	0.73
4	0.84	0.87	0.85
5	0.80	0.80	0.80
6	0.89	0.91	0.90
7	0.88	0.92	0.90
8	0.91	0.94	0.93
9	0.94	0.91	0.93

(c) Pyr + Aug + PAdaIN

	precision	recall	f1-score
0	0.85	0.90	0.88
1	0.93	0.95	0.94
2	0.80	0.84	0.82
3	0.77	0.70	0.74
4	0.83	0.89	0.86
5	0.78	0.82	0.80
6	0.91	0.90	0.91
7	0.95	0.83	0.89
8	0.92	0.94	0.93
9	0.94	0.91	0.92

(d) Pyr + Aug + MTL

Class wise DI

We also report DI (Disparate Impace) class wise of each technique. Since it is reported class wise. We can compare the techniques class wise only. So no network here is absolute best. For instance.

- For class 0, Pyr + PAdaIN is showing less bias
- For class 1, pyr + Augment is showing less bias
- For class 2, pyr is showing less bias
- For class 3, pyr + Aug + MTL is showing less bias
- For class 4, pyr + PAdaIN is showing less bias
- for class 5, pyr + PAdaIN is showing less bias

- for class 6, pyr + Aug + MTL is showing less bias
- for class 7, pyr + Aug + MTL is showing less bias
- for class 8, pyr + Augment is showing less bias
- for class 9, pyr + Aug + MTL is showing less bias

If we compare the methods Pyr + MTL and Pyr + PAdaIN shows quite similar and comparable performance.

```
DI for class 0: 19.859
DI for class 1: 54.381
DI for class 2: 28.052
DI for class 3: 12.973
DI for class 4: 25.555
DI for class 5: 15.116
DI for class 6: 25.050
DI for class 7: 66.179
DI for class 8: 86.005
DI for class 9: 86.404
```

(a) Pyr

```
DI for class 0: 41.431
DI for class 1: 106.470
DI for class 2: 22.229
DI for class 3: 20.066
DI for class 4: 30.722
DI for class 5: 22.054
DI for class 6: 46.905
DI for class 7: 62.314
DI for class 8: 96.725
DI for class 9: 78.410
```

(b) Pyr + Aug

```
DI for class 0: 53.335
DI for class 1: 106.157
DI for class 2: 27.212
DI for class 3: 18.464
DI for class 4: 37.844
DI for class 5: 23.428
DI for class 6: 52.916
DI for class 7: 51.901
DI for class 8: 70.143
DI for class 9: 105.010
```

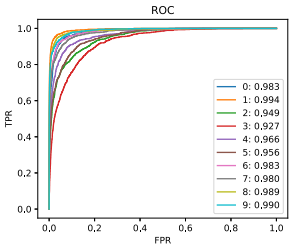
(c) Pyr + Aug + PAdaIN

```
DI for class 0: 41.326
DI for class 1: 100.765
DI for class 2: 27.852
DI for class 3: 21.614
DI for class 4: 36.061
DI for class 5: 23.390
DI for class 6: 66.324
DI for class 7: 113.374
DI for class 8: 80.273
DI for class 9: 112.984
```

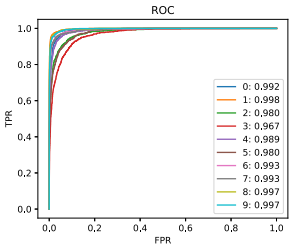
(d) Pyr + Aug + MTL

ROC curves

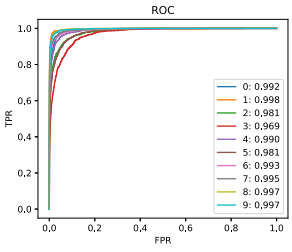
We also report ROC curves to see the performance of the models visually.



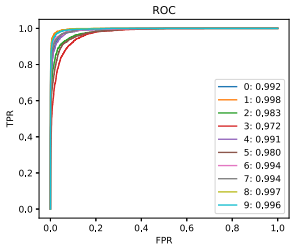
(a) Pyr



(b) Pyr + Aug



(c) Pyr + Aug + PAdaIN



(d) Pyr + Aug + MTL

References

[1] Han, Dongyoon, Jiwhan Kim, and Junmo Kim. "Deep pyramidal residual networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.

[2] Nuriel, Oren, Sagie Benaim, and Lior Wolf. "Permuted adain: Reducing the bias towards global statistics in image classification." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021.