Università degli Studi di Padova

ICT for Internet and Multimedia

COMPUTER VISION

PROJECT REPORT

WEATHER CLASSIFICATION

Written By

Beliz GUNAY
2080284

Supervisor
Prof. Pietro ZANUTTIGH

Contents

1 Introduction

In this paper, we examine weather classification using deep learning techniques. Our proposed approaches involve deep learning using a convolutional neural network (CNN). The validity of our approach lies in the increasing demand for accurate and effective image classification models in various fields.

We can also state that we have improvements with the BUN model because the BUN model gives better results in a shorter time. Our most recent model, BUNTT, has performed similarly to the first model. Unlike others, we tried different approaches and different combinations with our BUNTT model. The most effective combination for our model can be found by using instance normalization in conjunction with the cbam block, which produced better results.

In short, we use batch and instance normalization, regularization techniques and dense layer to improve the performance and generalization ability of the models. These changes help prevent overfitting and increase the robustness of the model.[1][2][3]

2 Models

In this study, we are required to do multi-class classification task of weather images based on neural network algorithms and find the 'best' classified model between them, calculate the accuracy value and evaluate the model's performance using additional metrics such as precision, recall, and f1 score to provide a comprehensive understanding of its effectiveness in classifying weather images.Furthermore, we will conduct a thorough analysis of the confusion matrix to identify specific patterns of misclassification and assess the model's strengths and weaknesses across different weather classes.

2.1 DenseNet121

We leverage the power of transfer learning by utilizing the pre-trained DenseNet 121 model. This enables us to benefit from the knowledge and features learned from a large dataset and adapt it to our specific image classification task. It loads the pre-trained DenseNet121 model, freezes its weights, and adds custom layers on top. The model is compiled and trained using various callbacks. Training history and evaluation results are displayed.

The classification head consisted of batch normalization, a dense layer with units, and a softmax activation layer for the output. The model was compiled with the Adam optimizer and categorical cross-entropy loss function.

Input Layer : The input is a 3D tensor of shape (128, 128, 3).Then we follow the steps of Zero Padding, Convolution Blocks(5 blocks), Batch Normalization, Activation (uses ReLU activation function), 2D Max Pooling, Flattening and Fully Connected Layers respectively.

- **Total params:** 7305285
- **Trainable params:** 265733
- **Non-trainable params:** 7039552

2.2 BUN

The second model, referred to as Model BUN, is a Custom CNN. It consists of several convolutional layers with batch normalization, max pooling, and a fully connected layer for classification. The model was trained compiled using the same process as the first model.The training history and evaluation results are displayed.

- **Total params:** 6432357
- **Trainable params:** 6428293
- **Non-trainable params:** 4064

2.3 BUNTT

The third model, named BUNTT, utilized a custom made block called CBAM Block, inspired by Convolutional Block Attention Module. The CBAM block consists of two attention modules.These are Channel Attention Module (CAM) and the Spatial Attention Module (SAM).[4]

Similar to the previous models, it was trained using the same process.However, BUNTT is different from other models: we use Instance Normalization instead of Batch Normalization, we use CBAM block instead of SE block.

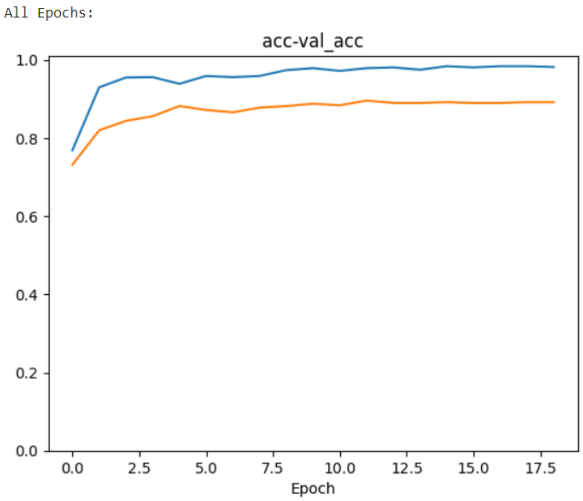
- **Total params:** 7292605
- **Trainable params:** 7292605
- **Non-trainable params:** 0

During training, several callbacks were used, including ReduceLROnPlateau, EarlyStopping, TensorBoard,CSVLogger, and ModelCheckpoint. These callbacks helped monitor the validation accuracy and loss, reduce the learning rate if no significant improvement was observed, early stop the training if the validation accuracy did not improve, and save the best model based on validation accuracy.

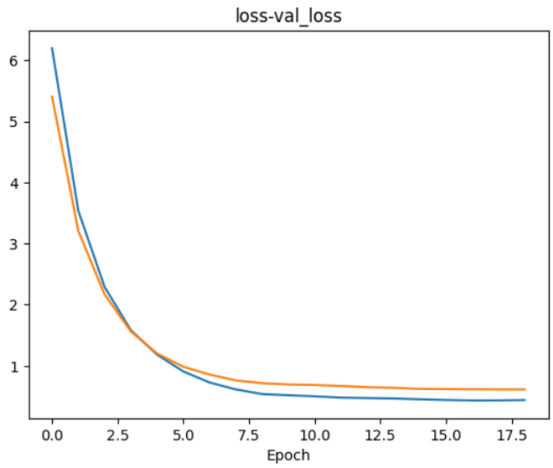
3 Results

3.1 Using DenseNet 121 Model

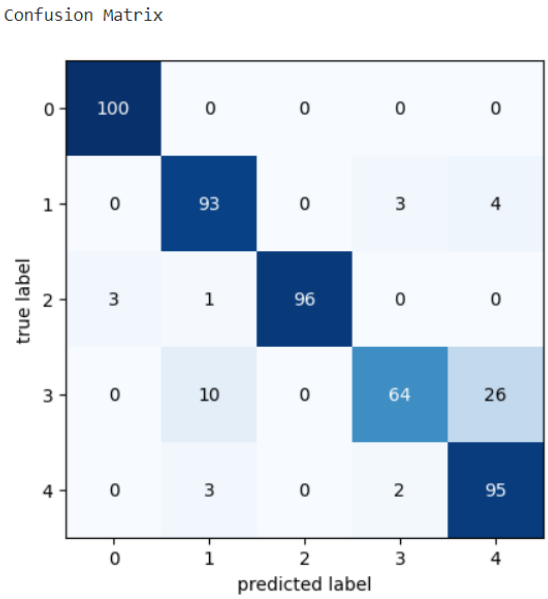
Firstly, we used the DenseNet 121 model and training validation accuracy and loss, classification output and confusion matrix are shown in separate figures below.



(a) DenseNet 121 training and validation accuracy



(b) DenseNet 121 training and validation loss



(c) DenseNet 121 confusion matrix

Training Accuracy	0.9819999933242798
Training Loss	0.43877026438713074
Validation Accuracy	0.8920000195503235
Validation Loss	0.6122590899467468

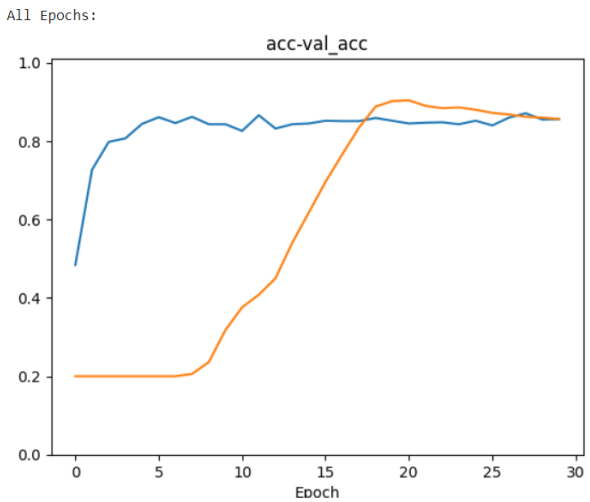
Table 1: Last Epoch Parameters

	precision	recall	f1-score
clear	0.9709	1.0000	0.9852
fog	0.8692	0.9300	0.8986
night	1.0000	0.9600	0.9796
rain	0.9275	0.6400	0.7574
snow	0.7600	0.9500	0.8444
accuracy	-	-	0.8960
macro avg	0.9055	0.8960	0.8930
weighted avg	0.9055	0.8960	0.8930

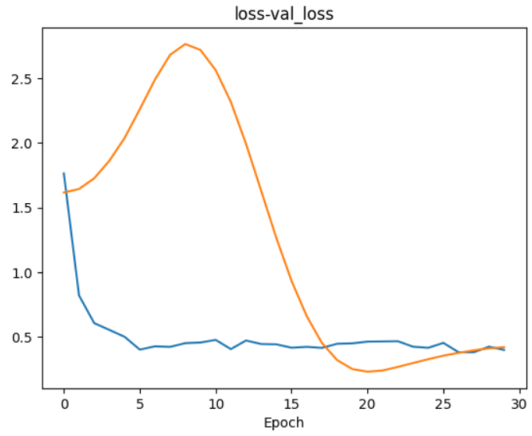
Table 2: DenseNet 121 Classification Output

3.2 Using BUN Model

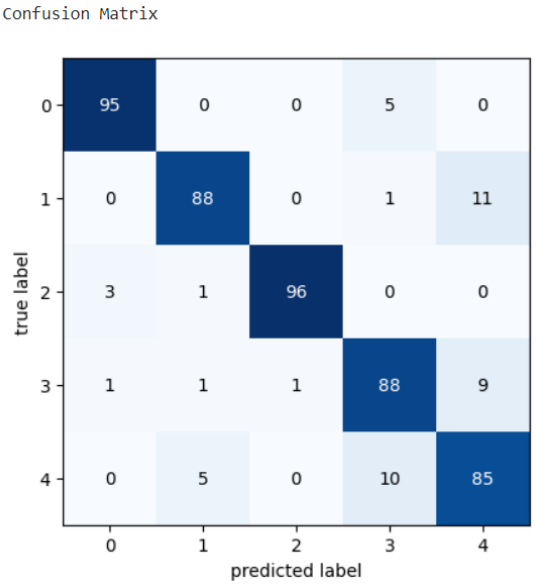
Secondly, we used the BUN model and training validation accuracy and loss, classification output and confusion matrix are shown in separate figures below.



(a) BUN training and validation accuracy



(b) BUN training and validation loss



(a) BUN confusion matrix

Training Accuracy	0.8560000061988831
Training Loss	0.39910122752189636
Validation Accuracy	0.8560000061988831
Validation Loss	0.4204504191875458

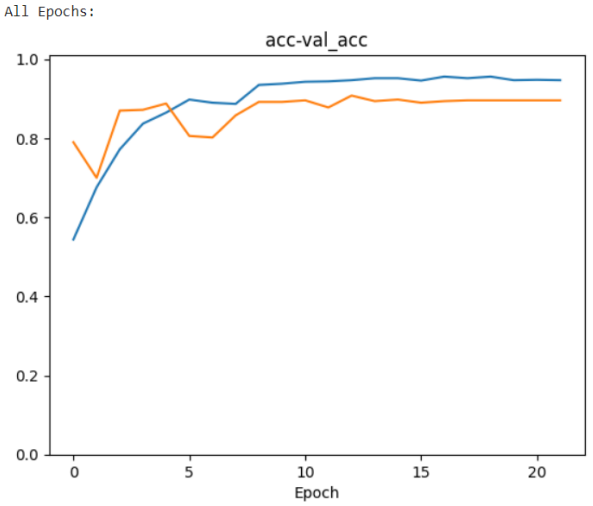
Table 3: Last Epoch Parameters

	precision	recall	f1-score
clear	0.9596	0.9500	0.9548
fog	0.9263	0.8800	0.9026
night	0.9897	0.9600	0.9746
rain	0.8462	0.8800	0.8627
snow	0.8095	0.8500	0.8293
accuracy	-	-	0.9040
macro avg	0.9063	0.9040	0.9048
weighted avg	0.9063	0.9040	0.9048

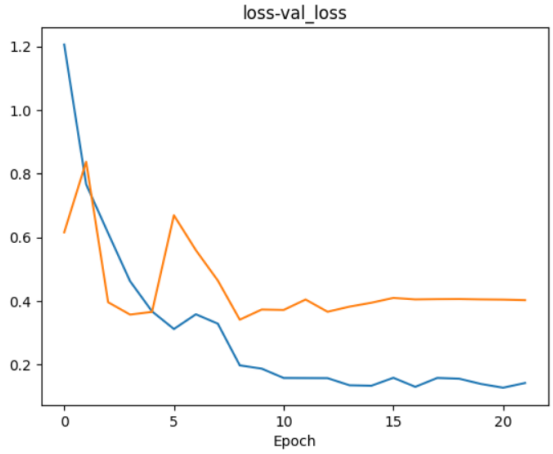
Table 4: BUN Classification Output

3.3 Using BUNTT Model

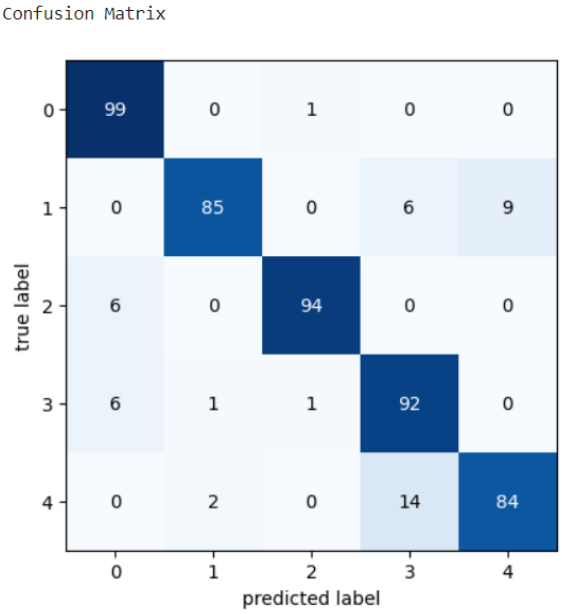
Lastly, we used the BUNTT model and training validation accuracy and loss, classification output and confusion matrix are shown in separate figures below.



(a) BUNTT training and validation accuracy



(b) BUNTT training and validation loss



(c) BUNTT confusion matrix

Training Accuracy	0.9470000267028809
Training Loss	0.14217838644981384
Validation Accuracy	0.8960000276565552
Validation Loss	0.40255120396614075

Table 5: Last Epoch Parameters

	precision	recall	f1-score
clear	0.8919	0.9900	0.9384
fog	0.9659	0.8500	0.9043
night	0.9792	0.9400	0.9592
rain	0.8284	0.9200	0.8679
snow	0.9032	0.8400	0.8705
accuracy	-	-	0.9080
macro avg	0.9123	0.9080	0.9080
weighted avg	0.9123	0.9080	0.9080

Table 6: BUNTT Classification Output

As we can see in the training accuracy, training loss, validation accuracy and validation loss figures and values, we built three models: DenseNet 121, BUN and BUNTT. Results are seperately seen. At the third model (BUNTT), we have the best accuracy and loss values.

4 Conclusion

In this study, we have addressed the classification of weather using deep learning techniques. We proposed a comprehensive approach that leverages transfer learning with DenseNet 121, incorporates CBAM for enhanced feature representation, and integrates batch normalization, instance normalization and regularization techniques to improve model performance and generalization. Our approach showed promising results.

The key contributions of our work include the utilization of transfer learning with DenseNet 121, we also introduced CBAM blocks to capture channel-wise dependencies and enhance feature representations, leading to more accurate predictions. Additionally, the incorporation of batch normalization and regularization techniques improved the model's robustness and generalization ability.

DenseNet 121 is a pre-trained model. We create and train our own network using this model. The purpose of this is to make comparisons with other models. In our other models (BUN and BUNTT), we aimed to extend the duration by increasing the number of epochs in our BUN model. We used an adaptive learning rate algorithm to check the validation accuracy value. If it did not exceed the previous max value during 3 epochs, that is, if there was no improvement, it reduced the learning rate in the 4th epoch. When we looked at the results we got, we saw that our value was gradually improving and finally we reached the highest in BUNTT model. Therefore, the use of BUNTT model is more effective compared to other models.

References

[1] Instance Normalization. Explained—Papers With Code.(n.d.). https://paperswithcode.com/method/instance-normalization

[2] Deep architecture based on DenseNet-121 model for weather image recognition. (n.d.). https://thesai.org/Downloads/Volume13No10/Paper_65-Deep_Architecture_based_on_DenseNet_121_Model.pdf

[3] Wikimedia Foundation. (2023, December 13). DENSENET121. Wikipedia. https://it.wikipedia.org/wiki/DenseNet121

[4] CBAM: Convolutional Block Attention Module. (n.d.-a). https://openaccess.thecvf.com/content_ECCV_2018/papers/Sanghyun_Woo_Convolutional_Block_Attention_ECCV_2018_paper.pdf