Deep Architecture based on DenseNet-121 Model for Weather Image Recognition

Saleh A. Albelwi
Department of Computer Science, Faculty of Computing and Information Technology
Industrial Innovation and Robotics Center, University of Tabuk
Tabuk, Saudi Arabia

***Abstract*—Weather conditions have a significant effect on humans' daily lives and production, ranging from clothing choices to travel, outdoor sports, and solar energy systems. Recent advances in computer vision based on deep learning methods have shown notable progress in both scene awareness and image processing problems. These results have highlighted network depth as a critical factor, as deeper networks achieve better outcomes. This paper proposes a deep learning model based on DenseNet-121 to effectively recognize weather conditions from images. DenseNet performs significantly better than previous models; it also uses less processing power and memory to further increase its efficiency. Since this field currently lacks adequate labeled images for training in weather image recognition, transfer learning and data augmentation techniques were applied. Using the ImageNet dataset, these techniques fine-tuned pre-trained models to speed up training and achieve better end results. Because DenseNet-121 requires sufficient data and is architecturally complex, the expansion of data via geometric augmentation—such as rotation, translation, flipping, and scaling—was critical in decreasing overfitting and increasing the effectiveness of fine-tuning. These experiments were conducted on the RFS dataset, and the results demonstrate both the efficiency and advantages of the proposed method, which achieved an accuracy rate of 95.9%.**

***Keywords*—Weather recognition; DenseNet-121; deep learning; data augmentation; transfer learning**

I. Introduction

Weather conditions have a significant effect on humans' daily lives and production [1], ranging from clothing choices to travel, outdoor sports, and solar energy systems. The growth of the intelligent transportation field has further prompted the development of a system that can automatically detect weather conditions [2]. Previously, traditional weather classification systems relied on human observation, but these methods are prone to error and are also time consuming. Current weather recognition systems depend on hardware equipment such as sensors [3], but the number of sensors, as well as their upkeep, necessitates both installation and regular maintenance, which can be expensive. This is especially so considering the impact of the weather on the sensors themselves—as the sensors weather various conditions, their accuracy may decrease, which may lead them to report erroneous weather conditions [4].

Recent advances in computer vision have shown notable progress in both scene awareness and image processing problems. One such study utilized multiple techniques to classify, segment, and localize pixels within urban images [5]. Research such as this has led to advances in intelligent vision systems, which can accurately recognize weather conditions using colored images. Due to these advances, as well as the presence of security cameras, computer vision systems are well-suited for automatic weather detection. Machine learning-based methods such as support vector machine (SVM), random forest, k-NN, and neural networks have since been proposed for weather condition recognition. They extract features from images, including saturation, contrast, noise, etc. The drawback is that these systems utilize multifaceted feature engineering as well as manual extraction, both of which increase their complexity and decrease their generalizability [6, 7].

Deep learning models such as AlexNet [8], VGG [9], Inception [10], and ResNet [11] have recently obtained outstanding performances in numerous computer vision applications, including image recognition, object detection, and semantic segmentation [12]. Deep learning models can obtain detailed data from weather images, which makes them more beneficial than classical machine learning methods. This is because deep learning techniques such as convolutional neural networks (CNNs) and ResNet have the capability to extract rich, abstract, and semantic information. For weather recognition in particular, they obtain better information than other techniques. Deep learning is, therefore, a more advanced machine learning method, one that can solve complicated problems that often stump traditional methods [13].

To further improve performance, researchers have constructed deeper and deeper CNNs, as the additional layers improve optimization. For example, AlexNet (the winner of ILSVRC2012) consists of eight layers, including five convolutional layers and three fully connected layers, VGG19 consists of 16 convolutional layers and three fully connected layers. Inception (the winner of ILSVRC2014) is composed of 21 convolutional layers and a single fully connected layer. Though these additional layers improve optimization, they increasingly expose CNNs to the critical issue of vanishing gradients. Several approaches have been proposed to address this, such as ReLU activation [14], batch normalization [15], and powerful weight initialization [13]. A second problem also arose, in that not all deep CNNs could be easily optimized. This has been addressed through two methods: the first is highway networks [16], which allow 2D-CNNs to connect via a memory device, thus training highway networks through gradient descent. The second is ResNets (the winner of ILSVRC2015), which simplified the former technique using a skip connection device, allowing deeper layers to receive data. The latter is more efficient than the former. DenseNet architecture further improved upon this by connecting all layers with the CNN, thereby improving data flow [17].

This paper proposes a system based on a DenseNet-121 deep learning model to classify weather conditions from images. DenseNet has achieved excellent performance, while also utilizing less memory and processing power than other state-of-the-art techniques. The proposed system can be utilized in the monitoring of traffic conditions, intelligent transit systems, and auxiliary driving features in automobiles, among others. The main advantage of DenseNet-121 is that it addresses vanishing gradients, which has multiple positive effects: one, it lessens the training burden of deep learning models; two, it allows for the reuse of features; and three, it lowers parameter use as compared to other popular deep learning models. The largest challenge in implementing this model was the lack of a labelled training dataset, and so two techniques were implemented: transfer learning and data augmentation. The former uses an ImageNet [18] dataset to pretrain the model, which decreases training time. The latter prevents overfitting, and also improves fine-tuning, by increasing the size of the training set based on image geometric transformations such as rotation, translation, flipping, and scaling. Overall, the proposed system is efficient, effective, and responsive, which allows it to make the best possible decisions even in poor weather conditions. The contributions of this paper are as follows:

- This paper presents a deep learning-based method using DenseNet-121 to effectively classify weather conditions from images in real time. The results have shown importance of model depth, as deeper models offer better results.
- To train the DenseNet-121 model, the proposed model needed large amounts of data. To circumvent the difficulty of obtaining additional weather images, this paper applied transfer learning and geometric image data augmentation techniques to reduce convergence time and increase performance.

The remainder of the paper is organized as follows: Section II provides the recent related works. Section III describes the model in detail. Section IV presents and discusses the experimental results. Finally, the conclusions and future works are presented in Section V.

II. Related Works

Early weather classification methods often extracted powerful features, a process that needed elaborate, hand-crafted features. Other methods utilized machine learning algorithms for classification, such as support vector machine (SVM), random forest, and K-NN. For example, Roser and Moosmann [6] proposed a method to define regions of interest. They employed a color histogram to extract the feature space, which was then passed into SVM to classify the weather images. The research in [19] concatenated the features extracted from a gradient amplitude histogram, the HSV color space, and road information. The extracted features were then fed into an AdaBoost algorithm to assign the image its weather label. Lagorio et al. [20] designed a statistical model based on a mixture of Gaussians to identity weather conditions from images. This mixture can identify both spatial and temporal changes, which assists the model in identifying the weather input. The downside of this method is that it cannot capture all weather events; it can only be implemented in particular conditions. Shen and Tan [21] built a weather condition detector, which can identify, for example, cloudy versus sunny weather using light illumination. In general, the methods that utilized hand-crafted weather features did not recognize weather conditions accurately.

Recently, improvements in the fields of deep learning and CNNs have been applied to weather recognition tasks. New research has utilized deep learning algorithms for classifying weather from images. For instance, Xiao et al. [22] presented a MetaCNN for classifying weather phenomena. MetaCNN is a modified version of the VGG16 [9] model. In MetaCNN, the authors replaced the fully connected layers in VGG16 with a single, global, average pooling layer. The final output was a softmax that estimated the probability distribution for each weather class label. The authors introduced a new dataset called the weather phenomenon database (WEAPD) that consists of 6,877 images for 11 different weather classes. Other researchers, such as Huang and Chang [23], employed a self-organizing map (SOM) to automatically classify weather images based on six variables at five different weather stations in Taiwan. Xia et al. [24] modified ResNet-50 to ResNet-15 to classify weather images, where the convolutional layers extracted the weather features that would be fed into the softmax function for classification. Guerra et al. [4] proposed a framework to classify weather as rain, fog, or snow using super-pixel masks that enhanced the input image. Then, the features were extracted using a pre-trained CNN and passed into an SVM for classification. Zhaoa et al. [1] developed a CNN-RRN model for multi-class weather on a single image, where the CNN was exploited to extract features. Next, the channel-wise attention model chose the most discriminative features; the convolutional LSTM was then used to estimate the weather class label. These deep learning methods of weather recognition are generally superior to traditional methods, but the limitations are as follows: one, these methods utilize large datasets, and two, they require a high-end GPU for training, making them computationally expensive.

Due to the great performance of DenseNet architecture, it has been used in diverse areas, including disease diagnosis, in the detection of Alzheimer's disease using an MRI [25], COVID-19 from chest images [26], and plant diseases [27].

Depth is an important consideration in improving the findings of deep learning models. Because of this, the proposed model is based on DenseNet-121, which has dense connections and thereby increases accuracy. These connections are advantageous because they utilize feature reuse to remain compact; use feed-forward networks; lower the number of parameters needed; increase feature propagation; and encourage feature generation, all of which increase accuracy and speed.

III. Proposed Model

The workflow of the proposed system is shown in Fig. 1. First, the pretrained DenseNet-121 model was loaded. This model was trained on ImageNet for weight initialization. The last layer was removed and replaced with a softmax with three neurons, where each neuron predicts the weather condition class (rain, fog, or snow). Then the model was re-trained and fine-tuned with the RFS dataset. In the testing phase, this trained model was efficiently used to estimate weather conditions.

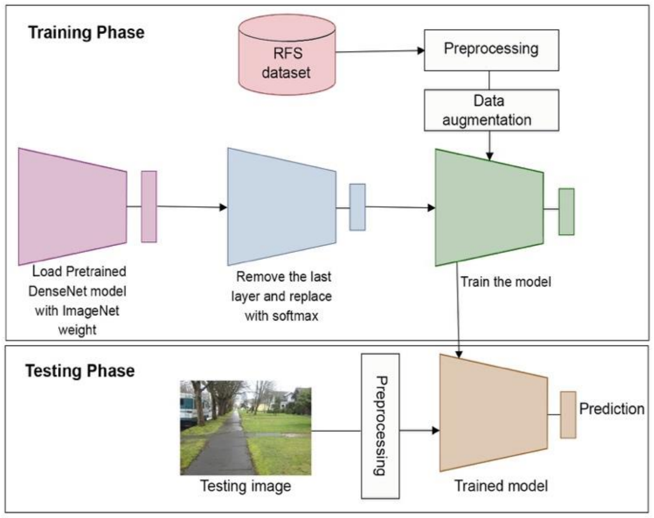


Fig. 1. The Workflow of the Proposed System.

A. Model Architecture

This paper utilized a Dense Convolutional Network (DenseNet) [28] comprised of 121 layers to classify weather images. DenseNet is a special kind of CNN that was originally proposed by Huang et al. It achieved state-of-the-art results on several image classification datasets, such as Cifar-10 and SVHN. In a DenseNet architecture, layers are connected with dense blocks, meaning that each layer utilizes inputs from all previous layers in order to create a feature map that will send data to all of the following layers. Therefore, the l^{th}-layer receives all previous features maps $(x_0, x_1, \dots\dots, x_{l-1})$ as inputs:

$$x_l = H_l([x_0, x_1, \dots\dots, x_{l-1}]) \quad (1)$$

Here, $[x_0, x_1, \dots\dots, x_{l-1}]$ represents the concentration of all previous feature maps of l^{th}- layer. x_l is the output of the l^{th} layer, and H_l represents l^{th} layer, which is a composition function consisting of three successive operations including batch normalization, a ReLU activation function, and convolution. DenseNet is similar to methods such as ResNet, but the latter combines previous layers with future layers while DenseNet concatenates layers instead. DenseNet approaches the problem of vanishing gradients by reusing features which also reduces the number of parameters. As shown in Fig. 2, DenseNet-121 utilizes four dense blocks. Between each block is a transition layer that utilizes down-sampling on the feature maps to create a 1× 1 convolution as well as a 2× 2 average pooling layer. The dense blocks comprise multiple convolutional layers, which are connected in series and serve as cross-layer connections between distant layers. To increase non-linearity, DenseNet-121 utilizes a ReLU activation function to increase non-linearity. The proposed ReLU activation is defined as follows:

$$ReLU(x) = \max(0, x) \quad (2)$$

The last layer is a fully connected layer with a softmax function that predicts the probability of a weather image class.

The softmax is defined by the following equation:

$$sm(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{C} e^{z_j}} \quad \text{for } i = 1, \dots\dots, C \quad (3)$$

Here, $z = (z_1, \dots\dots, z_C) \in \mathbb{R}^C$. z is the input vector, and the exponential is implemented for each value z_i. Note that the sum of output vector $sm(z)$ is equal to 1. A summary of the DenseNet-121 model layers is presented in Table I.

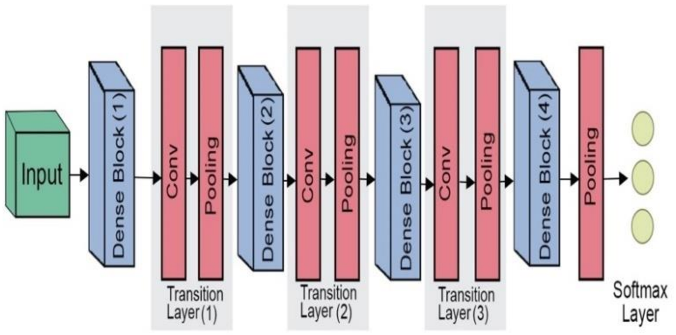


Fig. 2. The DenseNet-121 Architecture, which Consists of Four Dense Block Layers and Three Transition Layers.

TABLE I. LAYERS DETAILS OF DENSENET-121. THE DENSENET-121 HAS 6, 12, 24, 16 LAYERS IN FOUR DENSE BLOCKS

Layers	Output size	DenseNet-121
Convolution	112×112	7×7 conv, stride 2
Pooling	56×56	3×3 max pool, stride 2
Dense Block (1)	56×56	$\begin{bmatrix} 1 \times 1\ conv \\ 3 \times 3\ conv \end{bmatrix} \times 6$
Transition layer (1)	56×56	1×1 *conv*
	28×28	2×2 average pool, stride 2
Dense Block (2)	28×28	$\begin{bmatrix} 1 \times 1\ conv \\ 3 \times 3\ conv \end{bmatrix} \times 12$
Transition layer (2)	28×28	1×1 *conv*
	14×14	2×2 average pool, stride 2
Dense Block (3)	14×14	$\begin{bmatrix} 1 \times 1\ conv \\ 3 \times 3\ conv \end{bmatrix} \times 24$
Transition layer (3)	14×14	1×1 *conv*
	7×7	2×2 average pool, stride 2
Dense Block (4)	7×7	$\begin{bmatrix} 1 \times 1\ conv \\ 3 \times 3\ conv \end{bmatrix} \times 16$
Classification layer	1×1	7×7 global average pool
		Softmax layer

B. Transfer Learning

The deep DenseNet-121 model requires a large training set to increase its accuracy rate, but collecting labelled weather images is difficult. Transfer learning (TL) has been proposed as a common technique to address this limitation. The TL strategy pre-trains a model on a large, labelled dataset and then transfers the gained knowledge (learned weights) to other related tasks within the same architecture design. It treats the model as a starting point in the target task's training, which avoids the process of training the model from scratch with random weight initializations. Once TL is complete, the last layer must be altered to the number of required classes and then fine-tuned on the target dataset of interest. Recent research has demonstrated that TL improves performance rates compared to training a model from scratch on a small dataset. In addition, TL enhances generalization, reduces both overfitting training time, and decreases the labeled data required. Recently, TL has been applied widely in computer vision and natural language processing applications [29].

This research used a DenseNet-121 model pre-trained on the ImageNet dataset, which is composed of 1.2 million colored images in 1000 categories; therefore, the initialization of DenseNet-121 weights came from the pre-trained model. The output layer was removed and replaced with another layer containing three neurons, each matching up to a weather class label. A softmax function was applied in the final layer of the proposed model to predict the probability of each class as the output. Finally, the model was trained and fine-tuned across all layers on the RDF weather dataset.

C. Image Data Augmentation

Another way to deal with a limited training set is image data augmentation. This technique alters the images in the existing training set. This leads to an increase in the number of examples, and therefore the diversity, in the training set. Another advantage to data augmentation is the reduction of overfitting. Data augmentation can be divided into two main classes: geometric and color transformations. Geometric augmentation affects only the location of the pixels (e.g., flipping, shifting, cropping, and resizing) as illustrated in Fig. 3. In contrast, color augmentation modifies the values of the image pixels (e.g., blurring and color distortion). Since the color of weather images plays a critical role in recognizing the weather, this paper only applied geometric augmentation to increase the training set while keeping the image pixel values. The geometric augmentation was performed as follows:

- Rotation was performed by rotating the image between 0 and 360 degrees randomly. This research rotated the images randomly in range from 0 to 45.
- Flipping flipped the image across the x and/or y axis.
- Translation moved the image horizontally or vertically (or both). Width translation and height translation were ranges (as a fraction of the total width or height) within which images were randomly translated vertically or horizontally.
- Cropping removed part of the image.
- Scaling increased or decreased the size of the image.

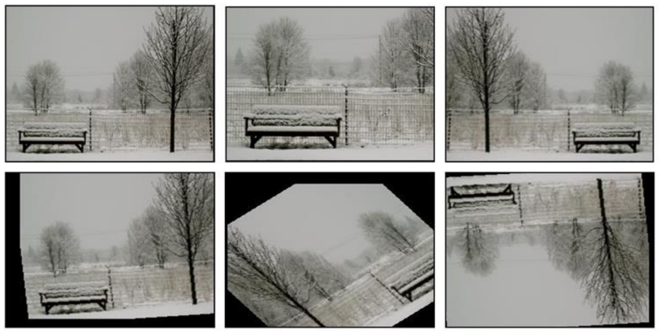


Fig. 3. Different Methods of Data Augmentation Including Rotation, Flipping, Translation, and Cropping and Resizing.

IV. Experimental Results and Discussion

A. Dataset

In this work, we used the Rain Fog Snow (RFS) weather dataset. This is an open source dataset that was proposed by [2] to support computer vision and deep learning applications in classifying weather via images. The images are collected from different locations with different environmental settings. Each class contains 1100 images. Fig. 4 shows a sample of the RFS dataset for each weather class. This dataset is particularly effective for this research because it contains quality images, various environments, and targeted labels. All images were resized to $224 \times 224 \times 3$ for suitable input into our DenseNet-121 model. We randomly selected 800 images for training and 300 for testing for each class.

B. Pre-processing

Image pre-processing is significant in deep learning models. Min-max normalization is one of the common techniques used to rescale and transform original image pixels into an appropriate size, typically a range between 0 and 1. This removes biases by eliminating differences in magnitude. As a result, this normalization accelerates training and enhances the model's performance. In this paper, we normalized the image pixels in a range from 0 to 1 using the following equation:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \quad (4)$$

where x is the original pixel value, x' is the new value, and $\min(x)$ and $\max(x)$ represent the minimum and maximum pixel values of the image, respectively.

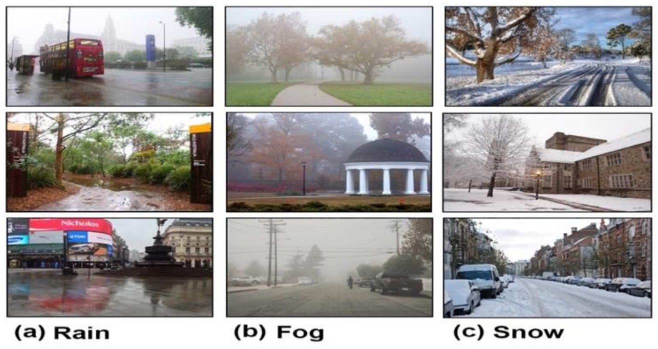


Fig. 4. A Sample of RFS Dataset for each Class. Each Column Represents One Class.

C. Training Setting and Model Implementation

The proposed system was implemented in Python. DenseNet-121, and other deep learning models, were developed using the PyTorch framework. The experiments were then performed using Google Colab Pro to access a faster GPU. Adam [29] optimizer was also used to optimize the weights and biases of the DenseNet-121 with a learning rate of 0.001 and a momentum set to 0.9. The batch size was set to 64 and the number of epochs to 100. The formula for updating weights based on Adam optimizer is defined as follows:

$$\theta_{t+1} = \theta_t + \Delta\theta_t \quad (5)$$

where

$$v_t = \beta_1 * v_{t-1} - (1-\beta_1) * g_t \quad (6)$$

$$s_t = \beta_2 * s_{t-1} - (1-\beta_2) \star g_t^2 \quad (7)$$

$$\Delta\theta_t = -\lambda \frac{v_t}{\sqrt{s_t+\epsilon}} * g_t \quad (8)$$

Here, λ indicates the initial learning rate, g_t is the gradient with respect to θ_t at time t. v_t represents the first moment estimate, s_t is the second moment, and β_1 and β_2 are hyperparameters. The values of β_1 and β_2 are set to 0.9 and 0.99, respectively. To minimize the loss function, the proposed model utilized a backpropagation algorithm. This research employed categorical cross-entropy as the loss function, defined as follows:

$$\mathcal{L}(\theta) = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{C} y_{i,j} \log(\hat{y}_{i,j}) \quad (9)$$

where y is the correct output, $\hat{y}$ is the predicted output, n represents the number of training samples, and C is the number of classes $(C = 3)$. L_2 regularization was implemented to alleviate overfitting with a weight decay value of 10^{-4}. Data augmentation was implemented using Transforms from the Torchvision library. It contains multiple transforms, which allowed for different types of augmentation on the images. The proposed model was initialized with weights that were pre-trained on ImageNet dataset. Doing this allowed this research to leverage the pretrained model. Then, the model was customized to weather recognition by training and finetuning it on the weather dataset.

D. Results and Discussion

To gauge the efficacy of the proposed system for labelling weather conditions from images using DenseNet-121, the proposed system was compared to deep learning models that obtained state-of-the-art results in a variety of computer vision tasks. These models include AlexNet [8], VGG16 [9], Inception [10], and ResNet-18 [11]. To evaluate the performance of the model, this research used accuracy to measure performance. The accuracy of the testing set was computed as follows:

$$Acc = \frac{Number\ of\ correct\ predictions}{\text{Total number of testing examples}} \quad (10)$$

To compute this, the proposed DenseNet-121 model was trained on the FRS training set. Once this was completed, this research tested the proposed model and received, as an output, the classification results, which were one of three weather conditions: fog, rain, or snow. The accuracy of this model's recognition was then tested on over 900 test images. The results are shown in Table II and Fig. 5. The proposed model obtained the best average accuracy l, with a rate of 95.6%, followed by ResNet-18, AlexNet, VGG16, and Inception, respectively. DenseNet-121 had an accuracy at least 2% higher than the other deep learning models.

The results also show the accuracy of each category for each deep learning model. Among them, the proposed model achieved an accuracy rate on foggy images of 97.55%, 93.55% on rainy images, and 96.98% on snowy images. This means that the proposed DenseNet-121 model achieved superior results on classifying weather conditions from all three image types.

This research also studied the effect of TL for training a model versus training one from scratch with random weights. To do this, this paper evaluated the performance of the proposed DenseNet-121, trained with initial weights set to the pre-trained model from ImageNet dataset, as compared to a second DenseNet-121 model that was trained from scratch with random weights at initialization. Table III displays the results. By utilizing weight initialization, the pre-trained model achieved an accuracy of 95.9% as compared to 91.3% for the model trained from scratch. Performing TL allows the necessary features to be extracted from ImageNet; by doing this, the data is used more efficiently and thereby the proposed model is trained to identify weather conditions more easily. A DenseNet-121 model with random weight initialization obtained accuracy rate of about 59% after the first iteration (see Fig. 6 (a)), while DenseNet-121 based on transfer learning obtained an accuracy rate of 85% after the first iteration. As shown in Fig. 6, the proposed TL model converged in fewer iterations than the model that started with random weights, in respect to loss and accuracy rate. This decreased the computation time.

TABLE II. ACCURACY COMPARISONS OF THE PROPOSED METHOD WITH DIFFERENT DEEP LEARNING MODELS

Model	Foggy (%)	Rainy (%)	Snowy (%)	Avg. Accuracy (%)
VGG16 [9]	93.81	93.54	92.56	93.2
Inception [10]	88.89	89.00	93.40	90.3
AlexNet [8]	92.43	92.64	95.53	93.5
ResNet-18 [11]	92.47	90.79	94.30	93.7
DenseNet-121 (proposed model)	97.55	93.55	96.98	95.9

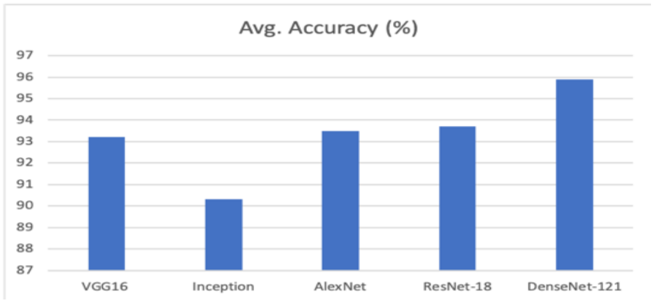
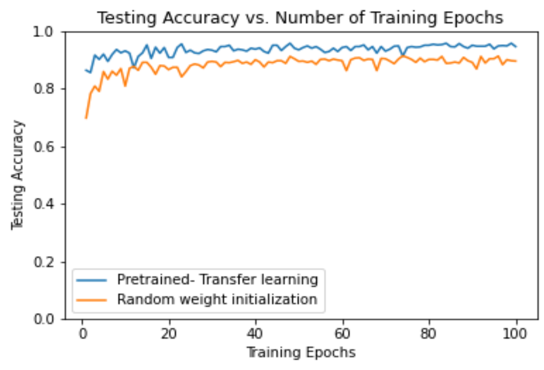
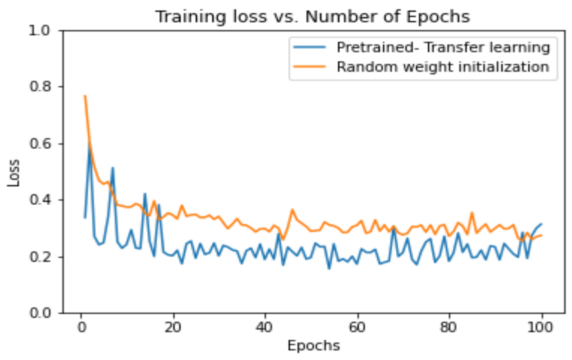


Fig. 5. Comparison of Average Accuracy for different Models.



(a)



(b)

Fig. 6. (a) The Testing Accuracy Achieved from the DenseNet-121 Models Trained from Scratch and with TL. (b) The Loss Values Obtained for the DenseNet-121 Models Trained from Scratch and TL.

TABLE III. COMPARISON OF ACCURACY RATE OF DENSENET-121 BASED ON WEIGHT INITIALIZATION APPROACH

Weight Initialization Approach	***Accuracy (%)***
Random weight initilization	91.3
Pretained from ImageNet	**95.7**

Deep learning models require massive amounts of training to achieve high accuracy. The training examples are augmented by rotation, flipping, and shifting, and the accuracy has been, therefore, improved. In addition, pre-training DenseNet-121 with a large dataset such as ImageNet further improved the performance, and the model learned faster than the one trained from scratch. Data augmentation plays a significant role in reducing overfitting in particular when the training set is small.

These results suggest that the performance of the proposed model outperformed previous research and state-of-the-art techniques. Continued hardware and software advances will enable researcher to build deeper neural networks, which have higher representation power than shallower ones. These results demonstrated the importance of network depth, as networks with more layers achieved better outcomes. One of DenseNet’s advantages is its increased flow of information and gradients, which improves training time. Each layer can access both the loss function and the original input, creating a deep supervision that trains deep architectures. In addition, feature propagation allows for the repetition, and efficient use, of features. It also reduces the parameters needed and, therefore, reduces the calculations needed, all of which is an advantage because DenseNet layers are narrow. They may have only 12 filters per layer, and may add only a small set of feature maps to the network’s collective knowledge; it is the final classifier that uses all the feature maps to make a decision. This paper’s results and analysis show that the proposed model classifies weather conditions accurately by utilizing DenseNet features effectively. Furthermore, the proposed model also works well in real-time environments, which is another advantage.

V. CONCLUSIONS AND FUTURE WORKS

Weather conditions have a significant impact on daily activities. Deep learning models have shown promising results in numerous computer vison and image analysis tasks. Recent research has demonstrated that CNNs can be deeper, and also more accurate and efficient, if there are shorter connections between the inputs and outputs. To test these hypotheses, this paper employed DenseNet-121 to recognize weather conditions from images. The results demonstrated that the proposed system, based on a DenseNet-121 model with transfer learning and data augmentation, maximized the accuracy rate with a small number of training examples. This paper also implemented and evaluated popular deep learning methods for recognizing weather conditions from images.

Future works will focus on simplifying the DenseNet-121 architecture to fit with edge devices while keeping the same performance. In addition, future research should apply self-supervised learning to utilize an unlabelled training set in pre-training, so that the proposed model can learn discriminative features.

REFERENCES

[1] B. Zhao, X. Li, X. Lu, and Z. Wang, "A CNN–RNN architecture for multi-label weather recognition," Neurocomputing, vol. 322, pp. 47-57, 2018.

[2] K. D. M.-M. McDonald-Maier, J. C. V. G. Guerra, Z. K. Khanam, S. E. Ehsan, and R. S. Stolkin, "Weather Classification: A new multi-class dataset, data augmentation approach and comprehensive evaluations of Convolutional Neural Networks," 2018.

[3] X. Li, Z. Wang, and X. Lu, "A multi-task framework for weather recognition," in Proceedings of the 25th ACM international conference on Multimedia, 2017, pp. 1318-1326.

[4] J. C. V. Guerra, Z. Khanam, S. Ehsan, R. Stolkin, and K. McDonald-Maier, "Weather Classification: A new multi-class dataset, data augmentation approach and comprehensive evaluations of Convolutional Neural Networks," in 2018 NASA/ESA Conference on Adaptive Hardware and Systems (AHS), 2018: IEEE, pp. 305-310.

[5] M. R. Ibrahim, J. Haworth, and T. Cheng, "WeatherNet: Recognising weather and visual conditions from street-level images using deep residual learning," ISPRS International Journal of Geo-Information, vol. 8, no. 12, p. 549, 2019.

[6] M. Roser and F. Moosmann, "Classification of weather situations on single color images," in 2008 IEEE intelligent vehicles symposium, 2008: IEEE, pp. 798-803.

[7] W.-T. Chu, X.-Y. Zheng, and D.-S. Ding, "Camera as weather sensor: Estimating weather information from single images," Journal of Visual Communication and Image Representation, vol. 46, pp. 233-249, 2017.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," Advances in neural information processing systems, vol. 25, pp. 1097-1105, 2012.

[9] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556, 2014.

[10] C. Szegedy et al., "Going deeper with convolutions," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 1-9.

[11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770-778.

[12] S. Albelwi, "Survey on Self-Supervised Learning: Auxiliary Pretext Tasks and Contrastive Learning Methods in Imaging," Entropy, vol. 24, no. 4, p. 551, 2022.

[13] K. Zhang, M. Sun, T. X. Han, X. Yuan, L. Guo, and T. Liu, "Residual networks of residual networks: Multilevel residual networks," IEEE Transactions on Circuits and Systems for Video Technology, vol. 28, no. 6, pp. 1303-1314, 2017.

[14] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in Icml, 2010.

[15] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in International conference on machine learning, 2015: PMLR, pp. 448-456.

[16] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Highway networks," arXiv preprint arXiv:1505.00387, 2015.

[17] R. Xian, "Synchronization of Stochastic Memristive Neural Networks with Retarded and Advanced Argument," Journal of Intelligent Learning Systems and Applications, vol. 13, no. 1, pp. 1-14, 2021.

[18] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in 2009 IEEE conference on computer vision and pattern recognition, 2009: Ieee, pp. 248-255.

[19] X. Yan, Y. Luo, and X. Zheng, "Weather recognition based on images captured by vision system in vehicle," in International Symposium on Neural Networks, 2009: Springer, pp. 390-398.

[20] A. Lagorio, E. Grosso, and M. Tistarelli, "Automatic detection of adverse weather conditions in traffic scenes," in 2008 IEEE Fifth International Conference on Advanced Video and Signal Based Surveillance, 2008: IEEE, pp. 273-279.

[21] L. Shen and P. Tan, "Photometric stereo and weather estimation using internet images," in 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009: IEEE, pp. 1850-1857.

[22] H. Xiao, F. Zhang, Z. Shen, K. Wu, and J. Zhang, "Classification of weather phenomenon from images by using deep convolutional neural network," Earth and Space Science, vol. 8, no. 5, p. e2020EA001604, 2021.

[23] A. Huang and F.-J. Chang, "Using a Self-Organizing Map to Explore Local Weather Features for Smart Urban Agriculture in Northern Taiwan," Water, vol. 13, no. 23, p. 3457, 2021.

[24] J. Xia, D. Xuan, L. Tan, and L. Xing, "ResNet15: Weather Recognition on Traffic Road with Deep Convolutional Neural Network," Advances in Meteorology, vol. 2020, 2020.

[25] B. Solano-Rojas, R. Villalón-Fonseca, and G. Marín-Raventós, "Alzheimer's disease early detection using a low cost three-dimensional densenet-121 architecture," in International conference on smart homes and health telematics, 2020: Springer, pp. 3-15.

[26] L. Sarker, M. M. Islam, T. Hannan, and Z. Ahmed, "COVID-DenseNet: a deep learning architecture to detect COVID-19 from chest radiology images," Preprint, vol. 2020050151, 2020.

[27] S. Nandhini and K. Ashokkumar, "An automatic plant leaf disease identification using DenseNet-121 architecture with a mutation-based henry gas solubility optimization algorithm," Neural Computing and Applications, vol. 34, no. 7, pp. 5513-5534, 2022.

[28] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 4700-4708.

[29] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014.