A Novel Transformer Network With Shifted Window Cross-Attention for Spatiotemporal Weather Forecasting

Alabi Bojesomo, Hasan AlMarzouqi, *Senior Member, IEEE*, and Panos Liatsis, *Senior Member, IEEE*

***Abstract*—Earth observation is a growing research area that can capitalize on the powers of artificial intelligence for short time forecasting, a now-casting scenario. In this work, we tackle the challenge of weather forecasting using a video transformer network. Vision transformer architectures have been explored in various applications, with major constraints being the computational complexity of attention and the data-hungry training. To address these issues, we propose the use of video Swin-transformer (VST), coupled with a dedicated augmentation scheme. Moreover, we employ gradual spatial reduction on the encoder side and cross-attention on the decoder. The proposed approach is tested on the Weather4Cast2021 weather forecasting challenge data, which requires the prediction of 8 h ahead future frames (4 per hour) from an hourly weather product sequence. The dataset was normalized to 0–1 to facilitate the use of the evaluation metrics across different datasets. The model results in an mse score of 0.4750 when provided with training data, and 0.4420 during transfer learning without using training data, respectively.**

***Index Terms*—Encoder–decoder video architecture, now-casting, shifted window cross attention, video Swin-transformer (VST), weather forecasting.**

I. Introduction

WEATHER forecasting is a crucial driving force in agriculture and the autonomous vehicle industry [1]. Accurate weather forecasting affects the successful deployment of autonomous vehicles and the management of food production. When designing autonomous navigation and collision avoidance technologies, for example, awareness of the weather is an essential part of the location context. Also, reasonable weather prediction is essential in monitoring soil nutrients and regulating crop yield in the agricultural industry.

AI is gradually gaining traction in weather forecasting due to its relative simplicity, when compared to numerical weather prediction (NWP) [2], [3], [4], [5], [6]. AI approaches applicable to weather forecasting can be categorized according to the network structure, e.g., convolutional neural networks (CNN) [7], [8], [9], [10], autoencoders [11], [12], and recurrent networks [13], [14], [15], [16].

CNN has been used in many state-of-the-art image classification [17], [18], [19] and semantic segmentation models [20], [21]. Spatiotemporal forecasting has also been tackled with CNN models, based on their demonstrated performance on segmentation tasks [21], [22], [23]. Recently, self-attention transformer-based models have gradually revolutionized computer vision applications [24], [25], [26], [27]. While natural language processing (NLP) uses transformer networks on encoded tokens, vision transformer-based models utilize patch-based image encoding [24]. Vision transformers are showing promise in many applications; however, they involve the computationally costly self-attention mechanism, which presents a challenge to be addressed. Various modifications of self-attention [28], [29], [30], [31] have been proposed in literature to address computational demands.

The vision transformer is widely used in many computer vision applications, including semantic segmentation [24], [25], [26], [32]. Dosovitskiy et al. [24] used patch-based image encoding, similar to word embedding in NLP, making vision problems amenable to the transformer network [32]. Other works explore modifications of the attention layer, resulting in more efficient transformer architectures for vision tasks, such as the pyramid vision transformer [25]. Spatiotemporal forecasting is similar to dense prediction tasks, e.g., depth estimation and segmentation, which have been tackled by CNN-based architectures [20]. However, more recently, the vision transformer is gradually becoming the dominant architecture in these applications [24], [25], [26], [29].

Among the widely adopted vision transformer architecture is the *shifted window transformer* (Swin transformer), which uses local window attention with shift operations between transformer layers [29], popular in classification tasks. The Swin transformer has also been used as an encoder (feature extractor) for downstream tasks, such as object detection, segmentation [33], and forecasting [21], [34], [35], [36]. Cao et al. [33] replaced all CNN blocks of the typical UNet architecture [20] with Swin transformer blocks, resulting in the Swin-UNet architecture. The Swin-UNet network uses an MLP-based layer (multilayer perceptron) as a patch expanding layer for upsampling in the decoding branch. This layer can be viewed as the opposite of the MLP-based downsampling layer (i.e., patch merging

Manuscript received 18 May 2022; revised 17 July 2022, 15 May 2023, and 1 August 2023; accepted 4 October 2023. Date of publication 13 October 2023; date of current version 23 November 2023. This work was supported by the ICT Fund, Telecommunications Regulatory Authority (TRA), Abu Dhabi, United Arab Emirates. *(Corresponding author: Alabi Bojesomo.)*

The authors are with the Department of Electrical Engineering and Computer Sciences, Khalifa University, Abu Dhabi 127788, United Arab Emirates (e-mail: alabi.bojesomo@ku.ac.ae; hasan.almarzouqi@ku.ac.ae; panos.liatsis@ku.ac.ae).

Digital Object Identifier 10.1109/JSTARS.2023.3323729

layer), used in pyramid vision transformers [25], [29], [33]. Lio et al. [30] proposed the 3-D (3-D/video) Swin transformer, replacing the patch embedding, patch merging, and shifted window transformer with their respective 3-D variants, resulting in a parameter-efficient network, as evidenced in performance on several video datasets (e.g., Something-Something v2 [37], Kinetics-400 [38], and kinetics-600) [30].

Low-rank architectures have also been applied for forecasting time-series data with the aim of improving performance. A novel multistep forecasting framework is proposed in [39] to address the challenges of ultra-short-term forecasting of wind speed and wind power. The approach combines selective Hankelization, tensor decomposition, feature selection, and a low-rank tensor learning-based predictor. The method converts the time series into a high-order tensor structure using Hankelization and utilizes a low-rank tensor learning network with an long short-term memory (LSTM)-based encoder and attention-based decoder. Feature selection using similarity search is employed to enhance accuracy. The approach improves the accuracy of wind power and wind speed forecasting, but requires higher computational costs. In [40], a structured low-rank matrix completion is applied for data imputation in time series forecasting. The method employs Hankel matrixes and convex relaxation based on the nuclear norm. The proposed approach introduces a new formulation that assigns different weights for previous observations, resulting in improved performance. Experiments demonstrate its superiority in multistep ahead prediction compared to state-of-the-art methods. Inspired by compressed sensing, Liu [41] introduced the convolution nuclear norm minimization (CNNM) approach to recover the future part of convolutionally low-rank time series. However, CNNM is sensitive to trends and dynamics. To address this, a learnable orthonormal transformation called learning-based CNNM (LbCNNM) is proposed. LbCNNM uses principal component pursuit to learn suitable transformations, integrating dictionary learning, model combination, and coherence. Although computationally complex, LbCNNM effectively handles time series components and incorporates forecasts from other methods. In [42], a recurrent neural network (RNN)-based time series model is combined with a Gaussian copula process output model with a low-rank covariance structure. This approach enables modeling time-varying correlations among numerous time series with non-Gaussian marginal distributions. A low-rank factorization of the covariance matrix and a recursive formula for parameter estimation and likelihood computation are employed. However, the computational resources necessary for practical deployment of low-rank-based methods in spatiotemporal forecasting are substantial.

In this article, we propose a computationally efficient architecture based on transformers capable of capturing long-term spatiotemporal interactions for weather forecasting. Our contributions include the following.

1) We propose a deep learning architecture capable of capturing complex spatial relationships in remote sensing images, through the novel paradigm of video Swin transformer (VST) with shifted window cross-attention. The incorporation of shifted window attention promotes the exploration of interwindow relationships and global correlation integration, while effectively mitigating computational overhead. In addition, we employ cross-attention in the decoder, to enahnce feature extraction and further optimize computational efficiency.
2) The performance of the proposed approach is evaluated on the Weather4Cast2021 weather forecasting challenge dataset. The objective is to predict the future frames 8 h ahead based on an hourly weather product sequence, with 4 frames per hour. Our model achieves competitive results compared to the state of the art, attaining an mse score of 0.4750 when trained with available data and 0.4420 through transfer learning without utilizing training data. Notably, this is achieved by utilizing less than 50% of the parameters used by the model achieving the first position on the competition leaderboard, leading to faster training and inference times.
3) Lastly, we present an ablation study that includes a quantitative analysis and systematic evaluation of the model's prediction performance, focusing on both the weather products and the network structure. A qualitative analysis of the model prediction is presented, offering detailed insights into its performance with respect to future time steps (e.g., Fig. 6).

The rest of this article is organized as follows. A summary of related works is presented in Section II. The details of the proposed architecture and its layers are given in Section III, while the experimental results of the proposed approach in the IEEE Big Data 2021 Weather4cast challenge data are presented in Section IV. Finally, Section V concludes this article.

II. Related Works

Weather forecasting is the foundation of meteorology. Traditionally, weather prediction is based on physical modeling of the associated physical phenomena. This NWP approach requires the solution of spatiotemporal partial differential equations (PDE), related to a variety of underlying atmospheric processes, including radiative, chemical, dynamic, thermodynamic, etc [43]. It is obvious that modeling and solving this multitude of atmospheric processes requires substantial computational resources, which is, indeed, one of the disadvantages of the NWP method [44]. However, the most important drawback of the PDE approach is the chaotic nature of weather [45], which makes model performance highly dependent on the initial conditions. The use of NWP in short-term weather forecasting is problematic, it is a popular approach in long-term forecasting, as it is able to track long-scale trends [46].

The spatiotemporal nature of weather prediction offers major opportunities and challenges for the use of AI and data analytics methods. Data-driven weather prediction helps to avoid both the known and unknown chaotic behavior of weather fluctuations by focusing on the evolution of available data [47]. Physical phenomena associated with the weather are too complex to model and sometimes not well understood, paving the way for a relatively simpler data-driven approach. AI methods for weather forecasting can be classified into machine learning (ML)-based

and deep learning-based. Furthermore, ML architectures, applied to weather forecasting, can be grouped into static and dynamic (recurrent) in terms of whether the prediction is generated sequentially [48], [49], [50], [51], or not. Static ML architectures include clustering (K-means, PCA) [52], [53]; artificial neural networks (ANN) [49], [54], [55]; graph neural networks (GNN) [56]; clustering + neural networks [57], [58]; decision trees, such as gradient boosting (XGBoost [59], AdaBoost [50], CatBoost [50], [60], [61]); and random forest [62], [63]. On the other hand, dynamic ML architectures for weather forecasting make use of the sequential nature of training data in generating the forecast. Methods in this category include partially recurrent architectures [e.g., Elman neural networks] [64], and fully recurrent ones, such as RNN, LSTM, and gated recurrent units (GRU). The principle of fully recurrent architectures is that they capitalize on the dynamic patterns in the input data sequences [8], [11], [12], [14], [65], [66]. The common challenge of ML-based methods is that they require a good understanding of weather parameters in order to perform appropriate feature engineering. This limits their applicability as not all contributing factors to weather changes are yet known or can be accurately measured and/or tracked.

Deep learning-based approaches alleviate the need for the feature engineering stage of ML methods. Instead, feature learning is automatically performed with the use of convolution blocks, allowing data-driven extraction of the required features. These features are then used as the input to the classification/regression layer, which is usually a fully connected layer. Considering the spatiotemporal setting of weather data, CNN can be readily applied [10], NeXtNow [67], U-STNx [68] . In [20], a UNet architecture with a densely connected backbone was used for weather prediction [69], where the continuous aggregation of the densely connected CNN in the backbone ensures the reuse of the intermediate-state results. Moreover, generative adversarial networks have been successfully applied in weather forecasting [70], [71], [72], [73], [74]. Spatiotemporal weather forecasting using deep learning requires appropriate treatment of the spatial and temporal information in the data. For instance, CNN can be used to extract features from the spatial information, while recurrent networks (RNN, LSTM, and GRU) can be used to model the temporal interrelationship. Combining CNN with recurrent networks results in architectures, such as ConvLSTM [15], [16], [75], [76], [54], PredRNN [77], PredRNN++ [78], MetNet [13], TrajGRU [79], ConvGRU [66], and MFNet [80]. Spatiotemporal weather forecasting can also be tackled with transformers [81], originally proposed for NLP [32]. This is because attention networks used in transformers can eradicate the need for time-consuming sequential forecasting, an approach common to recurrent networks (ConvLSTM [75], PredRNN [77], PredRNN++ [78], ConvGRU [66]).

The state-of-the-art (SOTA) architectures for spatiotemporal weather forecasting [66], [69], [70] are memory intensive and require a substantially large number of parameters. In light of this, there is a need to develop lightweight, efficient, and improved accuracy spatiotemporal weather prediction models.

In this research, we are motivated by the self-attention mechanism of transformers, as it can capture the relationship between

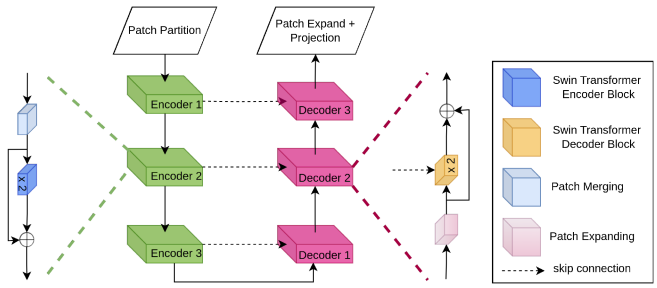

Fig. 1. Spatiotemporal encoder–decoder architecture with cross attention in the decoder.

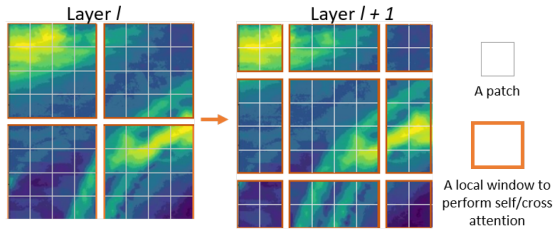

Fig. 2. Outline of the local window grouping in the shifted window-based architecture. The local window is shifted for a layer $n + 1$ following a regularly placed window in layer n. Since self(cross)-attention is window based, the shifting operation between layers provides the necessary connection, otherwise missing in local window-based attention. [29].

the input variables [32]. In fact, vision transformers are gradually dominating computer vision applications, including dense prediction [26], [33]. These architectures [24], [28], [29], [30], [31] can be applied to weather prediction. Solutions for this application involve an encoder (backbone, feature extractor) and a decoder (segmentor, predictor, forecaster) [57], [66], [69], [70]. This makes models designed for classification applicable as feature extractors. Furthermore, decoders, such as UNet [20], FaPN [82], UperNet [83], and SegFormer [84] among others have been proposed for dense prediction (e.g., weather forecasting). Among all these decoders, SegFormer uses attention layers, while others are based on CNN.

III. Methods

The proposed model (see Fig. 1) includes multiple stages for gradual downsampling of the spatial dimension in the encoder, which aid in capturing salient global representations. The encoder uses self-attention, and the decoder employs cross-attention to merge the skip connected input from the encoder with its main input [32], [35]. The developed model uses shifted window attention [30]. This encourages the exploitation of interwindow relationships as local window attention is used to reduce the computational overhead (see Fig. 2). Regular local transformer models exhibit a nonglobal correlation similar to convolutional networks. The interwindow relationship in shifted window transformer has the power to integrate the global correlation, even while being window-based.

As shown in Fig. 1, the input enters the network through a patch embedding layer, while the final output of the model comes

from a projected patch expanding layer. The patch embedding layer converts the spatiotemporal inputs into tokens, while the final output is a projection of tokens to spatiotemporal format. Three encoder and three decoder blocks are included in the model, each with four 3-D transformer layers (encoder/decoder). The limited number of layers helps to avoid the need for huge datasets to pretrain transformer-based models [24], [26], [29], [30]. The proposed model is composed of several layers and blocks; including the *patch partitioning layer*, *patch merging layer*, *patch expanding layer*, and the *shifted window transformer encoder* and *decoder* blocks.

A. Patch Transformation Blocks

The transformation of the input into workable patches as required by the transformer-based model is carried out by patch-specific layers. Also, the downsampling and upsampling in the intermediate blocks in the encoder and decoder, respectively, need the path transformation blocks.

In the *patch partitioning layer*, the spatiotemporal features are divided into patches and transformed using linear embedding. Two CNN layers are used to accomplish this. The first convolution has a stride value equal to its kernel size dimension to encourage patch partitioning, whereas the second convolution has a kernel size of 1 for linear embedding.

The *patch merging layer* downsamples features using linear [fully connected (FC)] layer, making it a knowledge-based operation contrary to traditional rule-based ones (average/max pooling operation). This layer flattens the features of each group of patches (2×2). This is followed by applying an FC-layer to convert the 4*C-dimensional features to 2*C-dimensional output, where C is the number of channels of the incoming features [29], [30]. On the decoding branch of the network, we have the *patch expanding layer*, which works in exactly the opposite way to the *patch merging layer*, thus resulting in a learned upsampling operation.

The projection head of the network (Fig. 1) includes a *patch expanding layer* to recover the original spatial dimensions. This is followed by an FC-layer for channel dimension projection.

B. Encoder Blocks

The proposed model's encoder backbone network includes a multistage VST. We employ three stages (Fig. 1), each with four 3-D transformer layers. Apart from the first encoder, which has linear embedding, each subsequent encoder block has a patch merging unit, followed by Swin-transformers.

A multihead self-attention (MSA) [24] layer is followed by a feed-forward network in the transformer layer used for computer vision tasks, with each of these layers preceded by layer normalization (LN) [85]. Because of the spatiotemporal nature of the input, the 3-D shifted-window MSA is used in this study (See Fig. 2). As highlighted in Fig. 3, the Swin transformer makes use of an interchange of sliding windows, where window (local) attention is next to another local but shifted window attention. This arrangement leads to (1) for any two attention layers

$$\bar{x}^l = \text{W-MSA}(\text{LN}(x^{l-1})) + x^{l-1}$$

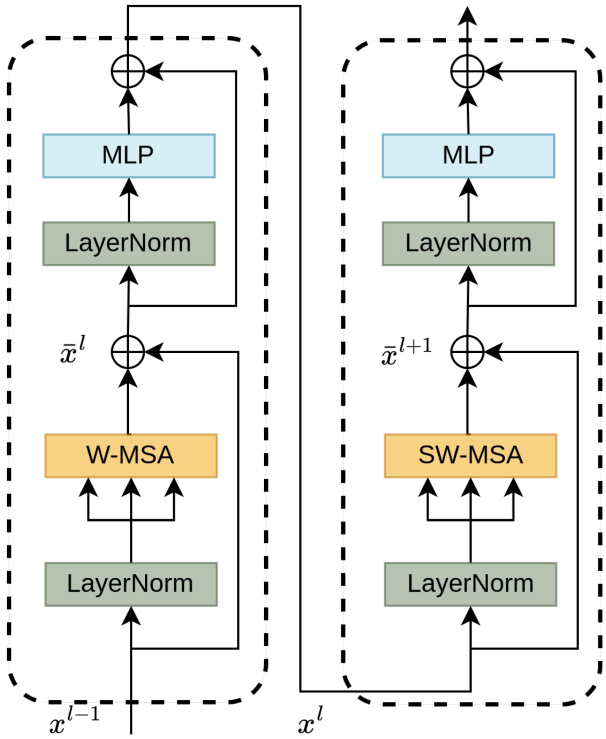

Fig. 3. Swin Transformer Encoder block: An encoder stacking of two concurrent self-attention layers is shown, with shifted-window attention always coming after nonshifted ones.

$$\begin{aligned} x^l &= \text{MLP}(\text{LN}(\bar{x}^l)) + \bar{x}^l \\ \bar{x}^{l+1} &= \text{SW-MSA}(\text{LN}(x^l)) + x^l \\ x^{l+1} &= \text{MLP}(\text{LN}(\bar{x}^{l+1})) + \bar{x}^{l+1} \end{aligned} \tag{1}$$

where x^l is the activation map (continuously processed input) of layer l, LN and MLP represent the LN and feed-forward layer, respectively, and W-MSA and SW-MSA represent windowed multi-head self-attention and *shifted window* multihead self-attention, respectively. In both W-MSA and SW-MSA, a relative position bias is used [29], [30]. However, the primary distinction between W-MSA and SW-MSA is a shift in window positioning before computing the local attention within the windowed blocks. In this research, we use (1, 7, 7) as the window size for all Swin transformers, with a shift size of 2 for the shifted window versions. In addition, the MLP layers include two FC-layers as

$$\text{MLP}(X) = f_1(f_2(X * W_1) * W_2) \tag{2}$$

where f_1 and f_2 are the transfer functions of the two layers, respectively, $X \in \Re^{\cdots \text{ x } d}$ is the input with... being the other possible dimensions of the input, $W_1 \in \Re^{d \text{ x } 4d}$ is the weight matrix of the first (hidden) fully connected layer, and $W_2 \in \Re^{4d \text{ x } d}$ is the weight matrix of (output) fully connected layer.

C. Decoder Blocks

In the decoder, we use multihead cross attention (MCA) units that enable the interaction of encoded tokens with decoding ones.

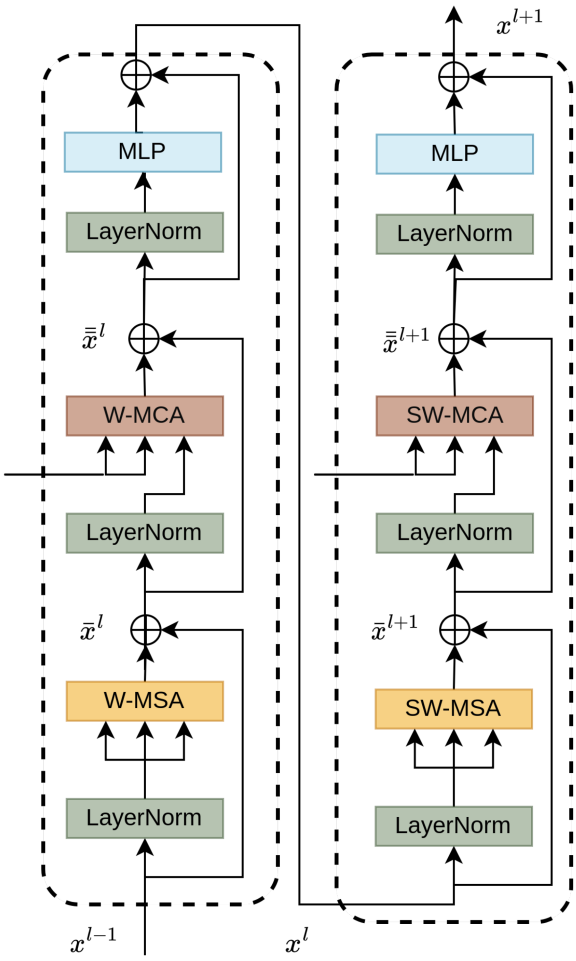

Fig. 4. Swin transformer decoder block: A decoder stacking of two concurrent cross-attention blocks is shown, with shifted-window attention always coming after nonshifted ones.

In contrast to MSA, which uses the same input as the *key*, *query*, and *value*; MCA uses one input as the *key* and *value*, while using another as the *query*. This ensures that we can explore the dependency of one input (*query*) on the other parameters.

The use of MCA only deals with the interaction between the skip connection (from the encoder) and the decoding input. There arises the need to further deal with self-dependency in the form of self-attention of the resulting output after MCA is applied before applying the MLP network. The MSA layer is followed by another MCA layer in the decoder [32]. Such an arrangement merges the skip-connected part with the decoding input. While this process is done using addition in LinkNet [86] and by concatenation in U-Net [20], we use MCA [32]. The building blocks of our cross-attention-based decoding are preceded by LN, an approach commonly used with vision transformers [24], [34], [35], [36].

Similar to the encoder layers, we use a 3-D shifted window MCA in the decoder. Related to (1), the Swin transformers in the decoder blocks alternately used self and cross attention, followed by shifted window self and cross attention, as illustrated in the following (Fig. 4):

$$\begin{aligned}
\bar{x}^l &= \text{W-MSA}(\text{LN}(x^{l-1})) + x^{l-1} \\
\bar{\bar{x}}^l &= \text{W-MCA}(\text{LN}(\bar{x}^l), y) + \bar{x}^l \\
x^l &= \text{MLP}(\text{LN}(\bar{\bar{x}}^l)) + \bar{\bar{x}}^l \\
\bar{x}^{l+1} &= \text{SW-MSA}(\text{LN}(x^l)) + x^l \\
\bar{\bar{x}}^{l+1} &= \text{SW-MCA}(\text{LN}(\bar{x}^{l+1}), y) + \bar{x}^{l+1} \\
x^{l+1} &= \text{MLP}(\text{LN}(\bar{\bar{x}}^{l+1})) + \bar{\bar{x}}^{l+1} \\
\bar{x}^{l+1} &= \text{SW-MSA}(\text{LN}(x^l)) + x^l \\
x^{l+1} &= \text{MLP}(\text{LN}(\bar{x}^{l+1})) + \bar{x}^{l+1} \qquad (3)
\end{aligned}$$

where y represents the output of the corresponding encoder (skip connection) as shown in Fig. 1. All other variables and functions follow the definitions in (1). The W-MCA and SW-MCA units are defined here with two inputs compared with W-MSA and SW-MSA in (1).

IV. Experimental Results

A. Data Description

The proposed system was tested on the challenging Traffic4cast 2021 [87], [88] weather dataset. The dataset used was part of the IEEE BigData Conference competition for weather movie snippet forecasting. As shown in Fig. 5, the dataset covers 11 regions including:

R1: Nile region (covering Cairo).
R2: Eastern Europe (covering Moscow).
R3: South West Europe (covering Madrid and Barcelona).
R4: Central Maghreb (Timimoun).
R5: South Mediterranean (covering Tripoli and Tunis).
R6: Central Europe (covering Berlin).
R7: Bosphorus (covering Istanbul).
R8: East Maghreb (covering Marrakech).
R9: Canary Islands.
R10: Azores Islands.
R11: North West Europe (London, Paris, Brussels, Amsterdam).

These regions are grouped into two for both *core challenge* and *transfer challenge*. The regions used for the *core challenge* are R1–R3, R7, and R8, and the data provided are divided into training, validation, and test sets. The *transfer challenge* only has the test set for testing the transferability of the trained model on data from the remaining regions without prior training.

The data are presented in 256 $\times$ 256 weather images for various weather parameters, as shown in Table I, categorized into five subgroups, with an area resolution of 4 $\times$ 4 km per pixel. These weather images are captured 4 times per hour, amounting to one in 15-min intervals.

Some static context variables are also provided, including *elevation* and *longitude/latitude*. This static information is provided in a format of 256 $\times$ 256 pixels, similar to the weather parameters. This information is unique to the specific regions and does not change over time.

The evaluation metric for these challenges (core and transfer challenges) considers the presence of missing values for each variable and attempts to remove the dominance of any variable

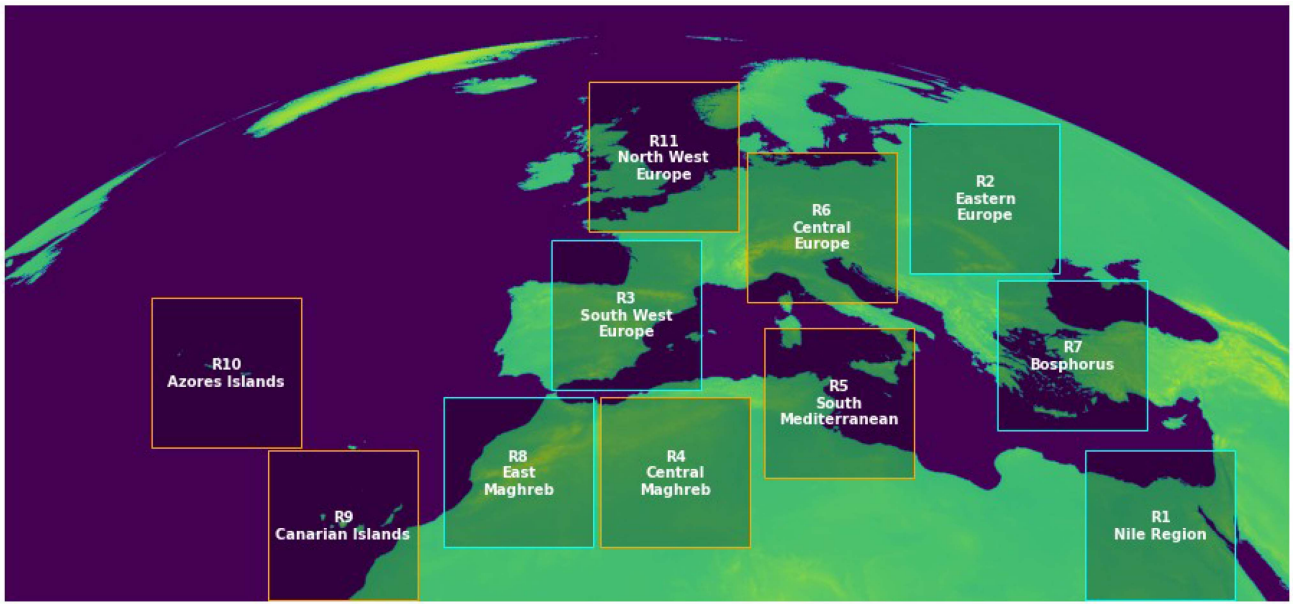

Fig. 5. IEEE Big Data Weather4Cast 2021 Data Localization. The core challenge is shown in blue squares, while the regions in orange squares are for the transfer learning challenge [88].

TABLE I
IEEE BIGDATA WEATHER4CAST COMPOSITION

Weather Product	Weather variables
Temperature at Ground/Cloud (CTTH)	***temperature***, *ctth_tempe*, *ctth_pres*, *ctth_alti*, *ctth_effectiv*, ctth_method, ctth_quality, ishai_skt, ishai_quality
Convective Rainfall Rate (CRR)	*crr*, ***crr_intensity***, *crr_accum*, crr_quality
Probability of Occurrence of Tropopause Folding (ASII)	***asii_turb_trop_prob***, asiitf_quality
Cloud Mask (CMA)	*cma_cloudsnow*, ***cma***, *cma_dust*, *cma_volcanic*, *cma_smoke*, cma_quality
Cloud Type (CT)	*ct*, *ct_cumuliform*, *ct_multilayer*, ct_quality

Training variables in *italic* represent the exogeneous inputs, while while in **bold** are the autogressive terms.

TABLE II
DATA CATEGORIES USED IN MODEL TRAINING

Data type	weather variables
target	temperature, crr_intensity, asii_turb_trop_prob, cma
static	elevation, latitude, longitude
dynamic	ctth_tempe, ctth_press, ctth_effectiv, crr, crr_accum, cma_cloudsnow, cma_dust, cma_volcanic, cma_smoke, ct, ct_cumuliform, ct_multilayer

in the metric calculation. The scaling factor used to remove such (target) variable depending on dominance is given by

$$\text{persistence}(v) = \begin{cases} 0.03163512, & \text{if } v = \text{ temperature} \\ 0.00024158, & \text{if } v = \text{ crr\_intensity} \\ 0.00703378, & \text{if } v = \text{ asii\_turb\_trop\_prob} \\ 0.19160305, & \text{if } v = \text{ cma} \end{cases}. \quad (4)$$

The evaluation metric using such persistence scaling leads to a value of 1 for persistence modeling and is given by

$$\text{Score}_C = \frac{1}{\text{DTR}_C V} \sum_{d=1}^{D=36} \sum_{t=1}^{T=32} \sum_{r \in R_C} \sum_{v \in V} \frac{w(v)}{P_{r,v}} \sum_{p=1}^{P_{r,v}} \times \left(y_{v,p}^{r,d,t} - \bar{y}_{v,p}^{r,d,t} \right) \quad (5)$$

where R_C represents the set of all regions involved for a given challenge, D is the total number of days in the testing set, T is the number of time steps, $w(v) := \frac{1}{\text{persistence}(v)}$ is the scaling factor attributed to each variable v in the set of all target variables V. Also, $P_{r,v} = 256 \times 256 - N_r$ is used to account for missing data for a target variable v in any region r, where N_r is the number of missing data (i.e., empty pixels).

B. Model Training

Pytorch was used to implement the model shown in Fig. 1. In conjunction with the Adam optimizer [89], we use the evaluation metric (5) as the loss function. We trained the model using this loss function in a multitask setting, as the scaling factor involved balances the effects of each variable on the total loss. The learning rate starts at 1e-4 and is gradually reduced by half when the validation set performance plateaus for more than 3 epochs. The model was trained with a dedicated data augmentation scheme for dense prediction purposes. The augmentation pipeline includes random horizontal (*RandomHorizontalFlip*) and vertical (*RandomVerticalFlip*) flipping of the data, as well as random rotations of the data block (90° *RandomRotation*) The model training uses either the expected target data only or together with any combination of additional static and/or dynamic data available (Table II). The parameters of the models

TABLE III
TRAINED MODELS PERFORMANCE COMPARISON

Model	Additional Data		#Parameters	Performance	
version	static	others		Core	Transfer
v2	No	No	5.74M	0.4936	0.4516
v6	Yes	No	5.78M	0.4840	0.4516
v7	No	Yes	5.85M	0.4810	0.4447
v8	Yes	Yes	5.87M	0.4750	0.4420
Top 3 models on the leaderboard					
ConvGRU [66] (2021)	Yes	No	18M	0.4729	0.4323
Dense UNet [69] (2021)	Yes	Yes	12M	0.4802	0.4376
Variational UNet [70] (2021)	No	Yes	16M	0.4857	0.4594

Blue, red, and brown colored results represents the 1st, 2nd and 3rd.

TABLE IV
TRAINED MODEL CONFIGURATIONS

model	Hyperparameters			Additional Data	
version	embed-dim	patch-size	weight decay	static	dynamic
v0	16	2x2	No	No	No
v1	32	2x2	No	No	No
v2	48	2x2	No	No	No
v3	48	2x2	Yes	No	No
v4	48	3x3	Yes	No	No
v5	48	4x4	Yes	No	No
v6	48	4x4	Yes	Yes	No
v7	48	4x4	Yes	No	Yes
v8	48	4x4	Yes	Yes	Yes

Starting with model v3, a weight decaying factor of 1e-6 was included during training to address possible overfitting.

v2, v6, v7, and v8 will be presented later in Table III. The additional data categories (i.e., static and dynamic) considered here are based on the fact that some weather variables are related, e.g., temperature and pressure. Also, the elevation map of a given location has some impact on weather fluctuations [90], [91].

C. Experiments

A forecasting model (Table III) was developed that follows the architecture in Fig. 1 with an embedding dimension of 48 and a patch size of 4. Training of the model configurations involved whether additional data will be used. One of the models was trained using only the target variables, as listed in Section IV-A, while the remaining models include either other dynamic or static data, or a combination of the two together with the target variables (Table III). We used different combinations of data in model training to investigate the predictive capability of combining input data during training.

As shown in Table (III), training with different combinations of inputs results in different model configurations in terms of the number of parameters (i.e., there is a change in the number of input channels).

We compared our models with the best performing models for this dataset in Table III [88]. Our model has the least number of parameters and does not include ensembling, used by other models.

D. Ablation Study

We developed several forecasting models (Table IV) that follow the architecture in Fig. 1 with changes in the hyper-parameters (i.e., embedding dimension, patch size, and weight decay). As previously mentioned, model configuration training involves selecting whether additional data will be used. Some of the models were trained using only the target variables, as listed in Section IV-A, while some models included either other dynamic or static data, or combinations of the two together with the target variables (Tables II and IV).

TABLE V
MODEL PERFORMANCE WITH VARYING EMBEDDING DIMENSION

Model	embed-dim	#Parameters	Core	Transfer
v0	16	688,080	0.5015	0.4572
v1	32	2,574,688	0.4940	0.4530
v2	48	5,708,528	**0.4936**	**0.4516**

The bold entities are to highlight the best-performing models compared to others.

TABLE VI
MODEL PERFORMANCE WITH VARYING PATCH SIZE

Model	patch size	#Parameters	Core	Transfer
v3	2×2	5,708,528	0.5016	0.4516
v4	3×3	5,721,008	0.4974	0.4516
v5	4×4	5,738,480	0.4945	0.4516

TABLE VII
MODEL PERFORMANCE WITH ADDITIONAL DATA

Model	static	dynamic	#Parameters	Core	Transfer
Models with embedding dimension of 48					
v6	Yes	No	5,780,048	0.4840	0.4516
v7	No	Yes	5,847,632	0.4810	0.4516
v8	Yes	Yes	5,866,064	**0.4750**	**0.4420**
Models with embedding dimension of 32					
v9	Yes	No	2,616,256	0.4845	0.4529
v10	No	Yes	2,661,312	*0.4764*	0.4481
v11	Yes	Yes	2,673,600	0.4777	*0.4446*

The bold entities are to highlight the best-performing models compared to others.

1) Embedding Dimension Variation: We trained several models that varied in the size of the embedding dimension, i.e., {16, 32, 48}. These dimensions were chosen to avoid an excessively large number of parameters, as the intention was to develop parameter-efficient models. All these models use a patch size of 2 × 2 without any additional data (static or dynamic) (Table IV). The experimental results in Table V show that we can obtain an improvement in performance by increasing the embedding dimension.

2) Patch Size Variation: We explored the effect of increasing the patch size used in the patch embedding layer of the model (Fig. 1). In this experiment, we use the best performing model of Table V, which uses an embedding dimension of 48. The results of these experiments are shown in Table VI , and indicate that a patch size of 4 achieved the best results. It is worth noting that increasing the patch size has a noticeable effect on performance, while only incurring a very modest increase in the number of parameters.

3) Using Additional Data: Earlier ablation experiments in this study used only the target variables stated in Table II. In the current experiment, we trained the best performing model in Table VI, i.e., with a patch size of 4 × 4 and an embedding dimension of 48, with different combinations of additional data. We also considered models with an embedding dimension of 32 for this experiment to reduce model size, and possibly enhance transferability, and reduce overfitting. However, the experiments demonstrated that reducing the embedding dimension does not lead to improvements in model transfer, instead the model is self resilient to overfitting.

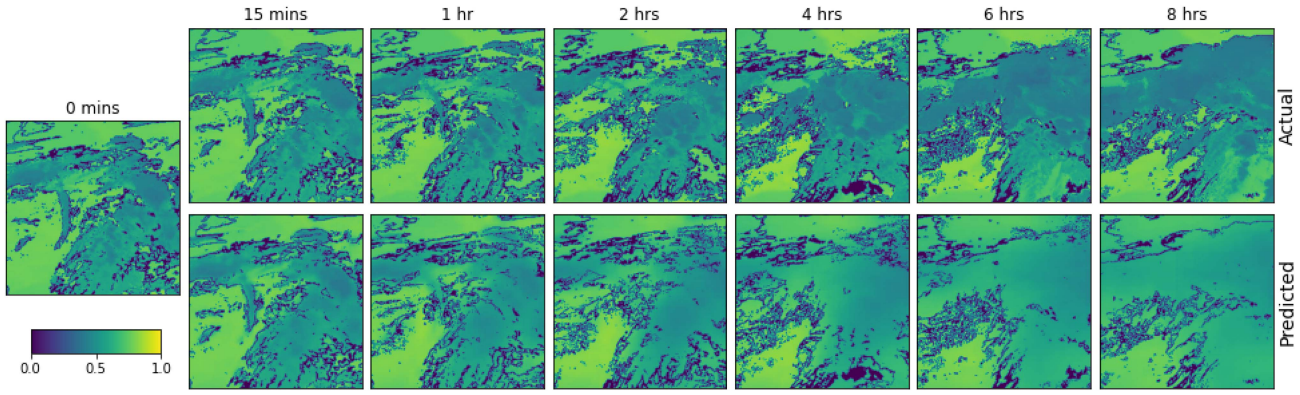

Fig. 6. Comparing the *temperature* forecast with ground truth. The left side shows the situation before forecasting. On the right side, the first row shows the expected values in a sample of future times, while the second row shows the predicted values. Values normalized to the range (0, 1).

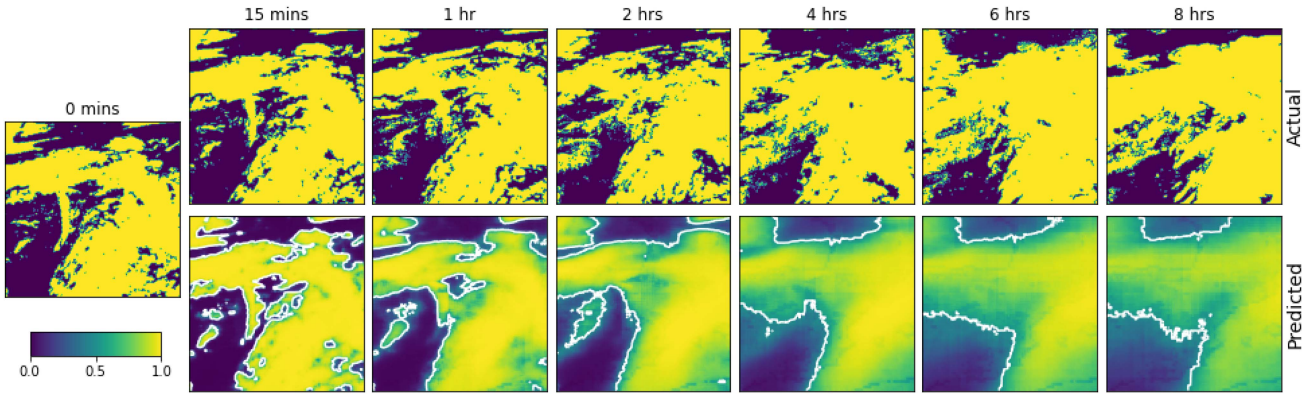

Fig. 7. Comparing the *cma* forecast with ground truth. The left side shows the situation before forecasting. On the right side, the first row shows the expected values in a sample of future times, while the second row shows the predicted values. The contour line in white is used to show the binary threshold at 0.5.

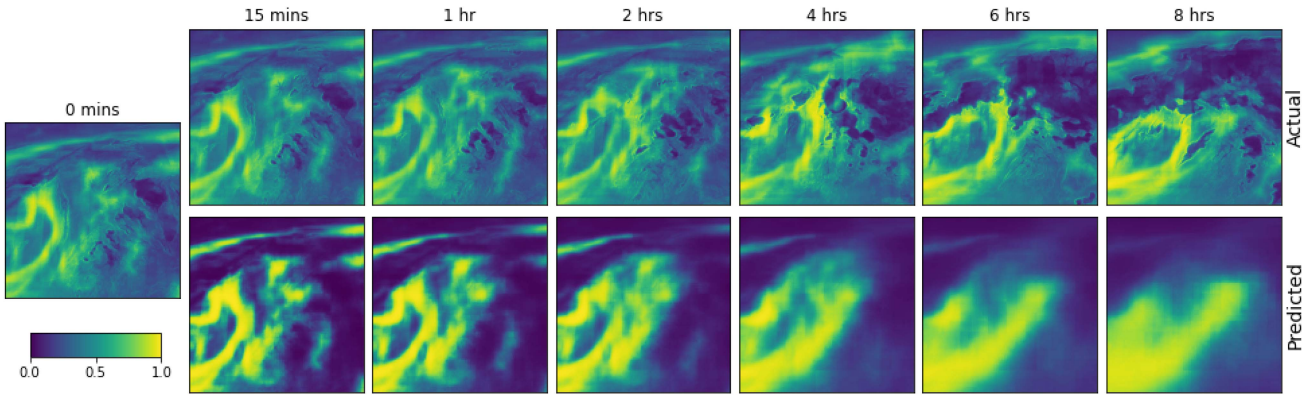

Fig. 8. Comparing the *asii_turb_trop_prob* forecast with ground truth. The left side shows the situation before forecasting. On the right side, the first row shows the expected values in a sample of future times, while the second row shows the predicted values. Values normalized to the range (0, 1).

E. Qualitative Analysis

We conducted a pictorial representation and analysis of the validation data based on the effect of time. Here, the variables are plotted individually to show the trend over the time of forecasting window. The weather conditions were compared to the ground truth, as shown in Figs. 6–9. This analysis shows that the model was able to accurately predict the weather variables quite well in the beginning but its reliability decreases with time. As an example, prediction in Fig. 6 follows the expected value but slight blurriness occurs toward the end of the prediction horizon. This is similar to the observation in the analysis of *cma* (Fig. 7), *asii_turb_trop_prob* (Fig. 8), and *crr_intensity* (Fig. 9). The *crr_intensity* shown in Fig. 9 indicates that the model can learn even in the scarcity of nonzero values.

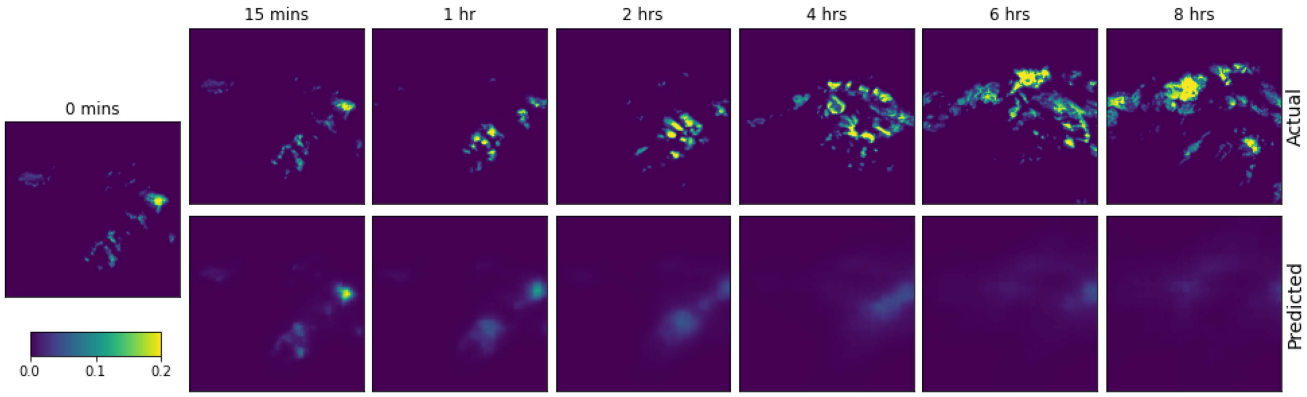

Fig. 9. Comparing the *crr_intensity* forecast with ground truth. The left side shows the situation before forecasting. On the right side, the first row shows the expected values in a sample of future times, while the second row shows the predicted values. Values normalized to the range (0, 1).

V. Conclusion

A short-time weather forecasting model was introduced for the first time that uses the 3-D Swin-transformer in a U-Net architecture, which resulted in competitive results, i.e., a leaderboard score (scaled multitask mse) of *0.4750* and *0.4420*, for the *core* and *transfer* challenges, respectively (IEEE Big Data Weather4Cast2021 [88]). The proposed model has only three blocks of Swin-transformers in both the encoder and decoder parts. It uses cross-attention in the decoder to merge data from the encoder with the upsampled decoding data. This ensures that the model focuses only on important information. We intend to investigate different types of attention layers in the future. Similarly, we intend to investigate token mixing using hypercomplex networks, e.g., sedenion networks [21].

References

[1] X. Ren et al., "Deep learning-based weather prediction: A survey," *Big Data Res.*, vol. 23, 2021, Art. no. 100178.
[2] D. N. Fente and D. Kumar Singh, "Weather forecasting using artificial neural network," in *Proc. 2nd Int. Conf. Inventive Commun. Comput. Technol.*, 2018, pp. 1757–1761.
[3] B. Wang et al., "Deep uncertainty quantification: A machine learning approach for weather forecasting," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, New York, NY, USA, 2019, pp. 2087–2095.
[4] N. Singh, S. Chaturvedi, and S. Akhter, "Weather forecasting using machine learning algorithm," in *Proc. Int. Conf. Signal Process. Commun.*, 2019, pp. 171–174.
[5] P. Hewage et al., "Temporal convolutional neural (TCN) network for an effective weather forecasting using time-series data from the local weather station," *Soft Comput.*, vol. 24, no. 21, pp. 16453–16482, Nov. 2020.
[6] P. Hewage, M. Trovati, E. Pereira, and A. Behera, "Deep learning-based effective fine-grained weather forecasting model," *Pattern Anal. Appl.*, vol. 24, no. 1, pp. 343–366, Feb. 2021.
[7] M. Qiu et al., "A short-term rainfall prediction model using multi-task convolutional neural networks," in *Proc. IEEE Int. Conf. Data Mining*, 2017, pp. 395–404.
[8] R. Castro, Y. M. Souto, E. Ogasawara, F. Porto, and E. Bezerra, "STConvS2S: Spatiotemporal convolutional sequence to sequence network for weather forecasting," *Neurocomputing*, vol. 426, pp. 285–298, 2021.
[9] K. Yonekura, H. Hattori, and T. Suzuki, "Short-term local weather forecast using dense weather station by deep neural network," in *Proc. IEEE Int. Conf. Big Data*, 2018, pp. 1683–1690.
[10] C.-J. Zhang, X.-J. Wang, L.-M. Ma, and X.-Q. Lu, "Tropical cyclone intensity classification and estimation using infrared satellite images with deep learning," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 14, pp. 2070–2086, 2021.
[11] M. Hossain, B. Rekabdar, S. Louis, and S. Dascalu, "Forecasting the weather of Nevada: A deep learning approach," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, 2015, pp. 1–6.
[12] S.-Y. Lin, C.-C. Chiang, J.-B. Li, Z.-S. Hung, and K.-M. Chao, "Dynamic fine-tuning stacked auto-encoder neural network for weather forecast," *Future Gener. Comput. Syst*, vol. 89, pp. 446–454, 2018.
[13] C. K. Sønderby et al., "MetNet: A neural weather model for precipitation forecasting," 2020, *arXiv:2003.12140*.
[14] Z. Karevan and J. A. K. Suykens, "Spatio-temporal stacked LSTM for temperature prediction in weather forecasting," 2018, *arXiv:1811.06341*.
[15] S. Hou et al., "D2CL: A dense dilated convolutional LSTM model for sea surface temperature prediction," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 14, pp. 12514–12523, 2021.
[16] T. Xiong, J. He, H. Wang, X. Tang, Z. Shi, and Q. Zeng, "Contextual sa-attention convolutional LSTM for precipitation nowcasting: A spatiotemporal sequence forecasting view," IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens., vol. 14, pp. 12479–12491, 2021.
[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Adv. in Neural Inf. Process. Syst.*, F. Pereira, C. J.C. Burges, L. Bottou, and K. Q. Weinberger, Eds., vol. 25. Red Hook, NY, USA: Curran Associates, Inc., 2012.
[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.
[19] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 2261–2269.
[20] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds. Berlin, Germany: Springer, 2015, pp. 234–241.
[21] A. Bojesomo, P. Liatsis, and H. Al-Marzouqi, "Traffic flow prediction using deep sedenion networks," 2020, *arXiv:2012.03874*.
[22] S. Choi, "Utilizing UNet for the future traffic map prediction task traffic4cast challenge 2020," 2020, *arXiv:2012.00125*.
[23] M. Kopp et al., "Traffic4cast at NeurIPS 2020 - yet more on the unreasonable effectiveness of gridded geo-spatial processes," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, H. J. Escalante and K. Hofmann, Eds., 2021, pp. 325–343.
[24] A. Dosovitskiy et al., "An image is worth 16x16 words: Transformers for image recognition at scale," 2021, *arXiv:2010.11929*.
[25] W. Wang et al., "Pyramid vision transformer: A versatile backbone for dense prediction without convolutions," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 568–578.
[26] R. Ranftl, A. Bochkovskiy, and V. Koltun, "Vision transformers for dense prediction," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Los Alamitos, CA, USA, 2021, pp. 12159–12168.
[27] Z. Dai, H. Liu, Q. V. Le, and M. Tan, "CoatNet: Marrying convolution and attention for all data sizes," in *Adv. in Neural Inf. Process. Syst.*, M. Ranzato, A. Y. Beygelzimer, P. Dauphin Liang, and J. W. Vaughan, Eds. Red Hook, NY, USA: Curran Associates, Inc., 2021, pp. 3965–3977.

[28] X. Dong et al., "CSWin transformer: A general vision transformer backbone with cross-shaped windows," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 12124–12134.

[29] Z. Liu et al., "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 9992–10002.

[30] Z. Liu et al., "Video Swin transformer," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 3202–3211.

[31] W. Wang, L. Yao, L. Chen, D. Cai, X. He, and W. Liu, "CrossFormer: A versatile vision transformer based on cross-scale attention," CoRR, vol. abs/2108.00154, 2021. [Online]. Available: https://arxiv.org/abs/2108.00154

[32] A. Vaswani et al., "Attention is all you need," in *Adv. in Neural Inf. Process. Syst.*, I. Guyon et al., Eds., vol. 30, Red Hook, NY, USA: Curran Associates, Inc., 2017.

[33] H. Cao et al., "Swin-UNet: UNet-like pure transformer for medical image segmentation," in *Proc. Eur. Conf. Comput. Vis.*, Springer, 2022, pp. 205–218.

[34] A. Bojesomo, H. Al-Marzouqi, and P. Liatsis, "Spatiotemporal vision transformer for short time weather forecasting," in *Proc. IEEE Int. Conf. Big Data*, 2021, pp. 5741–5746.

[35] A. Bojesomo, H. Al-Marzouqi, and P. Liatsis, "Spatiotemporal Swin-transformer network for short time weather forecasting," in *Proc. Workshops Colocated 30th ACM Int. Conf. Inf. Knowl. Manage.*, 2021, pp. 1–5.

[36] A. Bojesomo, H. Al-Marzouqi, and P. Liatsis, "SwinUNet3D—A hierarchical architecture for deep traffic prediction using shifted window transformers," 2022, *arXiv:2201.06390*.

[37] R. Goyal et al., "The "something something" video database for learning and evaluating visual common sense," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 5842–5850.

[38] W. Kay et al., "The kinetics human action video dataset," 2017, *arXiv:1705.06950*.

[39] T. Ji, Y. Jiang, M. Li, and Q. Wu, "Ultra-short-term wind speed and wind power forecast via selective Hankelization and low-rank tensor learning-based predictor," *Elect. Power Energy Syst.*, vol. 140, 2022, Art. no. 107994.

[40] J. Gillard and K. Usevich, "Structured low-rank matrix completion for forecasting in time series analysis," *Int. J. Forecasting*, vol. 34, pp. 582–597, Oct./Dec. 2018.

[41] G. Liu, "Time series forecasting via learning convolutionally low-rank models," *IEEE Trans Inf. Theory*, vol. 68, no. 5, pp. 3362–3380, May 2022.

[42] J. Gasthaus et al., "High-dimensional multivariate forecasting with low-rank Gaussian Copula processes," in *Proc. 33rd Conf. Neural Inf. Process. Syst.*, 2019, pp. 6827–6837.

[43] P. Bauer, A. Thorpe, and G. Brunet, "The quiet revolution of numerical weather prediction," *Nature*, vol. 525, pp. 47–55, 2015.

[44] I. Sandu, A. Beljaars, P. Bechtold, T. Mauritsen, and G. Balsamo, "Why is it so difficult to represent stably stratified conditions in numerical weather prediction (NWP) models?," *J. Adv. Model. Earth Syst.*, vol. 5, no. 2, pp. 117–133, 2013.

[45] A. Abraham, N. Philip, B. Nath, and P. Saratchandran, "Performance analysis of connectionist paradigms for modeling chaotic behavior of stock indices," in *Proc. 2nd Int. Workshop Intell. Syst. Des. Appl.*, 2002, pp. 181–186.

[46] C. Voyant, M. Muselli, C. Paoli, and M.-L. Nivet, "Numerical weather prediction (NWP) and hybrid ARMA/ANN model to predict global radiation," *Energy*, vol. 39, no. 1, pp. 341–355, 2012.

[47] J. Pathak, B. Hunt, M. Girvan, Z. Lu, and E. Ott, "Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach," *Phys. Rev. Lett.*, vol. 120, no. 2, 2018, Art. no. 024102.

[48] C. He, J. Wei, Y. Song, and J.-J. Luo, "Seasonal prediction of summer precipitation in the middle and lower reaches of the Yangtze river valley: Comparison of machine learning and climate model predictions," *Water*, vol. 13, no. 22, 2021, Art. no. 3294.

[49] B. Bochenek and Z. Ustrnul, "Machine learning in weather prediction and climate analyses–Applications and perspectives," *Atmosphere*, vol. 13, no. 2, 2022, Art. no. 180.

[50] E. Juarez and M. Petersen, "A comparison of machine learning methods to forecast tropospheric ozone levels in Delhi," *Atmosphere*, vol. 13, no. 1, 2022, Art. no. 46.

[51] M. Achite, M. Jehanzaib, N. Elshaboury, and T.-W. Kim, "Evaluation of machine learning techniques for hydrological drought modeling: A case study of the Wadi Ouahrane basin in Algeria," *Water*, vol. 14, no. 3, 2022, Art. no. 431.

[52] Y. Fang, H. Chen, Y. Lin, C. Zhao, Y. Lin, and F. Zhou, "Classification of northeast China cold vortex activity paths in early summer based on k-means clustering and their climate impact," *Adv. Atmospheric Sci.*, vol. 38, no. 3, pp. 400–412, 2021.

[53] D. Lou et al., "K-means and C4.5 decision tree based prediction of long-term precipitation variability in the Poyang Lake Basin, China," *Atmosphere*, vol. 12, no. 7, 2021, Art. no. 834.

[54] J. Abbot and J. Marohasy, "The application of machine learning for evaluating anthropogenic versus natural climate change," *GeoResJ*, vol. 14, pp. 36–46, 2017.

[55] W. Almikaeel, L. Čubanová, and A. Šoltész, "Hydrological drought forecasting using machine learning–Gidra river case study," *Water*, vol. 14, no. 3, 2022, Art. no. 387.

[56] M. Ma et al., "HistGNN: Hierarchical spatio-temporal graph neural networks for weather forecasting," *Inf. Sci.*, vol. 48, 2023, Art. no. 119580.

[57] B. P. Shukla, C. M. Kishtawal, and P. K. Pal, "Prediction of satellite image sequence for weather nowcasting using cluster-based spatiotemporal regression," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 7, pp. 4155–4160, Jul. 2014.

[58] "Day-ahead spatiotemporal solar irradiation forecasting using frequency-based hybrid principal component analysis and neural network," *Appl. Energy*, vol. 247, pp. 389–402, 2019.

[59] L. Huang, J. Kang, M. Wan, L. Fang, C. Zhang, and Z. Zeng, "Solar radiation prediction using different machine learning algorithms and implications for extreme climate events," *Front. Earth Sci.*, vol. 9, 2021, Art. no. 202.

[60] D. Niu, L. Diao, Z. Zang, H. Che, T. Zhang, and X. Chen, "A machine-learning approach combining wavelet packet denoising with catboost for weather forecasting," *Atmosphere*, vol. 12, no. 12, 2021, Art. no. 1618.

[61] V. Monego, J. Anochi, and H. de Campos Velho, "South America seasonal precipitation prediction by gradient-boosting machine-learning approach," *Atmosphere*, vol. 13, no. 2, 2022, Art. no. 243.

[62] L. Iverson, A. Prasad, and A. Liaw, "New machine learning tools for predictive vegetation mapping after climate change: Bagging and random forest perform better than regression tree analysis," in *Proc12th Annu. Conf. Landscape Ecol. Trees Forests*, Cirencester, U.K, 2004, pp. 317–320.

[63] P.-S. Yu, T.-C. Yang, S.-Y. Chen, C.-M. Kuo, and H.-W. Tseng, "Comparison of random forests and support vector machine for real-time radar-derived rainfall forecasting," *J. Hydrol.*, vol. 552, pp. 92–104, 2017.

[64] C. Song, X. Chen, P. Wu, and H. Jin, "Combining time varying filtering based empirical mode decomposition and machine learning to predict precipitation from nonlinear series," *J. Hydrol.*, vol. 603, 2021, Art. no. 126914.

[65] J. N. Liu, Y. Hu, J. J. You, and P. W. Chan, "Deep neural network based feature representation for weather forecasting," in *Proc. Int. Conf. Artif. Intell.*, 2014, pp. 1–7.

[66] J. Leinonen, "Improvements to short-term weather prediction with recurrent-convolutional networks," in *Proc. IEEE Int. Conf. Big Data*, 2021, pp. 5764–5769.

[67] A.-I. Albu, G. Czibula, A. Mihai, I. Czibula, and S. Burcea, "NextNow: A convolutional deep learning model for the prediction of weather radar data for noacasting purposes," *Remote Sens.*, vol. 14, no. 16, 2022, Art. no. 3890.

[68] A. Chattopadhyay, M. Mustafa, P. Hassanzadeh, E. Bach, and K. Kashinath, "Towards physics-inspired data driven weather forecasting: Integrating data assimilation with a deep spatial-transformer-based u-net in a case study with era5," *Geoscientific Model Develop.*, vol. 15, pp. 2221–2237, 2022.

[69] S. Choi, "UNet based future weather prediction on Weather4cast 2021 stage 2," in *Proc. IEEE Int. Conf. Big Data*, 2021, pp. 5747–5749.

[70] P. H. Kwok and Q. Qi, "Enhanced variational U-Net for weather forecasting," in *Proc. IEEE Int. Conf. Big Data*, 2021, pp. 5758–5763.

[71] S. Ravuri et al., "Skilful precipitation nowcasting using deep generative models of radar," *Nature*, vol. 597, no. 7878, pp. 672–677, 2021.

[72] C. Wang, P. Wang, P. Wang, B. Xue, and D. Wang, "Using conditional generative adversarial 3-D convolutional neural network for precise radar extrapolation," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 14, pp. 5735–5749, 2021.

[73] C. Li and X. Chen, "Future video frame prediction based on generative motion-assistant discriminative network," *Appl. Soft Comput.*, vol. 135, 2023, Art. no. 110028.

[74] X. Chen et al., "Weather radar nowcasting for extreme precipitation prediction based on the temporal and spatial generative adversarial network," *Atmosphere*, vol. 13, no. 8, 2022, Art. no. 1291.

[75] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-C. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *Proc. 28th Int. Conf. Neural Inf. Process. Syst.*, Cambridge, MA, USA, 2015, pp. 802–810.
[76] C.-A. Diaconu, S. Saha, S. Guennemann, and X. Zhu, "Understanding the role of weather data for Earth surface forecasting using a convLSTM-based model," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 1361–1370.
[77] Y. Wang et al., "PredRNN: Recurrent neural networks for predictive learning using spatiotemporal LSTMs," in *Adv. in Neural Inf. Process. Syst.*, vol. 30, Red Hook, NY, USA: Curran Associates, Inc., 2017.
[78] Y. Wang, Z. Gao, M. Long, J. Wang, and P. S. Yu, "PredRNN : Towards a resolution of the deep-in-time dilemma in spatiotemporal predictive learning," in *Proc. 35th Int. Conf. Mach. Learn.*,, Jul. 2018, pp. 5123–5132.
[79] X. Shi et al., "Deep learning for precipitation nowcasting: A benchmark and a new model," in *Proc. 31st Conf. Neural Inf. Process. Syst.*., 2017, pp. 5622–5632.
[80] X. Zhang, Q. Jin, S. Xiang, and C. Pan, "MFNet: The spatio-temporal network for meteorological forecasting with architecture search," *IEEE Geosci. Remote Sens. Lett.*, vol. 19, 2022, Art. no. 1006605.
[81] H. Wu, H. Zhou, M. Long, and J. Wang, "Interpretable weather forecasting for wordwide stations with a unified deep model," *Nature Mach. Intell.*, vol. 5, pp. 602–611, 2023.
[82] S. Huang, Z. Lu, R. Cheng, and C. He, "FaPN: Feature-aligned pyramid network for dense image prediction," in *Proc. Int. Conf. Comput. Vis.*, 2021, pp. 844–853.
[83] T. Xiao, Y. Liu, B. Zhou, Y. Jiang, and J. Sun, "Unified perceptual parsing for scene understanding," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 1–17.
[84] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "SegFormer: Simple and efficient design for semantic segmentation with transformers," in *Proc. Adv. Neural Inf. Process. Syst.*, A. Y. Beygelzimer, P. Dauphin Liang, and J. W. Vaughan, Eds., 2021, pp. 1–17.
[85] L. J. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," 2016, *arXiv:1607.06450*.
[86] A. Chaurasia and E. Culurciello, "LinkNet: Exploiting encoder representations for efficient semantic segmentation," in *Proc. IEEE Vis. Commun. Image Process.*, 2017, pp. 1–7.
[87] A. Gruca et al., "Cdceo'21: First workshop on complex data challenges in Earth observation," in *Proc. 30th ACM Int. Conf. Inf. Knowl. Manage.*, New York, NY, USA, 2021, pp. 4878–4879.
[88] "Multi-sensor weather forecast competition: IEEE Big Data cup 2021 Leaderboard," 2021. [Online]. Available: https://www.iarai.ac.at/weather4cast/competitions/ieee-big-data-core-final/?leaderboard
[89] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
[90] I. Rangwala and J. R. Miller, "Climate change in mountains: A review of elevation-dependent warming and its possible causes," *Climatic Change*, vol. 114, no. 3, pp. 527–547, 2012.
[91] M. Kuhn and M. Olefs, "Elevation-dependent climate change in the European Alps," *Oxford Res. Encyclopedia Climate Sci.*, 2020. [Online]. Available: https://oxfordre.com/climatescience/view/10.1093/acrefore/9780190228620.001.0001/acrefore-9780190228620-e-762

Alabi Bojesomo received the bachelor's degree (summa cum laude) in electronic and electrical engineering from Obafemi Awolowo University, Ife, Nigeria, in 2011, and the M.Sc. degree in microsystems engineering from Khalifa University, Abu Dhabi, United Arab Emirates, in 2016. He is currently working toward the Ph.D. degree with the Department of Electrical Engineering and Computer Science, Khalifa University of Science and Technology, Abu Dhabi, United Arab Emirates.

His research interests include deep learning, hardware security and spatiotemporal forecasting.

Hasan AlMarzouqi (Senior Member, IEEE) received the bachelor's degree (with honors) and the M.Sc. degree in electrical and computer engineering from Vanderbilt University, Nashville, TN, USA, in 2004 and 2006, respectively, and the Ph.D. degree in electrical and computer engineering from the Georgia Institute of Technology, Atlanta, GA, USA, in 2014.

He is currently an Assistant Professor with the Department of Electrical Engineering and Computer Science, Khalifa University of Science and Technology, Abu Dhabi, United Arab Emirates. His research interests include deep learning, artificial intelligence, digital rock physics, and bioinformatics.

Dr. Al-Marzouqi is a member of the IEEE Signal Processing Society.

Panos Liatsis (Senior Member, IEEE) received the diploma in electrical engineering from the University of Thrace, Campus, Greece, and the Ph.D. degree in electrical engineering and electronics from the University of Manchester, Manchester, U.K., in 1990 and 2002.

He commenced his academic career with the University of Manchester, before joining the City University of London, London, U.K., in 2003, where he was a Professor and Head of the Electrical and Electronic Engineering Department . He is currently a Professor with the Department of Electrical Engineering and Computer Science, Khalifa University of Science and Technology, Abu Dhabi, United Arab Emirates. His research interests include image processing, computer vision, pattern recognition, and machine learning.