Spatiotemporal Swin-Transformer Network for Short Time Weather Forecasting

Alabi Bojesomo, Hasan Al-Marzouqi and Panos Liatsis

Electrical Engineering and Computer Science Department, Khalifa University, Abu Dhabi, UAE.

Abstract

Earth Observatory is a growing research area that is using AI for short time forecasting, a Now-casting scenario. In this work, we tackle the challenge of weather forecasting using the video transformer network. In recent times, many variants of the vision transformer were explored, with major constraints being the computational complexity of Attention and the data hungry training. We explore the use of Video Swin-Transformer together with a carefully crafted augmentation scheme to tackle the data hungry transformer network. In addition, we use a gradual spatial reduction on the encoder side and cross-attention on the decoder. The proposed network is tested on the Weather4Cast2021 weather forecasting challenge data, which requires the prediction of 8 hours ahead future frames (4 per hour) from an hour weather product sequence. The model results in a highly competitive performance on both the validation and test datasets. The code is available online at *https://github.com/bojesomo/Weather4cast2021-SwinEncoderDecoder*.

Keywords

Video Swin-Transformer, Encoder-Decoder Video Architecture, Now-casting, Weather forecasting

1. Introduction

Weather forecasting is an important requirement in autonomous vehicles and food production [1], owing to the relationship between successful implementation of these applications and accurate weather prediction. For instance, knowledge of the weather is an important aspect of the location context, when designing autonomous navigation and collision avoidance systems. In food production, weather forecasting has already proven to be an important factor for crop yield management and adequate soil nutrient replenishment. Deep learning has been widely used in short time weather forcasting including long short-time memory (LSTM) [2, 3], Autoencoders [4, 5], CNN [6, 7, 3] and deeply connected neural networks [8].

Due to the success of attention based networks in natural language processing (NLP), many researchers have recently shifted their attention in exploring their use in computer vision, including image classification, object detection and semantic segmentation [9, 10, 11, 12]. A pioneering work in this area, which uses patch based image encoding, is the vision transformer [10]. *Dosovitskiy et. al.* [10], which carefully laid down the techniques for encoding image patches similar to word embedding in NLP, followed by one-to-one use of a transformer network as proposed by *Vaswani et. al.* [9]. Many researchers explored this direction by proposing highly efficient attention networks, external attention, and pyramid vision transformer among others. Dense prediction, including semantic segmentation, has been also investigated using vision transformer as backbone [12, 11].

Swin Transformer is among the promising Vision transformer architectures, which explore a carefully crafted patch information mixing using the shift window attention technique (hence the name Swin) [13]. The method resulted in high performing image recognition models using the vision transformer. In order to leverage its success in classification, researchers explored the Swin transformer as backbone for dense prediction. *Cao et. al.* used Swin transformer blocks in the encoder, bottleneck and decoder branch of a UNet structure [14]. The work also introduced the patch expanding layer, i.e., the opposite of the patch merging layer, used in the pyramid vision transformer [14]. The 3-dimensional (3D) variant of the Swin transformer (Video Swin transformer) was proposed by *Liu et. al.* [15]. This paper used 3D patch embedding, 3D shifted window multi-head self attention as well as patch merging, and the proposed approach led to a parameter efficient network with strong performance in a variety of datasets (Kinetics-400, kinetic-600, and Something-Something v2) [15].

In this research, we propose a number of improvements in the video Swin transformer [15], including 3D patch expanding, and using cross attention block in the decoder, as well as a carefully designed data augmentation process, which removes the need for pre-training the network on a large dataset. The proposed architecture and layers are given in Section 2, while the experimental results of the proposed solution in weather4cast2021 stage-1 challenge is presented in Section 3.

CIKM 2021: 1st Workshop on Complex Data Challenges in Earth Observation, Nov 01, 2021

100046384@ku.ac.ae (A. Bojesomo); hasan.almarzouqi@ku.ac.ae (H. Al-Marzouqi); panos.liatsis@ku.ac.ae (P. Liatsis)

2. Methods

2.1. Model Architecture

Weather forecasting can be viewed as a sequence-to-sequence task. To this end, we employed transformer models, which were used in NLP with very promising results. The proposed model uses multiple stages for gradual spatial dimensional reduction in the encoder. This dimensional reduction is important so as to capture salient representations of global features. While the encoder uses self attention, the decoder uses self attention for its main input and merges the skip connected input from the encoder using mixed attention [9]. The attention layer used in this research is the shifted window attention proposed in [15].

As shown in Fig 1, the input goes through a 3D patch embedding layer, which forms the token provided to the transformer architecture. Output tokens are expanded and projected back to the original format. The model includes three encoder-decoder blocks, each having four 3D transformer layers (encoder/decoder). We limited the number of blocks to three to better handle the data demanding nature of transformers as we do not pre-train our model on any other dataset [13, 15, 10, 12]. Likewise, we used equal number of transformer layers per block for simplicity (four in our case).

The building blocks of our model shown in fig. (1a) are detailed below.

2.2. Swin Transformer Block

The transformer layer proposed by *Vaswani et. al.* for NLP includes standard multi-head self attention (MSA), followed by a feed-forward network (MLP). Each of these layers is preceded by Layer Normalization (LN) in Vision transformer [10], as opposed to post normalization used in NLP [9]. In this research, 3D shifted window MSA is employed, owing to the spatiotemporal nature of the input. The Swin transformer uses an interchange of sliding windows, as shown in Fig (1b and c) with a window (local) attention, followed by another local but shifted window attention. With such a setup, any two layers of attention follow (1):

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

where LN, MLP, W-MSA and SW-MSA represent layer normalization [16], multilayer perceptron, windowed multi-head self-attention and shifted window multi-head self-attention, respectively. The main difference between W-MSA and SW-MSA is the shift in window positioning, prior to computing local attention within the windowed blocks. Also, relative position bias is used in both W-MSA and SW-MSA [13, 15]. Following the work of *Li et. al.*, we used a window size of (1, 7, 7), shift size of 2 in our implementation of the 3D Swin transformer block [15]. For the MLP, we used two fully-connected layers with a ratio of four for the hidden features (eqn. 2).

$$\text{MLP}(X) = (XW_1)W_2 \tag{2}$$

where $X \in \Re^{\cdots \text{ x } d}$ is 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 the second (output) fully-connected layer.

2.3. Patch merging layer

This layer concatenates the features of each group of 2 × 2 neighboring patches, and applies a fully-connected layer on the 4C-dimensional concatenated features to get 2C-dimensional output [13, 15]. This results in a learned down-sampling operation.

2.4. Patch expanding layer

Contrary to the patch merging layer introduced in the Swin tranformer [13, 15], we used a fully connected layer to scale up the dimension of the incoming data. This results in a learned up-sampling operation.

2.5. Cross Attention

For the decoding blocks of the architecture, we used an attention block to merge the skip-connection of the encoder to the decoding input. The skip-connected input is used as the *key* and *value* parameters, while the decoding input form the *query*.

2.6. Encoder

The encoder backbone network in our model includes a multi-stage Video Swin transformer. Specifically, we use three stages, each having four 3D transformer blocks, followed by a patch merging layer [13, 15]. The attention layer used here is the multi-head self attention as explained in section (2.2) [15].

2.7. Decoder

Here, we replaced the self attention in the encoder with a cross attention layer for feature mixing. The skip connection from the encoder serves as the key parameter K, and value parameter V , while the continuing input from the patch expanding layer serves as the query parameter Q in the attention block (fig 1a).

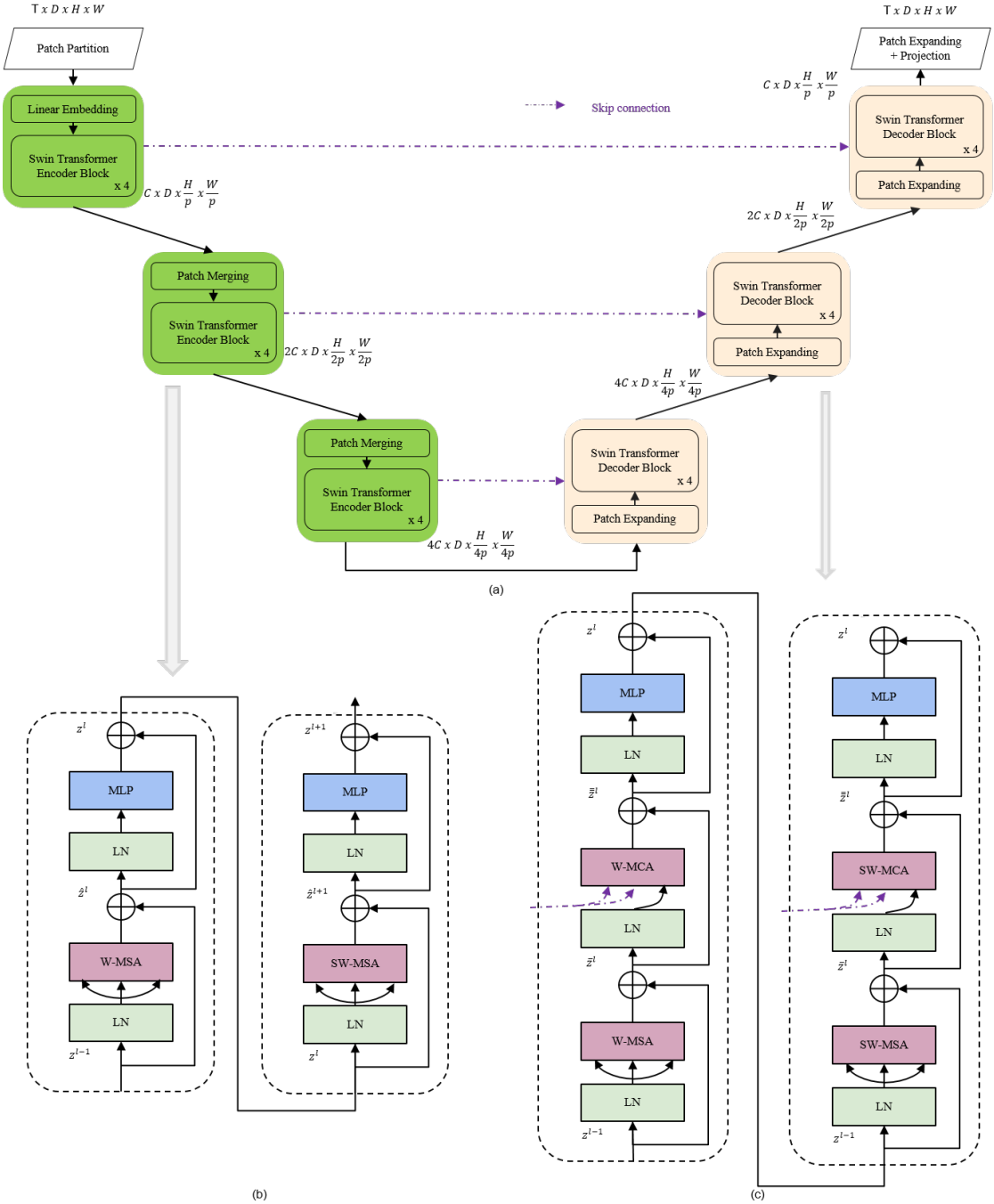

Figure 1: Details of the proposed Spatiotemporal Encoder-Decoder architecture. The network, shown in (a), has three transformer blocks and decoder blocks, respectively. Each of the encoder blocks has a Patch Merging unit, followed by a number of Swin-Transformers except for the first encoder, which has linear embedding. In (b), the stacking of two concurrent self-attention blocks used in the encoder is illustrated, where windowed attention always follows shifted window attention. In (c), the cross-attention used in the decoder is displayed. The input from the skip connection and the decoding input are merged in a cross-attention layer, instead of applying concatenation, which is commonly used in UNet architectures.

2.8. Neck

Although not considered in Fig (1a), our proposed model considered the possibility of linking the encoder to the decoder via a *neck* block. When considered, We use two Transformer blocks without a preceding patch merging layer as the *neck* [14]. Table (1) shows some models where we either use a neck or not in our experiments.

2.9. Prediction Head

This is used to project the output of the last decoder block (final output) to the expected dimensions and format. We employed a patch expanding layer for possible recovering of the spatial dimension that may have been updated by the multi-stage encoder-decoder network. Finally, a fully connected layer is used to project the features into the final dimension.

3. Experimental Results

Data Description : We used the datasets in the Weather4Cast2021 "stage-1" challenge for evaluation purposes [17]. This competition has two challenges:

- Core Challenge: data contain the training, validation, and test sets for three regions {R1 – Nile region (covering Cairo), R2 – Eastern Europe (covering Moscow), and R3 – South West Europe (covering Madrid and Barcelona)}
- Transfer Challenge: data contain only the test set for three additional regions {R4 – Central Maghreb (Timimoun), R5 – South Mediterranean (covering Tripoli and Tunis), and R6 – Central Europe (covering Berlin),}

Weather parameters including temperature (on accessible surfaces: top cloud or earth), convective rainfall rate, probability of occurrence of tropopause folding, and cloud mask are selected as target variables for the competition. Each weather image contains 256 x 256 pixels, with each pixel corresponding to an area of about 4 km x 4 km. The images were recorded at 15 minute intervals throughout the year.

Model Training : The model described in Fig. 1 was implemented in Pytorch. The mean squared error (MSE) was used as the loss function, with the Adam optimizer [18]. The learning rate was initially set to 1e-4 and was manually reduced to 1e-7, when performance plateaued on the validation set. Our model was trained with carefully considered data augmentation for segmentation purposes. Specifically, we use *RandomHorizontalFlip* and *RandomVerticalFlip*, which ensures that we can leverage data augmentation without making any change in the data presented except flipping. This is important as we train a single model with data from the three provided regions (R1, R2, R3), while we tested the model on all available regions (R1-R6) to account for the transfer learning challenge.

As shown in Table (2), the proposed model resulted in an MSE of *0.5337* and *0.4959*, for the *core* and *transfer* challenges, respectively, with only *688,080* parameters. Moreover, this is the first instance of using vision transformers in spatiotemporal forecasting. As shown in Table (1) , we equally trained another model with an embedding dimension of 32 but this could not be tested on the leaderboard due to restriction on number of model that can be submitted. We considered the use of a *neck* in our *UNet-like* model resulting in a conclusion that this does not help our model to improve (Table 1). In Table (2) , we compared our model with the baseline models (*persistence* and *Unet*) provided on the leaderboard which shows that our model performs better with less than 700,000 parameters. In Tables (1), it is important to note that the *validation* MSE is computed on the validation set (Region R1-R6 [17]) while the *core* and *transfer* MSE in Table (2) are computed using a specially crafted MSE computation which puts a segmentation mask into consideration [17].

Table 1
Validation results of our various model configurations on the Weather4Cast2021 Data

dim	neck	#Parameters	Validation MSE
16	False	688,080	0.0297
16	True	790,752	0.0299
32	False	2,574,688	0.0298
32	True	2,622,000	0.0305

Table 2
Comparing our result with baseline models' results on Weather4Cast2021 Data

Method	Core MSE	Transfer MSE
Persistence baseline	1	1
Unet-baseline	0.6688	0.6111
SwinNet3D(Ours)	0.5337	0.4959

4. Conclusions and Future Work

We presented the first ever use of 3D Swin-Transformer in a UNet architecture for short time spatiotemporal forecasting, which resulted in competitive results, i.e., an MSE of *0.5337* and *0.4959*, for the *core* and *transfer* challenges (Weather4Cast2021 [17]), respectively, with only *688,080* parameters. The model having only three blocks of four Swin-transformers in both encoder and decoder

was implemented in PyTorch and trained using *Pytorch-Ligthning* [19]. As transformer architecture is still relatively new to vision domain, we plan to explore other variants of attention layers in the future. Likewise, we equally plan to explore token mixing using hypercomplex networks like sedenion [20]. The code with the implementation of the proposed approach is available online at *https://github.com/bojesomo/Weather4cast2021-SwinEncoderDecoder*.

Acknowledgments

This work was supported by the ICT Fund, Telecommunications Regulatory Authority (TRA), Abu Dhabi, United Arab Emirates.

References

[1] X. Ren, X. Li, K. Ren, J. Song, Z. Xu, K. Deng, X. Wang, Deep learning-based weather prediction: A survey, Big Data Research 23 (2021) 100178. URL: https://www.sciencedirect.com/science/article/pii/S2214579620300460. doi:https://doi.org/10.1016/j.bdr.2020.100178.

[2] Z. Karevan, J. A. K. Suykens, Spatio-temporal stacked lstm for temperature prediction in weather forecasting, 2018. arXiv:1811.06341.

[3] C. K. Sønderby, L. Espeholt, J. Heek, M. Dehghani, A. Oliver, T. Salimans, S. Agrawal, J. Hickey, N. Kalchbrenner, Metnet: A neural weather model for precipitation forecasting, 2020. arXiv:2003.12140.

[4] M. Hossain, B. Rekabdar, S. Louis, S. Dascalu, Forecasting the weather of nevada: a deep learning approach, in: International Joint Conference on Neural Networks, IJCNN, IEEE, 2015, p. 1–6.

[5] S.-Y. Lin, C.-C. Chiang, J.-B. Li, Z.-S. Hung, K.-M. Chao, Dynamic fine-tuning stacked auto-encoder neural network for weather forecast, Future Gener. Comput. Syst 89 (2018) 446–454.

[6] M. Qiu, P. Zhao, K. Zhang, J. Huang, X. Shi, X. Wang, W. Chu, A short-term rainfall prediction model using multi-task convolutional neural networks, in: Data Mining (ICDM), IEEE International Conference on, IEEE, 2017, p. 395–404.

[7] R. C. Nascimento, Y. M. Souto, E. S. Ogasawara, F. Porto, E. Bezerra, Stconvs2s: Spatiotemporal convolutional sequence to sequence network for weather forecasting, CoRR abs/1912.00134 (2019). URL: http://arxiv.org/abs/1912.00134. arXiv:1912.00134.

[8] K. Yonekura, H. Hattori, T. Suzuki, Short-term local weather forecast using dense weather station by deep neural network, in: IEEE International Conference on Big Data, Big Data, IEEE, 2018, p. 1683–1690.

[9] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, I. Polosukhin, Attention is all you need, in: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems, volume 30, Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[10] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, N. Houlsby, An image is worth 16x16 words: Transformers for image recognition at scale, ArXiv abs/2010.11929 (2021).

[11] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, L. Shao, Pyramid vision transformer: A versatile backbone for dense prediction without convolutions, ArXiv abs/2102.12122 (2021).

[12] R. Ranftl, A. Bochkovskiy, V. Koltun, Vision transformers for dense prediction, 2021. arXiv:2103.13413.

[13] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, B. Guo, Swin Transformer: Hierarchical Vision Transformer using Shifted Windows (2021). URL: http://arxiv.org/abs/2103.14030. arXiv:2103.14030.

[14] H. Cao, Y. Wang, J. Chen, D. Jiang, X. Zhang, Q. Tian, M. Wang, Swin-Unet: Unet-like Pure Transformer for Medical Image Segmentation (2021) 1–14. URL: http://arxiv.org/abs/2105.05537. arXiv:2105.05537.

[15] Z. Liu, J. Ning, Y. Cao, Y. Wei, Z. Zhang, S. Lin, H. Hu, Video Swin Transformer (2021) 1–12. URL: http://arxiv.org/abs/2106.13230. arXiv:2106.13230.

[16] L. J. Ba, J. R. Kiros, G. E. Hinton, Layer normalization, CoRR abs/1607.06450 (2016). URL: http://arxiv.org/abs/1607.06450. arXiv:1607.06450.

[17] Multi-sensor weather forecast competition, https://www.iarai.ac.at/weather4cast/, 2021.

[18] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, CoRR abs/1412.6980 (2015).

[19] W. Falcon et. al., Pytorch lightning, GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning 3 (2019).

[20] A. Bojesomo, H. A. Marzouqi, P. Liatsis, Traffic flow prediction using deep sedenion networks (2020). arXiv:2012.03874.