Knowledge-Based Systems 252 (2022) 109445

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Deep coastal sea elements forecasting using UNet-based models

Jesús García Fernández [b], Ismail Alaoui Abdellaoui [b], Siamak Mehrkanoon [a,b,*]

[a] Information and Computing Sciences, Utrecht University, Utrecht, Netherlands
[b] Department of Data Science and Knowledge Engineering, Maastricht University, The Netherlands

ARTICLE INFO

Article history:
Received 15 December 2021
Received in revised form 14 June 2022
Accepted 8 July 2022
Available online 22 July 2022

Keywords:
Coastal sea elements
Time-series satellite data
Deep learning
Convolutional neural networks
UNet

ABSTRACT

Due to the recent development of deep learning techniques applied to satellite imagery, weather forecasting that uses remote sensing data has also been the subject of major progress. The present paper investigates multiple hours ahead coastal sea elements forecasting in the Netherlands using UNet based architectures. The hourly satellite image data from the Copernicus observation program spanned over a period of two years has been used to train the models and make the forecasting, including seasonal forecasting. Here, we propose 3D dimension Reducer UNet (3DDR-UNet), a variation of the UNet architecture, and further extend this novel model using residual connections, parallel convolutions and asymmetric convolutions which result in introducing three additional architectures, i.e. Res-3DDR-UNet, InceptionRes-3DDR-UNet and AsymmInceptionRes-3DDR-UNet respectively. In particular, we show that the architecture equipped with parallel and asymmetric convolutions as well as skip connections outperforms the other three discussed models.

1. Introduction

Renewable energy system has received increasing attention in the last years. In this context, the ability to accurately predict weather elements is crucial to the effective use of weather elements resources. In particular, it has been shown that weather forecasting affects sectors like agriculture, forestry, transportation and healthcare among others, thus having a major impact on the global economy [1–5].

Classical approaches to perform weather forecasting heavily relied on thermodynamics, Navier–Stokes equations, the statistical properties of the data, as well as the various properties of the atmosphere [6–9]. This set of methods belongs to the numerical weather prediction (NWP) approaches and generally require a large amount of computational resources since the processing is done on supercomputers [10]. Furthermore, it has been shown that NWP-based approaches might suffer from computational instability, mainly due to the initial conditions of the models [11].

Recent data-driven approaches on the other hand perform a simulation of an entire system in order to predict its next state. The main technique used by these methods is the usage of historical data to forecast the weather. Based on the success provided by machine learning models (i.e. support vector machines, random forests, Gaussian processes, and neural networks) to forecast time series, these approaches have also been used for weather data [12–18]. In particular, neural-networks-based models can use either shallow or deep architectures. As opposed to shallow networks that require domain knowledge and feature engineering, deep convolutional neural networks are less constrained by domain expertise. Indeed, these networks are capable of extracting the underlying complex patterns of the data by stacking of multiple nonlinear layers. Deep learning based models have already shown promising results in weather elements forecasting as well as several other domains such as biomedical signal analysis, healthcare, neuroscience, and dynamical systems [19–22]. Among successful deep models, the UNet architecture which consists of two main parts, i.e. the contracting and expanding, has been shown to be efficient in many computer vision related tasks. It is efficient at processing the input data at lower resolutions and restoring it to its original resolution. In particular, UNet-based architectures have been successfully applied to medical data for diverse tasks such as segmentation, cell counting, and reconstruction [23–25].

In this paper, we explore incorporating different elements to extend the core UNet architecture for weather forecasting task. The aim of the paper is to develop new deep architectures that can better learn the underlying complex mapping between a set of input satellite images and a set of out satellite images. In particular, the developed architectures are used to predict the values of different sea elements, i.e. eastward and northward water velocity, water salinity and water surface height, up to

* Corresponding author at: Department of Data Science and Knowledge Engineering, Maastricht University, The Netherlands.
E-mail addresses: j.garciafernandez@student.maastrichtuniversity.nl (J.G. Fernández), i.alaouiabdellaoui@student.maastrichtuniversity.nl (I.A. Abdellaoui), s.mehrkanoon@uu.nl, siamak.mehrkanoon@maastrichtuniversity.nl (S. Mehrkanoon).

https://doi.org/10.1016/j.knosys.2022.109445

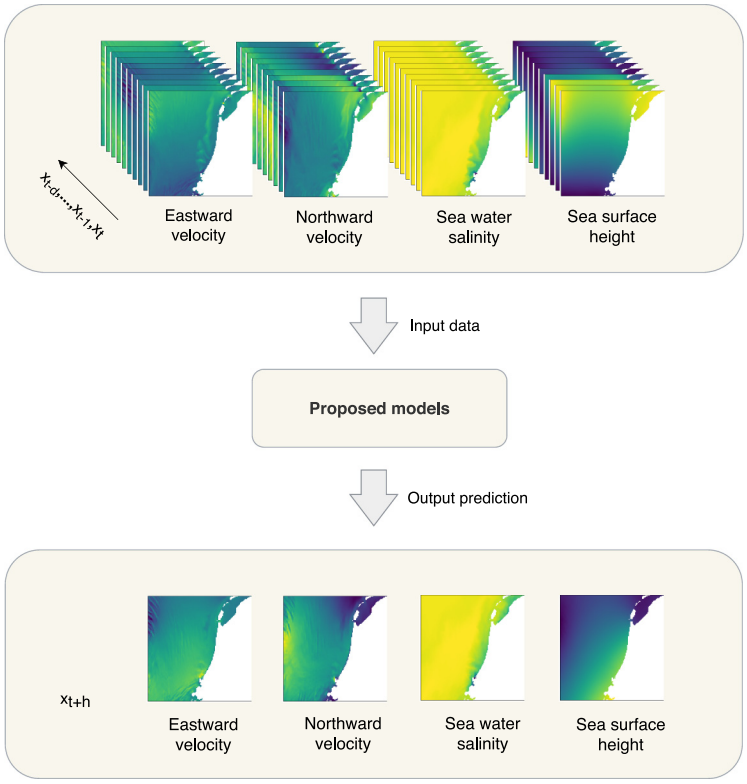

Fig. 1. Overview of the model input/output. Consecutive satellite images of different sea elements are concatenated and fed into our models. A single image per sea element is then generated in the output. The time steps from $t-d$ to t correspond to the lags (number of consecutive past images) of the models to predict the image at time step $t+h$, where d is the number of lags and h is the number of time steps ahead. Intercorrelated features are extracted across sea elements to generate the forecast.

72 h ahead, using historical satellite images of a coastal area of the Netherlands. More precisely, the models receive a set of past satellite images containing different sea elements at time steps $t-d,\ldots,t$. Then, the models output a single image per sea element at time step $t+h$ where in our case h ranges from 12 h to 72 h ahead. The input data is 4-dimensional with the shape (10, 128, 128, 4). The first dimension corresponds to the number of past images in time (here referred to as lag and denoted by d) used by the models to predict the feature. The second and third dimensions correspond to the height and width of each image. The fourth dimension corresponds to the number of sea elements. Similar to the input data, the models output is also 4-dimensional but with the shape (1, 128, 128, 4). Here, the first dimension is one indicating that the models predict only one image at a time. An overview of this procedure is shown in Fig. 1.

This paper proposes four different UNet core based models, each one being an extended version of the previous one. The proposed models simultaneously extract intercorrelated features from different sea elements to perform their forecasting. We show that the model that uses a combination of asymmetric, parallel convolutions and skip connections inside each residual block outperforms the other introduced models.

2. Related work

Weather forecasting based on deep-learning models has recently gain a lot of attention due to the rapid advance of neural network techniques and the availability of weather data [18,21,26–28]. The authors in [29], used a deep convolutional neural network to predict thunderstorms and heavy rains. The model was then compared against traditional machine learning models such as random forests and support vector machines. The authors in [30] incorporated multiple ConvLSTM layers to predict the precipitation rate using radar data. Moreover, the authors in [31] combine multistream convolutional neural networks with a self-attention mechanism [32,33] for precipitation forecasting.

In this paper, we are interested in weather forecasting based on satellite imagery. Successful approaches have been presented in the literature for performing frame prediction [34–36], and recently, it has also been applied to weather forecasting problems. For instance, the authors in [37], used satellite data in combination with deep-learning techniques to perform sea surface temperature (SST) prediction in a subarea of the East China sea. The main type of layer used in this work was the ConvLSTM layer and was compared to three different models: support vector regression (SVR) model, a persistence model that was simulating a naive forecast, and a third model based on LSTM layers. It was shown that the ConvLSTM model outperformed the other models for ten-days-ahead prediction in a recursive fashion.

Similarly to the previous work, sea surface temperature forecasts is also performed using deep learning and remote sensing imagery from satellite data in [38]. The methodology discussed in [38] is based on a multi-input convolutional neural networks that process the inputs using different spatial resolutions. The end goal of this work was to establish a relationship between sea surface temperature forecasting and tropical instability waves.

In [39], eight cyclone datasets were used for two main objectives: classifying whether the given image contains a storm or not, as well as predicting the storm's location. In contrast with the previous work, this methodology is not end-to-end since multiple preprocessing steps are performed before training the deep learning model. In particular, multiple optical-flow-based techniques are used to perform temporal interpolation and the result of this processing is then fed to the deep-learning models. The neural networks used are existing approaches that provide a fast inference time, namely YOLO [40] and RetinaNet [41].

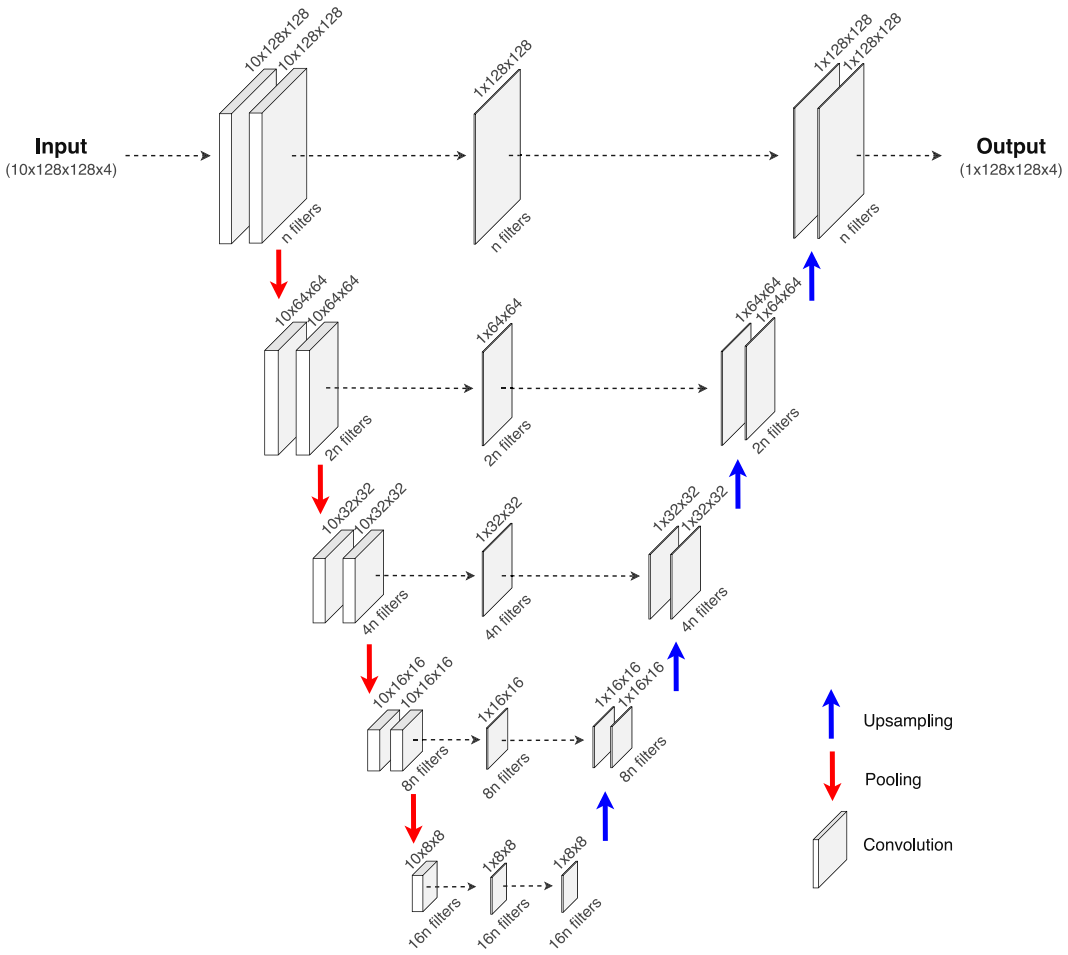

Fig. 2. Architecture of the 3DDR-UNet model. The annotations above the convolutions correspond to the output shape of those convolutions. Also, the number of filters is indicated below. We can appreciate that the first part reduces the dimensionality and then the second part upsamples the data to its original size (except for the temporal dimension). Between the reduction and expansion parts, intermediate convolutions reduce the temporal dimension (lags) from 10 to 1.

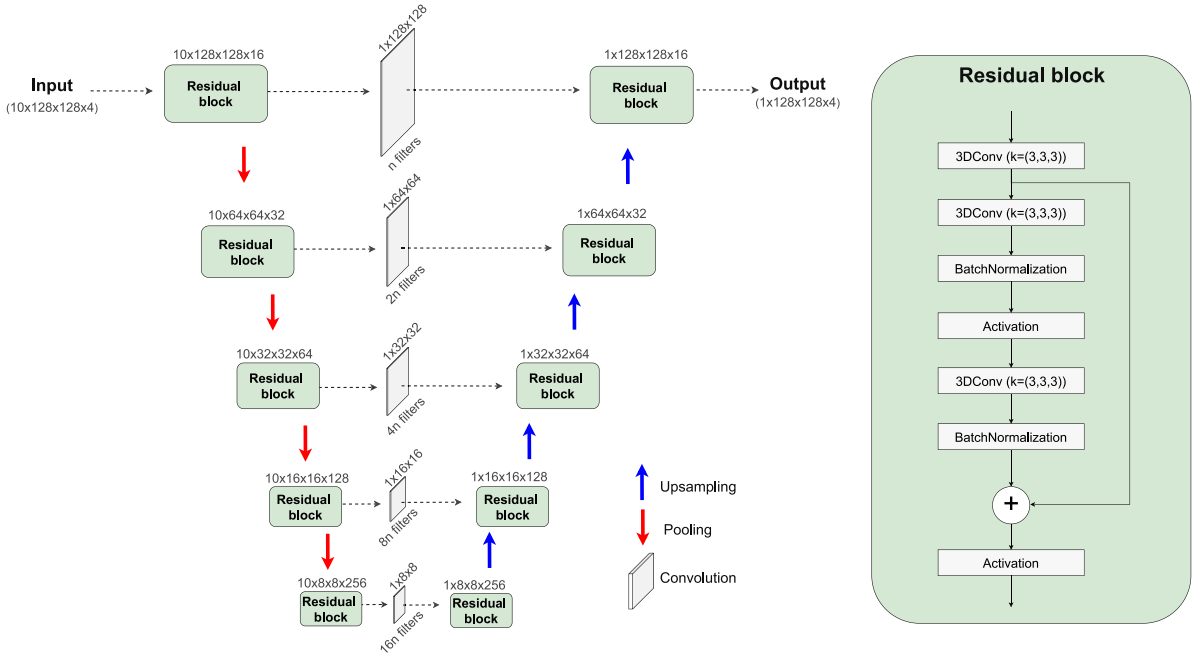

Fig. 3. Architecture of Res-3DDR-UNet model. The annotations above the residual blocks correspond with the output shape of such blocks. Similarly to 3DDR-UNet, the first part reduces the dimensionality, and the second part upsamples the data to its original size (except for the temporal dimension). Between the reduction and expansion parts, intermediate convolutions reduce the temporal dimension (lags) from 10 to 1.

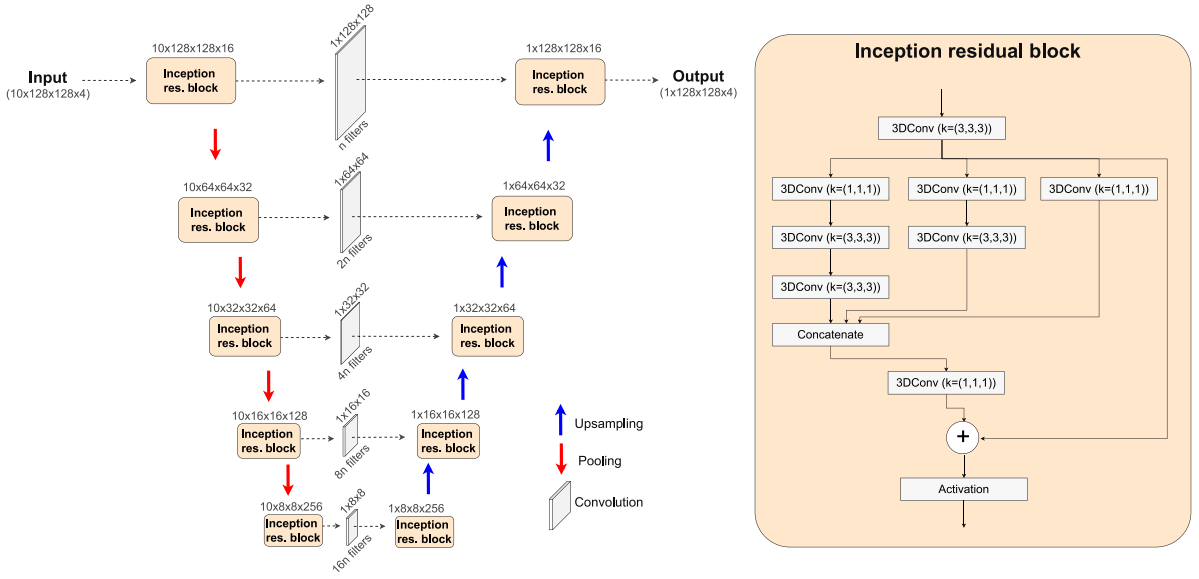

Fig. 4. Architecture of the InceptionRes-3DDR-UNet model. The annotations above the inception residual blocks correspond to the output shape of such blocks. Similarly to the previous models, the first part reduces the dimensionality, and the second part upsamples the data to its original size (except for the temporal dimension). Between the reduction and expansion parts, intermediate convolutions reduce the temporal dimension (lags) from 10 to 1.

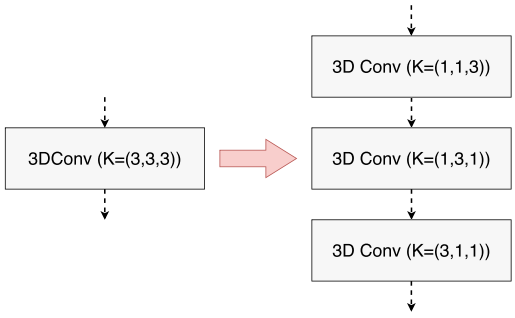

Fig. 5. Example of kernel decomposition in the asymmetric convolution operation.

The authors in [17,42] augment the UNet as well as TransUNet models with an attention mechanism to capture the most relevant features in the input data and have shown the efficiency of their approach for precipitation nowcasting tasks on radar images that cover the precipitation information in the Netherlands and its neighboring countries. Another similar research work in [43] performs precipitation nowcasting using artificial neural networks and satellite data. In this work, thermal infrared image prediction is first performed in order to get an estimate of the predicted precipitation. Hourly data is used and the neural network model is compared to other approaches such as linear interpolation, steady-state methodology and persistence prediction.

Future frame prediction is an active field of research in computer vision. In this context, within the field of deep learning, two main approaches are generally considered: auto-encoders and generative adversarial neural networks [44–47]. In particular, variations of the UNet model can efficiently perform future frame predictions applied to weather forecasting. The work in [48] extends the feature extraction ability of the UNet by adding modules that operate with the data at different scales. In this work, in order to extend the feature extraction capability of the core UNet model, we explore equipping it with different elements such as residual connection, parallel branches as well as asymmetric convolutions. As opposed to [49] where they use CNN as core model and augment it with residual connection for segmentation task with medical images as input, here in our Res-3DDR-UNet, UNet is used as core model and is augmented with similar residual connection as that of [49]. Moreover, Res-3DDR-UNet is used in coastal sea forecasting task.

3. Proposed models

We seek to propose a model that accurately maps a set of input images to a set of output images. To this end, the UNet architecture [23] is used as the core model and is enriched by incorporating more advanced elements suitable for the task under consideration. The UNet architecture is initially designed for medical image segmentation and has similar structure to that of an auto-encoder. A first contracting part, where the features are extracted from the input image, is followed by an expanding part that performs classification on each pixel.

In this paper, we propose an extended UNet architecture. Additionally modern enhancement techniques such as residual connections [50], inception modules [51,52] and asymmetric convolutions [53] are taken into account when designing these models. The residual or skip connections have been shown to improve the performance of deep networks by avoiding the vanishing of small gradients. On the other hand, inception modules apply convolutions with different kernels at the same level to capture features from larger and smaller areas in parallel. In the same way, asymmetric convolutions allow us to enlarge the network, and thus, its learning capacity, and at the same time, the number of parameters is reduced.

Here four different models, each one being an extended version of the previous one are proposed. These models are 3DDR-UNet, Res-3DDR-UNet, InceptionRes-3DDR-UNet and AsymmInceptionRes-3DDR-UNet.

3.1. 3D dimension reducer UNet (3DDR-UNet)

This section introduces the 3D Dimension Reducer UNet (3DDR-UNet) which is based on the UNet core architecture [23]. The classical UNet model is a fully convolutional neural networks with two parts: A contraction part or encoder and an expansion part or decoder. The first part is composed of stacked convolutions and

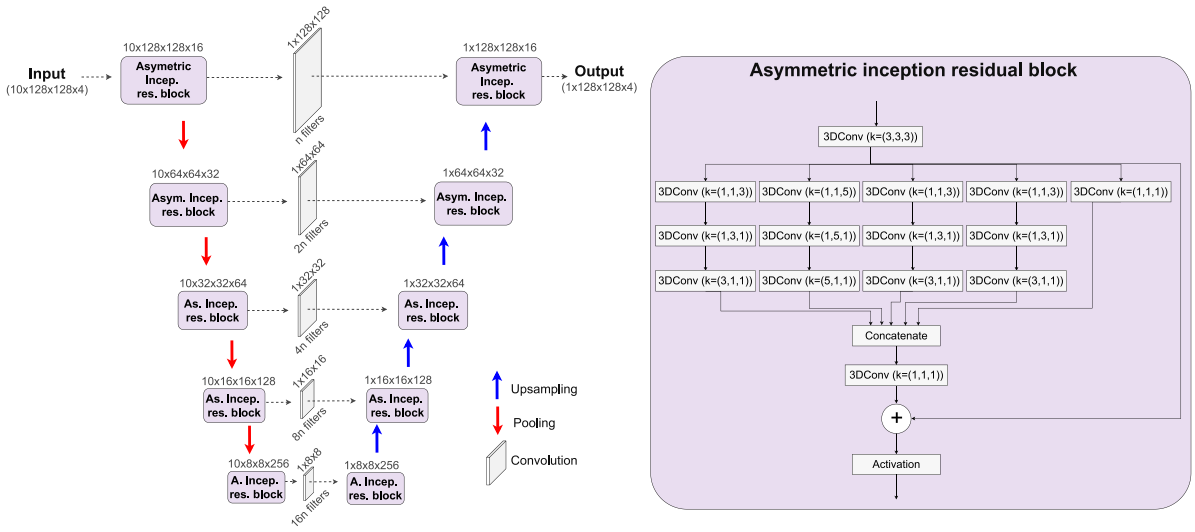

Fig. 6. Architecture of the AsymmInceptionRes-3DDR-UNet model. The annotations above the asymmetric inception residual blocks correspond to the output shape of such blocks. Similarly to the previous models, the first part reduces the dimensionality, and the second part upsamples the data to its original size (except for the temporal dimension). Between the reduction and expansion parts, intermediate convolutions reduce the temporal dimension (lags) from 10 to 1.

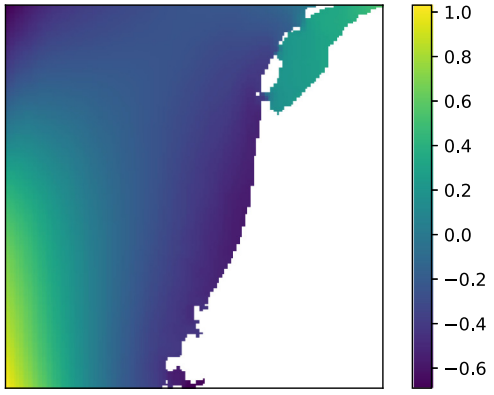

Fig. 7. Example of the sea surface height in meters of the studied region.

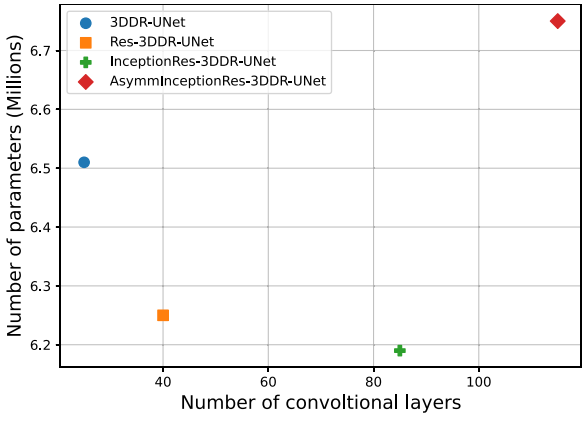

Fig. 8. Comparison of the number of parameters and convolutional layers between models.

pooling operations to extract features and capture the context in the input. The second symmetric part combines the features extracted in the contraction part with an upsampled output. In this way, the network expands the data to its original size and projects the learned features onto the pixel space to perform an accurate classification of them.

Here, we propose the 3DDR-UNet architecture, which manipulates 3-dimensional data in the encoder and 2-dimensional data in the decoder. This configuration allows the network to capture spatial and temporal dependencies from a stack of 2-dimensional images in the contracting part.

Then, the first input dimension (time dimension) is reduced from d (number of past time steps or lags, which is 10 in our case) to 1 in the horizontal connections of the network before it is concatenated in the decoder. Those extracted and combined features are later used to reconstruct one single image in the decoder. The reduction in the first input dimension is carried out by convolutions with kernel size $d \times 1 \times 1$ ($10 \times 1 \times 1$ in our case. Where 'd' corresponds to the number of lags) and valid padding. The first dimension of the kernel corresponding to the number of lags, and valid padding, results in a single value in the first dimension of the data after this convolution. The output of these operations is a weighted average of different time steps in the input. Essentially, this architecture extracts features in the encoder part and averages them in a weighted fashion over the first dimension before it is fed into the decoder part.

The number of convolutional filters grows exponentially after each pooling from n to $16n$ in the encoder part, and shrinks again to n in the decoder part. Here the kernel size is set to $3 \times 3 \times 3$ which means the kernel is applied to patches of 3×3 pixels over a stack of 3 images (in the temporal dimension). The size of both the pooling and the upsampling operations is set to $1 \times 2 \times 2$. This size means that the pooling is applied to patches of 2×2 pixels per image (in the temporal dimension). In this way, the temporal dimension of the data remains unchanged during the pooling and upsampling, while the spatial dimension of the data (i.e. second and third dimensions) is reduced in the encoder and later upsampled in the decoder.

Given the nature of the task, we train the network to perform a regression of every pixel. Traditionally UNet is used for segmentation tasks. Here, as opposed to the segmentation tasks, in which each pixel belongs to a class, we first normalize the input and output data. Then, the mean squared error (MSE) metric is

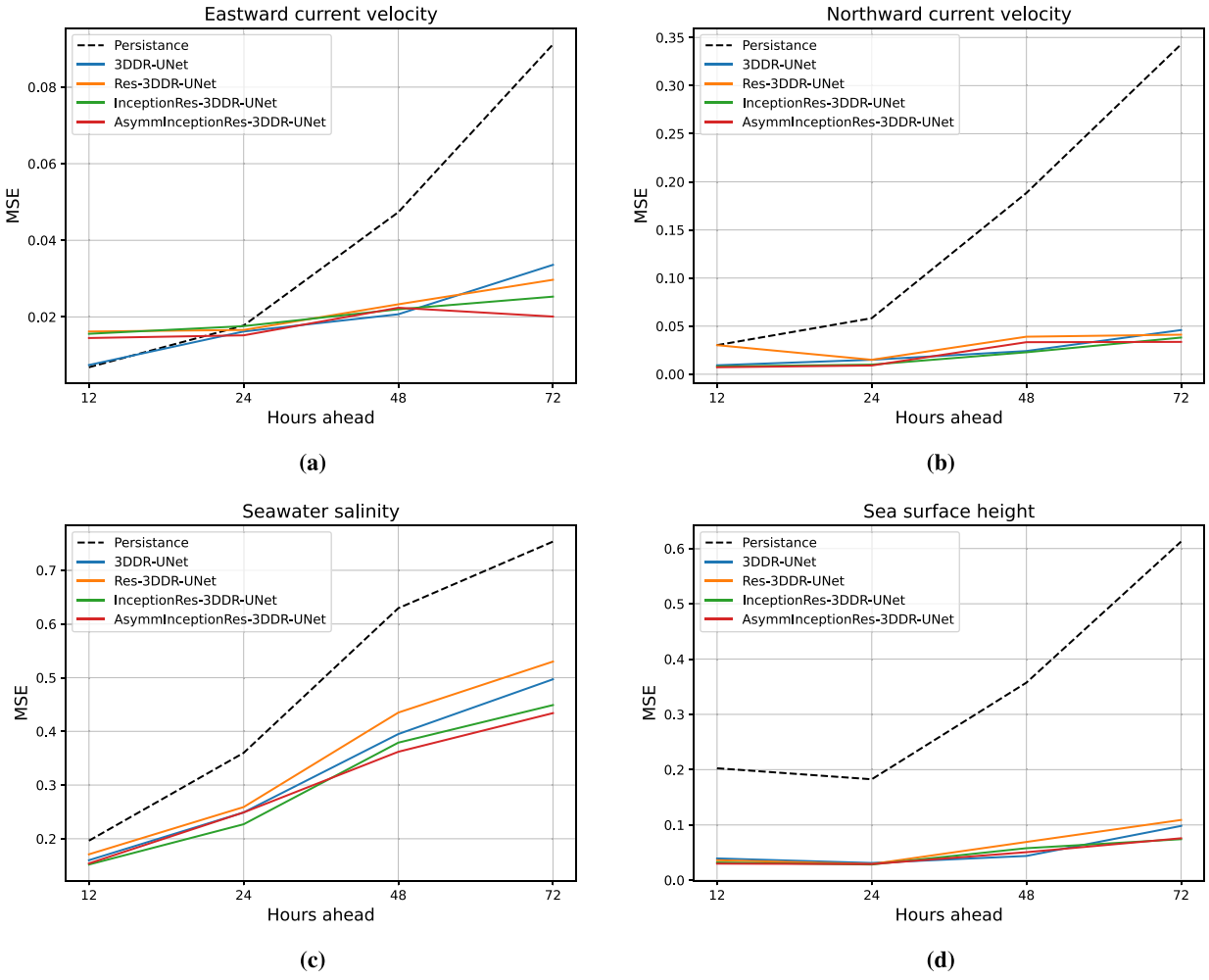

Fig. 9. MSE of each model for different variables at different hours ahead.

used during the training to minimize the difference between the predicted value of each pixel and the ground truth value. The architecture 3DDR-UNet of the network is shown in Fig. 2.

3.2. Residual 3D dimension reducer UNet (Res-3DDR-UNet)

The second proposed model, Residual 3D Dimension Reducer UNet (Res-3DDR-UNet), is an extension of the 3DDR-UNet model introduced previously. In order to augment its learning capacity, we scale up the model by adding a generous number of convolutional operations. We use three convolutional layers and a skip connection around the first layer and the final activation to avoid the vanishing of gradients. All these operations form a residual block.

Further, the outputs of the last convolutions in the block are normalized making use of a batch normalization layer. This normalization seek to increase the robustness of the network and alleviate the vanishing gradient problem.

Following the lines of [50], we skip two convolutional layers in each block which enhances the performance as well as faster training. As a result of these changes, the number of trainable parameters grows by 50% compared to the 3DDR-UNet. It should be noted that the kernel size of the convolutions, pooling, and upsampling, as well as the loss function, were similar to the ones used in 3DDR-UNet. The architecture of the Res-3DDR-UNet is shown in Fig. 3.

3.3. Inception residual 3D dimension reducer UNet (InceptionRes-3DDR-UNet)

Motivated by the effectiveness of the inception modules in CNN classifiers [51,54], we include similar modules in our residual blocks. Within this module, the data stream is split into parallel convolutions with different kernel sizes. Later, the branches are concatenated again. This structure is motivated by the ability to extract various features by using multiple kernel sizes applied to the same data. After these parallel operations, the features are concatenated and combined with a $1 \times 1 \times 1$ convolution, which is equivalent to a weighted average. Hence, the network learns to favor over time the branches with the most suitable kernels. Essentially, this module allows the network to use different kernels for the task, and give more importance to the most relevant ones.

Here, in particular, we use three parallel branches with $1 \times 1 \times 1$, $3 \times 3 \times 3$ and $5 \times 5 \times 5$ kernels. As suggested in [54], we approximate the $5 \times 5 \times 5$ convolution by two sequential $3 \times 3 \times 3$ convolutions, leading to a reduction in the computational cost. In addition, a convolution of $1 \times 1 \times 1$ is included at the beginning of each branch to reduce the dimensionality of the data, and thus, reduce the computational cost. Furthermore, inspired by the performance of [52], we kept the residual connection that skips the parallel branches. The number of convolutional filters, kernel sizes of pooling and upsampling as well as the loss function are the same as those of the previous models. The architecture of InceptionRes-3DDR-UNet is shown in Fig. 4.

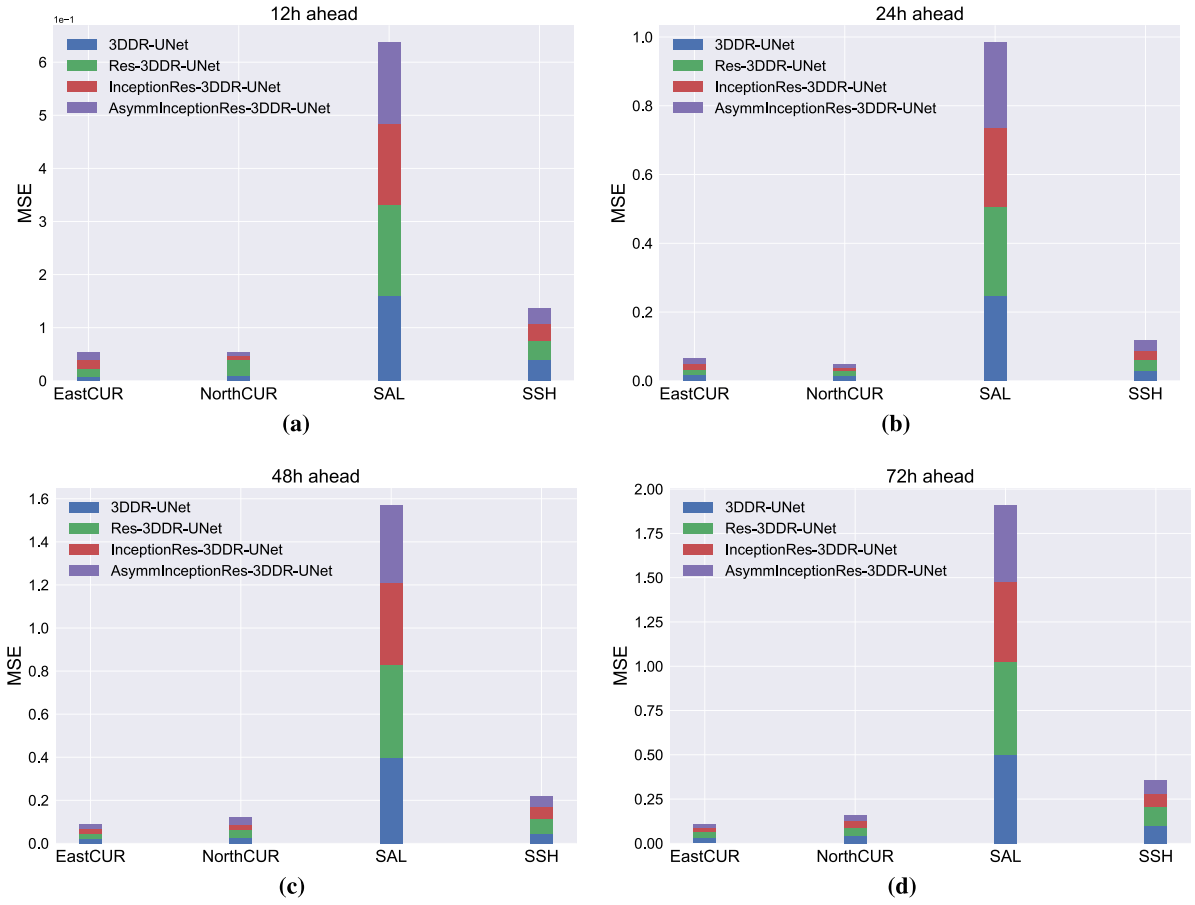

Fig. 10. MSE of individual sea elements for the different models. The forecasts are performed using (a) 12 h, (b) 24 h, (c) 48 h, and (d) 72 h ahead.

3.4. Asymmetric inception residual 3D dimension reducer UNet (AsymmInceptionRes-3DDR-UNet)

Driven by the need to reduce the parameters in InceptionRes-3DDR-UNet, we introduce a lighter, yet more effective model. Here we use asymmetric convolutions [53] to lower the complexity of the parallel convolutions. As shown in Fig. 5, each kernel is decomposed into three simpler ones and applied consecutively. The resulting combination of operations is an approximation of the original operation with considerably fewer parameters (see [53] for more details). In consequence of such a reduction of parameters, we can afford to remove the $1 \times 1 \times 1$ convolution at the beginning of each branch within the asymmetric inception residual block, whose purpose is to reduce the complexity of the data, and thus, making the model lighter. However, to make the model comparable to the previous ones, we increased the number of parameters. To this end, we include two more parallel branches in each block. The number of convolutional filters, the kernel size of pooling and upsampling as well as the loss function are the same as in the previous described models. The architecture of AsymmInceptionRes-3DDR-UNet is shown in Fig. 6.

4. Data description

The data used in this paper consists of satellite images. It is provided by Copernicus,[1] the observation program led by the European Commission and the European Space Agency (ESA). Specifically, it is part of the dataset "Atlantic - European North West Shelf - Ocean Physics Analysis and Forecast" [55], covering a geographical area from E 002° 000 to E 006° 000 and latitude from N 51° 600 to *N* 53° 400. The spatial resolution is approximately 1.5 km, so every pixel represents a region the size of 1.5×1.5 km. We chose such a geographical area since it covers both the land and sea of the Netherlands. Furthermore, the selected observations start on 1st March 2017 and end on 13th February 2019, with an hourly temporal resolution. The dataset consists of four weather variables, i.e. eastward current velocity (EastCUR), northward current velocity (NorthCUR), seawater salinity (SAL) and sea surface height (SSH).

Therefore, each time step of each variable is represented by a 135×135 image (see Fig. 7). For reproducibility purposes, the code as well as the dataset are available on Github.[2] More details on the included variables, can be found in the official documentation.[3]

5. Experimental results

5.1. Data preprocessing

As the data contains sea elements, the pixels in the image that represent the ground should not be taken into account by the network. In practice, the pixels corresponding to the land are masked initially. Hence, we apply a MinMax scaling with a

[1] https://marine.copernicus.eu/.

[2] https://github.com/jesusgf96/Sea-Elements-Prediction-UNet-Based-Models.

[3] https://resources.marine.copernicus.eu/documents/PUM/CMEMS-NWS-PUM-004-013.pdf.

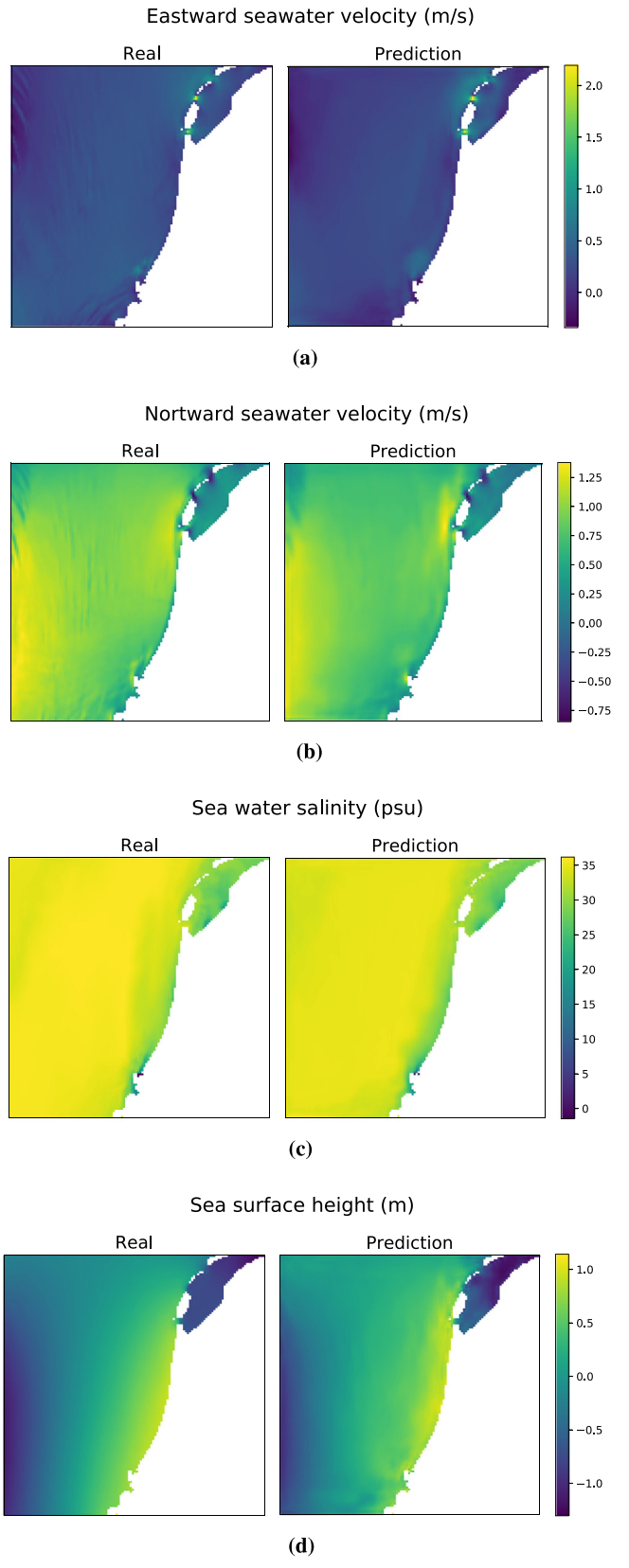

Fig. 11. AsymmInceptionRes-3DDR-UNet's forecast of all variables 48 h ahead in winter. The data is scaled back to its original values.

Table 1
Time steps used for validation and testing.

		Spring	Summer	Autumn	Winter
Validation	From	01/04/2018	01/07/2018	01/10/2018	01/01/2019
	To	22/04/2018	22/07/2018	22/10/2018	22/01/2019
Testing	From	23/04/2018	23/07/2018	23/10/2018	23/01/2019
	To	13/05/2018	12/08/2018	13/11/2018	13/02/2019

boundary of 0.1 and 1 to the pixels representing the sea elements as follows:

$$x = 0.1 + \frac{((x - x_{\min}) * (1 - 0.1))}{x_{\max} - x_{\min}}. \quad (1)$$

Then the pixels representing the ground are assigned a zero value. In this way, the pixels that belong to the ground are invisible to the models because of the ReLu activation function used within them. Moreover, we crop the images seven pixels from the right and the bottom sides, resulting in a 128 × 128 shape, which is suitable for the subsequent convolutional and pooling operations. The data is arranged in such a way that the resulting object is a four-dimensional array $\mathcal{T} \in \mathbb{R}^{L \times H \times W \times V}$, where L is the number of time-steps, which makes up the time dimension. H and W refer to the size of the image and form the spatial dimensions. The last element V corresponds to the sea variables.

5.2. Experimental setup

For all the models, we use all the variables as input, and we forecast the same variables in the future. The number of convolutional filters is chosen in such a way that all models contain a comparable total number of trainable parameters.

In our experiments, the number of lags is set to 10 as empirically it was found to yield better performance compared to other tested lag values. Therefore, the model receives ten hours of information to predict a single time step, which translate into having an input with shape (10, 128, 128, 4) and output with shape (1, 128, 128, 4). In addition, to test the forecasting ability of the models as well as their robustness, four different experiments are carried out. It consists of performing 12-, 24-, 48-, and 72-hours-ahead forecasts for all the variables. The models are trained with data ranging over a year, from 1st March 2017 to 1st February, which is a total 8760 h of training data. The validation data is composed of 2016 h, and represents 504 h from each season (spring, summer, autumn and winter). This validation data corresponds to a period of time after the training data. Similarly, the test data is composed of another 2016 h after the training data. The specific days used in both the validation and test can be found in Table 1.

5.3. Training

The same training setup is used in all the models. As mentioned previously in Section 3, the mean squared error (MSE) is used as the loss function to minimize the differences between the predicted and the ground-truth images. Adam optimization method [56], with TensorFlow default parameters, i.e. the learning rate = 0.001, β_1 = 0.9, β_2 = 0.999 and the epsilon = 1e-07, is used to optimize the loss function. The batch size and the dropout rate are set to 16 and 0.5 respectively. No regularization or data augmentation is used. We also implemented a checkpoint callback that monitors the validation loss and the models were trained for 100 epochs in each of the experiments. The best results are then saved based on the performance of the models on the validation data. Also, the models were trained to predict every element for a particular numbers of hours ahead.

Table 2
Test MSE of all models in all four seasons.

	Hours ahead			
Model	12 h	24 h	48 h	72 h
Persistence	1.09×10^1	1.55×10^{01}	3.06×10^1	4.50×10^1
3DDR-UNet	5.40×10^2	7.78×10^2	1.21×10^1	1.69×10^1
Res-3DDR-UNet	6.34×10^2	7.99×10^2	1.42×10^1	1.77×10^1
InceptionRes-3DDR-UNet	5.19×10^2	7.08×10^2	1.20×10^1	1.47×10^1
AsymmInceptionRes-3DDR-UNet	5.15×10^2	7.56×10^2	1.17×10^1	1.41×10^1

Table 3
Test MSE of all models in different seasons.

Season	Model	12-h-ahead	24-h-ahead	48-h-ahead	72-h-ahead
Spring	Persistence	8.51×10^2	1.49×10^1	2.84×10^1	4.29×10^1
	3DDR-UNet	7.45×10^2	9.95×10^2	1.37×10^1	2.14×10^1
	Res-3DDR-UNet	8.37×10^2	1.01×10^1	1.65×10^1	1.69×10^1
	InceptionRes-3DDR-UNet	6.63×10^2	9.37×10^2	1.43×10^1	1.78×10^1
	AsymmInceptionRes-3DDR-UNet	6.98×10^2	9.56×10^2	1.38×10^1	1.66×10^1
Summer	Persistence	5.27×10^2	8.78×10^2	2.16×10^1	3.81×10^1
	3DDR-UNet	4.10×10^2	7.92×10^2	1.37×10^1	1.69×10^1
	Res-3DDR-UNet	5.23×10^2	6.85×10^2	1.23×10^1	1.97×10^1
	InceptionRes-3DDR-UNet	4.37×10^2	6.55×10^2	1.06×10^1	1.49×10^1
	AsymmInceptionRes-3DDR-UNet	4.03×10^2	7.88×10^2	1.04×10^1	1.43×10^1
Autumn	Persistence	4.23×10^2	8.07×10^1	1.91×10^1	2.79×10^1
	3DDR-UNet	3.36×10^2	4.81×10^2	6.95×10^2	7.89×10^2
	Res-3DDR-UNet	3.97×10^2	5.61×10^2	8.92×10^2	1.27×10^1
	InceptionRes-3DDR-UNet	3.29×10^2	4.31×10^2	6.98×10^2	6.75×10^2
	AsymmInceptionRes-3DDR-UNet	2.97×10^2	4.66×10^2	6.79×10^2	7.11×10^2
Winter	Persistence	1.09×10^1	1.55×10^1	3.06×10^1	4.50×10^1
	3DDR-UNet	6.62×10^2	8.51×10^2	1.62×10^1	1.90×10^1
	Res-3DDR-UNet	7.66×10^2	9.44×10^2	1.88×10^1	1.99×10^1
	InceptionRes-3DDR-UNet	6.41×10^2	8.18×10^2	1.63×10^1	1.71×10^1
	AsymmInceptionRes-3DDR-UNet	6.51×10^2	8.28×10^2	1.58×10^1	1.62×10^1

5.4. Results and discussion

This section presents the results obtained from the experiments described. In addition, we include a persistence model as a baseline comparison. Given time series input data, the persistence model outputs the last value of the input. That means this model assumes that the value of the predicted variables will not change with respect to the last input. The MSEs obtained for all the models for each of the configurations are tabulated in Table 2. The MSE error is computed per feature per pixel over the denormalized values given in the forecasts. In Tables 2 and 3, the MSE is averaged across features. The results in Table 2 correspond to testing the models on the four combined seasons. It can be observed that, for all the models, the forecast error increases as the number of hours ahead increases. In addition, one can notice that, the performance difference of our models with respect to the persistence model increases as the number of hours ahead increases.

AsymmInceptionRes-3DDR-UNet model performs better than the other models in almost all the scenarios. This improvement in the performance is more apparent as the number of hours ahead increases.

In Table 3, the test MSE of each model is displayed separated for each season. Similar to the previous results, AsymmInception-Res-3DDR-UNet outperforms the other discussed models in most the setups. One may also observe that seasons with more changing weather conditions such as winter makes it more challenging for the networks to learn the underlying complex patterns. All of the four models yield noticeably better forecasts in seasons with more stable weather, like summer. Furthermore, a comparison of the final number of convolutional layers of each model can be found in Fig. 8. Fig. 9 shows the MSE of the test data obtained for each sea element. Fig. 10 (a,b,c,d) corresponds to 12-, 24-, 48- and 72-hours-ahead forecasts respectively. In general, we observe that, of the considered variables in this study, seawater salinity is the most challenging variables to predict.

An example of the 48-hours-ahead forecast with AsymmInceptionRes-3DDR-UNet model during winter is shown in Fig. 11. As it can be seen, the forecast is accurate, even when it comes to seawater velocity (EastCUR and NorthCUR), which contains quite different areas. The results obtained suggest that the inclusion of parallel branches of convolutions, presented in InceptionRes-3DDR-UNet and Asymm InceptionRes-3DDR-UNet models has led to a more noticeable performance improvement. Likely, the different simultaneous transformations applied to the data through parallel branches provide these models with an extended feature extraction capability. In particular, the building blocks in Asymm InceptionRes-3DDR-UNet contain more parallel branches than InceptionRes-3DDR-UNet, which might lead to more efficiency. Its decomposed kernels allow having many more convolutional operations without increasing the number of total parameters by a large number.

6. Conclusion

In this paper, four new models based on the UNet architecture are introduced for multi-step-ahead coastal sea element forecasting. The proposed models are examined under different setups, i.e. different seasons and numbers of hours ahead. Among the discussed models, AsymmInceptionRes-3DDR-UNet and InceptionRes-3DDR-UNet have shown better performance due to the use of parallel convolutions. However, the incorporation of asymmetric convolutions and additional parallel branches make the AsymmInceptionRes-3DDR-UNet perform slightly better than the latter, yielding the most promising results on the studied tasks.

CRediT authorship contribution statement

Jesús García Fernández: Methodology, Software, Validation, Investigation. **Ismail Alaoui Abdellaoui:** Methodology, Software, Validation, Investigation. **Siamak Mehrkanoon:** Conceptualization, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

Simulations were performed with computing resources granted by RWTH Aachen University.

References

[1] R.W. Katz, A.H. Murphy, Economic Value of Weather and Climate Forecasts, Cambridge University Press, 2005.
[2] R.G. da Silva, M.H.D.M. Ribeiro, S.R. Moreno, V.C. Mariani, L. dos Santos Coelho, A novel decomposition-ensemble learning framework for multi-step ahead wind energy forecasting, Energy 216 (2021) 119174.
[3] S.R. Moreno, V.C. Mariani, L. dos Santos Coelho, Hybrid multi-stage decomposition with parametric model applied to wind speed forecasting in Brazilian Northeast, Renew. Energy 164 (2021) 1508–1526.
[4] Z. Liu, R. Hara, H. Kita, Hybrid forecasting system based on data area division and deep learning neural network for short-term wind speed forecasting, Energy Convers. Manage. 238 (2021) 114136.
[5] S.R. Moreno, R.G. da Silva, V.C. Mariani, L. dos Santos Coelho, Multi-step wind speed forecasting based on hybrid multi-stage decomposition model and long short-term memory neural network, Energy Convers. Manage. 213 (2020) 112869.
[6] A.C. Lorenc, Analysis methods for numerical weather prediction, Q. J. R. Meteorol. Soc. 112 (474) (1986) 1177–1194.
[7] P. Bauer, A. Thorpe, G. Brunet, The quiet revolution of numerical weather prediction, Nature 525 (7567) (2015) 47–55.
[8] H.R. Glahn, Statistical weather forecasting, in: Probability, Statistics, and Decision Making in the Atmospheric Sciences, Westview Press, 1985, pp. 289–335.
[9] A. Holtslag, E. De Bruijn, H. Pan, A high resolution air mass transformation model for short-range weather forecasting, Mon. Weather Rev. 118 (8) (1990) 1561–1575.
[10] K. Saito, H. Seko, T. Kuroda, T. Fujita, T. Kawabata, K. Aonashi, T. Tsuyuki, Next generation supercomputer project toward cloud resolving NWP, CAS/JSC WGNE Res. Act. Atmos. Ocea. Model 41 (2011) 5–19.
[11] J. Zhongzhen, Z. Qingcun, Problems on nonlinear computational instability in NWP, J. Meteorol. Soc. Jpn. Ser. II 64 (1986) 255–261.
[12] K.-j. Kim, Financial time series forecasting using support vector machines, Neurocomputing 55 (1–2) (2003) 307–319.
[13] G. Dudek, Short-term load forecasting using random forests, in: Intelligent Systems' 2014, Springer, 2015, pp. 821–828.
[14] A. Girard, C.E. Rasmussen, J.Q. Candela, R. Murray-Smith, Gaussian process priors with uncertain inputs application to multiple-step ahead time series forecasting, in: Advances in Neural Information Processing Systems, 2003, pp. 545–552.
[15] Y. Radhika, M. Shashi, Atmospheric temperature prediction using support vector machines, Int. J. Comput. Theory Eng. 1 (1) (2009) 55.
[16] K. Rasouli, W.W. Hsieh, A.J. Cannon, Daily streamflow forecasting by machine learning methods with weather and climate inputs, J. Hydrol. 414 (2012) 284–293.
[17] K. Trebing, T. Stańczyk, S. Mehrkanoon, SmaAt-UNet: Precipitation nowcasting using a small attention-UNet architecture, Pattern Recognit. Lett. 145 (2021) 178–186.
[18] K. Trebing, S. Mehrkanoon, Wind speed prediction using multidimensional convolutional neural networks, in: IEEE Symposium Series on Computational Intelligence, IEEE-SSCI, 2020, pp. 713–720.
[19] S. Webb, Deep learning for biology, Nature 554 (7693) (2018).
[20] S. Mehrkanoon, Deep neural-kernel blocks, Neural Netw. 116 (2019) 46–55.
[21] T. Stanczyk, S. Mehrkanoon, Deep graph convolutional networks for wind speed prediction, in: European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, ESANN, 2021, pp. 147–152.
[22] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.
[23] O. Ronneberger, P. Fischer, T. Brox, U-Net: Convolutional networks for biomedical image segmentation, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2015, pp. 234–241.
[24] T. Falk, D. Mai, R. Bensch, O. Çiçek, A. Abdulkadir, Y. Marrakchi, A. Böhm, J. Deubner, Z. Jäckel, K. Seiwald, et al., U-Net: deep learning for cell counting, detection, and morphometry, Nature Methods 16 (1) (2019) 67–70.
[25] Y. Han, J.C. Ye, Framing U-Net via deep convolutional framelets: Application to sparse-view CT, IEEE Trans. Med. Imaging 37 (6) (2018) 1418–1429.
[26] M. Schultz, C. Betancourt, B. Gong, F. Kleinert, M. Langguth, L. Leufen, A. Mozaffari, S. Stadtler, Can deep learning beat numerical weather prediction? Phil. Trans. R. Soc. A 379 (2194) (2021) 20200097.
[27] M. Chantry, H. Christensen, P. Dueben, T. Palmer, Opportunities and challenges for machine learning in weather and climate modelling: hard, medium and soft AI, Phil. Trans. R. Soc. A 379 (2194) (2021) 20200083.
[28] S. Mehrkanoon, Deep shared representation learning for weather elements forecasting, Knowl.-Based Syst. 179 (2019) 120–128.
[29] K. Zhou, Y. Zheng, B. Li, W. Dong, X. Zhang, Forecasting different types of convective weather: A deep learning approach, J. Meteorol. Res. 33 (5) (2019) 797–809.
[30] S. Kim, S. Hong, M. Joh, S.-k. Song, Deeprain: Convlstm network for precipitation prediction using multichannel radar data, 2017, arXiv preprint arXiv:1711.02316.
[31] 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 preprint arXiv:2003.12140.
[32] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in: Advances in Neural Information Processing Systems, 2017, pp. 5998–6008.
[33] J. Ho, N. Kalchbrenner, D. Weissenborn, T. Salimans, Axial attention in multidimensional transformers, 2019, arXiv preprint arXiv:1912.12180.
[34] Z. Wang, Z. Yang, Y. Zhang, N. Su, G. Wang, Ts-Unet: A temporal smoothed unet for video anomaly detection, in: International Conference on Image and Graphics, Springer, 2021, pp. 789–798.
[35] Z. Chang, X. Zhang, S. Wang, S. Ma, Y. Ye, X. Xinguang, W. Gao, MAU: A motion-aware unit for video prediction and beyond, Adv. Neural Inf. Process. Syst. 34 (2021).
[36] W. Riaz, G. Chenqiang, A. Azeem, J.A. Bux, A. Ullah, et al., Traffic anomaly prediction system using predictive network, Remote Sens. 14 (3) (2022) 447.
[37] C. Xiao, N. Chen, C. Hu, K. Wang, Z. Xu, Y. Cai, L. Xu, Z. Chen, J. Gong, A spatiotemporal deep learning model for sea surface temperature field prediction using time-series satellite data, Environ. Model. Softw. 120 (2019) 104502.
[38] G. Zheng, X. Li, R.-H. Zhang, B. Liu, Purely satellite data–driven deep learning forecast of complicated tropical instability waves, Sci. Adv. 6 (29) (2020) eaba1482.
[39] S. Shakya, S. Kumar, M. Goswami, Deep learning algorithm for satellite imaging based cyclone detection, IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 13 (2020) 827–839.
[40] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, real-time object detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 779–788.
[41] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 2980–2988.
[42] Y. Yang, S. Mehrkanoon, AA-transUNet: Attention augmented transUNet for nowcasting tasks, 2022, arXiv preprint arXiv:2202.04996.
[43] G. Rivolta, F. Marzano, E. Coppola, M. Verdecchia, Artificial neural-network technique for precipitation nowcasting from satellite imagery, 2006.
[44] W. Liu, W. Luo, D. Lian, S. Gao, Future frame prediction for anomaly detection–a new baseline, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 6536–6545.
[45] V. Patraucean, A. Handa, R. Cipolla, Spatio-temporal video autoencoder with differentiable memory, 2015, arXiv preprint arXiv:1511.06309.
[46] J.-T. Hsieh, B. Liu, D.-A. Huang, L.F. Fei-Fei, J.C. Niebles, Learning to decompose and disentangle representations for video prediction, in: Advances in Neural Information Processing Systems, 2018, pp. 517–526.
[47] Y. Zhao, B. Deng, C. Shen, Y. Liu, H. Lu, X.-S. Hua, Spatio-temporal autoencoder for video anomaly detection, in: Proceedings of the 25th ACM International Conference on Multimedia, 2017, pp. 1933–1941.
[48] J.G. Fernández, S. Mehrkanoon, Broad-UNet: Multi-scale feature learning for nowcasting tasks, Neural Netw. 144 (2021) 419–427.
[49] L. Yu, X. Yang, H. Chen, J. Qin, P.A. Heng, Volumetric ConvNets with mixed residual connections for automated prostate segmentation from 3D MR images, in: Thirty-First AAAI Conference on Artificial Intelligence, 2017.

[50] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[51] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1–9.
[52] C. Szegedy, S. Ioffe, V. Vanhoucke, A. Alemi, Inception-v4, inception-resnet and the impact of residual connections on learning, 2016, arXiv preprint arXiv:1602.07261.
[53] H. Yang, C. Yuan, B. Li, Y. Du, J. Xing, W. Hu, S.J. Maybank, Asymmetric 3d convolutional neural networks for action recognition, Pattern Recognit. 85 (2019) 1–12.
[54] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2818–2826.
[55] M. Tonani, P. Sykes, R.R. Kin, N. McConnell, A.C. Péquignet, E. O'Dea, J.A. Graham, J. Polton, J. Siddorn, Atlantic - European north west shelf - ocean physics analysis and forecast dataset, Ocean Sci. 15 (2019) 1133–1158, [http://dx.doi.org/10.5194/os-15-1133-2019 The impact of a new high-resolution ocean model on the Met Office North-West European Shelf forecasting system].
[56] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2014, arXiv preprint arXiv:1412.6980.