UCLouvain

epl

École polytechnique de Louvain

Predicting the evolution of Antarctic sea ice with data science

Author: **Justin Lalieu**
Supervisors: **Jean-Charles Delvenne, François Massonnet**
Reader: **Eric Deleersnijder**
Academic year 2023–2024
Master [120] in Data Science: Information technology

Acknowledgments

First, I would like to express my sincere gratitude to my supervisors Jean-Charles Delvenne and François Massonnet for their time and valuable advices throughout the preparation of this thesis, which enabled me to work in the best possible conditions.

I would also like to thank all the people who contributed in any possible way to the successful completion of this work. In particular John Aoga for his technical advice and Eric Deleersnijder for being part of the jury and reader of this work.

Finally, I want to acknowledge my family and my friends for their precious support and encouragements throughout these years.

Computational resources have been provided by the supercomputing facilities of the Université catholique de Louvain (CISM/UCL) and the Consortium des Équipements de Calcul Intensif en Fédération Wallonie Bruxelles (CÉCI) funded by the Fond de la Recherche Scientifique de Belgique (F.R.S.-FNRS) under convention 2.5020.11 and by the Walloon Region.

Abstract

Data science makes it possible to apply a wide range of methods to problems in different fields. These include the prediction of sea ice concentration at the poles, which has a number of implications, including navigation, economic and diplomatic interests, and a better understanding of our planet. At the same time, during the year 2023, Antarctica experienced its lowest peak of sea ice extents ever recorded. This thesis evaluates the ability of neural networks, and more specifically the U-Net architecture, to predict Antarctic sea ice concentration by using and replicating data from a dynamical model. The models are compared with two baselines that represent the minimum performance that a more complex model must achieve in order to be relevant, a climatological forecast and a linear model. In addition, two different-sized versions of the U-Net architecture are used to compare the impact of model size on the results. Each model is evaluated from different angles to understand its strengths and weaknesses. The results show that the two U-Net architectures achieve better results than the baselines in terms of both mean error and error variance. However, no improvement is detected when using a larger model, with both versions delivering similar performances.

Contents

Chapter 1

Introduction and Context

The recent advances in data science allow applying a broad range of machine learning techniques to any problem as long as enough data are available, and earth sciences are no exception. The idea of this thesis is to apply some of these techniques to the forecasting of sea ice concentration (SIC) in polar regions. The SIC is a measure of the amount of sea ice for a given area expressed as a percentage. Studying and forecasting the SIC not only gives us a better understanding of how ice sheets react to climate change, but also greatly facilitates navigation in these waters and therefore the access to these areas, particularly for scientific purposes, but also recently for tourism.

This can either be done with a focus on the Arctic region or the Antarctic one. For diplomatic, economic and proximity reasons, most studies on the subject focus on the Arctic. On the other hand, Antarctica, which is more isolated and which seemed to resist the influence of climate change, has been the subject of much less research. Indeed, until 2007-2008, the ice pack at the South Pole did not seem to be affected by global warming and from then on even showed a tendency to extend its surface, but this was not to last, since from around 2015 the extent of sea ice began to diminish [16]. Furthermore, the year 2023 shows huge anomalies in the sea ice extent (SIE) in the Antarctic [8] as shown in Figure 1.1. It is important to keep in mind that the seasons in the northern and southern hemispheres are reversed, so in Antarctic the minimum SIE occurs in February, which corresponds to the end of the southern summer, and conversely, maximum SIE is reached in September, which corresponds to the end of the southern winter and the beginning of the southern spring.

This thesis focuses on the Antarctic, which allows to apply some existing SIC forecasting methods that have already proved their worth on the other pole. These methods can be classified in two categories: The dynamical methods and the statis-

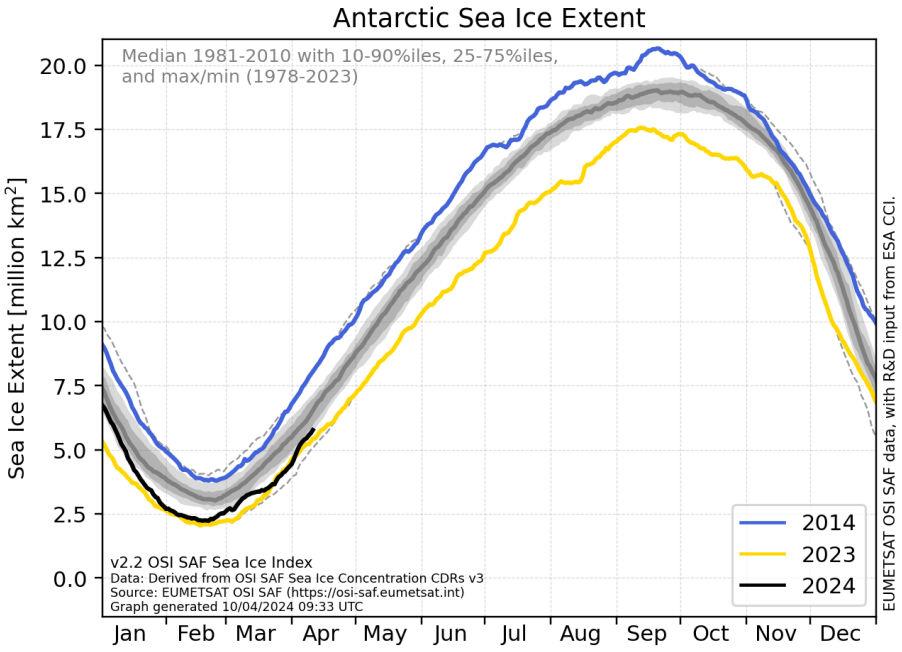

Figure 1.1: SIE observations made by the OSI SAF [8]

tical methods. On the one hand, dynamical methods use models based on physical laws while the statistical methods rely on the data collected whose are then feed to statistical models. However, those two approaches are still closely related as dynamical models can be fine-tuned using statistical ones and statistical models can use outputs from the dynamical ones as training input. Here, the focus is on statistical methods, but it is the outputs of the dynamical model NEMO-SI^3 [23] that is used as input for our statistical models.

Only talking about statistical methods is vague as this term covers a wide range of approaches, going from simple linear regressions and extrapolations to more complex Markov models [25] or Gaussian processes [11]. Hereby study focus on supervised machine learning and more specifically neural networks. Their versatility and extensive number of already existing architectures makes them applicable to a lot of fields, and amongst them, the computer vision and time series forecasting. Those combined frame perfectly the problem at hand: based on a satellite image of the Antarctica where each pixel, instead of containing information about colour, contains the SIC of the corresponding area, what will be the concentration like in one month ?

The neural network architecture this study focuses on is the U-Net architecture, an encoder-decoder network using convolutional layers primarily used in 2015 in biomedical image segmentation by Ronneberger et al. [21] and then successfuly applied to SIC forecasting in the Arctic by Andersson et al. [1]. Two versions of different sizes of that architecture are compared between each other to evaluate the impact of the model size on the results. To assess the relative performances of the neural networks, the two baseline methods used are the climatological forecast and a linear regression model, the goal being to compare, from a data science and earth science point of view, the performances of these methods [12].

Hereby study is conducted based on data in the form of daily grids containing the SIC as a percentage. Those can come either from satellite observations where passive microwave brightness temperatures are converted to SIC, or either as mentioned above, from dynamical models. Concerning the Antarctic, as seen in Figure 1.2 the data are represented using a lambert azimuthal equal-area projection centred on the South Pole which, as its name suggests, keeps the areas equal, with each pixel corresponding to an equivalent surface. The resolution is of 25km, meaning that each of the pixels represents a 25 square kilometres area. As a whole, each data sample's size is 432x432 pixels.

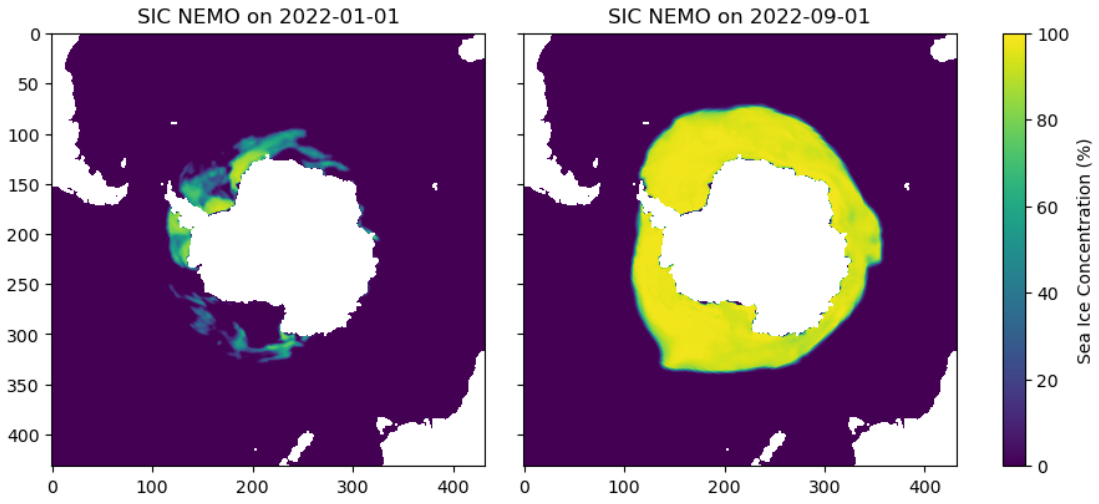

Figure 1.2: SIC data samples generated by NEMO-SI[3]

Then, as indicated by Figure 1.3, Antarctica can be divided in five main regions based on their geographical properties and inherent geophysical mechanisms. Therefore, SIC varies differently according to the regions. For instance, the Weddel

sea region is often the easiest to forecast, while the Indian, West Pacific and Ross regions are considered less predictable, the Amundsen-Bellingshausen (A and B) region being intermediate. However, this can vary according to seasons and not all studies show the same exact trends [16]. The other aspect in which SIC varies is according to the season. It is therefore relevant to compare the performances of the models according to the time of year. In the Southern Hemisphere, summer begins on December 1 and ends on February 28, autumn runs from March 1 to May 31, then comes winter from June 1 to August 31 and finally spring from September 1 to November 30. These regional and seasonal divisions are later used to assess the performance of the models and gain a better understanding of the regional and temporal performances disparities in the forecasts.

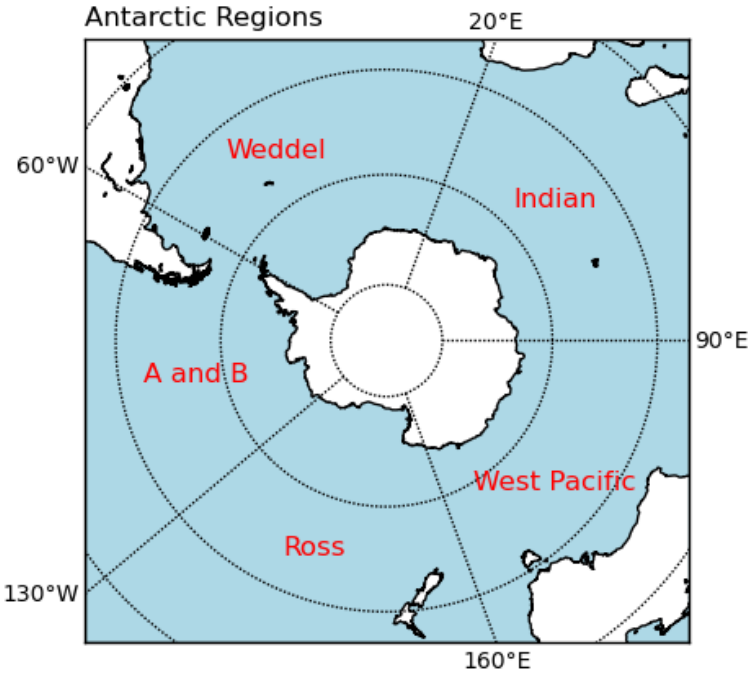

Figure 1.3: Oceans and seas around Antarctica

Therefore, the objectives of this study are multiple, each of them is described below:

- Compare the ability of the U-Net architecture to forecast the SIC at one month lead time compared with a climatological forecast and a linear model.
- Analyse each model's performances from a numerical, spatial, regional and seasonal perspective.

- Assess the usefulness of U-Net model's size in relation to performances.

This study aims to evaluate the U-Net deep learning architecture's effectiveness in predicting SIC one month ahead. We compare the U-Net's performances with those from the climatological forecast, which uses historical averages, and a linear model. This comparison highlights U-Net's potential advantages in capturing complex patterns and improving prediction accuracy over conventional methods due to its architecture and use of the ReLU non-linear activation function.

The second objective is to evaluate each model's ability to predict SIC by examining its performance from numerical, spatial, regional, and seasonal angles. Numerical analysis measures prediction accuracy using the MAE and NSE metrics, while spatial analysis assess the model's ability to forecast SIE by computing the IIEE and NIIEE of the predictions. Each of these metrics are described in the next chapter. Regional analysis focuses on the performances according to the regions described above, and seasonal analysis explores how accuracy varies throughout the year, depending on the seasons. This comprehensive evaluation provides insights into the model's strengths and weaknesses in different contexts.

The last objective is to compare the performances of two U-Net models, one small and one large, in predicting SIC. By evaluating these models, we aim to determine how model size impacts prediction accuracy and computational efficiency. This comparison helps to identify whether the larger model's increased complexity provides significant performances benefits over the smaller model, offering insights into the trade-offs between model size and forecasting performances.

The document is structured in such a way as to lead the reader through the research process in an ordered manner, beginning with this introduction setting out the general context and objectives of the study.

The second chapter contains a review of the literature on the subject, covering the many elements on which this study is based. It discusses the basic concepts, the different ways of framing the problem, the metrics used and the existing neural network architectures.

The third chapter describes the entire research process, starting with the dataset used to train and evaluate the models and then discussing the methods applied to the data and models to produce the forecasts.

The fourth chapter defines the baselines used, i.e. the climatological forecast and a linear model, and describes how they are implemented. This gives an

overview of the performances of basic approaches, and represents the performance to be beaten for any more complex model to be relevant.

The fifth chapter considers the implementation and structure of the two U-Net models. After a comparison between the number of parameters and the size of the small version and the large one, we discuss the tuning of the hyperparameters for both models as well as the most relevant set of input variables to use based on our dataset.

The sixth chapter compares the results of the models under the different aspects described above and using different visualisations. It shows the advantages of using the U-Net architecture compared to the baselines and describes their strengths and weaknesses. We also note that the large model gives similar performance to the small model, indicating that the architecture used has more impact in this case than the size of the model.

Finally, chapter seven draws conclusions from the study on the basis of the above-mentioned objectives. This is followed by a bibliography listing the works on which this study is based, and an appendix containing visualisations that provide more detailed information on the performances of each model regarding each metric. The code implementing the models discussed here is available on GitHub[1].

[1]https://github.com/JustinLalieu/Predicting-the-evolution-of-Antarctic-sea-ice-with-data-science

Chapter 2

Related works

Before delving into the core of this thesis, the following is a review of the literature mentioning some papers, concept and algorithms that were found useful to the research. Some of them provide a better understanding of the task at hand, while others have been used directly as a foundation for the work that follows. Not all of their aspects are discussed as those are quite comprehensive and address different approaches, the intention is rather to give the reader an overview of what is being done in the field and the wide range of methods that exist at all stages of the research.

2.1 The CNN approach and evaluation metrics

An important paper in the field of statistical SIC forecasting is *Prediction of monthly Arctic sea ice concentrations using satellite and reanalysis data based on convolutional neural networks* by Kim et al. (2019) [14]. They propose a 1-month SIC prediction model for the Arctic using a convolutional neural network (CNN) consisting of three convolutional layers and one fully connected layer, and compare its performances with a random-forest (RF) based ensemble model and an anomaly persistence model. Another objective was to assess the importance of each of their 8 input variables using a sensitivity analysis.

Models were trained by dividing the whole dataset in two sub-datasets, each then split in a training and a validation set. The first training set ranges from 1988 to 1999 and use the year 2000 as validation and the second go from 1988 to 2016 and data is validated on the year 2017. This aims at evaluating the models on years whose knew different trends due to the accelerated ice melt instead of only validating on recent years. Also, the somewhat small size of the whole dataset is noteworthy as later studies uses bigger datasets. Data itself is made of 8 input

variables starting with SIC itself, SIC anomalies, sea surface temperature, 2m air temperature, albedo and wind. They note that it is the previous SIC itself that is the most important input variable to SIC forecasting.

To compute the performances of their models, they use five different metrics starting with the mean absolute errors (MAE), root-mean-squared error (RMSE) and the normalised RMSE (nRMSE) but also using more specialised metrics with the Nash–Sutcliffe efficiency (NSE) and the anomaly correlation coefficient (ACC). Here are their mathematical expressions with SIC_g being the ground truth and SIC_p the prediction, an overline ($\overline{SIC_x}$) indicating the mean of the variable:

$$\text{MAE} = \text{mean}(|SIC_p - SIC_g|) \tag{2.1}$$

$$\text{RMSE} = \sqrt{\text{mean}((SIC_p - SIC_g)^2)} \tag{2.2}$$

$$\text{nRMSE} = \frac{\sqrt{\text{mean}((SIC_p - SIC_g)^2)}}{\text{std}(SIC_g)} \tag{2.3}$$

$$\text{NSE} = 1 - \frac{\sum(SIC_g - SIC_p)^2}{\sum(SIC_g - \overline{SIC_g})^2} \tag{2.4}$$

$$\text{ACC} = \frac{\sum(SIC_p - \overline{SIC_p})(SIC_g - \overline{SIC_g})}{\sqrt{\sum(SIC_p - \overline{SIC_p})^2}\sqrt{\sum(SIC_g - \overline{SIC_g})^2}} \tag{2.5}$$

Each of these metrics tells something different about the performances of a model. The MAE is a common metric used in data science for regression tasks. Then, nRMSE, the RMSE divided by the standard deviation of the actual SIC, allows to cope for the important number of low SIC pixels during the summer to evaluate the performances of the model by taking into account the whole range of SIC values. The NSE on the other hand, is a widely used measure in SIC forecasting that compares the observed data with the predicted values, providing a normalized statistic that indicates how well the predictions match the observations. An NSE value of 1 indicates perfect agreement between the model and the observations, while an NSE of 0 indicates that the model predictions are as accurate as the mean of the observed data. Negative NSE values indicate that the model performs worse than simply using the mean of the observed data. Finally, the ACC, which is also widely used in climatological forecasting, compute the cor-

relation between the prediction anomalies and the actual anomalies and lies thus between -1 and 1. When computing those metrics, they average them spatially and temporally to grasp a better understanding of the differences in performances according to different regions and seasons.

In terms of global results, the persistence model has the lowest performances on all metrics while RF and CNN show similar results in MAE, ACC and RMSE but CNN has a better nRMSE than the RF. They went into a deep analysis of these performances throughout the paper and using those metrics and by comparing them at regional and seasonal level, researchers have come up with a number of observations. Among them, they concluded that the CNN's error distribution was more stable than the others. Also, CNN and RF models might be weaker at predicting marginal SIC zones (sea ice close to the open ocean) than central zones and that does not apply to the persistence model, that might either be due to the fact that marginal ice is thinner and thus more prone to melting and detachment or, predictability might be smaller because the training sample over marginal zones is also smaller than for the central zones. A number of visualisations are also used, leading, among other things, to the observation that the error variance for each metrics is greater in the RF than in the CNN.

2.2 Other relevant metrics

Besides metrics described above, others exist and showcase different aspects of the performances of a model to accurately forecast SIC. In *Predicting the Daily Sea Ice Concentration on a Subseasonal Scale of the Pan-Arctic During the Melting Season by a Deep Learning Model*, Ren and Li (2023) [19] are using a revisited version of the U-Net architecture which is described in the following section (2.3). We here focus on the metrics they propose to evaluate their model and not on the results as this study focuses on the Arctic and not on the Antarctic but moreover, the framing of the problem is different than ours. Indeed, as they focus on the subseasonal scale during the melting season, they forecast SIC on a sequential manner starting from 1 to 90 lead days.

Researchers introduce a hybrid loss function based on the mean squared error (MSE) and the normalised integrated ice-edge error (NIIEE) which is intended to overcome the limitations of using only the MSE when predicting SIC. The first limitation is MSE's inability to measure the spatial similarity between the SIC_p and the SIC_g. Indeed, the MSE being only based on numerical differences, it could favour a prediction that is numerically better but spatially worse. The second

limitation is the fact that SIC varies throughout the year, thus, measures during summer when SIE is low and winter when SIE is larger could be unbalanced and lead to biased fit of the model according to seasons.

$$\text{MSE} = \text{mean}((SIC_p - SIC_g)^2) \tag{2.6}$$

$$\text{IIEE} = (SIC_p \cup SIC_g) - (SIC_p \cap SIC_g) \tag{2.7}$$

$$\text{NIIEE} = \frac{\text{IIEE}}{SIC_p \cup SIC_g} = 1 - \frac{SIC_p \cap SIC_g}{SIC_p \cup SIC_g} \tag{2.8}$$

$$\text{Hybridloss} = \text{MSE} + 0.01 \times \text{NIIEE} \tag{2.9}$$

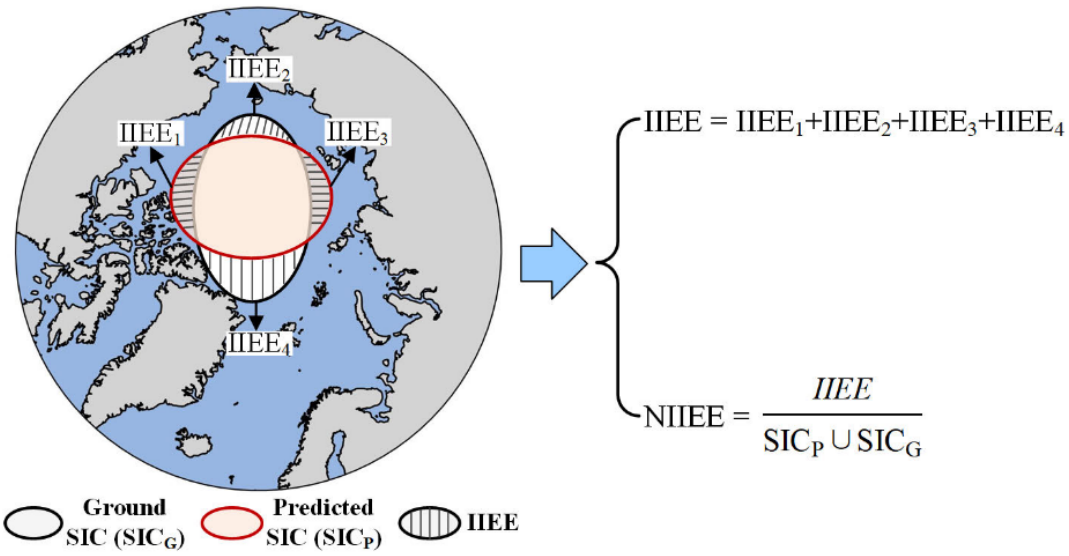

Figure 2.1: Schematic representation of IIEE and NIIEE [19]

To take into account such spatial performances of the model and not just its numerical performances, NIIEE is based on the integrated ice-edge error (IIEE) which is a commonly used metric in SIC and SIE forecasting. To use the IIEE, we binarize the SIC, values greater to 15% are set to 1 (considered as ice) and values lower are set to 0 (considered as open waters). This transformation is used

to represent the SIE based on the SIC. We then compute the sum of the areas where ice is present in SIC_g but not in SIC_p and where ice is present in SIC_p but not in SIC_g. Mathematically speaking, the IIEE is the union of SIC_p and SIC_g minus their intersection. The IIEE is then normalised to compute the NIIEE by being divided by the union between SIC_p and SIC_g, the result ranging from 0 (perfect spatial match) to 1 (no intersection). Finally, the hybrid loss is a simple function of the MSE and the NIIEE. It should be noted that in the paper, they consider SIC ranging between 0 and 1 instead of 0 and 100 which explains the loss ponderation. Figure 2.1 is a visual representation of the IIEE and NIIEE provided by the authors.

2.3 IceNet and The U-Net model architecture

One of the paper that motivated and constitutes the basis of this thesis is *Seasonal Arctic sea ice forecasting with probabilistic deep learning*, written by Andersson et al., (2021) [1]. They propose a data-driven deep learning approach to forecast the monthly averaged sea-ice concentration for the next 6 month in the Arctic. Here, the model called IceNet, is trained based on 50 input layers, among them SIC, climate variables, statistical SIC forecasts and metadata. Then, the training dataset spans on 2200 years of climate simulation data from the CMIP6 covering the years 1850-2100 on which forcing effects have been applied to account for a multitude of scenarios. The model is finally fine-tuned on the available observational data collected by satellites from 1979 to 2011. The data gathered between 2012 and 2017 are employed as validation dataset for hyperparameters tuning but not for training and the years ranging from 2018 to 2020 form the test dataset to assess the capacity of the model to accurately forecast on unseen data.

The point that really interests us is the architecture of the model used. They propose an adaptation of the U-Net architecture, an Encoder-Decoder neural network architecture using convolutional operations, initially used to segment medical data [21] and characterised by its distinctive U-shaped structure. Figure 2.2 contains a visual representation of the network given by the authors.

The encoder consists of the first five blocks with each block made up of convolutional layers followed by ReLU activations and max-pooling (downsampling) operations. This first half of the network progressively reduces spatial dimensions while increasing feature depth to capture abstract features. A batch normalisation layer is used between the convolutions and the max-poolings to standardizes the inputs of each layer, promoting smoother and faster training by maintaining con-

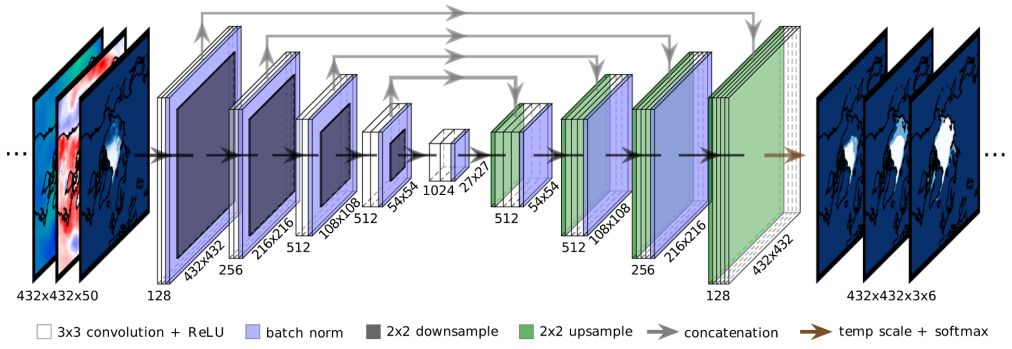

Figure 2.2: IceNet's U-Net architecture [1]

sistent data distributions throughout the network. Then, the decoder consists of the last four blocks and reverses the process of the encoder by using up-sampling (transposed convolutions) and convolutional layers to reconstruct the image's original resolution. Critical to U-Net's effectiveness are the skip connections that link corresponding layers in the encoder and decoder, preserving spatial information lost during down-sampling. The architecture ends with a series of convolutional layers to produce the final forecasted output, enabling precise predictions by leveraging both high-level context and fine details.

Regarding the problem formulation, instead of forecasting the exact SIC for a particular date, they predict the monthly averaged SIC. Moreover, they reframe the problem from the regression task of returning a percentage representing the SIC, to a classification task where the output can fall in three classes: open-water if SIC is below 15%, marginal ice if SIC is between 15% and 80% and full ice is SIC is above 80%. This explains the shape of the output, 432x432 corresponding to the data grid, x3 stands for each class and x6 is the prediction for each of the six following months. To make the model evaluation easier, they go even further and group together the marginal ice and full ice classes, the task becomes a binary classification problem that is equivalent to forecasting the sea-ice edge, i.e. where SIC is greater than 15%. Being in an imbalanced classification problem, they use a custom focal loss which is a variant of the common cross-entropy loss function. That loss is also weighted according to the month of the year and the active grid cell region history for that month. This balances the seasonal variations and the effect of the abundance of 0% SIC in the whole grid.

On results, they divided their evaluation according to the month of the year and the lead time and compared the performance of their model with that of a leading dynamical model at the time, the ECMWF's SEAS5 model [13]. IceNet

outperforms SEAS5 for 2 to 6 month lead times, SEAS5 being better only at a 1-month lead time which might be due to the fact that they forecast the averaged monthly SIC instead of the exact SIC for a particular date. Furthermore, the model's accuracy depends on the month that is predicted, its accuracy drops during summer because of what they call the spring predictability barrier, as the ice's melting is always harder to forecast than its freezing in winter, but this also depends on the area of interest, some sea's SIC being easier to forecast than others.

2.4 SIPNet, the U-Net in Antarctica

Then, in *Subseasonal Prediction of Regional Antarctic Sea Ice by a Deep Learning Model* by Wang et al. (2023) [26], authors propose a model named SIPNet to forecast the weekly averaged SIC in Antarctica at sub-seasonal time scales (i.e. 1 to 8 weeks lead time). The architecture of the model consists of an encoder-decoder structure, similar to a U-Net, but in which layers are composed of ResNet blocks. The model uses only previous SIC data at different time scales as input variables, with a dataset using satellite observations ranging from 1979 to 2021. Models are evaluated seasonally and by using MAE, ACC and IIEE. SIPNet is then compared with a persistence forecast, a linear Markov model and different dynamical models (ECMWF, NCEP, and GFDL).

Results shows that SIPNet outperforms them all, but that its performances vary according to the seasons. A better skill is observed in autumn which they explain by a higher predictability because of predictable ocean heat content anomalies, while the lowest skill occurs in spring with error mainly occurring in SIC marginal zones. Regionally, SIPNet is still performing better than the other models, its lowest average MAE being found in Weddel Sea. Although, the model shows less forecast skill in the West Pacific and Amundsen-Bellingshausen regions, but is still the best performing model even at 8 weeks lead time. Therefore, it is this study that is closest to the work that follows, even if, as we shall see in section 3.2, the framing of the problem is not the same. Nevertheless, certain comparisons are possible and are discussed in section 6.4.

2.5 The alternative ConvLSTM architecture

Other studies examine the ability of the Convolutional Long Short-Term Memory (ConvLSTM) architecture to forecast SIC [3][15][27]. Among them, the study *Deep Learning Shows Promise for Seasonal Prediction of Antarctic Sea Ice in a*

Rapid Decline Scenario written by Dong et al. (2024) [5]. Here, researchers use ConvLSTM to forecast both monthly sea-ice area (SIA) and SIE in Antarctica, with a focus on the months from December to June (summer and autumn).

As its name suggests, this architecture uses the advantages of convolutional operations directly in LSTM cells. The first allows extracting spatial information while the latter consider temporal information. Indeed, LSTM is a type of recurrent neural network (RNN) architecture designed to recognise patterns over time. Unlike traditional neural networks, which process data in a static manner, RNNs are designed to work with sequential data, making them well-suited for tasks involving time series analysis. By using this architecture, the framing of the problem is different than when working with a CNN or a U-Net as the input data format is not the same. Thus, instead of giving the model layers of punctual information, the ConvLSTM takes as input an ordered sequence of data and outputs an ordered sequence of data. They compared their model with the climatological forecast and the ECMWF dynamical model and found that the ConvLSTM performs better and shows great interannual and interseasonal performances.

Chapter 3

Materials and methodologies

As seen with related works, the problem can be framed many ways and the study can be based on a wide range of data sources. This section covers and describes the dataset used alongside a clear description of the problem. On the whole, the programming language used throughout the research is python. Data are stored using NetCDF files, which are convenient to store multidimensional geospatial data and is a widespread format in this field. All the neural networks discussed later are implemented using PyTorch and its framework PyTorch Lightning. Finally, all the plots are made using drawio and the packages Matplotlib, Seaborn and Basemap.

3.1 Dataset

Data is the most important aspect of the whole data science pipeline, it is often stated that a model can only be as good as the data it is being trained on and SIC forecasting makes no exception. Many organisms allow access to satellite observations of SIC, among them, the Copernicus Climate Change Service (CDS) [4] provides daily gridded data containing SIC for both the Arctic and the Antarctic collected by the EUMETSAT and the OSI SAF [7]. Figure 3.1 is an example data sample.

However, as mentioned in the introduction, this research is not conducted using actual satellite observations but instead using synthetic data produced by the dynamical model NEMO (Nucleus for European Modelling of the Ocean). This is a general ocean circulation model developed by a European consortium which is currently used in several countries for fundamental research, operational forecasts of the state of the ocean, seasonal forecasts and climate projections [23][17]. Data

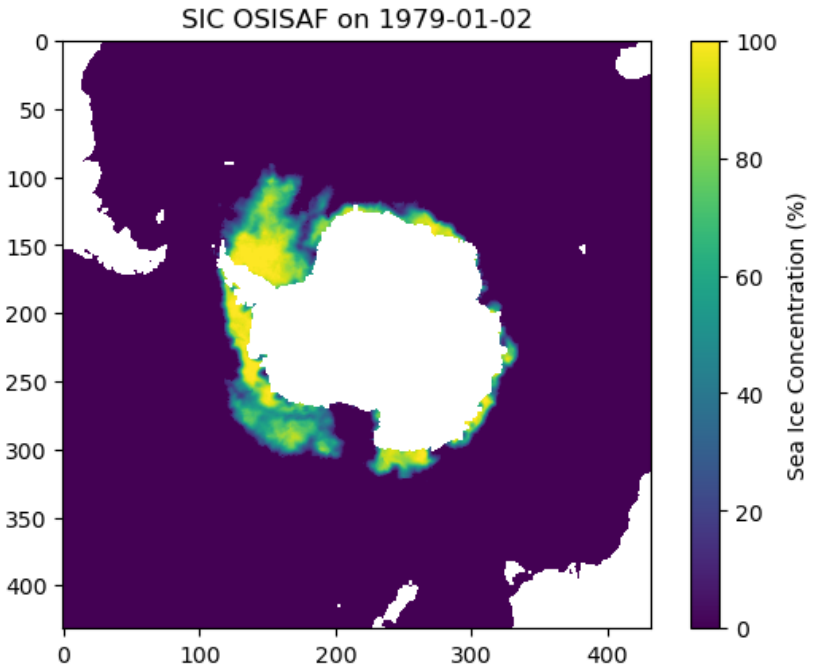

Figure 3.1: SIC data sample collected available on the CDS

were produced by the version 4.2.1 of NEMO coupled with the Sea Ice modelling Integrated Initiative (SI^3), the model as a whole being referred to as NEMO-SI^3.

One of the reasons to use synthetic data is that satellite observations are only available from 1979 to present and contain some gaps during which data collection using satellites could not be conducted. On the other hand, NEMO-SI^3 can simulate data on longer time spans and without interruptions. The dataset used in this study is therefore a simulation of NEMO-SI^3 from 1st January 1961 to 30 November 2023. This dataset has been divided into three parts, the training set from 1961 to 2016 inclusive, on which the statistical models are trained, the validation set from 2017 to 2019 inclusive which is used to tune the hyperparameters, and the test set from 2020 to November 30th 2023, which is used to evaluate the final performances of the models on unseen and recent data.

The only thing retained from the satellite observations available at CDS is the 432x432 grid (EASE-grid version 2.0) used to represent the SIC using a Lambert Azimuthal Equal-Area (LAEA) projection centred on the South Pole. Its focus on the Antarctica is great for visualisation purposes, but its real advantage is its squared shape, which is perfect to work with when using convolutional operations.

Indeed, raw NEMO-SI^3 data come using an eORCA1 grid, containing information about oceans and both poles and representing data by a 360x331 rectangular grid as seen in the left image of Figure 3.2. This is practical to store data about both poles, but only less than the half of that information being required in our case and the grid being less convenient both for visualisation and for working with convolutions, all the NEMO-SI^3 data are transformed to the LAEA grid using python's Basemap package's interp function. The interpolation result is shown in the right image of Figure 3.2.

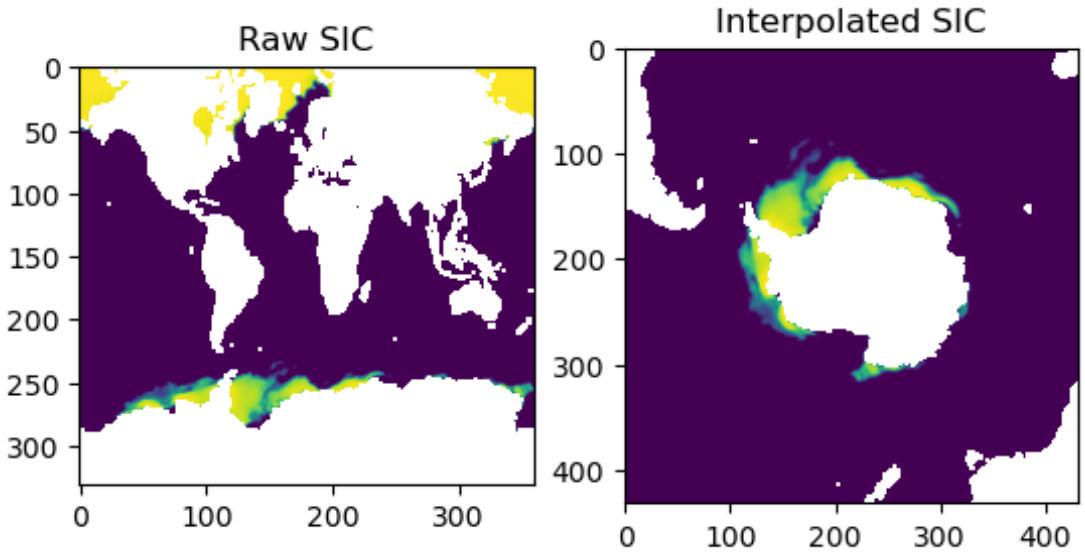

Figure 3.2: Comparison of the raw SIC outputted by NEMO-SI^3 and its interpolation

In addition to SIC, other variables are also produced by NEMO-SI^3 whose can be useful to use as input variables to our statistical model. Among them, two were selected manually based on their accessibility and the physical relationship between them and the SIC: the snow thickness and the mixed layer depth (MLD). Other important variables exists such as the heat content, the upward solar radiations or the sea level pressure [1] but the goal is not here to perform a comparative study of the impact of those variables but rather to see if their impact in the whole data science pipeline matter as much as data preprocessing or the model's architecture itself for example.

A brief description of the manually selected variables is necessary, Figure 3.3 contains a visual representation of each. Of the two, snow thickness is the most straightforward, it describes the amount of snow in metres that covers the underlying ice. Mixed layer depth is less obvious, as explained by Treguier et al. [22], the mixed layer is the interface between the ocean interior and the atmosphere or sea ice, its depth corresponds thus to the depth of the deepest layer where interactions occur between the ocean and the atmosphere or sea ice. Finally, when predicting for a particular date, the three variables available are SIC, snow thickness and MLD and each is available at 30, 60, 90 and 365 days before the date in question. These time intervals have been chosen to take account of the trend in the evolution of the SIC over the last three months and the state of the SIC the previous year, and to avoid cluttering the model with too many variables. The SIC climatological variable is also available as the mean SIC at the same date for a certain number of years.

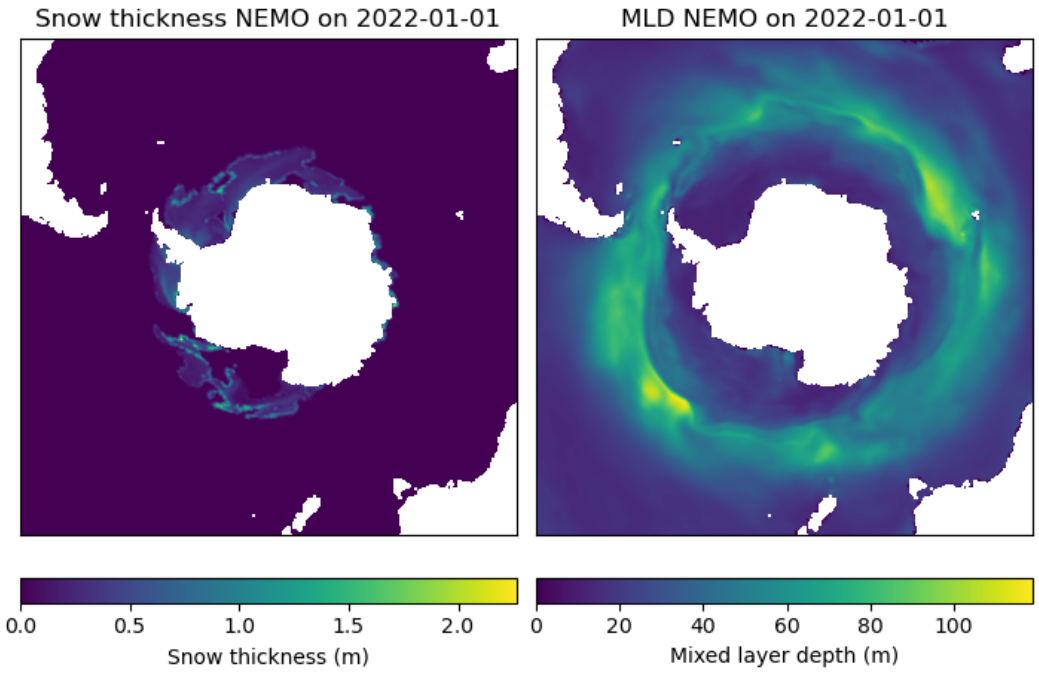

Figure 3.3: Snow thickness and MLD outputted by NEMO-SI^3, after interpolation

3.2 Methods

The problem can be framed many ways, by either forecasting the SIC or the SIE, for days or for months, with a focus on particular regions or studying the Antarctica as a whole. In this study we chose to forecast the SIC at a one month (30 days) lead time for the whole Antarctica. Thus, when forecasting for a particular date, each model only has access to information that is 30 day old or more than this date. While other studies try to forecast for longer time spans going up to 6 months, this research focuses more on the impact of the architecture and the hyperparameters used rather than on the maximum achievable prediction range.

Before being fed to the models, data are loaded on the fly using a custom PyTorch Dataset class to avoid taking up too much memory space. Training data are shuffled to avoid the gradient descent to be impacted by the seasonal pattern of the target variable, but the validation data are not. At this stage, data are normalised and standardised in order to improve the performance and stability of the training and predicting steps. This preprocessing of the data ensures that the input features have consistent scales and distributions. For instance, SIC is expressed as a percentage between 0 and 100, but MLD which is expressed in meters can take values between 0 and several hundred metres. First, the normalisation (Eq. 3.1) consists of a min-max feature scaling step which brings data values between 0 and 1. Then, data are standardised (Eq. 3.2) so that each variable has a mean of 0 and a standard deviation of 1. As the dataset used consists of variables with different scales and units, this avoids any feature to be too significant compared to the others due to greater values but also make the gradient descent converges faster and be more stable during training. Once data has been processed through the model, the inverse operations are applied to make the output comparable with the truth values of SIC.

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{3.1}$$

$$x' = \frac{x - \text{mean}(x)}{\text{std}(x)} \tag{3.2}$$

During training, a loss function is required to optimise the model's parameters by gradient descent and to determine their optimal state. No hybrid loss as proposed by Ren et Li (2023) [19] was used as no particular benefit or improvement was found. The MAE (2.1) is chosen as this is a widely used metric in SIC forecasting [14][19][26] and it has the advantage of being expressed in the same unit as

the output which makes it easily interpretable. Also, another advantage of MAE over MSE is that it is less impacted by outliers as by squaring the error, MSE gives a higher weight to large errors than to small ones. Besides the loss function, other metrics are used to monitor the performances of each model during both training and predicting on the validation and test sets. Each measure has been chosen on the basis of what it can tell us about the performance of a model and on the basis of the existing literature.

To assess the numerical performances, in addition to the MAE, the NSE (2.4) is also used, while spatial performances are computed using the IIEE (2.7) and NIIEE (2.8). Those metrics are used because they each provides a unique perspective on the performance of SIC forecasts. MAE helps in identifying the overall accuracy of the model by quantifying how far off the predicted SIC values are from the actual values on average while NSE indicates how well the model's predictions align with the actual observations beyond just the magnitude of the errors and compares it directly with the observed average value. IIEE evaluates the raw spatial accuracy of SIC forecasts while NIIEE which adjusts the IIEE by dividing it by the total length of the ice edge in the observed data, makes the models comparable on a consistent scale across different seasons with varying SIE.

After being trained, if using the same input for prediction, a model will always output the same forecast as its weights are set and no random parameters are part of its architecture. However, when training a new model, the weights are initialised randomly, which means that two distinct models trained using the same dataset and hyperparameters and predicting with the same input will output a different forecast. Thus, the random initialisation of the weights influences their convergence, and even if the models were trained long enough, the final weights might be different from one run to another. However, after experimentation, this variability is small enough to be negligible as soon as the number of training epochs is sufficient and the learning rate and optimiser are appropriate. In this case, the number of epochs and the learning rate are given for each model, and the optimiser used is Adam. Therefore, for each model discussed here, only the best of several runs is kept.

A small post-processing step is applied to the predictions of the models implemented using PyTorch as they do not have a sense of the lower and upper bound implied by a percentage value. Negative values are thus set to 0 while values above 100 are reduced to 100 as those values would not make sense otherwise.

Once trained, each model is thus evaluated on the validation set and based on the metrics discussed above in order to tune the associated hyperparameters which are described in each corresponding model section. For each model architecture, the main hyperparameters requiring tuning are the learning rate, the number of training epochs and the input variables. The learning rate determines the step size at each iteration while moving toward a minimum of a loss function. It controls how much to change the model in response to the estimated error each time the model weights are updated. If set too high, the model might exceed the optimal point and not converge towards the best solution, but if too low, the convergence might take a long time or get stuck in a suboptimal solution. The number of epochs is a hyperparameter that defines the number of times the learning algorithm will work through the entire training dataset. If too few epochs, the model might underfit the data and not learn its underlying patterns well enough, but if too many, it might overfit the data, learning the training set too well including its noise and outliers and thus performing poorly on unseen data. Finally, for the input variables, including irrelevant features can confuse the model and lead to poor performances, but not including all the relevant features might lead the model to miss some patterns from the data. The final performances of the selected models are then compared to the test set which consists of data unseen by the models, either during training or during hyperparameters tuning.

Chapter 4

Baselines

To assess the performance of a model, it is necessary to compare it with simpler models, the baselines. This is where the persistence and climatological forecasts and the linear regression model come in useful. As these are simple ways of providing a prediction, they represent the baseline performance that any more complicated model needs to achieve to be relevant. So that the performance of baselines can be easily compared with that of more complex models, they are implemented in PyTorch using trivial models.

4.1 Persistence and climatology forecast

The first and simplest way to provide a forecast is by predicting that the current weather conditions will continue for a certain period. This is the idea behind the persistence forecast, which is a basic method widely used as a baseline in climate forecasting. As in our case, the models have information on data from a month ago compared to the prediction, this is also the case for baselines. For example, the persistence forecast can predict that the SIC in one month time will be identical to what it is now. Or, since the SIC follows a seasonal cycle, it can forecast that it will remain the same as it was a year ago.

A slightly more complicated, but still simple, approach is the climatology forecast. This is a type of weather forecast based on average weather conditions for a specific period of the year. An example would be to forecast that the SIC on a given date will be the average of the SIC over the last 5 or 10 years on the same date.

4.2 Linear model

While persistence and climatology are using a single variable, a linear model can use multiple ones. By associating a weight to each variable, this approach can take several parameters into account and weight them to provide a more accurate forecast. In fact, the baselines discussed above are special cases of a linear model for which the weight of one variable is 1 and that of all the others are reduced to 0. The advantage of the linear approach is that its weights are interpretable. Thus, if the weight of a variable is 0, it is not useful in the context of a linear model, whereas if one weight takes on a higher value compared to the others, it means that the variable is relevant.

There are many possible approaches to implementing a linear model, we have chosen to use PyTorch here because of the great versatility it allows when implementing the model and the use it makes of the input data. In addition, it facilitates data loading by using the same methods for all the models discussed in this study, and also makes it easier to compare model performance during the development phase using tools such as TensorBoard. Therefore, in addition to the input variables, two other hyperparameters are used to train the models. The learning rate determines how much the weights of the model will change during training at each iteration while moving toward a minimum of a loss function. Then, the number of epochs refers to the total number of times the entire dataset is passed forward and backward through the neural network during training.

Here, the structure of the model is such that each variable input is assigned a weight, with the output of the model being the sum of the variable inputs multiplied by their respective weights. Thus, for three variables, the model is composed of three weights which are updated during training by gradient descent with a step determined by the learning rate in order to minimise the loss function. Other approaches are conceivable regarding the architecture of the linear model, one could consider one weight per pixel and per variable instead of one weight per variable. However, after experimentation, this approach turned out to be ineffective, as the model failed to minimise the loss function due to the excessive number of parameters and the lack of interaction between them, resulting in an unstable and divergent model that was overly influenced by the latest training samples encountered.

4.3 Results

First, the relative performances of persistence and climatological forecasts are studied based on the metrics described in section 3.2 Methods. Let us say we want to predict the SIC at date x, the information available to the model is all the data from date x-30 days and before. The two most relevant persistence forecasts are therefore those based on data from 30 days ago (Persist-30), as they are the closest in time to the date to be predicted, and data from 365 days ago (Persist-365), as the SIC follows a seasonal cycle and should be more or less identical from year to year. For climatological forecasts, it only makes sense to take the average of the SICs spaced 365 days apart, the remaining question is how many years are taken into account. Here we are considering climatological forecasts that take the average of the last 5 (Clim-5x365), 15 (Clim-15x365) and 30 (Clim-30x365) years. Table 4.1 contains the mean of each metric over the whole validation set. As a reminder, the lower the MAE, IIEE and NIIEE, the better, while for the NSE, the higher and closer it is to 1, the better.

	Persist-30	Persist-365	Clim-5x365	Clim-15x365	Clim-30x365
MAE	2.721	1.721	1.657	**1.615**	1.664
NSE	0.360	0.705	0.778	**0.816**	0.814
IIEE	5395.100	**3505.441**	4286.307	3953.296	3944.671
NIIEE	0.363	0.280	0.297	0.278	**0.276**

Table 4.1: Mean of each metric for each forecast over the validation set

The first thing that stands out is the poor performance of Persist-30 compared with the other approaches. This makes sense since this forecast is always 30 days behind the date to be forecast, whereas the others are much more relevant thanks to SIC's regular seasonal cycle. Then, the best numerical performances are achieved by the Clim-15x365 which seems to greatly average the SIC seasonal cycle, but it is the Persist-365 that has the best spatial performances, which by the way is a special case of climatological forecast equivalent to a Clim-1x365. This shows that forecasting SIC and SIE is not the same task and that a model can be good at one but weak at the other. Those two forecasts are kept to be compared with the linear models to select the best baselines.

Different set of input variables are given to the linear model to assess their relative usefulness in a linear context, details and final weights rounded to two figures are given in Table 4.2, if left blank, the variable was not considered by this model. The linear models considered here are trained with a learning rate of 0.01 for 5 epochs. However, as shown by Table 4.2 and Table 4.3, this does not lead to

a stable convergence of the weights. This can be explained by too great an impact from the random initialisation of the weights and by too few parameters to describe a complex dataset. This indicates that using gradient descent to implement a linear model for this particular task might not be the most stable and efficient approach. However, this method is sufficient to obtain better results than with persistence and climatological forecasts and gives an idea of the performance's order of magnitude that a linear model can achieve, which is sufficient in the scope of the present study.

	SIC Clim	SIC				Snow thickness				MLD			
Lead time	15x365	30	60	90	365	30	60	90	365	30	60	90	365
Lin_A	0.96	0.04											
Lin_B	0.39	0.02			0.59								
Lin_C	0.57	0.04	0		0.40	0							
Lin_D	0.51	0.03	0	0	0.47	0	0	0	0	0	0	0	0

Table 4.2: Weights for each variable after training with linear models

Despite the poor convergence, global trends can be interpreted. As expected, the most important variables are climatology and the 365-day SIC, while the 30-day SIC has only a slight influence over all the other variables, which are not considered at all. In the end, even though some models use more input variables than others, the variables they consider are the same, which is reflected in the metrics. Only the model that does not use the 365-day SIC has a lower spatial performance than the others.

	Persist-365	Clim-15x365	Lin_A	Lin_B	Lin_C	Lin_D
MAE	1.721	1.615	1.606	1.610	**1.583**	1.606
NSE	0.705	0.816	**0.823**	0.792	0.813	0.807
IIEE	3505.441	3953.296	3934.615	**3219.806**	3480.722	3376.413
NIIEE	0.280	0.278	0.276	**0.253**	0.257	**0.253**

Table 4.3: Mean of each metric for each potential baseline over the validation set

To gather more information on the performance of the models, we represent the error of each model as a function of each metric using boxplots in Figure 4.1. This visualisation helps to compare models by highlighting their variability, central tendency, and any outliers in their performance. By doing so, it becomes easier

to discern which model performs consistently better across different metrics and which ones may have issues with variability or outlier errors.

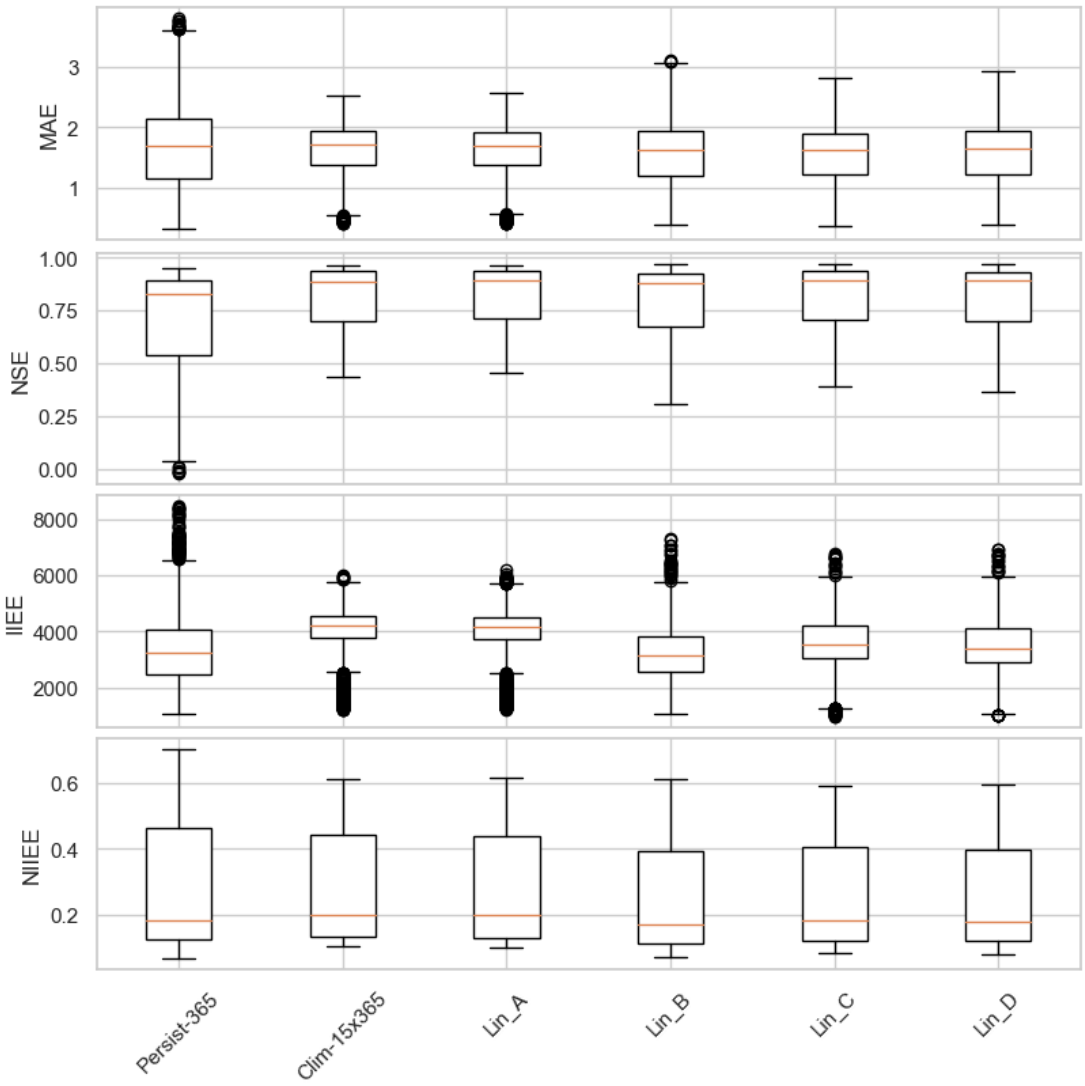

Figure 4.1: Median and variability of error of each metric for each potential baseline over the validation set

Each boxplot represents the performances of a model according to a metric, displaying the distribution of error values through its quartiles. The central box spans from the first quartile (Q1) to the third quartile (Q3), capturing the interquartile range (IQR) which contains the middle 50% of errors, with the line inside the box indicating the median error. The whiskers extend to the smallest

and largest values within 1.5 times the IQR from Q1 and Q3, respectively, depicting the range of the bulk of error values. Outliers, which are errors beyond this range, are plotted as individual points.

A single model is enough to form the baseline, this model should show good numerical and spatial performances but also a low error variance. From Table 4.3 and Figure 4.1, we can see that, despite its good spatial performance, the persistence forecast shows a high variance in its errors with some of them being below those of the other models. For the climatological forecast, the opposite is true, it is the spatial performances that are worse. However, despite a significant difference in IIEE between these two models, their NIIEE is almost equal. This is explained by the fact that the NIIEE is the total number of overestimated and underestimated pixels divided by the number of pixels of the union between the prediction and the observation. A possible interpretation is that, either the persistence tends to make higher spatial errors when SIE is low, or the climatology tends to make lower errors when SIE is high, or both. It shows the importance of using several metrics to understand the strengths and weaknesses of each model, other tools for further interpretation will be used later.

On the linear models side, Lin_A is necessarily similar to climatology as the impact of the 30-days SIC is slight. This leaves us with the three other models whose are actually using the same parameters, the only difference being the weight attributed to each of them. They still consider the SIC 30 days ago as less significant in this linear context but do not agree whether to give more impact to the 365-days SIC or to the climatology variable. The two opposite models are Lin_B and Lin_C, the first giving more weight to 365-days SIC and the second to the Clim variable. The fact that Lin_B shows better spatial performances because it gives more weight to the 365-days SIC means that the SIE and its shape tends to be similar one year after another but inconsistent in the long run. Lin_C however has great numerical performances meaning that SIC tends to vary one year after the other but is more consistent in the long run. As the error variance of the linear models is consistent, we can not favour one of them based on that. Lin_D finally is a compromise between the two leading to intermediate results.

The baseline used for the rest of this document will therefore be the Lin_D model that is later referred to as Linear.

Those observations are allowed by the interpretability and low number of the weights of linear models. Also, they only apply to a linear context. Indeed, the fact that snow thickness and MLD are not considered by the linear models is not

surprising, but this is not necessarily the case when using non-linear models, such as the U-Nets discussed in the next chapter, which use ReLU activation functions. The down-side of those models being that their weights are not interpretable due to the introduction of non-linearities but more importantly to the large number of weights that constitute them.

Chapter 5

U-Net models

Before delving into the heart of the matter, a brief history of the model is required. The model architecture on which this study focuses is that proposed in 2015 by Ronneberger et al. [21] and which was originally intended for medical image segmentation, the U-Net. In 2021, Andersson et al. have applied the same neural-network architecture to SIC forecasting in the Arctic and reported encouraging results. Since then, other studies have followed and investigated different aspects of this type of model, still in the Arctic [19][20]. In 2023, Wang et al. [26] used a U-Net architecture including ResNet blocks to forecast SIC in Antarctica. This makes this study the closest to the present work since the predicted variable is the same while the metrics and the neural network architecture used are similar. However, it should be borne in mind that they train their model on satellite observations whereas we use the output of the NEMO-SI^3 dynamic model. The other major difference is that Wang et al. predict from one to eight weeks lead time, whereas we have chosen to focus on one month lead time. A comparison of their model's results at four weeks lead time is therefore possible with the results of the present work and is discussed in section 6.4.

In this study, the required size of the network is compared using two versions of the U-Net, the first being a reproduction of the IceNet used by Andersson et al. whose architecture is shown in Figure 2.2 and the second is a reduced version with fewer blocks and convolutional layers within each block. We call the first U-NetLarge and the second U-NetSmall, the larger version having more weights and takes longer to train.

After discussing the implementation of the two U-Nets, the hyperparameter tuning procedure is described. Among other things, this makes it possible to consider the impact of the chosen input variables as the non-linearities induced by ReLU activation functions allow using other input variables than only SIC which

was not possible in a linear context. Each network is then evaluated numerically and spatially, as in the case of baselines, but also from a regional and seasonal point of view. Finally, we are briefly looking at the model's ability to predict on particular years which have been impacted by major climatic phenomena such as ENSO events.

5.1 Implementations

To compare the relevance of the model's size and parameters, two implementations of the U-Net architecture are used. Figure 5.1 contains a representation of U-NetLarge which has the same architecture as the IceNet model [1] in Figure 2.2, the only difference being that instead of using a softmax to form the output, we chose to use the output from the last convolutional layer because the problem framing is different here. Indeed, whereas Andersson et al. transform the problem into a binary classification using a softmax layer that assigns each output pixel a class, we chose to maintain the problem in the form of a regression in order to be able to evaluate the numerical performances of the model in addition to the spatial performances. It should be noted that n corresponds to the number of input layers given to the model, one input layer being a 432x432 representation of a single variable for a single day. The output is made up of just one layer since the model is predicting for a single precise date. The same applies to the U-NetSmall model which is represented in Figure 5.2.

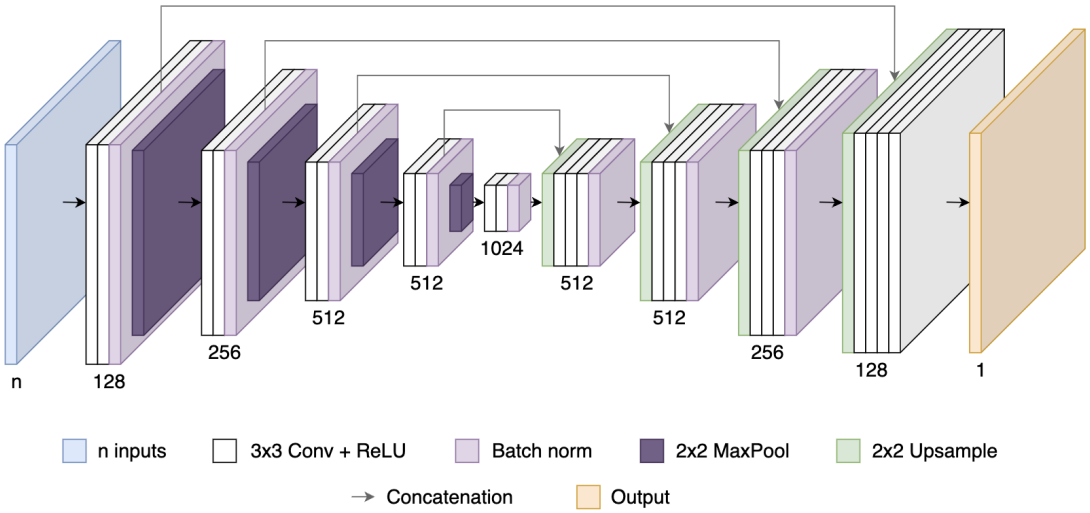

Figure 5.1: U-NetLarge, reproduction of the IceNet architecture [1]

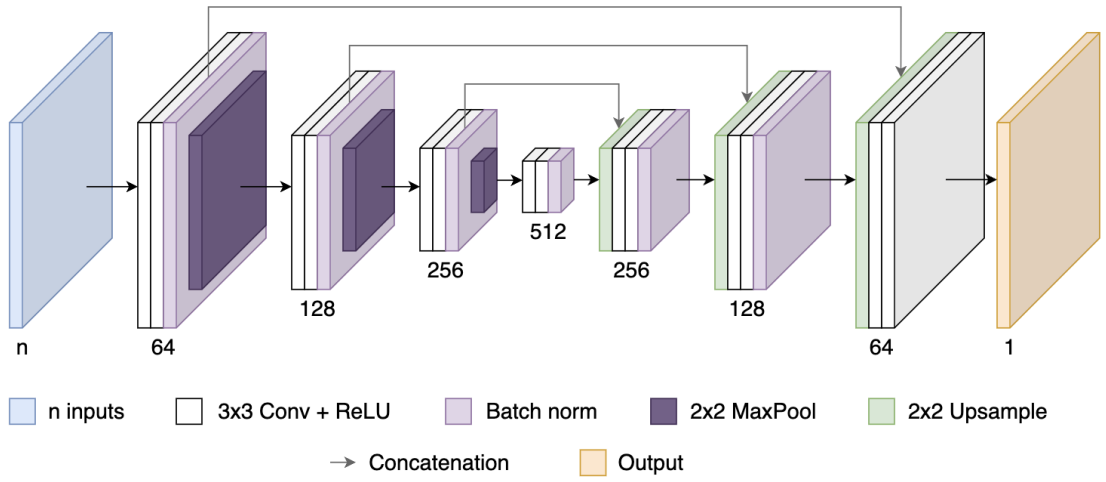

Figure 5.2: U-NetSmall, reduced version of the U-NetLarge architecture

As proposed by Andersson et al. [1], the U-NetLarge architecture consists of 5 encoder blocks and 4 decoder blocks, each containing between 128 and 1024 layers of feature maps. Meanwhile, U-NetSmall is made of 4 encoding blocks and 3 decoding blocks with between 64 and 512 layers of feature maps. The U-NetSmall's hyperparameters (number of encoding-decoding blocks and layers) have been chosen arbitrarily, the goal being to implement a U-Net significantly smaller than the U-NetLarge to compare the impact of the model size on the performances. Table 5.1 contains the number of parameters and the memory size required during training for a single input (batch size is equal to 1) of size 432x432 pixels according to the architecture and the number of input layers of the model.

	n input layers	Number of parameters	Input size (MB)	Forward/Backward pass size (MB)	Parameters size (MB)	Training size (MB)
Large	1	49,236,865	0.75	2790.40	196.95	2988.10
	13	49,250,689	9.70	2790.40	197.00	2997.11
Small	1	7,698,881	0.75	1112.28	30.80	1143.82
	13	7,705,793	9.70	1112.28	30.82	1152.81

Table 5.1: Number of parameters and training size for each architecture for a single input (batch size is equal to 1)

Here is a description of each column in Table 5.1. The n input layers correspond to the number of input variables and can range from 1 to 13. The number of parameters is the number of trainable weights and biases that are updated during the training of the model. The input size corresponds to the memory required to store the input data only. It depends on the number of input layers, but has very little influence on the overall size of the model. Most of the model size is due to the forward backward pass (FBP) size which corresponds to the memory required only during training to update the weights and biases. Then, the parameters size represents the memory space used to store the parameters and is thus directly proportional to their number. Finally, the training size corresponds to the memory required to train the model and is the sum of the input size, FBP size and parameters size. Once the model is trained, when only predicting, the space occupied by the model is only the size of the input plus the size of the parameters. Note that the numbers given in the table are for a batch size of 1. In practice, during training, the batch size used is larger and worth 16 or 32 depending on the resources available. The batch size has a direct impact on the input size and the FBP size, but not on the number of parameters and therefore the parameters size. In fact, a batch size of 16 multiplies the size of the input and the FBP by 16, resulting in a training size of around 22GB for the U-NetSmall model and of 45GB in the case of U-NetLarge.

The first observation in Table 5.1 is that the number of layers has little effect on the number of parameters. This is because it only influences the number of parameters in the first convolutional layer, but not the others. It is useful because it means that considering a large number of input variables does not cost significantly more resources. However, this does not necessarily mean that performances will improve with the number of input variables.

Secondly, the number of parameters and the training size are not proportional because of the FBP size. For a batch size of 1, the U-NetSmall, which has around 7.7 million parameters, has an FBP size of 1.1 GB, whereas the U-NetLarge, with its 49 million parameters, only has an FBP size of 3 GB. This means that U-NetLarge, which has more than 6 times as many parameters as U-NetSmall, only takes up 3 times as much space during training. These ratios remain the same regardless of the batch size, since only the number of parameters and therefore their size are not multiplied by the batch size and are negligible in relation to the final training size. This is one of the advantages of this type of architecture, which scales well with its number of parameters. It remains to be seen whether this greater number of parameters results in better performances.

5.2 U-NetSmall and hyperparameters tuning

Once the architectures have been implemented, the next step is to find the appropriate hyperparameters to optimise the performances. We begin with U-NetSmall due to its smaller size which implies a faster training time until convergence. Given that both neural networks share a U-Net architecture, some of the results of hyperparameters tuning of U-NetSmall will already provide a fair idea of the optimal hyperparameters for U-NetLarge. The first parameter to be set is the learning rate as it is a fundamental hyperparameter that directly influences how quickly and accurately the model learns from the data. It controls the step size at each iteration while moving toward a minimum of the loss function and if poorly chosen it can cause the model to converge too slowly, converge to a suboptimal solution, or even diverge. All the other hyperparameters are kept constant, models are all trained for 5 epochs and are considering the 30-days and 365-days SIC as well as the 15-year climatology, results are shown in Table 5.2.

	Linear	U-NetSmall (N epochs = 5)			
Learning rate		0.1	0.01	0.001	0.0001
MAE	1.606	7.978	1.210	**1.150**	1.728
NSE	0.807	-0.104	0.844	**0.849**	0.771
IIEE	3376.413	17320.422	**2144.506**	2159.638	2524.508
NIIEE	0.253	1.000	0.192	**0.186**	0.208

Table 5.2: Mean of each metric for the U-NetSmall model with different learning rates

As expected, the learning of 0.1 being too large leads to terrible results due to a complete divergence of the model, which fails to learn during training and returns only matrices full of zero values. Conversely, a learning rate of 0.0001 is too small and leads to slow convergence, which shows worse numerical results than the linear baseline. On the other hand, learning rates of 0.01 and 0.001 give similar results that are already better than the linear baseline. This similarity can be explained by the use of the Adam optimizer, which adjusts the learning rate during training to reduce the impact of its initial choice, leading to greater stability. Between these two learning rates, it is better to choose the smaller one, which could lead to a smoother learning process, with more stable convergence, and could also help to avoid overshooting the optimal solution. This does not mean that the best learning rate will be the same for U-NetLarge, but it is likely to be similar.

The second hyperparameter to adjust is the number of epochs. If it is too small, the model will underfit and thus not have time to learn the patterns in the input

data, but if it is too large, it risks learning them too well and therefore overfitting. The process is the same as above, this time the number of epochs changes but the learning rate and input data remain the same, results are shown in Table 5.3.

	Linear	U-NetSmall (Learning rate = 0.001)				
N epochs		1	3	5	7	10
MAE	1.606	1.632	1.268	**1.150**	1.291	1.384
NSE	0.807	0.750	0.839	**0.849**	0.790	0.832
IIEE	3376.413	2786.562	2167.628	**2159.638**	2400.468	2269.022
NIIEE	0.253	0.223	0.191	**0.186**	0.200	0.192

Table 5.3: Mean of each metric for the U-NetSmall model with different number of epochs

The best performances are obtained with 5 epochs, with less the model underfits and with more it overfits the input data. Once again, the optimal number of epochs for U-NetLarge should be slightly greater given the greater number of parameters, but not significantly greater. The final hyperparameter to tune is the input variables given to the model. Unlike the learning rate and the number of epochs, the best set of input variables for U-NetSmall should be identical to that for U-NetLarge. Due to the high number of combinations possible from our yet restricted dataset, only the most meaningful ones are tried here, they are indicated in Table 5.4 and the corresponding results are shown in Table 5.5 and Figure 5.3. Unlike linear models, the weights are not interpretable, so a cross is indicated if the variable is used by the model.

	SIC Clim	SIC				Snow thickness				MLD			
Lead time	15x365	30	60	90	365	30	60	90	365	30	60	90	365
UNetS_A		X											
UNetS_B	X	X			X								
UNetS_C	X	X	X	X	X								
UNetS_D	X	X	X	X	X	X	X	X	X				
UNetS_E	X	X	X	X	X					X	X	X	X
UNetS_F	X	X	X	X	X	X	X	X	X	X	X	X	X
UNetS_G	X	X	X	X	X	X				X			

Table 5.4: Input variables selected for each U-NetSmall model

	UNetS_A	UNetS_B	UNetS_C	UNetS_D	UNetS_E	UNetS_F	UNetS_G
MAE	1.785	1.150	1.179	1.209	**1.001**	1.038	1.060
NSE	0.782	0.849	0.856	0.868	0.878	0.880	**0.881**
IIEE	2786.197	2159.638	2119.615	2092.924	**1841.370**	1875.007	1899.472
NIIEE	0.259	0.186	0.185	0.174	**0.163**	0.166	0.166

Table 5.5: Mean of each metric for the U-NetSmall model with different input variables

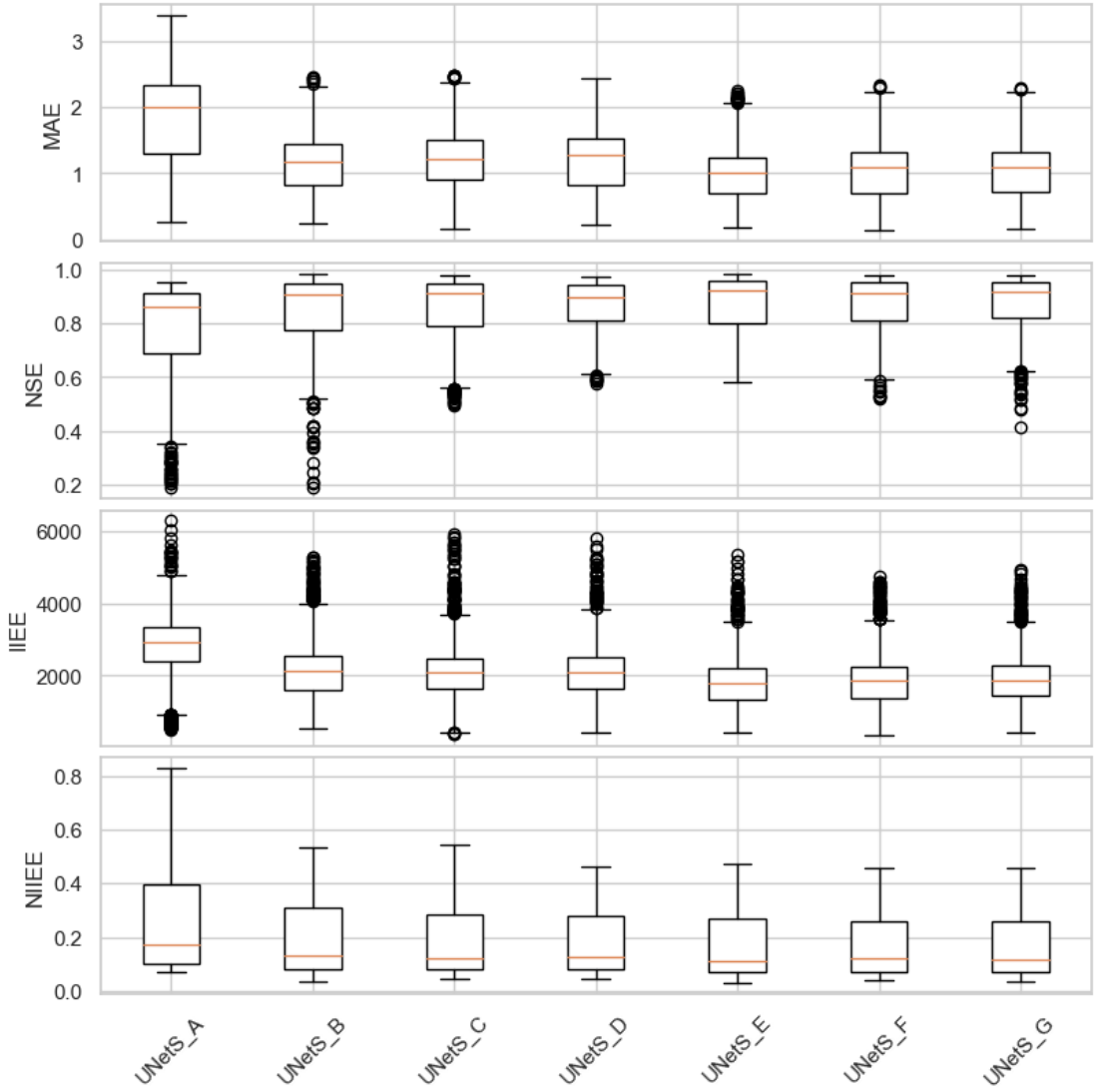

Figure 5.3: Median and variability of error of each metric over the validation set for U-NetSmall for each different set of input variables

The results have a clear tendency to improve the more input variables are used, but not all of them are responsible for the same improvement. Foremost, as seen with UNetS_A, using only the 30-days SIC, i.e. the most recent information available to the model, does not produce better results than the linear baseline and has the worst error variance among all the models. By considering more past SIC data, performance improves rapidly, both numerically and spatially, as indicated by the UNetS_B and UNetS_C results. In addition, the error variance is significantly reduced compared with the UNetS_A. model. All this shows the value of using several time ranges for the same type of variable input, in this case the SIC.

When comparing the benefits of adding the snow thickness and MLD variables to the set of input variables considered by the model, several things stand out. The difference between the UNetS_D and the versions previously discussed is small, with only a slight improvement in spatial performance and the NSE while no improvement on the MAE. This indicates that the model is better at predicting the position of the sea ice as well as the variance intrinsic to SIC variations, but does not improve in terms of numerical accuracy. However, the variance in the model's error is also slightly reduced, which means that by adding more information to the data available to the model, it is better able to predict the SIC accurately and consistently.

The actual improvement occurs when the MLD is considered by the model, as shown by the performance of UNetS_E, UNetS_F and UNetS_G, which have the best performance in all aspects and the lowest error variance. This suggests the relevance of taking the MLD into account when forecasting the SIC, whereas for the snow thickness, no improvement in performance has been observed. This is confirmed when these two variables are taken into account by the model, as is the case for UNetS_F, where performance is similar to that of the UNetS_E model using only the MLD. Finally, UNetS_G, which considers only the 30-days snow thickness and MLD, shows slightly poorer results than UNetS_E and F, which indicates the pertinence of considering these variables on different time scales in order to extract the inherent trends. In addition, since adding more variable inputs has a negligible impact on the number of parameters and therefore the training time of the model, it does not make the model more complicated to run.

This gives the optimal UNetSmall model which is trained with a learning rate of 0.001 for 5 epochs on the SIC and MLD variables at 30, 60, 90 and 365 days as well as the SIC climatology over 15 years at the same date. This model is referred to below as UNetS.

5.3 U-NetLarge and hyperparameters tuning

The information previously gathered during the tuning of the hyperparameters of the UNetSmall network can be used to facilitate the tuning of the UNetLarge network, whose training time is longer given its greater number of parameters. This already gives an order of magnitude to expect for the learning rate and the number of epochs, but above all it indicates the optimal input variables to use with this architecture for this particular problem. Therefore, only the learning rate and the number of epochs remain to be tuned in the context of UNetLarge, and the variables used are the same as those of the UNetS_E model in the previous section.

Since the UNetLarge is larger and more complex than the UNetSmall, its optimal learning rate is expected to be equal or smaller than 0.001. So, given that the learning rate of 0.1 used with UNetSmall led to the trivial output of 0% SIC everywhere, the smallest learning rate considered here is 0.01. Results are shown in Table 5.6, models are trained over 5 epochs.

	Linear	UNetS	U-NetLarge (N epochs = 5)			
Learning rate			0.01	0.001	0.0001	0.00001
MAE	1.606	**1.001**	7.978	1.069	1.079	1.146
NSE	0.807	**0.878**	-0.104	0.877	0.864	0.868
IIEE	3376.413	**1841.370**	17320.422	1845.040	2021.361	2134.007
NIIEE	0.253	**0.163**	1.000	0.165	0.173	0.183

Table 5.6: Mean of each metric for the U-NetLarge model with different learning rates

Here, given the greater complexity of UNetLarge, a learning rate of 0.01 results in a model that diverges completely from the solution by returning only SIC percentages with a value of zero. When trained over 5 epochs, the model performs best with a learning rate of 0.001, as is the case for UNetS, but it fails to perform better than the latter, both numerically and spatially. With smaller learning rates, performance decreases. However, it is relevant to compare the impact of a learning rate of 0.0001 as it might be able to achieve better convergence on a larger number of epochs. Table 5.7 contains the performances of the model when trained with a learning rate of 0.001 on different number of epochs, the same applies to Table 5.8 but with a learning rate of 0.0001.

	UNetS	U-NetLarge (Learning rate = 0.001)				
N epochs		3	5	6	7	10
MAE	**1.001**	1.169	1.069	1.009	1.014	1.105
NSE	0.878	0.873	0.877	**0.885**	0.879	0.864
IIEE	**1841.370**	1964.429	1845.040	1879.798	1861.161	1984.779
NIIEE	**0.163**	0.177	0.165	**0.163**	0.166	0.176

Table 5.7: Mean of each metric for the U-NetLarge model with a learning rate of 0.001 and with different number of epochs

	UNetS	U-NetLarge (Learning rate = 0.0001)				
N epochs		5	7	10	13	15
MAE	**1.001**	1.079	1.042	1.027	1.050	1.086
NSE	0.878	0.864	**0.879**	0.870	0.870	0.872
IIEE	**1841.370**	2021.361	1977.262	1953.705	1956.098	1985.871
NIIEE	**0.163**	0.173	0.169	0.169	0.170	0.169

Table 5.8: Mean of each metric for the U-NetLarge model with a learning rate of 0.0001 and with different number of epochs

The results show different trends between the two learning rates considered depending on the number of epochs used. For 0.001, the optimal number of epochs ranges from 5 to 7, the best results being attained after 6 epochs, if smaller the model does not completely converge and if larger it overfits the input data. With the smaller learning rate of 0.0001, the model converges later, at around 10 epochs. Indeed, we can see an improvement in the results as the number of epochs increases, but this trend is reversed once this threshold has been passed. However, and this is particularly true when the learning rate is equal to 0.0001, the difference in performance for each metric is small and the model delivers relatively constant performance as a function of the number of epochs, which indicates that for this configuration, the final weights are less influenced by the number of epochs because of the optimizer used.

Despite this tuning of the learning rate and the number of training epochs on the UNetLarge, the model delivers performances similar to the UNetSmall without being able to do better. However, it is still relevant to compare the performance of the two architectures in greater depth, given that they may show differences in terms of seasonal or regional performances. Therefore, the version of UNetLarge providing the best results is the one trained with a learning rate of 0.001 over 6 epochs, and it is this version that is used in the following section and referred to as UNetL.

Chapter 6

Results

Given that the models have been defined, trained on the training set and tuned on the validation set, it is now possible to evaluate their performances on the test set, which contains data unseen by the models. This allows to assess the real performances of a model to generalize to future data, as if the model were actually being applied. We are interested here in different aspects of these performances, primarily numerical and spatial, as in the case of hyperparameter tuning, then seasonal and regional, in order to understand them better and draw further observations from them. Four models are compared in this chapter, they are detailed in Table 6.1 as well as the hyperparameters they use.

	Learning rate	N epochs	SIC Clim	SIC				MLD			
			15x365	30	60	90	365	30	60	90	365
Climatology			X								
Linear	0.01	5	X	X			X				
UNetS	0.001	5	X	X	X	X	X	X	X	X	X
UNetL	0.001	6	X	X	X	X	X	X	X	X	X

Table 6.1: List of the models compared in this chapter with their hyperparameters

To better understand the differences in performance between these different models, their outputs are shown in Figure 6.1 for the year 2023, which is part of the test set. The predictions are made on the first day of the months of January, March, May, July, September and November to show, over the course of the year, the evolution of the SIE as well as the differences between the forecasts of the models.

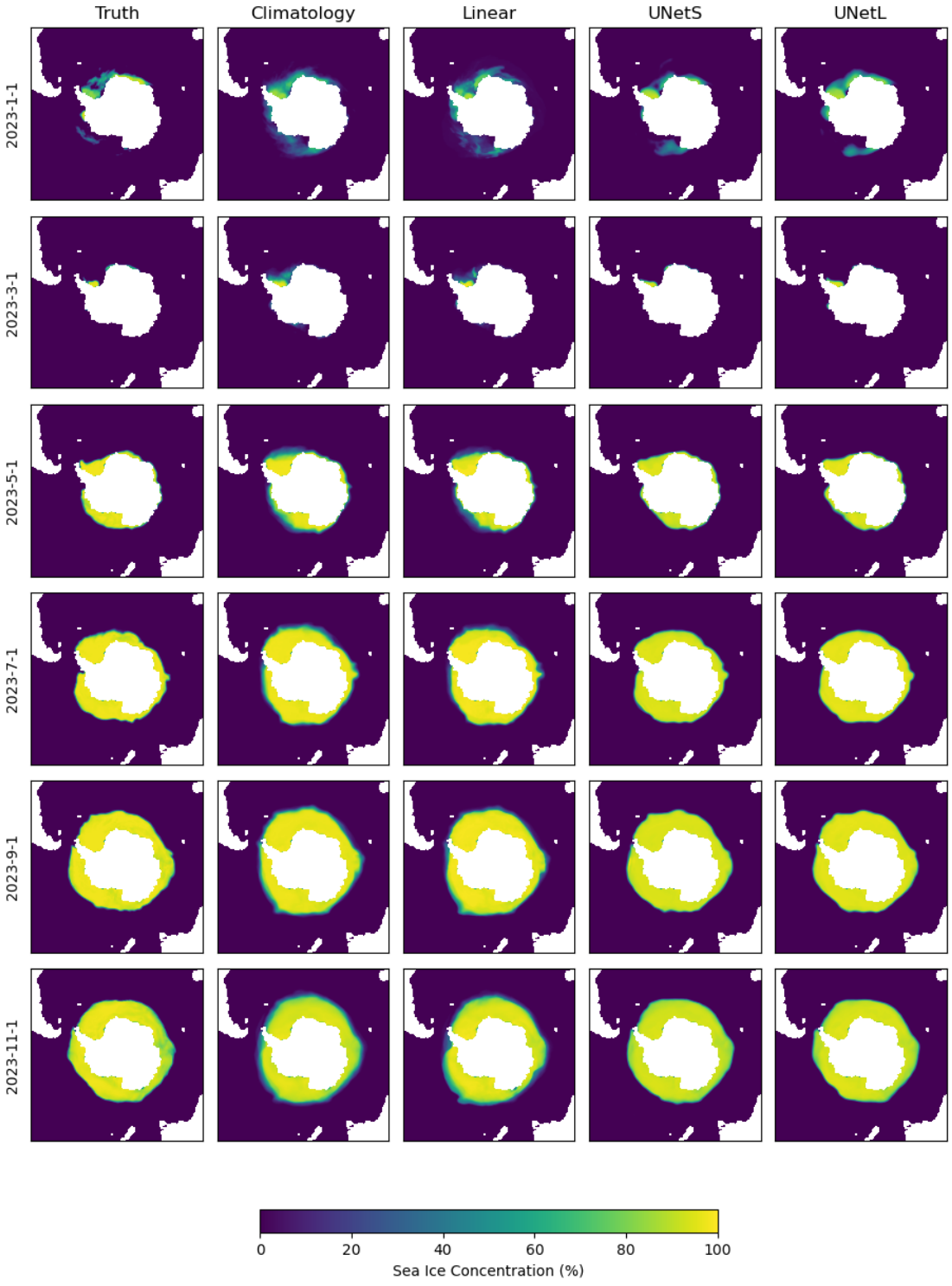

Figure 6.1: Comparison of the models outputs for the year 2023 (test set)

At first sight, the difference in forecasts between the baseline models (climatology and linear) and the U-Net models is already remarkable. On the one hand, the baselines keep the details observed on actual data, whereas the U-Net models give much smoothed forecasts. This is particularly noticeable for the month of January, which comes at the end of the ice melt period and for which the SIE is low but contains many local differences in SIC. This is less the case for the month of March, for which the SIE is very low, and also for the following months during which the ice is reforming more uniformly.

When SIE is large, as we can see it in July, September and November, the U-Nets are more faithful to the real data while being more precise than the baselines, with SIC values that are more pronounced in marginal zones, between the open ocean and the pack ice. The baselines, on the other hand, show more uncertain sea-ice boundaries and are not as accurate on the details in those marginal zones.

This suggests that the performance of the models is not constant throughout the year but that, it is easier for these models to predict the SIE when it is high in winter, than when it is low, in summer. This highlights the importance of comparing these models on a seasonal basis, given that the underlying mechanisms and patterns differ depending on the time of the year.

From a regional point of view, remarkable differences can be observed when the SIE is low, in January and March, with ice mainly present in the Weddel sector and a slight presence in the Ross sector, while the Amundsen-Bellingshausen, Indian and West Pacific regions are free of ice. These regional differences are still noticeable when the ice is forming, as is the case in May, but this diminishes as the SIE increases. As in the case of the seasons, this suggests that differences can be observed in the performances of the forecasts depending on the region that is being studied.

6.1 Numerical and spatial performances

In this section we begin by comparing the performance differences between the models on the training set and test set, Table 6.2 contains the results. This allows us to assess the model's ability to generalize to new data. If the model performs well on the validation set and on the test set, it is a good indication that the data in each set follow similar patterns which make the model able to perform well on new data. On the contrary, if a model performs well on the validation set but not on the test set this indicates that the splits have been poorly chosen and follow different patterns. In the present case, this would be possible if the test years were

significantly different from the years used for training, due to exceptional climatological phenomena that would not be represented in the training set.

		Climatology	Linear	UNetS	UNetL
Validation set	MAE	1.615	1.606	**1.001**	1.009
	NSE	0.816	0.807	0.878	**0.885**
	IIEE	3953.296	3376.413	**1841.370**	1879.798
	NIIEE	0.278	0.253	**0.163**	**0.163**
Test set	MAE	1.694	1.621	1.061	**1.052**
	NSE	0.736	0.743	**0.857**	0.843
	IIEE	4082.062	3495.989	**1885.585**	1965.015
	NIIEE	0.303	0.282	**0.185**	0.186

Table 6.2: Mean of each metric for the different models over the validation set and over the test set

The results are of a similar order of magnitude between the validation set and the test set, indicating that the data from these two sets follow similar patterns and that the model generalizes well to new data. However, it appears that performance is systematically lower in the test set than in the validation set. As mentioned in the introduction of this section, this may be due to several things, the first being that the models were trained to perform on the training set, then the hyperparameters were tuned to perform on the validation set. So, if the test data does not follow exactly the same patterns as the training and validation data, this can be reflected in the results.

The other option is that it may be due to the first signs of global warming in the Antarctic, but the limited data available on the effects of this change makes it difficult to assess this. Indeed, as the effects of global warming in the Antarctic are only visible since around 2015 [16], the years it impacted represent a very small part of the training set, while all the years in the validation and test set are affected by it.

Then, the differences between the forecasts of the baselines and the U-Net models observed in Figure 6.1 can also be seen in the numerical results, with climatology and the linear model performing similarly to each other on the one hand, and the two U-Nets also showing comparable performances between themselves on the other. This difference is equally marked when looking at the error variance of each model for each metric, as shown in Figure 6.2.

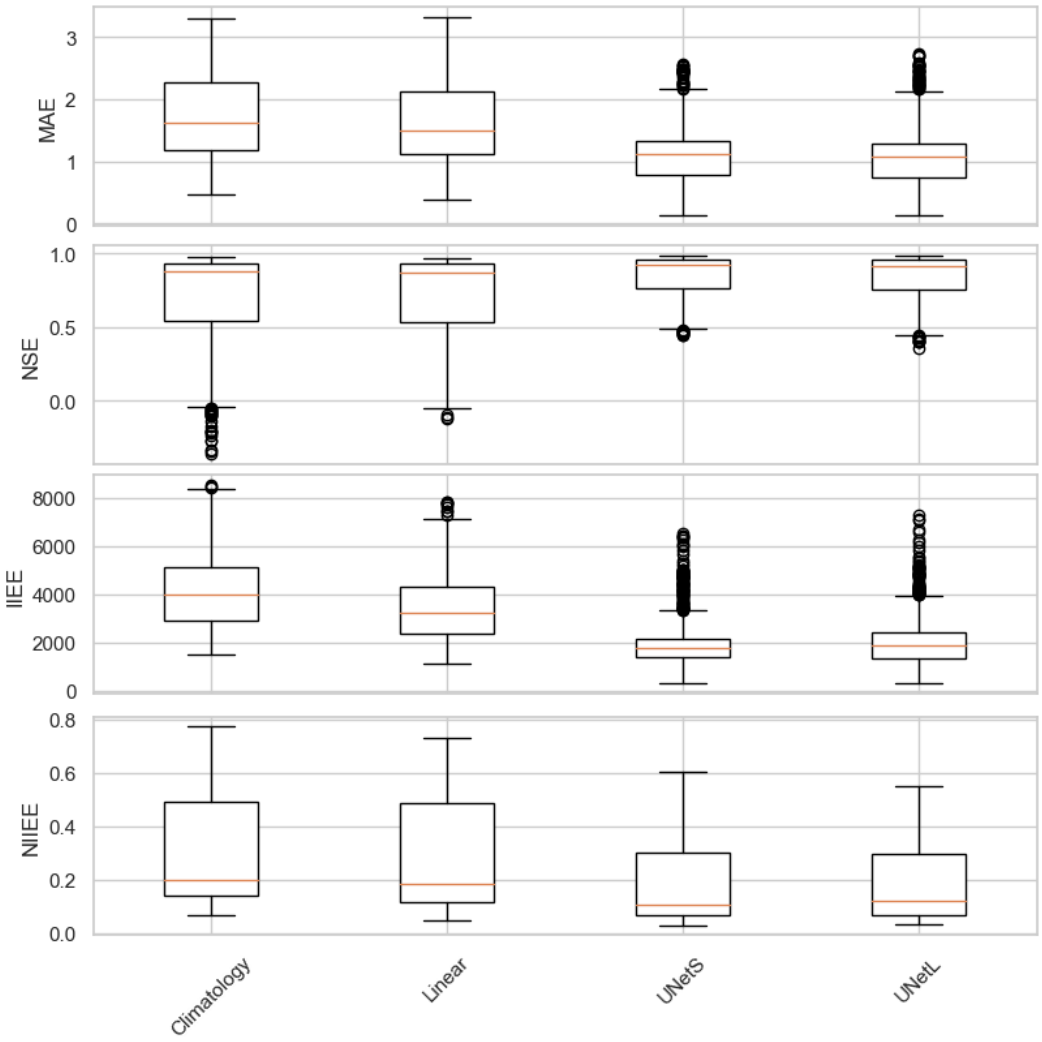

Figure 6.2: Median and variability of error of each metric for each model over the test set

6.2 Seasonal and regional performances

Performance analysis can now be taken a step further by looking at other aspects of SIC forecasting. The first step is to consider the differences in results depending on the season. Given that SIC follows a seasonal cycle of melting and freezing, the performance of predictions differs depending on the moment of the year. Figure 6.3 provides a visualisation of the results averaged by season. This allows a seasonal breakdown of the information previously shown in Figure 6.2, which only gives a global overview.

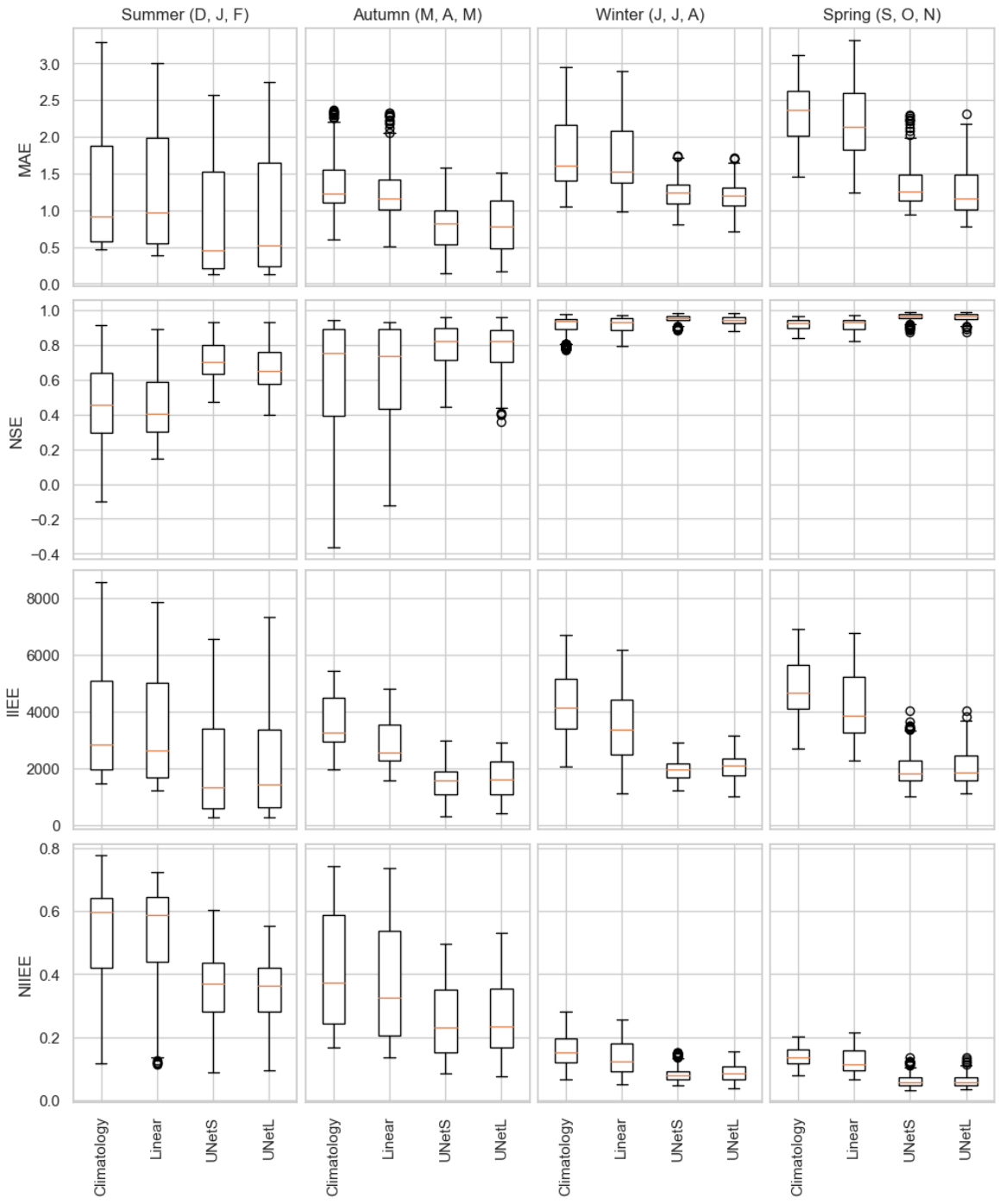

Figure 6.3: Median and variability of error of each metric for each model and for each season over the test set

By taking an interest in performance on a seasonal basis, we can make certain observations that were previously not possible using the average over an entire split. Overall, the difference between the performances of U-Nets and baselines is also noticeable here, the former outperforming the latter in all aspects, but not to the same extent for all configurations of metrics and seasons. Once again, of the two U-Nets, it is the small that performs slightly better than the large for almost all the metrics and seasons, by offering the best mean error but also with a lower error variability in most cases.

Regarding the metrics, we have already divided those indicating numerical performance with the MAE and NSE and those indicating spatial performance with the IIEE and NIIEE, but we also need to distinguish between those that are normalised and those that are not, i.e. those who do take the SIE into account and those that do not. Indeed, the MAE and IIEE are raw metrics whose calculation is not influenced by a smaller or larger actual SIE, whereas the NSE and NIIEE take these seasonal variations into account. In the case of the NSE, it compares the mean squared error of the model's predictions to the variance of the observed data relative to the performance of a simple baseline model that always predicts the mean of the observed observations whereas for the NIIEE, it is the IIEE divided by the extent of the union between observation and forecasting. This normalisation of the metrics can be seen in the results, since the MAE and IIEE follow similar patterns over the seasons, as do the NSE and NIIEE.

Therefore, on the performance of models overall, we note that without normalisation, the model's mean errors are smallest in summer and autumn, when the SIE is low, and increase in winter and spring, when SIE is large. However, when the normalised metrics are considered, the situation is reversed and the models perform best in winter and spring, when the SIE is high and the errors have less weight compared with larger areas of ice. In fact, when there is little ice, the models necessarily make small errors, but these errors are considerable in relation to the small amount of ice actually present. Conversely, when the SIE is large, the models make larger errors, but these are small in relation to the total SIE.

Then, the models show greater variability in errors during summer and autumn, once again when the SIE is small. On the other hand, in winter and spring, the error variability does not increase alongside with SIE. We notice that for the non normalised metrics, the variability remains of the same order of magnitude in autumn, winter and spring, but once considering normalized metrics, this variability is minimal for winter and spring while increasing in autumn.

All this indicates that when the SIE is low (summer and autumn) the errors are on average smaller than when the SIE is high (winter and spring) but these smaller errors are proportionally greater compared to the SIE.

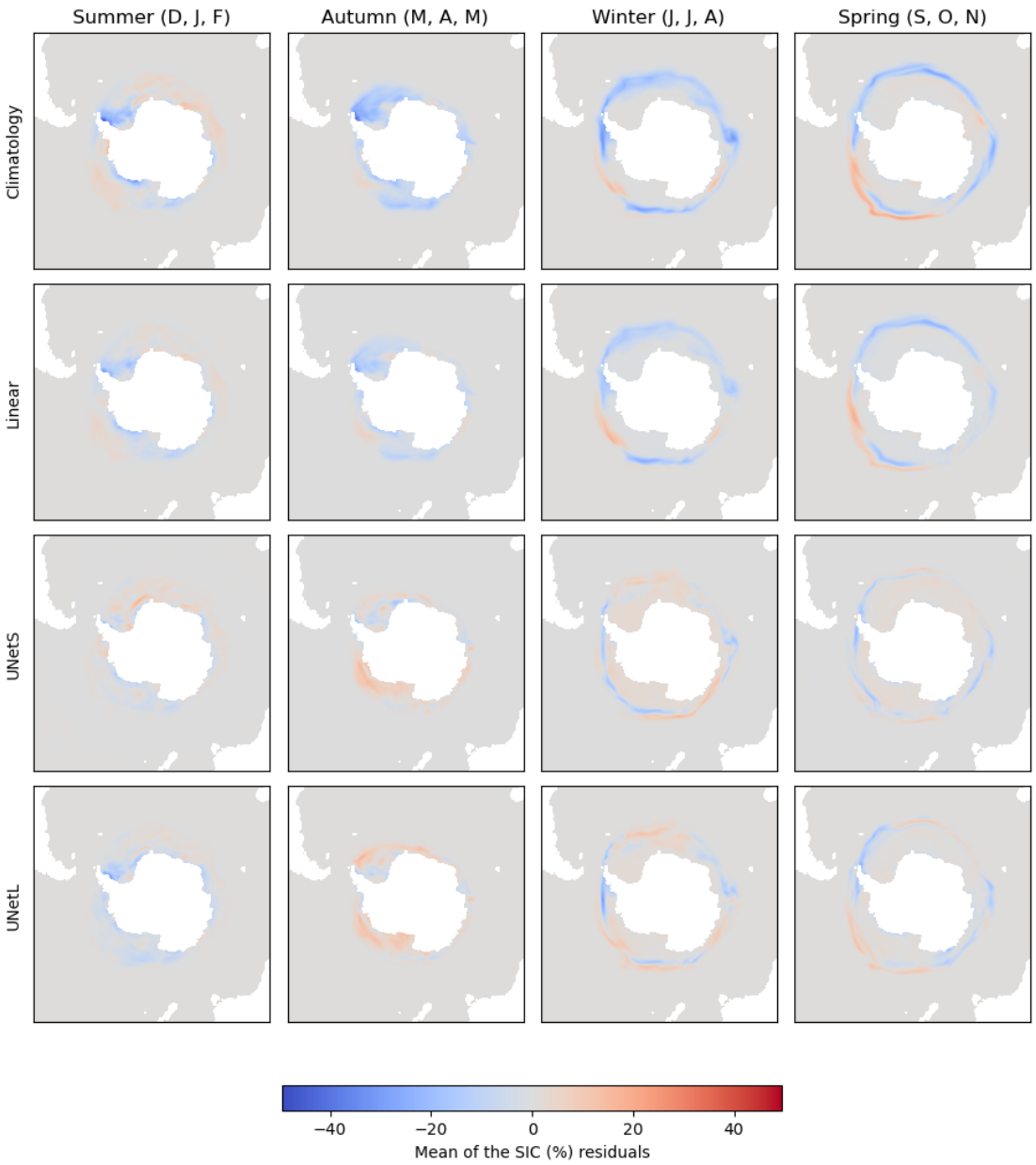

Figure 6.4: Mean SIC percentage residuals (truth - prediction) of each model for each season over the test set

For a better understanding of the geographical distribution of these errors, Figure 6.4 shows the mean of the residuals for each model and for each season. Here again, a number of trends emerge.

For baselines primarily, throughout the year, predictions give SIC values that are larger than they actually are, as indicated by the negative values present throughout the seasons. As baselines are based solely on previous data, this might be due to the recent impact of global warming and accelerated ice melt in Antarctica. Spatially, the trends are different when the SIE is low and when it is high. In summer and autumn, because the SIE is low, the errors are just as prevalent in the marginal zones as in the central zones, although there are regional differences. In winter and spring, on the other hand, errors occur mainly in marginal areas, with central areas consistently having a high SIC, which is straightforward to predict. Regionally, then, the differences are only noticeable for low SIE, with the Weddel and Ross sectors showing the greatest error, notably because during these periods the ice is mainly present in these regions.

For U-Nets, the errors are again smaller than for baselines. Here, unlike baselines, the models tend to provide SIC values that are smaller than they really are, with mainly positive residuals. The differences in residual patterns for low and high SIE observed for the baselines can also be seen here, with the errors being mainly central in summer and autumn and marginal in winter and spring. From a regional point of view, the differences in errors are once again more pronounced in the Weddel and Ross sectors for low SIE, since that is where the ice is located during these periods.

6.3 Inter-annual performances

We have seen that the performance of the models varies according to the season, which means that the U-Net models, although offering better performance than the baselines, do not predict the SIC and SIE with consistent accuracy over the year. If this consistency in intra-annual performance is difficult to achieve, it remains to be seen whether these models have better inter-annual consistency. Figure 6.5 shows the mean error for each year for each metric and for each model.

We observe that for all the metrics, the errors in the baselines vary greatly from one year to the other. This is potentially due to climatological events having an impact on Antarctic sea ice. On the other hand, the U-Nets models, in addition to providing better performance than the baselines, show much more consistent performance over time.

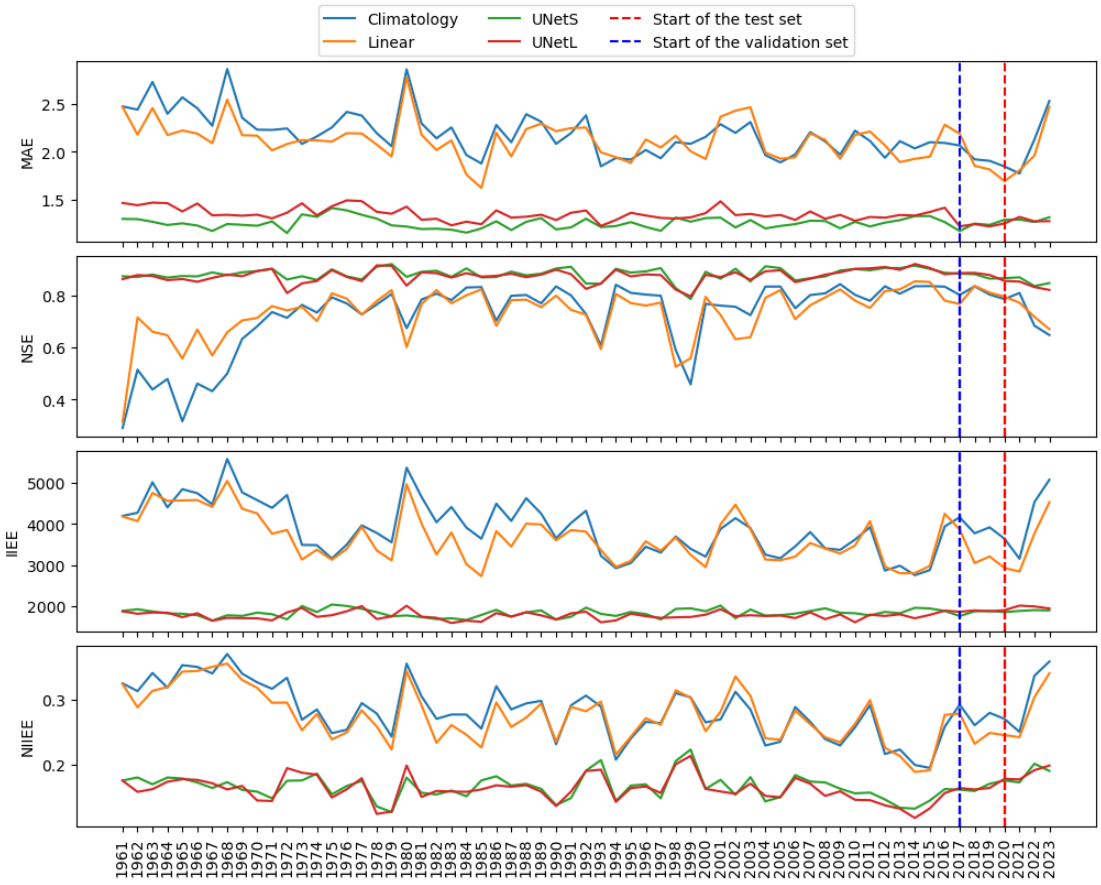

Figure 6.5: Yearly mean for each metric for each model over the whole dataset

However, normalised metrics show greater variance than their non-normalised counterparts, and this is particularly the case for the NIIEE, which, unlike the other metrics, follows a pattern similar to that of the baselines. This can be explained by the fact that the NIIEE takes direct account of the truth SIE and therefore marks spatial errors relative to it. These variations in the NIIEE therefore indicate changes in the SIE that were not predicted by the models.

One potential cause of these variations are ENSO (El Niño-Southern Oscillation) events, a significant climate phenomena originating in the tropical Pacific Ocean, involving periodic variations in sea surface temperatures and atmospheric pressures. These events have two main phases: El Niño and La Niña. El Niño is characterized by the warming of sea surface temperatures in the central and eastern Pacific Ocean, leading to widespread climatic effects such as altered weather

patterns, increased rainfall in some regions, and droughts in others. La Niña, on the other hand, involves the cooling of sea surface temperatures in the same regions, typically resulting in opposite weather anomalies [24].

The peaks in NIIEE observed in 1972 and 1973 are probably due to the ENSO event that occurred at the same time. The same applies to 1976, 1977, 1992, 1993 and 2006. However, not all ENSO events, whether El Niño or La Niña, are necessarily reflected in the NIIEE, with 1987 and 1988 experiencing a major El Niño event, while the NIIEE seemed stable at the same time. Conversely, not all spikes can be explained by an ENSO event, as in the case of 1980, a year that was not marked by any ENSO event, or the particularly low peak observed in 2014, which occurred after an El Niño event [18].

To carry the inter-annual analysis further, Figure 6.6 shows the average NIIEE for the UNetS model as a function of each month over the entire dataset, from January 1961 to November 2023. This provides a better idea of the impact of each month on the error, and therefore of which months are responsible for most of the annual errors. As previously observed in Figure 6.3, winter and spring have consistently low NIIEE, and it is mainly the months of December, January, February and March, during which the SIE is low, that are responsible for most of the NIIEE, but it is also during these months that the peaks observed in Figure 6.5 can be seen, whereas the months during which the SIE is high have a constant NIIEE over the years, whether they are marked by an annual peak or not.

6.4 Comparison with SIPNet

Some of the previously discussed results can be compared with those gathered by Wang et al. (2023) [26] with their SIPNet model, which is described in section 2.4. As a reminder, they only use MAE, IIEE and ACC to compute the performances of their model meaning that only the first two can be used for comparison. Foremost, from a seasonal point of view, both studies show the best MAE and IIEE forecasting skill in autumn and the worst in spring. To be more precise, our results show a better median performance in summer but accompanied by the greatest variability, while in autumn the median error is higher, but the variability is at its lowest. Regionally, we also find that the greatest error comes from the marginal ice zones, but the SIPNet shows the smallest error in the Weddel sector, while we have concluded that our models show the greatest error in this same zone. Finally, they also noted that their network is able to capture the variability caused by ENSO phenomena.

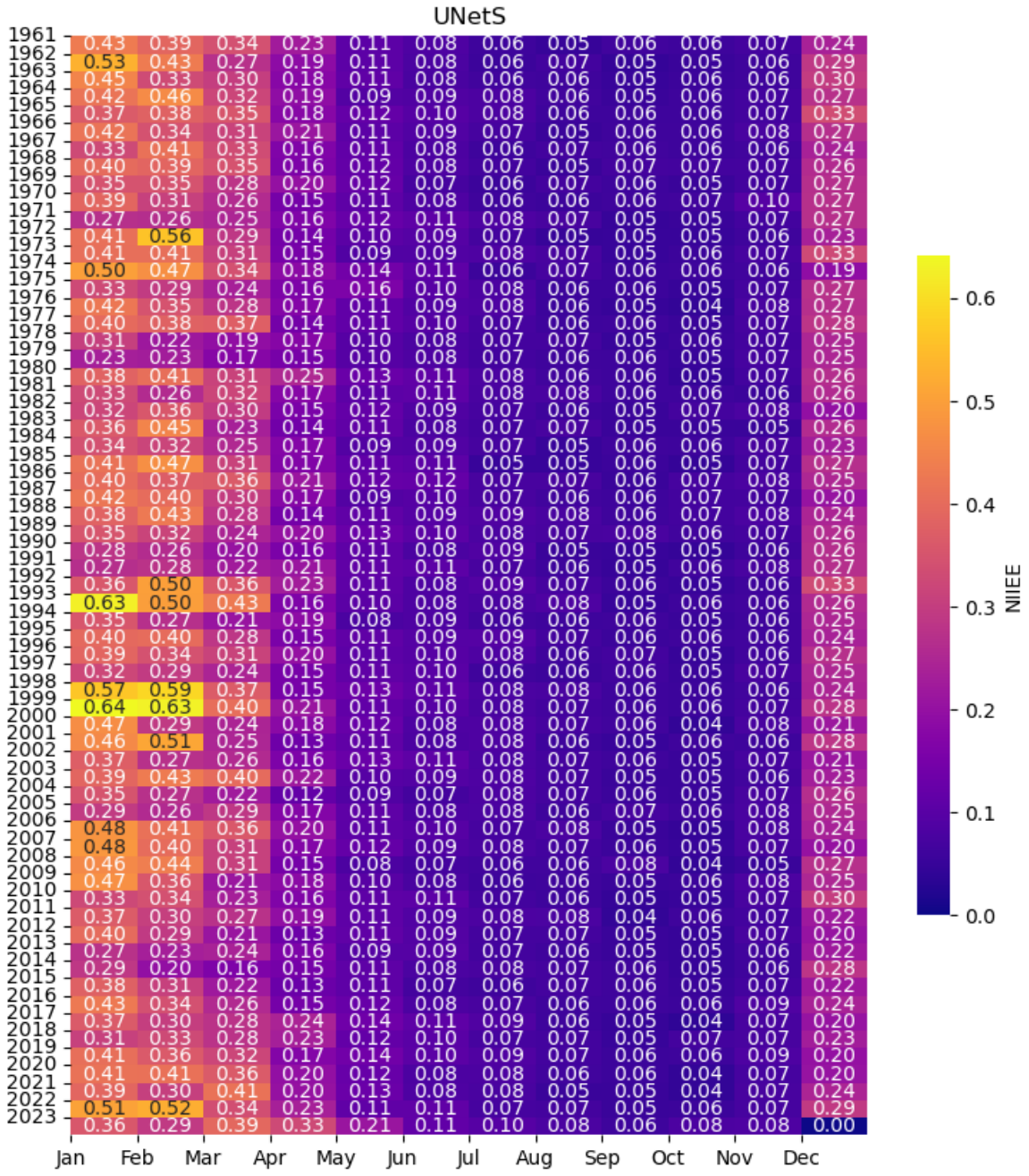

Figure 6.6: Mean NIIEE of the UNetS model for each month and for each year over the whole dataset

Chapter 7

Conclusions and perspectives for future works

The aim of this thesis is to evaluate the ability of the U-Net architecture to reproduce the outputs of the dynamical model NEMO-SI^3 for predicting sea ice concentration in Antarctica and to compare its performances with the climatological forecast and a linear model. Antarctica was chosen over the Arctic for two main reasons, the first being that many studies are already focusing on the North Pole for economic, diplomatic and proximity reasons. The second reason is that the Antarctic, which seemed to be unaffected by climate change, has been showing signs of accelerating ice melt since around 2015, and this was particularly the case in 2023, which saw the lowest peak in SIE ever recorded.

After a literature review introducing the concept, metrics and architecture used in this study, we describe the dataset used as well as the framing of the problem and the methods applied to obtain the model predictions. We then describe the U-Nets architectures used with a version similar to that used by Andersson et al. in Arctic [1] but adapted to the problem framing which is considered as UNetLarge and a reduced version with fewer encoder-decoder blocks, layers and parameters referred to as UNetSmall. After fine-tuning the hyperparameters of these models, they show better performances than the baselines, both numerically, with more accurate SIC values, and spatially, with predicted SIE closer to the truth values. The U-Nets models also produced better results, both intra-annually, with better resistance to seasonal variations, and inter-annually, with more consistent performance than the baselines over the years. From a regional point of view, for low SIE, i.e. in summer and autumn, it is the Weddel and Ross regions that show the most error, mainly because ice is predominantly present in these regions and not the others during these seasons. For large SIE values, the error appears mainly in the marginal zones, for all regions. Of the two U-Nets considered, the UNet-

Large model shows no improvement over the UNetSmall version, the latter even performing slightly better in many configurations.

The objectives described in the introduction have therefore been achieved. Nonetheless, certain aspects can be investigated in greater depth and could be the subject of potential future works. Some are discussed here, and the applications of deep learning to the prediction of sea ice are even broader, offering extensive opportunities for further exploration. When discussing the results, as the study focused mainly on temporal performances, it is possible to examine further the regional performances of the U-Net model by comparing numerically each sector's performances. Then, instead of predicting for a specific period of time, a month (30 days) in this instance, it is possible to forecast the SIC over several periods of time, either in terms of weeks, as Wang et al. [26], or over several months, as Andersson et al. [1]. Also, in this study, we were interested in the impact of using two additional input variables, snow thickness and MLD, but many others are conceivable and also produced by the NEMO-SI^3 model. Thus, a more comprehensive study of the importance of these input variables is relevant and has great potential for improving model performances, as it was the case with the integration of MLD in the training set.

As explained in chapter 2, other model architectures exist and show promising results. It is therefore relevant to compare extensively the performances of these different architectures within the same study, based on the same dataset, with an identical problem framing and using the same metrics. Furthermore, a number of ready-to-use time series forecasting models have recently been published by leading companies in the field, such as Google's TSMixer [2] or Nixtla's TimeGPT [10], which may also be relevant for comparison.

Lastly, in the fields of time series forecasting and climatology, model uncertainty is a major point that is particularly lacking in statistical models, as opposed to dynamical models. Indeed, dynamical models can provide a probability of forecast and evaluate their uncertainty by considering slightly different initial conditions through ensemble forecasting, which produces a range of possible outcomes and estimates the probability of different scenarios. In contrast, statistical models typically generate a single forecast based on historical data and patterns, lacking the ability to account for small variations in initial conditions and failing to provide a measure of forecast uncertainty. This limitation reduces their utility in scenarios where understanding the range of possible outcomes is crucial, however, studies [6][9] have shown that it is possible for statistical models to incorporate measures of uncertainty.

Bibliography

[1] Tom R Andersson, J Scott Hosking, María Pérez-Ortiz, Brooks Paige, Andrew Elliott, Chris Russell, Stephen Law, Daniel C Jones, Jeremy Wilkinson, Tony Phillips, et al. Seasonal arctic sea ice forecasting with probabilistic deep learning. *Nature communications*, 12(1):5124, 2021.

[2] Si-An Chen, Chun-Liang Li, Nate Yoder, Sercan O Arik, and Tomas Pfister. Tsmixer: An all-mlp architecture for time series forecasting. *arXiv preprint arXiv:2303.06053*, 2023.

[3] Junhwa Chi and Hyun-choel Kim. Prediction of arctic sea ice concentration using a fully data driven deep neural network. *Remote Sensing*, 9(12):1305, 2017.

[4] Copernicus Climate Change Service (C3S). Sea ice concentration daily gridded data from 1979 to present derived from satellite observations. Accessed on 18-Apr-2024, 2024.

[5] Xiaoran Dong, Yafei Nie, Jinfei Wang, Hao Luo, Yuchun Gao, Yun Wang, Jiping Liu, Dake Chen, and Qinghua Yang. Deep learning shows promise for seasonal prediction of antarctic sea ice in a rapid decline scenario, 2024.

[6] Leonid Erlygin, Vladimir Zholobov, Valeriia Baklanova, Evgeny Sokolovskiy, and Alexey Zaytsev. Uncertainty estimation for time series forecasting via gaussian process regression surrogates. *arXiv preprint arXiv:2302.02834*, 2023.

[7] EUMETSAT Ocean and Sea Ice Satellite Application Facility. Global sea ice concentration climate data record 1978-2020 (v3.0, 2022). Extracted on 2023-10-28 from the Copernicus Climate Change Service Climate Data Store, 2022. OSI-450-a.

[8] EUMETSAT Ocean and Sea Ice Satellite Application Facility. Osi saf sea ice index 1978-onwards, version 2.2 (2023). Data accessed 2024-04-10, 2023. OSI-420.

[9] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016.

[10] Azul Garza and Max Mergenthaler-Canseco. Timegpt-1. *arXiv preprint arXiv:2310.03589*, 2023.

[11] William Gregory, Michel Tsamados, Julienne Stroeve, and Peter Sollich. Regional september sea ice forecasting with complex networks and gaussian processes. *Weather and Forecasting*, 35(3):793–806, 2020.

[12] Lauren Hoffman, Matthew R Mazloff, Sarah T Gille, Donata Giglio, Cecilia M Bitz, Patrick Heimbach, and Kayli Matsuyoshi. Machine learning for daily forecasts of arctic sea ice motion: An attribution assessment of model predictive skill. *Artificial Intelligence for the Earth Systems*, 2(4):230004, 2023.

[13] Stephanie J Johnson, Timothy N Stockdale, Laura Ferranti, Magdalena A Balmaseda, Franco Molteni, Linus Magnusson, Steffen Tietsche, Damien Decremer, Antje Weisheimer, Gianpaolo Balsamo, et al. Seas5: the new ecmwf seasonal forecast system. *Geoscientific Model Development*, 12(3):1087–1117, 2019.

[14] Young Jun Kim, Hyun-Cheol Kim, Daehyeon Han, Sanggyun Lee, and Jungho Im. Prediction of monthly arctic sea ice concentrations using satellite and reanalysis data based on convolutional neural networks. *The Cryosphere*, 14(3):1083–1104, 2020.

[15] Quanhong Liu, Ren Zhang, Yangjun Wang, Hengqian Yan, and Mei Hong. Short-term daily prediction of sea ice concentration based on deep learning of gradient loss function. *Frontiers in Marine Science*, 8:736429, 2021.

[16] François Massonnet, Sandra Barreira, Antoine Barthelemy, Roberto Bilbao, Edward Blanchard-Wrigglesworth, Ed Blockley, David H Bromwich, Mitchell Bushuk, Xiaoran Dong, Helge F Goessling, et al. Sipn south: six years of coordinated seasonal antarctic sea ice predictions. *Frontiers in Marine Science*, 10:1148899, 2023.

[17] Yafei Nie, Chengkun Li, Martin Vancoppenolle, Bin Cheng, Fabio Boeira Dias, Xianqing Lv, and Petteri Uotila. Sensitivity of nemo4. 0-si 3 model parameters on sea ice budgets in the southern ocean. *Geoscientific Model Development*, 16(4):1395–1425, 2023.

[18] Jan Null. El niño and la niña years and intensities: Based on oceanic niño index (oni). `https://ggweather.com/enso/oni.htm`, 2024. Accessed: 2024-05-18.

[19] Yibin Ren and Xiaofeng Li. Predicting the daily sea ice concentration on a subseasonal scale of the pan-arctic during the melting season by a deep learning model. *IEEE Transactions on Geoscience and Remote Sensing*, 2023.

[20] Yibin Ren, Xiaofeng Li, and Wenhao Zhang. A data-driven deep learning model for weekly sea ice concentration prediction of the pan-arctic during the melting season. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–19, 2022.

[21] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015.

[22] Anne Marie Treguier, Clement de Boyer Montégut, Alexandra Bozec, Eric P Chassignet, Baylor Fox-Kemper, Andy McC. Hogg, Doroteaciro Iovino, Andrew E Kiss, Julien Le Sommer, Yiwen Li, et al. The mixed-layer depth in the ocean model intercomparison project (omip): impact of resolving mesoscale eddies. *Geoscientific Model Development*, 16(13):3849–3872, 2023.

[23] M. Vancoppenolle, C. Rousset, E. Blockley, Y. Aksenov, D. Feltham, T. Fichefet, G. Garric, V. Guémas, D. Iovino, S. Keeley, G. Madec, F. Massonnet, Jeff Ridley, D. Schroeder, and S. Tietsche. SI3, the NEMO sea ice engine. Publisher: Zenodo.

[24] Chunzai Wang, Clara Deser, Jin-Yi Yu, Pedro DiNezio, and Amy Clement. El niño and southern oscillation (enso): a review. *Coral reefs of the eastern tropical Pacific: Persistence and loss in a dynamic environment*, pages 85–106, 2017.

[25] Yunhe Wang, Xiaojun Yuan, Haibo Bi, Yibin Ren, Yu Liang, Cuihua Li, and Xiaofeng Li. Understanding arctic sea ice thickness predictability by a markov model. *Journal of Climate*, 36(15):4879–4897, 2023.

[26] Yunhe Wang, Xiaojun Yuan, Yibin Ren, Mitchell Bushuk, Qi Shu, Cuihua Li, and Xiaofeng Li. Subseasonal prediction of regional antarctic sea ice by a deep learning model. *Geophysical Research Letters*, 50(17):e2023GL104347, 2023.

[27] Yilin Zhu, Mengjiao Qin, Panxi Dai, Sensen Wu, Zhiyi Fu, Zhende Chen, Laifu Zhang, Yuanyuan Wang, and Zhenhong Du. Deep learning-based seasonal forecast of sea ice considering atmospheric conditions. *Journal of Geophysical Research: Atmospheres*, 128(24):e2023JD039521, 2023.

Appendix

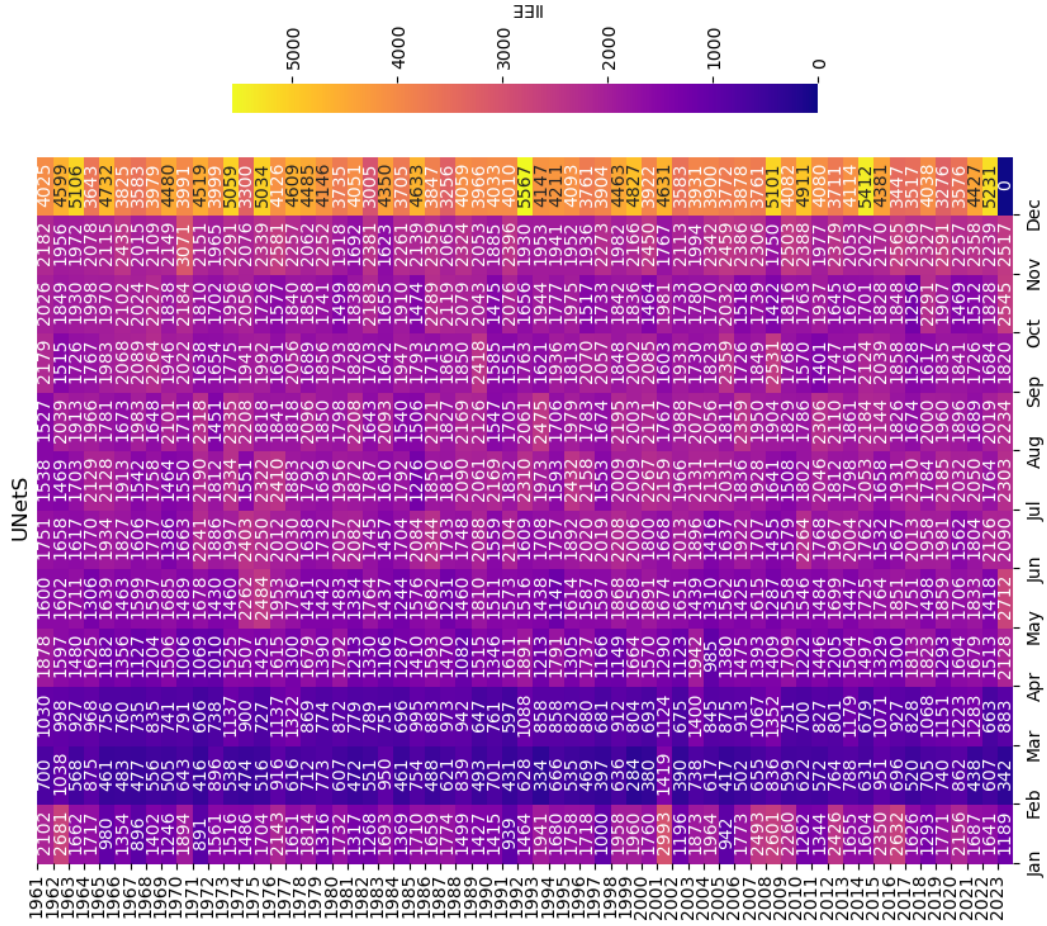

Figure 1: Mean IIEE of the UNetS model for each month and for each year over the whole dataset

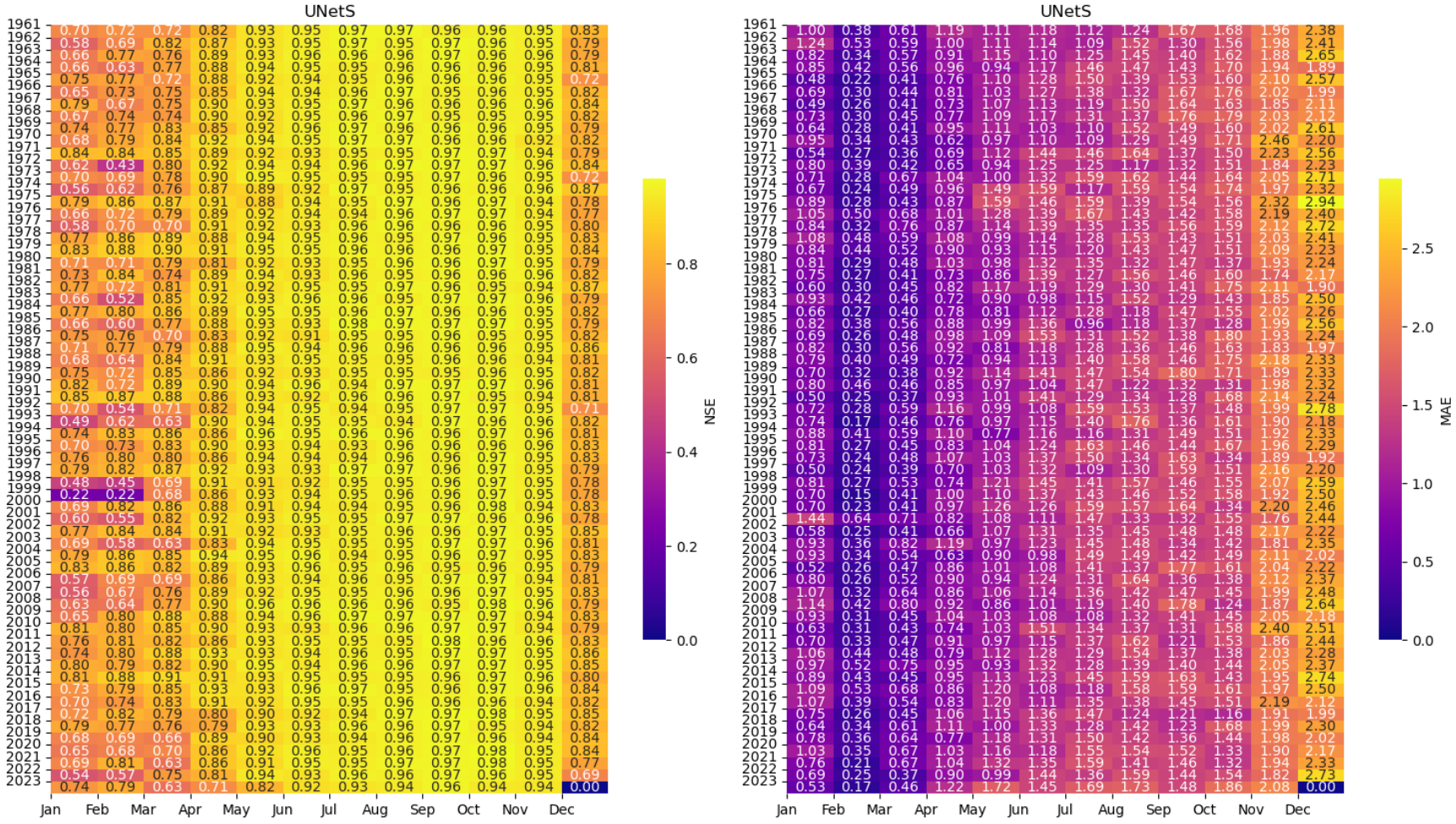

Figure 2: Mean NSE and MAE of the UNetS model for each month and for each year over the whole dataset

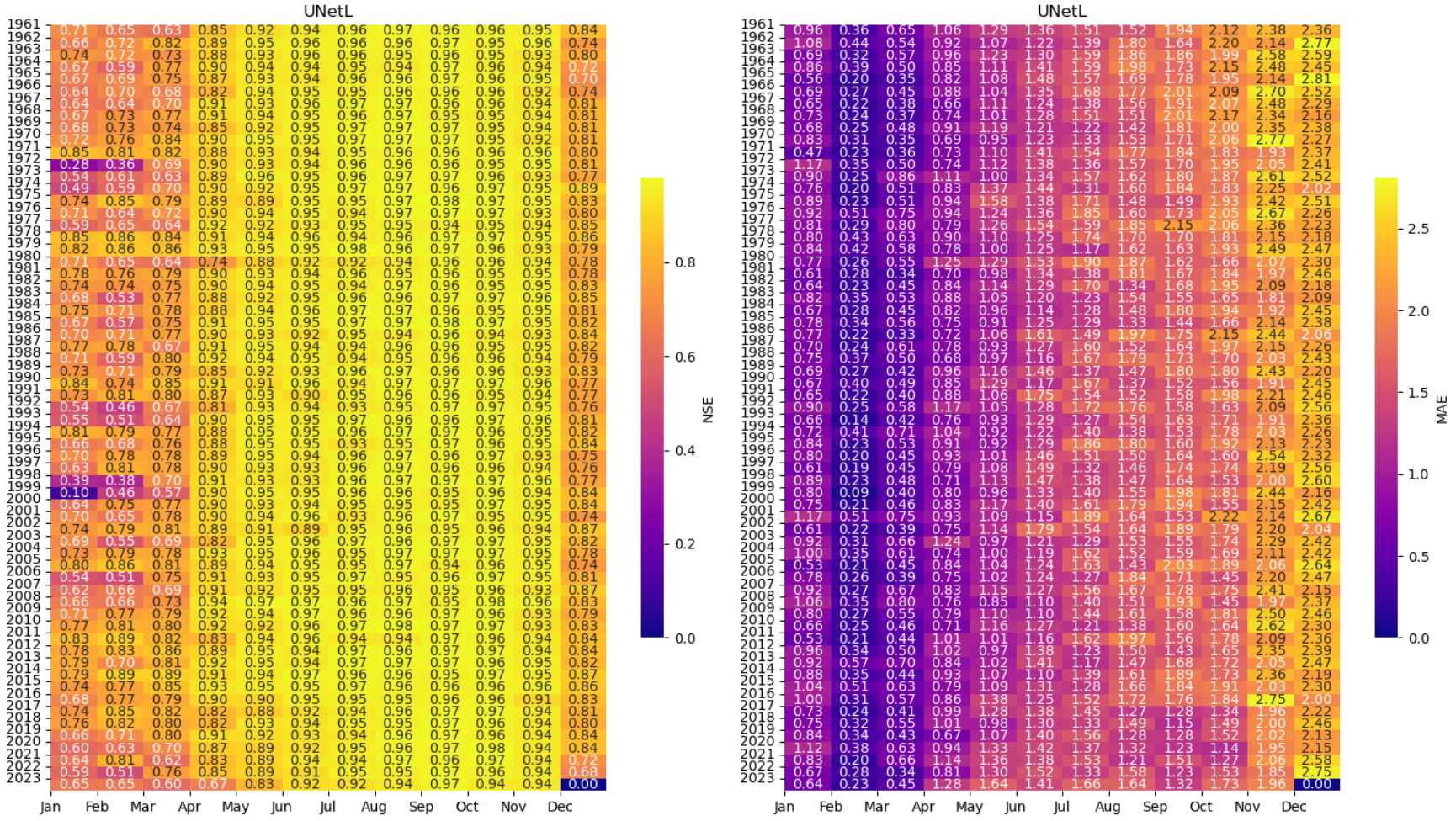

Figure 3: Mean NSE and MAE of the UNetL model for each month and for each year over the whole dataset

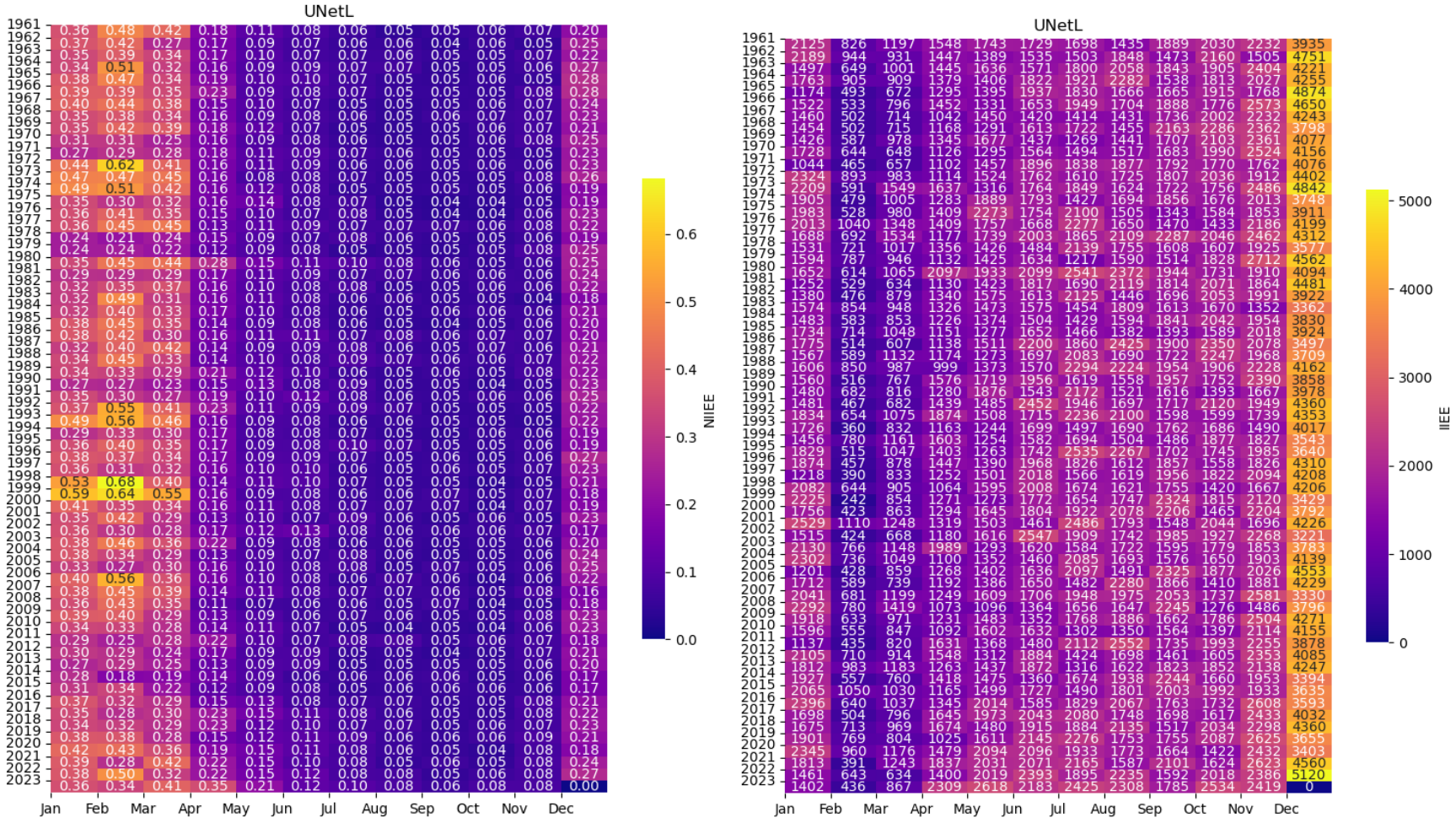

Figure 4: Mean NIIEE and IIEE of the UNetL model for each month and for each year over the whole dataset

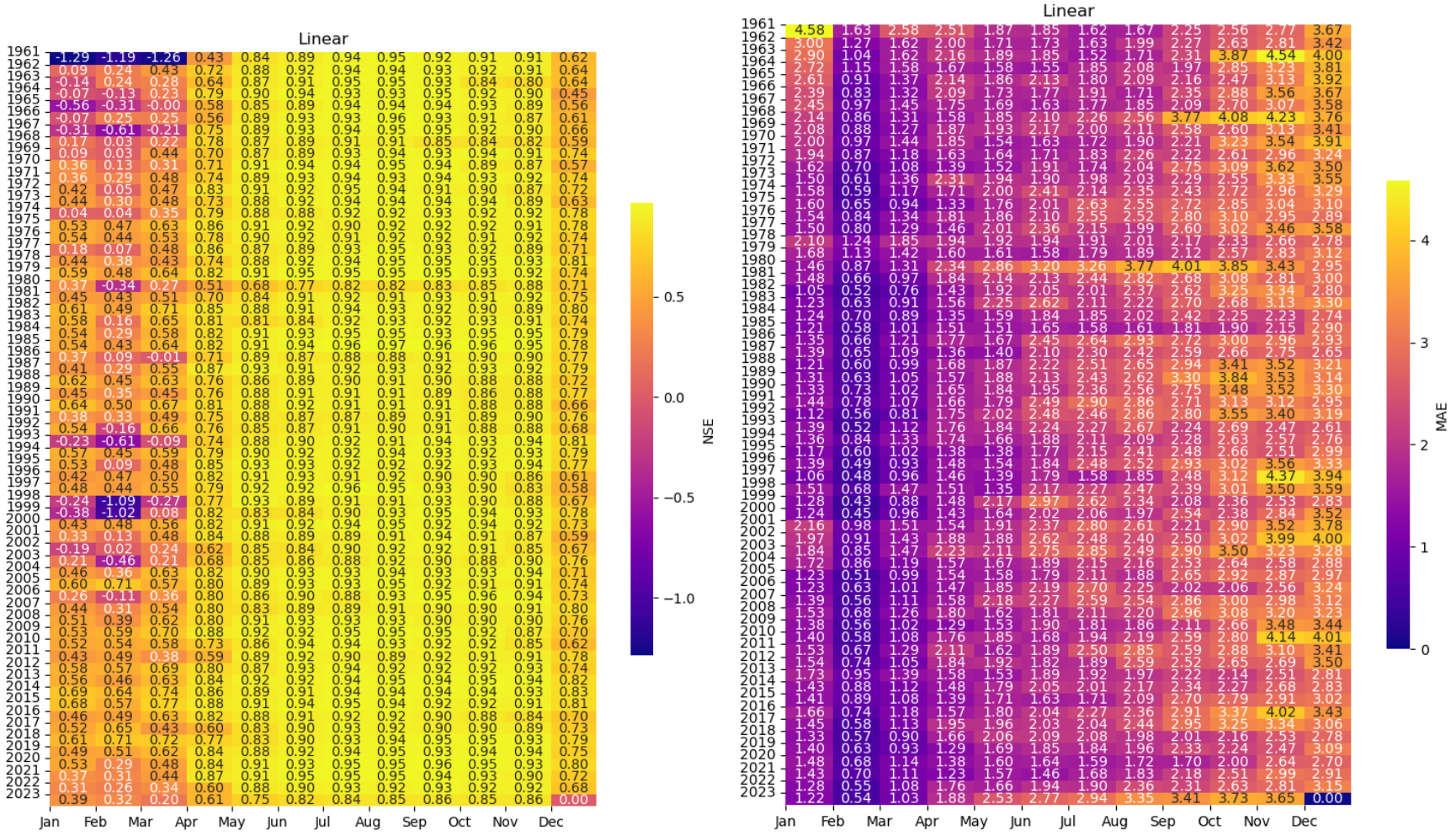

Figure 5: Mean NSE and MAE of the Linear model for each month and for each year over the whole dataset

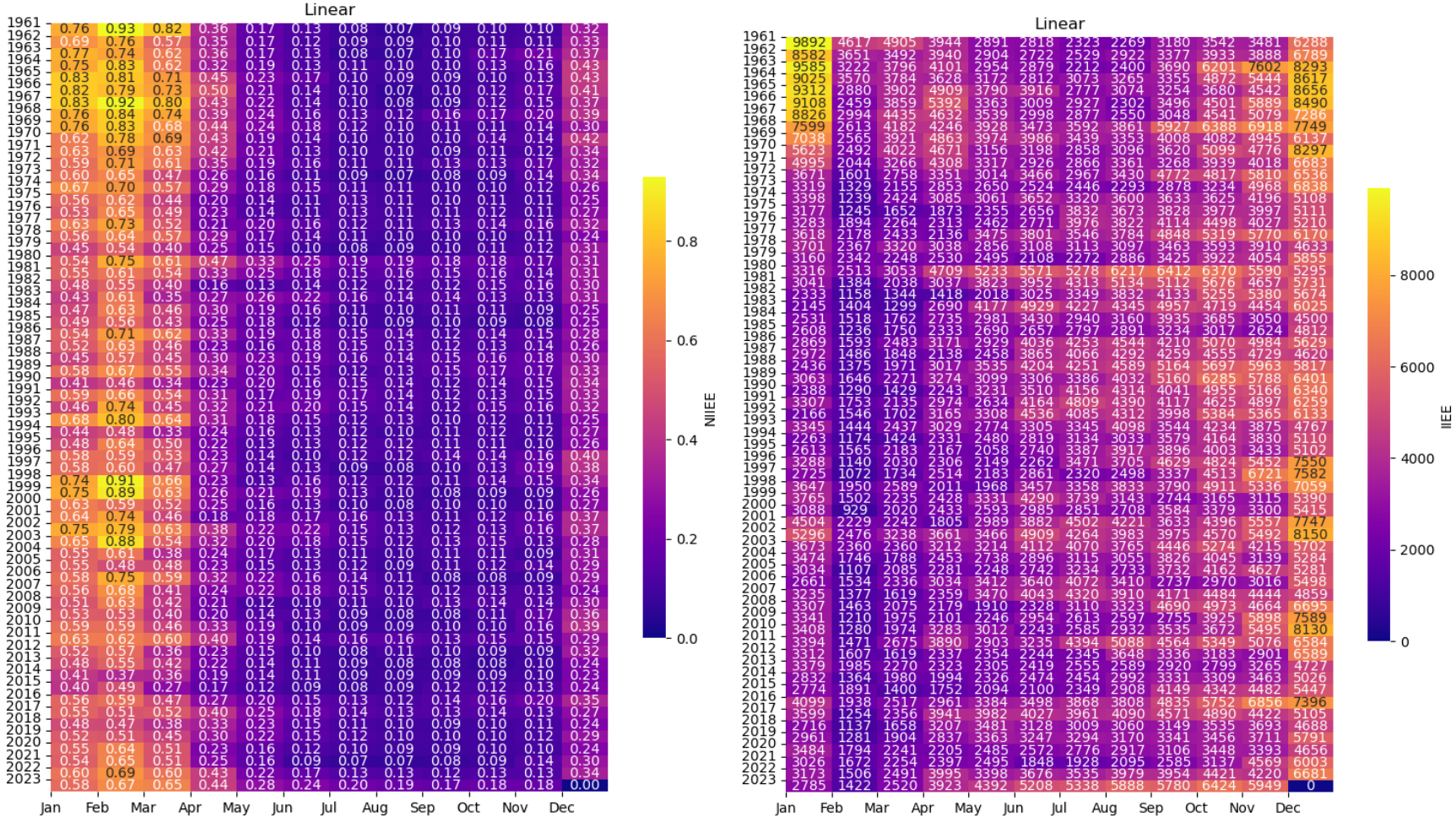

Figure 6: Mean NIIEE and IIEE of the Linear model for each month and for each year over the whole dataset

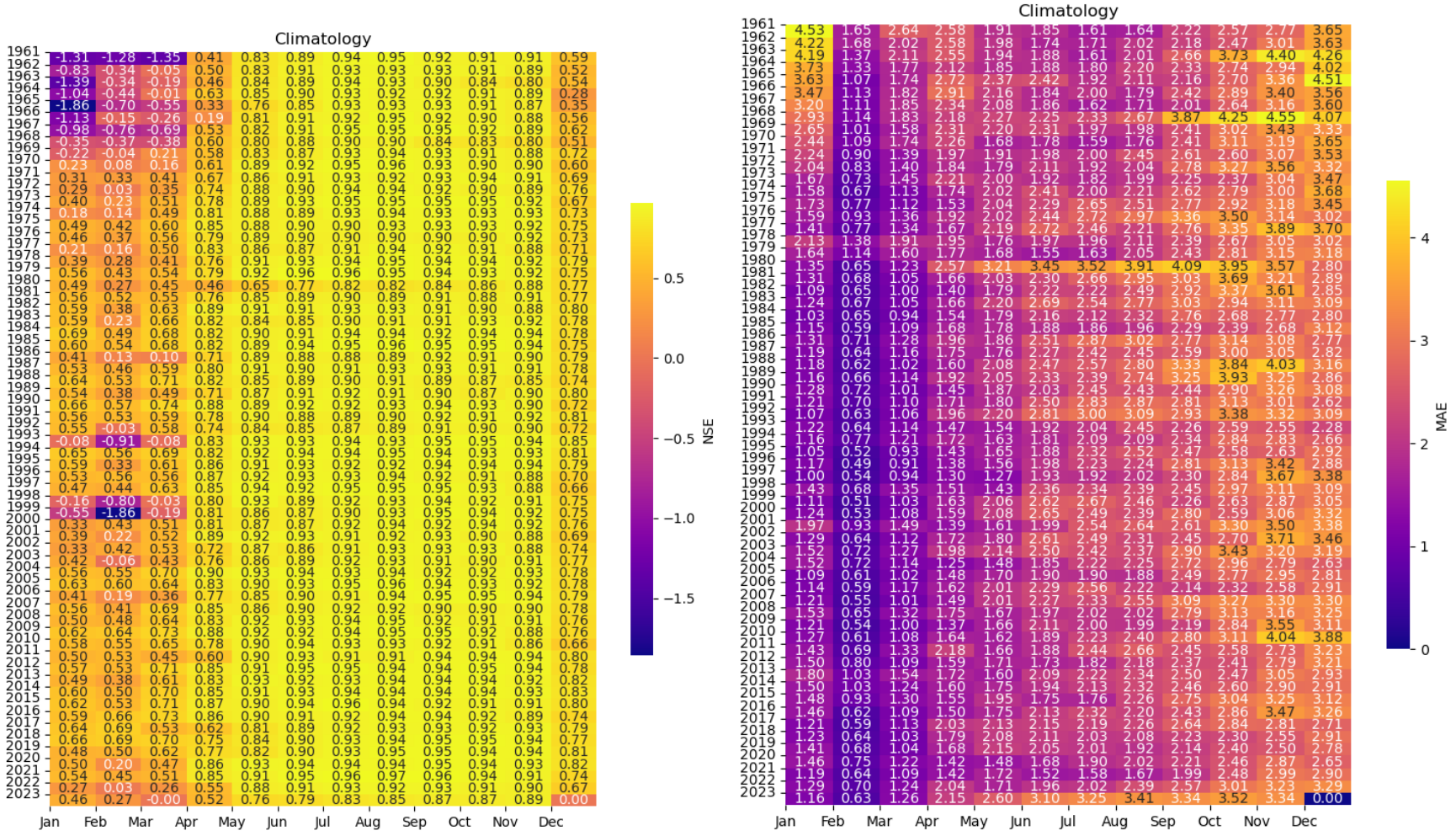

Figure 7: Mean NSE and MAE of the Climatology model for each month and for each year over the whole dataset

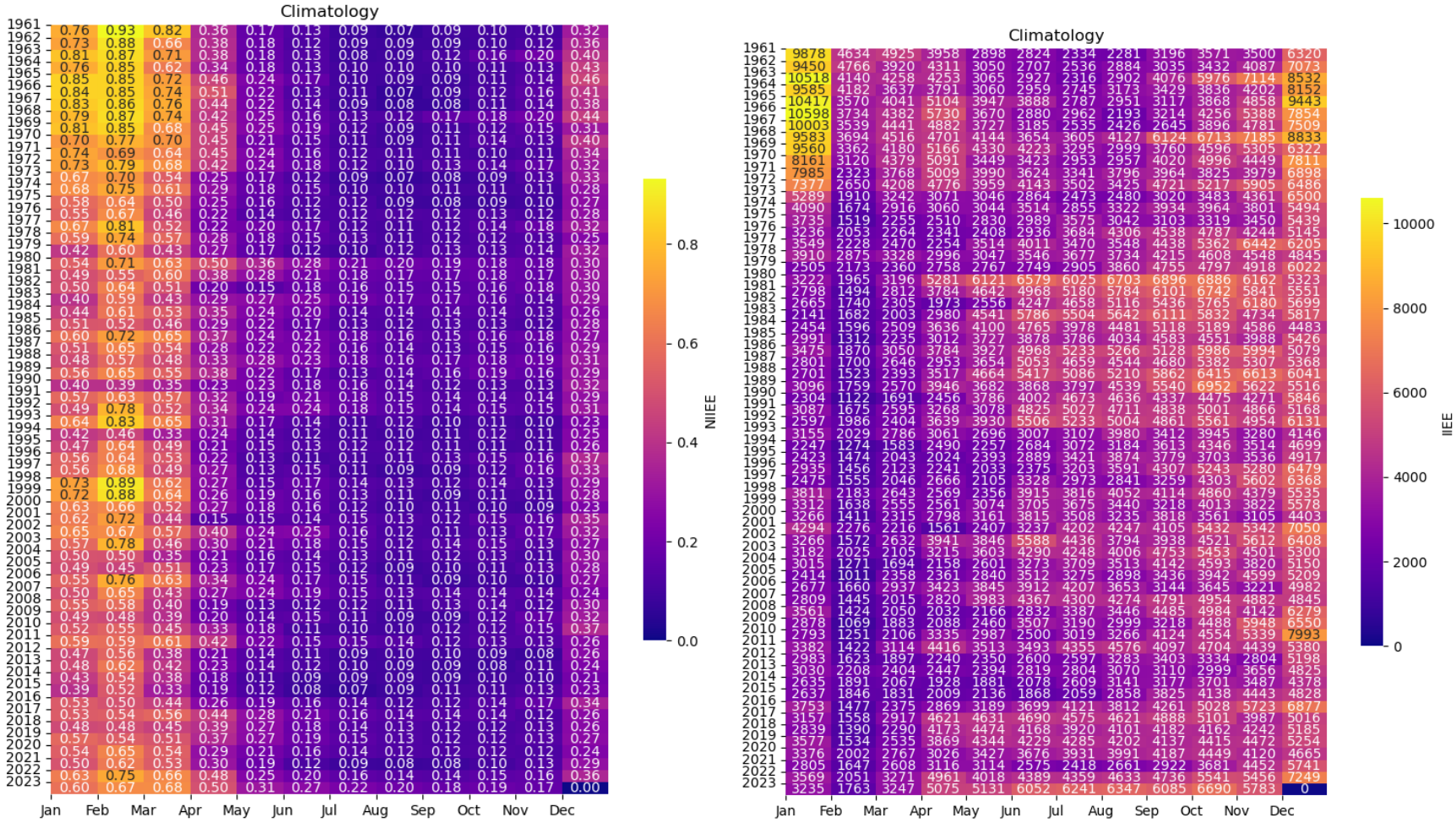

Figure 8: Mean NIIEE and IIEE of the Climatology model for each month and for each year over the whole dataset

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
École polytechnique de Louvain
Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve, Belgique | **www.uclouvain.be/epl**