Bachelor's Thesis

Combining additive regression and convolutional neural networks for classifying circulation patterns associated with droughts

Department of Statistics
Ludwig-Maximilians-Universität München

Laetitia Frost

Munich, August 2th, 2023

Submitted in partial fulfillment of the requirements for the degree of B. Sc.
Supervised by Dr. David Rügamer

Abstract

In recent years, both the frequency and severity of drought events have increased in Central Europe. Since certain circulation patterns are associated with these drought events, classifying them using modelling approaches is of vital interest to anticipate their occurrence and the effect that climate change is having on them. An already existing modelling approach uses a convolutional neural network (CNN) for the classification of these circulation patterns associated with droughts. Here, the daily images of atmospheric pressure conditions serve as unstructured predictor variables. In this thesis, this methodology is extended by adding structured covariates to these input features. To model the circulation patterns using this multimodal data semi-structured distributional regression (SSDR) is used, which combines structured additive distributional regression and CNNs via one unifying neural network. To quantify whether the inclusion of structured covariates leads to an improvement of the predictive performance compared the original approach, various specifications, differently combining the multimodal input features, are modeled. To estimate the model performance repeated-holdout splitting is used. Thanks to its rolling origin, expanding window design it takes the temporal structure of the data into account. To compare the different model specifications, the cross entropy, class-wise F1 scores, as well as the macro recall, macro precision and the accuracy across all classes are used. Confusion matrices and the resulting qualitative metrics are additionally used to analyse selected models in detail. It is shown that the inclusion of structured, predecessor-independent covariates does not lead to an overall improvement of the predictive performance. The inclusion of structured, predecessor-dependent covariates only increases the predictive performance when evaluated on a known-predecessor test set. Regarding additional unstructured covariates, only the inclusion of lead images leads to a slight improvement in the predictive performance.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Circulation patterns and drought events

In the past two decades the number of high-intensity droughts in Europe has been increasing steadily, severely impacting a variety of sectors such as society, economy, forestry, biodiversity and agriculture. For example the water and heat stress caused by the Euro-Mediterranean drought in 2022 lead to severe losses in agricultural yields, the implementation of drought emergency water restrictions and the widespread of severe wildfires all over the central and southern regions of Europe (Faranda et al. (2023)).
Besides thermodynamic factors (e.g. evaporation) dynamic factors are one of the key processes leading to those drought and heat events. These dynamic factors i.a. include certain atmospheric drivers, which themselves are part of large-scale atmospheric circulation patterns. Since these circulation patterns usually persist over multiple days in a larger region they define the overall character of both climate and weather in that temporal and spatial zone. Whilst multiple, different classification schemes for these circulation patterns exist, a very common one is the classification method introduced by Hess & Brezowsky. Here 29 types of circulation patterns exist, which are defined by both a certain sea level pressure $[hPa]$ and a certain geopotential height at 500 $hPa[m]$. Besides those atmospheric conditions, which need to be met for a classification as a specific circulation pattern, another requirement for the presence of a circulation pattern is a minimum duration of three days (Werner and Gerstengarbe (2010)). Especially for transition days between two subsequent circulation patterns, the thus assigned categories can be noisy due to the discrepancy between the continuous character of the atmospheric pressure systems used for classification and the discrete character of the classification itself.
Using this definition, the so called Hess & Brezowsky catalogue contains the daily classification of each day into one of the 29 circulation patterns since 1881 in Central Europe. Since this classification is done manually by experts, using information of both sea level pressure $[hPa]$ and geopotential height at 500 $hPa[m]$ on a respective day, the Hess & Brezowsky catalogue is often considered to be a subjective classification scheme. Additionally, it can also considered to be retrospective, as for classification the atmospheric conditions of both the previous and subsequent days are considered. This ensures the minimum-three-day-duration condition is met.
As already mentioned above, regardless of the specific classification scheme, a connection between certain circulation patterns and extreme drought and heat events exists. In the context of the Hess & Brezowsky catalogue these patterns are: *Zonal ridge across Central Europe (BM), Norwegian Sea-Iceland high, anticyclonic (HNA), North-easterly anticyclonic (NEA), Fennoscandian high, anticyclonic (HFA), Norwegian Sea-Fennoscandian high, anticyclonic (HNFA), and South easterly anticyclonic (SEA).* Due to their connection to drought events, it is vital to understand the occurrence of these patterns and the effect the anthropogenic climate change is having on them to in turn anticipate the occurrence and future evolution of drought events.
Contributing to this understanding, Mittermeier et al. (2021) used the daily classification from the Hess & Brezowsky catalogue for the time span 1900-2010 to classify the above mentioned six circulation patterns for Central Europe. For data analysis they additionally summarized the remaining 23 patterns into a residual class. Using these resulting

seven classes as values of the response variable they then trained a convolutional neural network (CNN). Analogously to the classification procedure implemented in the Hess & Brezowsky catalogue, Mittermeier et al. (2021) used daily images of both the sea level pressure $[hPa]$ and geopotential height at 500 $hPa[m]$of the period from 1900 to 2010 as atmospheric predictor variables for the circulation patterns of interest. To ensure the 3-day-minimum-duration condition, the authors replicated the retrospective character of the Hess & Brezowsky catalogue procedure by implementing a transition-smoothing step, which was applied onto the predicted labels stemming from the CNN. Using this classification procedure, they achieved an average macro F1-score of 38.4% and an average overall accuracy of 59.9%.

1.2 Research question and structure of the thesis

Besides using unstructured data such as images to classify the circulation patterns associated with droughts, one could also exploit the time series character (temporal sequence of daily observations) of the circulation pattern data at hand. In fact, several modelling approaches exist that simply use structured information stemming from this temporal character of the underlying process itself to model such a time series. Thus, the question arises, whether the classification approach introduced by Mittermeier et al. (2021) using unstructured data in the form of images can benefit from the additional inclusion of such structured data. Ergo, based on the already existing work of Mittermeier et al. (2021) the aim of this bachelor's thesis is to combine the classification approach via CNNs with additive regression, to thus enabled the joint modelling of both structured and unstructured for the classification of circulation patterns associated with droughts. In addition, is it to be quantified, whether the inclusion of structured data leads to an improvement in the predictive performance compared to the previous approach of using only unstructured image data.
The joint modelling is to be realized using the concept of semi-structured distributional regression, first introduced by Rügamer et al. (2023). Based on this concept, several models with different model specifications (using different subsets of the multimodal data) are to be defined and estimated. In addition, they are to be evaluated and compared regarding their predictive performance. In order to guarantee comparability, the same procedure for both estimation and performance evaluation is to be established for all models. Ideally, this bachelor thesis thus provides insight about potential benefits of combining structured data via additive regression and unstructured data via CNNs to model classification patterns associated with droughts.
Therefore, the second chapter provides some further insight into the data used in the context of this thesis. A thorough definition of the circulation patterns as time series is given and its (temporal) characteristics are analyzed via an exploratory data analysis. Chapter 3 focuses on the additive regression approach for modelling time series using purely structured data. Thus, a theoretical background of structured additive (distributional) regression (SADR) is provided. Based on both this definition and on the analysis conducted in chapter 2, the resulting structured covariates are presented and their usage in modelling is explained. Chapter 4 then addresses CNNs and how they can be used to model unstructured data such as images. Firstly a theoretical overview over CNNs is given. The

main components of a CNN and their purpose are presented. In addition, the general optimization procedure for CNNs is explained and an overview over some of the most important hyperparameters is provided. Analogously to chapter 3, after this theoretical overview, the specific CNN configuration is explained and the unstructured covariates are presented. Chapter 5 then focuses on the combination of the previously presented modelling approaches using the concept of SSDR. The underlying theoretical framework of SSDR is elaborated. Then, using the already specified covariates and configurations from the previous chapters, the final model specifications, that are to be estimated and compared in this thesis are presented. The sixth chapter focuses on some general methodical concepts, which are relevant to establish a reproducible and reliable training and performance estimation procedure. Firstly, based on the findings from chapter 2, the taken measures to account for the imbalanced distribution of the data are described. The second sub-chapter focuses on the different concepts that are are necessary to obtain a reliable estimate for the predictive performance of an individual model specifications . In the first section, different resampling strategies for time series data and their applicability in certain scenarios are explained. Then, repeated hold-out splitting, with a rolling origin and an expanding window design, as the chosen strategy for all the resampling steps conducted in this theses is presented and justified. In the second section, the concept of hyperparameter tuning is described and a reasoning for grid search as the selected search strategy is supplied. Based on these two previous sections, nested resampling is then explained and the specific implementation, using the already established implementations from the previous sections, is presented. In the last sub-chapter, the performance metrics calculated during the procedure of nested resampling are defined and the specific reasons for using these metrics are given. Chapter 7 then contains the results of the modelling procedure. Therefore, in the first sub-chapter an overview over the estimated performance metrics per model is given and possible reasons for these results are discussed. Then, for some models, further insight into their performance in given by providing class specific performance metrics. The last chapter contains a summary of the achieved results. In addition, some limitations of the chosen modelling procedure are discussed and based on these limitations an outlook w.r.t. future work is given.

2 Circulation patterns as response variable

2.1 Definition of circulation patterns as nominal time series

A data generating process (Y_t) with $t = 1, .., T$ serving as time/observation index and T representing the number of observations is considered a nominal process if the values of its state space $S = \{1, 2, ..C\}$ are unordered, finite and discrete. The time series $(y_t)_{t=1,..T}$ as specific realization of (Y_t) is then defined as a nominal time series (Weiß (2018), Fokianos and Truquet (2019)) with C possible classes per observation y_t. An observation t can be expressed using $y_t = (y_{t1}, ..., y_{tC})$

$$y_{tc} = \begin{cases} 1 & \text{if class c is observed at t,} \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

for $c = 1, ..., C$. The vector $\pi_t = (\pi_{t1}, ..., \pi_{tC})$ then denotes the conditional probabilities of y_t belonging to the respective classes c given the explanatory variables v_t, i.e. $\pi_{tc} = E(y_{tc}|v_t) = P(y_{tc} = 1|v_t)$. Based on this, one can assume individual y_t to follow a multinoulli distribution conditional on the given explanatory variables v_t

$$y_t|v_t \sim M(1, \pi_t) \tag{2}$$

where the distribution is completely parameterized by π_t. These π_t and thereby the conditional distribution of y_t are then to be estimated during the modelling process of a nominal time series Fahrmeir et al. (2013a).
As already stated in 1.2, the circulation patterns associated with droughts, namely BM, HNA, NEA, HFA, $HNFA$, SEA are of interest in this thesis. Analogously to Mittermeier et al. (2021), the remaining 23 classes are summarized into one residual class *other*. This procedure results in the discrete, finite and unordered state space $S = \{BM, HNA, NEA, HFA, HNFA, SEA, other\}$ of the time series of the circulation patterns. Thus, the time series of the circulation patterns is defined as a nominal one via $(y_t)_{t=1,...,T}$, with T = 40541 resulting from the 40541 observations of the time series. A single observation t is then represented via $y_t = (y_{tBM}, ..., y_{tother})$ with the corresponding conditional probabilities $\pi_t = (\pi_{tBM}, ..., \pi_{tother})$ forming the parameters of $y_t \sim M(1, \pi_t)$.

2.2 Exploratory analysis of circulation patterns

As Fokianos and Kedem (2003) state, nominal time series can consist of the same temporal components as their metric counterparts. For example, an inherent trend component can exist in the underlying, data generating process. In the context of metric time series, a trend simply refers to a long-term change in the mean of the time series. For nominal time series, a change in the frequency distribution of categories over time can indicate the presence of a trend component. In addition to an inherent trend, the pattern of the time series might be influenced by a seasonality component, defined by regular, repetitive, up-and-down fluctuations in the distribution of the time series. Thus, the presence of a trend or a seasonality component leads to the properties of a time series (e.g. mean, variance

etc.) to change over time, which is typically referred to as non-stationarity. Consequently, in an non-stationary time series a value y_t inherently depends on the time t at which it is observed. A component that does not affect the stationarity of a time series is the so called cyclical component also manifesting in up-and-down fluctuations in the distribution of the time series. However, in contrast to the seasonality, a cyclical pattern appears in irregular intervals. Besides these temporal components, further variables might exhibit an effect in the underlying process. Depending on the chosen modelling approach, these variables can be included as additional covariates directly into modelling (Joseph (2022)). Having accounted for all of the above mentioned components the remaining effects are bundled in the so called irregular component. This irregular component always incorporates a residual or error term representing the unpredictable noise in any process. Furthermore, the effect of further covariates and processes, not included into the modelling procedure can be part of this component (Joseph (2022)).
In addition to these components, further characteristics of the underlying categorical process need to be considered for modelling. One of these is the phenomenon of (inherent) autocorrelation, which refers to a time series y_t exhibiting a certain degree of dependence to a lagged version y_{t-l} of itself. As the name implies, this dependence is inherently present in the data generating process and can i.e. be caused by the presence of a seasonal or a cyclical component. In addition, a low frequency of change in the values of the time series can cause autocorrelation. Since all these aspects hold valuable information w.r.t. the inherent structure of (Y_t) the presence of autocorrelation should always be checked and, if necessary, accounted for properly during modelling (Joseph (2022), Weiß (2018)).

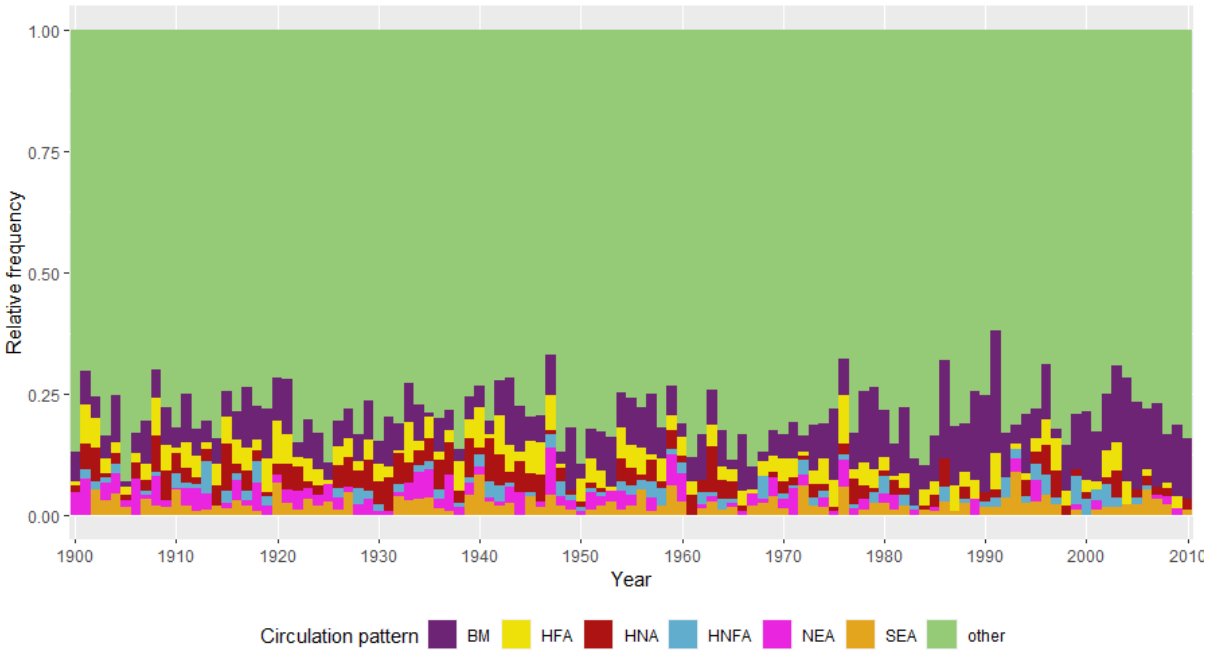

Figure 1: Relative frequency of the circulation patterns per year

In the context of metric time series multiple metrics and visualisation tools exist for analysis. For example, the respective distribution over time is often visualized via scatter plots. Whilst no exact equivalent to this procedure exists for nominal time series, the frequency of circulation pattern classes per year is plotted over time to thus obtain at least some degree of information regarding the temporal distribution of the circulation patterns (s. figure 1). It appears that the distribution has changed, i.e. the ratio of class BM seems to be increasing whilst HFA and NEA seem to be decreasing since the mid- 1970s. Another tool to detect certain changes in the distribution of a nominal time series is the rate evolution graph proposed by Ribler (1998). Here, the evolution of the cumulative sums of the binarized time series is plotted against time, resulting in an individual line per class. If the class-wise lines appear roughly linear, (weak) stationarity can be assumed for the underlying time series. Figure 2 shows the rate evolution graph for the six circulation patterns associated with droughts. Analogously to figure 1, the rate evolution graph also implies an increase in frequency for BM and a decrease of NEA roughly from 1975 onward.

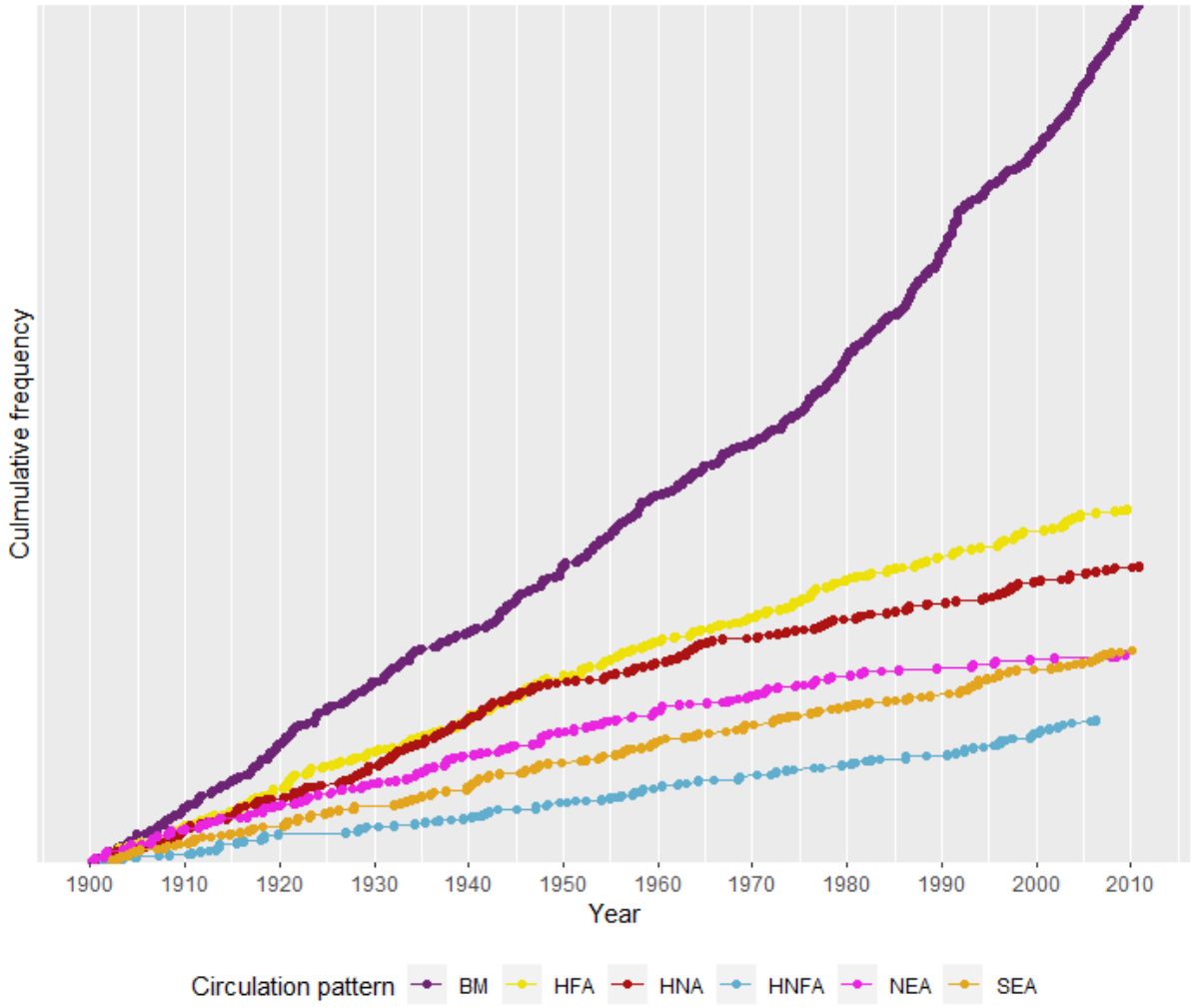

Figure 2: Rate evolution graph of the circulation patterns associated with droughts

In addition, all of these above figures indicate an imbalanced distribution among the different circulation pattern classes. This is confirmed by figure 3 visualizing the absolute frequency distribution of the circulation patterns. The residual class *other* contains approx. 32000 data points whereas the remaining 8500 are divided among the circulation pattern classes associated with droughts. Due to the procedure of combining the 23 remaining circulation patterns into the residual class *other*, this residual class also exhibits a longer mean length than the other circulation patterns (s. figure 4).

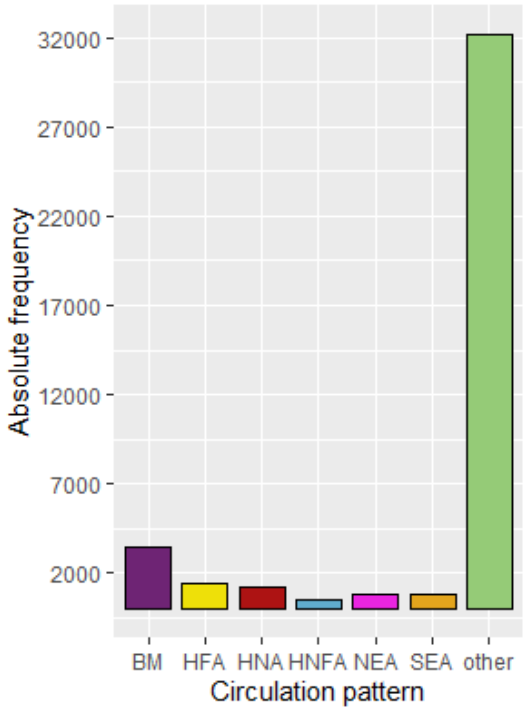

Figure 3: Absolute frequency of the circulation patterns

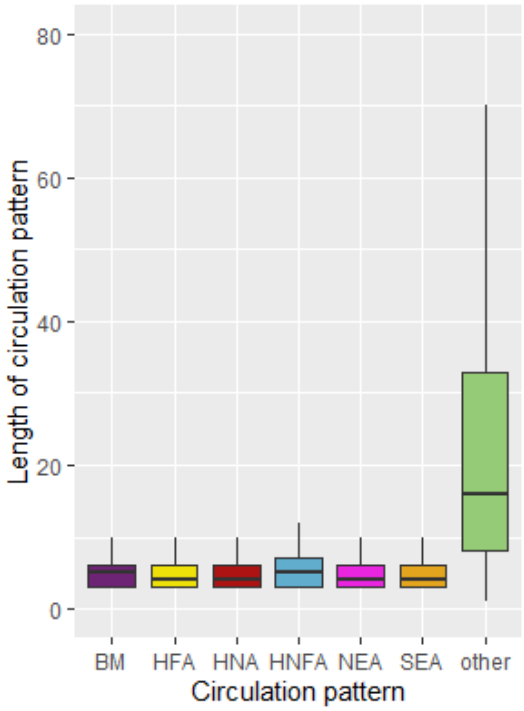

Figure 4: Length of the circulation patterns

According to Weiß (2018), a metric which can be used to investigate the presence of autocorrelation for nominal times series is Cramer's V, which measures the degree of dependece between two nominal varaibles. Having a domain from $[0, 1]$, values bigger than 0.6 imply strong association, values between 0.2 and 0.6 a medium one and values smaller 0.2 imply a weak association. Figure 5 shows the values of Cramer's V for the time series of circulation patterns (y_t) and lagged versions (y_{t-l}). For a maximum lag of degree 10, so y_{t-l} with $l = 1, ..., 10$, strong and medium associations for the lags until degree six can be identified. This autocorrelation is most likely caused by the minimum duration of three days for a circulation pattern (s. chap. 1.1). Additionally, a maximum lag of degree 365 is used to detect possible cyclical and seasonal patterns over a time span of 365 days. However, no strong association is identified between lags with a degree greater than 6 and (y_t). Thus, it remains questionable, whether the periodic fluctuations (s. figure 5, right plot) do play a decisive role at all in the data generating process and if so, whether they can be attributed to a seasonal component.

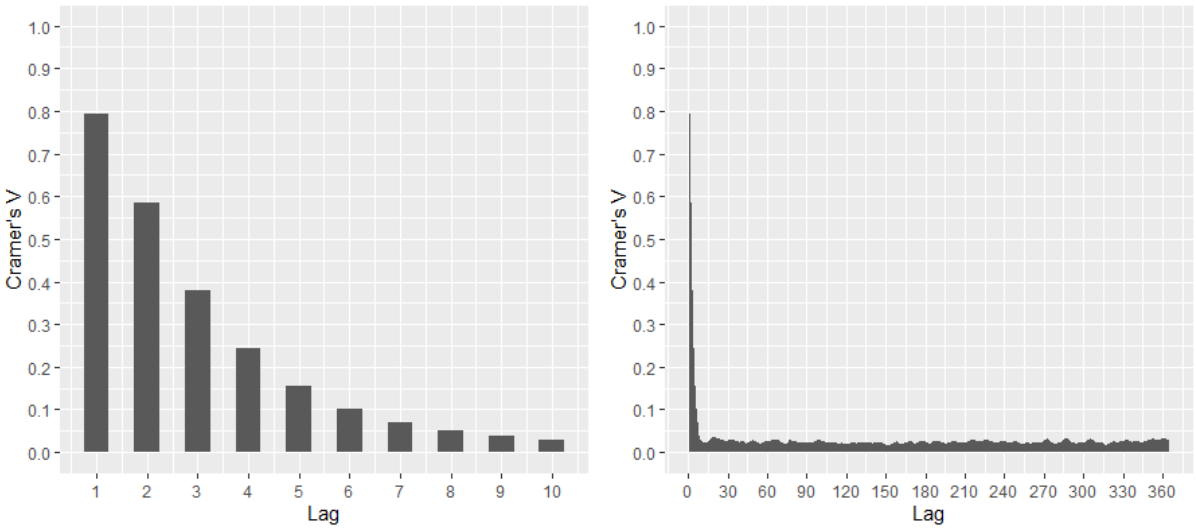

Figure 5: Left: Cramer's V κ for $l = 10$; Right: Cramer's V for $l = 365$

3 Modelling time series using structured data

3.1 Structured additive distributional regression

As stated in chapter 2.1 the goal is to model the parameters $\pi_t = (\pi_{t1}, ...\pi_{tC})^T$ of the conditional distribution $y_t|v_t \sim M(1, \pi_t)$ assumed for the respective response variable y_t. Making only use of the structured covariates s_t, one of the most flexible regression approaches are so called generalized additive models (GAMs). GAMs model the mean μ of any conditional distribution $y_t|s_t \overset{\text{i.i.d.}}{\sim} EF(\mu_t, \phi)$ stemming from the exponential family by using a so called response function h

$$\mu_t = E(y_t|s_t) = h(\eta_{structured}(s_t)). \tag{3}$$

Since h ensures possible parameter space restrictions of μ_t to be met, the specific form of the response function i.a. depends on the assumed distribution of $y_t|s_t$. In the context of GAMs, the structured predictor $\eta_{structured}$ allows for an additive inclusion of both linear and functional (nonlinear, spatial) effects of the structured covariates s_t. Thus $\eta_{structured}$ can be formulated as

$$\eta_{structured}(s_t) = X_t\beta_+ \sum_{j=1}^{J} f_j(z_{tj}) \tag{4}$$

where the purely linear effects of the covariates $x = \{x_1, ...x_P\}$ are represented via $X_t\beta$, with X_t being the t-th row of the design matrix X and β being the corresponding parameter vector. In addition to these purely linear effects, the smooth, non-linear effects of the continuous covariates $z = \{z_1, ..., z_J\}$ are included additively using $\sum f_j(z_j)$. Thus the structured covariates s can be further decomposed into $s = (x, z)$, with x representing the features with a linear effect and z representing the features with a non-linear effect. To enable a general representation as a linear model, each function f_j is represented with the help of a so called basis expansion, taking the form

$$f_j(z_j) = \sum_{d=1}^{D_j} B_d(z_{tj})w_{jd} \tag{5}$$

where B_d stands for the d-th basis function with the parameter wjd scaling the respective basis function (Fahrmeir et al. (2013b), Wood (2006). Thereby, the final shape of f_j is i.a. determined by the number D_j of basis functions with a higher number reducing the degree of smoothness and increasing the risk of overfitting. Due to this trade-off, a common approach is to initially choose a large D_j and then apply penalization in the process of estimating the parameter vector w (Wood (2006), Fahrmeir et al. (2013b)). A central component of these penalization strategies is the smoothing parameter λ controlling the trade-off mentioned above. The larger λ the smoother the function and vice versa. Since the estimation of the parameters depends on λ, it must be determined in advance as a kind of hyperparameter. Two common approaches to select an optimal λ are prediction error approaches (e.g. via the AIC or GCV) and marginal likelihoods approaches, based on the Bayesian/mixed model view of smoothing (Wood (2006)). Another aspect influencing the final form of f_j is the type of B_d. According to Wood (2006) a common strategy is to use

spline bases as regression splines, polynomial bases or B-splines in order to represent the smooth function f_j. Since the predictor structure of GAMs can be further extended to for example include random effects, varying coefficient models, Gaussian Markov random fields they are sometimes referred to as structured additive regression (SAR) (Rügamer et al. (2023), Fahrmeir et al. (2013b)).
However, depending on the distribution, a potential drawback of GAMs/SAR is that they only model the mean(s) of the respective conditional distribution. An extension, which allows the complete set of parameters $\theta = (\theta_1, ..., \theta_K)$ of the conditional distribution $y_t|s_t \sim D(\theta_{t1}, .., \theta_{tK})$ to depend on the available features s are so called generalized additive models for location, scale, and shape (GAMLSS) or more broadly structured additive distribution regression (SADR). Similarly to SAR, the individual relationship between a θ_k and s is modeled using both a parameter specific response function h_k and a parameter specific structured predictor $\eta_{structured,k}$

$$\theta_{tk} = h_k(\eta_{structured,k}(s_t)). \tag{6}$$

The response function h_k again guarantees that the parameter space restrictions for θ_{kt} are met (Klein et al. (2015); Rügamer et al. (2023)). Whilst SADR usually allow a more versatile modelling approach than SAR, for distributions that are only parameterized by its mean(s) SAR and SADR lead to the exact same modelling approach.

3.2 Structured covariates and specified usage in SADR predictors

Since the multinoulli distribution, which is assumed for the response variable y_t at hand, is only parameterized by its means, the resulting model specifications can be interpreted as both SDR or SADR. Regardless of the interpretation, the logistic softmax is used as response function to model the conditional probabilities π_t. Thus, according to Fahrmeir et al. (2013a) and based on the definition from 2.2, a conditional probability π_{tc} can be linked to possible covariates s_t via

$$\pi_{tc} = E(y_{tc}|s_t) = P(y_{tc} = 1|s_t) = \frac{exp(\eta_{structured,c}(s_t))}{1 + \sum_{q=1}^{C} exp(\eta_{structured,q}(s_t))}. \tag{7}$$

Due to the flexibility of SADR to differently include various covariates, the temporal components and characteristics described in chapter 2.1 can be easily incorporated in the structured predictor $\eta_{structured,c}$. Based on the analysis of the temporal components done in chapter 2.2 different covariates are designed/selected to be ultimately considered in the different model specifications. Since this analysis indicated the presence of autocorrelation of at least degree six, the six last, lagged values of the time series are taken into account as possible covariates (lag_1, lag_2, lag_3, lag_4, lag_5, lag_6) . Furthermore, hoping to facilitate the detection of transition days, the variable *curr_length* is designed, keeping track of the so-far duration of the current circulation pattern. Even though the data analysis in chapter 2.2 did not clearly indicate the presence of a seasonal component, the variables *season* and *month* are nevertheless considered. Furthermore, to account for a possible

Covariate	Values	Type	Usage in modelling	Abbreviation
month	{1,..,12}	categorical	Linear effect	M
season	{fall, spring, summer, winter}	nominal	Linear effect	S
year	[1900,..., 2010]	metric	Smooth effect	Y
lag_1	{BM, HFA, HNA, HNFA, NEA, SEA, other}	nominal	Linear effect, Interaction with *curr_length*	La1
lag_2	{BM, HFA, HNA, HNFA, NEA, SEA, other}	nominal	Linear effect	La2
lag_3	{BM, HFA, HNA, HNFA, NEA, SEA, other}	nominal	Linear effect	La3
lag_4	{BM, HFA, HNA, HNFA, NEA, SEA, other}	nominal	Linear effect	La4
lag_5	{BM, HFA, HNA, HNFA, NEA, SEA, other}	nominal	Linear effect	La5
lag_6	{BM, HFA, HNA, HNFA, NEA, SEA, other}	nominal	Linear effect	La6
curr_length	Min.=1, Mean=19.87 Max.:209	metric	Linear effect, Interaction with lag_1	Le

Table 1: Structured covariates and specified usage in modelling using SADR

trend in the data, the time variable *year* is included as possible explanatory variable. Table 1 gives further information w.r.t. these possible covariates.
Whilst the effect of nominal/categorical variables (e.g. *season*, lag_1 etc.) can not be modeled using smooth effects, the relationship between (quasi)-continuous variables and the response variable can be quantified via such a parametric approach (s. chapt. 3.1). Thus, the *year* variable is included via a smooth effect using thin plate regression splines as basis representation. Besides considering the remaining covariates as individual effects, an interaction between *curr_length* and $lags_1$ is additionally included. The main reason is that the mean length differs among the circulation patterns (s. chap 2.2). Thus, the effect of *curr_length* could differ depending on the value of the current circulation pattern. Table 1 contains an overview of these covariates that are considered during modelling.

4 Modelling unstructured data

4.1 Convolutional Neural Networks

For unstructured data with grid-like topology such as images, CNNs as an extension of NNs have become an established approach. A central part of the architecture of both NNs and CNNs are their layers, which are executed one after the other according to their order and with each layer consisting of m neurons. In the first layer (so-called input layer), these neurons simply receive the input and pass it on to the first of L of so-called hidden layers. In a respective hidden layer $i, i = 1, ...I$ in each neuron $z^{(ij)}, j = 1, ...J$ an affine transformation ϕ^i performed on the input passed on from the previous layer. The transformed input is then plugged into an activation function $\sigma^{(i)}$. Depending on the choice of function, it introduces a certain degree of non-linearity in order to derive at the new representation of the data. A common choice for an activation function is e.g. the rectified liner unit (ReLu) function, which simply returns the positive part of its argument z

$$g(z) = max\{0, z\}. \tag{8}$$

The last, so called output layer, does one final affine transformation. The activation function τ used in the output layer depends on the assumed data distribution of the target variable Y and thereby often corresponds to the response function used within regression approaches (Bishop (2006). Goodfellow et al. (2016)).
In the context of traditional NNs, the affine transformation done in each hidden layer i is nothing more than the weighted sum of the input, whereby the weights $W^{(i)}$ an the bias $B^{(i)}$ belong to the parameters that are optimised within the neural network. Since this weighted sum is usually calculated using all inputs Z^{i-1} from the previous layer $i-1$, these layers are often referred to as fully connected layers. For all m neurons of a certain layer i in traditional NN the 2-step computation consisting of an affine transformation and an activation function can be written as

$$Z^{(i)} = \sigma^{(i)}(\phi^{(i)}) = \sigma^{(i)}(Z^{(i-1)}W^{(i)} + B^{(i)}). \tag{9}$$

CNNs replace the matrix multiplication as affine transformation with a convolution operation in at least one of the hidden layers. Thus, receiving images with the dimensions (h,w,p) corresponding to (height, width, number of channels), a CNN passes this input to so called convolutional layers, where each convolutional layer is usually followed by a pooling /sub-sampling layer. Usually, a convolutional layer consists of F planes, with f depending on the number of filters that are to be applied on the input of the layer. In each plane the respective filter f of size (i,j,p), with $i < h$ and $j < w$, is convolved with the input. However, due to the smaller filter dimensions compared to the input dimensions, the filter is being applied iteratively to subregions of the input. Thus, the convolution is computed as the dot product between the filter and the respective subregion of the input. These subregions are known as receptive fields. Architecture-wise this concept of receptive fields is implemented via sparse connectivity, such that a neuron (which represents a single step in the step-wise application of the filter) is only connected to a subset of inputs from the previous layer. In addition, the neurons of a respective filter /plane are constrained to share the same weight values for the filter entries. Since

these filter entries are the to-be-optimized parameters of a CNN, this parameter sharing significantly reduces computational expense in CNN optimization. Analogously to traditional NNs, the convolution is then followed by an activation function resulting in so called feature maps, which then serve as input for the subsequent layer. Typically, a convolutional layer is followed by an sub-sampling/pooling layer, which serves to make the layer output invariant to small translations of the input. Therefore, each feature map of a layer is again subdivided into subsampling regions to which a pooling operation is then applied (similarly to the receptive fields in a convolutional layer). The pooling operation then calculates a corresponding summary statistic for the subsampling regions. A a popular pooling function is for example max pooling which extracts the maximum value of a receptive field (Liu et al. (2016), Bishop (2006), Goodfellow et al. (2016)). After an arbitrary number of convolutional and pooling layers, a CNN typically also includes a number of fully connected layers. In the case of the output layer, this fully connected layer serves the same purpose as in traditional NNs and is constructed using the same considerations regrading the type of activation function.

4.2 Optimization and Hyperparameters of a (C)NN

Regardless of the type of network (NN or CNN), there are several parameters used for the affine transformations which need to be estimated during training. First of all, for training the parameters are first initialized randomly with initial values. Then, so called gradient based (GD) algorithms are used to find the optimal parameter estimates. However, as using the entire training data set of size n for optimization often leads to high computational expenses, most gradient based algorithms sample B mini-batches $\{u^{(1)}, ..., u^{(M)}\}$ of size M with $M < T$ randomly from the training data. These mini-batches are then iterativley used for optimization. Therefore, for each iteration a specific mini-batch b is passed through the network. After this procedure known as forward propagation, the empirical risk between the network output and the true labels y is calculated using a certain loss function $L(f(u_t, \theta_b), y_t)$. Due to the gradient based character of the optimization algorithms, the chosen loss function must be differentiable. Since this is the case for the log-likelihood of most distributions, a common procedure is to choose the negative log-likelihood as loss function. After obtaining the empirical risk, the current parameter estimates are updated to further minimize this risk through this update. Therefore, the gradient estimate $\hat{g_b}$ of the loss w.r.t. to the respective parameter θ is used to calculate these updates

$$\hat{g_b} \leftarrow \frac{1}{m} \nabla_\theta \sum_{i=1}^{M} L(f(u_i, \theta_b), y_i) \tag{10}$$

The gradients themselves are computed successively using a procedure known as back propagation. Here, the chain rule of calculus is applied whereby the already calculated gradients are then propagated backwards through the network to enable the efficient calculation of the subsequent gradients. In addition to the gradient estimate, a learning rate α, which determines the scale of the parameter updates is used. In more basic algorithms such as stochastic gradient descent (SGD) an update step performed during a

specific iteration b can be written as

$$\hat{\theta}_b \leftarrow \hat{\theta}_{b-1} - \alpha\hat{g} \tag{11}$$

Whilst SGD still is a popular optimization algorithm, it sometimes suffers from a slow convergence. To accelerate the learning process, some algorithms incorporate a momentum v_b to the updating process. By computing this momentum as the exponentially decaying moving average of past gradients, information regarding the contribution of previous gradients to the current update is incorporated into the update step. Other approaches extend SDG by adaptively adjusting this learning rate per iteration. A very popular algorithm combining the use of both a momentum and an adaptive adjustment of the learning rate is ADAM, which was introduced by Kingma and Ba (2017). ADAM uses the bias-corrected, exponential moving averages of the gradient (= momentum) and the squared gradient (= adaptive learning rate) in order to update the parameter estimates per iteration. Therefore, besides the learning rate α, it initially requires the hyperparameters ϵ, β_1 and β_2. The latter two represent the exponential decay rates of the moving averages, whilst the former is a small constant required for numerical stability. Furthermore, at the beginning of ADAM, the parameter vector θ_0 is initialized and the two vectors m_0 (1st moment vector of the parameters) and v_0 (2nd moment vector of the parameters) are set to zero. Then, per iteration b, the gradient estimate w.r.t. to the parameters $\hat{g}_b$ is calculated using backprogpagation (s. eq. 10). Then, the biased update of both the first and second moment estimate is calculated via

$$m_b \leftarrow \beta_1 * m_{b-1} + (1 - \beta_1)g_b \tag{12}$$

$$v_b \leftarrow \beta_2 * v_{b-1} + (1 - \beta_2)\sqrt{g_b} \tag{13}$$

However, as both m_0 and v_0 are initialized as vectors of 0's, both m_b and v_b are biased towards zero especially during the first few iterations or small decay rates β. Therefore, the unbiased version is computed via

$$\hat{m_b} \leftarrow \frac{m_b}{1 - \beta_1^b} \tag{14}$$

$$\hat{v_b} \leftarrow \frac{v_b}{1 - \beta_2^b} \tag{15}$$

where the scale of correction depends on the current iteration number b and the size of the respective β. Using these bias-corrected terms, the update for the parameters is then calculated using

$$\hat{\theta}_b \leftarrow \hat{\theta}_{b-1} - \frac{\alpha\hat{m}_b}{\sqrt{\hat{v}_t} + \epsilon} \tag{16}$$

According to Kingma and Ba (2017), by combining the momentum and adaptive learning rates, ADAM is able to effectively optimize the model parameters, especially in scenarios with sparse gradients or noisy training data. In addition, Adam is generally regarded as being fairly robust to the choice of hyperparameters, though the learning rate sometimes needs to be changed from the suggested default (Goodfellow et al. (2016)).
Regardless of the choice of optimization algorithm, the general procedure of *Sampling a*

mini-batch - Forward propagation using the most recent parameter values - Loss calculation - Updating parameters using a gradient based algorithm is repeated as long as a certain stopping criterion is met. When all the mini-batches forming the complete training data have been used once for training, an epoch is completed. Usually, multiple epochs (and thereby the complete training data set) are used during training a NN.
Regarding the hyperparameters of a (C)NN, the learning rate α of the optimization algorithm is often considered to be one of the most important hyperparameters of a NN, as it can have a detrimental influence whether the GD-algorithm actually arrives at optimal parameter estimates. A learning rate that is too small not only slows down the training process unnecessarily, it can also lead to the GD-algorithm getting stuck at non-optimal estimates of the parameters. In contrast, a learning rate which too high, can cause too large weight updates leading to an oscillating loss on the training dataset (Goodfellow et al. (2016)). In addition, the learning rate also influences the effect of the chosen batch size, as smaller batch size might also require a smaller learning rate in order to arrive at a stable estimate. Furthermore, a smaller batch size increases the variance of the estimate. Due to this added noise, a smaller batch size also reduces the risk of over-fitting in comparison to larger batches (Goodfellow et al. (2016)).
Just as the learning rate and the batch size, the number of epochs can heavily influence the resulting parameter estimates and thereby the performance of the trained (C)NN. A large number of epochs generally increases the risk of over-fitting, whilst a low number of epochs amplifies the risk of under-fitting. This connection is also used in a method called early stopping, which is a common approach to select the optimal number of epochs. Here, besides quantifying the loss of the NN on the training data during optimization, the loss per epoch is also measured w.r.t. to an unseen validation set. Over the course of the epochs this validation error usually decreases up to a certain epoch. However, once the the network starts to over-fit the validation error starts to increase. So, to achieve a network with strong generalization performance, training can be halted when the smallest error w.r.t. to the validation dataset is reached. (Bishop (2006)). Using early stopping can be considered as a special case of hyperparameter tuning but also as a form of regularization in order to avoid overfitting. Another measure to avoid overfitting is to apply a method known a drop-out to the neural network. During each training iteration a random subset of neurons of one or multiple layers is set to zero and thereby excluded from the respective iteration. The so called dropout rate, which is yet another hyperparameter of a (C)NN, acts as probability of a neuron being dropped out in a respective iteration. This prevents the network to adapt certain co-dependencies among specific subset of neurons, which in turn helps to avoid overfitting (Srivastava et al. (2014)).
The hyperaprameters described in the previous section are only a few of the many hyperparameters that have to be defined for a (C)NN. Since they can heavily affect the modelling results it is vital to choose the hyperparameteres that are most likely to minimize the (estimated) generalization error. Further information w.r.t. to this hyper parameter tuning are presented in chapter 6.

Covariate	Usage in modelling	Abbreviation
$images_t$	Input for CNN d_1	I
$images_{-1}$	Input for CNN d_2	I-1
$images_{+1}$	Input for CNN d_3	I+1

Table 2: Unstructured covariates and usage during modelling using CNNs

4.3 Unstructured covariates and resulting CNN configurations

As already stated in chapter 1.2 the daily images of both the sea level pressure $[hPa]$ and geopotential height at $500hPa[m]$ are jointly used as unstructured covariate *images* for this thesis. These images have a spatial resolution of 5° x 5° corresponding to a grid of 16 x 29 (width x height) pixels per image (Mittermeier et al. (2021)). Whilst Mittermeier et al. (2021) only used the images of the respective day t as input variable per observation, in this thesis the lagged images of the 1st degree $t-1$ are also considered as $image_{-1}$. This is done to take the inherent autocorrelation between the daily classification patterns into account. Additionally, the lead images $image_{+1}$ of the subsequent day $t+1$ are also considered. The reason is to imitate the retrospective character of the classification procedure and to thereby indirectly consider the 3-day minimum duration rule during modelling. As already indicated by table 2, these different covariates are to be processed in separate CNNs. These CNNs are building blocks of the SSDR, which is described in the following chapter. Here they can be combined flexibly depending on the respective SSDR specification (s. tab. 5.2).
However, the same underlying architecture is used for each of the CNNs. Analogously to Mittermeier et al. (2021), a relatively simple architecture is used to model the respective input images. The input layer receives the images of the atmospheric predictor variables each one via a separate input channel. Then, the first of two convolutional operations is executed. Therefore, a convolutional layer follows (nmb. of filters: 8) with ReLu as subsequent activation function. Then, a layer executing batch normalization is followed by a pooling layer (max pooling, size pooling region: 2x2). Afterwards drop out (dropout rate = 0.2) is applied to the input. These steps are then repeated for the second convolutional operation, where 16 filters are applied during the convolutional layer. After the convolutional operations, ReLu is again applied as activation function, followed by a batch normalization and a dropout layer. In a "traditional" CNN setting, e.g. a fully connected output layer would then follow using e.g. the softmax function as activation and the number of classes as number of output units. However, since the thus defined CNNs are applied in the context of SSDR, the output of these network trunks is further processed by the penultimate layer of the respective SSDR specification, which is described in the subsequent chapter 5.

5 Combining structured and unstructured data

5.1 Semi-strucutred distributional regression

There are several approaches to combine statistical regression such as SADR with NNs in order to enable the joint modelling of multimodal data structures. However, many of these approaches either are limited to modelling a distributional mean regression or loose the interpretabilty of the structured effects stemming from the statistical regression approach. An approach which solves both challenges is semi-structured-distributional regression (SSDR). First introduced by Rügamer et al. (2023), it combines a SADR and a NN in one unifying NN structure . For each of the K distributional parameters θ_k of the distribution $D = (\theta_0(v), ..., \theta_k(v))$ a separate subnetwork is estimated. Each subnetwork receives the input $v = (x, z, u)$. Analogously to the previous chapters, x represents the features to be included in the structured linear part, z the features to be included in the structured non-linear part and u the (unstructured) features to be processed via a NN. Then in each subnetwork k, an additive predictor η_k is constructed such that the penultimate layer can be representend via

$$\eta_k(v_t) = f_{k,0}(x_t) + \sum_{j=1}^{r_k} f_{k,j}(z_{tj}) + \sum_{j=1}^{g_k} d_{k,j}(u_t). \tag{17}$$

Here, the $f_{k,0}(x)$ and $\sum_{j=1}^{r_k} f_{k,j}(z_j)$ can be considered part of a structured predictor analogously to the structured predictor of a SADR from 3.1. Thus, $f_k, 0(x) = X_t\beta_k$ represents the structured linear effects of x and $\sum_{j=1}^{r_k} f_{k,j}(z_{tj}) = \sum_{d=1}^{D} B_d(z_{jt})w_{jd}$ the structured non-linear effects of z. If there is an overlap in the features of x and z, the structured non-linear parts a firstly reparameterized to ensure identifiability.
If (x, z) and u do not overlap, $\sum_{j=1}^{g_k} d_{k,j}(u_t) = \hat{u}_{k,j}^T\gamma_{k,j}$ simply represents the unstructured predictor with the latent feature representation $\hat{u}_{k,j}$ learned by the (C)NN using the input features u. If however, (x, z) and u overlap an orthogonalization is done. Here, a projection of matrix $P_X^\perp$ is constructed, which enables the linear projection onto the orthogonal complement of the column space spanned by X, with $X = (x_1^\top, ..., x_T^\top)^\top$. Then $\tilde{U}_k = P_X^\perp\hat{U}_k$ is calculated, where $\tilde{U}_k$ then replaces $\hat{U}_k$ in the predictor $\eta_k = (\eta_k(v_1), ..., \eta_k(v_T))$. This orthogonalization again is done to ensure the idenfiability of the SSDR.
Regardless whether an orthgonalization has taken place, the individual η_k then serve as input for the last layer of the complete network architecture. This layer is a so called distributional layer, consisting of K output neurons one for each of the distributional parameters $(\theta_0(v), ..., \theta_k(v))$. Figure 6 shows an examplary architecture used to represent a SSDR. Estimating the complete model can then be done by optimizing the negative log-likelihood using the optimization procedure described in chapter 4.2.

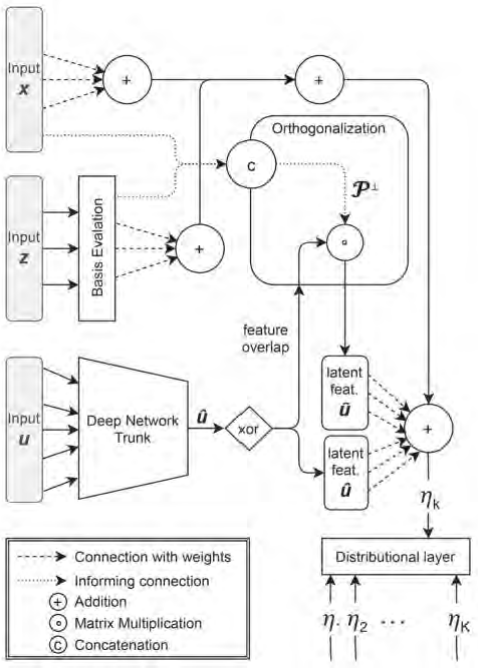

Figure 6: Exemplary SSDR architecture with orthgonalization (Rügamer et al. (2023))

5.2 Implemented SSDR

Since the assumed distribution for the circulation patterns consists of seven distributional parameters, namely $\pi_t = (\pi_{tBM}, ..., \pi_{tother})$, seven subnetworks $k = 1, .., 7$ are constructed per model specification s. Each subnetwork receives the exact same, specification dependent subset of explanatory variables $v^s = (s^s, u^s) = (x^s, z^s, u^s)$. Since there is no overlap between the specified structured covariates x, z (s. chapter 3.2) and the images as unstructured inputs u (s. chapter 4.3), neither reparameterization nor orthgonalization has to be performed. Thus, the individual specifications can be simply expressed by using their differing η_k^s. These η_k^s are obtained by combining the already defined covariates from table 1 and the specified CNN configurations from table 2 in a different manner per model. Table 5.2 provides an overview over the resulting specifications.
For each of the specifications, the last layer consists of 7 output neurons each receiving the output η_k^s of the penultimate layer as input. Since a multinoulli distribution of the circulation pattern is assumed (s. chapt. 2.1), the logistic softmax is used as activation function in the last layer (analogously to the response function used in chapter 3.2). Optimization is then done via minimizing the negative log-likelihood using ADAM as gradient descent algorithm. Additionally, to evade overfitting, early stopping (patience = 6) is applied during training. The maximum number of epochs is set to be 50. Furthermore, the optimal values of the hyperparameters learning rate and batch size are selected via automated hyperparameter tuning. The concrete implementation in R is done using the deepregression package by Rügamer et al. (2022).

s	Model s	$\eta^s_{k,structured}$	$\eta^s_{k,unstructured}$
1	I	-	d1(images)
2	I, M	$season$	d1($images$)
3	I, S	$month$	d1($images$)
4	I, Y	s($year$)	d1($images$)
5	I, S, Y	$season$ + s($year$)	d1($images$)
6	I, L	$lag_1 + lag_2 + lag_3 + lag_4 + lag_5 + lag_6$	d1($images$)
7	I, L*Le	$lag_1 + lag_2 + lag_3 + lag_4 + lag_5 + lag_6 + curr\_length{:}lag_1$	d1($images$)
8	I, L, Y	$lag_1 + lag_2 + lag_3 + lag_4 + lag_5 + lag_6$ + s($year$)	d1($images$)
9	I, I-1	-	d1($images$) + d2($images_{-1}$)
10	I, I+1	-	d1($images$) + d3($images_{+1}$)
11	I, I-1, I+1	-	d1($images$) + d2($images_{-1}$) + d3($images_{+1}$)

Table 3: Predictors η^s_k per SSDR model specification

6 Methodical procedure

6.1 Accounting for imbalanced classes

As already stated in chapter 2.2, a high disproportion among the number of observations of each class is present in the given data. This preponderance of observations in one class can cause models to inherently only learn the characteristics of this majority class (in this case the residual class). This in turn leads to the model struggling to adequately differentiate observations of the minority classes from the majority class and thereby misclassifying these instances. Thus, failing to take this imbalance into account can result in models that are highly biased towards the majority class in their performance. This tendency is especially problematic in cases where the minority classes are the instances of interest. As this is the case in this thesis, it is vital to account for the present class imbalance and to thereby construct models that are able to distinguish properly between all classes (Fernández et al. (2018)).
A variety of approaches exist to address this phenomenon of imbalanced data. For example, data level approaches try to rebalance the uneven class proportions by over- or undersampling. Other, so-called algorithm level approaches, try to modify existing algorithms to prioritize the learning regarding the minority classes. A specific type of this approach is so-called cost-sensitive learning, which assigns a specific misclassification cost per class. Usually, the minority classes receive higher costs than the majority classes. Then, there are several approaches to consider these costs within the framework of modelling. One of them is to include them into the respective loss function. This direct consideration leads to them being minimized in the process of training resulting in a model more sensitive to the minority classes (Kukar and Kononenko (1998)). There are several methods to obtain these costs. A very common strategy which is also used in this thesis is to use the inverse class frequency as cost per class. For a certain class c it is calculated via

$$weight_c = \frac{T}{T_c} \tag{18}$$

with T denoting the complete number of observations and T_c referring to the number of observations belonging to c. Table 4 shows the resulting class weights thus obtained and used during modelling.

	BM	HFA	HNA	HNFA	NEA	SEA	other	$\sum$
T_c	3451	1424	1195	573	838	854	32206	40541
$weight_c$	11.75	28.47	33.92	70.74	48.37	47.46	1.26	-

Table 4: Implemented class weights

6.2 Performance estimation

6.2.1 Resampling strategies for performance estimation

As the predictive performance of the models on unseen data is of interest in the scope of this thesis, it is necessary to obtain a reliable and unbiased estimate for this performance. Therefore, simply evaluating the performance on the training data does not suffice, as using the same data for both training and performance estimation usually leads to overly optimistic estimates regarding the ability to generalize.
To acquire such an estimate so-called resampling strategies are used. In settings with cross-sectional, purely i.i.d. data the established approach is to use K-fold cross validation as a resampling strategy. In this procedure the entire data set D is usually randomly divided into K disjoint subsets. Then the cross-validation is done K-times, whereby each of the K subsets serves as test set $D_{test,k}$ to calculate the performance metrics of interest in the respective iteration k. The remaining $K-1$ subsets are combined into a training set $D_{train,k}$ and used to estimate a model in the respective iteration k. For purely i.i.d. data, the thus obtained K performance estimates are then averaged. This results in a less biased, more reliable estimator for the performance of the models on unseen data.
However, according to Joseph (2022), there is no resampling strategy which is considered the gold standard approach in modelling time series. If either no serial dependence exists between the respective observations or the model is a purely autoregressive representation of a stationary time series, traditional K-fold cross validation can be used (Bergmeir et al. (2018)). However, in cases where an inherent temporal dependence (e.g. autocorrelation) is present, stationarity is not guaranteed and/or additional covariates are to be included, using k-fold cross-validation can lead to an unreliable estimator of future performance.
As Joseph (2022) states, in these cases it is more preferable to use some type of repeated hold-out splitting (repeated HOS), which respects the inherent temporal structure of the underlying data generating process. For each repetition/iteration k the data D is split into two subsequent parts using a so-called origin point. This origin point marks the end of the respective training set $D_{train,k}$, and thereby the beginning the corresponding test set $D_{test,k}$ used for performance estimation in the respective iteration k. Analogously to K-fold CV the resulting K performance estimates of the repeated hold-out splitting are averaged to obtain a more stable estimate w.r.t. future performance of the model. In the context of a (C)NN, which applies early stopping (s.4.2) this 2-way data partition has to be modified. As mentioned in chapter 4.2, in early stopping the performance of the model per epoch is evaluated during training using a so-called validation set D_{val} to detect the optimal stopping epoch. However, since it results directly from the training process, this performance estimate can not be used for performance evaluation as it might lead to an overly optimistic estimate of the performance. Thus, in the context of (C)NNs performing early stopping, the data must be split into $D = (D_{train,k}, D_{val,k}, D_{test,k})$ for a respective iteration k of the chosen resampling strategy. This three-way split thus results in two origin points, one indicating the end of the training set and the other the end of the validation set.
The specific design of the repeated hold-out procedure depends on multiple aspects, which can be divided into a sampling strategy, a window strategy and a calibration strategy (Joseph (2022)). The sampling strategy decides how the origin points are to be sampled

from the data set. For example, the origin points can be sampled randomly for each iteration. Another approach is to create a rolling origin in which the origin point moves closer to the present in fixed intervals over the course of the hold-out iterations (Hewamalage et al. (2023)). Whilst the length of the test and validation set (L_{test}, L_{val}) is usually simply set in advance, the length of the training data L_{train} set is determined by the window strategy. Usually, two different approaches are distinguished in this regard. In the expanding window approach, L_{train} depends on temporal position of the origin point as the training split starts at the beginning of the available data. So, for example, in a rolling origin set-up the training data expands with each iteration. On the other hand, in the fixed window approach the training data has a fixed length L_{train}. The calibration strategy determines whether a model should be retrained for a new iteration (and thereby a new origin point) of the repeated hold-out. Whilst retraining or recalibrating usually is the preferred way to go, in some cases, e.g. a highly granular rolling origin set up, retraining the model for each origin point might be computationally very expensive or even non-feasible. In these cases, instead of retraining the model, the model (e.g. the test data) is simply updated with data (Joseph (2022), Tashman (2000), Hewamalage et al. (2023)).
In the context of this thesis, a repeated HOS is chosen as resampling strategy. The main reason for this is that the chosen strategy should lead to reliable results across all model specifications. Since K-fold CV can only be applied in certain cases in the context of time series, it can not be used in this thesis. Thus, repeated HOS is chosen, whereby the temporal character is adequately taken into account across all specifications. To use the data in the most systematic way possible, a rolling origin approach is used to select the respective origin points from the data. Since early stopping is applied in this thesis, for each iteration k of the repeated HOS the model is (re-)trained using $D_{train,k}$ and $D_{val,k}$. This results in recalibration/retraining as a calibration strategy.
As nested resampling is used in this thesis, this general resampling strategy is applied twice, using different specifications w.r.t. set lengths and origin points. To avoid redundancies the concrete specifications can be found in chapter 6.2.3, where both the theoretical concept and the concrete implementation of nested resampling are presented.
In addition to defining the general resampling strategy, a specific procedure for performance evaluation using $D_{test,k}$ must also be defined for this work. The standard procedure is to conduct the performance estimation per iteration k simply "feeding" the out-of-sample test set $D_{test,k}$ to the respective retrained model. For the model specifications, that do not include covariates depending on previous observations (e.g. lagged values) this gives an reliable estimate w.r.t. the future performance of the model. However, for specifications using predecessor-covariates, this estimate can be too optimistic, as performance of the model per point t depends on previous prediction results, where a prediction error might have taken place. Thus, for these modelling approaches an additional testing procedure is done, where the performance estimate is created by iteratively updating the test data $D_{test,k}$ using the previous predictions.

Hyperparameters	
Learning rate	batch size
1e-04	64
1e-03	64
1e-04	128
1e-03	128

Table 5: Implemented hyperparameter search space

6.2.2 Hyperparameter Tuning

As already stated in chapter 4.2, the hyperparameters of a model can have a decisive influence on its generalization ability and therefore its (predictive) performance. Thus, the goal of hyperparameter tuning is to find the set of hyperparameters that optimize the performance. Whilst manual hyperparameter tuning requires highly in-depth knowledge about the effect of the hyperparameters w.r.t. the chosen modelling approach and the underlying data generating process, automated tuning procedures offer a more systematic, data driven search for the optimal hyperparameters (Goodfellow et al. (2016), Bischl et al. (2012)). Therefore, a search space is defined, which contains all possible combinations of the hyperparameters to be tuned. Depending on the chosen search strategy this search space is then used differently. The traditional strategy is grid search in which the entire search space is used once for tuning. Other strategies include random search in which potential hyperparameter combinations are drawn randomly from the search space or more advanced search strategies such as genetic algorithms. However, whilst these alternative approaches might be beneficial in high dimensional settings, grid search still leads to reliable results in low-dimensional search spaces. In addition, it is easy to implement and trivial to parallelize (Bergstra and Bengio (2012)). Since only the learning rate and the batch size are to be tuned for this thesis, the resulting search space is 2-dimensional. Thus, grid search is chosen as search strategy in this thesis. Table 5 shows the implemented search space.
Regardless of the chosen search strategy, a model is then trained per possible hyperparameter combination and its performance is evaluated. To obtain a more stable and reliable estimate this is usually done using a resampling strategy such as K-fold CV or repeated HOS (s. chap. 6.2.1). Then, the hyperparameter combination is chosen with the lowest associated estimate of the model's generalization error. This hyperparameter combination can then be used to either train a model on the complete data set or to be used in the outer step of so-called nested resampling, which is explained in the following subsection 6.2.3.

6.2.3 Nested resampling

If no automated hyperparameter tuning is executed, it is sufficient to estimate the model's performance on unseen data using a resampling strategy as described in chapter 6.2.1. However, if automated hyperparameter tuning is done, tuning the hyperparameters and evaluating the performance using the same resampling steps will lead to information leakage from the testing steps to the training procedure. This 'training on the test set' can thus lead to overfitting. To avoid this and to obtain a reasonable performance estimator, the model selection must be part of the training process through hyperparameter tuning. The actual test set, which is then used for performance evaluation, should not be used in this process. Therefore nested resampling is used. Here an inner resampling procedure is used to tune the hyperparameter used in the respective outer resampling step, which is conducted for performance evaluation (Bischl et al. (2012)). Thus, given that a NN with early stopping is trained, in each of the K_{outer} iterations of the outer resampling a model trained using $D_{train,k_{outer}}, D_{val,k_{outer}}$ and its performance is estimated on $D_{test,k_{outer}}$. The optimal hyperparameter combination for this iteration $\lambda_{k_{outer}}$ is obtained using the inner resampling procedure. Here, for each iteration k_{inner} the training data $D_{train,k_{outer}}$ from the respective outer resampling iteration k_{outer} is divided into $D_{train,k_{outer}} = \{D_{train,k_{outer,inner}}, D_{val,k_{outer,inner}}, D_{test,k_{outer,inner}}\}$. The hyperparameter tuning is then done analogously to the procedure described in the chapter 6.2.2
Since hyperparameter tuning is to be performed for the SSDR specifications, nested resampling is used to obtain performance estimates as accurate as possible. Based on the rolling origin, expanding window approach established as resampling strategy for this work (s. chap 6.2.1), starting from 1970, the origin point of the training set is moved back 10 years per iteration of the outer resampling split. The length of the following validation L_{val} and test sets L_{test} is set to 10 years each $L_{val,k_{outer}} = 10$, $L_{test,k_{outer}} = 10$, resulting in 3 outer resampling iterations $K_{outer} = 5$. Figure 7 visualizes the folds of the outer resampling. W.r.t. to the inner resampling procedure $L_{val,k_{inner,outer}} = 5$, $L_{test,k_{inner,outer}} = 5$ and $K_{inner,outer} = 3$ is used. The resulting design of the inner nested resampling splits is displayed in table 6.

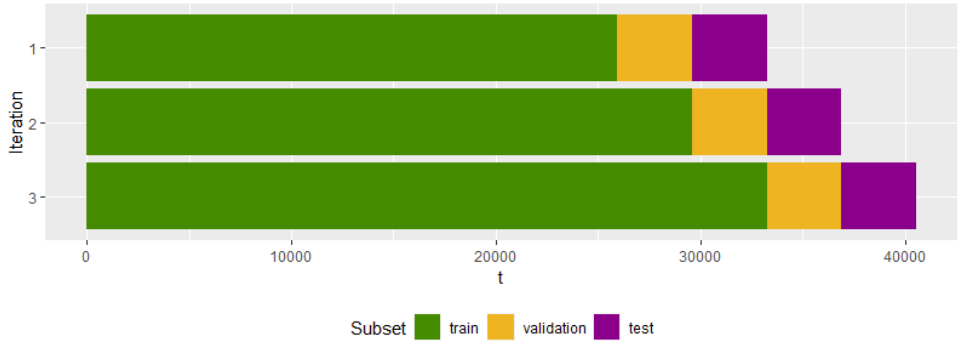

Figure 7: Implemented outer resampling splits

outer	1	2	3
$D_{train,k_{outer}}$	1900-1970	1900-1980	1900-1990
$D_{train,k_{outer,1}}$	1900-1950	1900-1960	1900-1970
$D_{val,k_{outer,1}}$	1950-1955	1960-1965	1970-1975
$D_{test,k_{outer,1}}$	1955-1960	1965-1970	1975-1980
$D_{train,k_{outer,2}}$	1900-1955	1900-1965	1900-1975
$D_{val,k_{outer,2}}$	1955-1960	1965-1970	1970-1980
$D_{test,k_{outer,2}}$	1960-1965	1965-1975	1975-1985
$D_{train,k_{outer,3}}$	1900-1960	1900-1970	1900-1980
$D_{val,k_{outer,3}}$	1960-1965	1970-1975	1980-1985
$D_{test,k_{outer,3}}$	1965-1970	1975-1980	1985-1990

Table 6: Implemented inner resampling splits

6.3 Performance evaluation metrics

To assess the performance of the different model specifications as holistically as possible, both the initial overview of the performance of the different model specifications (chap. 7) and the in-depth, class wise analysis of selected specifications (chap. 7) should include different metrics that reflect different aspects of the performance.
According to Ferri et al. (2009), metrics for performance evaluation consist of i.a. probabilistic and qualitative metrics. Probabilistic metrics focus on reliability of a given model, not only measuring when it fails but whether it has selected the wrong class with a high or low probability. In case of a softmax activation function in the output layer of a neural network, a common choice is to use categorical cross entropy as probabilistic metric

$$ce = \frac{1}{n} \sum_{t=1}^{T} \sum_{c=1}^{C} y_{tc} log f_c(u_t, \theta). \tag{19}$$

where y_{tc} is the c'th element of the one-hot-encoded label y_t of an observation t (s. chapter 2.1). Due to use of the log-function it inherently penalizes confidently wrong predictions more heavily and thus gives an indication of how reliable the probability estimates are (Ferri et al. (2009)). Due to this the categorical cross entropy is used as probabilistic overview metric.
In contrast to probabilistic metrics, qualitative metrics evaluate the performance w.r.t. the absolute number of errors. The qualitative metrics used for the in-depth, class wise analysis are calculated based on a confusion matrix, which compares the classification by a model and the true classification for each class c. Figure 8 exemplifies such a confusion matrix with $C = 3$, C being the number of classes.

Figure 8: Exemplary multi-class confusion matrix

For a specific class c (in figure 8: $c = 2$), the abbreviation TP_c refers to the number of correctly classified observations of a class c, while FP_c refers to the observations that were mistakenly assigned to class c. The FN_cs in a respective column represent to observations falsely classified as another class, whilst TN_c refer to the observations correctly classified to a different class. Two metrics, which give an intuition about the performance w.r.t. a specific class c, are the class specific recall and the class specific precision

$$recall_c = \frac{TP_c}{TP_c + FN_c} \tag{20}$$

$$precision_c = \frac{TP_c}{TP_c + FP_c} \tag{21}$$

The class specific recall quantifies the ratio of correctly predicted true positives among all truly positive observations belonging to a class c and thereby reflects the sensitivity of a model w.r.t. this specific class. The class specific precision measures the ratio of accurately identified positives out of all positive predictions made by the model for a class c (Grandini et al. (2020)). Furthermore, precision and recall can be considered to be competing metrics, since the recall can simply increase via solely predicting positives, which in turn decreases the precision. Due to this precision-recall trade off and since both metrics regard different aspects of the performance w.r.t. a specific class c, both metrics are used during performance evaluation. A metric that combines both precision and recall and thereby quantifies the models ability to optimize both, is the class-specific F1-score $F1_c$. It is calculated as the harmonic mean of the precision and the recall of a specific class c (Grandini et al. (2020)):

$$F1_c = 2(\frac{precision_c * recall_c}{precision_c + recall_c}). \tag{22}$$

Besides reflecting different aspects of the performance, the metrics used for the overview should additionally account for the imbalanced distribution of the response variable such that each class is weighted equally per metric. Thus, the macro-averaged pendants of the above explained class-specific metrics are used. In addition to providing a comprehensive overview, these "overview"-metrics assign equal importance to all classes regardless of their size, since they are first calculated on class level and then averaged. Thus, the macro precision, the macro recall and the macro F1-score over all classes C are calculated as (Grandini et al. (2020), Opitz and Burst (2021)):

$$MAP = \frac{\sum_{c=1}^{C} precision_c}{C} \tag{23}$$

$$MAR = \frac{\sum_{c=1}^{C} recall_c}{C} \tag{24}$$

$$MF1 = \frac{\sum_{c=1}^{C} F1_c}{C}. \tag{25}$$

Furthermore, since they give an initial intuition regarding the class-wise performances of the models, the class-specific F_1-scores are additionally provided as an overview metric. Another common overview metric is the accuracy, which simply reflects the number of correctly classified observations. However, in the context of imbalanced data this metric is not necessarily a reliable indicator for a high classification quality w.r.t. all classes. For example, if a high proportion of observations stemming from the residual class (the majority class) is correctly classified and a high proportion of days stemming from the remaining circulation pattern classes (the minority classes) is incorrectly classified, the accuracy will still be reasonably high. Whilst the accuracy is nevertheless supplied as an overview metric, it has to be interpreted with caution due to the high imbalance present in the circulation patterns.

7 Results

7.1 Overview and discussion of results

Using the overview metrics established in chapter 6.3, table 7 provides an initial insight to the performance of the model specifications. As already explained in chapter 6.2.1, the performance of the specifications that include predecessor-dependent covariates, is once tested using the original test data set containing the true values of predecessor-dependent variables. However, this known-predecessor evaluation is likely to be too optimistic, as prediction errors might have occurred for the predecessor, which then influence the following predictions. Thus, for these specifications, the performance is additionally estimated using an iterative procedure, where the test data set is updated step-by-step with previous predictions. Due to this, table 7 provides both performance estimates for these specifications, with the known-predecessor one being presented in brackets.
Regarding the general performance of all models, it is noticeable that all of them have a higher $F1_c$-score for the residual class *other* than for the remaining circulation patterns associated with droughts. A possible reason could be that the inverse class frequency does not lead to a sufficient re-weighting of the classes and thus to a preference for the majority class *other* during training.
The following sub-chapters summarize the findings from table 7 and provide possible explanations for certain results. To avoid redundancies, structurally similar models are presented together. Additionally, model 1, which serves as a kind of base line, is examined separately. Based on this, model 6 is also investigated in more detail, as it shows at least some predictive potential. Theses further examinations are done using both class specific confusion matrices and plots regrading the predictive behaviour of the models. The corresponding confusion matrices for the remaining models can be found in the appendix A. In addition, the loss curves for each model during the training/validation iterations of the outer resampling procedure are provided in the appendix A. The lower validation losses in these figures can be attributed to the class weights only being applied during training not for validation (Chollet (2023)). Due to this, these loss-functions can not be properly used for the comparison between training and validation curves. However, they can still give an intuition whether the models converges during training. Since all the training and the validation curves stagnate after a certain number of epochs, one can conclude that the specified models arrive at least at a local optimum during these iterations.

7.1.1 Models 2-5: Structured, predecessor-independent covariates

The inclusion of structured, predecessor-independent covariates (models 2-5), does not seem to lead an overall, systematic increase in performance across all classes compared to model 1. More specifically, whilst they increase the $F1_c$-score for some classes, neither the variable *month* nor *season* lead to an overall improvement in recall, precision or accuracy. Model 4, which includes the *year* covariate, also does not lead to an overall increase of the $F1_c$-scores for all classes. Nevertheless, it achieves a marginally higher MAP (+0.01) and accuracy (+0.05) than the baseline model 1. However, this increased accuracy can be attributed to the increased performance on the residual class *other* compared to model 1. Additionally, model 4 is the only one of models 2-5 that has a lower categorical cross

entropy than model 1. This indicates that model 4 is more robust leading to more reliable probability estimates than model 1. Including *season* and *year* jointly (model 5), leads to an equally ambiguous picture as models 2 and 3, with neither a class-wide improvement in $F1_c$-scores nor in the general overview metrics.
There could be several reasons, why the inclusion of structured, predecessor-independent covariates does not lead to any noteworthy performance improvement. First of all, these covariates might just not be very informative/discriminative for the classification of circulation patterns. Including such redundant features, can cause a model to capture noise or random variations instead of meaningful patterns for classification. This in turn can reduce the model's ability to accurately predict the target variable and thereby reduces the model's predictive performance. Misinterpreting this noise/variations introduced by the redundant variables, can also lead the model overfitting on the specific noise in it's training data - which in turn decreases the performance during testing. Since the EDA in chapter 2.2 did not indicate the presence of any seasonal component, the variables *month* and *season* thus might simply be irrelevant for the classification of circulation patterns associated with droughts. The same might be true for *year*. However, given that the EDA indicated at least the existence of a slight trend, the challenge could also lie in the interpolation of the trend on the subsequent test data. Another explanation for the decreased performance could be of course, that the SSDR models simply can not find the true optimal estimates e.g. due to misspecifications in the model itself, in the hyperparameters or in the chosen data split.

7.1.2 Models 6-8: Structured, predecessor-dependent covariates

Comparing the performance of models 6 and 7 with model 1, one has to distinguish between the known-predecessor and unknown-predecessor performances. Regarding the known-predecessor performance, it appears that including lagged values of the circulation patterns (model 6) leads to a definite increase in performance. This is especially apparent w.r.t. the circulation patterns associated with drought. Including an interaction between lag_1 and *curr_length* (model 7) seems to even further improve the performance. Additionally, both models have a much lower categorical cross entropy than model 1, indicating that they lead to much more reliable probability estimates.
In contrast, using the unknown-predecessor testing procedure leads to completely opposite results. For model 6, the cross entropy and the performance regarding the circulation patterns associated with droughts decreases compared to model 1. Only for the residual class *other* a small improvement of the $F1_{other}$-score can be quantified. Including the interaction does not lead to any kind of improvement. On the contrary, the $F1_c$-scores of model 6 actually worsens drastically across all classes. All in all, this indicates, that the lagged values hold great explanatory power w.r.t. the classification of circulation patterns. However, in an unknown-predecessor test setting, this also leads to false predictions having an immense negative effect on the predictive power of the model.
Including *year* in addition to the lagged values of the circulation pattern (model 8) slightly decreases the performance on the known-predecessor test set compared to model 6. In contrast, it interestingly increases the performance on the unknown-predecessor test set. One reason for this behaviour could be, that including the *year* variable decreases the esti-

mated effects of the lagged covariates, as the model (maybe falsely) attributes some effect to the *year* covariate. On one hand, in the known-predecessor context, this could lead to a decrease of performance, since the information provided by the (known) lagged values is not used to full extent. On the other hand, in the unknown-predecessor setting, this decreases the negative effect that false predictions have on the predictive power of the model.

7.1.3 Models 9-11: Further unstructured covariates

Including the lagged images or a combination of both the lagged and lead images (models 8 and 10) does not lead to an overall improvement w.r.t. the class-wise $F1_c$-scores. Both models outperform model 1 regarding their average accuracy, average MAP and average cross entropy, with model 10 being slightly better than model 8. The inclusion of lead images (model 9) improves the predictive performance the most. Compared to model 1 it has slightly higher/equivalent values for the $MF1$ (+0.01), the MAR (+0.02) and the MAP (+0.00). Only its average accuracy (0.67) decreases by 0.02, which can be probably attributed to its poorer performance on class *other*. For the classes BM, HNA and HFA this model has a definite increase in performance compared to model 1. This could indicate, that by including these lead values, the 3-day minimum duration rule can be modeled a little bit more adequately.
Generally speaking the inclusion of these further covariates does not yield a substantive increase in predictive performance. One on hand this could be simply caused by a possible redundancy of these features w.r.t. classifying circulation patterns associated with droughts. On the other hand, the chosen CNN architecture for these images could fail to arrive at a representation that contains relevant/discriminative information w.r.t. to the classification of circulation patterns. This could be due to e.g. suboptimal hyperparameters or due to a too simple CNN architecture leading to under-fitting. Then, the possible explanatory/predictive power of these additional covariates could not be use to full extent.

7.2 Model 1: Images

Since it acts as a kind of base line model, the specification using only images is further evaluated. Its confusion matrix averaged over the three test sets of the nested cross-validation is shown in table 8. It becomes apparent that the model performs best w.r.t. the residual class *other*, achieving a class-specific recall of 0.82 and a class-specific precision of 0.86. In contrast, it struggles to correctly classify circulation patterns truly associated with droughts as such, resulting in low recall values for these classes. This is not only caused by the relatively low ratio of true positives, but also due to high number of false negatives in these classes. Looking at the class-wise distribution of false-negatives it appears that the model has a tendency of falsely assigning the residual class *other* to instances actually belonging to circulation patterns associated with drought. The precision values of these classes are also relatively low, indicating that the model also tends to falsely assign observations from other classes to a respective circulation pattern associated with droughts. Interestingly, the absolute distribution of these false positive values shows, that the class that is mostly falsely classified into a circulation pattern associated with

	1 (I)	2 (I, M)	3 (I, S)	4 (I, Y)	5 (I, S, Y)	6 (I, L)	7 (I, L, Le)	8 (I, L, Y)	9 (I, I-1)	10 (I, I+1)	11 (I, I-1, I+1)
						$F1_c$					
other	0.83	0.62	0.78	0.86	0.79	0.86 (0.93)	0.08 (0.93)	0.82 (0.93)	0.85	0.81	0.86
BM	0.22	0.23	0.26	0.23	0.27	0.08 (0.74)	0.00 (0.77)	0.09 (0.74)	0.24	0.33	0.13
HFA	0.21	0.08	0.31	0.22	0.23	0.19 (0.63)	0.03 (0.63)	0.33 (0.65)	0.21	0.29	0.2
HNA	0.23	0.17	0.15	0.16	0.07	0.13 (0.61)	0.00 (0.71)	0.21 (0.62)	0.09	0.24	0.16
HNFA	0.17	0.09	0.14	0.13	0.15	0.00 (0.58)	0.00 (0.62)	0.15 (0.56)	0.21	0.16	0.10
NEA	0.10	0.01	0.07	0.09	0.03	0.07 (0.42)	0.00 (0.57)	0.01 (0.49)	0.04	0.08	0.09
SEA	0.28	0.06	0.08	0.18	0.14	0.13 (0.46)	0.00 (0.62)	0.19 (0.55)	0.04	0.20	0.23
Avg. $MAF1_c$	0.29	0.18	0.26	0.27	0.24	0.21 (0.62)	0.02 (0.69)	0.26 (0.65)	0.24	0.30	0.25
Avg. MAR	0.38	0.26	0.32	0.31	0.27	0.24 (0.64)	0.13 (0.76)	0.32 (0.72)	0.25	0.40	0.29
Avg. MAP	0.30	0.29	0.28	0.31	0.25	0.25 (0.70)	0.11 (0.67)	0.26 (0.64)	0.33	0.30	0.34
Avg. acc	0.69	0.52	0.64	0.74	0.66	0.74 (0.88)	0.06 (0.89)	0.68 (0.87)	0.74	0.67	0.74
Avg. ce	0.91	1.21	0.98	0.85	0.92	1.09 (0.39)	15.03 (0.42)	1.09 (0.41)	0.90	0.91	0.87

Table 7: Avg. performances evaluated during the outer resampling procedure; In brackets: Avg. performance on the known-predecessor test sets

droughts is the residual class *other*. Also, when quantifying the relations between the class-wise precision and recall-values, the trade-off between precision and recall becomes apparent. The circulation patterns associated with droughts that have the smallest recall values (BM, HFA, $HNFA$) have precision values higher than their recalls. Vice versa, the classes with higher recalls (HNA, NEA, SEA) have a lower precision compared to the corresponding recall values.
To take a further look into the predictive behaviour of the model, figure 9 shows the predicted class of each observation in test set vs. its true class per iteration of the outer resampling strategy. Whilst no clear pattern is visible (e.g. the model being "stuck" at a certain class once this class was predicted once), is appears that the model sometimes lags behind or is too hasty in assigning a new circulation pattern. Thus, it could be possible, that the model struggles especially with predicting transition days, i.e. the end/start day of a circulation pattern. However, since this notion is not entirely conclusive from figure 9, the performance of the model w.r.t. these transition days is further evaluated.

	Labels								
	BM	HFA	HNA	HNFA	NEA	SEA	other	$\sum$	Precision
BM	69.33	0.67	0.67	0.00	1.00	3.33	93.00	168.00	0.43
HFA	8.00	18.33	1.33	4.33	2.00	6.33	32.00	72.33	0.24
HNA	29.33	1.67	37.67	12.00	2.00	3.00	201.33	287.00	0.16
HNFA	1.33	5.33	2.33	14.00	1.00	1.67	31.00	56.67	0.18
NEA	36.00	14.67	2.00	2.33	9.33	0.67	87.33	152.33	0.06
SEA	11.00	22.67	3.33	5.33	0.00	33.33	88.00	163.67	0.21
other	303.33	27.00	22.67	18.00	11.67	24.33	2345.33	2752.33	0.86
$\sum$	458.33	90.33	70.00	56.00	27.00	72.67	2878.00	-	-
Recall	0.16	0.19	0.53	0.16	0.33	0.49	0.82	-	-

Table 8: Model 1: Avg. confusion matrix

As the confusion matrix (s. table 9) displays, the model indeed performs worse w.r.t. transition days, which results in both lower recall and precision values over all classes compared to table 8. This decreased performance in the context of transition days might be attributed to the sometimes noisy labels for these transitions days caused by the discrepancy between the continuous character of the pressure systems used for classification (the images) and the discrete character of the classification itself.

	Labels								
	BM	HFA	HNA	HNFA	NEA	SEA	other	$\sum$	Precision
BM	17.33	0.67	0.00	0.00	0.33	2.00	6.00	26.33	0.66
HFA	3.00	3.00	0.67	0.33	0.67	2.67	6.67	17.00	0.17
HNA	10.00	1.00	10.33	4.33	0.67	2.00	20.67	49.00	0.24
HNFA	0.67	1.33	1.33	3.00	0.33	0.33	3.67	10.67	0.23
NEA	12.33	4.67	1.00	1.33	2.33	0.33	13.00	35.00	0.08
SEA	3.67	6.67	2.00	1.33	0.00	9.00	11.00	33.67	0.28
other	113.00	11.33	10.00	6.33	5.67	12.33	170.67	329.33	0.52
$\sum$	160.00	28.67	25.33	16.67	10.00	28.67	231.67	NA	NA
Recall	0.12	0.10	0.42	0.15	0.24	0.35	0.73	NA	NA

Table 9: Model 1: Avg. confusion matrix of transition days

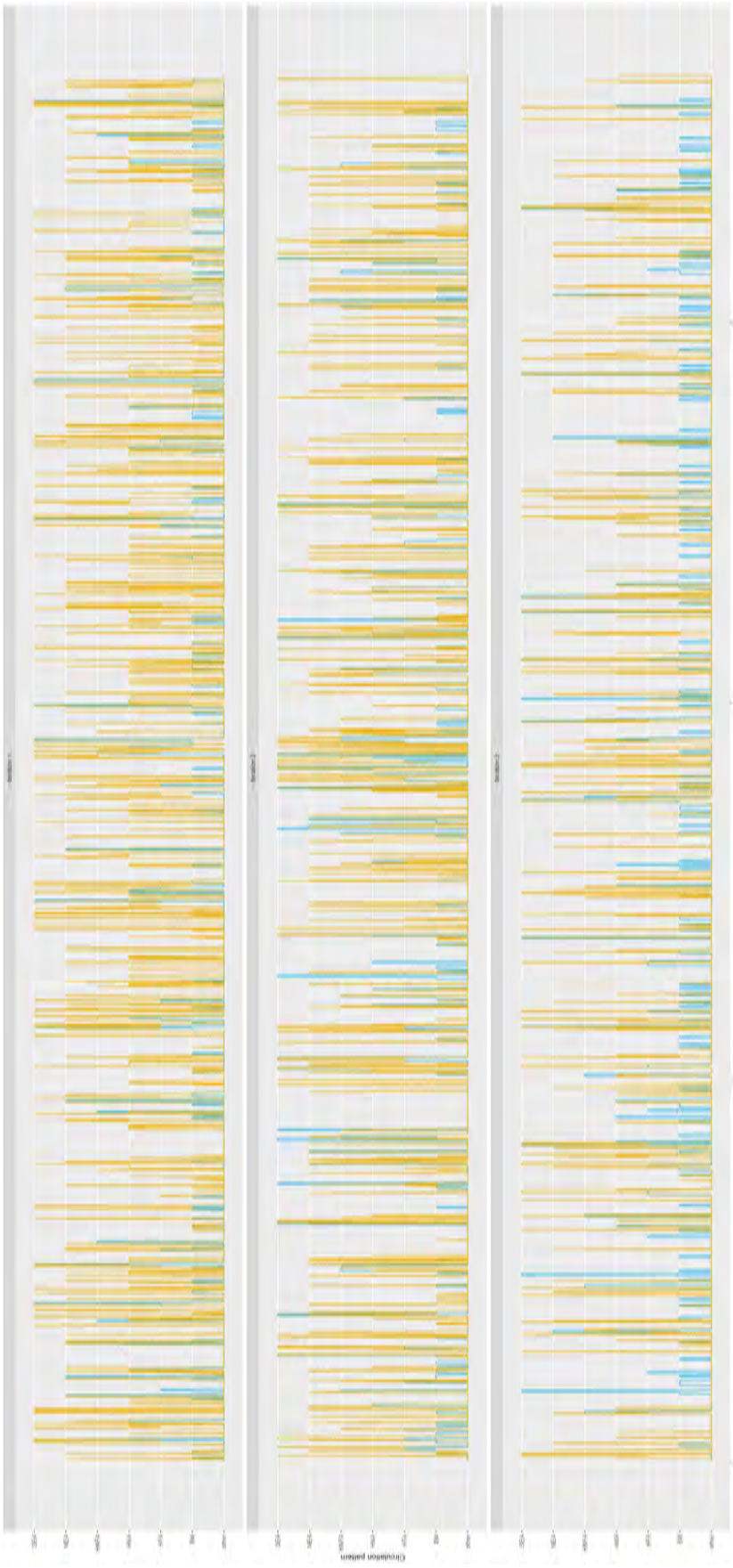

Figure 9: Model 1: Predictive behaviour

7.3 Model 6: Images and lagged circulation patterns

As stated in chapter 3.2, the underlying idea of including lagged versions of (y_t) as structured covariates is to account for the autorcorrelation present in the data. However, this strategy proves only partly successful. When evaluated on the known-predecessor test set, the inclusion of lags indeed leads to an increased performance across all classes compared to the model using only images. Table 10 shows the resulting confusion matrix, where both the recall and the precision values have improved for all classes.

	Labels								
	BM	HFA	HNA	HNFA	NEA	SEA	other	$\sum$	Precision
BM	296.33	2.00	0.33	2.33	0.33	3.33	43.67	348.33	0.86
HFA	5.00	55.33	0.00	0.67	0.33	3.00	22.33	86.67	0.65
HNA	2.33	1.33	44.33	1.67	0.33	0.00	24.00	74.00	0.64
HNFA	0.00	1.00	0.67	25.67	0.00	0.00	9.00	36.33	0.74
NEA	1.33	2.00	1.00	1.00	17.00	1.00	17.33	40.67	0.45
SEA	2.00	4.67	1.00	0.67	0.00	30.33	12.33	51.00	0.68
other	151.33	24.00	22.67	24.00	9.00	35.00	2749.33	3015.33	0.91
$\sum$	458.33	90.33	70.00	56.00	27.00	72.67	2878.00	-	-
Recall	0.66	0.63	0.63	0.56	0.58	0.44	0.96	-	-

Table 10: Model 6: Avg. confusion matrix on the known-predecessor test sets

Using the unknown-predecessor procedure for testing leads to an entirely different result. As table 11 shows, both the precision and the recall decrease across all classes compared to table 10. Looking at the absolute distribution of false negatives, it appears that the model now has as an increased tendency of simply predicting *other* for the circulation patterns associated with droughts. More specifically, excluding SEA, the amount of falsely as *other* classified values has nearly doubled for the remaining classes. In addition, it becomes apparent that model 1 outperforms model 6, when model 6 is evaluated using the unknown-predecessor test set. To further compare the predictive behaviour of model 6, figure 10 displays the predictions of the different testing procedures and the true classes. As to be expected, the predictions of the known-predecessor testing procedure are relatively similar to the true classes. However, the model still seems to struggle to exactly predict the exact ending/beginning of a circulation pattern. In contrast, when relying on previous predictions during testing, the model sometimes seems to get "stuck" at the residual class *other*. This leads to large sections simply being classified as *other*. One reason could be, that the model (as already indicated) struggles to correctly classify transition days. Due to the dependence on the previous predictions, failing to detect these can then result in this failure to trail along for the following predictions.

	Labels								
	BM	HFA	HNA	HNFA	NEA	SEA	other	$\sum$	Precision
BM	24.33	0.33	0.00	1.67	0.00	6.00	63.67	96.00	0.37
HFA	4.67	15.00	0.33	3.33	2.67	8.33	36.33	70.67	0.21
HNA	6.67	1.67	10.67	3.00	0.00	1.00	56.33	79.33	0.21
HNFA	0.00	0.00	1.00	0.00	0.00	0.67	7.67	9.33	0.00
NEA	26.00	16.00	9.00	2.33	6.00	4.00	43.33	106.67	0.04
SEA	4.67	10.67	1.67	6.33	0.00	10.67	30.33	64.33	0.11
other	392.00	46.67	47.33	39.33	18.33	42.00	2640.33	3226.00	0.82
$\sum$	458.33	90.33	70.00	56.00	27.00	72.67	2878.00	-	-
Recall	0.06	0.18	0.15	0.00	0.20	0.19	0.92	-	-

Table 11: Model 6: Avg. confusion matrix on the unknown-predecessor test sets

Suspecting that the model struggles with transition days, table 12 displays the confusion matrix for these days. This confusion matrix is calculated using the known-predecessor testing procedure, as the resulting performance can be seen as an upper bound. The decreased performance values across all classes compared to table 10 indeed indicate that the model fails to reliably classify transition days. Just as with model 1, this failure can be partly attributed to the inherent labelling process of the circulation patterns in the Hess & Breswosky catalogue. However, the performance differences between all days and transition days for this model is much higher than the differences for model 1 (s. table 9). Thus, it could be possible that the already existing problem regarding the classification of transition days is exacerbated by the inclusion of lagged values.

	Labels								
	BM	HFA	HNA	HNFA	NEA	SEA	other	$\sum$	Precision
BM	54.33	1.67	0.33	1.00	0.33	3.00	29.67	90.33	0.61
HFA	3.67	8.00	0.00	0.33	0.33	2.33	6.67	21.33	0.35
HNA	1.33	1.00	8.67	0.67	0.33	0.00	6.00	18.00	0.48
HNFA	0.00	0.67	0.67	3.33	0.00	0.00	2.00	6.67	0.33
NEA	1.33	1.67	0.00	0.67	4.33	0.33	6.67	15.00	0.30
SEA	0.67	2.00	1.00	0.00	0.00	4.67	3.33	11.67	0.48
other	98.67	13.67	14.67	10.67	4.67	18.33	177.33	338.00	0.53
$\sum$	160.00	28.67	25.33	16.67	10.00	28.67	231.67	-	-
Recall	0.36	0.28	0.35	0.28	0.40	0.19	0.76	-	-

Table 12: Model 6: Avg. confusion matrix of transition days on the known-predecessor test sets

8 Summary and limitations

In this thesis several SSDR specifications were trained. Their predictors included different combinations of both the unstructured image data and the structured temporal covariates. To obtain an estimate of the individual model performances, nested, repeated hold-out splitting was used. Grid search was applied to select the optimal values of both the learning rate and the batch size for each outer split of the nested resampling. Performance evaluation was done using cross entropy and the qualitative metrics stemming from the resulting confusion matrices.
All in all, it was shown that including predecessor-independent, structured covariates, does not lead to an overall improvement of the predictive performance compared to model 1. Whilst these models did improve the performance w.r.t. to some classes, they never lead to a joint increase of performance for all classes. Considering predecessor-dependent, structured covariates did lead to a substantial performance improvement when evaluated on the known-predecessor test set. In contrast, the evaluation on the unknown-predecessor test set showed a drastic decrease in performance. Both this and the analysis of transition days indicated, that these models especially struggled with the correct classification of transition days, which in turn lead to a potentation of prediction errors in an unknown predecessor setting. The inclusion of further unstructured covariates did not lead to any substantive performance improvements across all classes. However, the inclusion of lead images did increase the average macro performance metrics compared to model 1.
Regarding the modelling procedure some limitations exist. First of all, the implemented search space represents no exhaustive form of hyperparameter optimization. Both the number of hyperparameters to be optimized and the number of possible values could be increased to possibly improve the effective capability of all model specifications. Secondly, such an adaption would also require the use of a more sophisticated tuning algorithm such as bayesian hyper parameter tuning or genetic algorithms. Thirdly, another approach to account for the imbalanced distribution in the circulation patterns could be tested. Since all models seem to have a tendency towards the residual class, identifying an approach that further emphasizes the learning of the circulation patterns associated with drought could improve the overall predictive performance across all specifications. Finally, the inclusion of lagged values seems to generally be a promising approach to improve the effective capacity of the model. However, since its performance seems to especially depend on the ability of the model to detect transition days, a possible approach is to try a two-step modelling procedure. The first step would be a binary classification, whether a specific day is a transition day or not. This classification could then be used as additional structured covariate together with both the lagged circulation patterns and the images of the atmospheric predictor variable of a respective day. Depending on the predictive performance of the first modelling step, this could lead to an improved classification especially w.r.t. transition days.

Figure 10: Model 6: Predictive behaviour

A Appendix

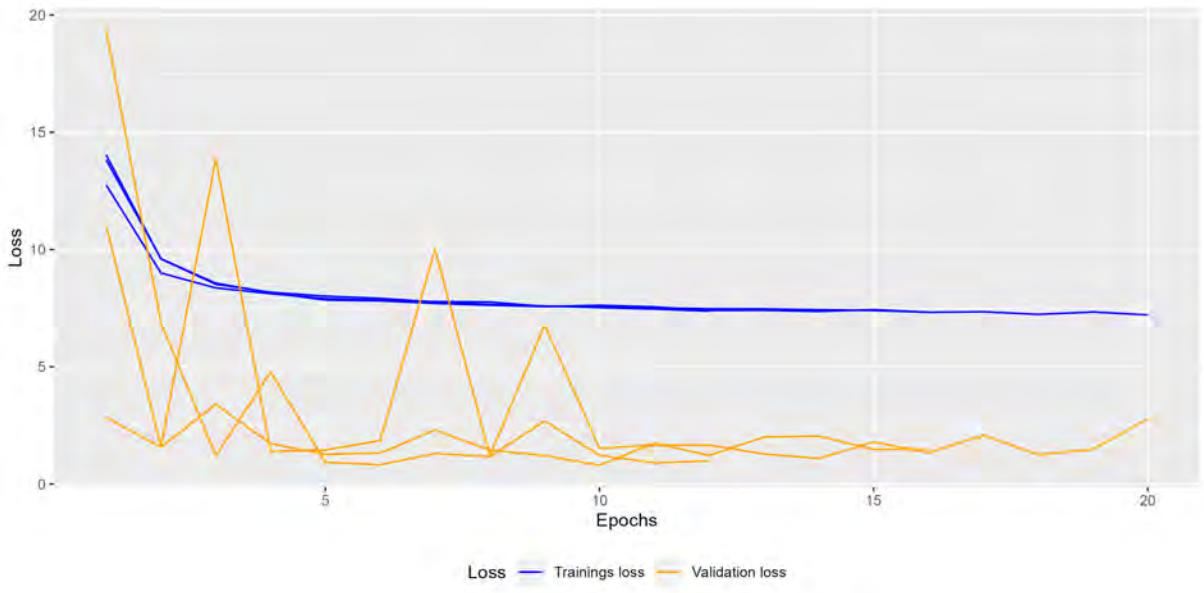

Figure 11: Model 1 (I): Loss curves during training iterations

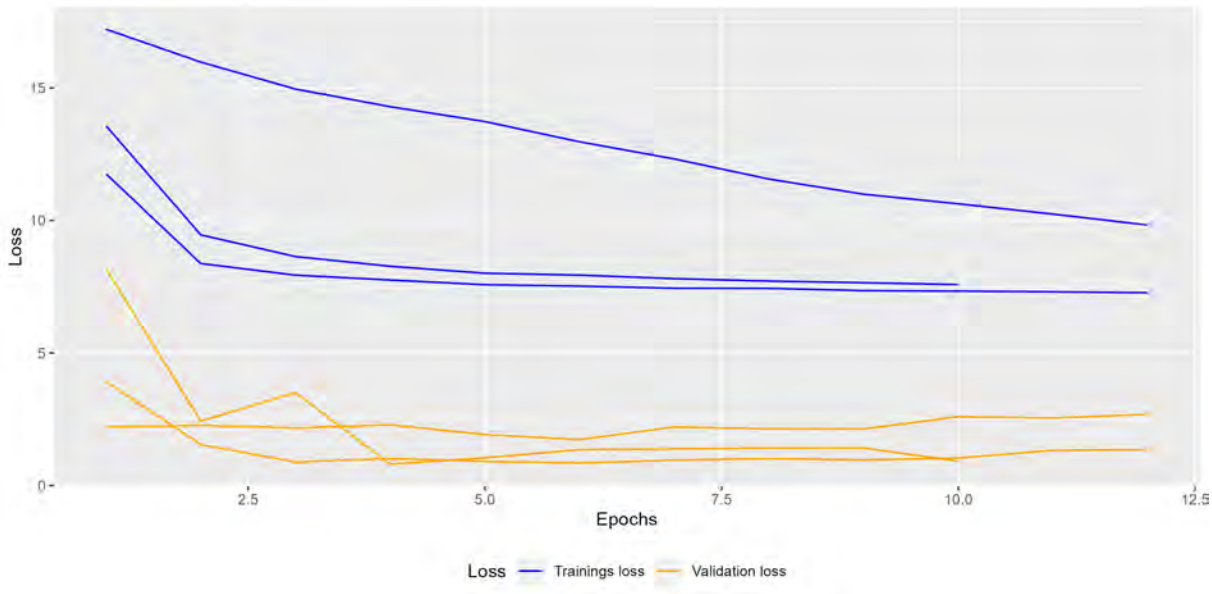

Figure 12: Model 2 (I, M): Loss curves during training iterations

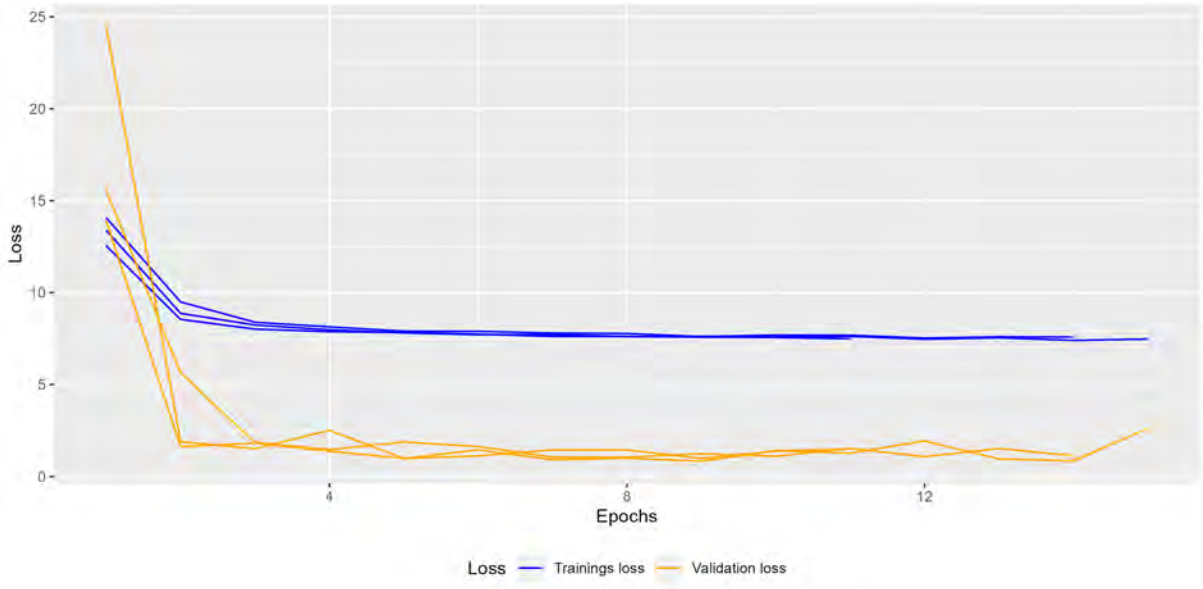

Figure 13: Model 3 (I, S): Loss curves during training iterations

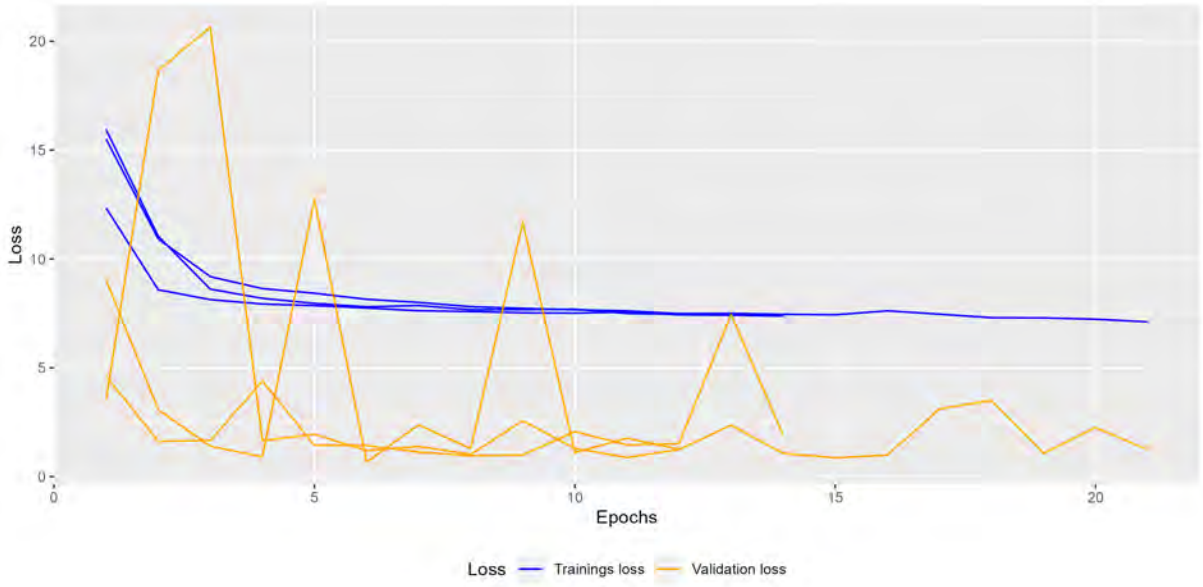

Figure 14: Model 4 (I, Y): Loss curves during training iterations

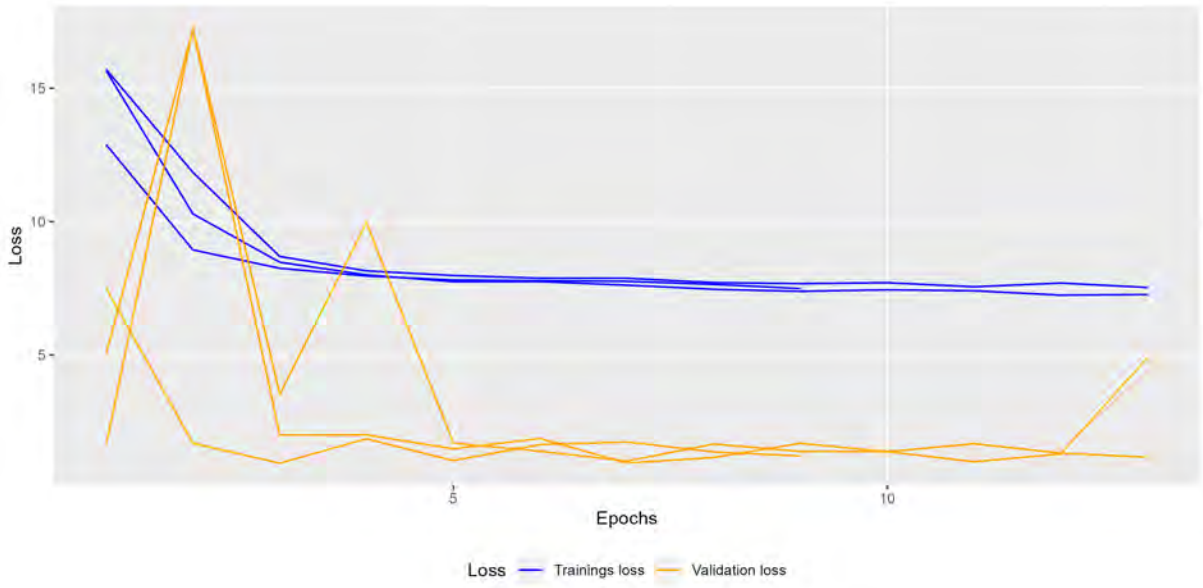

Figure 15: Model 5 (I, S, Y): Loss curves during training iterations

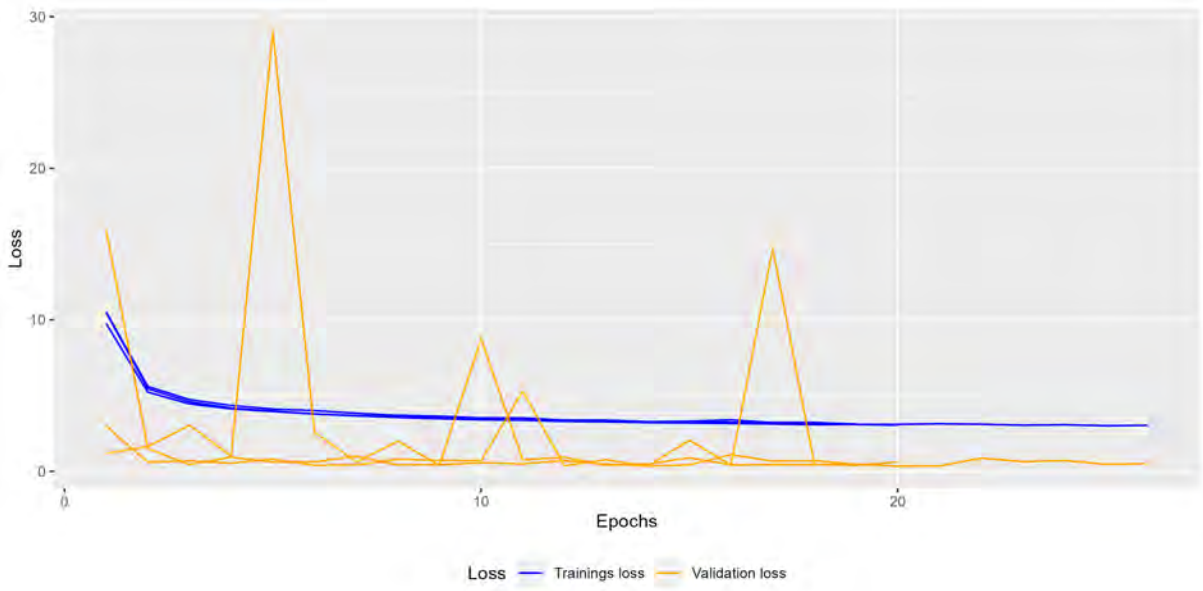

Figure 16: Model 6 (I, L): Loss curves during training iterations

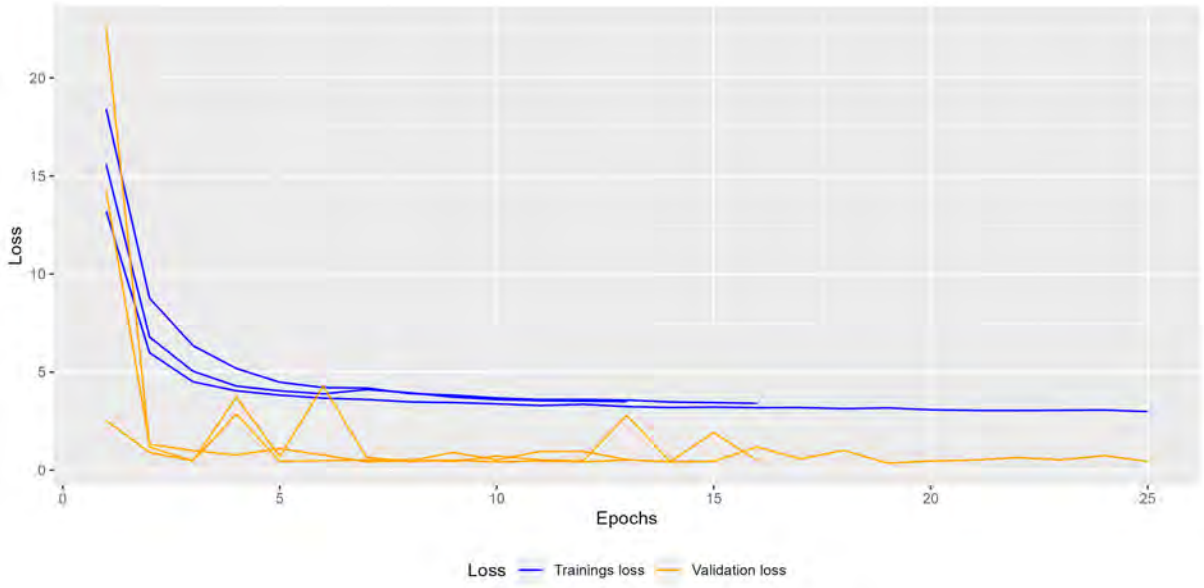

Figure 17: Model 7 (I, L, L*Le): Loss curves during training iterations

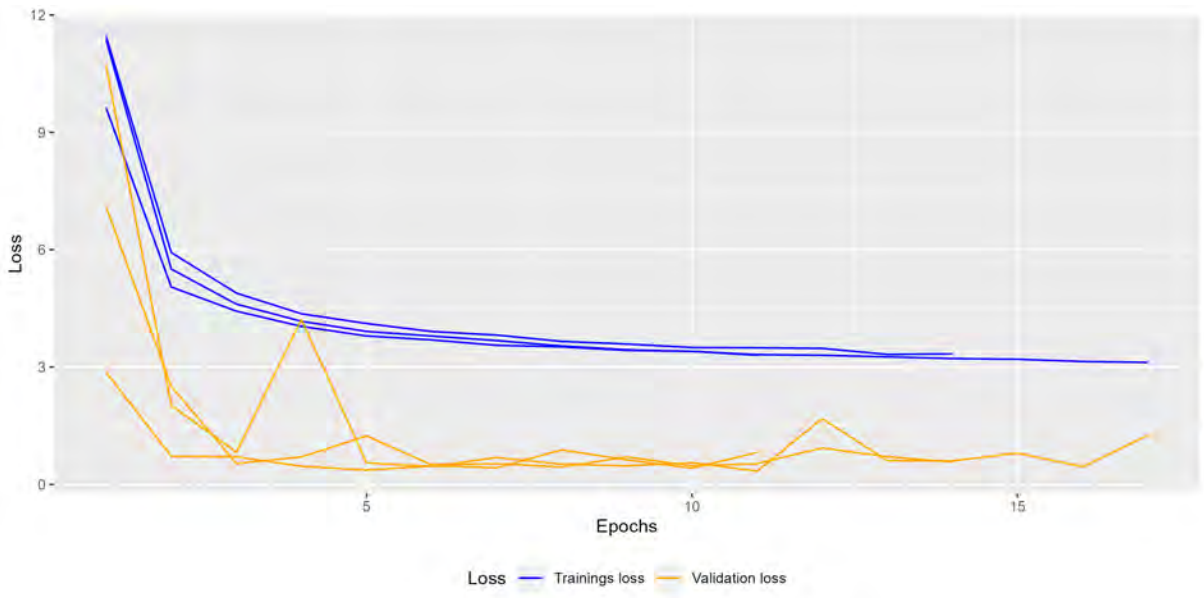

Figure 18: Model 8 (I, L, Y): Loss curves during training iterations

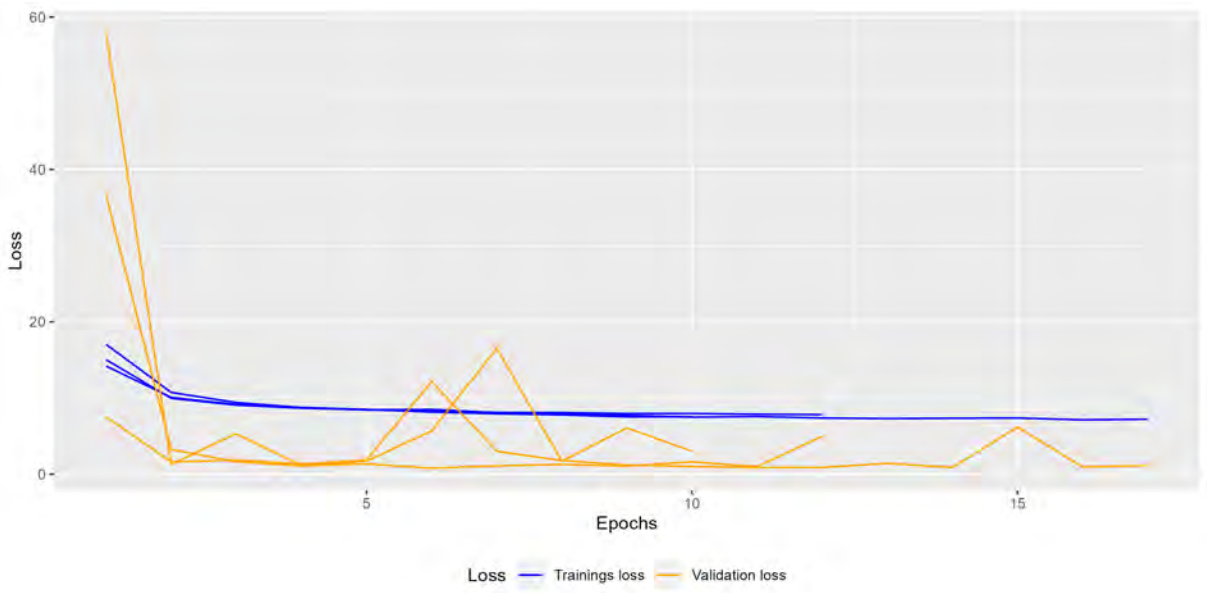

Figure 19: Model 9 (I, I-1): Loss curves during training iterations

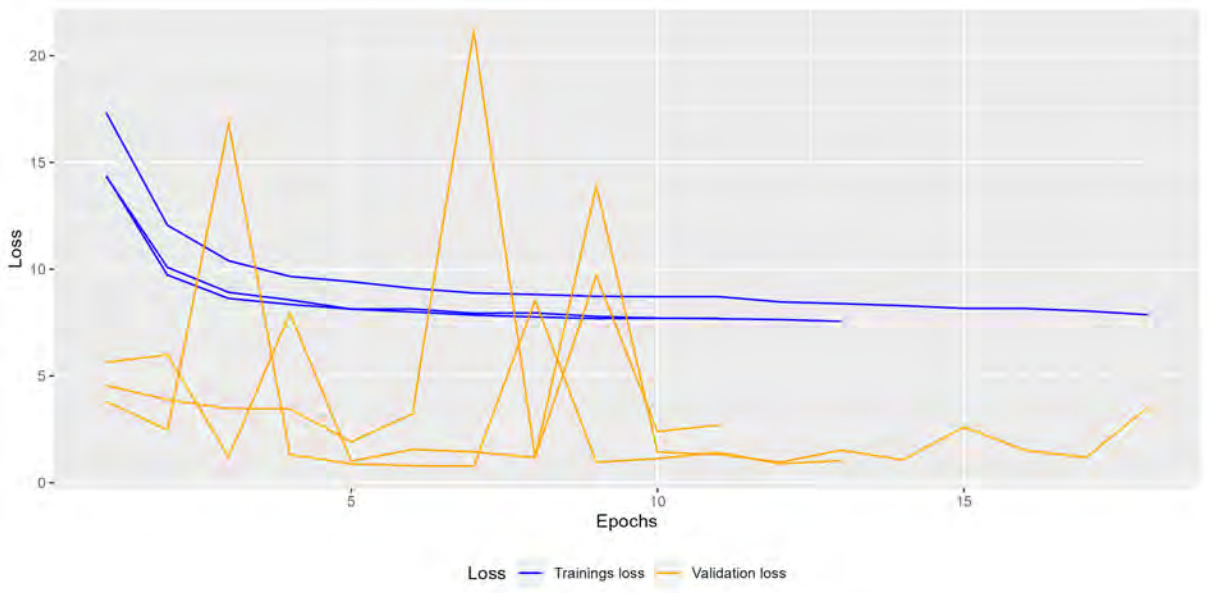

Figure 20: Model 10 (I, I+1): Loss curves during training iterations

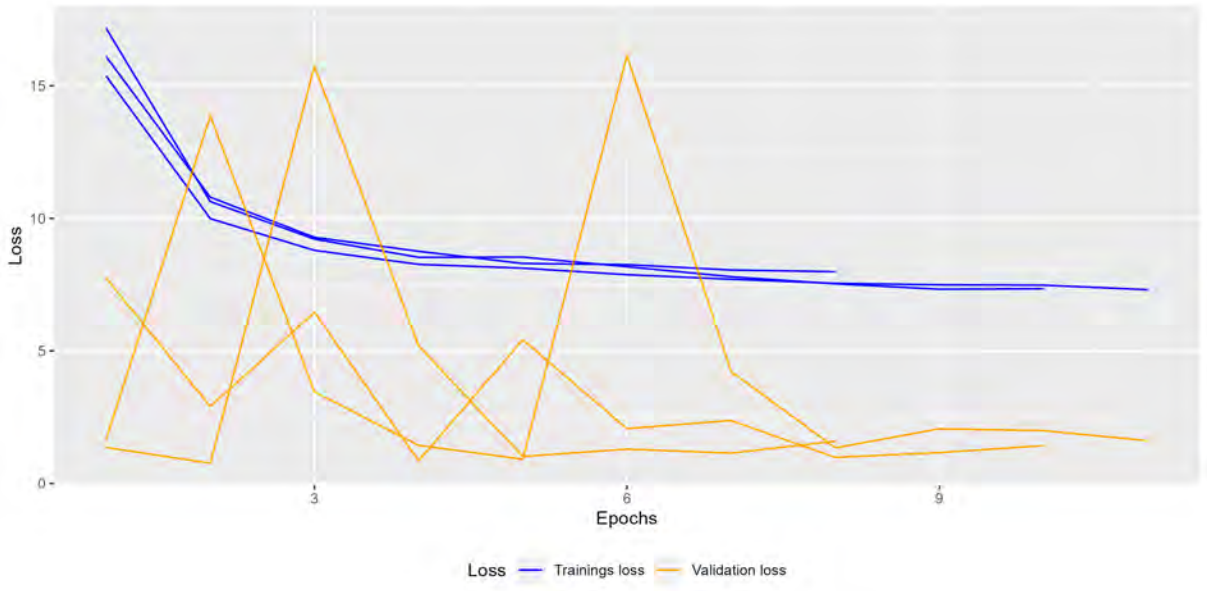

Figure 21: Model 11 (I, I-1, I+1): Loss curves during training iterations

	Labels								
	BM	HFA	HNA	HNFA	NEA	SEA	other	$\sum$	Precision
BM	181.67	13.00	6.00	8.00	10.00	19.67	682.33	920.67	0.28
HFA	0.33	5.33	0.33	2.67	0.00	3.67	9.00	21.33	0.31
HNA	12.67	3.33	26.33	10.00	2.33	7.00	172.33	234.00	0.45
HNFA	1.33	3.00	7.33	6.33	2.33	2.33	51.33	74.00	0.07
NEA	48.67	22.33	6.00	6.33	4.00	5.67	253.33	346.33	0.00
SEA	2.00	9.67	0.00	7.00	0.00	5.00	24.00	47.67	0.06
other	211.67	33.67	24.00	15.67	8.33	29.33	1685.33	2008.00	0.88
$\sum$	458.33	90.33	70.00	56.00	27.00	72.67	2877.67	-	-
Recall	0.43	0.06	0.37	0.17	0.13	0.05	0.59	-	-

Table 13: Model 2 (I, M): Avg. confusion matrix

	Labels								
	BM	HFA	HNA	HNFA	NEA	SEA	other	$\sum$	Precision
BM	58.00	4.67	2.33	1.67	3.00	7.67	55.67	133.00	0.51
HFA	0.33	1.00	0.33	0.67	0.00	1.00	1.67	5.00	0.38
HNA	4.33	0.67	6.67	3.33	1.33	3.67	13.67	33.67	0.47
HNFA	1.00	0.67	2.67	2.00	1.00	0.00	4.67	12.00	0.14
NEA	16.00	9.00	3.00	3.33	1.67	2.33	28.00	63.33	0.01
SEA	1.67	3.00	0.00	0.67	0.00	1.67	4.67	11.67	0.11
other	78.67	9.67	10.33	5.00	3.00	12.33	123.33	242.33	0.56
$\sum$	160.00	28.67	25.33	16.67	10.00	28.67	231.67	-	-
Recall	0.40	0.04	0.28	0.17	0.14	0.05	0.51	-	-

Table 14: Model 2 (I, M): Avg. confusion matrix for transition days

	Labels								
	BM	HFA	HNA	HNFA	NEA	SEA	other	$\sum$	Precision
BM	149.00	2.00	2.00	0.00	1.67	9.00	428.00	591.67	0.27
HFA	11.67	41.33	1.33	16.33	1.67	21.00	75.67	169.00	0.24
HNA	1.00	0.00	9.33	0.67	0.00	1.67	27.67	40.33	0.33
HNFA	4.00	9.00	20.00	20.00	4.33	2.00	157.33	216.67	0.10
NEA	17.33	10.00	0.67	0.33	5.67	1.33	69.67	105.00	0.04
SEA	0.33	0.00	0.33	0.67	0.00	5.00	27.67	34.00	0.12
other	275.00	28.00	36.33	18.00	13.67	32.67	2091.67	2495.33	0.84
$\sum$	458.33	90.33	70.00	56.00	27.00	72.67	2877.67	-	-
Recall	0.35	0.44	0.12	0.35	0.19	0.07	0.73	-	-

Table 15: Model 3 (I, S): Avg. confusion matrix

	Labels								
	BM	HFA	HNA	HNFA	NEA	SEA	other	$\sum$	Precision
BM	45.67	1.33	0.33	0.00	1.33	4.00	32.67	85.33	0.56
HFA	5.67	11.00	1.00	4.00	0.67	7.33	12.00	41.67	0.25
HNA	0.67	0.00	2.33	0.33	0.00	1.00	3.67	8.00	0.31
HNFA	1.33	3.00	6.33	5.33	2.00	0.67	16.67	35.33	0.16
NEA	6.00	3.33	0.00	0.00	0.67	0.67	10.33	21.00	0.02
SEA	0.00	0.00	0.33	0.33	0.00	2.00	1.67	4.33	0.37
other	100.67	10.00	15.00	6.67	5.33	13.00	154.67	305.33	0.51
$\sum$	160.00	28.67	25.33	16.67	10.00	28.67	231.67	-	-
Recall	0.32	0.36	0.09	0.35	0.06	0.07	0.65	-	-

Table 16: Model 3 (I, S): Avg. confusion matrix for transition days

	Labels								
	BM	HFA	HNA	HNFA	NEA	SEA	other	$\sum$	Precision
BM	80.00	0.33	0.33	0.33	0.33	3.00	97.33	181.67	0.47
HFA	4.67	19.00	1.00	8.00	1.67	7.33	33.00	74.67	0.32
HNA	6.33	0.00	17.67	3.67	1.00	1.67	70.00	100.33	0.14
HNFA	5.67	12.00	6.67	13.00	2.33	1.67	70.33	111.67	0.12
NEA	13.33	5.33	0.33	0.00	4.33	0.00	20.00	43.33	0.16
SEA	7.33	15.33	1.00	8.33	0.00	20.00	47.00	99.00	0.14
other	341.00	38.33	43.00	22.67	17.33	39.00	2540.00	3041.33	0.84
$\sum$	458.33	90.33	70.00	56.00	27.00	72.67	2877.67	-	-
Recall	0.17	0.22	0.24	0.22	0.14	0.32	0.88	-	-

Table 17: Model 4 (I, Y): Avg. confusion matrix

	Labels								
	BM	HFA	HNA	HNFA	NEA	SEA	other	$\sum$	Precision
BM	19.33	0.33	0.00	0.33	0.33	1.00	9.67	31.00	0.63
HFA	2.33	3.00	0.67	1.33	0.67	3.67	6.00	17.67	0.25
HNA	2.67	0.00	3.67	2.00	0.33	1.00	8.33	18.00	0.14
HNFA	1.67	3.33	2.67	2.33	0.67	0.67	6.00	17.33	0.12
NEA	4.67	2.67	0.33	0.00	0.67	0.00	3.33	11.67	0.02
SEA	2.67	5.00	1.00	1.67	0.00	4.67	7.00	22.00	0.14
other	126.67	14.33	17.00	9.00	7.33	17.67	191.33	383.33	0.50
$\sum$	160.00	28.67	25.33	16.67	10.00	28.67	231.67	-	-
Recall	0.11	0.11	0.15	0.13	0.06	0.21	0.82	-	-

Table 18: Model 4 (I, Y): Avg. confusion matrix for transition days

	Labels								
	BM	HFA	HNA	HNFA	NEA	SEA	other	$\sum$	Precision
BM	142.00	1.33	0.67	0.33	1.33	11.00	379.00	535.67	0.26
HFA	2.00	19.00	0.33	5.00	2.67	10.33	29.00	68.33	0.29
HNA	6.67	1.00	10.67	3.67	1.33	0.67	76.33	100.33	0.07
HNFA	2.00	11.33	16.33	18.00	2.33	4.33	111.67	166.00	0.11
NEA	4.33	0.67	0.00	1.00	0.67	1.00	22.33	30.00	0.07
SEA	3.00	7.33	0.33	2.33	0.00	12.00	51.67	76.67	0.14
other	298.33	49.67	41.67	25.67	18.67	33.33	2207.67	2675.00	0.83
$\sum$	458.33	90.33	70.00	56.00	27.00	72.67	2877.67	-	-
Recall	0.31	0.21	0.18	0.28	0.02	0.15	0.77	-	-

Table 19: Model 5 (I, Y, S): Avg. confusion matrix

	Labels								
	BM	HFA	HNA	HNFA	NEA	SEA	other	$\sum$	Precision
BM	46.00	0.67	0.67	0.00	0.67	4.67	34.00	86.67	0.52
HFA	1.67	4.33	0.33	1.33	1.33	3.33	4.33	16.67	0.30
HNA	2.67	0.67	4.33	1.00	0.67	0.67	11.33	21.33	0.18
HNFA	1.33	2.33	4.67	4.33	0.67	1.33	11.33	26.00	0.17
NEA	1.00	0.33	0.00	0.33	0.00	0.67	3.00	5.33	0.00
SEA	1.67	2.33	0.33	0.67	0.00	3.67	5.00	13.67	0.19
other	105.67	18.00	15.00	9.00	6.67	14.33	162.67	331.33	0.49
$\sum$	160.00	28.67	25.33	16.67	10.00	28.67	231.67	-	-
Recall	0.30	0.15	0.17	0.26	0.00	0.11	0.71	-	-

Table 20: Model 5 (I, Y, S): Avg. confusion matrix for transition days

	Labels								
	BM	HFA	HNA	HNFA	NEA	SEA	other	$\sum$	Precision
BM	329.00	2.33	0.67	0.00	0.33	2.33	62.00	396.67	0.83
HFA	6.33	69.33	1.00	2.67	1.67	5.67	45.33	132.00	0.56
HNA	1.67	0.67	53.67	0.67	0.33	0.67	23.67	81.33	0.66
HNFA	1.33	2.00	1.00	45.00	0.00	0.00	30.00	79.33	0.53
NEA	3.33	1.00	0.67	0.33	20.00	0.67	22.00	48.00	0.51
SEA	1.67	2.00	0.67	0.33	0.33	43.00	20.00	68.00	0.67
other	115.00	13.00	12.33	7.00	4.33	20.33	2674.67	2846.67	0.94
$\sum$	458.33	90.33	70.00	56.00	27.00	72.67	2877.67	-	-
Recall	0.73	0.77	0.77	0.80	0.74	0.61	0.93	-	-

Table 21: Model 7 (I, L, L*Le): Avg. confusion matrix on the known-predecessor test sets

	Labels								
	BM	HFA	HNA	HNFA	NEA	SEA	other	$\sum$	Precision
BM	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
HFA	275.00	63.00	50.67	42.67	22.33	47.67	1911.67	2413.00	0.02
HNA	0.00	0.00	0.00	0.67	0.00	0.00	0.67	1.33	0.00
HNFA	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
NEA	158.33	25.33	19.33	12.67	4.67	23.67	829.00	1073.00	0.00
SEA	0.00	0.00	0.00	0.00	0.00	0.00	1.33	1.33	0.00
other	25.00	2.00	0.00	0.00	0.00	1.33	135.00	163.33	0.76
$\sum$	458.33	90.33	70.00	56.00	27.00	72.67	2877.67	-	-
Recall	0.00	0.67	0.00	0.00	0.22	0.00	0.05	-	-

Table 22: Model 7 (I, L, L*Le): Avg. confusion matrix on the unknown-predecessor test sets

	Labels								
	BM	HFA	HNA	HNFA	NEA	SEA	other	$\sum$	Precision
BM	64.00	2.33	0.67	0.00	0.33	2.33	45.00	114.67	0.56
HFA	3.67	13.67	1.00	2.67	1.00	3.33	13.33	38.67	0.38
HNA	1.33	0.67	13.67	0.33	0.00	0.33	13.00	29.33	0.46
HNFA	0.33	1.00	1.00	8.33	0.00	0.00	7.67	18.33	0.44
NEA	2.33	1.00	0.00	0.33	5.00	0.33	7.00	16.00	0.35
SEA	1.33	1.00	0.67	0.33	0.33	10.33	7.00	21.00	0.50
other	87.00	9.00	8.33	4.67	3.33	12.00	138.67	263.00	0.52
$\sum$	160.00	28.67	25.33	16.67	10.00	28.67	231.67	-	-
Recall	0.42	0.48	0.54	0.50	0.50	0.37	0.59	-	-

Table 23: Model 7 (I, L, L*Le): Avg. confusion matrix for transition days on the known-predecessor test sets

	Labels								
	BM	HFA	HNA	HNFA	NEA	SEA	other	$\sum$	Precision
BM	325.67	1.33	1.33	0.67	0.33	5.33	77.33	412.00	0.81
HFA	8.67	75.67	1.67	3.00	1.00	2.00	48.67	140.67	0.54
HNA	1.67	0.67	49.00	0.67	0.33	0.67	32.00	85.00	0.57
HNFA	1.33	2.00	1.67	45.00	0.33	0.33	40.67	91.33	0.45
NEA	2.00	0.67	0.00	0.00	17.00	0.67	16.00	36.33	0.53
SEA	2.67	0.67	1.00	0.67	0.00	37.67	20.00	62.67	0.64
other	116.33	9.33	15.33	6.00	8.00	26.00	2643.00	2824.00	0.94
$\sum$	458.33	90.33	70.00	56.00	27.00	72.67	2877.67	-	-
Recall	0.71	0.83	0.69	0.76	0.59	0.55	0.92	-	-

Table 24: Model 8 (I, L, Y): Avg. confusion matrix on the known-predecessor test sets

	Labels								
	BM	HFA	HNA	HNFA	NEA	SEA	other	$\sum$	Precision
BM	40.33	0.67	0.33	0.67	2.00	4.67	176.67	225.33	0.25
HFA	8.33	46.67	4.33	18.00	7.33	14.33	92.67	191.67	0.26
HNA	6.67	0.33	17.33	4.33	1.00	1.00	72.33	103.00	0.20
HNFA	10.00	4.33	11.00	19.33	1.33	1.67	93.33	141.00	0.10
NEA	16.33	5.33	4.67	2.67	1.00	3.00	42.33	75.33	0.01
SEA	7.00	3.67	0.33	0.33	0.00	15.33	44.33	71.00	0.19
other	369.67	29.33	32.00	10.67	14.33	32.67	2356.00	2844.67	0.83
$\sum$	458.33	90.33	70.00	56.00	27.00	72.67	2877.67	-	-
Recall	0.10	0.49	0.24	0.28	0.03	0.27	0.82	-	-

Table 25: Model 8 (I, L, Y): Avg. confusion matrix on the unknown-predecessor test sets

	Labels								
	BM	HFA	HNA	HNFA	NEA	SEA	other	$\sum$	Precision
BM	67.67	1.33	0.67	0.67	0.33	3.67	48.00	122.33	0.57
HFA	5.67	16.33	1.67	2.33	0.67	2.00	13.33	42.00	0.39
HNA	1.00	0.67	10.67	0.33	0.33	0.33	11.00	24.33	0.44
HNFA	0.33	1.00	1.00	8.33	0.33	0.00	8.67	19.67	0.40
NEA	1.00	0.67	0.00	0.00	4.33	0.33	4.00	10.33	0.44
SEA	1.33	0.67	1.00	0.67	0.00	8.00	5.67	17.33	0.47
other	83.00	8.00	10.33	4.33	4.00	14.33	141.00	265.00	0.53
$\sum$	160.00	28.67	25.33	16.67	10.00	28.67	231.67	-	-
Recall	0.43	0.57	0.43	0.49	0.40	0.32	0.61	-	-

Table 26: Model 8 (I, L, Y): Avg. confusion matrix for transition days on the known-predecessor test sets

	Labels								
	BM	HFA	HNA	HNFA	NEA	SEA	other	$\sum$	Precision
BM	124.00	8.00	0.33	1.00	1.33	7.67	238.00	380.33	0.39
HFA	0.33	16.33	0.00	4.00	1.00	7.67	17.00	46.33	0.37
HNA	1.33	0.00	4.00	0.00	0.00	0.00	10.67	16.00	0.24
HNFA	1.67	12.33	14.00	22.00	1.33	1.33	67.67	120.33	0.16
NEA	2.67	2.67	0.00	1.00	0.67	0.33	17.33	24.67	0.17
SEA	0.33	1.33	0.00	0.33	0.00	2.00	11.67	15.67	0.13
other	328.00	49.67	51.67	27.67	22.67	53.67	2515.33	3048.67	0.83
$\sum$	458.33	90.33	70.00	56.00	27.00	72.67	2877.67	-	-
Recall	0.27	0.16	0.06	0.32	0.02	0.02	0.88	-	-

Table 27: Model 9 (I, I-1): Avg. confusion matrix

	Labels								
	BM	HFA	HNA	HNFA	NEA	SEA	other	$\sum$	Precision
BM	39.67	4.00	0.33	0.33	0.67	3.33	26.33	74.67	0.60
HFA	0.33	3.67	0.00	0.33	0.33	1.33	2.67	8.67	0.42
HNA	1.00	0.00	1.00	0.00	0.00	0.00	0.67	2.67	0.33
HNFA	1.00	3.33	5.33	4.00	0.67	0.67	8.33	23.33	0.16
NEA	0.67	0.67	0.00	0.67	0.00	0.33	1.00	3.33	0.00
SEA	0.33	1.00	0.00	0.33	0.00	1.00	2.33	5.00	0.17
other	117.00	16.00	18.67	11.00	8.33	22.00	190.33	383.33	0.50
$\sum$	160.00	28.67	25.33	16.67	10.00	28.67	231.67	-	-
Recall	0.24	0.12	0.04	0.21	0.00	0.03	0.82	-	-

Table 28: Model 9 (I, I-1): Avg. confusion matrix for transition days

	Labels								
	BM	HFA	HNA	HNFA	NEA	SEA	other	$\sum$	Precision
BM	125.33	1.33	0.33	0.33	1.00	5.33	143.67	277.33	0.45
HFA	13.67	33.00	2.33	6.00	3.00	14.67	66.33	139.00	0.27
HNA	29.67	3.00	44.67	12.00	4.33	5.00	236.67	335.33	0.16
HNFA	2.67	5.67	3.00	18.67	3.00	3.00	41.33	77.33	0.13
NEA	30.33	11.67	1.33	1.67	6.33	1.00	102.00	154.33	0.05
SEA	8.67	19.33	1.33	8.67	0.00	24.67	91.33	154.00	0.16
other	248.00	16.33	17.00	8.67	9.33	19.00	2196.33	2514.67	0.87
$\sum$	458.33	90.33	70.00	56.00	27.00	72.67	2877.67	-	-
Recall	0.27	0.36	0.65	0.21	0.23	0.32	0.76	-	-

Table 29: Model 10 (I, I+1): Avg. confusion matrix

	Labels								
	BM	HFA	HNA	HNFA	NEA	SEA	other	$\sum$	Precision
BM	36.33	1.33	0.00	0.33	0.33	2.67	18.33	59.33	0.61
HFA	5.33	6.33	1.33	1.67	1.00	5.33	11.67	32.67	0.20
HNA	11.33	1.67	14.33	5.00	1.33	3.33	28.33	65.33	0.23
HNFA	1.00	1.67	1.33	3.67	0.67	0.33	5.33	14.00	0.15
NEA	9.67	3.33	0.67	0.33	1.67	1.00	14.67	31.33	0.06
SEA	2.67	7.00	0.67	2.33	0.00	6.67	8.00	27.33	0.20
other	93.67	7.33	7.00	3.33	5.00	9.33	145.33	271.00	0.54
$\sum$	160.00	28.67	25.33	16.67	10.00	28.67	231.67	-	-
Recall	0.22	0.21	0.56	0.16	0.17	0.22	0.63	-	-

Table 30: Model 10 (I, I+1): Avg. confusion matrix for transition days

	Labels								
	BM	HFA	HNA	HNFA	NEA	SEA	other	$\sum$	Precision
BM	37.33	0.00	0.00	0.00	0.00	2.67	35.67	75.67	0.49
HFA	6.67	14.33	0.00	3.67	1.33	6.67	16.33	49.00	0.36
HNA	20.33	1.67	21.00	5.33	4.00	1.33	125.67	179.33	0.18
HNFA	2.33	8.00	4.33	6.00	1.00	0.00	38.33	60.00	0.23
NEA	7.67	7.00	1.00	2.00	3.67	1.00	22.00	44.33	0.08
SEA	4.33	15.00	3.00	7.33	0.33	23.67	54.67	108.33	0.20
other	379.67	44.33	40.67	31.67	16.67	37.33	2585.00	3135.33	0.82
$\sum$	458.33	90.33	70.00	56.00	27.00	72.67	2877.67	-	-
Recall	0.08	0.16	0.33	0.12	0.13	0.31	0.90	-	-

Table 31: Model 11 (I, I+1, I-1): Avg. confusion matrix

	Labels								
	BM	HFA	HNA	HNFA	NEA	SEA	other	$\sum$	Precision
BM	8.00	0.00	0.00	0.00	0.00	1.33	5.00	14.33	0.69
HFA	1.33	3.00	0.00	0.00	0.33	2.67	3.00	10.33	0.39
HNA	7.67	1.00	7.67	1.67	1.33	0.33	13.67	33.33	0.55
HNFA	1.33	2.33	1.33	1.00	0.67	0.00	3.67	10.33	0.29
NEA	3.33	0.00	0.00	1.00	0.33	0.33	3.00	8.00	0.03
SEA	1.33	4.00	1.00	1.67	0.00	7.67	7.67	23.33	0.33
other	137.00	18.33	15.33	11.33	7.33	16.33	195.67	401.33	0.49
$\sum$	160.00	28.67	25.33	16.67	10.00	28.67	231.67	-	-
Recall	0.05	0.10	0.29	0.07	0.04	0.25	0.85	-	-

Table 32: Model 11 (I, I+1, I-1): Avg. confusion matrix for transition days

References

Christoph Bergmeir, Rob J. Hyndman, and Bonsoo Koo. A note on the validity of cross-validation for evaluating autoregressive time series prediction. *Computational Statistics Data Analysis*, 120:70–83, 2018. ISSN 0167-9473. doi: https://doi.org/10.1016/j.csda.2017.11.003. URL https://www.sciencedirect.com/science/article/pii/S0167947317302384.

James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.*, 13(null):281–305, feb 2012. ISSN 1532-4435.

Bernd Bischl, O Mersmann, Heike Trautmann, and Claus Weihs. Resampling methods for meta-model validation with recommendations for evolutionary computation. *Evolutionary computation*, 20:249–75, 02 2012. doi: 10.1162/EVCO_a_00069.

Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006. ISBN 0387310738.

François Chollet. Training amp; evaluation with the built-in methods, Mar 2023. URL https://keras.io/.

Ludwig Fahrmeir, Thomas Kneib, Stefan Lang, and Brian Marx. *Categorical Regression Models*, pages 325–347. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013a. ISBN 978-3-642-34333-9. doi: 10.1007/978-3-642-34333-9_6. URL https://doi.org/10.1007/978-3-642-34333-9_6.

Ludwig Fahrmeir, Thomas Kneib, Stefan Lang, and Brian Marx. *Structured Additive Regression*, pages 535–595. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013b. ISBN 978-3-642-34333-9. doi: 10.1007/978-3-642-34333-9_9. URL https://doi.org/10.1007/978-3-642-34333-9_9.

Davide Faranda, Salvatore Pascale, and Burak Bulut. Persistent anticyclonic conditions and climate change exacerbated the exceptional 2022 european-mediterranean drought. *Environmental Research Letters*, 18, 02 2023. doi: 10.1088/1748-9326/acbc37.

Alberto Fernández, Salvador García, Mikel Galar, Ronaldo C. Prati, Bartosz Krawczyk, and Francisco Herrera. *Foundations on Imbalanced Classification*, pages 19–46. Springer International Publishing, Cham, 2018. ISBN 978-3-319-98074-4. doi: 10.1007/978-3-319-98074-4_2. URL https://doi.org/10.1007/978-3-319-98074-4_2.

Cèsar Ferri, Jose Hernandez-Orallo, and R. Modroiu. An experimental comparison of performance measures for classification. *Pattern Recognition Letters*, 30:27–38, 01 2009. doi: 10.1016/j.patrec.2008.08.010.

Konstantinos Fokianos and Benjamin Kedem. Regression theory for categorical time series. *Statistical Science*, 18, 08 2003. doi: 10.1214/ss/1076102425.

Konstantinos Fokianos and Lionel Truquet. On categorical time series models with covariates. *Stochastic Processes and their Applications*, 129(9):3446–3462, 2019. ISSN 0304-4149. doi: https://doi.org/10.1016/j.spa.2018.09.012. URL https://www.sciencedirect.com/science/article/pii/S0304414918305325.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

Margherita Grandini, Enrico Bagli, and Giorgio Visani. Metrics for multi-class classification: an overview, 2020.

Hansika Hewamalage, Klaus Ackermann, and Christoph Bergmeir. Forecast evaluation for data scientists: common pitfalls and best practices. *Data Mining and Knowledge Discovery*, 37(2):788–832, Mar 2023. ISSN 1573-756X. doi: 10.1007/s10618-022-00894-5. URL https://doi.org/10.1007/s10618-022-00894-5.

Manu Joseph. *Modern time series forecasting with python explore industry-ready time series forecasting using modern machine learning and Deep Learning*. Packt Publishing Limited, 2022.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

Nadja Klein, Thomas Kneib, and Stefan Lang. Bayesian generalized additive models for location, scale, and shape for zero-inflated and overdispersed count data. *Journal of the American Statistical Association*, 110(509):405–419, 2015. URL https://EconPapers.repec.org/RePEc:taf:jnlasa:v:110:y:2015:i:509:p:405-419.

Matja Kukar and Igor Kononenko. Cost-sensitive learning with neural networks. In *European Conference on Artificial Intelligence*, 1998.

Yunjie Liu, Evan Racah, Prabhat, Joaquin Correa, Amir Khosrowshahi, David Lavers, Kenneth Kunkel, Michael Wehner, and William Collins. Application of deep convolutional neural networks for detecting extreme weather in climate datasets, 2016.

Magdalena Mittermeier, Maximilian Weigert, and David Rügamer. Identifying the atmospheric drivers of drought and heat using a smoothed deep learning approach. *CoRR*, abs/2111.05303, 2021. URL https://arxiv.org/abs/2111.05303.

Juri Opitz and Sebastian Burst. Macro f1 and macro f1, 2021.

Randy L. Ribler. Visualizing categorical time series data with applications to computer and communications network traces. 1998.

David Rügamer, Chris Kolb, Cornelius Fritz, Florian Pfisterer, Philipp Kopper, Bernd Bischl, Ruolin Shen, Christina Bukas, Lisa Barros de Andrade e Sousa, Dominik Thalmeier, Philipp Baumann, Lucas Kook, Nadja Klein, and Christian L. Müller. deepregression: a flexible neural network framework for semi-structured deep distributional regression. *Journal of Statistical Software*, 2022. Accepted.

David Rügamer, Chris Kolb, and Nadja Klein. Semi-structured distributional regression. *The American Statistician*, 0(0):1–12, 2023. doi: 10.1080/00031305.2022.2164054. URL https://doi.org/10.1080/00031305.2022.2164054.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 06 2014.

Leonard J. Tashman. Out-of-sample tests of forecasting accuracy: an analysis and review. *International Journal of Forecasting*, 16(4):437–450, 2000. ISSN 0169-2070. doi: https://doi.org/10.1016/S0169-2070(00)00065-0. URL https://www.sciencedirect.com/science/article/pii/S0169207000000650. The M3- Competition.

Christian Weiß. *Analyzing Categorical Time Series*, chapter 6, pages 121–132. John Wiley Sons, Ltd, 2018. ISBN 9781119097013. doi: https://doi.org/10.1002/9781119097013.ch6. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119097013.ch6.

P. Werner and F. Gerstengarbe. Katalog der grosswetterlagen europas (1881-2009). *PIK Report*, pages 1–146, 01 2010.

Simon N. Wood. *Generalized additive models : an introduction with R*. Texts in statistical science. Chapman & Hall/CRC, Boca Raton, FL, 2006. ISBN 1584884746; 9781584884743.

Declaration of authorship

I hereby declare that the report submitted is my own unaided work. All direct or indirect sources used are acknowledged as references. I am aware that the Thesis in digital form can be examined for the use of unauthorized aid and in order to determine whether the report as a whole or parts incorporated in it may be deemed as plagiarism. For the comparison of my work with existing sources I agree that it shall be entered in a database where it shall also remain after examination, to enable comparison with future Theses submitted. Further rights of reproduction and usage, however, are not granted here. This paper was not previously presented to another examination board and has not been published.

Munich, 02.08.2023

Laetitia Frost