Analysis of Extreme Weather Events
ETH Zurich - AI4Good 2022

Ying Xue
yinxue@ethz.ch

Emilia Arens
earens@ethz.ch

Katarzyna Krasnopolska
krasnopk@ethz.ch

Alexis Tabin
atabin@ethz.ch

Amalie Kjaer
akjaer@ethz.ch

Abstract

This project builds on the work of Prabhat et. al. [7] to predict the occurrence of extreme weather events (tropical cyclones (TC) and atmospheric rivers (AR)). This is achieved by segmenting expert-labelled world maps with 16 types of atmospheric data measurements (such as zonal and meridional winds and precipitation rates) for each latitude-longitude pair. The main challenge of the segmented dataset is its high class imbalance. Firstly, we assess the performance of multiple neural network architectures (namely DeepLabv3++ [3], UPerNet [17], U-net [12] and CGNet [16]) on the segmentation task, and conclude that CGNet and U-net offer the best performance: CGNet yields the best mean performance (with the highest achieved IoU score of 0.941 on background (BG) segmentation) while U-net yields the best performance on TC and AR segmentation (with IoU scores of 0.359 and 0.404 respectively). Secondly, we explore Curriculum Learning (CL) as an alternative approach to the segmentation task and conclude that the best performance is achieved on larger image patches (128 x 128) using either U-net or DeepLabv3++ (depending on curriculum choice). The first curriculum (CL I) outperformed both the baseline's mean IoU (achieving a score of 0.5353) and TC IoU (achieving a score of 0.3162). Overall, all curricula explored proved more performant than the baseline in detecting a specific class, at the cost of offering lower performance on other classes. [1]

1. Introduction

In the context of climate change, it is of paramount importance to understand how a changing climate may impact the intensity and frequency of occurrence of extreme weather events (such as tropical cyclones and atmospheric rivers). Tropical cyclones (TCs, also known as hurricanes) are rotating storms that lead to high-speed winds and heavy rainfalls, often causing extreme destruction. Atmospheric rivers (ARs) are atmospheric regions that carry large amounts of water vapour, causing strong winds and extreme rainfalls. Gaining a better understanding and predictive power of such events could help examine the environmental impacts of climate change and implement adequate preventive measures.

Historically, climate scientists have developed a range of heuristics to better understand how climate patterns affect extreme weather events. However, the different methods developed yield significantly different results. Furthermore, heuristic-based methods cannot be applied blindly to different datasets since they often require tuning of thresholds.

In order to overcome high uncertainty in predictions and threshold tuning from different heuristics, this project aims to learn the association between weather patterns and the occurrence of TCs and ARs using deep learning (DL), which has been shown to be successful in related tasks of pattern recognition. It extends the baseline proposed by Prabhat et. al. [7] and explores curriculum learning (CL) as an alternative method to predict the occurrence of TCs and ARs more accurately. The ground truth data was labelled by experts as described by Prabhat et. al. [7], and is available from https://portal.nersc.gov/project/ClimateNet/. The main challenge presented by this dataset is the large class imbalance (93.7% background-labelled pixels, 5.8% AR-labelled pixels and 0.5% TC-labelled pixels).

2. Related Work

Heuristics-based Models Numerous heuristics-based methods have been developed to detect extreme weather events, of which the most commonly used algorithms were summarised and compared by Neu et. al. [9]. These methods present two main disadvantages, namely:

1. A high uncertainty in predictions (particularly for less extreme or fast-moving events). For example, one

[1] Code available from https://github.com/earens/ClimateNet_AI4Good

study found that the number of instances of ARs could vary up to a factor of 10 depending on the heuristics used [13]),

2. The specificity of the algorithm. In other words, the necessity to tune model thresholds specifically to each dataset studied (to be contrasted with inference using weights from neural nets).

Baseline DL Model ClimateNet [7] is the baseline against which the results of this project are compared. The baseline model segments input world maps with atmospheric data (16 features for each longitude-latitude pair on a 25km square grid) into the background, TC and AR labels. The ground truth data is expert-labelled (as opposed to labelled using heuristics-based algorithms). The neural network architecture used is DeepLabv3++ [3] with cross-entropy loss, trained on the following features: TMQ (total vertically integrated precipitable water), U850 (zonal wind at 850 mbar pressure surface), V850 (meridional wind at 850 mbar pressure surface), and PRECT (total convective and large-scale precipitation rate). We refer to this feature subset as Group I. Baseline per-class results are summarised in terms of intersection-over-union (IoU) in Tab. 4.

3. Method

3.1. Extending the baseline

3.1.1 Feature Selection

The dataset presents 16 features for each latitude-longitude pair on the world map. In order to reduce computational cost and improve model performance, the set of features considered was reduced to 4. Two methods were used for feature selection, yielding different sets of features of interest. Analysis of variance (ANOVA) yielded TMQ, U850, UBOT and VBOT (Group II), while maximum mutual information yielded QREFHT, PRECT, Z200 and ZBOT (Group III). Each set of features was then evaluated using DeepLabv3++ to compare performance with the baseline model. Results are shown in Tab. 1.

3.1.2 Model

After selecting the features, we conducted a series of experiments with different models designed for semantic segmentation.

Our first experiment was to reproduce the results from the paper that introduced the ClimateNet dataset [10]. The model they used the DeepLabv3++ model. DeepLabv3++ is a model for semantic image segmentation that uses atrous convolution to capture context at multiple scales and improve performance. Atrous convolution allows the model to adjust the field-of-view of filters and control the resolution of feature responses, which can be useful for segmenting objects at different scales. The model also includes an Atrous Spatial Pyramid Pooling module that probes convolutional features at multiple scales and uses image-level features to encode global context. We obtained similar results to the one presented in the paper, and thus, we used it as our baseline.

In our second experiment, we tried the UPerNet [17] architecture. UPerNet is a multi-task network for the task of Unified Perceptual Parsing. It is designed to recognize as many visual concepts as possible from a given image and is trained using a multi-task framework and a specific training strategy. UPerNet is able to segment a wide range of concepts from images and can also be used to discover visual knowledge in natural scenes. We chose it for its training easiness as well as its promising performance on heterogeneous tasks. But on our task, UPerNet failed to detect TCs, so we did not experiment further.

After this failure, we decided to try a more traditional architecture well-known for semantic segmentation tasks: U-net [12]. Its architecture is known for its ability to handle large class imbalances and its efficiency at processing high-resolution images. U-net consists of an encoder network and a decoder network, which are connected by a series of skip connections. We used a pre-trained ResNet [6] encoder to extract feature maps from the input image. The skip connections between the encoder and decoder allow the network to propagate information from the encoder to the decoder, which can improve the network's performance and reduce the amount of information that needs to be processed by the decoder. It showed great performance on the Climatenet dataset but did not outperform the baseline consistently.

Later on, we found that some of the authors of the original paper [7] worked on an extension of their work. They implemented a faster and more scalable approach using a light-weighted network called CGNet [16]. The idea behind CGNet is based on an observation of the human visual system. It learns a joint feature of both the local features and surrounding context features in the encoder phase. The joint feature is then refined with a weight vector that represents the global contextual information.

In summary, our results suggest that U-net and CGNet are strong choices for the semantic segmentation of atmospheric rivers and tropical cyclones. The ability of CGNet to consider the surrounding context of each pixel, in addition to the local features, makes it well-suited for distinguishing between similar classes and handling challenging cases. While UPerNet and U-net did not always perform as well in our experiments, CGNet unfailingly demonstrated strong performance while being lighter than DeepLabv3++, making it our preferred model for our following experi-

ments. Details about the results and performance of the different architectures are provided in Sec. 4.1.2.

3.1.3 Loss

Our model suffers from strong data imbalance, as there are around 94% of the pixels in the ClimateNet data corresponding to the "background" class [7]. To address this issue, Jaccard loss has been used in the original paper [7] to train the neural network, and IoU (intersection over union) has been used to evaluate the performance of the model [11]. To compare how different losses can affect the training process, we train the CGNet, which has shown strong performance, using four different loss functions: Jaccard loss, dice loss, weighted cross-entropy, and unweighted cross entropy [2].

The Jaccard index is the intersection of the predicted and ground truth class divided by the union of the predicted and ground truth class. Jaccard loss is typically used when the classes are imbalanced, as it is less sensitive to class imbalance than other losses (e.g. cross-entropy loss). Jaccard index can directly optimize the segmentation metric (IoU). Similar to Jaccard loss, dice loss is also widely used to address the data imbalance problem. It is first introduced in 1945 [5]. The dice coefficient is the intersection of the predicted and ground truth class divided by the sum of their sizes. Cross entropy loss is a commonly used loss function for semantic segmentation. It is defined as the negative log probability of the correct class. This loss is sensitive to class imbalance, as it naturally gives more weight to the more frequent classes. While weighted cross-entropy is a variant of cross-entropy loss, where each class is assigned a weight that reflects the importance of that class, the weight for each class is inversely proportional to the frequency of the class in the dataset. This way, the model will be penalized more for making mistakes on underrepresented classes.

We train the CGNet with different losses for 15 - 20 epochs over the training set using the Adam optimizer [8]. The performance of different losses are compared in Sec. 4.1.3.

3.2. Tackling the complexity - Curriculum Learning

As pointed out in Sec. 1, one of the main challenges in the segmentation task at hand is significant class imbalance. To address this problem, Kashinath et al. [7] suggested the application of CL. This learning strategy is strongly inspired by human learning, which shows that breaking down the task into less complex subtasks and gradually increasing the difficulty while learning can have positive effects [1].

In the context of machine learning, this translates to designing a schedule for the data that is provided to the model during training. Based on this schedule, the training would start out with less complexity e.g. easy samples or a subset of classes and slowly merge into vanilla training with random examples from all classes in all difficulties. The scheduler thereby determines the choice of samples per training round as well as the speed at which the training progressively becomes harder [14].

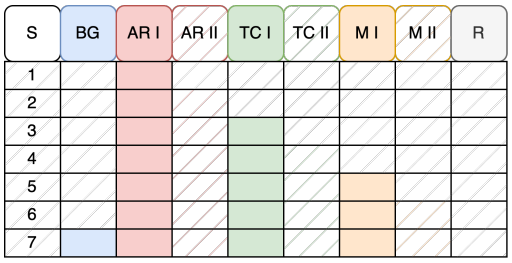

Figure 1. Illustration of CL I. The first column displays stages 1 to 7. Each box that is coloured indicates, that the corresponding group is included during the respective training stage.

3.2.1 Setup

Mapping this idea to the provided segmentation task initially requires difficulty ranking among the samples. As we specifically want to alleviate the problem of class imbalance, the difficulty here refers to the complexity of the three classes, where we consider the difficulty to increase with decreasing representation. This approach, however, implies that samples contain different distributions of labels, meaning that some only contain background, others background and TCs and again others all three classes. As a consequence, we can no longer feed entire maps during training but patches of that map containing different class distributions.

Specifically, we define seven groups of label distributions with increasing complexity: BG solely (BG), BG and many AR pixels (AR I), BG and at least one AR pixel (AR II), BG and many TC pixels (TC I), BG and at least one TC pixel (TC II), all three classes with many AR and TC pixels (M I) and all three classes (M II). Based on these groups, we then construct a schedule for training the model. In line with the intuition behind CL we thereby start out with a subset of the groups that are considered to be less complex and slowly integrate more complex label distributions over time. We refer to each change in input distributions as stages.

Concretely, three different types of curricula are evaluated: CL I (Fig. 1)contains a simple build-up of the predefined seven groups by difficulty.

BG patches are introduced last as training on them solely would be trivial but including them among other samples leads to a strong distribution shift which potentially complicates the task. CL II is more fine-grained and trains the binary subtasks BG vs. AR and BG vs. TC first before merging all classes together. Lastly, CL III resembles CL

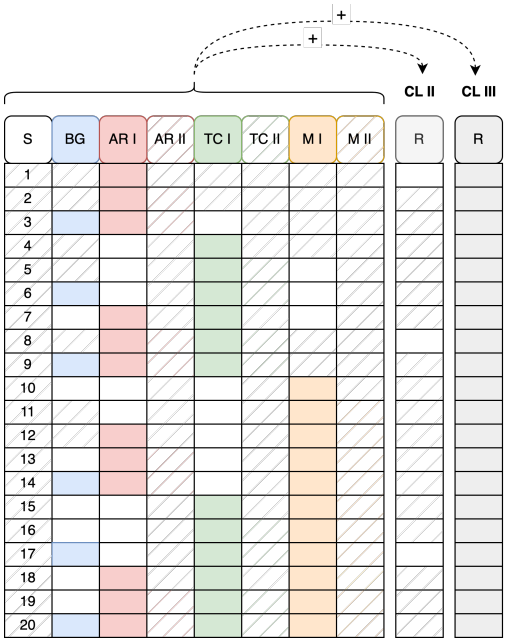

Figure 2. Stages for CL II and CL III. CL III resembles CL II but has some randomness in each stage to smoothen transitions.

II but introduces some randomness to each stage in order to smoothen the transition between stages. Both curricula are displayed in Fig. 2.

Given a schedule, its corresponding dataset is generated. For each stage, all training and validation images are partitioned into squared patches where the patch size and stride are adjustable hyperparameters. Then, each group occurring in that stage gets assigned a fixed amount N of patches from each image where the patches are drawn from the set of patches fulfilling the group criteria. Here, groups I and II for ARs, TCs, and M differ solely in the drawing procedure: Whereas in group II N patches are drawn randomly, in group I, the patches are sorted by their BG-AR/BG-TC/BG-M ratio and the N lowest ones are chosen, hence those containing the highest amount of non-background pixels. Fig. 3 displays this extraction procedure.

3.2.2 Model and Hyperparameters

As CL requires restructuring the training procedure, all experiments related to CL are based on a custom model instead of the pipeline provided by Kashinath et al. [7]. Even though this seemingly hinders comparability, the model at hand achieves competitive to outperforming results on the base task. At the core, we use Torchgeo [15], which provides support for pretrained U-net and DeepLabv3++ segmentation architectures [3] with various backbones. The hyperparameters are tuned for each task respectively, as sequentially tuning (full image - patches -CL) was not applicable given the immense distribution shifts between the steps.

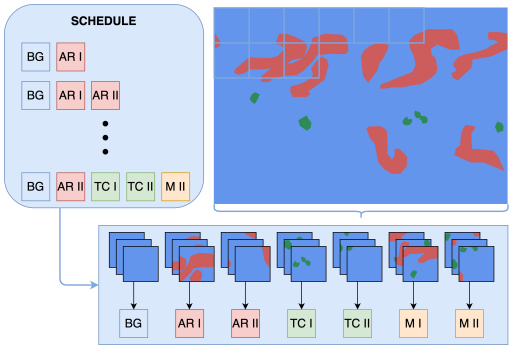

Figure 3. Visualization of the patching process. Given a schedule, at each stage from all patches of the image, N are drawn per group of interest. Groups are separated by the contained label distributions.

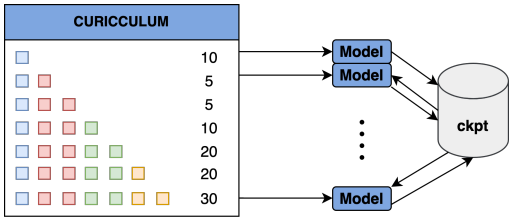

Figure 4. Implementation of CL. The model reads in the plan for the stage at hand, trains for a given amount of epochs and writes the resulting weights to a checkpoint logger. It then continues with the following stage, reads in the weights from the checkpoint folder and continues training.

3.2.3 Training

The introduction of multiple stages has two effects on the training: The data samples and the number of epochs change with entering a new stage during training. To realize this, suitable datasets for each step are extracted before the training starts and stored in separate folders. In the first stage, the model is then trained in a conventional manner, reading the corresponding data and optimizing for the desired amount of epochs. After finishing training, the weights are saved externally. Then, for each following stage, the dataset is exchanged according to the schedule, the exported weights are reloaded, the model trains and the updated weights override the previous ones. Fig. 4 visualizes this procedure.

Metrics are logged over all classes independent of the stage and hence gain expressiveness throughout the training procedure, as some classes might not be present in the beginning. Furthermore, the distributions in the training and validation set follow from the same operations. Therefore, the validation metrics provide insights into the performance

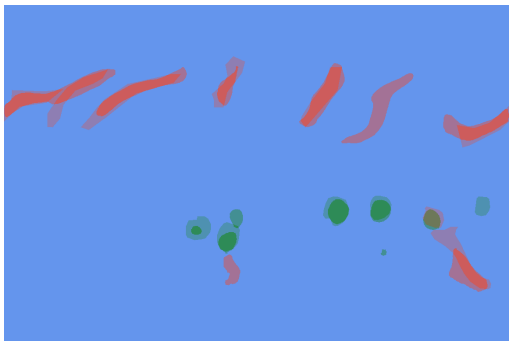

Figure 5. Quantitative example after training in the base settings. BG appears in blue, TCs appear in green and ARs in red. High opacity indicates that the ground truth and prediction coincide, whereas low opacity corresponds to wrongly predicted pixels.

on the so far provided distributions but do not necessarily represent the true performance on random patches from the image. This difference vanishes over the stages as more and more label distributions are included, slowly merging into the true distribution over the entire map.

3.2.4 Inference

For comparability of CL to the base approach [7] at inference, we evaluate the performance on every patch of the test set and provide the average Jaccard index. Details about hyperparameters and performance are provided in Sec. 4.2.

4. Results

4.1. Extending the baseline

4.1.1 Feature selection results

Being provided 16 channels with weather information, we decided to evaluate feature importance before tweaking the model. On all feature groups, we trained with CGNet. Tab. 1 displays the numerical results.

	Group I (Baseline)	Group II	Group III
BG IoU	**0.941**	0.94	0.31
TC IoU	**0.342**	0.31	0
AR IoU	**0.401**	0.42	0
Mean IoU	**0.564**	0.56	0.1

Table 1. Results for different feature selections.

This clearly reveals that the features chosen by experts outperform those that statistically seem relevant. Therefore, all following experiments are conducted on the features proposed by the baseline paper [7].

	DeepLabv3++	UPerNet	U-net	CGNet
BG IoU	0.938	0.936	**0.941**	**0.941**
TC IoU	0.283	0.	**0.359**	0.342
AR IoU	0.397	0.396	**0.404**	0.401
Mean IoU	0.542	0.443	**0.568**	0.564

Table 2. Results of the different model architectures.

4.1.2 Model selection results

As the DeepLabv3++ only obtained a mean IoU of 0.542, we had great hopes that we could outperform its performance by a certain margin. The performance of the different model architectures we tried is shown in table 4. We can clearly see how UPerNet failed to detect TCs. The U-net model achieved the best results overall, but CGNet had comparable outcomes. Even though CGNet was not the best-performing model overall, we decided to stick with it for further experiments. This choice was motivated by the comparison of their respective weights and how fast they were to train.

4.1.3 Loss Comparison

The performance of the CGNet with different losses is shown in the table 3. Surprisingly, CE (cross-entropy) loss achieves the best mean IoU, while weighted CE has the worst performance. Jaccard loss performs slightly better than dice loss, which is as expected since Jaccard loss optimizes the IoU metric directly. But when taking a closer look at the IoU for different classes, we can see that the high mean IoU of CE is due to the high background and AR IoU, but its ability to classify TC is not as good as dice or Jaccard loss.

CGNet	Jaccard Loss	Dice Loss	weighted CE	CE
BG IoU	0.942	0.922	0.857	**0.957**
TC IoU	0.348	**0.332**	0.226	0.312
AR IoU	0.403	0.377	0.272	**0.401**
Mean IoU	0.564	0.544	0.452	**0.584**

Table 3. Results of the different loss on CGNet

4.2. Curriculum Learning

As CL alters two components of the baseline training, namely feeding patches instead of full maps and changing the input distribution during training, we did not only conduct experiments with different curricula. For comparability purposes and in order to get a baseline for patch-based training, we also evaluated the performance of our model on the base task and on patched inputs.

	Baseline	Ours
BG IoU	0.9389	**0.9542**
TC IoU	0.2441	**0.4856**
AR IoU	**0.3910**	0.3383
Mean IoU	0.5247	**0.5927**

Table 4. Results of the base training. The custom model outperforms the baseline for BG and TCs but cannot achieve competitive results for ARs. On average, regardless, it scores higher.

4.2.1 Standard Training

As discussed in Sec. 3.2.2 we prepend experiments with the base model on the standard task without any patching or application of curricula. Due to limited resources, no extensive grid search was performed. However, the U-net model architecture consistently outperformed DeepLabv3++ and worked best in combination with a ResNet50 backbone. We enabled automatic learning rate detection as well as early stopping and trained with a batch size of two. All backbones were pretrained on ImageNet [4]. With these design choices, the model achieved competitive results compared to the baseline [7] and even outperformed precision for some classes. Numerical results are displayed in Tab. 4 and Fig. 5 provides a qualitative sample.

4.2.2 Patch based Training

Maintaining the previously defined architecture intact, we then experimented with patching hyperparameters. As the locality component is partially lost during this procedure and given distribution shifts between patches, we expect a drop in performance for vanilla training on patches. Hence, it is crucial to evaluate this new baseline to later quantify the effect of CL. The U-net architecture requires patch shapes that are divisible by 32. We further restricted the search space to squared patches only. Lastly, as the parameters are further constrained by memory and the need for distinct label distributions, we only evaluated the performance for patches between 128 and 224 pixels side length where multiple stride values are evaluated, especially for bigger patches. The model hyperparameters were not changed from the base task.

As expected, the performance decreases drastically with the introduction of patches. Particularly, the detection of TCs becomes impossible for the model and precision seems to correlate with patch size. Tab. 5 displays the performance for all classes where the patches were extracted with no overlap.

	128	160	192	224
BG IoU	0.8272	0.7093	**0.945**	0.9207
TC IoU	0	0	0	0
AR IoU	**0.3648**	0	0.2236	0.5514
Mean IoU	0.3973	0.2364	0.3895	**0.4907**

Table 5. Results of patch-based training. Patching the image drastically reduces performance. Seemingly, the class imbalance has an even broader impact, as TCs cannot be detected at all.

	Baseline	224 x 224	CL I
BG IoU	**0.9389**	0.9207	0.9324
TC IoU	0.2441	0	**0.3162**
AR IoU	0.3910	**0.5514**	0.3572
Mean IoU	0.5247	0.4907	**0.5353**

Table 6. Results for CL I. Introducing a curriculum positively impacts the detection abilities of the model. TCs are detected, and the overall performance slightly outperforms the main baseline.

4.2.3 Curriculum based Learning

Finally, the previous experiments merge into the application of CL. Even though the model performed best on patches with size 224x224 for the curriculum experiments, patches of size 128x128 were extracted with a stride of 90. This choice was mainly guided by trading-off performance and availability of distributions in the image (with increasing patch size, patches only containing BG and TCs become rare). From each image, five patches were drawn for each group, respectively, and the model was trained on each stage for three epochs. Only the last stage lasted for six epochs.

Furthermore, the accuracy benefited from using a less complex backbone given smaller input sizes. Hence, instead of training with ResNet50, all three curricula were applied to a ResNet18 backbone. Additionally, the batch size could be increased without reaching memory limits. Therefore, all three curricula were trained on a batch size of 16. Lastly, on CL I, DeepLabv3++ outperformed the U-net architecture and was therefore applied for experiments with CL I. Evaluating CL I supports the assumption that carefully structuring the training benefits the performance. In contrast to training on patches without providing a curriculum, the model is now able to detect TCs and even achieves competitive results as visible in Tab. 6.

The detection of TC increases through CL II and CL III as well, but generally, these more sensitive curricula do not lead to an improvement over CL I as displayed in Tab. 7. Interestingly, their positive effect on TCs comes at the cost of misinterpreting or missing near to all AR pixels. Potentially this loss is rooted in the more enhanced distribution shifts in CL II and CL III. Whereas CL I sequentially builds

	CL I	CL II	CL III
BG IoU	0.9324	0.934	**0.941**
TC IoU	0.3162	**0.3501**	0.3483
AR IoU	**0.3572**	0.1381	0.0855
Mean IoU	**0.5353**	0.4740	0.4583

Table 7. Results of CL II and CL III. CL II has a positive effect on TC detection, and CL II contributes to BG detection. However, their overall performance is worse than pure patch training due to close to zero correctly identified AR pixels.

up the curriculum by adding new distributions, CL II and III classes appear and disappear throughout the training, which requires the model to "memorize" previous stages.

5. Discussion

The light-weight CGNet allowed us to slightly improve the baseline, but with IoUs scores still lower than 60 per cent, it is not satisfactory. As we only have hundreds of expert-labelled images, models trained on this limited dataset cannot achieve outstanding performances.

Therefore, we look for other techniques to improve our model performance.

By introducing CL to the training procedure, we aim at not only tuning the parameters occurring during training but rather designing the entire learning procedure adapted to the obstacles at hand, namely significant class imbalance. This approach theoretically provides multiple benefits over vanilla training. Mainly, the curriculum allows us to directly address the imbalance in difficulty by slowly paving the path to be sensitive to underrepresented classes. However, there are more benefits arising from relying on patches. Clearly, the bottleneck of data sparsity is loosened as training on subsets of the entire world map increases the number of samples during training. As a consequence, oversampling of underrepresented classes and undersampling of overrepresented classes now arise as additional tools to address the problem.

The experimental results partially support these assumptions. On the one hand, achieving competitive performance is feasible with CL I, where TC detection is enhanced compared to the baseline at the cost of lower AR detection. On the other hand, training on patches solely reveals the importance of a locality component and broader context, as TC detection is impossible. This indicates that CL has enhancing capabilities but is rooted in a task that is generally harder. Potentially, including spatial encoding during training could alleviate this effect; thus, this approach will be considered in future work. Furthermore, the IoU variance among different curricula strengthens the assumption that the curriculum itself significantly influences the performance and, therefore, its tuning can be decisive. Consequently, our approach indicates the general potential of CL but possibly does not exploit it to its fullest yet.

Additionally, CL provides benefits with respect to computation as the reduced input size accelerates training. This, however, must be treated as a trade-off considering the possible increase in samples if no over- or undersampling is applied.

6. Conclusion

Climate change is one of the most pressing issues faced by humans. Therefore, understanding how a varying climate may affect extreme weather events is crucial. This project tackled the prediction of extreme weather events using deep learning based on expert-labelled atmospheric data maps. The same model architecture as presented by Prabhat et. al. [7] was used as a baseline model, and other methods were explored to obtain better segmentation performance. Firstly, feature selection showed that the best features for high IoU scores were consistent with those selected by Prabhat et. al. [7]. Secondly, model selection showed that the baseline model was outperformed by a U-net architecture, yielding higher IoU scores for all classes except AR. The low scores observed for the TC and AR classes (relative to BG scores) were mainly due to the highly imbalanced nature of the dataset. To overcome this imbalance, we finally explored curriculum learning as an alternative approach to predict the occurrence of extreme weather events with even higher IoU scores. The three main variable components in this approach were the following:

- The format of the input data, more precisely the size of the input patches. 224 x 224-sized patches showed the highest mean IoU, but 128 x 128-sized patches were used in the curriculum learning for data availability reasons.

- The network architecture and hyperparameters used. DeepLabv3++ showed the best results for curriculum I, while U-net showed the best results for the remaining two curricula.

- The curriculum followed. Three curricula were trialled. Nevertheless, more curricula could be explored.

Overall, tuning these three components above had a strong influence on the resulting IoU scores. In this data-sparse setting, this is due to the large difference between samples input to the network. We, therefore, conclude that this curriculum approach is already able to outperform the baseline model provided by Prabhat et. al. [7] and yields promising results. Regardless, this approach could benefit from additional tuning to guarantee high-accuracy predictions.

7. Future Work

This work indicates that extensive tuning and adaptation of the loss function, as well as restructuring the learning process, influences the performance positively, especially given limited data availability and high class imbalance among the samples. Still, the results revealed some weaknesses of our suggested methods and thereby give rise to possible future adaptations.

Hyperparameters: The introduction of CL comes at the cost of an increasing amount of hyperparameters that need to be tuned carefully. Especially for the design of curricula, tuning strategies are diverse while the results underline the importance of suitable stages. Thus, finding well suited curricula arguably is the main challenge in CL and needs further exploration and evaluation. Currently, two components seem to mainly hinder accuracy: strong distribution shifts between stages and a lack of 'memory' of the model to integrate earlier stages. We would therefore dedicate future work to these limitations first.

Difficulty metric in CL: Currently, we are associating difficulty with representation among samples which means that TCs with the lowest share among all pixel labels are considered difficult, whereas BG with the highest share is considered simple. However, other strategies could be considered as well. One example would be to sort samples by their loss value achieved through a separate model and then sort the data from low loss to high loss as suggested by [14].

Merging Results As both approaches addressing the task at hand yield promising results, it would be interesting to evaluate their combined effects. This implies building CL around a model with an optimized architecture asides DeepLabv3++ and U-net and a sensitive loss function.

Ensemble Methods Especially for CL, some results indicate that enhanced performance on TCs comes at the cost of a higher misclassification rate for ARs. Hence, the application of ensemble methods could be beneficial in order to decrease this performance variance and combine the strengths of different approaches.

Spatial Information Encoding Lastly, we already indicated that spatial information gets lost during training on patches and, thus, also during CL. As the performance is seemingly affected by this loss of information, reintroducing a spacial encoding might enhance performance.

References

[1] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48, 2009. 3

[2] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Semantic image segmentation with deep convolutional nets and fully connected crfs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3340–3348, 2017. 3

[3] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018. 1, 2, 4

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 6

[5] L.R. Dice. A method for establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on danish commons. *Biometrika*, 33(4):305–307, 1945. 3

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. 2

[7] Karthik Kashinath, Mayur Mudigonda, Sol Kim, Lukas Kapp-Schwoerer, Andre Graubner, Ege Karaismailoglu, Leo Von Kleist, Thorsten Kurth, Annette Greiner, Ankur Mahesh, et al. Climatenet: an expert-labeled open dataset and deep learning architecture for enabling high-precision analyses of extreme weather. *Geoscientific Model Development*, 14(1):107–124, 2021. 1, 2, 3, 4, 5, 6, 7

[8] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv e-prints*, page arXiv:1412.6980, Dec. 2014. 3

[9] Urs Neu, Mirseid Akperov, Nina Bellenbaum, R. Benestad, Richard Blender, Rodrigo Caballero, Angela Cocozza, H. Dacre, Yang Feng, Klaus Fraedrich, Jens Grieger, Sergey Gulev, John Hanley, Tim Hewson, Masaru Inatsu, Kevin Keay, Sarah Kew, Ina Kindem, G.C. Leckebusch, and Heini Wernli. Imilast: A community effort to intercompare extratropical cyclone detection and tracking algorithms. *Bulletin of the American Meteorological Society*, 94:529–547, 04 2013. 1

[10] Prabhat, K. Kashinath, M. Mudigonda, S. Kim, L. Kapp-Schwoerer, A. Graubner, E. Karaismailoglu, L. von Kleist, T. Kurth, A. Greiner, A. Mahesh, K. Yang, C. Lewis, J. Chen, A. Lou, S. Chandran, B. Toms, W. Chapman, K. Dagon, C. A. Shields, T. O'Brien, M. Wehner, and W. Collins. Climatenet: an expert-labeled open dataset and deep learning architecture for enabling high-precision analyses of extreme weather. *Geoscientific Model Development*, 14(1):107–124, 2021. 2

[11] Md Rahman and Yang Wang. Optimizing intersection-over-union in deep neural networks for image segmentation. volume 10072, pages 234–244, 12 2016. 3

[12] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015. 1, 2

[13] C. A. Shields, J. J. Rutz, L.-Y. Leung, F. M. Ralph, M. Wehner, B. Kawzenuk, J. M. Lora, E. McClenny, T. Osborne, A. E. Payne, P. Ullrich, A. Gershunov, N. Goldenson, B. Guan, Y. Qian, A. M. Ramos, C. Sarangi, S. Sellars, I. Gorodetskaya, K. Kashinath, V. Kurlin, K. Mahoney, G. Muszynski, R. Pierce, A. C. Subramanian, R. Tome, D. Waliser, D. Walton, G. Wick, A. Wilson, D. Lavers, Prabhat, A. Collow, H. Krishnan, G. Magnusdottir, and P. Nguyen. Atmospheric river tracking method intercomparison project (artmip): project goals and experimental design. *Geoscientific Model Development*, 11(6):2455–2474, 2018. 2

[14] Petru Soviany, Radu Tudor Ionescu, Paolo Rota, and Nicu Sebe. Curriculum learning: A survey. *International Journal of Computer Vision*, pages 1–40, 2022. 3, 8

[15] Adam J Stewart, Caleb Robinson, Isaac A Corley, Anthony Ortiz, Juan M Lavista Ferres, and Arindam Banerjee. Torchgeo: deep learning with geospatial data. In *Proceedings of the 30th International Conference on Advances in Geographic Information Systems*, pages 1–12, 2022. 4

[16] Tianyi Wu, Sheng Tang, Rui Zhang, and Yongdong Zhang. Cgnet: A light-weight context guided network for semantic segmentation. *CoRR*, abs/1811.08201, 2018. 1, 2

[17] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. *CoRR*, abs/1807.10221, 2018. 1, 2