Post-processing cloud cover forecasts using Deep Learning

Yinghao Dai
ETH Zurich
yidai@ethz.ch

Claudio Ferrari
ETH Zurich
ferraric@ethz.ch

Rhea Sukthanker
ETH Zurich
srhea@ethz.ch

July 31, 2021

Abstract

Cloud cover prediction is an important problem in modern weather forecasting. Recently, areas like Natural Language Processing (NLP), Computer Vision and Recommender Systems have seen a wave of Deep Learning models which prove to perform much better than handcrafted rules and features. Perhaps the most significant gain of these models is the ability to leverage simple features to learn better feature representations for the model; thus minimizing the handcrafting of rules and dependency on manual processing. In this project, we use Deep Learning to tackle the challenging problem of cloud cover post-processing. We show that our models significantly reduce prediction error and also lead to better calibrated forecasts. To the best of our knowledge, this work presents the first meteorological post-processing model that allows one to predict a non-parametric forecast distribution.

1 Introduction

For centuries long, people have been attempting to predict weather as accurately as possible. This is not only useful for planning our activities, but can also be more vital, for instance when airlines need to decide whether it is safe to fly, or when there is an urgent citizen evacuation to be done before a natural calamity strikes.

The earliest forms of weather forecasting were based on, mostly inaccurate, folk wisdom, such as: "When dew is on the grass, rain will never come to pass". Modern weather forecasting, however, is done by computers using numerical weather models, an example of which is called COSMO-E. Such models are based on partial differential equations describing the changing states of the atmosphere, and approximate the solutions to these equations numerically. Systematic biases in the numerical forecasts arise and can be attributed to two causes. Firstly, the forecasts rely heavily on an initial condition which can not be measured exactly;

and secondly, the numerical model is in itself an approximation to the real world and therefore creates systematic errors in the modeling process.

Therefore, the numerical weather forecasts need to be post-processed, in order to make them more accurate. At the Swiss Federal Office of Meteorology and Climatology (in the following abbreviated as *MeteoSwiss*), this is mostly done manually, by experts who have seen many forecasts and observations, and could thus use their domain knowledge to rectify the forecasts to a certain extent.

This is an example of statistical post-processing, in which one considers the statistical relationship between the forecasts and the actual weather, and tries to identify patterns to make the forecasts more accurate. This is exactly what we have attempted during the course of this project, using a Deep Learning model. Our goal was to obtain probabilistic forecasts for cloud coverage at any point in Switzerland, based on the available forecasts from the COSMO-E model.

2 Dataset

The COSMO-E model provides numerical weather predictions for more than twenty weather-related metrics, including temperature, humidity, rainfall, and cloud coverage, for each point on a 2 km $\times$ 2 km grid over Switzerland, with hourly resolution. The model is initialized twice per day, at midnight and at noon, respectively. Every model run gives forecasts from 0 to 120 hours into the future. For any of these 121 so-called *lead times*, there are 21 forecasts given, which represent realizations of a probabilistic forecast. The way these 21 so-called ensemble members are created is by adding 20 random perturbations to the initial conditions at the time when a new model run is started. This implies that all ensemble members - except the first one - cannot be tracked over time, since they are newly created at every initialization.

As our ground truth labels, we use cloud coverage measurements, which are derived from EUMETSAT CM-SAF satellite data [1] on a 5 km $\times$ 5 km grid. In order to match the label grid to the forecast grid we do a nearest neighbor matching, i.e. for every point on the 2 km $\times$ 2 km COSMO-E grid we find the closest measurement point on the 5 km $\times$ 5 km grid. More detail on this matching will be given in subsection 2.3.

Both the forecasts and the measurements have values ranging from 0 (clear sky) to 100 (completely cloudy).

2.1 Scale of the data

A major roadblock we faced during the initial stages of the project was the size of the dataset. As described earlier, COSMO-E provides predictions (21 ensemble members) for more than twenty weather-related metrics. With a simulation being initialized twice a day and the simulations running for 120 hours, or five days, it means that at any given point there are ten simulations running at the same time. This leads to a fairly large dataset of roughly 20 TB. To overcome this issue, we firstly focus only on one feature, namely the one indicating the

total cloud cover. Secondly, we subsampled the dataset by taking only every second grid point in both latitude and longitude direction and only every fifth day of COSMO-E model initializations. From the 21 ensemble members we first extracted mean and variance, later in the project we instead extracted 7 equispaced quantiles of the members. This is explained in more detail in section 4.

2.2 Train/Validation/Test Split

Since we have strong temporal and spatial correlations in our data, the choice of Train/Validation/Test split has to be done carefully in order to minimize information leakage between the sets. We chose to do the split by year; using the years 2014-2016 for training, the year 2017 for validation and the year 2018 for testing. This means we have only minimal temporal information leakage from December to January between the sets.

Note that through this assumption we do not take into consideration the effects of climate change. We feel this assumption is reasonable because of the limited time frame of five years.

We choose to evaluate by season, as we aim for a better performance over all seasons rather than for a particular one. Thus if a model performs well, we in some sense need to ensure that the model does not just do well in one season but performs really poorly in another. As an example, for us winter corresponds to the months January, February and December of a particular year. We also split by the initialization times (midnight and noon). This choice is motivated by knowledge from our domain experts that the models initialized at different times differ significantly from each other.

2.3 Coordinate System Mismatch

Both the numerical weather forecasts and the cloud coverage observations are given on a rectangular grid of coordinates. As mentioned before, the grid resolution is roughly 2 km $\times$ 2 km for the forecasts, and 5 km $\times$ 5 km for the observations. The observations grid is rectangular on the standard longitude/latitude coordinates, which means it is in fact not perfectly rectangular; one degree of difference in longitude reflects a smaller spatial distance in the north than in the south of Switzerland, due to the earth's spherical shape.

To address this issue, the forecasts are given on a grid that is rectangular on a rotated coordinate system, in which the north pole (90°N 0°W) has been rotated to -170°E 43°N (in standard coordinates). In this rotated system, the equator passes through Switzerland, which means the distortions referred to above are minimal. The mismatch of the forecasts and observations grids is illustrated in Figure 1. We decided to map the rotated coordinates to standard coordinates, and then do the nearest neighbor matching, as mentioned in section 2.

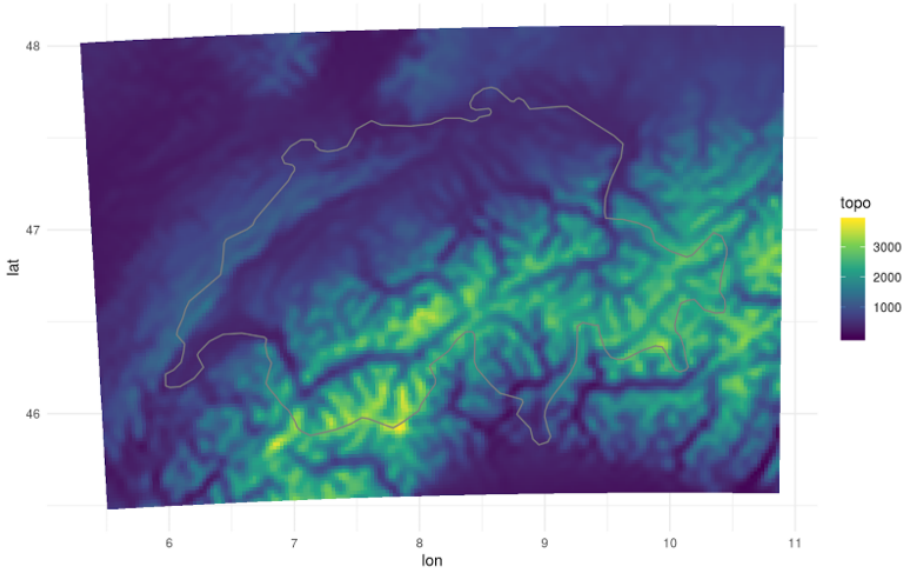

Figure 1: Forecast grid topography mapped to standard (observations grid) coordinates.

2.4 Evaluation metrics

2.4.1 Continuous ranked probability score

As an evaluation metric for comparing a probabilistic forecast to an observation, we used the continuous ranked probability score (abbreviated as **CRPS** or as CRPS score[1]). For a forecast f and an observation o, it is defined as

$$\mathrm{CRPS}(f, o) = \int_{-\infty}^{\infty} (F_f(x) - F_o(x))^2 \, \mathrm{d}x, \tag{1}$$

where F_f and F_o are the cumulative distribution functions for f and o, respectively. We can see that the CRPS is inversely proportional to the similarity of the distributions of the forecast and observation: the more similar those distributions are, the smaller the CRPS. Since the cloud coverage observation is a single value $c \in [0, 100]$, we can substitute $F_o(x) = \mathbf{1}[x > c]$ into (1). Moreover, without loss of generality, we can assume that the forecasts are within the interval $[0, 100]$ as well (in other words, values outside that interval have zero probability mass). This is because we can always obtain a forecast that is at least as good (in terms of CRPS) by clipping the values to $[0, 100]$. Therefore, it suffices to compute the integral over the range $[0, 100]$, instead of the whole real line.

[1] This is a typical example of the RAS syndrome.

If the forecast f consists of M values $f_1, \ldots, f_M \in [0, 100]$ (with the COSMO-E forecasts, we have $M = 21$), then we treat $f_1, \ldots, f_m \sim P_f$ as samples from a forecast distribution P_f, and F_f is the empirical cumulative distribution function of P_f based on these samples. In this case, as shown in [2], the CRPS for a forecast $f = (f_1, \ldots, f_M)$ and cloud coverage observation c reduces to:

$$\mathrm{CRPS}(f, c) = \frac{1}{M} \sum_{m=1}^{M} |f_m - c| - \frac{1}{2M^2} \sum_{n=1}^{M} \sum_{m=1}^{M} |f_n - f_m|. \tag{2}$$

We refer to the above metric as the *Ensemble CRPS* metric. Another version of the CRPS metric is the *Gaussian CRPS* metric, where the forecast $f_{\mu,\sigma^2} \sim \mathcal{N}(\mu, \sigma^2)$ is normally distributed and parametrized by a mean μ and variance σ^2. The Gaussian CRPS with respect to a variable x is given by [3]

$$\mathrm{CRPS}(f_{\mu,\sigma^2}, c) = \sigma \left(\frac{c-\mu}{\sigma} \left[2\Phi \left(\frac{c-\mu}{\sigma} \right) - 1 \right] + 2\phi \left(\frac{c-\mu}{\sigma} \right) - \frac{1}{\sqrt{\pi}} \right) \tag{3}$$

where Φ and ϕ represent the standard Gaussian probability density and cumulative distribution functions, respectively.

2.4.2 Probability integral transform histogram

The forecasts produced should not only minimize the CRPS but should also be well-calibrated. That means that forecast probabilities should correspond to actual average occurrence frequencies. This can be assessed using so-called probability integral transform (PIT) histograms. If we assume the prediction consists of 21 ensemble members, they are computed by taking the rank of one observation with respect to the ordered predictions and then plotting a histogram of the occurrences of those ranks.

For a well-calibrated forecast, the PIT histogram should roughly correspond to that of a uniform distribution. One way to think about this is as follows: if the observation were actually sampled from the predicted distribution, the ranks would be uniformly distributed.

If the histogram shows skewness, it indicates a bias. A U-shape of the PIT histogram shows over-confidence of the member predictions and an inverse U-shape shows over-dispersion of the forecast, meaning that the members are too much spread out. As mentioned before, ideal case would be a histogram that reflects a uniform distribution.

In our case, we have a lot of ties at 0 and 100. It is important to break these ties randomly, since otherwise the obtained histogram will exhibit bias. If we assume the prediction consists of mean and variance, we can compute the CDF value of the induced normal distribution at the observation. Then we can again plot a histogram of the obtained values. Since the support of the normal distribution goes beyond our range of $[0, 100]$, similar to the random tie breaking of the ranks, we do the following. For observations that are zero, we sample a value uniformly at random from $[0, \mathrm{CDF}(0)]$. For observations that equal 100, the analogous procedure is done at the other end of the range. This again ensures that we obtain an unbiased histogram.

3 Baselines

The most natural baseline to compare our post-processing results to is the direct model output (abbreviated as *DMO*) we get from the COSMO-E Model, i.e. the unprocessed input to our models.

In addition to that, we looked into two more baselines that are used by meteorologists: persistence and a climatological baseline.

The persistence baseline model assumes that the cloud coverage will stay constant from the time a simulation is initialized. That means it predicts the same cloud coverage for all 121 lead times, and this prediction equals the observation at initialization time.

The climatological baseline is based on the assumption that cloud coverage is somewhat constant over years per season. To understand the way predictions are computed, let us look at an example. Suppose we want to predict the cloud coverage on February 24, 2019, at 9am Swiss time.[2] This corresponds to lead time 22 for a simulation initialized at noon (UTC) the day before. We take all predictions (labels) for the same grid point (defined by its latitude and longitude), same time of day and same lead time from all available years for the winter months (December, January, February). We leave out a window of 10 days before and after our date (and the date itself we want to predict) to avoid information leakage from the actual ground truth value. Our final cloud coverage predictions are an average of all values fulfilling our set criteria.

Initialization Time	Season	Persistence	Climatological	DMO
00:00	Winter	34.852	19.505	**17.42**
12:00		34.294	19.505	**17.37**
00:00	Spring	37.084	20.185	**15.22**
12:00		33.415	20.185	**15.19**
00:00	Summer	42.410	22.602	**17.80**
12:00		40.327	22.602	**17.42**
00:00	Fall	39.524	23.245	**17.99**
12:00		38.778	23.245	**17.80**

Table 1: Baselines CRPS on the test set (2018). The best scores are highlighted in bold.

The results seem to give a clear ranking of the baselines. The direct model output has - by far - the smallest CRPS error (best performance). Note that the climatological baseline is invariant to the initialization time and hence the aggregated CRPS values are same for the two initialization times, as depicted in the table above. Moreover, the climatological baseline seems to give better

[2] In Switzerland, Central European Time (CET) is used, which is UTC+1 during winter.

results than the persistence baseline. Due to those results, we decided to compare the performance of our models only to the direct model output.

4 Models

We tried two completely different model architectures, the first of which is an Encoder-Decoder architecture and the second one is our best-performing model, which we refer to as an embedding based model.

4.1 Input and Output

Since we are post-processing the DMO, it will serve as the input to our model. The only feature we are using is the total cloud coverage, therefore the input does not need to be modified in order to produce a valid result. In fact, since the DMO already gives a good estimate of cloud coverage, the identity model (which just outputs the DMO) actually performs quite well.

For every data point, defined by its longitude, latitude, initialization time and lead time, we have 21 input values (ensemble member predictions) in the range $[0, 100]$. The corresponding label is a single value in the same range. Since we are tasked with producing a probabilistic forecast, the output of our model either has to represent the parameters of a given distribution, or, as in the DMO, some members of an ensemble.

In order to control the size of the dataset, we needed a more compact representation of the 21 input values. In addition to that, 20 of the 21 input values are random perturbations of the first one and therefore, permutation of those twenty values leads to an equivalent forecast.

4.1.1 Parametric approach

Initially we extracted mean and variance from the 21 ensemble members, and from that predicted mean and variance again. Conceptually, this means predicting a normal distribution to fit our labels. This has been standard practice for the post-processing of numerical weather predictions [4] and was also recommended to us by our domain expert.

4.1.2 Non-parametric approach

In our previous parametric approach, we restrict ourselves in two ways. First of all, we only take mean and variance from the input, which - in case the 21 ensemble members do not follow a normal distribution - means losing a lot of information. Secondly, we restrict our predictions to normal distributions, which might be a poor reflection of the actual observation distribution.

In order to get rid of the first restriction, we decided to extract seven empirical quantiles, equispaced between zero and one, from our 21 input values. From those we directly predicted 21 cloud coverage values, thus also removing the second restriction.

Our intuition was that this non-parametric approach would outperform the parametric one, which it did. In order to ensure that this is not just because of the input restriction, we provide comparison with a model in which we took seven quantiles as input to predict the mean and variance.

4.2 Encoder-Decoder Architecture

We used the so called *U-Net* architecture [5]. This architecture has recently proven to yield very good results in various biomedical image segmentation tasks, for example brain image segmentation [6] or skin tissue segmentation [7]. Since one can conceptually interpret a cloud coverage forecast as an image, the thought was that it could perform well on our task as well.

Our setting differs mainly in two aspects. First of all we are conceptually working with data along three dimensions - latitude, longitude and time - instead of two. Secondly we want to do pixel-wise regression instead of classification as in the semantic segmentation tasks outlined above. However, the U-Net architecture already has been successfully applied to three-dimensional input [8]. Most interestingly, it has been applied to the pansharpening problem, a post-processing technique for sharpening images [9]. This is conceptually fairly close to our case of "sharpening" (or improving) cloud cover forecasts.

The input to the model is one complete five-day model simulation with size 90 (longitude) $\times$ 59 (latitude) $\times$ 121 (lead time). The output is of the same shape and contains the post-processed forecasts. It is necessary to input all these values as one sample in order to leverage the model's ability to recognize local patterns using convolutions.

Unfortunately, the obtained results were sub-par and were in fact not even able to improve on the DMO. Also, since the input size is fairly large, going from the parametric to the non-parametric approach proved to be computationally infeasible with the available hardware. Therefore, we decided to abandon this approach fairly early in the process.

4.3 Embedding-based models

In this section we elaborate on our best performing model which derives from the recent success of the use of embeddings in a broad range of areas in Deep Learning. The use of dense embeddings first became popular in the field of Natural Language Processing (NLP) [10]. In this area, the embeddings are primarily used to convey meaning of a word in a text [11, 12]. This idea can further be generalized to embed anything that can be represented as a discrete symbol as a dense embedding. Recently, many areas beyond NLP have seen a huge benefit from using this type of dense embedding maps; an example of this is Deep Collaborative Filtering [13]. Basically any feature or symbol which represents a particular category can be effectively mapped to a dense vector which in some sense can learn to capture more information than a model just relying on raw features. We follow a similar ideology for our model inspired by [14], variants of which will be delineated in the subsections below.

4.3.1 Mean Variance to Mean Variance (MVtoMV) model

The architecture of the mean variance model is as depicted in Figure 2. The input to this model is formed by the mean and the variance of the 21 ensemble members and the output is the corrected mean and variance of the ensemble members as described in section 4.1. We also use some additional context information as input to the model. More precisely, we include the latitude, longitude, the initialization time, the month under consideration, the hour of the day and the lead time corresponding to the initialization. We convert the context level features, e.g. longitude-latitude coordinate into a set of ids, where each id represents a unique pair. We further map these unique symbols to a dense embedding which captures the meaning of the symbol to its entirety. The output (corrected mean and variance) are further used to compute and optimize a Gaussian CRPS loss function (see equation (3)). We use two separate feed-forward neural networks (all with ReLU activation except the final layer), both of which take the concatenated input into consideration to predict the mean and variance, respectively.

However, there is an issue with doing so. The variance is always non-negative, and there is no clear way to enforce this behavior in our model. Hence, before optimizing the loss function, we take an absolute value of the variance to enforce its non-negative behavior. Also, we use a somewhat arbitrary assumption that a Gaussian distribution is a good choice for the distribution of our ensemble members (which is in general not true).

The results depicted in table are the results for our validation metric, namely the Ensemble CRPS (see equation (2)). To compute the Ensemble CRPS and compare it to the CRPS of the DMO, we sample the 21 members as equi-spaced quantiles from the predicted Gaussian distribution. We further clip these values to a range of $[0, 100]$ before computing the metric.

4.3.2 Seven quantiles to Mean Variance (7qtoMV) model

The seven quantiles to mean variance model is different from the mean variance to mean variance model only in terms of the input data. That is, we now replace the input of the previous model (mean and variance) with the seven equi-spaced quantiles of the ensemble members as defined in section 4.1, preserving the same context information as in the MVtoMV model. The intuition behind doing so, is that we expect the seven quantiles to preserve more information about the ensemble members' distribution than just their mean and variance. Also another advantage of this design choice of considering the seven quantiles instead of the 21 ensemble members themselves is that we now effectively handle the permutation invariance of the 21 ensemble members as the quantiles can be easily tracked over time. In this model we again optimize on a Gausssian CRPS loss function and validate on the Ensemble CRPS metric.

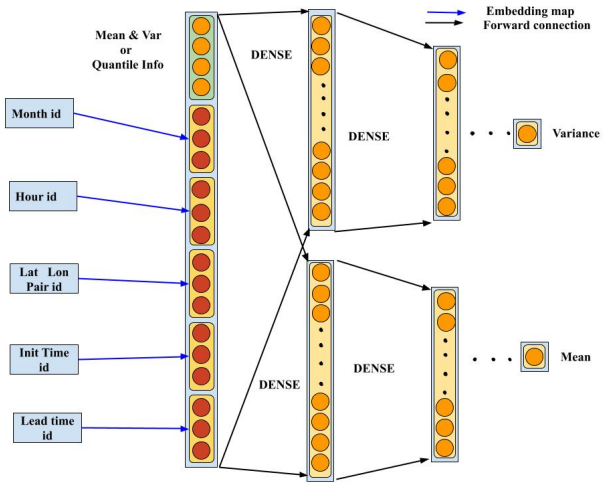

Figure 2: Architecture of the two parametric models: MVtoMV and 7qtoMV.

4.3.3 Seven quantiles to 21 Ensemble members (7qto21) model

Our best performing model was a non-parametric model that directly predicts the 21 ensemble members from the seven quantiles. The input to this model is the same as for the 7qtoMV model, i.e. the seven quantiles and the relevant context information. This model mainly differs from the others in terms of its output. We now choose to directly output the 21 calibrated ensemble members, which means we directly optimize on a closed form of the Ensemble CRPS loss (see equation (2)). The 7 quantiles (with meta information) are now passed through fully connected feed forward layers (all with ReLU activations except the final layer) the output of which are the 21 calibrated ensemble members. A major advantage here is that we make no prior assumptions on the distribution of the ensemble members. As we see in the results section this non-parametric model clearly outperforms the others. The architecture of this model is depicted in Figure 3.

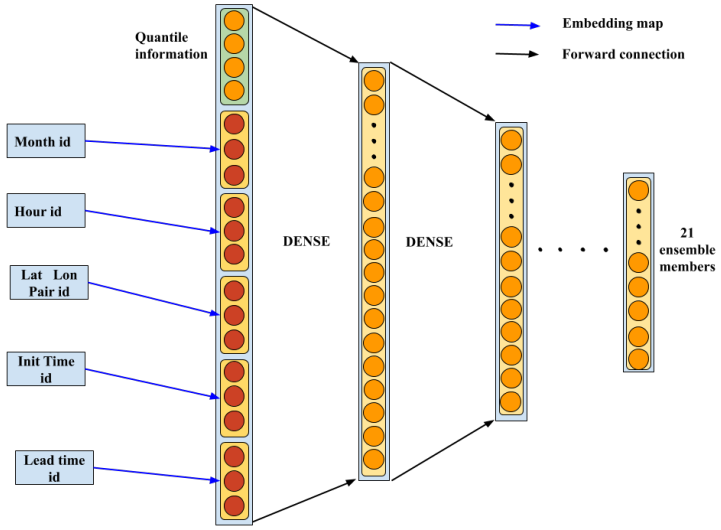

Figure 3: Architecture of the non-parametric model: 7qto21.

5 Results

Initialization Time	Season	DMO	MVtoMV	7qtoMV	7qto21
00:00	Winter	17.42	16.19	15.97	**15.42**
12:00		17.37	16.40	16.25	**15.56**
00:00	Spring	15.22	14.17	14.61	**14.09**
12:00		15.19	14.54	14.26	**13.74**
00:00	Summer	17.80	**16.68**	17.64	16.90
12:00		17.42	16.34	16.83	**16.23**
00:00	Fall	17.99	17.19	17.54	**16.49**
12:00		17.80	16.95	17.19	**16.56**

Table 2: Mean CRPS of proposed models vs. DMO on the test set (2018). The best scores are highlighted in bold.

Table 2 briefly summarizes our results for the three models (MVtoMV, 7qtoMV, and 7qto21). The MVtoMV model outperforms the DMO by about 1.0 CRPS on average. By shifting to a non-parametric model (7qto21), we gain about another 0.5 in terms of CRPS value. This observation agrees with our intuition that a

parametric Gaussian model may not suffice to capture the distribution of the Cloud Cover ensemble members to its entirety. We can also see that this is not only because the input of the non-parametric model captures more information, as the 7qtoMV model's performance is on par with the MVtoMV model.

We can also see the decreased CRPS scores of the post-processed methods when visualizing them on a map over Switzerland, as done in Figure 4. One interesting thing we observe is that both 7qtoMV and 7qto21 exhibit vertical line patterns. We assume the labels are relatively smooth in space, especially since we are averaging over seasons and lead times here. This would mean that the patterns we see in CRPS should roughly correspond to patterns in predictions. So seemingly, these two models pick up better on correlations in latitude than in longitude.

It might be insightful to see whether these map plots significantly change when only considering a particular season. For these plots, in which we also included the performance of the baselines mentioned in Section 3, we refer the reader to Figures 6-9 in the Appendix.

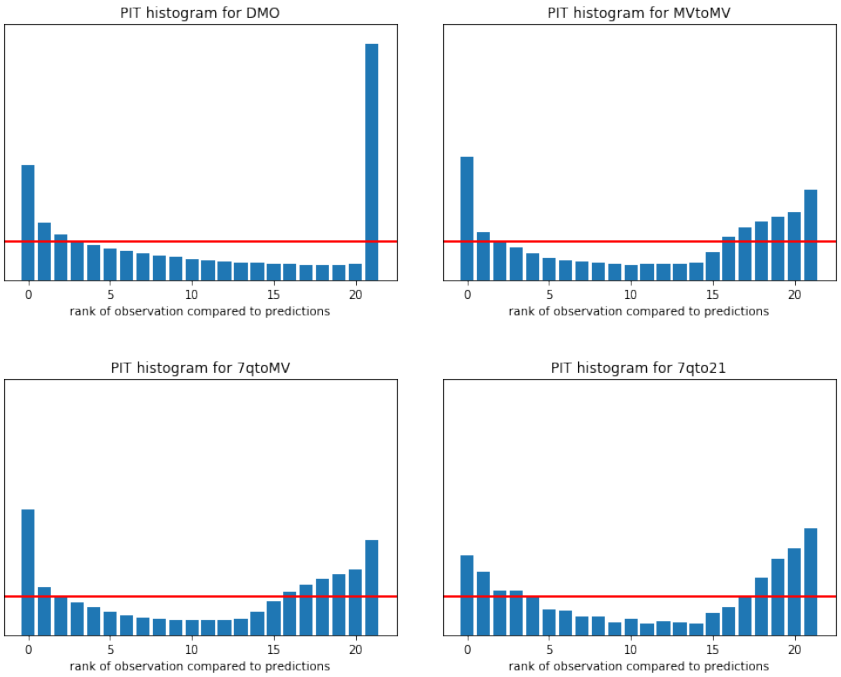

Figure 5: PIT histograms of different models

The PIT histograms of the DMO and our models are given in Figure 5. We observe that all forecasts are under-dispersed, as can be seen by the U-shaped histograms. For the DMO, rank 21 has by far the highest frequency, meaning that the DMO exhibits a tendency to underestimate cloud coverage. All post-

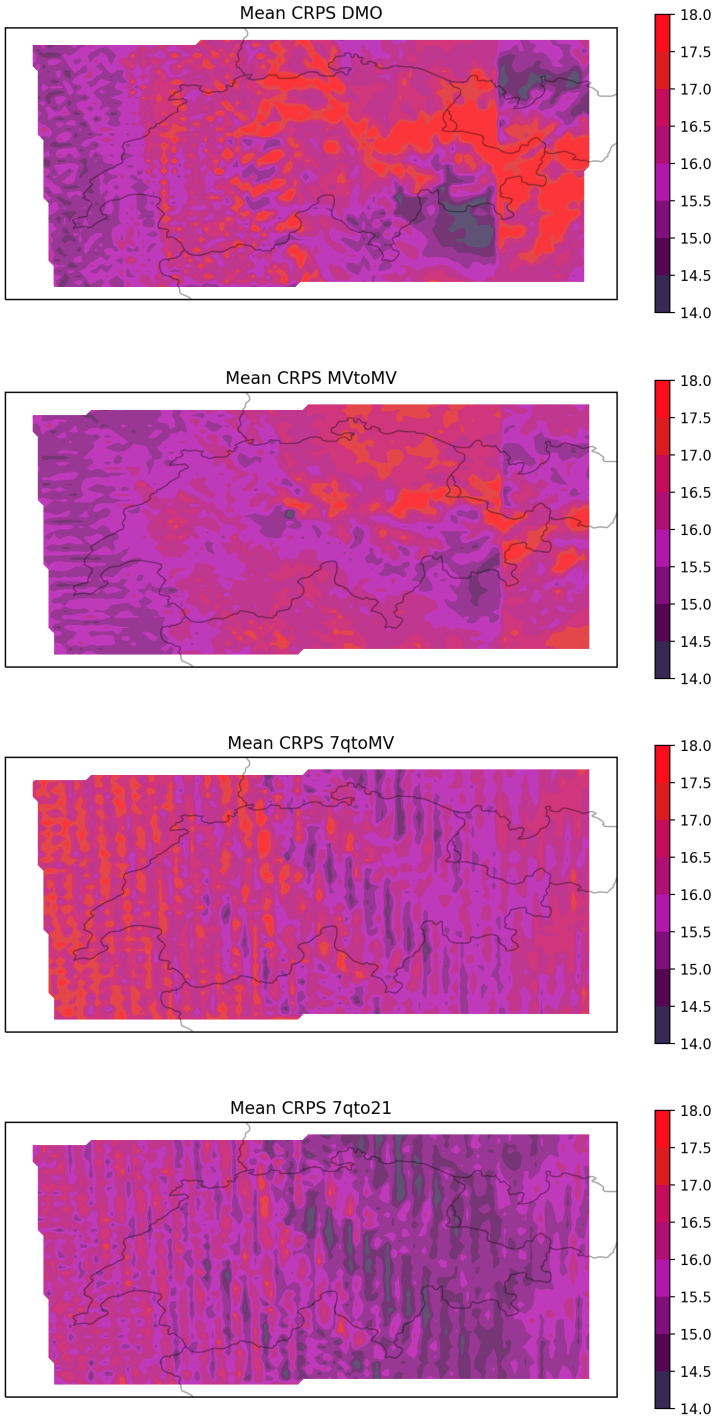

Figure 4: Mean CRPS by location of different models.

processed forecasts seem to be better-calibrated, with the differences between them being minimal.

6 Performance at unseen grid points

Since we would like to be able to make a cloud coverage forecast for an arbitrary point in Switzerland, we analyzed the performance of the models at points which they have not seen during training. For training, as mentioned in subsection 2.1, we took every second grid point in both the latitude and longitude direction. So, we took the even grid point indices, and now we evaluate the models at grid points with odd grid point indices to see whether our model is able to generalize in space. At those points we have almost everything available to make a prediction. The only input that is not readily available is the longitude-latitude pair id. We worked around this by taking the id of one of the four nearest neighbors that were used for training (and thus was assigned a longitude-latitude pair id). One can see that as an interpolation of the input value.

Initialization Time	Season	DMO	MVtoMV	7qtoMV	7qto21
00:00	Winter	17.45	16.23	16.22	**15.85**
12:00		17.40	16.45	16.51	**16.01**
00:00	Spring	15.22	**14.18**	14.84	14.49
12:00		15.20	14.55	14.51	**14.19**
00:00	Summer	17.77	**16.66**	17.91	17.40
12:00		17.41	**16.33**	17.11	16.79
00:00	Fall	18.00	17.21	17.70	**16.77**
12:00		17.80	16.97	17.36	**16.82**

Table 3: Mean CRPS at unseen grid points DMO v/s Our Models on test set (2018)

The results of this interpolation for the different models are given in Table 3. We see that the post-processed forecasts are still better than the DMO, although by a lesser margin. Also the MVtoMV model seems to generalize better in space than 7qto21; their performances are now comparable, whereas 7qto21 performed clearly better on the grid points used in training. The 7qtoMV model has the highest CRPS of all our models. It seems that the models taking seven quantiles as input generalize less in space than the one taking only mean and variance. A possible explanation for this might be that mean and variance of the 21 ensemble members vary more smoothly in space than seven quantiles.

7 Conclusion and Future Directions

We have devised a post-processing method, similar to [14], for cloud coverage forecasts, which is able to improve the CRPS and calibration of the DMO. Furthermore, we extend the architecture in order to predict a non-parametric distribution and by doing so, improve the CRPS even more. The models are able to generalize to unseen points while still outperforming the DMO.

The CRPS scores of the non-parametric model exhibit interesting spatial patterns, and we do not have a satisfying explanation of this phenomenon at the current time. We leave it to future research to investigate this further.

Another interesting avenue for future work would be to see how incorporating more features of the COSMO-E model (like humidity, temperature, etc.) would affect the performance. Though in general we observe that the model is able to learn from relatively small amounts of data, it still remains to be verified whether sampling more grid points or more initialization times gives a significant performance gain.

References

[1] "MS Windows NT kernel description," https://wui.cmsaf.eu/safira/action/viewDoiDetails?acronym=CFC_METEOSAT_V001, accessed: 2019-12-18.

[2] M. Zamo and P. Naveau, "Estimation of the continuous ranked probability score with limited information and applications to ensemble weather forecasts," *Mathematical Geosciences*, vol. 50, no. 2, pp. 209–234, 2018.

[3] E. P. Grimit, T. Gneiting, V. Berrocal, and N. A. Johnson, "The continuous ranked probability score for circular variables and its application to mesoscale forecast ensemble verification," *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography*, vol. 132, no. 621C, pp. 2925–2942, 2006.

[4] T. E. Fricker, C. A. Ferro, and D. B. Stephenson, "Three recommendations for evaluating climate predictions," *Meteorological Applications*, vol. 20, no. 2, pp. 246–255, 2013.

[5] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015.

[6] W. Chen, B. Liu, S. Peng, J. Sun, and X. Qiao, *S3D-UNet: Separable 3D U-Net for Brain Tumor Segmentation: 4th International Workshop, BrainLes 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 16, 2018, Revised Selected Papers, Part II*, 01 2019, pp. 358–368.

[7] K. R. J. Oskal, M. Risdal, E. A. M. Janssen, E. S. Undersrud, and T. O. Gulsrud, "A u-net based approach to epidermal tissue segmentation in whole slide histopathological images," *SN Applied Sciences*, vol. 1, no. 7, p. 672, Jun 2019. [Online]. Available: https://doi.org/10.1007/s42452-019-0694-y

[8] Özgün Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3d u-net: Learning dense volumetric segmentation from sparse annotation," 2016.

[9] W. Yao, Z. Zeng, C. Lian, and H. Tang, "Pixel-wise regression using u-net and its application on pansharpening," *Neurocomputing*, vol. 312, pp. 364 – 371, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925231218307008

[10] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of machine learning research*, vol. 3, no. Feb, pp. 1137–1155, 2003.

[11] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[12] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[13] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proceedings of the 26th international conference on world wide web*. International World Wide Web Conferences Steering Committee, 2017, pp. 173–182.

[14] S. Rasp and S. Lerch, "Neural networks for postprocessing ensemble weather forecasts," *Monthly Weather Review*, vol. 146, no. 11, pp. 3885–3900, 2018.

A Appendix

A.1 Map plots Baselines

In Figures 6 and 7, we can see the performance for different seasons and different regions in Switzerland, of the climatology and persistence baseline, respectively. These baselines are defined in Section 3, in which we saw that the climatology baseline performs better than the persistence baseline. This is also visible in Figures 6 and 7 (note that the two plots have different color scales, in order to show a higher level of detail). In both figures, we can clearly see a distinction between different geographical regions of Switzerland: lowlands, Alps, Jura, lakes, et cetera.

In the top plot of Figure 6, the lakes of Geneva, Neuchâtel, and Constance are clearly visible (and to a lesser extent those of Zurich, Lucerne, and Thun are visible as well). The CRPS value in the lakes is smaller than in neighboring regions, which could mean that cloud coverage above lakes in winter is somewhat constant over different years. The opposite seems true for cloud coverage in the lowlands in summer, and also the clouds above Ticino seem harder to predict using only information about the date.

In the top plot of Figure 7, we can also see some of the Swiss lakes, though not so clearly as with the climatology baseline. It seems that the blue region (consisting of mainly lowlands and lakes) have more constant cloud coverage over a period of five days in winter. On the other hand, cloud coverage in summer seems a lot more variable over short periods of time (< 5 days).

A.2 Map plots DMO vs. Our Best Model

We made analogous plots for comparing the NWP direct model output with our best-performing model, shown in Figures 8 and 9 respectively. We can observe that the geographical regions of Switzerland are no longer visible in the plots. Note that unlike in the baseline plots, the color scales are the same in both figures. This means we can compare the colors of Figures 8 and 9 to see that the DMO is indeed outperformed by our best-performing model.

In all plots of the DMO (Figure 8), we can see a clear vertical line that cuts off the eastern part of Graubünden (Grisons), exactly at the prime meridian (longitude zero) of the rotated coordinate system (see subsection 2.3). It is interesting that this vertical cutoff is not visible in the plots of our best-performing model (Figure 9). In those plots, we do see that the vertical line patterns discussed in Section 5 are visible in every individual season.

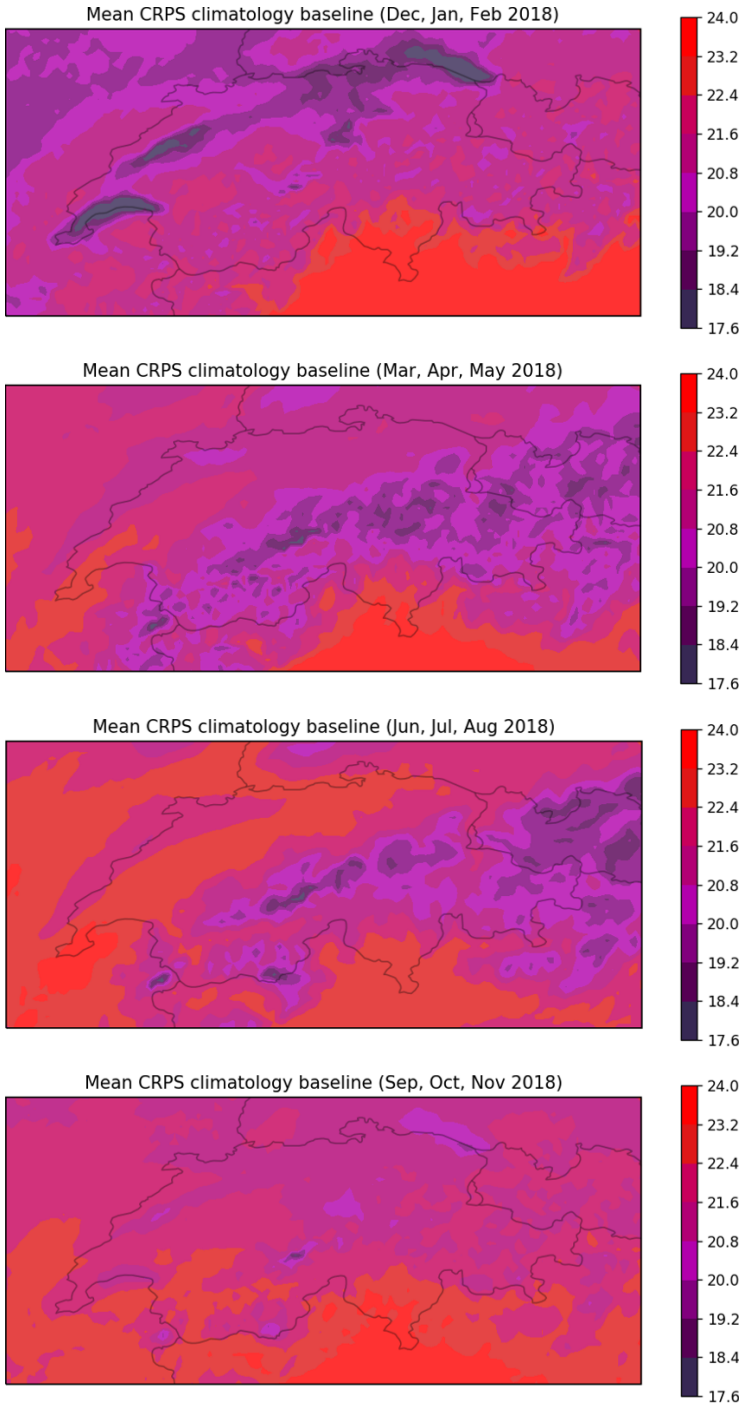

Figure 6: Mean CRPS by location of climatology baseline for different seasons: winter, spring, summer, and autumn (from top to bottom).

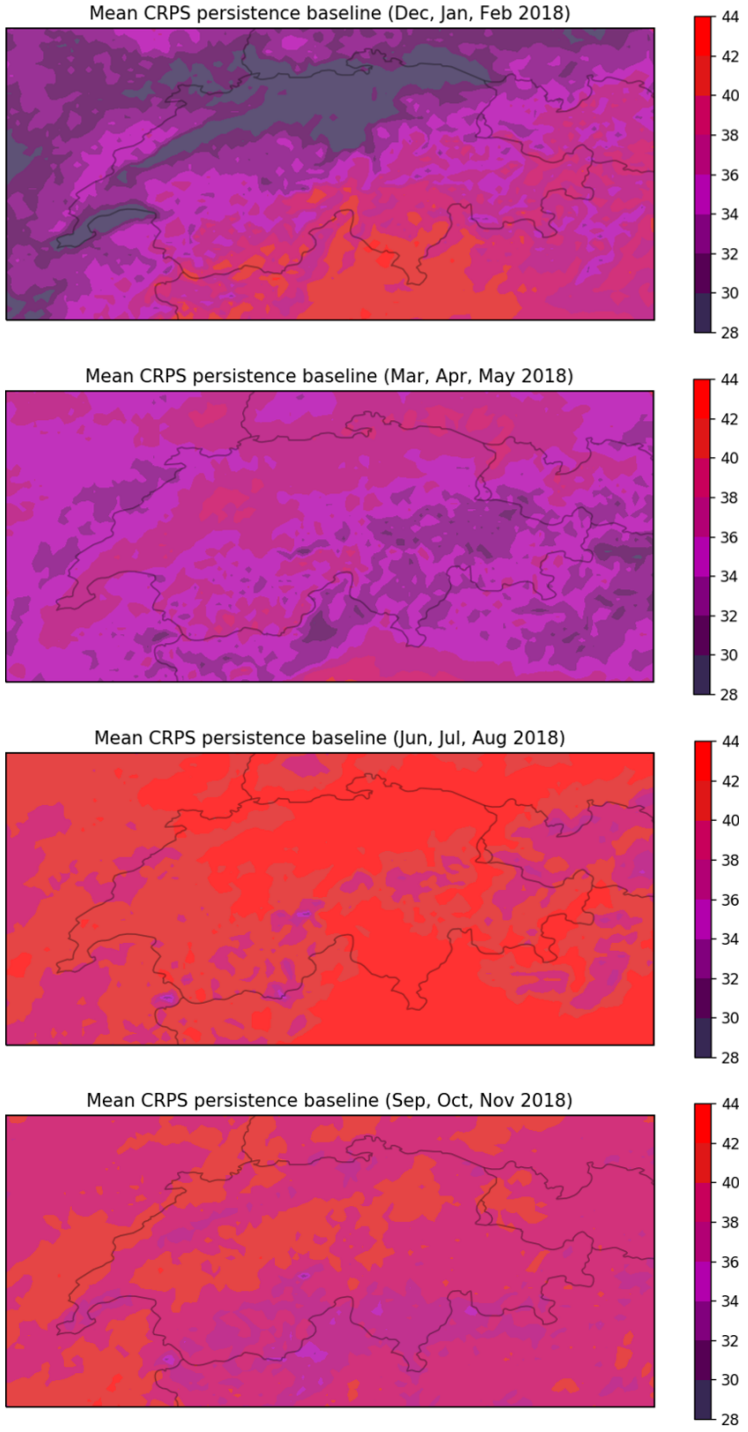

Figure 7: Mean CRPS by location of persistence baseline for different seasons: winter, spring, summer, and autumn (from top to bottom).

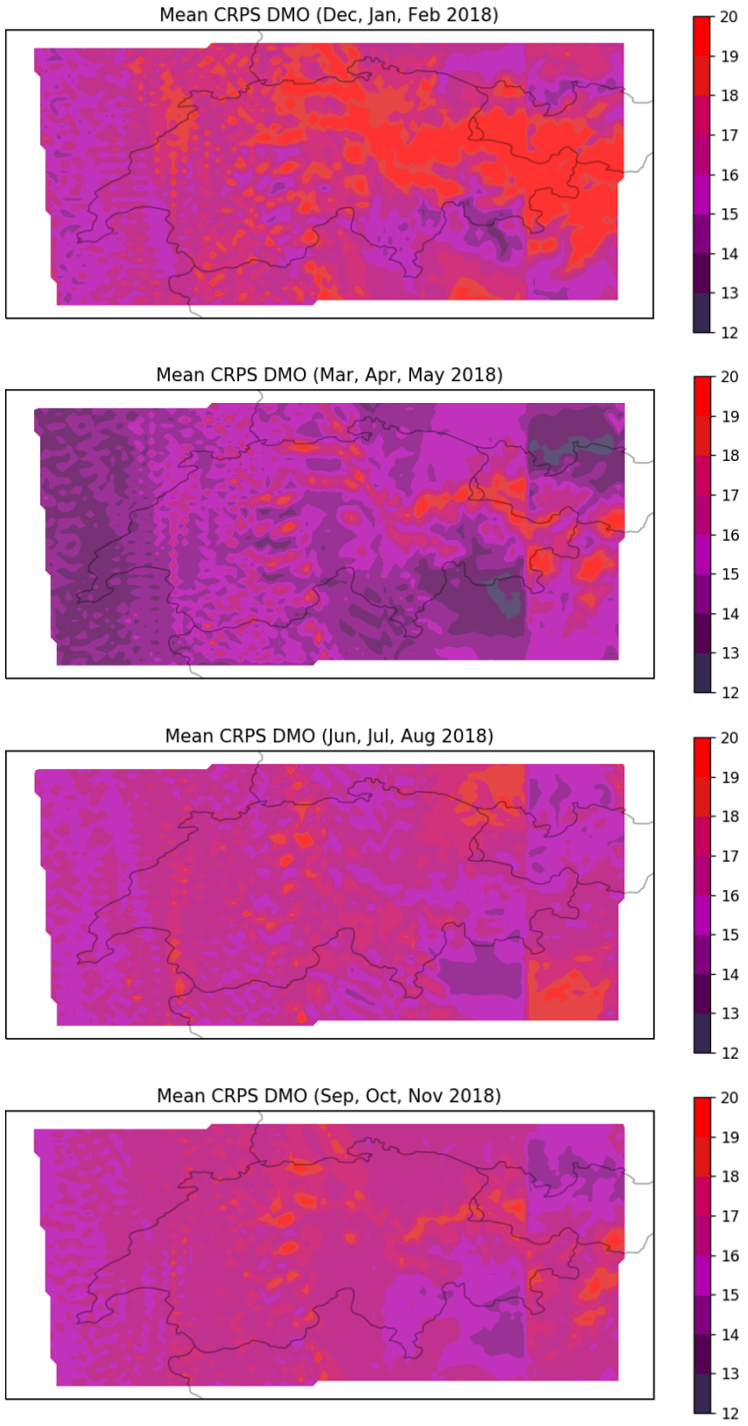

Figure 8: Mean CRPS by location of Direct Model Output for different seasons: winter, spring, summer, and autumn (from top to bottom).

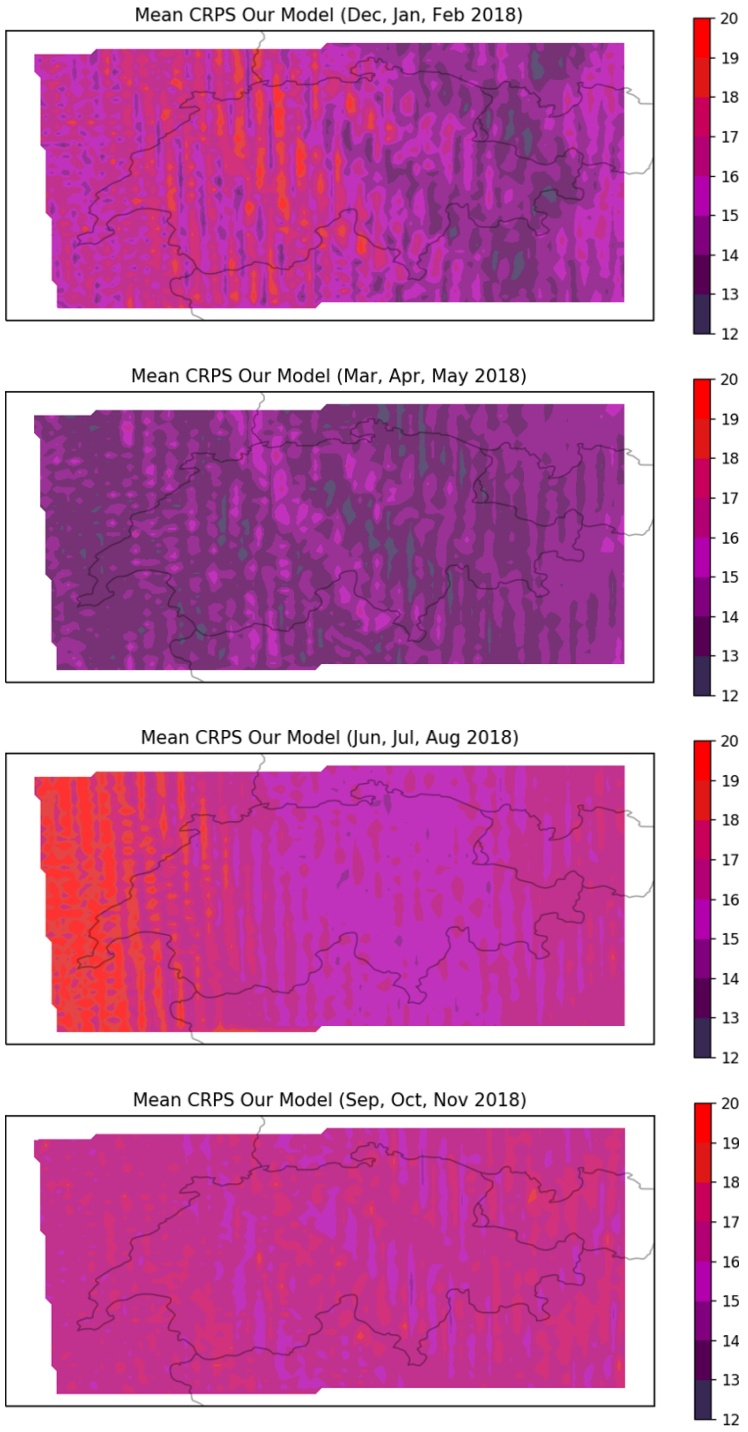

Figure 9: Mean CRPS by location of Our Best Model (7qto21) for different seasons.