Improving rainfall rate estimations with Commercial Microwave Link Signals

in Sri Lanka using Deep Transfer Learning

Ludo Diender

Supervisors: Hidde Leijnse, Ruben Imhoff and Kirien Whan

February 2022

MSc thesis Hydrology and Quantitative Water Management Group
Wageningen University

Abstract

Don't make the abstract too long. See the list of tricks on Blackbord.

Contents

1 | Introduction

1.1 Context and motivation

Having ample and correct precipitation data is important for a plethora of applications, including flood warnings, agriculture, river safety and shipping routes (Chwala & Kunstmann, 2019). In urban areas, an even higher spatial and temporal resolution of rainfall is needed, due to the complex and quickly responding urban hydrological system (Overeem et al., 2011). Due to their high density in populated areas, Commercial Microwave Links (CML) from telecommunication networks could help in retrieving these high-resolution urban precipitation data. CML are back-haul links used by telecommunication companies to transfer information from one telecom station to the next using microwave signals. The links' signals get attenuated by rainfall by means of scattering and absorption. The received signal level is measured and stored by the telecommunication companies for monitoring purposes and can be used to retrieve path-averaged rainfall rates by calculating the attenuation. In the past 25 years, CML have been recognized as a valuable opportunistic method to measure rainfall (Leijnse et al., 2007; Ruf et al., 1996). Especially in data-scarce areas, where little precipitation is measured, CML have proven to be an excellent additional information source for precipitation amounts (Overeem et al., 2021; Doumounia et al., 2014; Diba et al., 2021).

The first studies on the use of CML signals to retrieve rainfall rates were done by using a specific Power Law (PL) to relate the attenuation of the signal and the rainfall rate (Overeem et al., 2011; Leijnse et al., 2007). This method, which includes a wet-dry classification, baseline estimation, wet antenna attenuation estimation and finally a rainfall rate retrieval, has yielded good results in multiple studies (de Vos et al., 2019; Graf et al., 2020; Fencl et al., 2017). Recently, the community has been exploring a more data-driven approach in the form of neural networks of different architectures. Neural networks are a subpart of Machine Learning that is inspired by the neural structure of the brain. A neural network consists of different layers of nodes (neurons) that are connected to each other. If the network contains more than one layer, it is considered a deep network and Deep Learning is used as equivalent naming. A neural network is able to learn the relationship between the input and output, without intervention from a researcher. Studies have been performed in Sweden, Israel (Habi, 2019), Germany (Polz et al., 2020), South Korea and Ethiopia (Diba et al., 2021) on the use of such data-driven networks in relating CML signals to rainfall rates. Previous studies have shown that data-driven models can be more accurate, less time-demanding and more robust in estimating rainfall rates compared to the PL method (Polz et al., 2020; Pudashine et al., 2020). Neural networks are not a novelty in predicting rainfall (French et al., 1992), but the application to CML data has recently gained popularity.

One of the disadvantages of using data-driven methods like neural networks, is the dependency on a large training data set. Neural networks need this large dataset to learn the relationships between input and output. In areas with less or little available training data, transfer learning provides the opportunity to adapt an already existing model with a certain structure to do a slightly different task (Tan et al., 2018). The concept of transfer learning is based on the way humans learn. When humans learn a new task, we do not start from scratch. We use previous knowledge and skills to quickly adapt to the new task. When learning to ride a motorcycle, it helps if you already know how to ride a bike. Concepts like balance and steering can be transferred from the latter task to the former, thus speeding up the learning process. The technique exploits the availability of data in the source domain and is able to transfer that knowledge to the target domain.

It does so by relaxing the underlying assumption that training and test data for a Machine Learning model should be independent and identically distributed (Weiss, Khoshgoftaar, & Wang, 2016). A conceptual picture of transfer learning can be seen in figure **??**. Although used quite often in different applications (Zhuang et al., 2021), transfer learning concerning environmental sciences has so far primarily been applied in wind farm modelling (Hu et al., 2016; Qureshi et al., 2017). The potential of transfer learning in CML rainfall estimations has not been studied before.

Recent research focused on the use of CML data to measure rainfall in tropical regions, more specifically Sri Lanka (Overeem et al., 2021) and Brazil (Gaona et al., 2017a). The relatively small amount of reference rain gauges in Sri Lanka especially made this research more challenging compared to well-equipped countries

like the Netherlands (Overeem et al., 2013). Both of the two studies mentioned above (Sri Lanka and the Netherlands) are based on the PL method. There have not been any efforts yet to analyze the potential of CML for rainfall retrieval using data-driven methods for neither the Netherlands nor Sri Lanka. This research aims to fill this knowledge gap by creating a neural network used for CML rainfall retrieval in the Netherlands and by studying the potential of transer learning within the Dutch context.

1.1.1 Relevance of precipitation data

text text text

1.1.2 Current methods for precipitation data

text text text

1.1.3 Commercial Microwave Links Advantages

text text text

1.1.4 Opportunity for AI/DL models to improve estimations

text text text

1.1.5 How do DL models work (briefly)

text text text

1.1.6 Shortcomings of DL models

text text text

1.1.7 Transfer learning to improve predcitions

text text text

1.2 Research questions

This study focusses on two main research questions. 1) How does a Neural Network perform on estimating rainfall rates from Dutch Commercial Microwave Link data? 2) How can the concept of transfer learning be applied to improve this performance?

1.3 Thesis contents

This thesis starts with a description of the data that is used (Section 2). In Section 3 the methodology of this study is described. This includes an explanation on Neural Networks, data preprocessing, transfer learning and the set-up of this model study. Afterwards, in Section 4 the results are presented. These results and the approach used in this study are discussed in Section 4.3, which is followed by the conclusion in Section 5.

2 | Field site and data

Some text here about how the field site and data are structured?

2.1 Commercial Microwave Link data

2.1.1 Netherlands

Location of the links First paragraph text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text

Static vs dynamic data First paragraph text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text

Freq,temp res and timespan First paragraph text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text

2.1.2 Sri Lanka

Location of the links First paragraph text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text

Static vs dynamic data First paragraph text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text

Freq,temp res and timespan First paragraph text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text

2.2 Reference precipitation data

2.2.1 Netherlands

Obtained from (provider) First paragraph text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text

Temporal and spatial resolution First paragraph text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text

Quality and availability First paragraph text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text

2.2.2 Sri Lanka

Obtained from (provider) First paragraph text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text

Temporal and spatial resolution First paragraph text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text

Quality and availability First paragraph text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text

3 | Methods

This chapter describes the methodology of this study. The chapter starts with a general explanation on Neural Networks and their functioning, to get acquainted with terms and parameters in Section 3.1. Section 3.2 deals with the preprocessing of the input data to the model and subsequently model selection is described in Section 3.3. The combination of Neural Networks and the input data in this study is discussed in Section (»»). Section 3.4 is concerned with the validation of the model results and performance of the model related to the benchmark method RAINLINK. The final part of this chapter deals with transfer learning and how this is applied in this study (Section 3.5)

3.1 Neural network explanation

3.1.1 Goal and concepts of neural networks

Neural networks are a subset of models within the Machine Learning (ML) environment. All models within the ML space are data-driven, i.e. without pre-specified relationships between input and output. These data-driven models aim to figure out this relationship via self-learning techniques. The biggest advantages of such models are that 1) they are very versatile: they can adapt to various circumstances and can be used in a wide range of applications, and 2) they can learn relationships that we do not have a physical explanation for yet. Through multiple iterations, the models can improve by minimizing or maximizing a loss function. The loss function can be interpreted as a score for the model: the worse the score, the more the model needs to adjust itself (see Section 3.1.3).It improves by optimizing parameters of differential equations in the model. Similar to physically-based models, it is up to the modeler to determine when the model performs well enough. The self-learning ML models will stop learning after a certain criterion is met, or when a specified number of iterations has passed.

Neural networks are inspired by the structure of the human brain with neurons and synapses, hence the name Neural network. The network consists of one or multiple layers of nodes that are connected with each other. When the network has more than one layer, it is called a Deep Neural Network and it is referred to as Deep Learning. Every single connection between nodes (or between nodes and input/output) is represented by a weight and a bias. These weights and biases are the bread and butter of neural networks. They determine the underlying relationships between different parts of the input signal, the connections within the model and the relation to the final output. Training a neural network is all about adjusting the weights and biases in such a way that the desired output is created. At every node, a weighted sum is calculated based on these weights and biases and it is passed to an activation function. This activation function yields one value, which will be passed to the following nodes in the model. Because of the very general structure of neural networks, they form a very versatile set of models. Research efforts in neural networks span from image classification and speech recognition to time series prediction and weather forecasts.

3.1.2 Neural network architectures

The versatility of neural networks allows for a wide range of different model architectures to choose from. Every architecture has its own strengths and weaknesses. For image recognition for example, the most common type of neural network is a Convolutional Neural Network (CNN). In time series prediction, the most common architecture is the Recurrent Neural Network (RNN), which is therefore most suited for precipitation estimation using Commercial Microwave Links.

RNNs are looping-based architectures, which make them ideal for dealing with sequences. The core unit of RNNs, the Recurrent Unit (the lightgreen blocks in Figure 3.1), is used multiple times. The output of the unit is added to the input of the next step in the same recurrent unit. This creates an architecture that is well suited for time series with sequential data patterns. How many layers of these recurrent units are used in a model depends on the modeler and can be tuned (see Section 3.3)

In this study, a default RNN was used for the rainfall rate estimation. A many-to-one architecture is used, which means that a sequence of signal levels is used to predict one rainfall rate at a specific timestep.

3.1.3 Learning process of neural networks

The self-learning ability of Neural Networks is mainly relying on the loss function. The model calculates the error between its output and the target output and tries to adjust accordingly. The selected loss function deter-

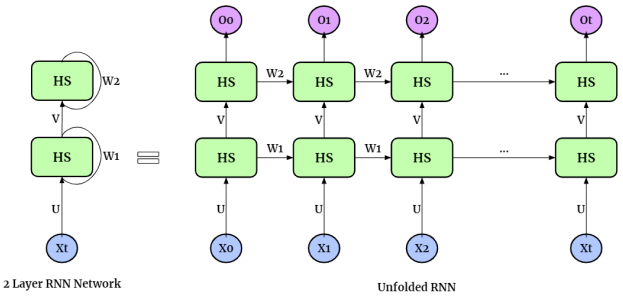

Figure 3.1: *Schematic representation of a two-layer unfolded RNN. HS represents the hidden state,X is the input, O is the output and U, V and W are weight and bias vectors associated with the model*

mines amongst others the behavior of the model. In this study, a standard Mean Square Error (MSE) loss function is used (see Eq. 3.1).

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y})^2 \tag{3.1}$$

where y_i is the predicted 15-min interval precipitation and $\hat{y}$ is the observed 15-min interval precipitation. MSE as a common loss function has been used before in similar studies (Pudashine et al., 2020; Diba et al., 2021) The effect of this choice and the relation with the inherently skewed nature of precipitation distributions will be discussed later on in this thesis. (SIDENOTE FOR THESIS RING: Ultimately, a different metric will be tried as well to see if the predictions improve by using a different error metric to train the model. A statistical metric that puts more emphasis on higher rainfall rate values could outperform the standard MSE metric.)

To update and improve the model prediction, a propagation algorithm is selected. Such a propagation algorithm is a mathematical set of equations that determines how the model learns. In this study, Stochastic Gradient Descent (SGD) will be used. SGD is one of the most commonly used algorithms to improve neural networks. SGD is based on a gradient descent algorithm to optimize the loss function In simple terms: the method is looking for an optimum (minimum or maximum) per weight. Based on this gradient, the weight is adjusted up or down to minimize the loss function (MSE in this study). As calculating the full gradient field for all weights present in the model is a tedious task, SGD takes a randomly selected subset of the data and uses that to calculate the gradients. This might lead to more steps, but the steps are less computationally intensive and typically lead to an improved performance. The SGD method is back-propagated. This implies that the final layer of the network is updated first and the learning process works itself back to the start of the model. In this study, the fully connected layer at the end of the RNN is updated first before the model improves and adjusts in the recurrent unit itself. A back-propagated model is chosen in this study since it is the most common type of neural network, easy to implement, fast, flexible and generally good.

3.1.4 Available hyperparameters

The versatility of Neural Networks is generally perceived as one of the biggest advantages of these type of models. However, it does leave the modeller with a lot of options to choose from in creating a Neural Network for their specific task. The parameters that refer to the architecture and set-up of the Neural Network are called hyperparameters. Table 3.1 highlights the available hyperparameters in this model. The hyperparameters can be divided in three groups: Structure, Data and Effiency. Structure hyperparameters relate to the structure of the model: how is it designed and how often will it run. Data parameters are related to the input data into the model; the type of data and the goal with the data determine these hyperparameters. The Efficiency group entails hyperparameters that relate to the time it takes the model to run. In this group, there's always a balance between performance and computational intensity.

The process of calibrating hyperparameters in order to find an optimal combination (if it exists) is refered to as tuning. Not all hyperparameters were automatically tuned. Only the Structure hyperparameters were tuned, as the other two hyperparameter groups need to be determined based on the input data and the available computational resources. More details on the automatically tuned Structure hyperparameters can be found in Section 3.3.1. The Data and Efficiency group of hyperparameters are set based on convenience and literature study and will be elaborated on in Section 3.3.2.

3.2 Data preprocessing

3.2.1 Connecting CML and reference data

To train the Neural Network, a feature set and a target set is needed. For every link in the CML dataset, the corresponding amount of precipitation has been determined. To obtain this value, the middle point of each link has been determined and connected to the corresponding radar cell. This midpoint method is commonly

Table 3.1: Available hyperparameters in this study.

Group	Hyperparameter	Description
Structure	Hidden size	Size of the intermediate hidden output. Higher hidden size allows for more complex relations
	Learning rate	A multiplication factor that influences the speed at which the weights in the model are updated with each SGD step. A high learning rate makes the model learn faster, but also makes it more prone to overfitting.
	Number of epochs	The number of iterations the model performs to reach the final prediction. More epochs improve the performance of the model but take sginificantly longer to run.
	Number of layers	Total number of layers in the model. A higher number of layers creates a model that can better identify complex dependencies in the data.
Data	Sequence length	Length of datasample used to predict one target value.
	Loss function	Metric that is used to judge the model performance on. Can be tailored towards the goal of the model.
Efficiency	Activation function	Function within each node that determines the value of the node as a consequence of the weighted average
	Batch size	The size of each batch of data that is fed to the model. The larger the batch size, the faster the model will be able to learn, but the less possiblities it has to update its weights and biases.
	Learning algorithm	The algorithm that is used to update the weights and biases of the model internally.
	Scheduler	The algorithm that is used to run multiple combinations of hyperparameters simultaneously and decides when to quit with some of the runs

used in CML research (refs). More elaborate methods have been used by Leijnse et al. (2007) for example, where a weighted sum of all cells that the link passes through was created. The impact of this decision will be discussed further on in this thesis.

3.2.2 Error filtering

A large part of the RAINLINK algorithm consists of error filtering. Because the algorithm cannot distinguish a signal distortion by rain or by a bird, this filtering is necessary. The spirit of Deep Learning models is that these errors don't need to be filtered out manually. Similar to Habi et al., just the 'no data' values have been removed from the dataset. The model is expected to handle other noise in the data by learning the patterns and adjust accordingly.

3.2.3 Data splitting and scaling

A Deep Learning model with a varying set of hyperparameters needs three independent datasets to run. The training dataset is used to improve the models weights and parameters and is fed through the SGD algorithm, in which the model is updated in a backwards propagating fashion. The testing dataset is subsequently used to independently judge the capabilities of the model. After training, this set is used to calculate an error metric to see how well the model is doing. This error metric (MSE in this study) is used to update the weights in the model. An independent dataset is needed to prevent overfitting of the model, a very common problem in Machine Learning models. As different models with different combinations of hyperparameters are tested, a third independent dataset is needed to check how well the best model out of this sample set performs. Again, the main goal is to prevent overfitting.

In creating the three separate datasets, multiple approaches can be taken. The core of the split is based on the IID principle (Independent and Identically Distributed):

- There is little to no information leaked from one set to the other (truly independent datasets)

- There is enough data in each of the sets to cover a wide range of events and situations (comparable datasets and thus assumed identically distributed)

The data for the Netherlands has been split in a 40/40/20 way of training, testing and validating respectively. Previous studies use varying ratios in splitting the data, with the one common aspect being a smaller validation set compared to the trainig and testing (Polz et al., 2020; Diba et al., 2021; Pudashine et al., 2020). The year 2011 is used for training, 2012 for testing and 2013 (up until July 21st) for validating. By taking full years for training and testing, all seasons are incorporated and the datasets are assumed to be closer to being identically distributed. NOTE ABOUT 2011, WHICH HAD A VERY VERY DRY SPRING. There is a tradeoff in the IID assumption that created the split as decided upon in this study. Because of the dry spring, the years 2011 and 2012 are not identical. However, wherever the two datasets meet in time, there is a chance of information leaking from one dataset to the other. Therefore, this split by full years was still prefered. For the validation set, half a year is taken due to limitations in data availability. As the validation set is only used to judge the final model, and not to update the model anymore, this dataset can be smaller as in (LITERATURE).

Having large outliers and ranges in the input data can cause the SGD algorithm to have exploding gradients. Therefore, a scaling has been applied to the features and targets. Similar to (Kilmartin & Peterson, 1972), the precipitation data were log-transformed. Afterwards, they were scaled with the minimum and maximum per dataset, to create a value between 0 and 1. A similar scaling has been applied to the features (RSL), as they are scaled to their min and max value. The different datasets are scaled differently. They all fit between 0 and 1, which is convenient for the SGD algorithm. If all datasets were scaled in the exact same way, it would inevitably lead to data leakage from one dataset to the other, violating the IID assumption as discussed before.

3.3 Model selection

3.3.1 Automated hyperparameter tuning

A full grid search on all hyperparameters available to find the best combination is computationally infeasible. Therefore, this study uses 25 random combinations of hyperparameters, all within a limited range of values. This number of combinations is chosen to have a trade-off between computational intensity and having ample combinations to ensure a decent search. Selecting the combinations is done using the RayTune package, which is a tuning-specific Python package that interacts with PyTorch (used in this study) and other Deep Learning packages. This study uses the Async Successive Halving Algorithm Scheduler (ASHA) to effectively search the hyperparameter space. The ASHA Scheduler is able to effectively implement parallelism (to decrease runtime) and a hard early stopping mechanism to terminate runs if they are not amongst the best few models after a while (Li et al., 2018). The ASHA Scheduler is the most common scheduler available in RayTune and performs well in most cases when no specific instructions regarding hyperparameter tuning are needed. The random combinations allow for quickly identifying the most sensitive hyperparameter and tune it specifically if needed. The best combination of hyperparameters is used to calculate the model performance based on the final independent validation dataset. The ranges for the calibrated hyperparameters can be found in Table 3.2

Table 3.2: Ranges for the automatically tuned hyperparameters using RayTune.

Hyperparameter	*Range*
Hidden size	$[64, 128, 256]$
Learning rate	$1e^{-6} - 1e^{-3}$
Number of epochs	$(1 - 20)$
Number of layers	$[2, 4, 8, 16]$

3.3.2 Non-automated hyperparameter tuning

Hyperparameters that are related to the data rather than the model itself are not tuned automatically. Hyperparameters in the Data group are tuned, but in a slightly less rigid way as it is easier to deduct which values should perform well on the input data. The sequence length is partly determined by how long rainfall events typically last in the Netherlands and their autocorrelation. As a starting value, a sequence length of 4 (15-min) timesteps is used. This means that the past hour is used to predict the rainfall rate at the current time step. Other larger sequence lengths will be explored to see whether this improves the performance of the model. (SIDENOTE FOR THESISRING: number

still needs to be explored) The loss function has a direct relation to the distribution of the input data. As precipitation data is not normally distributed, a standard MSE loss function might put more emphasis on keeping the prediction lower and therefore missing out on large precipitation events. However, a slightly different adjusted loss function is used as well (STILL SEE WHICH ONE MIGHT BE USEFUL) to see if that improves the results on the independent validation set.

3.4 Evaluation metrics and summary statistics

3.5 Transfer learning

The act of Transfer learning relies on using a pretrained model for a different task. In this study, the pretrained model on the Netherlands will be used to estimate rainfall rates in Sri Lanka. Transfer learning of RNN models can be done in two different ways, which will shortly be discussed in the following sections.

3.5.1 Architecture transfer

The architecture of neural networks is not fixed. Finding the optimal architecture for the job requires a lot of training data and three datasets under the IID assumption. Since the data in Sri Lanka is limited, a first architecture transfer would include keeping all the hyperparameters as they are tuned for the Netherlands. This removes the need of 1 dataset and therefore can increase the training material for the neural network. In this type of transfer, the weights and biases are not transfered and will need to be learned again.

3.5.2 Weights transfer

A weights transfer is an extension on the previous type of transfer. The weights and biases as trained for the Netherlands will directly be transferred to the Sri Lankan situation. This requires a similar architecture as well, as the number of weights that is available depends on e.g. the number of layers of the model and the hidden size. This method would assume that the precipitation patterns in the Netherlands and Sri Lanka are similar to a large extent, which is not valid assumption.

3.5.3 Partly weights transfer

An intermediate method is to partly transfer the weights. At the end of the RNN, a fully-connected layer translates the hidden output inside the model to a final prediction of the amount of rain per time step. By transfering all weights and biases except for those related to this fully-connected layer allows the model to base its prediction on Dutch features, while still being flexible to adjust to Sri Lankan situations. The general features of the time series are extracted by the transfered part, the final translation still needs to be trained. In this version of transfer learning, the data still needs to be split in a training and a test set.

4 | Results

4.1 The Netherlands

Ideas: table with different hyperparameter sets and the best one out of all these.

Performance on training, testing and validation set for the best one

Scatterplot: predicted vs estimated precipitation for different loss functions

Summarizing table with statistics on RAINLINK vs My Model

4.2 Transfer learning to Sri Lanka

Ideas: table with different hyperparameter sets and the best one out of all these.

Performance on training, testing and validation set for the best one

Scatterplot: predicted vs estimated precipitation for different loss functions

Summarizing table with statistics on RAINLINK vs My Model

4.3 Discussion

Including more information like neighbouring links

Trying out different types of NN and different hyperparameters: thousands to choose from

5 | Conclusion

Does my model outperform RAINLINK in the NL? DL models have potential but there is a lot a lot to customize

Acknowledgements

Don't forget to thank the people who gave you data!

References

Chwala, C., & Kunstmann, H. (2019). Commercial microwave link networks for rainfall observation: Assessment of the current status and future challenges [Journal Article]. *WIREs Water*, *6*(2), e1337. doi: https://doi.org/10.1002/wat2.1337

de Vos, L. W., Overeem, A., Leijnse, H., & Uijlenhoet, R. (2019). Rainfall estimation accuracy of a nationwide instantaneously sampling commercial microwave link network: Error dependency on known characteristics [Journal Article]. *Journal of Atmospheric and Oceanic Technology*, *36*(7), 1267-1283. doi: 10.1175/JTECH-D-18-0197.1

Diba, F., Samad, M., Ghimire, J., & Choi, D. (2021). Wireless telecommunication links for rainfall monitoring: Deep learning approach and experimental results [Journal Article]. *IEEE*, *9*, 11. doi: 10.1109/ACCESS.2021.3076781

Doumounia, A., Gosset, M., Cazenave, F., Kacou, M., & Zougmore, F. (2014). Rainfall monitoring based on microwave links from cellular telecommunication networks: First results from a west african test bed [Journal Article]. *Geophysical Research Letters*, *41*(16), 6016-6022. doi: https://doi.org/10.1002/2014GL060724

Fencl, M., Dohnal, M., Rieckermann, J., & Bareš, V. (2017). Gauge-adjusted rainfall estimates from commercial microwave links [Journal Article]. *Hydrol. Earth Syst. Sci.*, *21*(1), 617-634. (HESS) doi: 10.5194/hess-21-617-2017

French, M. N., Krajewski, W. F., & Cuykendall, R. R. (1992). Rainfall forecasting in space and time using a neural network [Journal Article]. *Journal of Hydrology*, *137*(1), 1-31. doi: https://doi.org/10.1016/0022-1694(92)90046-X

Gaona, M., Overeem, A., Raupach, T., Leijnse, H., & Uijlenhoet, R. (2017a). Rainfall retrieval with commercial microwave links in sao paulo, brazil [Journal Article]. *Atmos. Meas. Tech.*, *11*(7), 11. doi: 10.5194/amt-11-4465-2018

Graf, M., Chwala, C., Polz, J., & Kunstmann, H. (2020). Rainfall estimation from a german-wide commercial microwave link network: optimized processing and validation for 1 year of data [Journal Article]. *Hydrol. Earth Syst. Sci.*, *24*(6), 2931-2950. (HESS) doi: 10.5194/hess-24-2931-2020

Habi, V. (2019). *Rain detection and estimation using recurrent neural network and commercial microwave link* (Thesis).

Hu, Q., Zhang, R., & Zhou, Y. (2016). Transfer learning for short-term wind speed prediction with deep neural networks [Journal Article]. *Renewable Energy*, *85*, 83-95. doi: https://doi.org/10.1016/j.renene.2015.06.034

Kilmartin, R. F., & Peterson, J. R. (1972). Rainfall-runoff regression with logarithmic transforms and zeros in the data [Journal Article]. *Water Resources Research*, *8*(4), 1096-1099. Retrieved from `https://dx.doi.org/10.1029/wr008i004p01096` doi: 10.1029/wr008i004p01096

Leijnse, H., Uijlenhoet, R., & Stricker, J. N. M. (2007). Hydrometeorological application of a microwave link: 2. precipitation [Journal Article]. *Water Resources Research*, *43*(4). doi: https://doi.org/10.1029/2006WR004989

Li, L., Jamieson, K., Rostamizadeh, A., Gonina, K., Hardt, M., Recht, B., & Talwalkar, A. (2018). *Massively parellel hyperparameter tuning (unpublished)* [Conference Paper].

Overeem, A., Leijnse, H., Leth, T., Bogerd, L., Priebe, J., Tricarico, D., ... Uijlenhoet, R. (2021). Tropical rainfall monitoring with commercial microwave links in sri lanka [Journal Article]. *Environmental Research Letters*, *16*. doi: 10.1088/1748-9326/ac0fa6

Overeem, A., Leijnse, H., & Uijlenhoet, R. (2011). Measuring urban rainfall using microwave links from commercial cellular communication networks [Journal Article]. *Water Resources Research*, *47*(12). doi: https://doi.org/10.1029/2010WR010350

Overeem, A., Leijnse, H., & Uijlenhoet, R. (2013). Country-wide rainfall maps from cellular communication networks [Journal Article]. *Proceedings of the National Academy of Sciences*, *110*(8), 2741-2745. doi: 10.1073/pnas.1217961110

Polz, J., Chwala, C., Graf, M., & Kunstmann, H. (2020). Rain event detection in commercial microwave link

attenuation data using convolutional neural networks [Journal Article]. *Atmos. Meas. Tech.*, *13*, 18.

Pudashine, J., Guyot, A., Petitjean, F., Pauwels, V. R. N., Uijlenhoet, R., Seed, A., ... Walker, J. P. (2020). Deep learning for an improved prediction of rainfall retrievals from commercial microwave links [Journal Article]. *Water Resources Research*, *56*(7), e2019WR026255. doi: https://doi.org/10.1029/2019WR026255

Qureshi, A. S., Khan, A., Zameer, A., & Usman, A. (2017). Wind power prediction using deep neural network based meta regression and transfer learning [Journal Article]. *Applied Soft Computing*, *58*, 742-755. doi: https://doi.org/10.1016/j.asoc.2017.05.031

Ruf, C. S., Aydin, K., Mathur, S., & Bobak, J. P. (1996). 35-ghz dual-polarization propagation link for rain-rate estimation [Journal Article]. *Journal of Atmospheric and Oceanic Technology*, *13*(2), 419-425. doi: 10.1175/1520-0426(1996)013<0419:Gdpplf>2.0.Co;2

Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C., & Liu, C. (2018). A survey on deep transfer learning [Conference Proceedings]. In (p. 270-279). Springer International Publishing.

Weiss, K., Khoshgoftaar, T. M., & Wang, D. (2016). A survey of transfer learning [Journal Article]. *Journal of Big Data*, *3*(1), 9. doi: 10.1186/s40537-016-0043-6

Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., ... He, Q. (2021). A comprehensive survey on transfer learning [Journal Article]. *Proceedings of the IEEE*, *109*(1), 43-76. doi: 10.1109/JPROC.2020.3004555

A | Additional figures