

Improving Surface Soil Moisture Estimation with Distributed Deep Learning and HPC

T. Curtis*, M. Riedel*[†], H. Neukirchen*, J. Busch[†], C. Montzka[‡], M. Aach[†], R. Hassanian*, C. Barakat[†]

* School of Engineering and Natural Sciences, University of Iceland, Iceland

[†] Jülich Supercomputing Centre, Forschungszentrum Jülich, Germany

[‡] Institute of Bio- and Geosciences: Agrosphere (IBG-3), Forschungszentrum Jülich, Germany

thorarnar@hi.is, morris@hi.is, j.busch@fz-juelich.de, c.montzka@fz-juelich.de, m.aach@fz-juelich.de, seh@hi.is

Abstract—Soil Moisture (SM) is crucial for land surface hydrology, impacting agriculture, ecology, wildlife, and public health. This paper explores the use of Machine Learning (ML) and Deep Learning (DL) models for SM data analysis and estimation, leveraging High-Performance Computing (HPC) for efficient model training and hyperparameter tuning. The research was performed with a commercial partner and SM experts. However, utilising HPC environments requires knowledge of many low-level HPC modules (e.g., various application libraries or vendor-specific drivers like Nvidia’s CUDA, NCCL, etc.) and their specific versions needed for interoperability. Such challenges can be overcome using the Unique AI Framework (UAIF) developed in The European Center of Excellence in Exascale Computing “Research on AI- and Simulation-Based Engineering at Exascale” (CoE RAISE) project. The research in this paper contributed to the co-design of several UAIF components using HPC to search for the optimal hyperparameter setup for each ML model. It compares Artificial Neural Networks (ANNs) and Recurrent Neural Networks (RNNs) to a baseline Random Forest (RF) model. Our result demonstrates a significant improvement in accuracy through distributed learning and systematic hyperparameter tuning. The findings suggest that HPC-driven DL can offer scalable, high-resolution SM predictions while leveraging the UAIF by application-domain-specific experts (e.g., SM experts, etc.), enabling easier use of HPC.

Keywords—Soil Moisture Estimation; High-Performance Computing; Machine Learning; Distributed Deep Learning

I. INTRODUCTION

Knowledge of Soil Moisture (SM) is valuable in many fields to both scientific and commercial users. When studying the Earth’s climate and weather, SM is a crucial state variable in the complex global cycles of water, energy, and carbon. Natural hazards like drought, floods, and landslides can be better understood with the help of prediction models, where SM is a crucial variable. In agriculture, SM data can be vital to crop yield, and where irrigation is possible, it allows for the most efficient water usage. The global food and energy demand is increasing, making crop yield

monitoring essential for food security. In many countries, crops grow without irrigation, and the pre-harvest yield prediction has become fundamental for supporting export-import policies [1]. Operationally available soil moisture products exist, but they suffer from only offering medium to coarse spatial resolution ($\geq 1km$). Satellite Remote Sensing (RS), on the other hand, presents the possibility of high spatial resolution continuous measurement of surface SM [2]. For model training, globally distributed in situ data from the International Soil Moisture Network (ISMN) can be used as ground reference [3].

Leo Breiman [4] introduced the Random Forest (RF) model, an ensemble learning technique that can be applied to both regression and classification tasks, in the 2000s. The RF model serves as the baseline model in this study, and the results are compared to new Machine Learning (ML) and Deep Learning (DL)-based approaches consisting of both Artificial Neural Network (ANN) and Recurrent Neural Network (RNN) models.

ANNs are ML models inspired by the neuronal structure of human brains. An RNN is a more advanced feed-forward Neural Network (NN) that can handle input sequences of varying lengths. Long Short-Term Memory (LSTM) models and Gated Recurrent Units (GRUs) were used in this study. While the training of the RF model is relatively fast and does not necessarily require the use of High-Performance Computing (HPC), the training of Deep Neural Networks (DNNs) [5] is a significant computing task that places high demands on computing resources. Model training is significantly accelerated using Graphics Processing Units (GPUs), particularly when dealing with massive amounts of data. Using many GPUs simultaneously in a distributed fashion to train DNNs has also been shown to provide super-linear speedups [6] without harming the ML and DL model accuracy.

Similar to DNN training, hyperparameter tuning is a computationally expensive and time-consuming process that can greatly benefit from the power of HPC. However, due to the heterogeneous nature of the HPC systems available to researchers and scientists through the EuroHPC Joint Undertaking¹ with respect to their underlying software and hardware technologies, it is a complex task for users (e.g., SM domain experts) to seamlessly use

This work was performed in the Center of Excellence (CoE) Research on AI- and Simulation-Based Engineering at Exascale (RAISE) receiving funding from EU’s Horizon 2020 Research and Innovation Framework Programme H2020-INFRAEDI-2019-1 under grant agreement no. 951733. Icelandic HPC National Competence Center is funded by the EuroCC-1 projects that has received funding from the EU HPC Joint Undertaking (JU) under grant agreement No 951732. Parts of the work have been also supported by the European Digital Innovation Hub (EDIH) of Iceland (EDIH-IS) funded in parts by the Digital Europe Programme. Finally, parts of this work were performed in the SMITH Project receiving funding from the German Federal Ministry of Education and Research.

¹https://eurohpc-ju.europa.eu/index_en

these systems and to easily port their applications between them. With the introduction of the Unique AI Framework (UAIF) [7] Load AI Modules, Environments, and Containers (LAMEC)² job script generator this has become a simpler task, with the interface enabling streamlined access to HPC systems by abstracting from low-level software versions and module environment details. It contains job scripts for specific HPC systems with the correct configuration of modules needed to use specific UAIF AI tools (i.e., no low-level information is needed by users).

The Cross-Industry Standard Process for Data Mining (CRISP-DM)³ is an open standard process model widely used by data mining professionals, and is also increasingly used for ML and DL projects. Its structured approach was used throughout this paper.

This paper is structured as follows. After the introduction in Section 1, Section 2 gives an overview of related work. Section 3 describes the methodology, Section 4 then describes the business understanding, data understanding, data preparation, and modeling phases using the CRISP-DM process model. Section 5 contains the evaluation phase of the CRISP-DM model. The paper ends with some concluding remarks and conclusions in Section 6.

II. RELATED WORK

There is a lot of published work available on the topic of soil moisture estimation using RS data, some that relate to this paper are briefly discussed below.

Bauer-Marschallinger et al. [8] presented a method to retrieve surface soil moisture from the Sentinel-1 satellites, which carry CSAR sensors. The method adapted well-established change detection algorithms to build the first globally deployable soil moisture observation data set with a resolution of 1 km.

Greifeneder et al. [3] presented a ML-based approach for surface soil moisture estimations with Google Earth Engine (GEE). This is the method behind the SM product provided via the Copernicus Land Monitoring Service⁴. The study introduces an ML-based approach for the high-spatial resolution (50 m) mapping of soil moisture based integrating Landsat-8 optical and thermal data, Copernicus Sentinel-1 C-band Synthetic Aperture Radar (CSAR) data, and modelled data, executable in GEE.

Rudner et al. [9] propose a novel approach for rapid segmentation of flooded buildings by fusing multiresolution, multisensor, and multitemporal satellite imagery in a Convolutional Neural Network (CNN). The method uses multiple streams of encoder-decoder architectures to extract spatio-temporal information from medium-resolution images and spatial information from high-resolution images. The resulting representations are then fused into a single medium-resolution segmentation map.

Most recently, Fan et al. 2025 published the first global SM data set at a resolution of 1km, based on a soil scattering and vegetation water cloud model [10].

Hughes et al. [11] present a three-step framework for sparse image matching of Synthetic Aperture Radar (SAR) and optical imagery. The first step involves predicting regions in each image that are most suitable for matching. Then a correspondence heatmap is generated through a multi-scale, feature-space cross-correlation operator. The final step involves removing outliers by classifying the correspondence surface as a positive or negative match. This approach can help overcome the inaccuracy in the geolocalization of optical images, caused by the propagation of angular measurement errors.

III. METHODOLOGY

A. CRISP-DM

During the data analysis for this paper, CRISP-DM was applied to systematically address business objectives, from understanding the problem and preparing the data to building, evaluating and deploying the model. An overview of the model is shown in Figure 1.

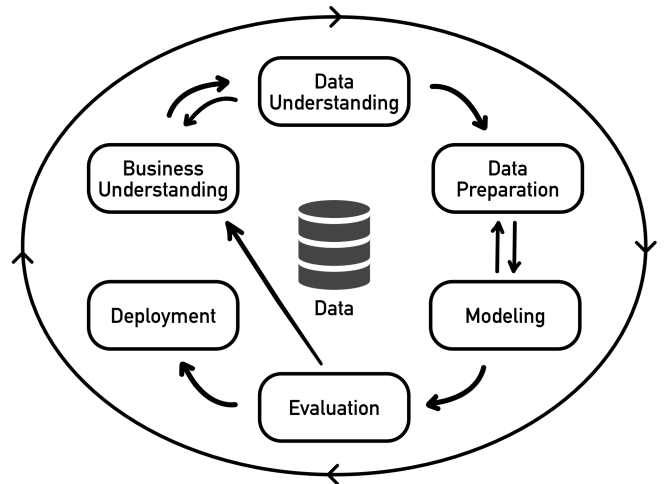


Fig. 1. An overview of the CRISP-DM process model.

The first phase, *Business Understanding*, focuses on understanding the objectives and requirements of the project. The *Data Understanding* phase adds to the foundations of the previous phase and focuses on initial data collection, describing the data, data exploration, and verifying data quality. The *Data Preparation* phase is where various data sets are prepared for modeling. The *Modeling* phase covers the modeling techniques to be used, generation of the test design, model building and assessment. The *Evaluation* phase focuses on model evaluation and looking more broadly at which model best meets the business and what the next steps are. The final phase is *Deployment*, consisting of planning deployment, monitoring and maintenance, producing the final report, and project review.

Alternative data science process models were also considered, such as Knowledge Discovery in Databases

²<https://apps.fz-juelich.de/jsc/lamec/>

³<https://www.datascience-pm.com/crisp-dm-2/>

⁴<https://land.copernicus.eu/en>

(KDD) [12], Sample, Explore, Modify, Model, and Assess (SEMMA)⁵ and Agile Data Science⁶. Each has its strengths, but CRISP-DM was chosen as it provides a balanced and adaptable framework well suited to the interdisciplinary nature of this research and has been used before by the authors [13].

B. Deep Learning Networks for Sequential Data

An RNN is an extended version of a conventional feed-forward ANN that supports input sequences of variable length such as within time-series datasets. This is done by using a recurrent hidden state (memory) with the activation at each time being dependent on that of the previous time. The two more recently introduced model types of RNNs used in this study are LSTMs and GRUs.

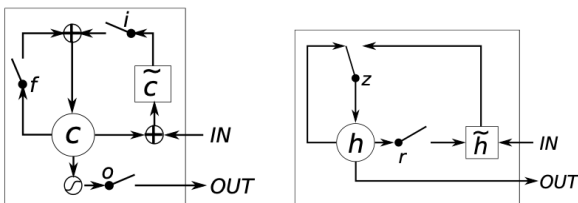


Fig. 2. An LSTM unit (left) and a GRU unit (right) [14].

LSTMs can process entire data sequences and are therefore well-suited to making predictions based on time series data. An LSTM unit is commonly composed of a cell, an input gate, an output gate, and a forget gate. While the gates regulate the flow of information into and out of the cell, the cell itself remembers values over time intervals. Figure 2 (left) shows a typical LSTM unit. Standard RNNs have difficulty bridging more than 5-10 time steps due to the vanishing gradient problem [15]. LSTMs deal with this problem using a unique additive gradient structure that has direct access to the forget gate’s activations [16].

A gating mechanism for RNNs was first introduced in 2014 by Cho et al [17]. GRUs are like LSTMs, but they address a weakness of LSTMs where the internal state values may grow indefinitely, causing the network to break down. That is achieved with an added forget gate, allowing the cell to reset its state [18]. GRUs have fewer parameters than LSTMs as they have no output gate but have been shown to outperform LSTM units in terms of convergence in computing time and in terms of parameter updates and generalization on some datasets (e.g., in an experiment) [14]. As shown in Figure 2 (left), the LSTM unit is comprised of i , f and o , which are the input, forget and output gates, c is the memory cell content and \tilde{c} is the new memory cell content. The slightly lighter GRU unit in Figure 2 (right) has reset and update gates (r and z), and h and \tilde{h} are the activation and the candidate activation.

C. Hyperparameter Tuning

Hyperparameter tuning, or Hyperparameter Optimization (HPO), involves finding hyperparameters that produce

an optimal ML model that minimizes the predefined loss function. Hyperparameters can be either architectural hyperparameters or parameters of the optimization process. Examples of the former are the type of ML model, the number and type of layers, the number of neurons per layer and the activation function to be used. Examples of optimization parameters are the optimizer type (e.g., SGD, Adam), the learning rate, and momentum. We employed two different tools for hyperparameter optimization; the Adaptive Experimentation Platform (Ax)⁷ and Ray Tune⁸.

D. HPC Systems

The Earth sciences have seen a dramatic growth in the volume and dimensionality of data, to the extent that traditional data mining tools, such as R and MATriXLABoratory (MATLAB), cannot handle or even load the dataset into memory. Furthermore, they are essentially unable to make use of parallel computing due to their often sequential algorithm implementations, resulting in lengthy processing or ANN training times (there are certain MATLAB Parallel Computing Toolbox⁹ algorithm implementations that perform parallel computing, and some scikit-learn estimators and utilities parallelize costly operations using multiple CPU cores). These limitations can be overcome with the use of HPCs with parallel and scalable techniques [19].

The training of DNNs [5] is a significant computing task that places high demands on computing resources and memory bandwidth. The use of GPUs can greatly speed up such training, especially when working with large quantities of data. As GPUs have a large number of parallel arithmetic units called many-core design, they can handle thousands of threads concurrently [20]. DNN training is also possible by leveraging multiple GPUs simultaneously in a distributed manner, which has been shown to offer super-linear speedups [6]. Hyperparameter tuning, like the training of DNNs, is a computationally expensive task that can also benefit greatly from the power of HPC.

E. UAIF RAISE and LAMEC

Compared to classic simulation sciences (e.g., using numerical libraries based on known physical laws, etc.), DL model training can benefit greatly from the use of HPC systems for a wide range of application fields. In order to offer smooth solutions for a wide range of intricate DL applications, The European Center of Excellence in Exascale Computing “Research on AI- and Simulation-Based Engineering at Exascale” (CoE RAISE) has developed the UAIF. A number of National Competence Centres (NCCs), including Iceland and Germany, already use some aspects of UAIF for various purposes [7]. A component of the reference architecture elements of UAIF is the LAMEC API batch script repository and job script generator. Based on the selection of the end user, the LAMEC job script generator automatically selects the

⁵<https://documentation.sas.com/doc/en/emref/15.2/n061bzurmej4j3n1jnjb8bjm1a2.htm>

⁶<https://www.datascience-pm.com/agile-data-science/>

⁷<https://ax.dev>

⁸<https://www.ray.io/ray-tune>

⁹<https://www.mathworks.com/products/parallel-computing.html>

correct module configuration for a given ML framework and sets sensible default values for the Simple Linux Utility for Resource Management (SLURM)¹⁰ script based on the software and HPC system to be used. Its usage allowed for a quick setup of necessary scripts for both the training of DNN models and HPO.

IV. SOIL MOISTURE CONTENT ANALYSIS

This section is structured according to the steps of the CRISP-DM open standard process model (cf. Section 2).

A. Business Understanding

Relying solely on traditional soil moisture products has many drawbacks, as they offer medium to coarse spatial resolution, whereas this approach, where satellite data is used as features for model training, has the potential to offer continuous measurements of the surface SM at high spatial and temporal resolution at a global scale. The goal of this study was to experiment with ANN and DNN models and compare their results to those of the baseline model, an RF model using the *better_default* hyperparameter template. The best possible model entails the optimal number of hidden layers, the optimal number of neurons per layer, and the hyperparameters that produce the lowest Root Mean Square Error (RMSE) value.

B. Data Understanding

The data used in this project was collected from many sources, dynamic and static data. GEE was used to download the relevant features from RS collections, and static data from other sources was incorporated into the training dataset. The spatial location of RS data was mapped to the location of the soil moisture data. Static data can be in the form of single images, providing elevation information and ground reference data from ISMN. The RS data used is from the Sentinel-1, Sentinel-2, and Landsat 8 satellites. From Sentinel-1, bands VV and VH are used, bands B3, B4, B8, B11, and B12 from Sentinel-2, and bands B10 and B11 from the Landsat 8 satellite. The static data is from a variety of sources. Global precipitation was sourced from the Global Precipitation Measurement (GPM)¹¹ mission, a joint mission between National Aeronautics and Space Administration (NASA) and Japan Aerospace Exploration Agency (JAXA)¹², temperature is from NASA Earth Observations (NEO), day of the year is a calculated feature, ground type of sand, clay or silt is from SoilGrids, soil texture of sand provided by the International Soil Reference and Information Centre (ISRIC). Finally, elevation, aspect, and slope are sourced from Deutsches Zentrum für Luft- und Raumfahrt (DLR)'s¹³ TerraSAR-X add-on for Digital Elevation Measurement (TanDEM-X) mission.

Ground reference data is from the Terrestrial Environmental Observatories (TERENO) network¹⁴, and from ISMN¹⁵, an international cooperation to establish and maintain a global in situ soil moisture database [21]. The geographical location of each station is shown in Figure 3. The data used in this study was from 317 stations and contained 533,794 total samples with 27 features per row and the SM as ground reference.

C. Data Preparation

The data was split into three datasets; training, test and validation sets. As the stations are from various global locations, the range of each feature can be slightly different. When splitting the data, the split was performed on a per station basis, ensuring that all data from a relevant station was within the same dataset. All stations with extreme values were selected for the training set. The final split between training, validation, and test sets was approximately 80%, 10%, and 10% respectively.

Given the nature of the data and the differing update frequencies of the various sources, there was a significant amount of missing data. While precipitation data is available hourly, satellite data is updated much less frequently, depending on the number of days between each satellite passing each geographic area. The Pandas dataframe's interpolate method was used to fill the NA/NaN values in both directions, but with a limit equal to half of the locations' revisit time in each direction. We also noted different feature ranges due to the data being from different stations, often in geographically diverse locations. We therefore limited the interpolation to each station individually.

As the dataset contained outliers which could potentially be problematic, an outlier detection approach was implemented to train the model both with and without outliers and compare the results. The optimal z-score threshold, the number of standard deviations above or below the mean, was calculated within a range of 3 and 5. Data scaling was performed using Scikit-learn's MinMaxScaler.

When preparing the data for RNN modeling, a different approach was used than before. Instead of treating data from all stations in the training, validation, and test sets equally, each station was now considered separately, and a three-dimensional matrix of time window data was constructed for each dataset where the time was on the X axis, the features on the Y axis, and finally the label on the Z axis. When handling stations separately, interpolation of features where no data is available for a particular station is not possible, so stations with missing data columns were dropped from the dataset. Additionally, each dataset was split into 30 day time windows for use by the RNN model.

D. Modeling

The sequential model from TensorFlow's Keras framework was utilized for both ANN and RNN models. With

¹⁰<https://slurm.schedmd.com/documentation.html>

¹¹<https://gpm.nasa.gov/missions/GPM>

¹²<https://global.jaxa.jp>

¹³<https://www.dlr.de/>

¹⁴<https://www.tereno.net>

¹⁵<https://ismn.earth/en/>

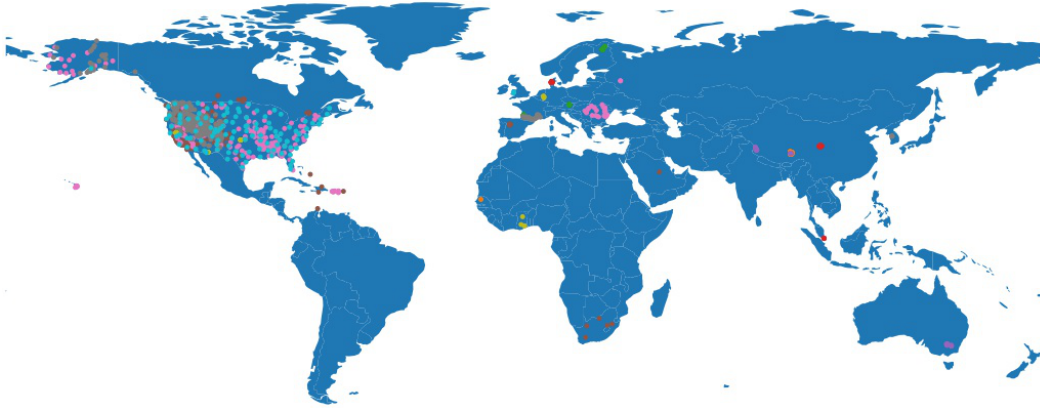


Fig. 3. An overview of the geographical locations of stations contained in the global dataset.

TABLE I
THE PARAMETER SEARCH SPACE FOR THE RAY TUNE
HYPERPARAMETER TUNING OF THE ANN MODEL.

Parameter name	Bounds
learning_rate	[0.001, 0.1]
dropout_rate	[0.1, 0.2]
num_hidden_layers	[1, 10]
num_neurons_per_layer	[64, 256]
activation	['tanh', 'sigmoid', 'relu']
optimizer	['adam', 'rms', 'sgd']

two distinct methods for hyperparameter tuning and multiple scalars, several ANN model designs were investigated. We used Mean Squared Error (MSE) as the loss function, RMSE as the metric and utilized early stopping to avoid extensive model training when performance was dropping. Models were trained both with and without outliers.

We performed hyperparameter tuning using both the Ax tool, and Ray Tune. While Ax offers many excellent features, Ray Tune was selected for further HPO as it allows for fast hyperparameter tuning at scale. By utilizing the Dynamical Exascale Entry Platform - Extreme Scale Technologies (DEEP-EST) HPC system, multiple trials could be run in parallel by utilizing multiple Central Processing Units (CPUs) and GPUs simultaneously. The parameter search space for the trials are shown in Table I.

We created two different RNN models for this study, a GRU model and an LSTM model. Due to the increased data volume when working with time window data, it was necessary to perform all training and evaluation on the DEEP-EST system. Due to the time window approach, the size of each dataset was considerably larger than what was used for other models. With a window size of 30 days, the data overlap is significant, as each new window shares 29 days of data with the previous one, resulting in a substantial increase in the number of training samples and a higher computational demand. We therefore decided to limit the number of trials for optimal hyperparameter search to 20, instead of 100 as was done with the ANN model.

V. EVALUATION AND RESULTS

This section covers the evaluation section of the CRISP-DM open standard process model.

When evaluating the results from each modeling approach, the RMSE of the global test set is used to compare each model. The lowest test set RMSE of the NN models

was obtained using Ray Tune for hyperparameter tuning and the MinMaxScaler to scale the data. The final results of all modeling techniques can be seen in Table II.

TABLE II
FINAL RESULTS OF ALL MODELS ON THE GLOBAL DATASET.

Name	Training set	Validation set	Test set
	RMSE	RMSE	RMSE
RF	0.046	0.094	0.111
NN	0.081	0.093	0.103
GRU	0.095	0.109	0.122
LSTM	0.095	0.107	0.123

The model that produced the lowest RMSE on the test set of the global dataset was the ANN model optimized with Ray Tune hyperparameter tuning with 2 hidden layers, 254 neurons per layer, the sigmoid activation function, Adam optimizer, a learning rate of 0.0156, and a dropout rate of 0.1. The results from various trials on an ANN model using different scalars and outlier handling methods can be seen in Table III. The resulting RMSE was 0.103, which is approximately a 7% improvement when compared to the baseline RF model.

Initial HPO was done using the Ax tool on Google Colab¹⁶ with a GPU kernel, where we ran a total of 25 trials for 5 epochs. The running time was over 3 hours. While Ax offers many excellent features, Ray Tune was used for further hyperparameter tuning, as it allows for fast hyperparameter tuning at scale. By utilizing the DEEP-EST system, multiple trials could be run in parallel by utilizing multiple CPUs and GPUs simultaneously. This provided a dramatic decrease in training time, where the running time for 100 trials running for 5 epochs was just over 2 hours.

The performance of the GRU and LSTM models was slightly worse than that of the best ANN model, which may be attributed to the lack of data often required by these innovative DL techniques. Due to the amount of data involved when working with a time window approach, it proved impossible to run such trials on Google Colab due to memory problems. The running time of both LSTM and GRU models was over 17 hours on DEEP-EST, utilizing a total of 16 CPUs and four GPUs. The soil moisture domain scientists benefitted from using the LAMEC job script generator for job script creation for this task.

¹⁶<https://colab.google>

TABLE III
RESULTS FROM THE VARIOUS ANN MODEL TRIALS.

Name	RMSE
ANN (MinMaxScaler)	0.119
ANN (RobustScaler)	0.107
ANN (no outliers)	0.116
ANN (Ax)	0.113
ANN (Ray Tune / MinMaxScaler)	0.103
ANN (Ray Tune / RobustScaler)	0.110
ANN (Ray Tune / no outliers)	0.122

VI. CONCLUSIONS

When comparing the results to the baseline RF model, the ANN model produced the best results, an RMSE of 0.103 which is a 7% improvement from the RF models' RMSE of 0.111. The computed p-value confirms that the improvement is statistically significant and translates to more consistent and reliable SM predictions. In agriculture, this can mean fewer irrigation mistakes, more efficient water usage, and better long-term crop health. The RNN models failed to improve on the performance of the other models, with the GRU producing an RMSE of 0.122, and the LSTM model producing an RMSE of 0.123. This may be attributed to the limited volume of temporally available data available to effectively train sequential models. We conclude that DL techniques often promise improved performance but require large amounts of data to work well.

The key to further improvement in accuracy may lie in obtaining more data features and data from even more stations. More features may possibly limit the applicability of the trained model for areas with fewer features. Therefore, having more features may not necessarily be better, but there may be new features that are more relevant and can replace other features that are less relevant.

Integrating multiple data sources using data fusion techniques might also prove invaluable. 1D Convolutional models and the use of k-fold validation was briefly considered, but initial results were similar to those of the RNN models. Studies have shown that the RF model is one of the best performing when it comes to estimating soil moisture [22], so instead of focusing on the model, we believe that the next step should be to gather more data where possible.

By eliminating the manual effort of job script creation and tuning, often consuming several hours per week, LAMEC significantly streamlines the data science workflow, allowing researchers to focus on model development and experimentation.

REFERENCES

- [1] M. Holzman, R. Rivas, and M. Piccolo, "Estimating soil moisture and the relationship with crop yield using surface temperature and vegetation index," *International Journal of Applied Earth Observation and Geoinformation*, vol. 28, pp. 181–192, 2014. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0303243413001748>
- [2] E. Babaeian, C. Montzka *et al.*, "Ground, proximal, and satellite remote sensing of soil moisture," *Reviews of Geophysics*, vol. 57, no. 2, pp. 530–616, 2019. [Online]. Available: <https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2018RG000618>
- [3] F. Greifeneder, C. Notarnicola, and W. Wagner, "A machine learning-based approach for surface soil moisture estimations with Google Earth Engine," *Remote Sensing*, vol. 13, no. 11, 2021. [Online]. Available: <https://www.mdpi.com/2072-4292/13/11/2099>
- [4] L. Breiman, *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001. [Online]. Available: <https://doi.org/10.1023/a:1010933404324>
- [5] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015. [Online]. Available: <https://doi.org/10.1038/nature14539>
- [6] G. Cong, G. Domeniconi *et al.*, "Fast neural network training on a cluster of GPUs for action recognition with high accuracy," *Journal of Parallel and Distributed Computing*, vol. 134, pp. 153–165, 2019.
- [7] M. Riedel, C. Barakat, S. Fritsch, M. Aach, J. Busch, A. Lintermann, A. Schuppert, S. Brynjólfsson, H. Neukirchen, and M. Book, "Enabling hyperparameter-tuning of AI models for healthcare using the coe raise unique ai framework for hpc," in *2023 46th MIPRO ICT and Electronics Convention (MIPRO)*, 2023, pp. 435–440.
- [8] B. Bauer-Marschallinger, V. Freeman *et al.*, "Toward global soil moisture monitoring with Sentinel-1: Harnessing assets and overcoming obstacles," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 1, pp. 520–539, 2019.
- [9] T. G. J. Rudner, M. Rußwurm *et al.*, "Multi³net: Segmenting flooded buildings via fusion of multiresolution, multisensor, and multitemporal satellite imagery," 2018. [Online]. Available: <https://arxiv.org/abs/1812.01756>
- [10] D. Fan, C. Montzka *et al.*, "A Sentinel-1 SAR-based global 1-km resolution soil moisture data product: Algorithm and preliminary assessment," *Remote Sensing of Environment*, vol. 318, p. 114579, 2025. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0034425724006059>
- [11] L. H. Hughes, D. Marcos, S. Lobry, D. Tuia, and M. Schmitt, "A deep learning framework for matching of SAR and optical imagery," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 169, pp. 166–179, 2020. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0924271620302598>
- [12] O. Maimon and L. Rokach, *Chapter 1 - Introduction to Knowledge Discovery in Databases*, 01 2005, pp. 1–17.
- [13] M. Riedel, A. S. Memon *et al.*, "High productivity data processing analytics methods with applications," in *2014 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 2014, pp. 289–294.
- [14] J. Chung, C. Gulcehre *et al.*, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014. [Online]. Available: <https://arxiv.org/abs/1412.3555>
- [15] R. C. Staudemeyer and E. R. Morris, "Understanding LSTM—a tutorial into long short-term memory recurrent neural networks," *arXiv preprint arXiv:1909.09586*, 2019.
- [16] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–80, 12 1997.
- [17] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *arXiv preprint arXiv:1409.1259*, 2014.
- [18] F. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: continual prediction with LSTM," in *1999 Ninth International Conference on Artificial Neural Networks ICANN 99. (Conf. Publ. No. 470)*, vol. 2, 1999, pp. 850–855 vol.2.
- [19] M. Götz, G. Cavallaro *et al.*, "On scalable data mining techniques for earth science," *Procedia Computer Science*, vol. 51, pp. 2188–2197, 2015, international Conference On Computational Science, ICCS 2015. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1877050915013022>
- [20] S. Dong, P. Zhao *et al.*, "Exploring GPU acceleration of deep neural networks using block circulant matrices," *Parallel Computing*, vol. 100, p. 102701, 2020. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0167819120300909>
- [21] H. Bogen, C. Montzka *et al.*, "The TERENO-Rur hydrological observatory: A multiscale multi-compartment research platform for the advancement of hydrological science," *Vadose Zone Journal*, vol. 17, no. 1, p. 180055, 2018. [Online]. Available: <https://access.onlinelibrary.wiley.com/doi/abs/10.2136/vzj2018.03.0055>
- [22] S. K. Chaudhary, P. K. Srivastava *et al.*, "Machine learning algorithms for soil moisture estimation using Sentinel-1: Model development and implementation," *Advances in Space Research*, vol. 69, no. 4, pp. 1799–1812, 2022, advances in Spaceborne SAR Remote Sensing for Characterization of Natural and Manmade Features - Part 1. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0273117721006724>