Distributed training and inference of deep learning solar energy forecasting models

Javier Campoy*†, Ignacio-Iker Prado-Rujas*‡, José L. Risco-Martín§¶, Katzalin Olcoz§¶ and María S. Pérez*‡

* *Ontology Engineering Group, ETSI Informáticos, Universidad Politécnica de Madrid, 28660 Madrid, Spain*
§ *Architecture and Technology of Computing Systems, Universidad Complutense de Madrid, 28040 Madrid, Spain*
† javier.campoy@alumnos.upm.es, ‡ {iker.prado.rujas|maria.s.perez}@upm.es, ¶ {jlrisco|katzalin}@ucm.es

***Abstract*—Different accurate predictive models have been developed to forecast the amount of solar energy produced in a given area. These models are usually run in a centralized manner, considering irradiance inputs taken from a set of sensors that are deployed in that area. CAIDE is a framework that supports the deployment and analysis of solar plants following Model Based System Engineering (MBSE) and Internet of Things (IoT) methodologies. However, the current solution performs the training and inference phases of the solar energy forecasting models in a central way, not taking advantage of the distributed environment modeled by means of CAIDE. This work presents an extension of CAIDE that allows us to distribute the training and inference phases, obtaining performance improvements, and achieving a greater adaptation to the inherently distributed topology of the deployment of the sensors.**

***Index Terms*—Complex Systems, Discrete Event System Specification, Deep Learning, Solar Irradiance, Parallel and Distributed Simulation.**

I. Introduction and related work

The increasing number of climatic hazards and extreme weather events highlight the need to switch to renewable energy sources and achieve net zero emissions by 2050 [1]. Despite their associated variability, some renewable energies are growing very fast, and such is the case for solar energy [2]. However, effective solar plant management requires deploying solar irradiance sensors that monitor the amount of energy that can be produced and the support of forecasting models to estimate its values over time. In addition, these models and monitoring systems are usually run in a centralized way, which limits their usability and neglects the distributed topology of the overall system.

This article builds on our previous work [3], where the CAIDE framework was introduced. As in that case, we consider a complex scenario in which multiple independent solar irradiance sensor farms are deployed to analyze the possibilities of Photovoltaic (PV) solar production of a given region. Following an architecture based on the Internet of Things (IoT), the edge layer consists of a set of sensors that continuously send data to the fog layer, which also communicates with a cloud layer periodically (see Figure 1). The framework provides several services for domain experts and high-level authorities, such as data analysis, inference, or outlier detection. However, it did not initially consider the possibility of retraining the Deep Learning (DL) model once the forecast performance deteriorates. The main contribution of this work is the enlargement of CAIDE's capabilities by enabling it to perform periodic model training at the cloud layer. Three different retraining techniques are studied, and the benefits and downsides of each of them are presented.

This paper is organized as follows. Section II reviews the CAIDE framework and the DL-based solar forecasting model. Section III presents the methodology followed to study different retraining techniques. Section IV illustrates the experiments performed to test our hypothesis. Finally, Section V draws some conclusions and introduces future research lines.

This work has been supported by the Autonomous Region of Madrid through the program CABAHLA-CM (GA No. P2018/TCS-4423).

II. Background

As mentioned in Section I, this work extends the one presented by Almendras et al. [3]. There, CAIDE is introduced as a framework based on Discrete Event System Specification (DEVS) with several capabilities for the management of a set of solar plants. Among those characteristics, the system includes a forecasting service that allows one to estimate the forthcoming solar energy for short-term horizons. This section summarizes the most relevant aspects of the framework and its DL-based forecasting model.

A. The CAIDE framework

The integrative framework presented in [3] must be able to run scalable simulation scenarios for the problem studied. CAIDE's simulation framework is based on parallel DEVS, and, more specifically, it is implemented using the xDEVS Modeling and Simulation (M&S) engine [4].

In terms of the architecture of the model, it is divided into the three classical IoT layers: edge, fog, and cloud. This setup can be observed in Figure 1 for the management of sensor farms. The technical details of the CAIDE framework can be studied in [3].

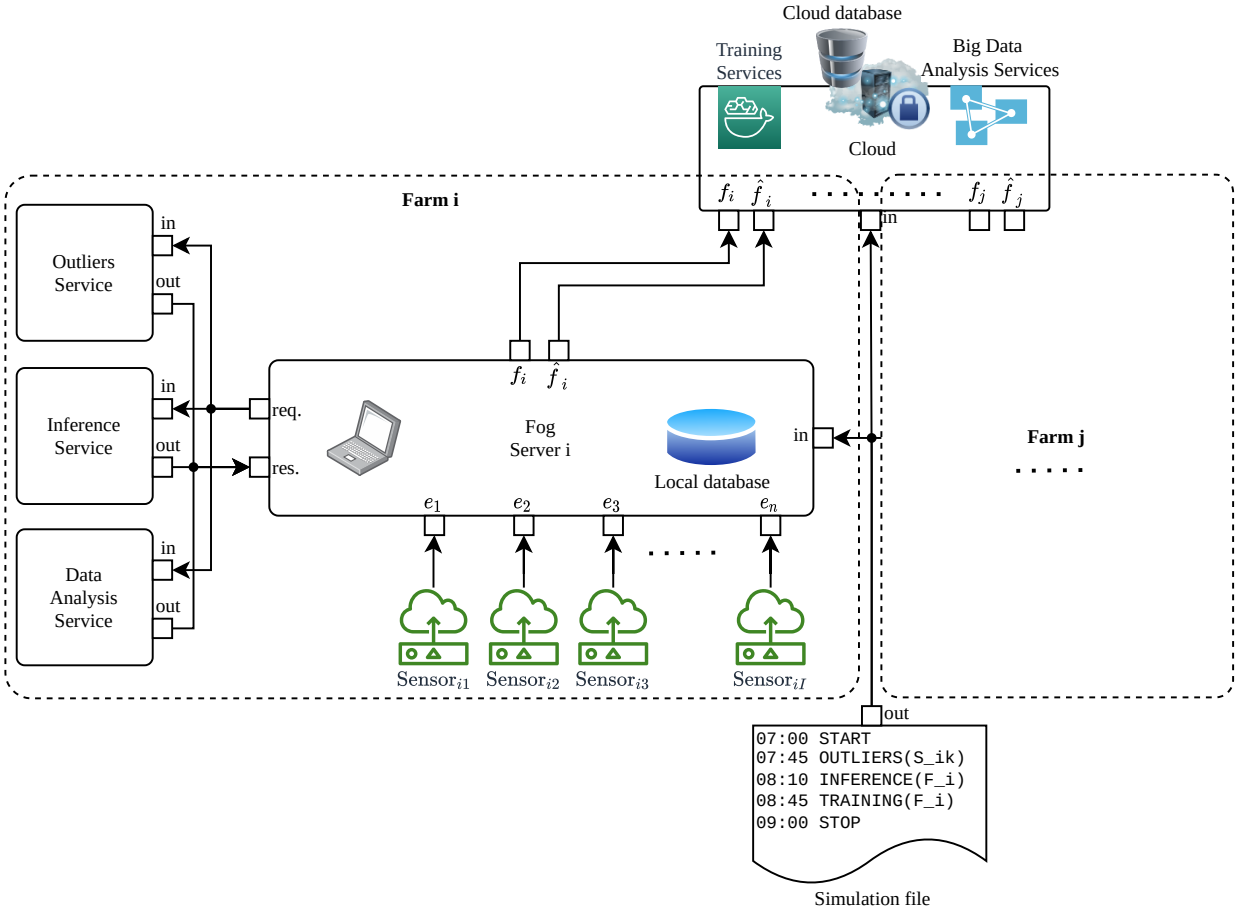

Fig. 1. CAIDE's model architecture.

B. Deep Learning-Based Method for Solar Irradiance Forecasting

As discussed in Section I, predicting how much solar energy will be absorbed in any given area is a key aspect of the management of any PV plant. The described problem devises a scenario with multiple solar farms, each with a variable distribution and number of sensors. Therefore, the forecasting models employed must exhibit flexibility and robustness beyond accuracy. Such characteristics are commonly known as *non-functional* and have been studied in previous work [5]–[7].

As in [3], we adapt the DL-based model from our previous work [6] for the experiments proposed in Section IV. This predictive model incorporates spatial and temporal features, which are inherent in the problem of solar irradiance forecasting. The data utilized to train it was obtained from [8], which comprises 20 months of Global Horizontal Irradiance (GHI) values recorded at Oahu (Hawaii) by 17 pyranometers. In Section III we propose different ways to improve upon the previously tested model, which forecast GHI for four horizons up to one hour. To do so, different methods to update the model's predictive capabilities are studied, taking advantage of the distributed nature of the CAIDE framework.

III. Methodology

In this section, we discuss different ways to implement a retraining function. To do so, we need to study both its effectiveness in improving its predictions and the required resources. We propose three different alternatives that shall be compared with a baseline model, each suited for a different situation.

The three proposed methods for retraining our model are the following:

- **A1**: *Greedy approach*. Discard the current version of the model and train a new one using historical and newly available data.
- **A2**: *Sliding approach*. Train a new model with the same amount of data as the original but replace the oldest instances with new ones in a sliding window manner.
- **A3**: *Fine-tuning approach*. Add a few additional training epochs to the original model, just with new batches of data.

The procedure to evaluate such alternatives will be as follows. From the 20 months of available data, we will use 12 to train a baseline model. We then fix the model update marks at one, two, and three months after the 12$^{\text{th}}$ month. At those

marks, we will update the model as described by approaches A1, A2, and A3. We will then use the four months following each mark to evaluate the performance of each model, using the Root-Mean-Squared Error (RMSE) metric. Thus, we can evaluate the model's performance for each situation in a realistic environment, where an existing model is updated and tested as new data becomes available.

As can be seen, each method carries a different computational load. For A1, training time increases linearly with the amount of new data. On the other hand, A2 maintains a constant cost, and A3 varies depending on the amount of new data but uses fewer resources overall. To evaluate the performance of each approach, two values will be considered: skill and training time. The former quantifies the percentage improvement over the baseline model (see Eq. 1), and the latter measures the computational cost, in terms of time.

$$S = \left(1 - \frac{\text{error}_{\text{proposed}}}{\text{error}_{\text{baseline}}}\right) \cdot 100\% \tag{1}$$

The error metric used in Eq. 1 is the RMSE, averaged over sensors and horizons for the entire testing period (four months). The forecast horizons are 1, 11, 31, and 61 minutes for every requested inference.

Once these approaches are tested, we can implement the retraining method of choice into the CAIDE framework to obtain a fully functional and dynamic forecasting model.

IV. Experiments

In this section, we discuss the experiments that were implemented and their results. First, we comment on the computing units used to train the models. Each experiment had a Graphics Processing Unit (GPU) available to perform the training. More specifically, the employed GPUs were:

- NVIDIA GeForce RTX 2070 (8192MiB).
- NVIDIA GeForce GTX TITAN X (12288MiB).
- NVIDIA GeForce RTX 2080 Ti Rev. A (11264MiB).

The computing capabilities of such a setup exceed those expected in a typical fog layer, but can be achieved in a more complex cloud layer. This is important when assessing training times as a computational load metric, since expected times should be properly rescaled to the final set-up of choice.

Training times are displayed in Table I for each approach and update mark. The training time of the baseline model is also depicted. Training times scale as expected (see Section III). The training time of A1 scales linearly with the number of new months. For A2, the training time remains practically constant (and similar to the baseline). Finally, A3 takes significantly less time than the other two approaches, also growing proportionally with the amount of new data.

Figure 2 shows the skill values computed for each approach and update mark. In the graph, the three proposed methods for the update of the model are compared with each other, using the baseline model as a reference (see Section III). The vertical axis represents the skill value mentioned above (see Eq. 1). On the other hand, the horizontal axis takes discrete values at each update mark. Each update mark indicates an increase of one month in training data compared to the previous mark. Also, the testing period is shifted one month forward on each update mark.

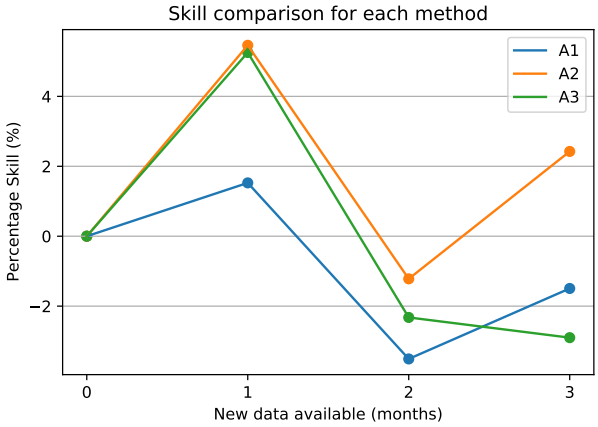

Fig. 2. Skill comparison for the different approaches

	0 months	+1 month	+2 months	+3 months
A1		1h 45'	1h 58'	2h 02'
A2		1h 39'	1h 36'	1h 41'
A3		2'	4'	6'
Baseline	1h 41'			

TABLE I
TRAINING TIME FOR EACH APPROACH AND UPDATE MARK.

V. Conclusions

With the results of the experiments described in Section IV, we can draw some conclusions regarding the different methods proposed. The first aspect to note is that A2 provides better predictive performance than the other two methods for the proposed study cases, outscoring the other approaches on every update mark and reaching a positive skill of 5.46% for the second mark. Approach A1 performs worse than A2, both in predictive capacity and training time. This should be reason enough to disregard this approach in favor of the A2 method.

When looking at training times, one might consider the A3 approach as the best, since the drop in skill might look negligible when compared to the computational cost. However, its predictive performance does not improve that of the baseline model. This might be due to a phenomenon commonly known as *catastrophic forgetting* [9]. Additional experimentation might be necessary in order to discard this approach, but due to a limitation in the amount of data available, it seems sensible to focus on the other approaches at the moment.

Therefore, this first exploration suggests the use of A2 as the best model update method for our needs. This method shall be incorporated into the CAIDE framework, where model training may be performed decentralizedly, taking advantage of the distributed nature of the architecture.

REFERENCES

[1] S. Bouckaert, A. F. Pales, C. McGlade, U. Remme, B. Wanner, L. Varro, D. D'Ambrosio, and T. Spencer, "Net zero by 2050: A roadmap for the global energy sector," https://trid.trb.org/view/1856381, Jun. 2021.

[2] "EU Solar Energy Strategy," https://energy.ec.europa.eu/topics/renewable-energy/solar-energy_en, European Commission, Tech. Rep., 2022.

[3] L. Almendras, J. Campoy, I.-I. Prado-Rujas, J. L. Risco-Martín, M. S. Pérez, and K. Olcoz, "A distributed iot-based simulation framework for solar energy management and forecasting," in *Jornadas SARTECO 2022*, Alicante, Spain, Sep. 2022, doi:10.5281/zenodo.7075818.

[4] J. L. Risco-Martín, S. Mittal, K. Henares, R. Cardenas, and P. Arroba, "xdevs: A toolkit for interoperable modeling and simulation of formal discrete event systems," *Software: Practice and Experience*, pp. 1–42, 2022.

[5] I.-I. Prado-Rujas, E. Serrano, A. García-Dopico, M. L. Córdoba, and M. S. Pérez, "Predicción espacio-temporal: más allá del error/precisión," in *Congreso Español de Informática 2021 (CEDI'21)*, Málaga, Spain, Sep. 2021, doi:10.5281/zenodo.5530048.

[6] I.-I. Prado-Rujas, A. García-Dopico, E. Serrano, and M. S. Pérez, "A flexible and robust deep learning-based system for solar irradiance forecasting," *IEEE Access*, vol. 9, pp. 12 348–12 361, 2021, doi:10.1109/ACCESS.2021.3051839.

[7] I.-I. Prado-Rujas, E. Serrano, A. García-Dopico, M. L. Córdoba, and M. S. Pérez, "Combining heterogeneous data sources for spatio-temporal mobility demand forecasting," *Information Fusion*, vol. 91, pp. 1–12, 2023, doi: 10.1016/j.inffus.2022.09.028.

[8] M. Sengupta and A. Andreas, "Oahu Solar Measurement Grid (1-Year Archive): 1-Second Solar Irradiance; Oahu, Hawaii (Data)," doi:10.5439/1052451, March 2010, type: dataset (last accessed 22/6/2022).

[9] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the national academy of sciences*, vol. 114, no. 13, pp. 3521–3526, 2017, doi: 10.1073/pnas.1611835114.