CENAERO Challenge 1 Additive Manufacturing

Report

Authors : Yann CLAES, Gaspard LAMBRECHTS, Antoine GRATIA

Supervisors : Caroline SAINVITU, Tariq BENAMARA

August 30, 2021 — September 10, 2021

Contents

1 Introduction

Additive manufacturing (AM) is the construction of 3D objects based on computer-aided design models or digital 3D models. The term "additive manufacturing" can refer to various processes in which materials are deposited, joined or solidified under computer control to create three-dimensional objects, and the materials are added together (for example, plastic, liquid or powder particles are fused together), usually layer by layer. 3D printing is a form of AM.

The product design and manufacturing are still expensive and time-consuming. Generating (experimental or simulated) data can be quite expensive as well.

In our case, a large amount of domain prior knowledge is available (physics laws, empirical verification rules, etc.). Multi-source data is a valuable resource for gaining new insights into engineering processes and decision-making. The overall goal is to enhance the mastery of industrial processes to improve the quality and/or the cost of a simulation of complex physical systems in order to make better decisions in the engineering process and to shorten the time to market products whose manufacturing supports artificial intelligence.

AI-enabled AM is a better way to master this process since one can control process parameters and the process in general, but also to certify a quality insurance and real time closed-loop feedback.

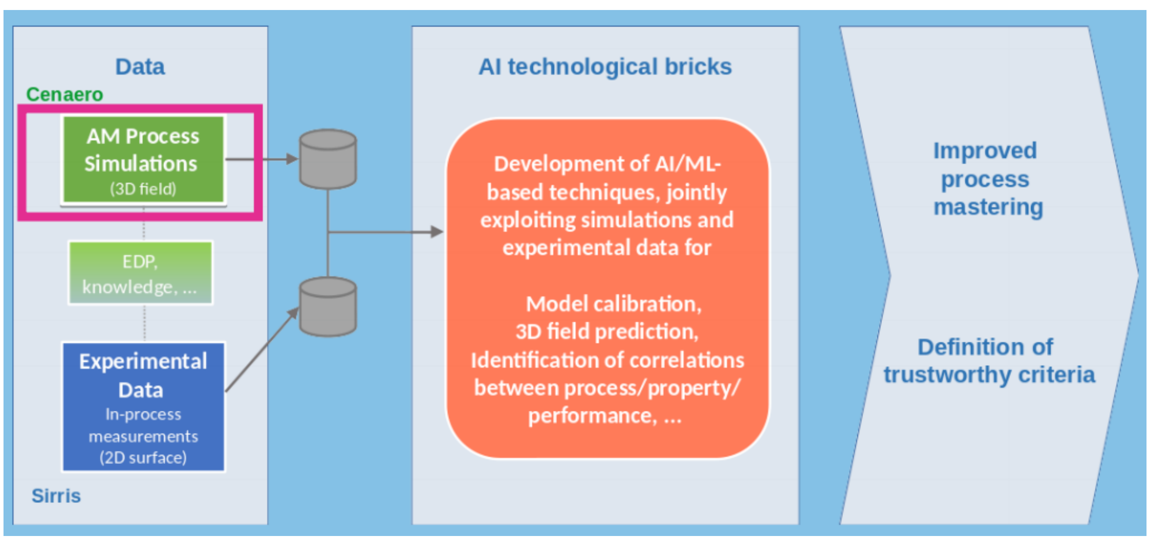

Figure 1: AI-enabled Additive Manufacturing long-term project.

Figure 1 shows the long-term scope of the project. This work will focus on the prediction of the temperature history during the AM process with ML techniques trained on simulated data (i.e. pink square). More details are given in Section 2.

In Section 2, we introduce the scope of this work. In Section 3, we discuss our methodology and team work. In Section 4, we present our results and experimental setting. In Section 5, we conclude.

2 Problem statement

In this project, we focus on a simple problem. The purpose is to predict the temperature at six points in the material, at different time steps. In this problem, we consider a two-dimensional metallic part for which the laser beam is heating the top. For now, it is hypothesised that no material is added. In other words, the domain remains constant. The piece of material and the cooling support are represented in Figure 2, along with the six points of interest. The laser beam is moving on top of the part, at a constant speed, and performs two round trips.

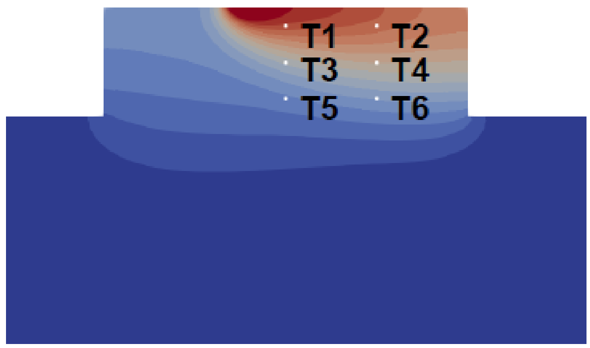

Figure 2: Example of temperature evolution.

The laser beam excitation is characterised by:

- The nominal power, P
- The break time, b.

These two parameters allow to predict the temperature at the six points, exactly, for the complete time sequence. In other words, there exists a function $f : \mathbb{R}^3 \to \mathbb{R}^6$ such that:

$$\begin{pmatrix} T_t^1 \\ T_t^2 \\ T_t^3 \\ T_t^4 \\ T_t^5 \\ T_t^6 \end{pmatrix} = f(P, b, t)$$

However, this function is complicated to evaluate, as it requires to conduct a finite-element method over all time steps until the considered time instant t, with a low enough time step.

The alternative method that we consider here is to approximate f for any (P, b, t) by $\hat{f}$ using samples from the target function f. The approximation should generalize to unseen values of P, b or t. Two types of models have been considered. The first type of models, called *stationary* models, try to regress the temperature at any time t given all necessary information (P, b, t). The second type of models, called *recurrent* models, try to regress the complete sequence of temperature, given the sequence of laser position, laser powers and time steps $(\Delta_0, P_0, x_0, \ldots, \Delta_{T-1}, P_{T-1}, x_{T-1})$.

2.1 Stationary models

Formally, we have samples $(x_i, y_i),\ i = 0, \ldots, N-1$ with $x_i = (P_i, b_i, t_i)$ and $y_i = (T_i^1, T_i^2, T_i^3, T_i^4, T_i^5, T_i^6)$ such that $x_i = f(y_i)$. Different approximation techniques are experimented and the objective, given the hypothesis space $\mathcal{F}$, is to find $\hat{f}^* \in \mathcal{F}$ such that:

$$\hat{f}^* \in \underset{\hat{f} \in \mathcal{F}}{\arg\min}\, \mathbb{E}_{(x_i, y_i)} \mathcal{L}(\hat{f}(x_i), y_i)$$

where $\mathcal{L}$ is an appropriate loss function.

2.2 Recurrent models

Formally, we have N sequences $(\mathbf{x}_i, \mathbf{y}_i)$ of variable lengths T_i. We have $x_{i,n} = (\Delta_{i,n}, P_{i,n}, x_{i,n})$ and $y_{i,n} = (T_{i,n}^1, T_{i,n}^2, T_{i,n}^3, T_{i,n}^4, T_{i,n}^5, T_{i,n}^6),\ i = 0, \ldots, N-1,\ t = 0, \ldots, T_i - 1$. And we define:

$$\Delta_{i,n} = \begin{cases} t_{i,n} - t_{i,n-1} & \text{if } n > 0 \\ t_0 & \text{if } n = 0 \end{cases}.$$

We are thus looking for a recurrent estimator that would consume the successive time steps recurrently, and would output the successive prediction successively along the discrete time sequence. If the space of recurrent estimator is $\mathcal{R}$, we are looking for the function $\hat{r}^* \in \mathcal{R}$ such that:

$$\hat{r}^* \in \underset{\hat{r} \in \mathcal{R}}{\arg\min}\, \mathbb{E}_{(\mathbf{x}_i, \mathbf{y}_i)} \frac{1}{T_i} \sum_{n=0}^{T_i - 1} \mathcal{L}(\hat{r}(x_0, \ldots, x_n), y_n)$$

3 Methodology

3.1 Team work

We worked remotely with the team. We make use of Github for sharing the code and editing it easily (lgaspard/cenaero).

We have a talk each day at 3:00 p.m. with the supervisors to discuss about our work.

3.2 Work scheduling

We first explored the data set to verify if the data was clean (i.e. missing data), where we did not spot any curious pattern. Then we tried some Machine Learning (ML) techniques like a Decision Tree, a Random Forest and an Extremely Randomised Tree. For these methods, we have decided not to constrain the depth of the generated trees, although it could lead to some overfitting to the training data. After that, we tried some Deep Learning (DL) techniques like a Multilayer Perceptron (MLP) to quantify the increase in temperature prediction quality when moving to a technique that embeds more representative power.

4 Results

This section exposes the experimental settings and the results obtained.

4.1 Experimental settings

4.1.1 Dataset

We have 121 simulations for the dataset. Those simulations have two parameters that change for each simulation: the laser power and the break time during the trajectory. The values for the laser power are between 50W and 250W with a step of 20W (50, 70, 90, etc.). The values of the break time range between 0s and 10s with steps of 1 second. The parameters for each simulation are contained in a specific text file called `MinamoParameters-Wall2D.txt`, located in the folder related to the considered simulation. The simulation is contained in an `.npz` file which contains some features, such as:

- The time step t, in seconds
- The vector of temperatures along the top line of the metallic piece, $T_t^{x,\text{top}}$, measured at all x, at every time step t
- The temperatures $T_t^1 \dots T_t^6$ at 6 six points (see Figure 2), at every time step t
- The coordinates of the temperature measurements, X and Y
- The matrix of temperatures $T_t^{x,y}$, measured at all x and y locations, at every time step t
- The laser position, x_t, at every time step t
- The laser power, P_t, at every time step t

We think that we don't have enough data to train neural network. So we can have two new datasets the ADDITIONAL dataset which is the same the INITIAL datasets (i.e. first datasets) and the seconds dataset, OUTSIDE, which contains data out of the scope (i.e. break time > 10, power > 250, power < 50).

For all methods, we have decided to split the 121 available simulations into training (70%), validation (15%) and testing set (15%).

4.1.2 Features

For the methods described in Section 3.2, we have considered two stationary sets of features, containing:

- The nominal power, P
- The break time, b.
- The time step, t

and the following ones, that can be derived from the first ones

- The laser power, P_t
- The laser position, x_t
- The delta, Δ_t

where both the laser nominal power and break time are either included or not in the set of features. Indeed, the initial idea was to consider the first three features only (i.e. x_t, P_t, t), but it was soon noticed that the information they conveyed was not sufficient to produce reliable generalizable results at critical time slices, as is highlighted in Figure 3.

4.1.3 Metrics

In order to evaluate the prediction quality of each model, it seemed relevant to consider the standard Mean-Square Error (MSE) metric, that computes the average squared discrepancy between model predictions and targets.

4.1.4 Hardware

We use a MacBook Pro (13-inch, 2016, Four Thunderbolt-3 Ports) laptop, with an Intel i5-6267U 2.90GHz CPU and 8Go ram.

4.2 Decision Tree

4.2.1 Decision Tree - 3 features

In Figure 3, we can observe the temperature predictions at the six points of interest, against the true temperatures for the corresponding sequence in the test set, for a decision tree trained on 3 features. As can be seen in Figure 3, the model performs quite good

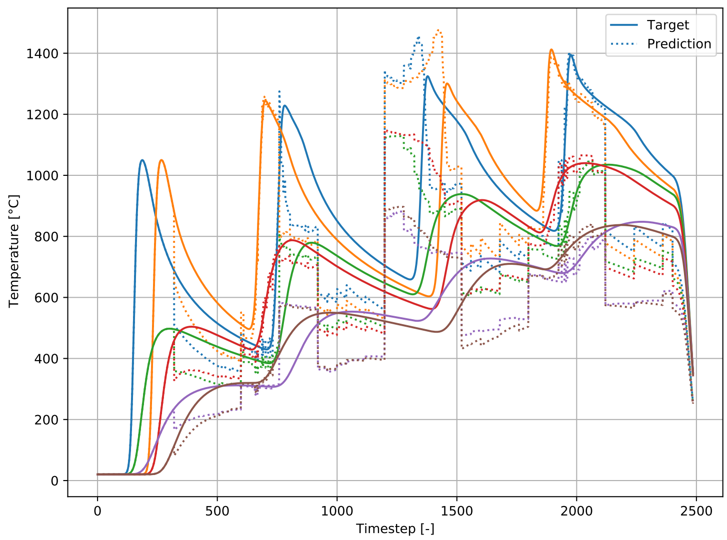

Figure 3: Temperature prediction at the six points, with a Decision Tree trained on 3 input features, on the test set.

when predicting the early stages, up to the first 250 timesteps. Afterwards, we can observe that prediction quality becomes poorer when the metallic part is supposed to cool down, due to break times. This can be explained by two reasons. Firstly, break time is not considered as an input feature. Therefore, the model cannot differentiate between cases where no break time is involved and cases where a non-zero break time is considered, for an identical laser power and at a given timestep. This will thus result in those instances being categorized into the same bin while they represent two unique situations. An another reason is that decision trees (and tree methods in general) do not embed enough representative power. Moving to a more powerful one could improve prediction quality.

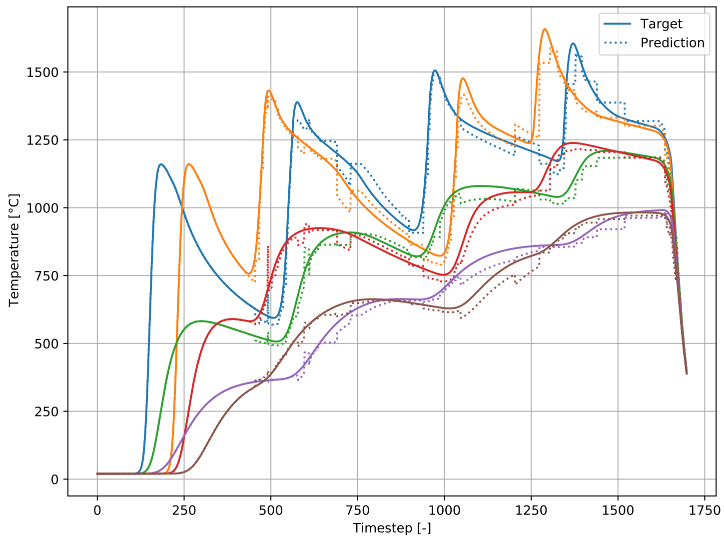

Figure 4: Temperature prediction at the six points, with a Decision Tree trained on 5 input features, on the test set.

4.2.2 Decision Tree - 5 features

In Figure 4, we can observe the predictions made by a decision tree trained on 5 input features. The introduction of the nominal power and the break time allow to model and predict more complex patterns more accurately than with four input features. However, we can notice that tree methods are not powerful enough, even with two supplementary features. Indeed, some prediction regions appear to be piecewise linear, which does not correspond to the true temperature evolution process.

4.2.3 Decision Tree - Feature Importance

As can be seen from Figure 5, the most important feature for the decision tree model is the timestep, which makes sense since it gives the model a strong information about the process phase. Together with the timestep, the nominal power is the second most important variable. This can be explained by the fact that, during break times, $P_t = 0$ and P is the only information that can help the model to differentiate between cases.

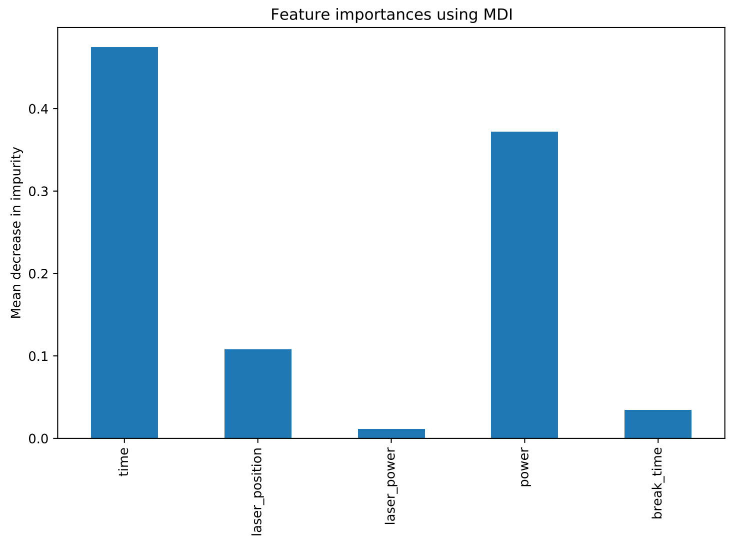

Figure 5: Feature importance for a decision tree trained on 5 features.

4.3 Random Forest

In Figure 6, we can observe the predictions made by a random forest trained on 5 input features, which appear quite similar on this test sample sequence.

4.3.1 Random Forest - Feature Importance

As was the case for the previous decision tree model, feature importance scores are similar for the random forest estimator, as can be observed in Figure 7.

4.4 Extra Trees

In Figure 8, we can observe the predictions made by an extra tree trained on 5 input features. As can be observed from Table 1, this model seems to perform the best on the test set, although this is not quite visible on this test sample. Still, the lowest MSE that tree models achieve is roughly 6 000, which yields an average Root-MSE (RMSE) of $32°C$ per point.

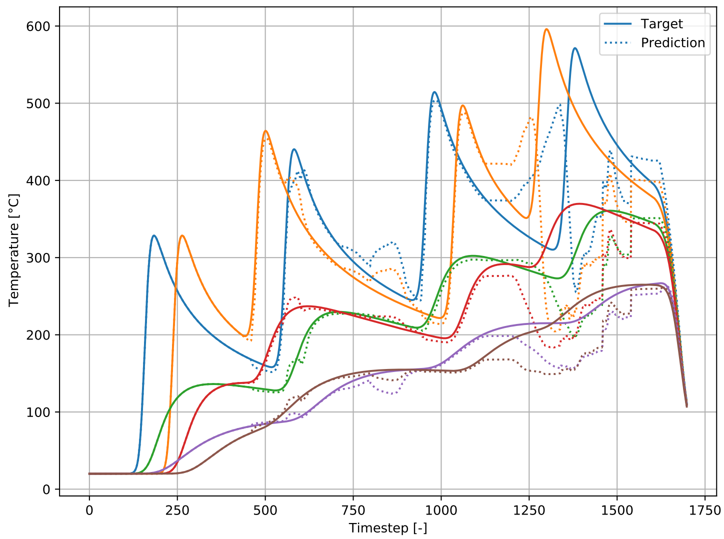

Figure 6: Temperature prediction at the six points, with a Random Forest trained on 5 input features, on the test set.

	3 features	5 features
Decision Tree	207 261	17 892
Random Forest	193 305	10 873
Extra Trees	195 961	5 610

Table 1: Mean-Square Error metrics obtained on the test set, for both situations (3 features and 5 features).

4.4.1 Extra Trees - Feature Importance

Identically to the previous tree models, feature importance scores are roughly the same for the Extra Tree estimator, as can be seen in Figure 9.

4.5 Multilayer Perceptron

Multilayer perceptrons (MLPs) have been considered. They have been trained on the 5 features (t, P_t, x_t, P, b). It can be noted that we use *early stopping*, which means that the training is stopped when no improvement was observed on the validation set losses for C epochs. We can also note that the data was normalized at the input and the output. Table 2 shows the hyperparameters chosen for this training the MLP.

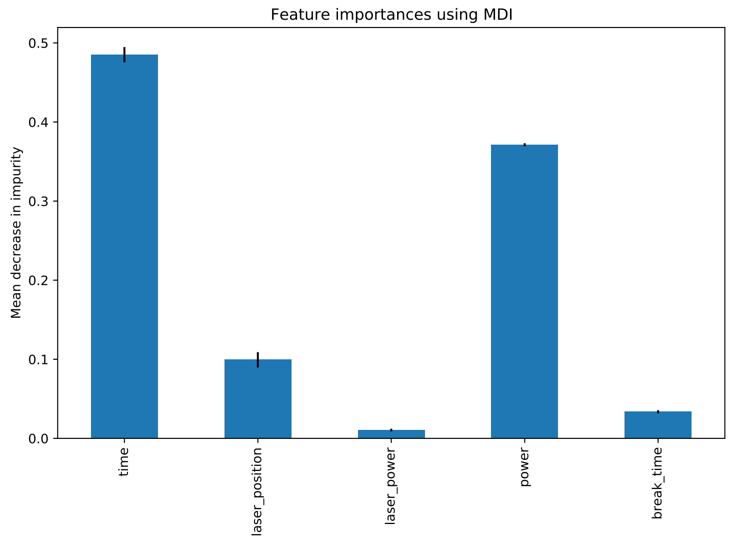

Figure 7: Feature importance for a random forest trained on 5 features.

Name	Symbol	Value
Number of epoch without improvement for early stopping	C	8
Learning rate	α	1e-4
Hidden size	H	256
Number of layers	L	2

Table 2: Multilayer perceptron hyperparameters

The lowest loss on the validation set was 10.1642 with a corresponding training loss of 9.4413. These weights were evaluated on a separate test set, leading to MSE Loss of 9.8932. A sample sequence from the test set can be seen on Figure 10.

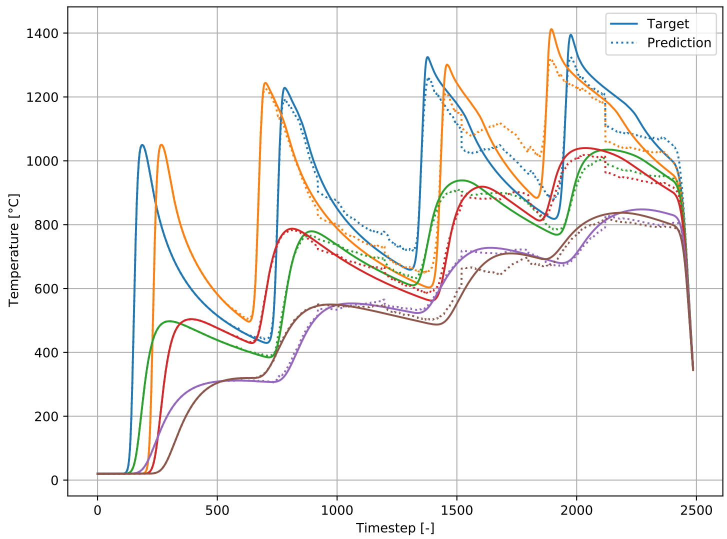

Figure 8: Temperature prediction at the six points, with an Extra Tree trained on 5 input features, on the test set.

4.6 Recurrent Neural Network

Recurrent neural networks (RNNs) have also been considered. It can be noted that we use *early stopping*, which means that the training is stopped when no improvement was observed on the validation set losses for C epochs. We can also note that the data was normalized at the input and the output. Table 3 shows the hyperparameters chosen for this training of the RNN. After the output of the RNN, an MLP is added in order to post-process the hidden states computed at each time step of the sequence.

Name	Symbol	Value
Number of epochs without improvement for early stopping	C	8
Learning rate	α	1e-3
Hidden size (RNN)	R	256
Hidden size (MLP)	H	1024
Batch size	B	16
Number of layers	L	2

Table 3: Recurrent neural network hyperparameters

The first neural network that is considered takes the 3 following features as input: (Δ_t, P_t, x_t).

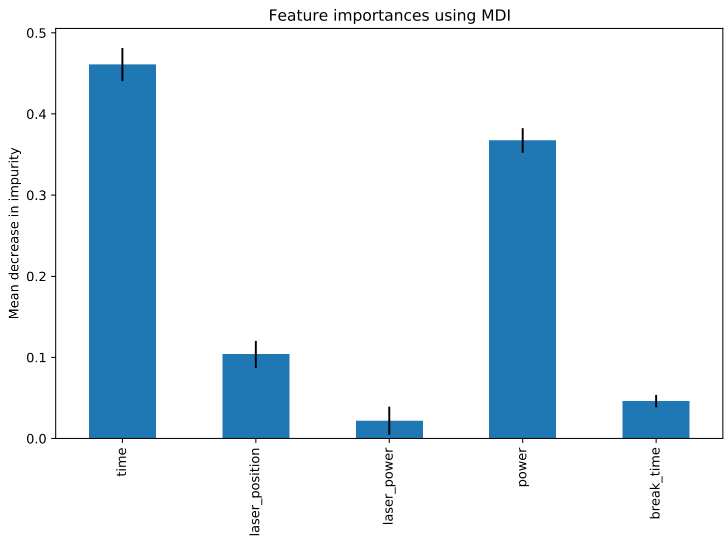

Figure 9: Feature importance for an extra trees trained on 5 features.

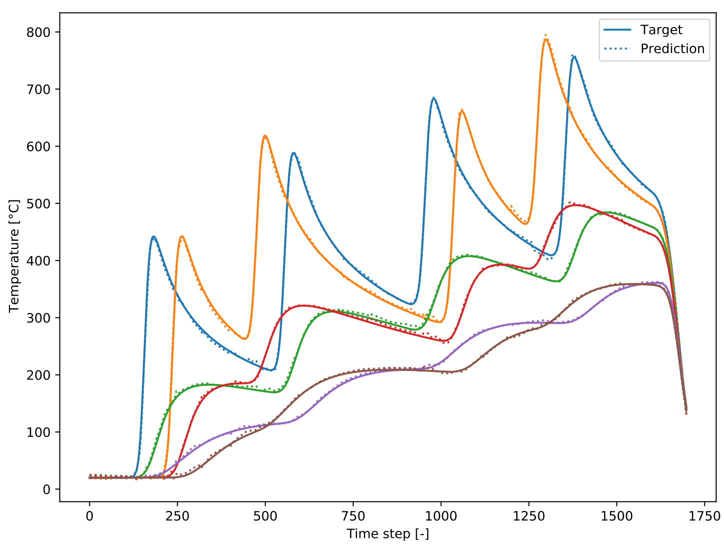

Figure 10: Sample sequence from the test set for the MLP.

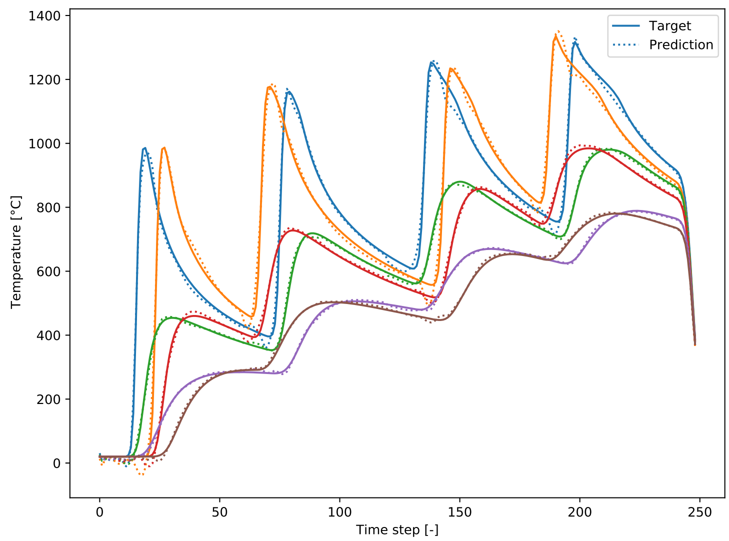

Figure 11: Sample sequence from the test set for the RNN trained of 3 features.

It was evaluated to 222.5278 on the test set. In view of the results (see Figure 11), it was hypothesised that adding b and P in the inputs could help the RNN to remember the important information in order to predict the temperature during the break times. Indeed, training a second RNN with the 5 features lead to better prediction (as can bee seen in Figure 12) and a MSE Loss of 118.9624 on the test set, which is a considerable improvement.

Afterwards, we hypothesised that the physical state and the memory needed for predicting each of the six points, was very different, especially at different depths y. We thus trained an RNN with the depth y as additional feature. In this case, the RNN was only tasked at outputting the temperature at the two points at this depth. It lead to a MSE loss of 87.0881 on the test set. Finally, we also experimented with the position of the point (x, y), with the RNN tasked at outputting a single temperature at a time in this case. It lead to a MSE loss of 17.7400 on the test set.

Finally, we also trained 6 different RNNs, with the 5 features as input, in order to predict the temperature at one of the six point for each RNN. The average MSE Loss on the test set for these six RNNs was 139.9507, which is not that much better than the RNN that predicted for thee six points directly. It lead to the conclusions that having more sequences was interesting.

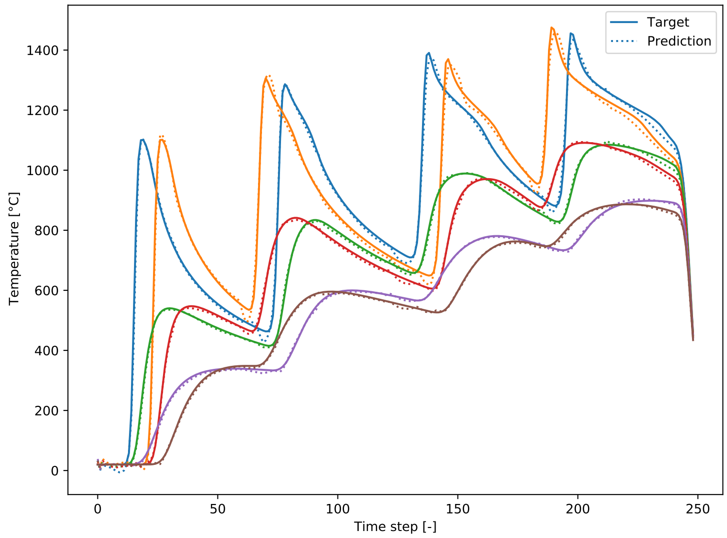

Figure 12: Sample sequence from the test set for the RNN trained of 5 features.

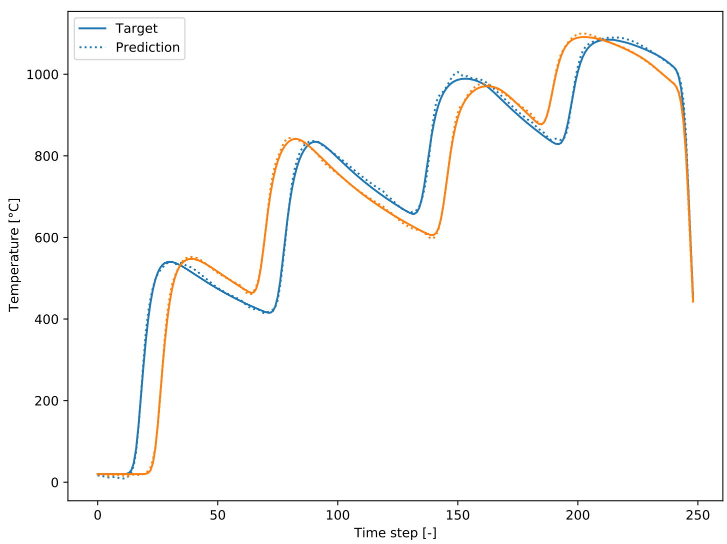

Figure 13: Sample sequence from the test set for the RNN trained of 5 features with y as input and two temperatures as output.

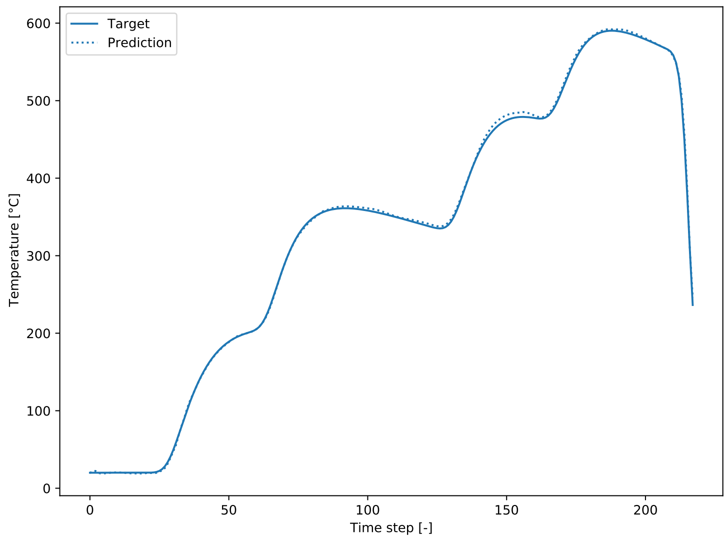

Figure 14: Sample sequence from the test set for the RNN trained of 5 features with y as input and one temperature as output.

RNN	MSE (test set)
3 features	222.5278
5 features	118.9624
5 features and y	87.0881
5 features and (x, y)	17.7400
Average of the six RNNs	139.9507

Table 4: MSE Losses for the different RNNs

5 Conclusion and Future Work

5.1 Conclusion

The aim of our work is to predict the temperature at six points. We use diverse techniques of Machine Learning/ Deep Learning (e.g. Random Forest, RNN, ...). we don't have a clear opinion between MLP (Section 4.5) and RNN (Section 4.6) wihch is our best models. We think that the RNN got must be more general and can extrapolate to new data.

In the following, we discuss different perspectives and some ideas that can help to improve the work.

5.2 Future Work

Lister en future work ce qu'on pourrait faire comme autres paramètres/tailles de séquence, incertitude/variance dans training

List of Figures