CENAERO - Challenge 1: Additive Manufacturing

Report

Authors :
Yann CLAES,
Gaspard LAMBRECHTS,
Antoine GRATIA

Supervisors :
Caroline SAINVITU,
Tariq BENAMARA

August 30, 2021 — September 10, 2021

1 Problem and data description

1.1 Problem statement

The purpose of the problem 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 1, 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. Two parameters define an execution: the laser power P (in Watts) and the break time b (in seconds), i.e. the time during which the laser stops after a forward (resp. backward) pass, at the right (resp. left) of the piece before resuming its path. Thus, given P, b, we wish to predict the temperatures, at any time step k and any point i, i.e. we want to predict $T_k^i,\ i = 1, \ldots, 6,\ k = 0, \ldots, K-1$.

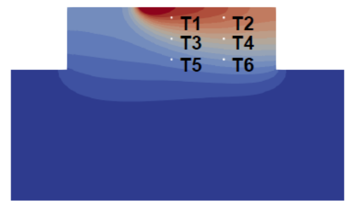


Figure 1: Example of temperature evolution, at a given time step.

1.2 Data description

We have access to three data sets, that we named `grid`, `inside` and `outside`. Each data set consists of a certain number of simulations, respectively 121, 121 and 60. `grid` contains simulations conducted with a 2D-grid of 11 equidistant powers between 50W and 250W, and 11 equidistant break times between 0s and 10s. `inside` consists of simulations for (P, b) sampled (CVT sampling algorithm) within the rectangle $\{(P, b) \in \mathbb{R}^+ \mid b < 10, 50 < P < 250\}$. Finally, `outside` consists in 3 subsets of 20 simulations for (P, b) sampled (CVT sampling algorithm) within the following rectangles: $\{(P, b) \in \mathbb{R}^+ \mid 10 < b < 15, 50 < P < 250\}$, $\{(P, b) \in \mathbb{R}^+ \mid b < 10, 25 < P < 50\}$, and $\{(P, b) \in \mathbb{R}^+ \mid b < 10, 250 < P < 300\}$.

From the simulation i, we can extract K_i time steps for which the following four variables are defined: the time t_k at time step k, the time step Δ_k between time t_{k-1} and time t_k, the laser power P_k at time step k, the position of the laser x_k at time step k. We also have the nominal power and break time (P, b) that are used as input, despite being constant on a given sequence. And we also have (x, y) that are constant on a given sequence, and identify the point at which we want to predict the temperature. Finally, we have the i^{th} temperature T_k^i at time step k that are the output variables.

2 Main results

2.1 Methods

We first tested some tree-based methods but quickly noticed that they couldn't be a good fit since they are unable to extrapolate to more extreme values. Thus, we used two methods of Deep Learning, namely a Multilayer Perceptron (MLP) and a Recurrent Neural Network (RNN). The MLP takes $[t_k, P_k, x_k, P, b, x^i, y^i]$ and predicts the output $[\hat{T}_k^i]$. The RNN takes the sequence $[x_0, \ldots, x_k]$ with $x_l = [\Delta_l, P_l, x_l, P, b, x^i, y^i]$ and predicts the output sequence $[\hat{y}_0, \ldots, \hat{y}_k]$ with $\hat{y}_l = [\hat{T}_l^i]$. Both the MLP and the RNN are trained using the MSE Loss on mini batches of timesteps/sequences, with the Adam optimizer. The training hyperparameters are reported in the Appendix 2 in Table 6a for the MLP and in Table 6b for the RNN.

2.2 Results

2.2.1 Train and test on `grid`

Table 1 compares the results of the MLP and of the RNN on the `grid` dataset. This dataset is split in three parts: 70% for the training set, 15% for the validation set and 15% for the test set. We see that the MLP is better than the RNN, with a test MSE loss of roughly 48, which represents a standardized root mean square error (SRMSE) of approximately 2%.

Model	Train	Validation	Test	Training Time (s)	Epochs
MLP	39.4655	40.3051	48.2344	153.67	31
RNN	83.4463	83.4463	107.0806	13660.38	31

Table 1: Comparison of the MLP vs RNN on the `grid` dataset.

2.2.2 Train on `grid`, test on `inside`

Table 2 compares the results of the MLP and of the RNN trained on `grid` and tested on `inside`. The `grid` dataset is split into 80% for the training set and 20% for the validation set. 100% of the `inside` dataset is used for the test set. We see that the loss is better for both (see Table 1) and the MLP remains better than the RNN. This can be explained by the fact that the models have been trained on more data than in the first scenario. We can conclude that bpth models learn a good representation of the process for the input space defined by `grid`.

Model	Train	Validation	Test	Training Time (s)	Epochs
MLP	27.3852	31.9765	25.3137	650.01	35
RNN	62.8600	75.0986	74.0471	-	36

Table 2: Comparison of the MLP vs RNN trained on `grid` and tested on `inside`.

2.2.3 Train on `grid`, test on `outside`

Table 3 compares the results of the MLP and the RNN trained on `grid` and tested on `outside`. The `grid` dataset is split into 80% for the training set, 20% for the validation set. 100% of `outside` is used for the test set. We see that the MLP is still better than the RNN but performance is worse on the test set (i.e. unseen data) than for the first two scenarios, with a SRMSE of 6% for the RNN. Thus, generalisation to unseen inputs is not as good as generalisation to inputs drawn from the same space.

Model	Train	Validation	Test	Training Time (s)	Epochs
MLP	46.8008	53.0518	152.4605	707.35	32
RNN	158.4393	181.5045	445.5791	4270.24	20

Table 3: Comparison of the MLP vs RNN trained on `grid` and tested on `outside`.

2.2.4 Train and test on `grid` and `inside`

Table 4 compares the results of the MLP and of the RNN trained and tested on a fusion of the `grid` and the `inside` datasets. This new dataset is split into 70% for the training set, 15% for the validation set and 15% for the test set. We see that RNN is now better than MLP. The former achieves better performance than for the first two scenarios, while the latter achieves similar performance. This shows that, for the RNN, training on a larger set of data helps to increase prediction quality.

Model	Train	Validation	Test	Training Time (s)	Epochs
MLP	27.1497	30.1303	31.5513	511.08	35
RNN	13.8814	16.3408	16.5343	18375.29	50

Table 4: Comparison of the MLP vs RNN trained and tested on `grid` and `inside`.

2.2.5 Train on `grid` and `inside`, test on `outside`

Table 5 compares the results of the MLP and of the RNN trained on a fusion of the `grid` and of the `inside` datasets, and tested on `outside`. This new dataset is split into 80% for the training set and 20% for the validation set. 100% of `outside` is used for the test set. We see that we have a improvement for both models compared to Table 3, and that the MLP stays better than the RNN. This performance increase can be attributed to the increase of training data size.

Model	Train	Validation	Test	Training Time (s)	Epochs
MLP	24.8166	26.2279	133.6510	573.67	23
RNN	68.3561	74.9535	265.1466	-	21

Table 5: Comparison of the MLP vs RNN trained on `grid` and `inside` and tested on `outside`.

3 Perspective and future work

3.1 Conclusion

In our work, we want to predict the temperature at six points in a metallic piece. To succeed in this task, we try diverse tree-based techniques (e.g. Decision Tree, Random Forest, Extra Tree). These techniques did not give convincing results so we decided to use Deep Learning techniques (e.g. MLP and RNN). These models show better results than the tree-based models. We manually optimise these models to get the best results and we compare them to find which one is the best. We also noticed that the RNN model was not able to generalise better than the MLP model, despite we initially hypothesised that it should be more robust thanks to its hidden state (memory) component.

3.2 Limitations

We work on a simplified 2D model of additive manufacturing, as it would be more difficult on a 3D model. In the state of art, they use mostly 3D models of additive manufacturing [4, 1, 2, 3]. We are also limited by the amount of data, because it is simulated and it can take some time to compute additional data. We also have limited our models to predicting one time step every 10 time steps, for training time constraints, which should however not affect the learning potential too much. Finally, we did not have the time to evaluate the variance in performance, by training several models on different dataset splits.

3.3 Future work

As a first step, it would be interesting to evaluate the variance in performance for the different models, when trained on different dataset splits. Then, it would be interesting to use additional data that are available from the simulations. Indeed, the matrix of temperatures in the entire piece is available, as well as the temperature on the top of the part. On the long term, the latter could be measured in real-time, and fed as input to our model as a real-time feedback during prediction.

1 References

The codes developed in this project can be found in the following GitHub repository: lgaspard/cenaero.
In addition, a more extensive report was written and is available at lgaspard/cenaero/main/report.pdf.

2 Hyperparameters

It can be noted that the hyperparameters have been chosen such that increasing the neural networks complexities does not provide a significant advantage in terms of performance, both for the RNN and the MLP. The significantly higher complexity of the RNN is reflected on its training times.

Name	**Symbol**	**Value**
Number of hidden layers	L	2
Number of hidden units	H	256
Activation functions	-	ReLU
Learning rate	α	0.001
Batch size	B	32
Early stopping epochs	C	8

(a) Multilayer perceptron hyperparameters.

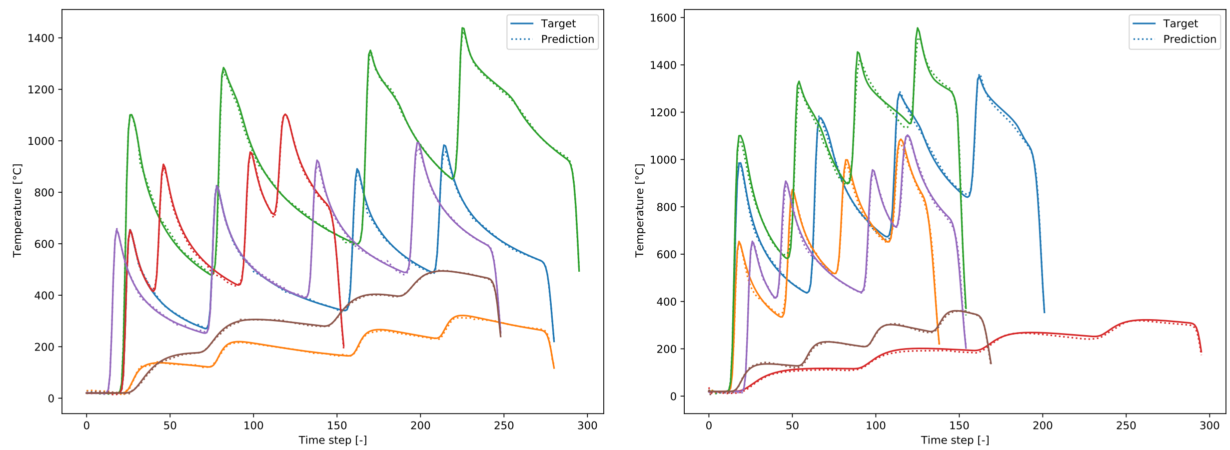
Name	**Symbol**	**Value**
Recurrent cell	-	GRU
Number of recurrent layers	L_{RNN}	2
Hidden state size	H_{RNN}	256
Number of hidden layers	L_{MLP}	1
Number of hidden units	H_{MLP}	1024
Activation functions	-	ReLU
Learning rate	α	0.001
Batch size	B	16
Early stopping epochs	C	8

(b) Recurrent neural network hyperparameters.

Table 6: Comparative tables between MLP vs RNN

3 Additional results

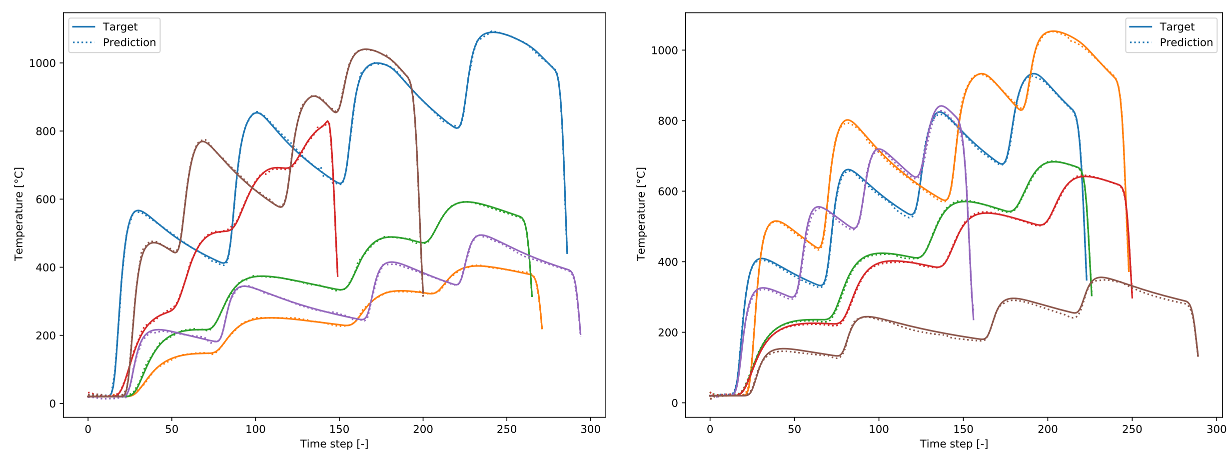
Illustrations of scenarios 1 to 5.



(a) MLP predictions.

(b) RNN predictions.

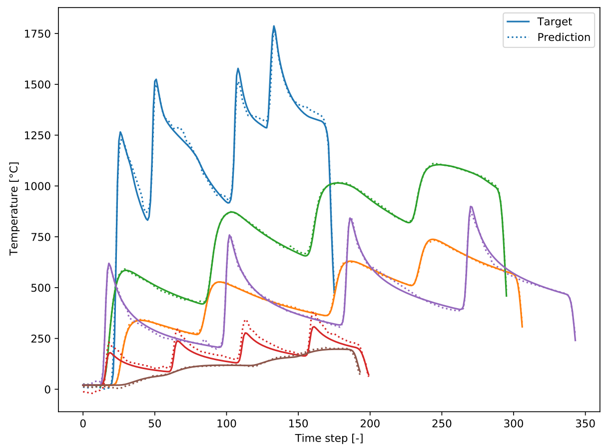
Figure 2: Illustration of the predictions on six random temperature sequences on the test set, for scenario 1.



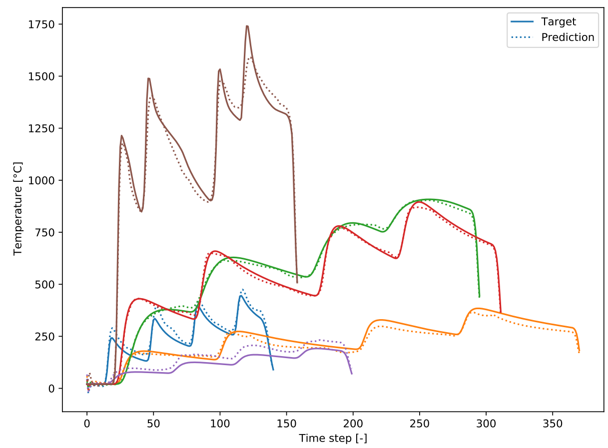
(a) MLP predictions.

(b) RNN predictions.

Figure 3: Illustration of the predictions on six random temperature sequences on the test set, for scenario 2.

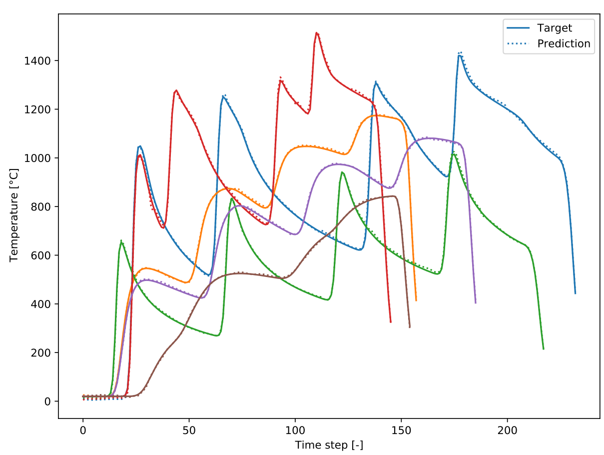


(a) MLP predictions.

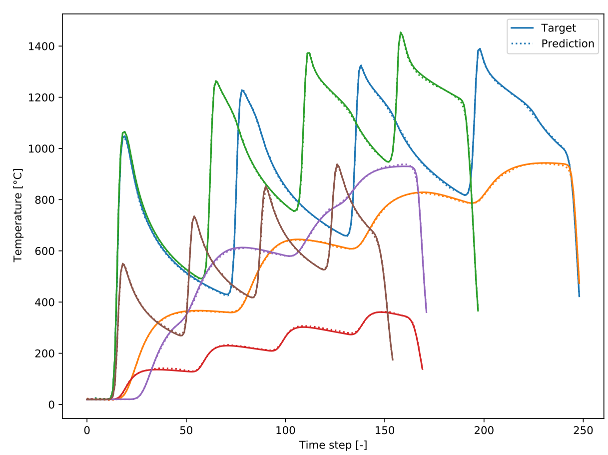


(b) RNN predictions.

Figure 4: Illustration of the predictions on six random temperature sequences on the test set, for scenario 3.

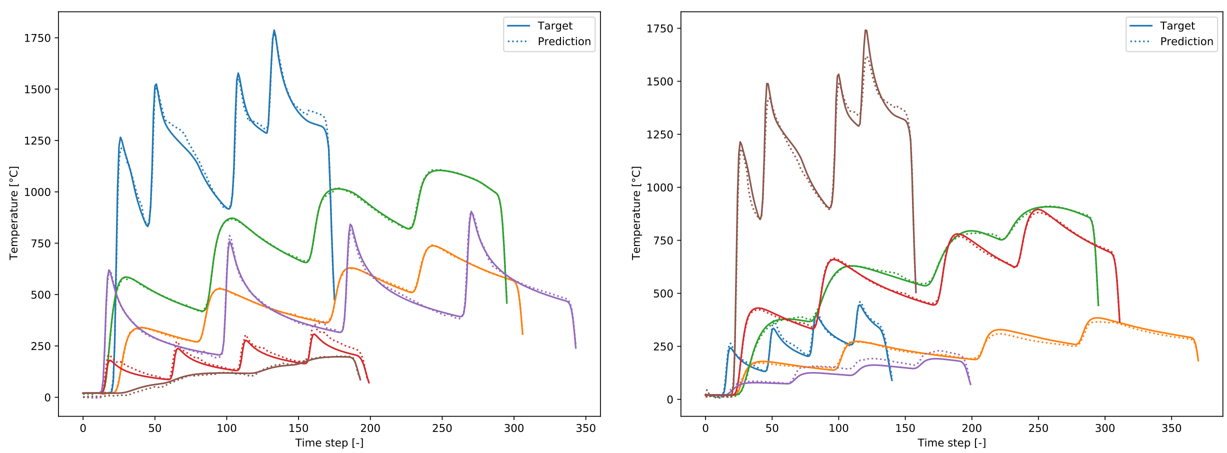


(a) MLP predictions.



(b) RNN predictions.

Figure 5: Illustration of the predictions on six random temperature sequences on the test set, for scenario 4.



(a) MLP predictions.

(b) RNN predictions.

Figure 6: Illustration of the predictions on six random temperature sequences on the test set, for scenario 5.

Bibliography

[1] Seifallah Fetni et al. “Thermal field prediction in DED manufacturing process using Artificial Neural Network”. In: (2021).

[2] Lingbin Meng et al. “Machine Learning in Additive Manufacturing: A Review”. In: (2020). DOI: 10.1007/s11837-020-04155-y.

[3] Mojtaba Mozaffar et al. “Data-driven prediction of the high-dimensional thermal history in directed energy deposition processes via recurrent neural networks”. In: (2018). DOI: 10.1016/j.mfglet.2018.10.002.

[4] Arindam Paul et al. “A real-time iterative machine learning approach for temperature profile prediction in additive manufacturing processes”. In: (2019).