Neural networks for large deformation plasticity. Towards real-time interactive simulations

Louis Lesueur, Anders Thorin, Daniel Weisz-Patrault

Neural networks for large deformation plasticity. Towards real-time interactive simulations

Louis Lesueur
Université Paris-Saclay, CEA, List, F-91120, Palaiseau, France
Laboratoire de Mécanique des Solides, CNRS UMR 7649, École Polytechnique
Institut Polytechnique de Paris, F-91128 Palaiseau, France
louis.lesueur@cea.fr

Anders Thorin
Université Paris-Saclay, CEA, List, F-91120, Palaiseau, France
anders.thorin@cea.fr

Daniel Weisz-Patrault
Laboratoire de Mécanique des Solides, CNRS UMR 7649, École Polytechnique
Institut Polytechnique de Paris, F-91128 Palaiseau, France
daniel.weisz-patrault@polytechnique.edu

Abstract

In the last years, neural networks have been used to learn physical simulations in a wide range of contexts. The present work tackles the training of neural networks for large deformation plasticity. There are two sources of nonlinearity: geometric (large deformation) and material (plasticity). Traditional numerical methods for plastic simulations (such as the Finite Element method) are computationally expensive. NNs architectures have been proposed in plasticity in some simple cases, in the small deformations framework. The main contributions are i) the application of NN for plasticity in large deformations, ii) a review and comparison of the existing methods, iii) the use of a Temporal Convolutional Network that trains faster than the existing methods, iv) an open source benchmark to the scientific community. Altogether, these contributions enable real-time simulation of plastic behaviors.

1 Introduction

This work tackles the training of neural networks to learn the simulation of materials undergoing large elasto-plastic deformations. Nonlinearity stems from two sources: i) a material nonlinearity, coming from the plastic constitutive law, and ii) a geometric nonlinearity, due to the large deformation (also referred to as finite strains) framework.

Plasticity is an irreversible mechanical process which occurs in many applications such as industrial processes (stamping, rolling), metallic additive manufacturing, damage, etc. Because it is a history-dependent process, solving its governing equations requires dedicated and computationally-expensive numerical methods [Dunne and Petrinic, 2005]. In this work, we investigate the training of surrogate models composed of neural networks to accelerate simulations. Speeding up elasto-plastic simulations opens doors to plasticity-based interactive simulations. It may also used in optimization such as Reinforcement Learning (RL), when the cost functional involves plasticity.

Preprint. Under review.

History-dependent processes In terms of Neural Network (NN) architecture, the history-dependent property of plasticity guides the choice of potentially appropriate NNs. In the general framework of AI, history-dependent processes are widely used in natural language processing [Graves, 2014]. Indeed, in a text, the meaning of a word is determined by its predecessors, i.e. its history. History-dependent neural networks have also allowed great advances in the fields of time series prediction [Vaswani et al., 2017].

Machine learning and physics simulation In recent years, machine learning techniques have been successfully applied to a wide variety of mechanical problems.

PINNs are among the most popular techniques [Raissi et al., 2019, Raissi, 2018, Raissi et al., 2018]. In these approaches, a neural network is used to solve a given Partial Differential Equation (PDE). These networks take in input a space-time coordinate and outputs the value of the corresponding learned quantities. They are "physics-informed" in the sense that the cost function is applied both on the data and on a physical model of these data. The problem with most of these approaches is that they require the knowledge of the loading evolution on the whole resolution domain beforehand. They are therefore unsuitable for an interactive paradigm, as the user changes the external load in an unpredictable manner.

Another approach consists in learning the constitutive laws [Zhou et al., 2021, Liu et al., 2021, Linka et al., 2021], which relates stresses and strains. Such laws are slow to simulate with Finite Element software, because of the material and geometric nonlinearities. Once correctly trained, NN evaluate much faster. One way of taking advantage of this speed up lies in the possibility of injecting them into finite elements, and thus circumventing the difficulties related to nonlinearities. This speeds up computations and allows interactivity.

Another approach, chosen here, is to use recurrent networks which are well-suited for sequential simulations in hyperelasticity [Hashash et al., 2004], viscoplasticity [Chen, 2021] or thermoelasticity [Chen et al., 2021].

Sequential machine learning Recurrent networks are designed to process I/O time series (x_t, y_t), and can predict on the fly the output of a step, based on its input, while taking into account the history of previous predictions through a state variable h_t. Three main recurrent network architectures exist: RNNs [Rumelhart et al., 1986], GRUs [Chung et al., 2014] and LSTMs [Hochreiter and Schmidhuber, 1997]. These three architectures have undergone various variations.These architectures are called *recurrent*, since the same network is used to loop recursively on the entries of the time series. A comparison between them can be found in [Cahuantzi et al., 2021].

Note that some recurrent architectures, such as PhyLSTM [Zhang et al., 2020], can adapt to a given class of physical problems and therefore also be considered as a type of "physics-informed" network.

More recently, non-recurrent but also efficient architectures have appeared to process time series. In particular, transformers [Vaswani et al., 2017], which rely on the attention mechanism, have shown their effectiveness in language processing. There are also Temporal Convolutional Network (TCN) [Lea et al., 2016], based on temporal convolutions. Unlike recurrent networks, which only take the input of the current stage, these networks take the entire time series from its beginning to the current stage. In practice, the inference is parallelizable so that it does not take longer than a RNN, but requires more RAM. These architectures are an active and fruitful research topic [Zhou et al., 2020, Grigsby et al., 2021]. Comparison between recurrent networks and TCNs can found in [Lai et al., 2017, Bai et al., 2018].

Machine learning and plasticity The majority of the existing literature regarding machine learning and plasticity is focused on learning the material plastic behavior, in order to eliminate the nonlinearity and to be able to inject the learned model into classical finite elements [Mozaffar et al., 2019, Gorji et al., 2020, Bonatti and Mohr, 2022]. More Recently, other articles have focused on direct learning of free energy [Zhang and Mohr, 2020, Li et al., 2022]. Such approaches are more versatile. They can be used within other numerical methods than finite elements, such as the Material Point Method (MPM) methods. The free energy has also be learned by means of TCNs [Abueidda et al., 2021a].

The methods proposed here are standalone: they are not to be implemented within a numerical method.

Worth noting are PINN-type methods adapted to plasticity, such as [Arora et al., 2022] and [Haghighat et al., 2022]. Also, some similar methods prefer to optimize energy functions rather than displacements [He et al., 2022, Abueidda et al., 2021b]. These methods use conventional dense networks, but it is also possible to determine an appropriate architecture to enforce compliance of plasticity with thermodynamics [He and Chen, 2022].

Finally, although neural network approaches are the most prevalent within ML, it should be noted that one can also find attempts to model plasticity with more traditional data processing algorithms. Notably by symbolic regression [Versino et al., 2017], by minimization of a distance over stress-strain space [Ciftci and Hackl, 2021], or by using kernel-based methods [Gerbaud et al., 2022].

Outline The main difference in this work is that, contrary to all the above-mentioned references, it is not limited to small deformations. The results are demonstrated using a paper clip undergoing large quasi-static transformations, described in Section 2. The dataset generation, the proposed method, the comparison methodology with existing architectures and the training details are provided in Section 3. Results are presented and discussed in Section 4.

2 Problem statement

2.1 Large transformation plasticity

The basis of the theory of plasticity at finite strains is briefly recalled. It will help emphasizing important aspects guiding the choice of appropriate NNs. Readers who are mostly interested in the mathematical governing equations can skip directly to Eqs (14).

The transformation from the reference configuration Ω_0 to the current configuration Ω_t is parametrized by $\underline{x} = \underline{\Phi}(\underline{X}, t)$, where $\underline{\Phi}$ is a bijective function called *transformation*, $\underline{X}$ denotes a particle in the reference configuration, t the time, and $\underline{x}$ the location of the particle in the current configuration. The theory is derived on a volume element $\mathrm{d}\Omega_0$ transformed into $\mathrm{d}\Omega$ by using the *transformation gradient* $\underline{\underline{F}} = \underline{\underline{\nabla}}_{\underline{X}} \underline{\Phi}$. Classical finite strains theory involves a stress-free configuration called *released configuration* that is obtained by unloading the volume element $\mathrm{d}\Omega$. The transformation gradient can therefore be decomposed into the second order tensor $\underline{\underline{F}}_E$ representing the elastic part of the transformation while $\underline{\underline{F}}_P$ represents the plastic part:

$$\underline{\underline{F}} = \underline{\underline{F}}_E \cdot \underline{\underline{F}}_P \tag{1}$$

$\underline{\underline{F}}_P$ is often assumed to be isochoric i.e., $\det \underline{\underline{F}}_P = 1$ so that the volume variation reduces to $J = \det \underline{\underline{F}} = \det \underline{\underline{F}}_E$. Both $\underline{\underline{F}}_E$ and $\underline{\underline{F}}_P$ are incompatible in the sense that they are not gradients of any transformation in general. The decomposition (1) is not unique a priori. The macroscopic plasticity uniqueness is ensured by imposing the symmetry of $\underline{\underline{F}}_E$ (i.e., $\underline{\underline{F}}_E^\top = \underline{\underline{F}}_E$).

Introducing the velocity $\underline{\dot{x}}$, the symmetric part of the velocity gradient denoted by

$$\underline{\underline{d}} = \mathrm{sym}\left[\underline{\underline{\nabla}}_{\underline{x}} \underline{\dot{x}}\right] = \mathrm{sym}\left[\underline{\dot{\underline{\underline{F}}}} \cdot \underline{\underline{F}}^{-1}\right] \tag{2}$$

plays a major role in defining state variables in mechanics. By using the decomposition (1) one defines:

$$\underline{\underline{d}}_E = \mathrm{sym}\left[\underline{\dot{\underline{\underline{F}}}}_E \cdot \underline{\underline{F}}_E^{-1}\right] \quad \text{and} \quad \underline{\underline{d}}_P = \mathrm{sym}\left[\underline{\dot{\underline{\underline{F}}}}_P \cdot \underline{\underline{F}}_P^{-1}\right]. \tag{3}$$

The cumulative plastic strain rate denoted by $\dot{p}_{\mathrm{cum}}$ is then defined as follows:

$$\dot{p}_{\mathrm{cum}} = \sqrt{\frac{2}{3} \underline{\underline{d}}_P : \underline{\underline{d}}_P} \geq 0. \tag{4}$$

The following Green-Lagrange strain tensor characterizes the elastic part of the state:

$$\underline{\underline{e}} = \frac{1}{2}\left(\underline{\underline{F}}_E^\top \cdot \underline{\underline{F}}_E - \underline{\underline{1}}\right) \tag{5}$$

In contrast, plasticity depends a priori on the entire history of the plastic part of the transformation, that is, at any time t, the function $[0, t] \ni \tau \mapsto \underline{\underline{F}}_P(\underline{X}, \tau)$. However, it is often assumed for metals that

only the current value of the plastic tensor $\underline{\underline{F}}_P(\underline{\underline{X}}, t)$ along with the current value of the cumulative plastic strain $p_{\text{cum}}(\underline{\underline{X}}, t)$ are sufficient to fully characterize the plastic part of the state. Therefore, only three state variables are required: the current values of $\underline{\underline{e}}, \underline{\underline{F}}_P$ and p_{cum}.

In addition to the Cauchy stress tensor $\underline{\underline{\sigma}}$, the Mandel and Piola–Kirchoff stress tensors are introduced:

$$\underline{\underline{\Pi}} = J\, \underline{\underline{F}}_E^{-1} \cdot \underline{\underline{\sigma}} \cdot \underline{\underline{F}}_E^{-1\,T} \quad \text{and} \quad \underline{\underline{\kappa}} = J\, \underline{\underline{F}}_E^{-1} \cdot \underline{\underline{\sigma}} \cdot \underline{\underline{F}}_E \tag{6}$$

Within the framework of standard generalized media, the behavior is derived from the balance equation:

$$\underline{\underline{\sigma}} : \underline{\underline{d}} - \rho \left(\dot{\Psi} + \dot{T} s \right) - \frac{\underline{q} \cdot \underline{\underline{\nabla}}_{\underline{x}} T}{T} = D \geq 0 \tag{7}$$

where ρ the density in the current configuration, Ψ is the Gibbs free energy density per unit mass, T the temperature, s the entropy density per unit mass, $\underline{q}$ the heat flux and D the dissipated power per unit volume, which positive in virtue of the second law of thermodynamics. The free Gibbs energy is assumed to only depend on $\underline{\underline{F}}_E, \underline{\underline{F}}_P$ in the following way. It is decomposed into a free energy of distortion $\Psi_{\text{dis}}(\underline{\underline{e}})$ (note that $\underline{\underline{e}}$ depends only on $\underline{\underline{F}}_E$, see Eq. (5)) and free energy blocked in the microstructure $\Psi_{\text{blo}}(\underline{\underline{F}}_P)$:

$$\Psi(\underline{\underline{e}}, \underline{\underline{F}}_P) = \Psi_{\text{dis}}(\underline{\underline{e}}) + \Psi_{\text{blo}}(\underline{\underline{F}}_P) \tag{8}$$

In addition, the dissipation power D is assumed to only depend on the cumulative plastic strain p_{cum}. And since plasticity is rate-independent deformation process, the dissipated power is proportional to p_{cum}:

$$D(p_{\text{cum}}) = \frac{\sigma_Y}{J}\, p_{\text{cum}} \geq 0, \tag{9}$$

where $\sigma_Y > 0$ is the yield stress, which is a function of the states $\underline{\underline{F}}_E, \underline{\underline{F}}_P, p_{\text{cum}}$. It is common to assume that σ_Y only depends on p_{cum}.

Based on previous assumptions and the balance equation (7), we finally obtain the following constitutive relations and flow rule for large deformation (or finite strains) plasticity:

$$\begin{cases} \underline{\underline{\Pi}} = \rho_0 \dfrac{\partial \Psi_{\text{dis}}}{\partial \underline{\underline{e}}} & \text{(10a)} \\ \underline{\underline{X}} = \rho_0\, \underline{\underline{F}}_P \cdot \dfrac{\partial \Psi_{\text{blo}}}{\partial \underline{\underline{F}}_P} & \text{(10b)} \\ \underline{\underline{d}}_P = \dfrac{3}{2} \dfrac{\dot{p}_{\text{cum}}}{\sigma_Y(p_{\text{cum}})} \text{dev}\left[\underline{\underline{\kappa}} - \underline{\underline{X}} \right], & \text{(10c)} \end{cases}$$

where ρ_0 is the density in the reference and released configurations[1], dev denotes the deviatoric part of the tensor it applies to and $\underline{\underline{X}}$ is the center of the elastic domain. In this work, the elastic domain is the convex set defined by the Von Mises yield criterion, which is the set of $\underline{\underline{X}}$ that satisfies the inequality:

$$\sqrt{\frac{3}{2} \text{dev}\left[\underline{\underline{\kappa}} - \underline{\underline{X}} \right] : \text{dev}\left[\underline{\underline{\kappa}} - \underline{\underline{X}} \right]} \leq \sigma_Y(p_{\text{cum}}). \tag{11}$$

In the next subsection, we will see how these governing equations (10) reduce in the use case of a paper clip.

2.2 Paper clip in large deformations

To demonstrate and assess the capability of neural networks to learn elasto-plastic displacements in large transformations, we introduce a use case consisting in a paper clip undergoing large deformation under two mechanical loads, see Fig. 2a. Each mechanical load represents the fingers of someone manipulating the paper clip. The two loading points are selected randomly (uniform distribution). The first point is subjected to a random force sequence $\underline{F}_1$ and a random torque sequence $\underline{M}_1$, with $\|\underline{F}_1\| \leq 1.25\,\text{N}$ and $\|\underline{M}_1\| \leq 0.02\,\text{N m}$. The force $\underline{F}_2$ and torque $\underline{M}_2$ are selected such that the paper clip is globally balanced at each sequence step.

[1] Since $\det\left[\underline{\underline{F}}_P\right] = 1$ the density of the reference and released configurations are identical.

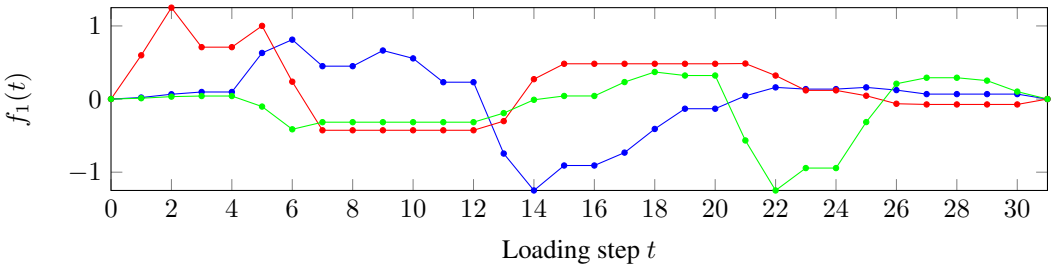
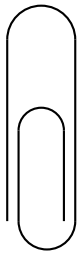
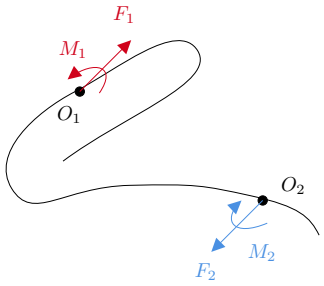


Figure 1: Three examples of randomly chosen f_1 sequences of 32 loading steps.



(a) Geometry of the paper clip at rest.



(b) The paper clip deforms in response to two random balanced forces and torques are exerted.

Figure 2: From an initial condition at rest, the paper clip is subjected to 32 successive quasi-static load steps at two randomly chosen points O_1 and O_2. Each load consists in a applying two balanced random forces and torques at these points, as per Eqs. (12) and (13).

During one sequence, the evolution is a quasi-static, history-dependent problem. Although it does not correspond to physical time, the quasistatic steps are indexed by $t \in \mathbb{N}$. With $\underline{u}_1$ and $\underline{v}_1$ the force and torque unit direction at the first point O_1, the overall evolution is such that:

$$\underline{F_{1,t}} = f_1(t)\underline{u}_1; \quad \underline{M_{1,t}} = m_1(t)\underline{v}_1 \tag{12}$$

where f_1 and m_1 are random functions defined for $t \in \{0, \dots, 31\}$ built by interpolating sine functions between randomly chosen points, see Fig. 1.

For the equilibrium to be verified at any step $t \in [\![0, 31]\!]$, the second force must be defined by:

$$\underline{F_{2,t}} = -\underline{F_{1,t}}; \quad \underline{M_{2,t}} = -\underline{M_{1,t}} + \underline{O_{1,t}O_{2,t}} \times \underline{F_{1,t}}. \tag{13}$$

The elasto-plastic problem now consists in solving Eqs. (10) for the paper clip subject to the forces $\underline{F}_1, \underline{F}_2$ and the torques $\underline{M}_1, \underline{M}_2$, for each of the successive steps $t \in [\![0, 31]\!]$.

Taking advantage of the slenderness of a paper clip and taking into account the large rotations of the sections, Timoshenko beam theory is used to reduce the three-dimensional fields of Eqs. (10) to one-dimensional fields. The corresponding equations, introducing the curvilinear coordinate s, involve the internal forces $\underline{R}_t(s)$ and the internal bending and torque moments $\underline{M}_t(s)$: the local equilibrium equations, written in the Frenet frame of tangent $\underline{\tau}$ for each part where the paper clip is not loaded ($[AO_1)$, (O_1O_2), $(O_2B]$):

$$\frac{\mathrm{d}\underline{R}_t}{\mathrm{d}s} = \underline{0}; \qquad \frac{\mathrm{d}\underline{M}_t}{\mathrm{d}s} + \underline{\tau} \times \underline{R}_t = \underline{0} \tag{14}$$

complemented with the boundary conditions involving $\underline{F}_1, \underline{F}_2$ and $\underline{M}_1, \underline{M}_2$.

Then, material nonlinearities comes into play through a chosen nonlinear hyperelasto-plastic law $\mathcal{G}$. Together with the internal fields $\underline{R}_t(s)$ and $\underline{M}_t(s)$, it determines through the Timoshenko beam kinematics the curvature and elongation:

$$\underline{\gamma}_t, \underline{\eta}_t = \mathcal{G}(\underline{R}_t, \underline{M}_t, p_{\text{cum},t}). \tag{15}$$

$\mathcal{G}$ gathers a hyperelastic constitutive law (10a), which governs elasticity as long as the current state strictly satisfies the von Mises yield criterion (11), and a plastic flow evolution (10c) to compute plasticity contribution to displacements. The paper clip is assumed to be made of steel, so the hardening is purely isotropic and $\underline{\underline{X}} = \underline{\underline{0}}$ in Eq. (10b).

The displacement field $\underline{\xi} = \underline{x} - \underline{X}$ can be recovered from the two geometric quantities $\underline{\gamma}_t$ and $\underline{\eta}_t$ by solving an ODE.

In the end, the problem of interest is therefore to find a function $\mathcal{F}$ able to predict the displacements $\underline{\xi}_{t+1}$, as a function of the loading $(\underline{F}_t, \underline{M}_t)$, the plastic deformation history $p_{\text{cum},t}$ and $\underline{\xi}_t$:

$$\underline{\xi}_{t+1} = \mathcal{F}(\underline{F}_t, \underline{M}_t, \underline{\xi}_t, p_{\text{cum},t}) \tag{16}$$

which is the typical form of a recurrent network.

3 Methodology

3.1 Data generation for supervised learning

The data set corresponding to the described problem is generated with the finite element software CAST3M. Within CAST3M, the paper clip is discretized in regular Timoshenko beam finite elements of length $0.5\,\text{mm}$ (178 elements in total). A Python wrapper was developed to wrap this software, which is not designed for massive data generation. 2500 sequences of 32 loading steps each have been generated. Since the use case involve strong nonlinearities which are numerically demanding, the finite element method within CAST3M was not always able to converge. Therefore, the simulation results was filtered to detect absurd behaviors. The resulting dataset is provided as a supplementary material of the present article, with the objective of proposing a benchmark problem for neural network architectures to the whole community. All the mechanical, geometric and numerical parameters are provided in `json` supplementary material.

The dataset consists in 500 examples of time-series. The input of each sample is a tensor containing the values of applied loads at each step and on every point of the discretized paper clip. The output contains displacements, internal torques and forces, elongations and curvatures, plastic elongations and curvatures,for each step and on every point of the paper clip. Note that in this paper only displacements are learned, but the dataset allows more learning possibilities.

As an illustration of the dataset content, Figure 3 depicts the simulation results for the first steps.

3.2 Networks architectures

3.2.1 Three Recurrent networks

The three classical architectures of recurrent networks are compared on the dataset: RNN, GRU and LSTM. The three considered networks take as input the loading at step t in the form of a tensor of size $(N_{\text{pts}} \times 6)$, as well as the state vector denoted by h_t, and gives as output the displacements, the plastic strains, as well as the state vector at step $t + 1$.

In order to be objective, the same encoder for the loads, with two hidden layers, is used in the three cases. It projects the loadings in a latent space of size 256. Three recurrent cells are chained, then the same decoder is used in output in the three cases, see Figure 4.

3.2.2 TCN

By design, TCN takes as input the whole sequence of loadings up to step t (in the form of a tensor of size $(S \times N_{\text{pts}} \times 6)$), and gives as output the whole sequence of displacements and corresponding plastic deformations, see Figure 5.

To have a point of comparison with the recurrent architectures, the TCN used has the same order of magnitude of parameters (see table). It is possible to use either a shared encoder between the loads, the same as for the recurrent cells, or a transformer encoder.

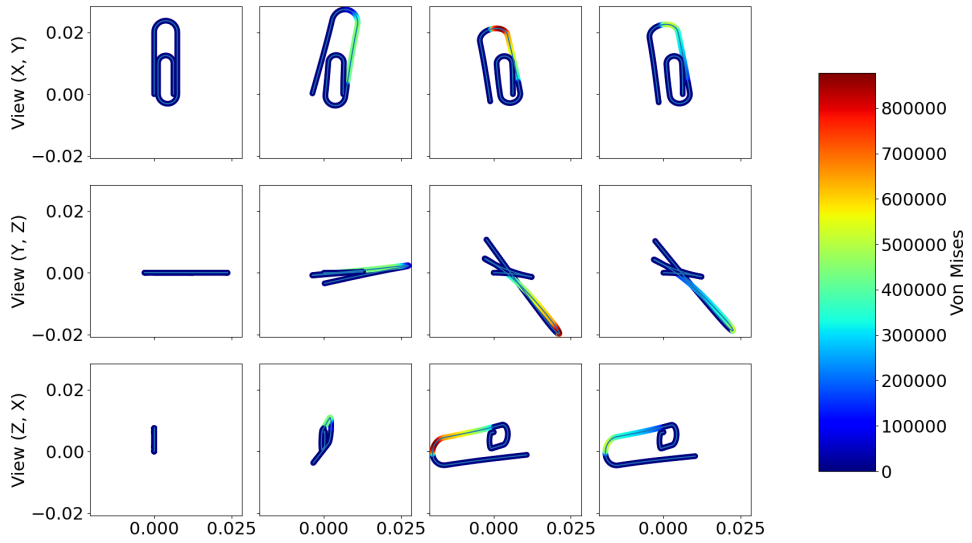


Figure 3: Example of the first 4 steps of a sequence in the dataset generated with CAST3M. Units are meters, and Pascal for the stress.

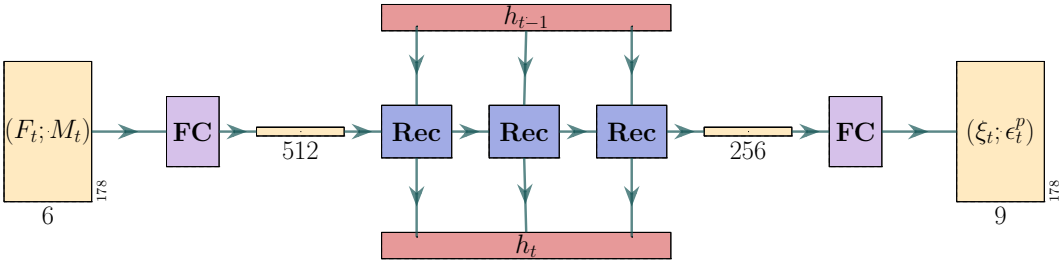


Figure 4: Recurrent neural network testing architecture. Yellow elements are tensors, violet are FC layers, blue are recurrent cells. The hidden state h_t is initialized at 0.

In addition, a variant of the TCN is investigate: it is designed by replacing the FC encoder by a transformer (referred to as TCN-tf in the tables).

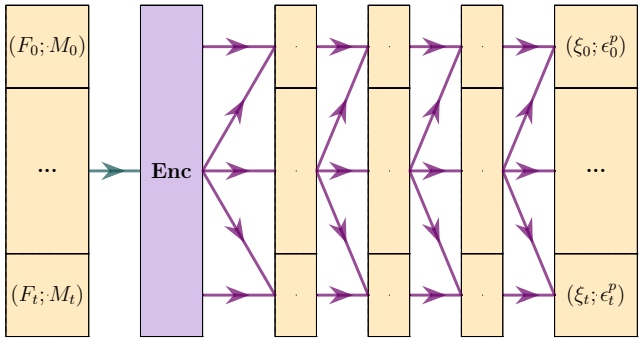


Figure 5: TCN network testing architecture. Purple arrows represent dilated convolutions. At each level, the same dilated convolution is used on each temporal input. The encoder can be the samed shared FC encoder as in the recurrent framework, or a transformer encoder.

3.2.3 Methodology for the architecture comparisons

Number of parameters With regard to recurrent networks, the difference between the number of parameters of each network, as the encoder and decoder are fixed, is only related to the architecture of the recurrent cell that composes it. In the case of TCN, it contains a similar number of parameters to that of GRU. The comparisons that follow therefore focus on the ability of each of these architectures to represent plasticity, at a fixed latent space size.

However, the TCN with transformer encoder has a significant higher amount of parameters than the other tested networks. And this gap is explained by the use of a different encoder. So, to be fair, it can only be compared to the TCN with fully connected encoder, and the comparison will hold on the capacity of the transformer encoder to represent input data in a same sized latent space.

Network	RNN	LSTM	GRU	TCN	TCN-tf
Parameters	22406	23894	23398	23038	91790

Inference time and memory usage As explained in the literature review, the main difference between recurrent networks and TCNs lies in their management of the input series. While the recurrent network loops on each time input, the TCN takes the entire series as input. This has the advantage of being able to parallelize calculations, but generates greater memory usage.

Network	RNN	LSTM	GRU	TCN	TCN-tf
Inference time	0.00071	0.00064	0.00062	0.00061	0.00061

3.3 Networks training

All experiments where conducted on an NVIDIA A100 GPU. Empirically, using an L_1 loss on outputs yielded better results in convergence velocity. This is probably related to the fact that the L_2 loss favors the average values, and therefore converges more easily towards the equilibrium position of the paper clip (zero displacements) rather than to the deformed configuration.

4 Results and discussion

Training times For each architecture, training times are similar (around 1 hour). However, the convergence of TCN with transformer and TCN is faster than others (by 10 to 15min).

Metrics To evaluate the performance of the model on the dataset, several metrics are calculated on the test set:

- Precision at n mm: inspired by 3d reconstruction algorithms. For each predicted position, the distance with its ground truth position is calculated, if it is less than n, the prediction is considered correct. The metric is then the ratio of good predictions to the total number of predictions. It must tend close to 1.
- MSEs between predictions and ground truths.
- Maximum errors on predicted values.

Network	RNN	LSTM	GRU	**TCN**	**TCN-tf**
$\text{prec}_{1\,\text{mm}}$	0.95	0.97	0.97	$\mathbf{0.98}$	$\mathbf{0.98}$
$\text{prec}_{0.5\,\text{mm}}$	0.85	0.87	0.88	0.88	$\mathbf{0.89}$
MSE_ξ	$0.35\,\text{mm}$	$0.28\,\text{mm}$	$0.30\,\text{mm}$	$\mathbf{0.23\,mm}$	$0.25\,\text{mm}$

All tested architectures give satisfying results. However, TCN seems to perform better than classical recurrent units. And this advantage can be amplified by adding a transformer encoder.

The drop of precision between $0.5\,\text{mm}$ and $0.1\,\text{mm}$ is due to the precision used in CAST3M during data generation, which is equal to $0.1\,\text{mm}$ and prevent the network from being more precise.

Visual results Figures 6 and 7 depicts examples of predictions made with the best network (TCN with transformer encoder)

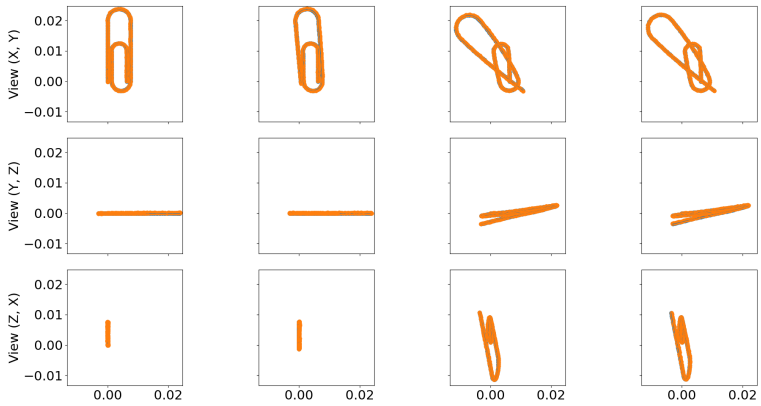


Figure 6: Example of predictions (ground truth in blue, prediction in orange) made with the best model (TCN with transformer).

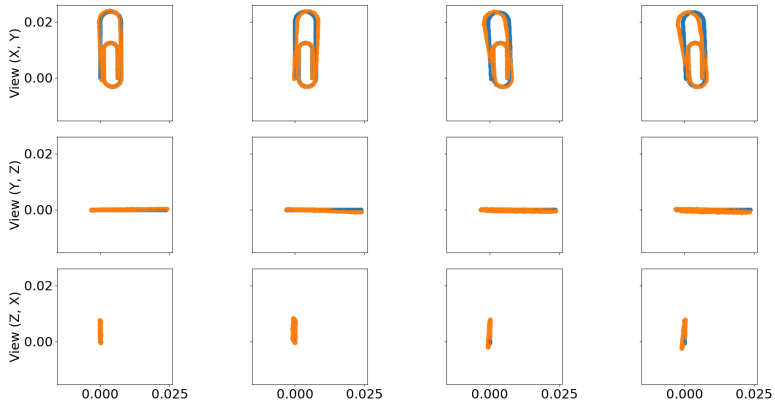


Figure 7: Example of predictions with the worse error in the test set (ground truth in blue, prediction in orange) made with the best model (TCN with transformer).

5 Conclusion

This work tackles the design and training of NNs for the computation of elasto-plastic deformations in the large deformations framework. Because of the material and geometric nonlinearities, traditional solvers (such as the Finite Element Method) for such problems are slow. That prevents interactive simulations involving the manipulation of elasto-plastic objects. Indeed, interactivity requires real-time solving — real time in the sense that it does not lag from the user point of view, ie. time steps of $10\,\mathrm{ms}$, or $1\,\mathrm{ms}$ for *haptic* interactive simulations.

This work focuses on the interactive simulation of the manipulation of a metallic paper clip: this use case combines plasticity, large deformations, and was not possible to simulate interactively before. Indeed, FEM is too slow on this example and NN surrogate models in plasticity were limited to small deformations, which strongly restricts the interest of interactive simulation. The dataset of this benchmark problem is made open source for future comparisons.

Once trained, the proposed TCN is faster than the Finite Element Method by several orders of magnitude: between two orders for small nonlinearities and three orders for strong nonlinearities (which increases the computation time in the FEM).

On the benchmark problem, The TCN outperformed the other classical architectures (RNN, GRU, LSTM),some of which had been proposed in small deformation plasticity.

The next steps will consist in assessing the capability of TCN to simulate large elasto-plastic deformations on more industrial cases: metallic or electric cables, manipulation of flexible items made of plastic or polymers, press-forming, etc. and in the long run, elasto-plasto dynamics.

References

Diab Abueidda, Seid Koric, Nahil Sobh, and Huseyin Sehitoglu. Deep learning for plasticity and thermo-viscoplasticity. International Journal of Plasticity, 136:102852, 01 2021a. doi: 10.1016/j.ijplas.2020.102852.

Diab W. Abueidda, Qiyue Lu, and Seid Koric. Meshless physics-informed deep learning method for three-dimensional solid mechanics. International Journal for Numerical Methods in Engineering, 122(23):7182–7201, oct 2021b. doi: 10.1002/nme.6828. URL https://doi.org/10.1002%2Fnme.6828.

Rajat Arora, Pratik Kakkar, Biswadip Dey, and Amit Chakraborty. Physics-informed neural networks for modeling rate- and temperature-dependent plasticity, 2022. URL https://arxiv.org/abs/2201.08363.

Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling, 2018. URL https://arxiv.org/abs/1803.01271.

Colin Bonatti and Dirk Mohr. On the importance of self-consistency in recurrent neural network models representing elasto-plastic solids. Journal of the Mechanics and Physics of Solids, 158: 104697, 2022. ISSN 0022-5096. doi: https://doi.org/10.1016/j.jmps.2021.104697. URL https://www.sciencedirect.com/science/article/pii/S0022509621003161.

Roberto Cahuantzi, Xinye Chen, and Stefan Güttel. A comparison of lstm and gru networks for learning symbolic sequences. ArXiv, abs/2107.02248, 2021.

Guang Chen. Recurrent neural networks (RNNs) learn the constitutive law of viscoelasticity. Computational Mechanics, 67(3):1009–1019, March 2021. ISSN 0178-7675, 1432-0924. doi: 10.1007/s00466-021-01981-y. URL http://link.springer.com/10.1007/s00466-021-01981-y.

Yu Chen, Jihong Chen, and Guangda Xu. A data-driven model for thermal error prediction considering thermoelasticity with gated recurrent unit attention. Measurement, 184:109891, 2021. ISSN 0263-2241. doi: https://doi.org/10.1016/j.measurement.2021.109891. URL https://www.sciencedirect.com/science/article/pii/S0263224121008319.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling, 2014. URL https://arxiv.org/abs/1412.3555.

Kerem Ciftci and Klaus Hackl. Model-free data-driven simulation of inelastic materials using structured data sets, tangent space information and transition rules, 2021. URL https://arxiv.org/abs/2101.10730.

Fionn Dunne and Nik Petrinic. Introduction to computational plasticity. Oxford University Press, Oxford ; New York, 2005. ISBN 9780198568261.

Paul-William Gerbaud, David Néron, and Pierre Ladevèze. Data-driven elasto-(visco)-plasticity involving hidden state variables. Computer Methods in Applied Mechanics and Engineering, 402: 115394, 2022. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2022.115394. URL https://www.sciencedirect.com/science/article/pii/S004578252200456X. A Special Issue in Honor of the Lifetime Achievements of J. Tinsley Oden.

Maysam B. Gorji, Mojtaba Mozaffar, Julian N. Heidenreich, Jian Cao, and Dirk Mohr. On the potential of recurrent neural networks for modeling path dependent plasticity. Journal of the Mechanics and Physics of Solids, 143:103972, 2020. ISSN 0022-5096. doi: https://doi.org/10.1016/j.jmps.2020.103972. URL https://www.sciencedirect.com/science/article/pii/S0022509620302076.

Alex Graves. Generating sequences with recurrent neural networks, 2014.

Jake Grigsby, Zhe Wang, and Yanjun Qi. Long-range transformers for dynamic spatiotemporal forecasting, 2021. URL https://arxiv.org/abs/2109.12218.

Ehsan Haghighat, Sahar Abouali, and Reza Vaziri. Constitutive model characterization and discovery using physics-informed deep learning, 2022. URL https://arxiv.org/abs/2203.09789.

Youssef Hashash, Sungmoon Jung, and Jamshid Ghaboussi. Numerical implementation of a neural network based material model in finite element analysis. INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING Int. J. Numer. Meth. Engng, 59:989–1005, 02 2004. doi: 10.1002/nme.905.

Junyan He, Diab Abueidda, Rashid Abu Al-Rub, Seid Koric, and Iwona Jasiuk. A deep learning energy-based method for classical elastoplasticity, 2022. URL https://arxiv.org/abs/2209.06467.

Xiaolong He and Jiun-Shyan Chen. Thermodynamically consistent machine-learned internal state variable approach for data-driven modeling of path-dependent materials. Computer Methods in Applied Mechanics and Engineering, 402:115348, dec 2022. doi: 10.1016/j.cma.2022.115348. URL https://doi.org/10.1016%2Fj.cma.2022.115348.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. Neural computation, 9:1735–80, 12 1997. doi: 10.1162/neco.1997.9.8.1735.

Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long- and short-term temporal patterns with deep neural networks, 2017. URL https://arxiv.org/abs/1703.07015.

Colin Lea, Michael D. Flynn, Rene Vidal, Austin Reiter, and Gregory D. Hager. Temporal convolutional networks for action segmentation and detection, 2016. URL https://arxiv.org/abs/1611.05267.

Xuan Li, Yadi Cao, Minchen Li, Yin Yang, Craig Schroeder, and Chenfanfu Jiang. Plasticitynet: Learning to simulate metal, sand, and snow for optimization time integration. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, Advances in Neural Information Processing Systems, 2022. URL https://openreview.net/forum?id=_WqHmwoE7Ud.

Kevin Linka, Markus Hillgärtner, Kian P. Abdolazizi, Roland C. Aydin, Mikhail Itskov, and Christian J. Cyron. Constitutive artificial neural networks: A fast and general approach to predictive data-driven constitutive modeling by deep learning. Journal of Computational Physics, 429:110010, 2021. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2020.110010. URL https://www.sciencedirect.com/science/article/pii/S0021999120307841.

Xin Liu, Su Tian, Fei Tao, and Wenbin Yu. A review of artificial neural networks in the constitutive modeling of composite materials. Composites Part B: Engineering, 224:109152, 2021. ISSN 1359-8368. doi: https://doi.org/10.1016/j.compositesb.2021.109152. URL https://www.sciencedirect.com/science/article/pii/S1359836821005321.

M. Mozaffar, R. Bostanabad, W. Chen, K. Ehmann, J. Cao, and M. A. Bessa. Deep learning predicts path-dependent plasticity. Proceedings of the National Academy of Sciences, 116(52):26414–26420, December 2019. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.1911815116. URL https://pnas.org/doi/full/10.1073/pnas.1911815116.

M. Raissi, P. Perdikaris, and G.E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. Journal of Computational Physics, 378:686–707, 2019. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2018.10.045. URL https://www.sciencedirect.com/science/article/pii/S0021999118307125.

Maziar Raissi. Deep hidden physics models: Deep learning of nonlinear partial differential equations, 2018. URL https://arxiv.org/abs/1801.06637.

Maziar Raissi, Alireza Yazdani, and George Em Karniadakis. Hidden fluid mechanics: A navier-stokes informed deep learning framework for assimilating flow visualization data, 2018. URL https://arxiv.org/abs/1808.04327.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. Nature, 323(6088):533–536, October 1986. ISSN 0028-0836, 1476-4687. doi: 10.1038/323533a0. URL http://www.nature.com/articles/323533a0.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017. URL https://arxiv.org/abs/1706.03762.

Daniele Versino, Alberto Tonda, and Curt A. Bronkhorst. Data driven modeling of plastic deformation. Computer Methods in Applied Mechanics and Engineering, 318:981–1004, 2017. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2017.02.016. URL https://www.sciencedirect.com/science/article/pii/S0045782516314499.

Annan Zhang and Dirk Mohr. Using neural networks to represent von mises plasticity with isotropic hardening. International Journal of Plasticity, 132:102732, 2020. ISSN 0749-6419. doi: https://doi.org/10.1016/j.ijplas.2020.102732. URL https://www.sciencedirect.com/science/article/pii/S0749641919307119.

Ruiyang Zhang, Yang Liu, and Hao Sun. Physics-informed multi-lstm networks for metamodeling of nonlinear structures. Computer Methods in Applied Mechanics and Engineering, 369:113226, 2020. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2020.113226. URL https://www.sciencedirect.com/science/article/pii/S0045782520304114.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting, 2020. URL https://arxiv.org/abs/2012.07436.

Xu-Hui Zhou, Jiequn Han, and Heng Xiao. Learning nonlocal constitutive models with neural networks. Computer Methods in Applied Mechanics and Engineering, 384:113927, 2021. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2021.113927. URL https://www.sciencedirect.com/science/article/pii/S0045782521002644.