# COOPERATIVE DATA-DRIVEN MODELING: CONTINUAL LEARNING OF DIFFERENT MATERIAL BEHAVIOR

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Training deep neural networks (DNNs) in solid mechanics is challenging due to data scarcity. Even when using synthetic datasets obtained from computational simulations, these datasets have limited size because each simulation is time-consuming. One way to address this challenge is to use transfer learning, i.e. finetuning a model pretrained in a different context (e.g., computer vision) such that it can learn a new task while using less data (e.g., learning the behavior of a material with a given microstructure). Unfortunately, a model obtained by transfer learning loses the ability to solve the original task. Therefore, each new task that is being learned destroys the ability to perform the previous one with the same model. We present a Cooperative Data-driven Modeling (CDDM) network that can continually learn tasks without forgetting, accumulating knowledge such that less training data is required when facing a new task or that leads to smaller prediction test error for each new task. We provide our numerical experiments on predicting the plastic behavior of different materials using recurrent neural networks, as they have been shown to handle history-dependent problems.

## 1 INTRODUCTION

Machine learning permeated almost every scientific discipline (Shanmuganathan, 2016; Wuest et al., 2016; Schmidt et al., 2019), and Solid Mechanics is no exception (Bessa et al., 2017; Capuano & Rimoli, 2019; Thakolkaran et al., 2022). With all their merits and flaws (Karniadakis et al., 2021), these algorithms provide a means to understand large datasets, finding patterns and modeling behavior where analytical solutions are challenging to obtain or not accurate enough. Focusing on plasticity modeling, its path-dependency posed a specific machine learning challenge that was recently addressed using recurrent neural networks, where time (or pseudo-time) can be naturally incorporated (Mozaffar et al., 2019).

Nevertheless, there is a serious issue that obstructs the synergistic use of machine learning models by the community. Artificial neural networks, unlike biological neural networks, suffer from catastrophic forgetting (McCloskey & Cohen, 1989; French, 1999; Goodfellow et al., 2013). Human beings when learning a new task, e.g. playing tennis, do not forget how to perform past tasks, e.g. swimming. Unfortunately, artificial neural networks fail at this because they are based on updating their parameters (weights and biases) for the task and data being considered, but this changes the previous configuration obtained for a past task (that led to different values of weights and biases). This catastrophic forgetting has important implications in practice, as illustrated by the following scenario.

Imagine that Team A of scientists collects computational or experimental data about the constitutive behavior of Material A, and then trains an artificial neural network to predict the behavior of that material. In the end, Team A publishes the artificial neural network model and corresponding data according to FAIR (Wilkinson et al., 2016; Jacobsen et al., 2020) principles. Later, if Team B aims to create a model that predicts the behavior of Material B then it faces two options: 1) collect data and train a model from scratch for this new material; or 2) use the model developed by Team A in an attempt to get a better model for Material B and use less data during the training process. If the material behavior for B has some commonality with the one for A, there is an advantage in leveraging the work from Team A. However, the state of the art in the literature is to use transfer learning or meta-learning methods (Caruana, 1994; Vilalta & Drissi, 2002; Pan & Yang, 2010; Tan

et al., 2018) to adapt Model A and retrain it for Team B's scenario (Liu et al., 2019; Lejeune & Zhao, 2020). In this case, the new model obtained by Team B is no longer valid for Team A's scenario. Although Team B may create a model that is valid for its purposes, this would not be a truly cooperative effort with Team A because a general model valid for both scenarios would not be obtained. This represents a significant challenge to cooperative data-driven modeling because it discourages different groups from working towards a common model, ultimately leading to many independent models. Note that the problem gets worst as more tasks accumulate (more materials and more teams).

A new branch of machine learning called continual or lifelong learning (Thrun & Pratt, 1998; Zenke et al., 2017; Parisi et al., 2019) is recently opening new avenues to address the catastrophic forgetting issue. Despite being at an early stage, we believe that addressing this limitation will unlock a new era of cooperative data-driven modeling traversing all fields of application.

## 2 Proposed Method

We advocate here that architectural continual learning algorithms such as PackNet (Mallya & Lazebnik, 2018) or CP&S (Dekhovich et al., 2023) enable Cooperative Data-Driven Modeling (CDDM). The method creates different subnetworks within an artificial neural network that are associated with particular tasks without forgetting past tasks. For example, in our context, each subnetwork is associated with a specific material model for a class of materials with a given microstructure, constituent properties and external conditions (see e.g., Bessa et al. (2017)). Architectural methods, however, are limited by the availability of free neural connections that can be trained for a new task. A particular advantage of CP&S is that it is based on the NNrelief iterative pruning strategy (Dekhovich et al., 2021) which was developed aiming to create sparser subnetworks (using fewer connections) than other pruning methods such as magnitude pruning (Han et al., 2015) or neurons pruning (Hu et al., 2016). Additionally, although CP&S was implemented in the original article for convolutional and fully connected networks, it can easily be adapted to GRU networks as shown in this work.

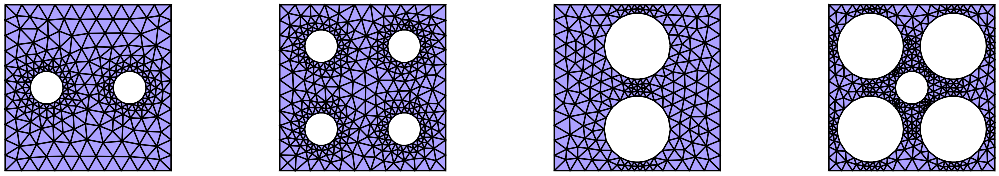
CP&S is based on a set of simple steps to create a group of overlapping subnetworks, each of them learning a particular task without disrupting the knowledge accumulated by the other subnetworks. Importantly, the subnetworks can (and usually do) share knowledge among themselves by sharing connections that are useful to each other. This mechanism allows to learn different tasks, and transfer knowledge between them but avoids forgetting, unlike transfer learning methods. Overall, cooperative data-driven modeling via the proposed CP&S method is described in Algorithm 1.

**Algorithm 1** Pseudocode for training procedure

**Require:** network $\mathcal{N}$, datasets $\{\mathcal{D}^t\}_{t=1}^{T}$. Initialize learning parameters $p$ (learning rate, weight decay, number of epochs, etc. ), pruning parameters (for NNrelief algorithm: $\alpha$ and the number of pruning iterations $k$)

1: **for** $t = 1, 2, \ldots, T$ **do**
2: $\quad\mathcal{N}^t \leftarrow$ Pruning$(\mathcal{N}, \mathcal{D}^t, \alpha, k)$
3: $\quad$freeze parameters $w \in \mathcal{N}^t$ and do not update them again
4: **end for**
5: **return** network $\mathcal{N}$ that learned tasks $1, 2, \ldots, t+1$.

LSTMs (Hochreiter & Schmidhuber, 1997) and GRUs (Cho et al., 2014) have been shown to learn history-dependent phenomena in mechanics (Mozaffar et al., 2019; Chen, 2021). In this work, we follow the scheme proposed by Mozaffar et al. (2019) and use GRU to model the constitutive law of materials, i.e. the ability to predict the average stress tensor (forces per unit area) of a material domain when it is subjected to an average strain tensor (displacements per unit length) – see Bessa et al. (2017); Mozaffar et al. (2019) for details. Figure 1 shows four examples of two-dimensional material domains, where each of them represents a particular material microstructure for which we aim to learn its constitutive law.



(a) Task A. (b) Task B. (c) Task C. (d) Task D.

Figure 1: Four different material domains defining the four different tasks to be learned. Each task is a simple square domain of a porous material (a set of holes in a matrix) that is then deformed by stretching the top edge uniformly according to a defined average strain (displacement at the top divided by the length of the side edge). Training consists of mapping the average strains (input) to the path-dependent average stresses (output), where data is obtained from finite element simulations.

## 3 NUMERICAL EXPERIMENTS

The above-mentioned four different materials have different porous microstructures but the same matrix phase – a simple von Mises plasticity model de Souza Neto et al. (2011). For each task, the corresponding material domain is subjected to the same 1000 paths of deformation. These 1000 paths are obtained from 100 end displacement values of the top boundary of the domain that are sampled from a Gaussian Process posterior, conditioned on 20 displacement values sampled from a uniform distribution for each path. Then the average stress obtained for each path is calculated via the finite element method using FEniCS (Logg et al., 2012).

The dataset is made available, and the simulations used to obtain it can be replicated via the source code. Each presented task was subjected to the same deformation paths to calculate the domain-specific average stresses. Our notation refers to the average strain as $\bar{\boldsymbol{\varepsilon}}_{1:t}$, and Cauchy stress as $\bar{\boldsymbol{\sigma}}_{1:t}$ where $1:t$ indicates the $t$ points of strain that form each path (every path has the same number of points, see more details in Appendix A.1). Then the learning problem for a single task can be defined as, $\bar{\boldsymbol{\sigma}}_{1:t} = f(\bar{\boldsymbol{\varepsilon}}_{1:t})$, where a machine learning model $\mathcal{M}$ is utilized to find the relationship $f : \bar{\boldsymbol{\varepsilon}}_{1:t} \mapsto \bar{\boldsymbol{\sigma}}_{1:t}$.

We highlight that different strategies for sampling the paths of plasticity material laws have been proposed (Wu et al., 2020) and that this choice can affect the number of paths needed to train the neural network up to the desired accuracy.

### 3.1 RESULTS

To evaluate the performance of CDDM we measure the error $\mathcal{E}_i$ for every test path $i$ as follows:

$$\mathcal{E}_i = \frac{1}{3}\left(\frac{||\boldsymbol{\sigma}^i_{11} - \hat{\boldsymbol{\sigma}}^i_{11}||_F}{||\boldsymbol{\sigma}^i_{11}||_F} + \frac{||\boldsymbol{\sigma}^i_{22} - \hat{\boldsymbol{\sigma}}^i_{22}||_F}{||\boldsymbol{\sigma}^i_{22}||_F} + \frac{||\boldsymbol{\sigma}^i_{12} - \hat{\boldsymbol{\sigma}}^i_{12}||_F}{||\boldsymbol{\sigma}^i_{12}||_F}\right)\cdot 100\%,\ i = 1, 2, \ldots, N, \quad (1)$$

where $N$ is the number of paths, $||\cdot||_F$ is a Frobenius norm, $\hat{\boldsymbol{\sigma}}^i_{11}, \hat{\boldsymbol{\sigma}}^i_{22}, \hat{\boldsymbol{\sigma}}^i_{12} \in \mathbb{R}^t$ are predicted stress components along path $i$ formed by $t$ points, and $\boldsymbol{\sigma}^i_{11}, \boldsymbol{\sigma}^i_{22}, \boldsymbol{\sigma}^i_{12} \in \mathbb{R}^t$ are the test ones. To show the robustness of CDDM to the task ordering, we consider four orders: 1) Task A $\rightarrow$ Task B $\rightarrow$ Task C $\rightarrow$ Task D; 2) Task B $\rightarrow$ Task D $\rightarrow$ Task A $\rightarrow$ Task C; 3) Task C $\rightarrow$ Task A $\rightarrow$ Task D $\rightarrow$ Task B; 4) Task D $\rightarrow$ Task C $\rightarrow$ Task B $\rightarrow$ Task A.

Then, we compute the average over all $N$ test points to compute the final test error:

$$Err = \frac{1}{N}\sum_{i=1}^{N} \mathcal{E}_i. \quad (2)$$

We train the GRU with 2 cells and 128 units in the sequence of four tasks, see more training details in Appendix A.2. We train the first task with 800 training paths, and for each of the following tasks, we consider the cases of 800, 400, 200 and 100 training paths. We compare these results with the conventional case (non-cooperative) where every new task is trained with the same GRU but independently of the other tasks. Fig. 2 shows this comparison for one of the orderings, where the blue bars refer to the cooperative model (CDDM) and the orange ones to the conventional case (standard training). The test error is computed using Eq. 1. It is clear that CDDM significantly outperforms standard training when we decrease the number of training points. This effect is consistent across all four orders, independently of which task is considered to be the first. In Table 1, we present the average error for every task when considering a different number of training points (number of paths); note that this error is the average over the four task orderings. Figure 5 includes the error obtained for each task ordering. It is clear that CDDM shows almost the same result if we have a big amount of training data (800 paths) and starts to outperform standard training with one network per task when we decrease the number of training paths. The gap for the first task is explained by a small increase in performance after pruning and retraining.

Moreover, we learn all four tasks with one network, while four separate networks are necessary for the conventional case. Therefore, using fewer parameters and achieving better performance which can be explained by the knowledge transfer that happens between subnetworks (see Appendix A.3). In addition, it should be noted that the network still has free space to learn future tasks, although saturation would occur soon if more tasks were considered because the neural network is small.

Table 1: Test error (%) averaged over four considered task orders.

| | task 1 | tasks 2–4 | | | |
|---|---|---|---|---|---|
| | 800 paths | 800 paths | 400 paths | 200 paths | 100 paths |
| standard training | 2.02 | **2.02** | 2.82 | 3.75 | 6.43 |
| CDDM | **1.83** | 2.06 | **2.44** | **3.08** | **4.37** |

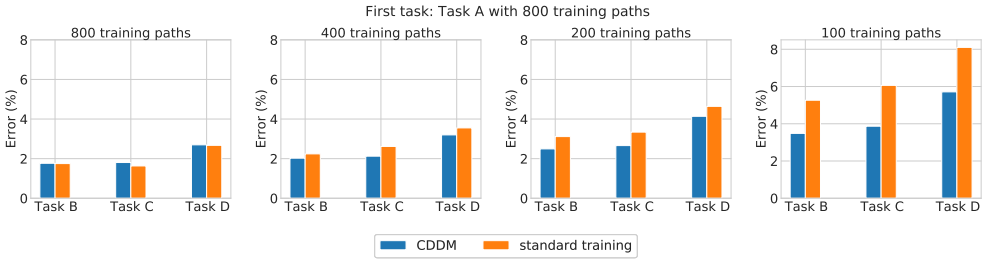



Figure 2: CDDM results on tasks ordering A → B → C → D.

## 4 Conclusion

This work introduces the concept of continual learning in mechanics and illustrates it by considering a history-dependent plasticity problem. To the best of our knowledge, this is the first example of the application of continual learning in this context. We demonstrate that a recurrent neural network can sequentially learn multiple tasks, without replaying data from previous tasks and without forgetting – an important distinction when comparing to transfer learning methods, and a key enabler of cooperative modeling. The proposed method is based on creating task-related subnetworks that transfer knowledge from each other by sharing neural connections. This is demonstrated to decrease the number of training data required to learn a new task (in this case, a new material law). The approach is robust to different task orders. The authors share their belief that the proposed cooperative data-driven modeling concept has a lot of potential for application and future development.

## REFERENCES

M.A. Bessa, R. Bostanabad, Z. Liu, A. Hu, Daniel W. Apley, C. Brinson, W. Chen, and Wing Kam Liu. A framework for data-driven analysis of materials under uncertainty: Countering the curse of dimensionality. *Computer Methods in Applied Mechanics and Engineering*, 320:633–667, June 2017. ISSN 00457825. doi: 10.1016/j.cma.2017.03.037.

German Capuano and Julian J Rimoli. Smart finite elements: A novel machine learning application. *Computer Methods in Applied Mechanics and Engineering*, 345:363–381, 2019.

Rich Caruana. Learning many related tasks at the same time with backpropagation. *Advances in neural information processing systems*, 7, 1994.

Guang Chen. Recurrent neural networks (RNNs) learn the constitutive law of viscoelasticity. *Computational Mechanics*, pp. 11, 2021.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

Eduardo A de Souza Neto, Djordje Peric, and David RJ Owen. *Computational methods for plasticity: theory and applications*. John Wiley & Sons, 2011.

Aleksandr Dekhovich, David MJ Tax, Marcel HF Sluiter, and Miguel A Bessa. Neural network relief: a pruning algorithm based on neural activity. *arXiv preprint arXiv:2109.10795*, 2021.

Aleksandr Dekhovich, David MJ Tax, Marcel HF Sluiter, and Miguel A Bessa. Continual prune-and-select: class-incremental learning with specialized subnetworks. *Applied Intelligence*, pp. 1–16, 2023.

Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.

Ian J Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv preprint arXiv:1312.6211*, 2013.

Song Han, Jeff Pool, John Tran, and William J Dally. Learning both weights and connections for efficient neural networks. *arXiv preprint arXiv:1506.02626*, 2015.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Hengyuan Hu, Rui Peng, Yu-Wing Tai, and Chi-Keung Tang. Network trimming: A data-driven neuron pruning approach towards efficient deep architectures. *arXiv preprint arXiv:1607.03250*, 2016.

Annika Jacobsen, Ricardo de Miranda Azevedo, Nick Juty, Dominique Batista, Simon Coles, Ronald Cornet, Mélanie Courtot, Mercè Crosas, Michel Dumontier, Chris T Evelo, et al. Fair principles: interpretations and implementation considerations, 2020.

George Em Karniadakis, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

Emma Lejeune and Bill Zhao. Exploring the potential of transfer learning for metamodels of heterogeneous material deformation, October 2020.

Zeliang Liu, C. T. Wu, and M. Koishi. Transfer learning of deep material network for seamless structure–property predictions. *Computational Mechanics*, 64(2):451–465, August 2019. ISSN 0178-7675, 1432-0924. doi: 10.1007/s00466-019-01704-4.

Anders Logg, Kent-Andre Mardal, and Garth Wells (eds.). *Automated Solution of Differential Equations by the Finite Element Method*, volume 84 of *Lecture Notes in Computational Science and Engineering*. Springer, Berlin, Heidelberg, 2012. ISBN 978-3-642-23098-1 978-3-642-23099-8. doi: 10.1007/978-3-642-23099-8.

Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 7765–7773, 2018.

Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pp. 109–165. Elsevier, 1989.

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.

Sinno Jialin Pan and Qiang Yang. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, October 2010. ISSN 1558-2191. doi: 10.1109/TKDE.2009.191.

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.

Jonathan Schmidt, Mário RG Marques, Silvana Botti, and Miguel AL Marques. Recent advances and applications of machine learning in solid-state materials science. *npj Computational Materials*, 5(1):1–36, 2019.

Subana Shanmuganathan. Artificial neural network modelling: An introduction. In *Artificial neural network modelling*, pp. 1–14. Springer, 2016.

Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. A survey on deep transfer learning. In *International conference on artificial neural networks*, pp. 270–279. Springer, 2018.

Prakash Thakolkaran, Akshay Joshi, Yiwen Zheng, Moritz Flaschel, Laura De Lorenzis, and Siddhant Kumar. Nn-euclid: deep-learning hyperelasticity without stress data. *arXiv preprint arXiv:2205.06664*, 2022.

Sebastian Thrun and Lorien Pratt (eds.). *Learning to Learn*. Springer US, Boston, MA, 1998. ISBN 978-1-4613-7527-2 978-1-4615-5529-2. doi: 10.1007/978-1-4615-5529-2.

Ricardo Vilalta and Youssef Drissi. A perspective view and survey of meta-learning. *Artificial intelligence review*, 18(2):77–95, 2002.

Mark D Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E Bourne, et al. The fair guiding principles for scientific data management and stewardship. *Scientific data*, 3(1):1–9, 2016.

Ling Wu, Nanda Gopala Kilingar, Ludovic Noels, et al. A recurrent neural network-accelerated multi-scale model for elasto-plastic heterogeneous materials subjected to random cyclic and non-proportional loading paths. *Computer Methods in Applied Mechanics and Engineering*, 369: 113234, 2020.

Thorsten Wuest, Daniel Weimer, Christopher Irgens, and Klaus-Dieter Thoben. Machine learning in manufacturing: advantages, challenges, and applications. *Production & Manufacturing Research*, 4(1):23–45, 2016.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, pp. 3987–3995. PMLR, 2017.

# A APPENDIX

## A.1 DATA GENERATION DETAILS

In this work, an elastoplastic von Mises material without hardening is investigated to create a path-dependent problem. A fixed-sized square is utilized as a domain and holes of varying sizes and locations are placed inside the domain to create different tasks. The tasks originating from different domains can be seen in Figure 1. For all the tasks the bottom part of the domain is fixed and the top part is deformed according to a uniform displacement in $u_{x_{1:t}}$ and $u_{y_{1:t}}$ (see Figure 3).

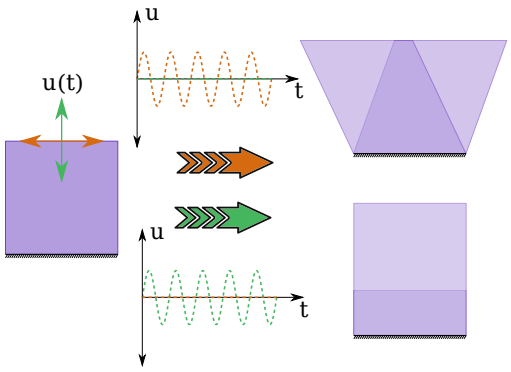



Figure 3: A square domain fixed on the bottom and displaced on the top. Displacement is done in a pseudo-time

In Figure 4, we show one example of strain and stress paths.

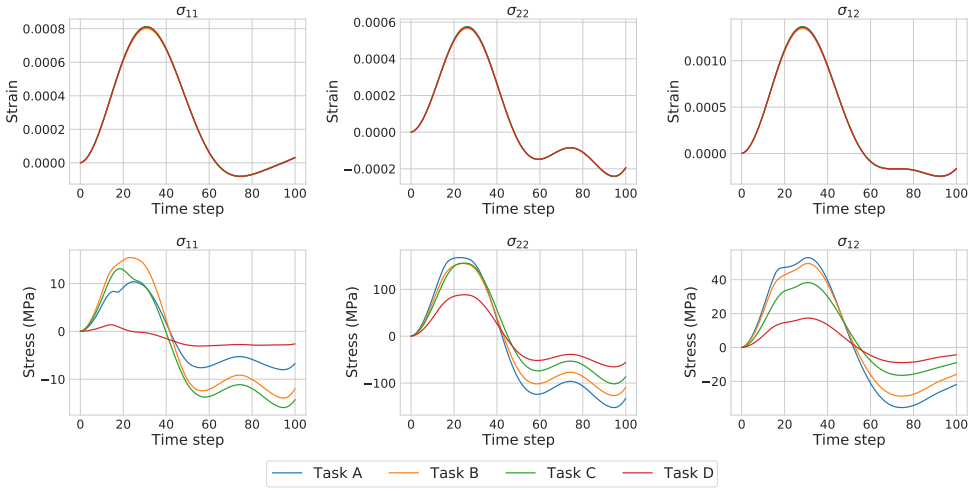



Figure 4: An example of strain and stress paths.

## A.2 TRAINING HYPERPARAMETERS

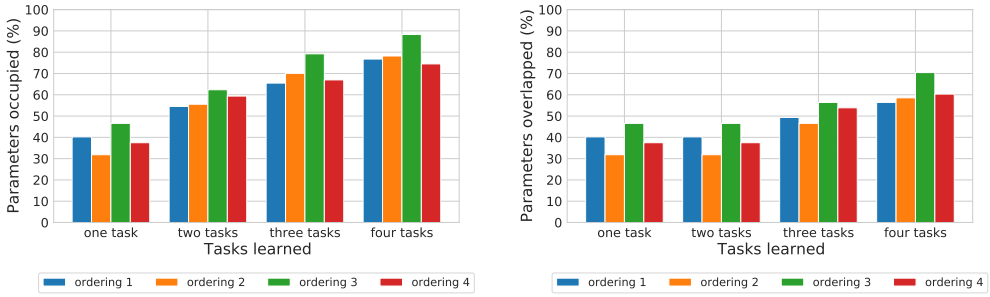
In Table 2, we present the training details for CDDM. The number of pruning iterations and parameter $\alpha$ and retraining epochs refer to NNreilef pruning algorithms.

## A.3 KNOWLEDGE TRANSFER

A crucial characteristic of CDDM is robustness to different orderings, both in terms of predictive error and the number of parameters used for every task. We show the relation between subnetworks

Table 2: Training hyperparameters.

| training epochs | learning rate | weight decay | pruning iterations | pruning parameter $\alpha$ | retraining epochs |
|---|---|---|---|---|---|
| 1000 | 0.01 | $10^{-6}$ | 1 | 0.95 | 200 |




(a) Occupied parameters.

(b) Shared parameters.

Figure 5: Subnetworks analysis: occupied and shared parameters.

in the case where the first task is trained with 800 paths and all the other tasks with 200 training paths. In Fig. 5(a), we show which percentage of the total number of parameters is occupied after a new task is learned. We compare these percentages across four orderings, and in general, we observe the consistency in the number of used parameters with insignificant differences. This means that the ordering of tasks has a negligible effect on how many parameters are occupied in the end.

At the same time, in Fig. 5(b), we demonstrate the percentage of shared connections while the model learns a new task. So, for instance, we observe in ordering 3 that up to 70% of parameters are assigned to more than one subnetwork when all tasks are learned. Overall, at least 55% of the parameters are shared between multiple tasks without a negative impact on model performance. From the figure, it is clear that the changes in the numbers of shared parameters are consistent across all orderings meaning the robustness of CDDM to different task sequences.