Sparse Training Theory for Scalable and Efficient Agents

Blue Sky Ideas Track

Decebal Constantin Mocanu
University of Twente
Enschede, the Netherlands

Elena Mocanu
University of Twente
Enschede, the Netherlands

Tiago Pinto
Polytechnic Institute of Porto
Porto, Portugal

Selima Curci
Eindhoven University of Technology
Eindhoven, the Netherlands

Phuong H. Nguyen
Eindhoven University of Technology
Eindhoven, the Netherlands

Madeleine Gibescu
Utrecht University
Utrecht, the Netherlands

Damien Ernst
University of Liège
Liège, Belgium

Zita A. Vale
Polytechnic Institute of Porto
Porto, Portugal

ABSTRACT

A fundamental task for artificial intelligence is learning. Deep Neural Networks have proven to cope perfectly with all learning paradigms, i.e. supervised, unsupervised, and reinforcement learning. Nevertheless, traditional deep learning approaches make use of cloud computing facilities and do not scale well to autonomous agents with low computational resources. Even in the cloud, they suffer from computational and memory limitations, and they cannot be used to model adequately large physical worlds for agents which assume networks with billions of neurons. These issues are addressed in the last few years by the emerging topic of sparse training, which trains sparse networks from scratch. This paper discusses sparse training state-of-the-art, its challenges and limitations while introducing a couple of new theoretical research directions which has the potential of alleviating sparse training limitations to push deep learning scalability well beyond its current boundaries. Nevertheless, the theoretical advancements impact in complex multi-agents settings is discussed from a real-world perspective, using the smart grid case study.

KEYWORDS

Intelligent Agents, Autonomous Agents, Sparse Training, Sparse Neural Networks, Scalable Deep Learning, Smart Grid

1 INTRODUCTION

In recent years, deep learning has become an alternative name for artificial intelligence (AI), while many other fields have remained behind. One one side, this is due to the success of deep learning in solving some real-world open problems in computer vision and

natural language processing [22, 42], and on the other side due to the slower pace of advancement made by the other AI fields, including here the world of autonomous agents [9]. Nevertheless, one of the main goals of autonomous agents is to behave *intelligently* and to be able to learn and reason in order to take optimal decisions and actions based on their perception of their world (environment).

The main difficulty in using deep learning capabilities in the field of autonomous agents is that deep learning needs extensive computing facilities in the cloud and almost production-ready software libraries to work properly. The high computational cost of deep learning [18], given mainly by the very high number of parameters (or connections) in dense neural networks (NNs), prohibits the agents which many times run in low-resource (or embedded) devices to properly make use of the deep learning architectures – including here the capacity of (re)training the model directly in the agent environment without the need of the cloud infrastructure. Even the agents running in the cloud suffer and are limited by the high deep learning costs. Up to our best knowledge, these problems have started to be preliminary discussed in our AAMAS 2019 tutorial [1]. Herein, we take these ideas to the next level, by starting to formalize the theory and by formulating the open research questions, challenges, and possible solutions.

Firstly, this paper advocates that alternative solutions for the high computational costs of traditional deep learning paradigms have started to emerge, i.e. the sparse-to-sparse training paradigm (or, on short, sparse training) [27] to train sparse NNs from scratch [7, 31, 32]. Auxiliary, and not fully understood yet, this paradigm may also bring better performance and generalization power than dense NNs as empirically observed by [11, 28, 37]. Secondly, the paper presents the advantages of using sparse training for autonomous agents and discusses the main challenges which have to be solved in order to take full benefits of these advantages. To the end, the paper analyzes hypothetically intelligent agents enhanced with sparse training capabilities in real-world complex systems.

2 SPARSE NEURAL NETWORKS

2.1 Definitions

A typical neural network relies on a dataset $\mathcal{D}$ of labeled pairs (x, y) drawn from a distribution D over the input/output spaces X and Y. The goal is to use $\mathcal{D}$ to search a neural network function
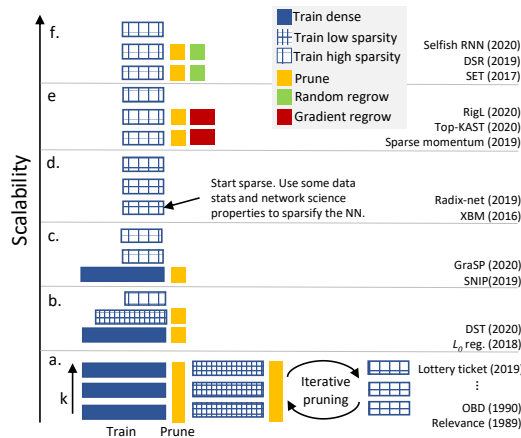
Figure 1: Schematic representation of various method types used to obtain sparse neural networks and a rough estimation of their scalability; a. Pruning, b. Simultaneously training and pruning, c. One-shot pruning, d. Sparse training (static), e. Sparse training (dynamic - gradient), f. Sparse training (dynamic - random).

parameterized by ϕ, with $\phi_w \in \mathbb{R}^d$ that has low expected test loss $l(\phi_w(x), y)$ when using x to predict the associated y on unseen samples drawn from $\mathcal{D}$, as measured by the loss $l : Y \times Y \rightarrow [0, \infty)$.

Definition 2.1. A dense neural network has an underlying connectivity graph $G(\mathcal{V}, \mathcal{E})$, with $\mathcal{V}$ representing the set of vertices (nodes or neurons) and $\mathcal{E}$ representing the set of edges (connections or parameters) composed by many densely – even fully – connected components organized in layers.

Definition 2.2. A sparse neural network has an underlying connectivity graph $G(\mathcal{V}, \mathcal{E}')$ induced on the same set $\mathcal{V}$ of nodes and a subset $\mathcal{E}' \subset \mathcal{E}$ of edges, where the cardinal number of set $|\mathcal{E}'| \ll |\mathcal{E}|$, leading to a much sparser components connectivity.

There is a myriad of options to define the graph G underlying a NN and this makes it very difficult to study a general form of it. For simplicity, and without the loss of generality, further we will refer to the graph $G(\mathcal{V}, \mathcal{E})$ as one of the most used and simple component of a NN, i.e. a fully-connected (FC) bipartite graph between two consecutive layers of neurons.

2.2 Dense training

In the case of FC bipartite layers, $|\mathcal{E}| \approx |\mathcal{N}|^2$. In practice, training a dense NN on $\mathcal{T} \in \mathcal{D}$ data-points and a batch size b requires a large number of weights updates. A typical NN training procedure using Stochastic Gradient Descent would perform a series of weight updates for each parameter ϕ of the network (the elements of $\mathcal{E}$) to minimize the loss l. Let's note $k \in \mathbb{N}$ the hyperparameter reflecting how many times the training set is parsed (in other words, the number of training epochs) until l converges. The number of weight updates until convergence is given by:

$$k \frac{|\mathcal{T}|}{b} |\mathcal{E}| \tag{1}$$

2.3 Dense-to-sparse training

The first class of methods used to obtain sparse neural networks offers benefits just in the inference phase. Broadly speaking, they start training from densely connected networks followed by pruning of unimportant connections.

Traditional iterative pruning (Fig. 1a). Aiming to reduce the computational complexity of a NN, a large amount of research was focused on *pruning*. The term was coined in earlier '90 by Mozer and Smolensky [36] and LeCun et al. [23] who proposed Optimal Brain Damage (OBD). It introduces the idea of first training completely a dense neural networks, then prune unimportant connections, and after that train and prune successively for p steps the remaining sparse network until a trade-off between performance and network size is reached. Han et al. [15] refined the idea in the context of deep neural networks, while the Lottery Ticket Hypothesis [12] shows that it is sufficient to set $p = 2$ if the pruned networks is trained from random initialized weight values. The typical number of weight updates is given by:

$$\sum_p k \frac{|\mathcal{T}|}{b} |\mathcal{E}'_p|, \text{ where } \mathcal{E}'_0 = \mathcal{E} \text{ such that } \mathcal{E}'_{p+1} \subset \mathcal{E}'_p, \forall p \tag{2}$$

Simultaneously training and pruning (Fig. 1b). Louizos et al. [29] proposed in 2018 a slightly more efficient procedure which starts from a dense NN and trains and prunes simultaneously the network parameters using L_0 regularization. DST revised the idea in [25]. Typically, the total number of weight updates are given by:

$$\sum_k \frac{|\mathcal{T}|}{b} |\mathcal{E}'_k|, \text{ where } \mathcal{E}'_0 = \mathcal{E} \text{ such that } |\mathcal{E}'_{k+1}| < |\mathcal{E}'_k|, \forall k \tag{3}$$

One-shot-pruning (Fig. 1c). SNIP in 2019 [24] and GrasSP in 2020 [43] have shown that it is possible to prune many large NNs faster, by training the dense network just for a very short period of time (not until convergence) on a data subset $\mathcal{T}' \subset \mathcal{T}$. To obtain also a decent performance the training process is continued on the remained sparse network on the whole dataset $\mathcal{T}$ [48]. Approximately, the number of weight updates is:

$$\sum_i \frac{|\mathcal{T}'|}{b} |\mathcal{E}| + \sum_k \frac{|\mathcal{T}|}{b} |\mathcal{E}'|, \text{ where } \mathcal{T}' \subset \mathcal{T}, i \ll k, |\mathcal{E}'| \ll |\mathcal{E}| \tag{4}$$

2.4 Sparse-to-sparse training

In the last years, many works have started to study also the efficiency of the training phase by training end-to-end sparse NNs[1]. The concept recently has started to be known as *sparse training*.

Static sparse connectivity (Fig. 1d) In 2016, Mocanu et al. [31] have introduced the idea of training directly from scratch (without pruning from a dense networks) sparse Restricted Boltzmann Machines (named XBMs) by using static sparsity. The concept has been further studied in [20] (i.e. Radix-net) and [2, 3, 39] for other NN types. The connectivity graph is obtained before training with data statistics, network science and graph theory concepts. After that, the sparse NN is trained normally using gradient descent or an alternative method. The number of weight updates can be approximated with:

$$k \frac{\mathcal{T}}{b} |\mathcal{E}'| \quad \text{such that } |\mathcal{E}'| \ll |\mathcal{E}| \tag{5}$$

The main disadvantage of sparse NNs with static sparsity is that the connectivity pattern is designed by hand, and cannot model very well data distribution. This leads, in general, to a relatively weaker performance than their dense counterparts.

---

[1]It worth mentioning that the term "sparse neural networks" appears in arXiv preprints title and abstracts as follows: between 2010 and 2017 just one or two papers per year contain it; 2018 - 9 papers; 2019 - 11 papers; and 2020 - 27 papers. Of course, other terms may refer to the same concept, and by relaxing the search query, we obtain 128 papers on the topic in 2020, in comparison with 76 in 2019.

Dynamic (adaptive) sparse connectivity.
Random regrow (Fig. 1f). To alleviate this problem, Mocanu et al. [30, 32] have introduced in 2017 and 2018, respectively, the key concept of adaptive (or dynamic) sparsity and the Sparse Evolutionary Training (SET) algorithm in order to automatically discover an optimal sparse connectivity pattern during training. On very short, SET starts from a randomly generated sparse connectivity and uses after each training epoch a prune-and-regrow strategy to automatically model the data distribution and to find an optimal sparse connectivity. More exactly, during each prune-and-regrow step, a percentage of the non-important connections (the ones closest to zero) are pruned, and the same amount of pruned connections is added to the resulted sparse network in randomly selected positions. SET reduces quadratically the number of connections, while obtaining better accuracy than the dense counterparts in the case of MultiLayer Perceptron (MLP). Further on, [5] trained very sparse networks by sampling the sparse connectivity based on a Bayesian posterior, while [35] introduced Dynamic Sparse Reparameterization (DSR) to train Convolutional Neural Networks which dynamically adjust the sparsity levels of different layers. Based on SET and DSR, [27] proposed a specialized sparse training method for sparse Recurrent Neural Networks (RNNs), named Selfish RNN, which outperforms its dense counterpart at reasonable sparsity levels (less sparser than in the case of MLPs).

Gradient regrow (Fig. 1e). To enhance a faster convergence, [7] and [10] introduced the idea of using momentum and gradient information (quantified in two methods named, Sparse Momentum and RigL, respectively) from non-existing connections during the regrow steps. Evci et al. [10] show that if they trained with RigL sparse CNNs for a long enough time, they can reach the performance of the dense counterparts. Nevertheless, using information from non-existing connections is not scalable, and [17] improves RigL by considering just the gradients of a subset of the non-existing connections in Top-KAST. For both, random and gradient regrow, the number of weight updates can be approximated with:

$$k \; \frac{\mathcal{T}}{b} |\mathcal{E}'_k| \quad \text{such that } |\mathcal{E}'_k| = |\mathcal{E}'_{k-1}| \propto |\mathcal{N}| \text{ and } \mathcal{E}'_{k-1} \neq \mathcal{E}'_k \quad (6)$$

3 SPARSE TRAINING AGENTS

While sparse training seems a promising research direction (as reflected also by Equations 1-6), there are many open questions waiting to be answered and challenges which have to be solved in order to obtain real scalable NN solutions to be further used in any form of intelligent agent. We group them in two main categories: hardware and software support, and theoretical open questions.

3.1 Hardware and software support

Due to the relatively small number of parameters in comparison with the dense models, theoretically, sparse training allows a jump with few orders of magnitude in the representational power of NNs, while having better generalization capabilities than the dense counterparts [28, 37, 40]. These have the potential of making sparse training the *de facto* approach for NNs in the future. The main obstacle in making this jump is that most deep learning specialized hardware is optimized for dense matrix operations, and it practically ignores sparse matrix operations. Due to this reason, almost all research done in sparse NNs uses a binary mask over weights to simulate sparsity. In terms of hardware, [46, 47] have started

paving the ground for sparsity proper hardware support. Also, it worth mentioning NVIDIA A100, released in May 2020, which supports 2:4 sparsity (i.e. 50% fixed sparsity level) [50]. These suggest that more hardware support will be developed for sparse NNs in the medium-term future. While waiting for new hardware with proper support for sparse NNs to be conceived, e.g. [8], for both edge and cloud, the problem can be addressed as a programming and software engineering challenge. Indeed, mainstream libraries are not optimized for sparse matrix multiplication, but taking an alternative approach and looking to the basic programming and data structure principles, we can devise new parallel algorithms and implementations for CPUs and GPUs to enable truly sparse NNs. Up to our knowledge, there are very few works on this topic.

For training efficiency, Liu et al. [26] have been able to build truly sparse MLPs with over one million neurons on a typical laptop using only basic Python libraries. In [4], a truly sparse denoising autoencoder is proposed in order to perform fast and robust feature selection. Interestingly, when the number of neurons is sufficiently large, this sparse autoencoder running just on one CPU core has a faster training time than its dense counterparts running on GPU, while being able to perform better feature selection and to have a much more environmentally friendly energy consumption. For inference efficiency, it worth mentioning here the recent MIT/IEEE/Amazon Sparse Deep Neural Network Challenge [19], for which the authors of [34] have been able to perform inference (no training at all) with 125 million neurons in random networks. Nevertheless, all of the above can be reduced to two practical unanswered questions:

- How much we can improve the *training and inference execution time* using sparse NNs instead of dense NNs?
- Where is the trade-off between speed and performance in the case of sparse neural networks?

3.2 Theoretical open questions

3.2.1 Understanding Sparse Training at Scale. Up to now, no one had the opportunity to study any form of NN with billions of neurons (to model, for instance, large realistic worlds), partly also because the algorithmic novelty is too much driven nowadays by the hardware limitations [16] and the ease of using almost production-ready deep learning solutions. In [28], it has been demonstrated that a vast number of entirely different sparse topologies, unveiled easily using dynamic sparsity, reach a very similar performance level after training. Nevertheless, the effect of sparsity on various NN models is non-proportional and non-derivable directly from their dense counterparts. It seems that there is no simple unified way of treating sparsity similarly in MLPs, RNNs, CNNs, and so on. Altogether, this leads to several questions:

- To what extend sparse training applied to sparse NNs with billions of neurons will resemble its learning behaviour observed currently in networks with thousands of neurons?
- How can we combine multiple sparse NNs into one for parallel sparse training algorithms or federated learning [51]?
- Dynamic sparsity naturally can help with regularization, but what makes it to lead to better generalization as empirically shown, for instance, in [37]? How is this even possible as, by design, sparse NNs do not benefit of overparameterization?
- To what extend the capacity of sparse training to outperform dense NNs is sensitive to the NN model type and settings?

- Is dynamic sparsity in sparse training always converging if we consider that it addresses a combinatorial and a continuous optimization problem simultaneously?
- Can we formulate in a trustable and understandable format the relations between the output and the input data in a sparse neural networks using its sparse graph properties?

3.2.2 Sparse Training and Reinforcement Learning. At the core of agents' world is reinforcement learning. Up to now, sparse training has been developed and studied just in the context of supervised and unsupervised learning. To enhance Deep Reinforcement Learning (DRL) algorithms with sparse training, some fundamental challenges have to be solved:

- Can we create sparse DRL models, where the NNs used as function approximators are NNs with dynamic sparsity?
- To what extend the above-created models will be successful as their adaptive connectivity would have to cope also with the dynamicity and uncertainty of the DRL environments?
- Are there any theoretical guarantees regarding the convergence of such sparse DRL models?

3.2.3 Realistic Continual Learning at Scale. Currently, continual learning, even if it is considered a very important topic by the research community, it is immature. For instance, we do not have good enough algorithms to outperform in continual learning settings single task learners. With increased representational power, sparse networks represent a natural candidate in implementing and improving continual learning [41, 44]. In order to have autonomous intelligent agents continuously learning, next question arises:

- To what extend sparse training can help in learning a long series of tasks without suffering from catastrophic forgetting and without storing all historical data?

4 REAL-WORLD COMPLEX SYSTEMS

Neural Networks applications at Scale. Billions of neurons will open many new possibilities of exploiting NNs, well beyond what we know and imagine today. The natural question arising is: *What can we do with them?* For instance, [26] showed the benefit of sparse training for microarray data and its potential of replacing the traditional two steps approach (feature selection, followed by classification). Thus, we would have to rethink the utility of such large networks ahead of time.

Multi-agent systems. Imagine, for instance, myriads of autonomous and intelligent agents perceiving, cooperating, and acting in a realistic environment to fulfill some common goals (e.g. safe and reliable autonomous driving). Many of these heterogeneous agents would have by design and hardware constraints limited capabilities, but in order to properly model and interact among them and with the environment some of them would need to have extremely large representational and generalization capacity.

Autonomous NNs in Low-Resource Devices. At the other end of the spectrum, by using sparsity, we can devise NNs which can be entirely independent of the cloud and can be trained directly on low-resource devices. Such a scenario can be useful for many reasons, for instance, data privacy, energy-saving, harsh, remote environments (e.g. IoT sensors in the ocean), communication overheads or failures in autonomous driving, and so on.

The Smart Grid case study. Large and complex systems, such as electricity grid, relies on the multi-agents learning capabilities
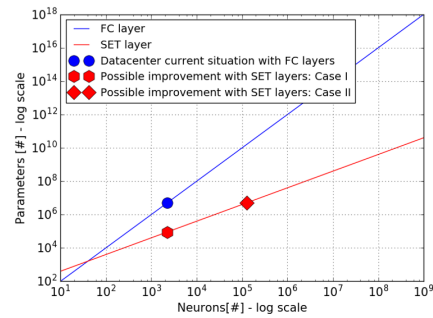

Figure 2: Sparse NN hypothetical improvement for [18].

to act in an accurate, fast, cheap, reliable and trustworthy manner helping the transition towards a more human-centred system at the local level. Still, the multi-agents coordination is needed at all levels [21]. Specialized agents, using DRL have been proved to have superior learning capabilities. For example, the building resource allocation problem typically requests two steps: a) building energy prediction, and b) building energy optimization. In [33] it has been shown that it is possible to use DRL to replace both, the prediction an optimization steps, with an on-line adaptive solution. In [49], a detailed overview of the DRL application in the energy system highlights the huge interest, as well as a general recommendation to overcome the scalability open problems. Even if in 2020, an edge computing hardware prototype [13] for coordination of prosumer agents appeared, the most advanced multi-agent based real-time energy infrastructure [38] and specialized agents [14, 33] are relying on cloud computing and do not scale [6, 49]. We argue in the favour of a further investigation of sparse DRL in multi-agent systems with respect to different types of strategies as a way to infuse more scalable AI technology into the energy field and help the ongoing transition towards a more sustainable society.

On the other side, it is good to highlight the tremendous energy costs induced by the deep learning approaches. For example, Fig. 2 presents a typical dense MLP model from the Google Datacenter (representing 61% of the load) [18] and hypothesizes how a sparse MLP may improve the performance in two cases: (I) much faster training and inference time, lower memory, cost reduction, and probably better accuracy; (II) much higher representational power.

At the same time, we shall not forget that if we want to solve problems such as climate change [45], we shall not look to AI just as to a problem solver, but also to try avoiding making AI a burden of the energy system. Thus, cost-efficient and environmental friendly AI models shall be an improvement goal in itself, considering *sparse AI for energy and energy for sparse AI*.

5 DISCUSSION

All the directions and questions presented in the paper may have a bit of a spaghetti effect at a quick look. Although, probably this is the only solution to obtain intelligent agents and heterogeneous systems of agents which can interact among them in order to solve vital societal challenges and to contribute to the advancement of artificial intelligence. We believe that a way to move forward is to study these challenges as a whole within the research community. Consequently, instead of a concluding phrase, by listing an arguable quote from Metaphysics of Aristotle: *The whole is more than the sum of its parts*, we invite researchers to brainstorm with us.

REFERENCES

[1] 2019. AAMAS 2019 tutorial, Scalable deep learning: from theory to practice. https://sites.google.com/view/scalable-deep-learning.

[2] Kale ab Tessera, Sara Hooker, and Benjamin Rosman. 2021. Keep the Gradients Flowing: Using Gradient Flow to Study Sparse Network Optimization. arXiv:2102.01670 [cs.LG]

[3] Nir Ailon, Omer Leibovich, and Vineet Nair. 2020. Sparse Linear Networks with a Fixed Butterfly Structure: Theory and Practice. arXiv:2007.08864 [cs.LG]

[4] Zahra Atashgahi, Ghada Sokar, Tim van der Lee, Elena Mocanu, Decebal Constantin Mocanu, Raymond Veldhuis, and Mykola Pechenizkiy. 2020. Quick and Robust Feature Selection: the Strength of Energy-efficient Sparse Training for Autoencoders. arXiv:2012.00560 [cs.LG]

[5] Guillaume Bellec, David Kappel, Wolfgang Maass, and Robert Legenstein. 2018. Deep Rewiring: Training very sparse deep networks. In *International Conference on Learning Representations*.

[6] Mathijs M. de Weerdt, Michael Albert, Vincent Conitzer, and Koos van der Linden. 2018. Complexity of Scheduling Charging in the Smart Grid. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems* (Stockholm, Sweden) *(AAMAS '18)*. 3.

[7] Tim Dettmers and Luke Zettlemoyer. 2019. Sparse networks from scratch: Faster training without losing performance. *arXiv preprint arXiv:1907.04840* (2019).

[8] S. Dey, K. Huang, P. A. Beerel, and K. M. Chugg. 2019. Pre-Defined Sparse Neural Networks With Hardware Acceleration. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* 9, 2 (2019), 332–345.

[9] Virginia Dignum and Frank Dignum. 2020. Agents Are Dead. Long Live Agents! *(AAMAS '20)*. 1701–1705.

[10] Utku Evci, Trevor Gale, Jacob Menick, Pablo Samuel Castro, and Erich Elsen. 2020. Rigging the Lottery: Making All Tickets Winners. In *International Conference on Machine Learning*.

[11] Utku Evci, Yani A. Ioannou, Cem Keskin, and Yann Dauphin. 2020. Gradient Flow in Sparse Neural Networks and How Lottery Tickets Win. arXiv:2010.03533 [cs.LG]

[12] Jonathan Frankle and Michael Carbin. 2019. The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks. In *International Conference on Learning Representations*.

[13] Daniel Gebbran, Gregor Verbič, Archie C. Chapman, and Sleiman Mhanna. 2020. Coordination of Prosumer Agents via Distributed Optimal Power Flow: An Edge Computing Hardware Prototype. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems* (Auckland, New Zealand) *(AAMAS '20)*. 2104–2106.

[14] Golnaz Habibi, Zachary Kingston, Zijian Wang, Mac Schwager, and James McLurkin. 2015. Pipelined Consensus for Global State Estimation in Multi-Agent Systems *(AAMAS '15)*. 1315–1323.

[15] Song Han, Jeff Pool, John Tran, and William Dally. 2015. Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems*. 1135–1143.

[16] Sara Hooker. 2020. The Hardware Lottery. arXiv:2009.06489 [cs.CY]

[17] Siddhant Jayakumar, Razvan Pascanu, Jack Rae, Simon Osindero, and Erich Elsen. 2020. Top-KAST: Top-K Always Sparse Training. In *Advances In Neural Information Processing Systems*. 1379–1387.

[18] Norman P Jouppi, Cliff Young, Nishant Patil, David Patterson, Gaurav Agrawal, Raminder Bajwa, Sarah Bates, Suresh Bhatia, Nan Boden, Al Borchers, et al. 2017. In-datacenter performance analysis of a tensor processing unit. In *Computer Architecture (ISCA), 2017 ACM/IEEE 44th Annual International Symposium on*.

[19] Jeremy Kepner, Simon Alford, Vijay Gadepally, Michael Jones, Lauren Milechin, Ryan Robinett, and Sid Samsi. 2019. Sparse Deep Neural Network Graph Challenge. *2019 IEEE High Performance Extreme Computing Conference (HPEC)* (2019).

[20] Jeremy Kepner and Ryan Robinett. 2019. Radix-net: Structured sparse matrices for deep neural networks. In *2019 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*. IEEE, 268–274.

[21] Koen Kok. 2010. Multi-Agent Coordination in the Electricity Grid, from Concept towards Market Introduction *(AAMAS '10)*. 1681–1688.

[22] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep Learning. *Nature* 521, 7553 (2015), 436–444.

[23] Yann LeCun, John S Denker, and Sara A Solla. 1990. Optimal brain damage. In *Advances in neural information processing systems*. 598–605.

[24] Namhoon Lee, Thalaiyasingam Ajanthan, and Philip Torr. 2019. SNIP: single-shot network pruning based on connection sensitivity. In *International Conference on Learning Representations*.

[25] Junjie Liu, Zhe XU, Runbin SHI, Ray C. C. Cheung, and Hayden K.H. So. 2020. Dynamic Sparse Training: Find Efficient Sparse Network From Scratch With Trainable Masked Layers. In *International Conference on Learning Representations*.

[26] Shiwei Liu, Decebal Constantin Mocanu, Amarsagar Reddy Ramapuram Matavalam, Yulong Pei, and Mykola Pechenizkiy. 2020. Sparse evolutionary deep learning with over one million artificial neurons on commodity hardware. *Neural Computing and Applications* (2020), 1–16.

[27] Shiwei Liu, Decebal Constantin Mocanu, Yulong Pei, and Mykola Pechenizkiy. 2021. Selfish Sparse RNN Training. arXiv:2101.09048 [cs.LG]

[28] Shiwei Liu, Tim van der Lee, Anil Yaman, Zahra Atashgahi, Davide Ferraro, Ghada Sokar, Mykola Pechenizkiy, and Decebal Constantin Mocanu. 2020. Topological Insights into Sparse Neural Networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*.

[29] Christos Louizos, Max Welling, and Diederik P Kingma. 2018. Learning Sparse Neural Networks through L_0 Regularization. In *International Conference on Learning Representations*.

[30] Decebal Constantin Mocanu. 2017. *Network computations in artificial intelligence*. Ph.D. Dissertation. Eindhoven University of Technology. https://pure.tue.nl/ws/portalfiles/portal/69949254

[31] Decebal Constantin Mocanu, Elena Mocanu, Phuong H. Nguyen, Madeleine Gibescu, and Antonio Liotta. 2016. A topological insight into restricted Boltzmann machines. *Machine Learning* 104, 2 (01 Sep 2016), 243–270.

[32] Decebal Constantin Mocanu, Elena Mocanu, Peter Stone, Phuong H Nguyen, Madeleine Gibescu, and Antonio Liotta. 2018. Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science. *Nature Communications* 9, 1 (2018), 2383.

[33] E. Mocanu, D. C. Mocanu, P. H. Nguyen, A. Liotta, M. E. Webber, M. Gibescu, and J. G. Slootweg. 2019. On-Line Building Energy Optimization Using Deep Reinforcement Learning. *IEEE Transactions on Smart Grid* 10, 4 (2019), 3698–3708.

[34] Yousuf Ahmad Mohammad Hasanzadeh Mofrad, Rami Melhem and Mohammad Hammoud. 2021. Accelerating Distributed Inference of Sparse Deep Neural Networks via Mitigating the Straggler Effect. In *In proceedings of IEEE High Performance Extreme Computing (HPEC), Waltham, MA USA*. under review.

[35] Hesham Mostafa and Xin Wang. 2019. Parameter efficient training of deep convolutional neural networks by dynamic sparse reparameterization. In *International Conference on Machine Learning*.

[36] Michael C. Mozer and Paul Smolensky. 1989. Using Relevance to Reduce Network Size Automatically. *Connection Science* 1, 1 (1989), 3–16.

[37] Joshua C. Peterson, David Bourgin, Daniel Reichman, Thomas L. Griffiths, and Stuart J. Russell. 2019. Cognitive model priors for predicting human decisions. In *Proceedings of the 36th International Conference on Machine Learning*.

[38] Tiago Pinto, Luis Gomes, Pedro Faria, Filipe Sousa, and Zita Vale. 2020. MARTINE: Multi-Agent Based Real-Time INfrastructure for Energy. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems* (Auckland, New Zealand) *(AAMAS '20)*. 2114–2116.

[39] Ameya Prabhu, Girish Varma, and Anoop Namboodiri. 2018. Deep expander networks: Efficient deep networks from graph theory. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 20–35.

[40] Gobinda Saha and Kaushik Roy. 2021. Gradient Projection Memory for Continual Learning. In *International Conference on Learning Representations*.

[41] Ghada Sokar, Decebal Constantin Mocanu, and Mykola Pechenizkiy. 2020. SpaceNet: Make Free Space For Continual Learning. arXiv:2007.07617 [cs.LG]

[42] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, Vol. 30.

[43] Chaoqi Wang, Guodong Zhang, and Roger Grosse. 2020. Picking Winning Tickets Before Training by Preserving Gradient Flow. In *International Conference on Learning Representations*.

[44] Mitchell Wortsman, Vivek Ramanujan, Rosanne Liu, Aniruddha Kembhavi, Mohammad Rastegari, Jason Yosinski, and Ali Farhadi. 2020. Supermasks in Superposition. arXiv:2006.14769 [cs.LG]

[45] Vahid Yazdanpanah, Sara Mehryar, Nicholas R. Jennings, Swenja Surminski, Martin J. Siegert, and Jos van Hillegersberg. 2020. Multiagent Climate Change Research. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems* (Auckland, New Zealand) *(AAMAS '20)*. 1726–1731.

[46] X. Zeng, T. Zhi, X. Zhou, Z. Du, Q. Guo, S. Liu, B. Wang, Y. Wen, C. Wang, X. Zhou, L. Li, T. Chen, N. Sun, and Y. Chen. 2020. Addressing Irregularity in Sparse Neural Networks Through a Cooperative Software/Hardware Approach. *IEEE Trans. Comput.* 69, 7 (2020), 968–985.

[47] S. Zhang, Z. Du, L. Zhang, H. Lan, S. Liu, L. Li, Q. Guo, T. Chen, and Y. Chen. 2016. Cambricon-X: An accelerator for sparse neural networks. In *2016 49th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. 1–12.

[48] Shunshi Zhang and Bradly C. Stadie. 2020. One-Shot Pruning of Recurrent Neural Networks by Jacobian Spectrum Evaluation. In *International Conference on Learning Representations*.

[49] Z. Zhang, D. Zhang, and R. C. Qiu. 2020. Deep reinforcement learning for power system applications: An overview. *CSEE Journal of Power and Energy Systems* 6, 1 (2020), 213–225.

[50] Aojun Zhou, Yukun Ma, Junnan Zhu, Jianbo Liu, Zhijie Zhang, Kun Yuan, Wenxiu Sun, and Hongsheng Li. 2021. Learning N:M Fine-grained Structured Sparse Neural Networks From Scratch. In *International Conference on Learning Representations*.

[51] H. Zhu and Y. Jin. 2020. Multi-Objective Evolutionary Federated Learning. *IEEE Transactions on Neural Networks and Learning Systems* 31, 4 (2020), 1310–1322.