Towards an Optimal Initial Sparsity Distribution for Automatic Noise Filtering in Deep Reinforcement Learning

ECATERINA DRAGOMIR, University of Twente, The Netherlands

Efficiently learning how to select relevant input features remains a daunting task in the development of artificial intelligence models, especially given high noise levels (80% or more). However, when autonomous systems tauntingly spring into our daily, noisy lives, the importance of automatically and accurately detecting relevant features cannot be ignored. Considerable progress in this direction has been made in the last few years, by using partially sparse (i.e. the output layer is dense) Deep Reinforcement Learning models, such as Automatic Noise Filtering (ANF) [18], which can outperform previously existing networks in environments augmented with up to 98% Gaussian noise. This discovery begs the question of what an optimal sparsity distribution looks like, given a noisy medium. Thus, the present research aims to answer this question, by analysing the effects that various sparsity distributions - inspired from fields such as artificial intelligence, cognitive neuroscience and mathematics - have on the learning efficacy and speed of ANF [18] in highly noisy environments. The results validate the decision to maintain a dense output layer, as well as support the proposal of a series of novel sparsity distributions, among which an inverse Erdős–Rényi model.

Additional Key Words and Phrases: deep reinforcement learning, sparse training, optimal sparsity distribution, noise filtering

1 INTRODUCTION

The clacks of a mechanical keyboard, the clicks of the mouse, whispers hitting walls, then reflecting off every corner of the room, and the wailing sounds of car honks and pedestrians somewhere in the distance. This is the environment that even the quietest library halls have to offer nowadays and the picture doesn't even begin to depict the dozens of movements, colours, lights and smells that our brains are bombarded with every living second.

Although humans have learnt how to efficiently process this colossal amount of daily collected data and filter relevant information from noise (i.e. knowledge that is not significant for a given task), Artificial Intelligence (AI) requires increasingly large, computation power (and thus energy) hungry models, in order to quickly and reliably cope with the continuous changes of the environment. Even Deep Reinforcement Learning (DRL) - an exceptionally advanced, adaptive machine learning paradigm, with SAC [19] and TD3 [12] being prime examples - struggles to accommodate highly imperfect input, which motivated the creation of state-of-the-art models that strive to balance resource-awareness with reliable performance.

Excellent such agents can be obtained through the use of dynamic sparse training on several DRL algorithms (including SAC [19], PPO [35] and DQN [29]), by reducing their connections to as little as 10% of the original number [17]. For this purpose, dynamic sparse training follows three steps: (1) sparsely initialise the model, (2) remove the least important connections and (3) grow an equal amount of links. Its application on DRL networks was introduced by Sokar et al. [36], who managed to successfully equate the performance of dense agents, while decreasing both their parameter count and their training time by 50%.

Only one year later, in 2023, Grooten et al. [18] proposed a model for Automatic Noise Filtering (ANF), that applies the dynamic sparse training to DRL algorithms to yield astonishing results in noisy environments. Their network employs a partially sparse architecture to outperform both its fully sparse and fully dense analogues [18], which begs the question of the influence that partially sparse designs might have on the efficiency of Artificial Neural Networks (ANN). Building upon the idea, the present thesis aims to grasp the role of a sparse initialisation in extremely sparse, noisy mediums, by answering the research questions outlined below.

RQ 1 How can theories from related fields (e.g. neuroscience, mathematics) be used to improve the performance of ANF in noisy (i.e. 80% or more Gaussian noise) environments, given high (i.e. 80% or over) sparsity levels?
  RQ 1.1 Which theories are the most suitable for this purpose?
  RQ 1.2 How can the identified theories be applied to ANF?
RQ 2 What impact does the application of the selected mathematical model(s) have on the performance (i.e. learning efficacy and speed) of the network?
  RQ 2.1 How does the impact of the selected sparsity distribution models differ across the alternatives?

Briefly, the main **contributions** are: (1) studying how four global sparsity levels and ten initial distributions affect the learning efficacy and speed of ANF in noisy mediums; (2) contrasting the influence of global and local sparsity to validate the partially sparse design of ANF [18] and show that its performance can be further increased; (3) proposing novel sparsity distributions (some of which appear to outperform existing standards), such as an Inverse Erdős–Rényi initialisation which is suitable for high noise levels.

The paper shall thoroughly describe the conducted experiments and consequent conclusions, without assuming any prior knowledge of the reader. Therefore, the next sections form a cohesive explanation of essential background information such as basics of probabilities, graph theory and ANN models (Section 2), then related advancements in the domain of sparse training for reinforcement learning (Section 3), the selected setup and algorithm for performing the experiments (Section 4) and finally, the results (Section 5) on which the answers to the research questions and recommendations for future work have been based (Section 6).

2 BACKGROUND

Before delving into an explanation of the model training and the experiments, it is worth understanding the underlying theory of artificial neural networks (ANN), as well as a range of basic information from fields that artificial intelligence branched from at its dawn.

TScIT 39, July 7, 2023, Enschede, The Netherlands

Consequently, the present section starts with explaining where the architecture of ANNs draws inspiration from, then proceeds to give an overview of useful probability and graph theory concepts and, finally, describes the creation and training of ANNs.

2.1 Cognitive neuroscience

With its name coined at the end of the 1970s, cognitive neuroscience deals with explaining the connections between the human brain and the mind [16], including the emergence of intelligence through the acquisition, storage and use of knowledge. Allured by a firm belief that replicating the base elements of the human brain in a digital environment could lead to the appearance of artificially intelligent systems, many used to draw inspiration from this new field when designing AI. Moreover, the view that the structure of the brain can be a reliable development model (but arguably not an earmark) for ANNs [20], proved its verity in the past few years.

One area that has intensely served as an inspiration for performant autonomous systems has been research into the brain connectivity. While some knowledge about the workings of this organ has been accepted long ago by the academia (e.g. the fact that "the nervous system is made up of individual cells" [16], namely neurons, which communicate using electrical and chemical signals [26]), mapping the structure of the brain (i.e. the human connectome) is a rather novel field, which receives increasing attention [22]. Moreover, an interesting discovery has been the small-world property of the brain, which can be explained as follows: on an anatomical level, neurons are grouped in clusters which sparsely connect to each other [22]; this knowledge has inspired state-of-the-art training methods, as explained within Subsection 2.3.

2.2 Mathematics

2.2.1 Probability theory. Our brains represent the outside world by means of probabilistic distributions and can detect such information in data, since the youngest ages [24], [3], [10]. Additionally, probability theory found applications in a variety of fields, such as graph theory and artificial intelligence, for which a basic understanding of concepts like the probability distribution and expected value of random variables (as explained below) are highly relevant.

First, taking an isolated event A, we can assign to it: a chance (i.e. probability) of happening, which is denoted as $P(A)$ and can take a value between 0 and 1, where 0 denotes an impossible event and 1 - a certain event [27]; a set of all possible outcomes of this event, usually denoted S.

Then, proceeding from set S, to each element of the series the following can be assigned: a numerical value, via a function known as a *random variable* (usually denoted X); a probability of happening (i.e. the probability function of X), represented as $P(X = x)$, where x is any value that X can take.

Lastly, given a random variable X, we consider: its **(probability) distribution** is a cohesive way of representing the probability of X for each value that it can take, as $P(X \in B) = \sum_{x \in B} P(X = x)$; its **expected value** (i.e. the mean of the distribution) could be interpreted as the weighted average of the probabilities of each value that X can take [27] and it is formally defined as $\mathbb{E}(X) = \sum xP(X = x)$.

A relevant example for the presented theory would be studying the direction of movement of a rook in a game of chess: we can define $S = \{up, down, left, right\}$, X = *"the direction of the rook"* where X = 0 when the rook moves up, X = 1 when the rook moves down and so on, then assign a probability $P(X = x) = \frac{1}{4}, \forall x \in \{0, 1, 2, 3\}$. Moreover, given this information, we could calculate the expected value of moving the rook, $\mathbb{E}(X) = \frac{1}{4}$, and the homogeneous distribution of X, $P(X = 0) = P(X = 1) = P(X = 2) = P(X = 3) = \frac{1}{4}$. However, depending on the probability function of X, several distributions can be defined and visually represented (e.g. Figure 1 [27]).

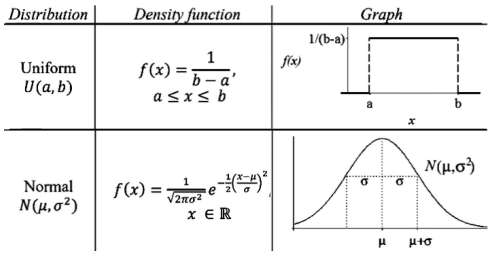

Fig. 1. Probability distribution functions [27]

2.2.2 Graph theory. From street guidance to computer networks, graph theory has found one of its more interesting applications in mapping the connections of the brain, in particular its functional and effective structures [9]. While the set of methods is vast, valuable insight can be gained from the application of any graph (defined as "a finite set of vertices (or nodes) that are connected by links called edges (or arcs)" [9]).

Graph theory can be used to represent the architecture of artificial neural networks such that each arc can be fully defined by the neurons that it links and by a probability of existence. These concepts are also employed in random graphs (i.e. graphs in which all arcs have an equal probability of appearance), such as the Erdős–Rényi model for the generation of random graphs [6] and the Erdős–Rényi model of random graphs evolution [7].

2.3 Artificial neural networks

The creation of intelligent agents (i.e. any system that is able to perceive its environment through sensors and act upon it via actuators [33]), broadly referred to as artificial intelligence (AI), has been a subject of interest in the scientific community for almost a century, since McCulloch and Pitts [26] published what is now considered to be the first work in the field [33]. The present research is concerned with machine learning, which is the study of how machines learn by observing information (i.e. input data), creating a representational model of it and then inferring accurate knowledge (i.e. output data) based on the model. The computer would then receive feedback from its environment and aim to correct its model in order to minimise representational and thus, output errors [33].

Depending on the type of input data, model, training and feedback that an agent deals with, multiple areas of study can be differentiated. A relevant example are artificial neural networks (ANN) which are representational models vaguely resembling the structure of the brain [33]. An existent connection allows the flow of information

between two neurons, such that nodes on the first (i.e. input) layer receive information and pass it for processing to the middle (i.e. hidden) and last (i.e. output) layers.

2.3.1 Reinforcement learning. Depending on the feedback that the agent receives, three learning paradigms have been defined (i.e. supervised, unsupervised and reinforcement learning) and ANNs found applications in all of them. In reinforcement learning, agents learn by interacting with their environment, thus inferring one behaviour which is periodically reinforced or punished through *rewards* [33]. This can be formally defined as a Markov Decision Process, a tuple that contains all the relevant setting information: $\langle \mathscr{X}, \mathscr{A}, \mathscr{P}, \mathscr{R}, \gamma \rangle$, where $\mathscr{X}$ is the state space (i.e. the set of all possible states of the agent); $\mathscr{A}$ is the action space (i.e. all possible actions that the agent can take); $\mathscr{P}$ - the transition dynamics (i.e. a function that maps a combination of the current state s and an action a taken by the agent, to the transition probability of the agent to find itself, in the consequent moment, in a state s'); $\mathscr{R}$ is the reward function ; $\gamma \in [0, 1)$ - the discount factor (i.e. a parameter which controls the importance of new information in the learning experience of the agent) [17].

Although hundreds of RL algorithms have been devised, they all learn a desirable behaviour, which is stored as policy $\pi : \mathscr{X} \rightarrow \delta(\mathscr{A})$ (i.e. a function that maps each possible state of the agent to a probability distribution of its possible actions). This policy also entails learning a value function $V^{\pi} : \mathscr{X} \rightarrow \mathbb{R}$ (i.e. the expected value of acting according to the policy, when starting from current state s) and, possibly, a value-action function $Q^{\pi} : \mathscr{X} \times \mathscr{X} \rightarrow \mathbb{R}$ (i.e. the expected value of acting according to the policy, when starting from current state s **and** picking action a) [17].

Depending on how the value and action-value functions are stored, Russell and Norvig [33] classifies RL algorithms as follows.

- **action-utility learning**: the agent stores V^{π} and Q^{π} directly in a table, which becomes rather "impractical" for large $\mathscr{X}$ [17];
- **policy search**: it is usually applied by DRL algorithms, which learn an approximation of the (action-) value function (instead of the actual V^{π} or Q^{π}) and store it in one or more deep networks. Depending on which approximators the algorithm stores, Graesser et al. [17] further categorises them into:
  - value-based: used in discrete control environments (i.e. spaces with a finite set of actions), they learn an approximation $Q_{\theta} \approx Q^{\pi}$ of the action-value function, which is then stored in a deep network and used to infer the optimal policy [17];
  - policy-gradient: also known as actor-critic methods, these algorithms are typically used in continuous control environments grace to their ability to learn both an action-value Q_{θ} (i.e. the critic) and a policy π_{ψ} (i.e. the actor) estimator [17].

2.3.2 Deep learning. With application in several domains, including reinforcement learning, deep learning (DL) is a family of techniques that are widely used nowadays due to their baffling success [33]. The name comes from the complex structure of the ANN that DL builds, with several interconnected layers of neurons, such as the one depicted in Figure 2.

2.3.3 Sparse training. One of the most exciting, neuroscience inspired contributions to AI has been sparse training. As described in the previous section, deep neural networks are stacked layers of interconnected neurons. Depending on how well connected the model is, ANNs can be dense (i.e. each neuron is connected to all neurons in the preceding and consecutive layer) or sparse, otherwise.

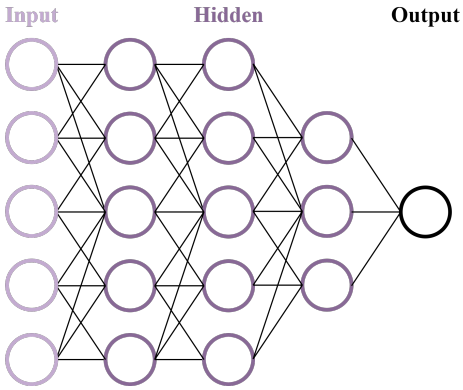

Fig. 2. Deep learning network with 1 input layer, 3 hidden layers and 1 output layer

While the concept of sparse networks has been coined since the late 1980s, generating and training them proved to either be computationally expensive or yield inaccurate results [17]. Two main approaches can be identified in this direction:

- **dense-to-sparse training** gradually prunes (i.e. removes) connections through repeated train-prune-train cycles, in order to reach a sparse network;
- **sparse-to-sparse training** sparsifies the model at initialisation and proceeds to maintain the same sparsity level.

Although previously considered inefficient, recent years have brought novel approaches in the latter training method, thus managing to train sparse ANNs that considerably outperform their dense counterparts. The performance is achieved by efficiently utilising all the stored parameters, consequently driving down the required power consumption [36]. The discussed innovations further split the sparse-to-sparse training techniques into two classes:

- **static** (the sub-optimal solution [17]): the sparsity level is maintained throughout the training because the structure of the ANN remains unchanged;
- **dynamic**: after pruning the network, an update time period is defined, such that the topology (i.e. structure) of the network is changed continuously, by cutting and regrowing a fixed number of neural connections.

3 RELATED WORK

3.1 Sparse training

Proposals of training ANNs in a way that utilises parameters more efficiently start as early as 1989 [17], when Mozer and Smolensky [32] defined a technique called **"skeletonization"**; its aim was, among others, to speed up and better understand the learning process of agents. Since then, several attempts have been made at improving the inference procedure of ANNs. The first to be employed were dense-to-sparse methods, as further presented by Graesser et al. [17],

with a notable contribution in the field being the work of Frankle and Carbin [11]. They proposed **"The Lottery Ticket Hypothesis"** stating that any randomly initialised, dense network can be further pruned and retrained without denting its original accuracy [11].

Despite outstanding results, this manner of training can sometimes increase the energy consumption of the model [17], thus making it an unfavourable alternative to sparse-to-sparse techniques, both static and dynamic. While the rest of this paper will be concerned with the dynamic alternative (whose main variants are presented below), further information about sparse training methods can be found in survey papers like [14], [21] and [30].

Proposed in 2018, **Sparse Evolutionary Training** (SET) is a dynamic sparse training method applied on ANNs [31], possibly the first to train a fully sparse model from the beginning [8]. It initialises the network with an Erdős–Rényi random graph [6], such that for each layer k that has n^k neurons, connections have a probability of appearance of $\frac{\varepsilon(n^k+n^{k-1})}{n^k n^{k-1}}$, where n^{k-1} is the number of neurons on the previous layer. From the formula, one can notice that the more neurons a layer has, the sparser it will be. Then, throughout the training, at predefined update periods ΔT, SET iteratively changes a fraction of the existing connections, such that the arcs with absolute weight values closest to zero are dropped and an equal number of links is regrown at random.

A different, but similar method is **The Rigged Lottery (RigL)** [8]; drawing inspiration from "The Lottery Ticket Hypothesis" [11], it slightly differs from SET because it selects new weights to be grown from those with the highest magnitude, rather than at random.

3.2 Sparsity in Deep Reinforcement Learning

3.2.1 DRL algorithms. Successful implementations of DRL agents can be traced back to their introduction in 2013, when Mnih et al. [28] applied reinforcement learning concepts to deep neural networks, thus creating the value-based, **Deep Q Learning (DQN)** algorithm. Remarkably, this managed to surpass all contemporary agents when tested on Atari games and to nearly reach human-level performance [29]. Following these outstanding results, several policy-gradient actor-critics have been proposed, with the most notable examples lying at the foundation of state-of-the-art models. Depending on the policies that are optimised and respectively, used for decision-making, these can be grouped into: **on-policy**, which employ the same function for both updates and decisions (e.g. Trust Region Policy Optimization (TRPO) [34], later improved by Proximal Policy Optimization (PPO) [35]) and **off-policy**, that employ two separate functions, one for each purpose (e.g. Deep Deterministic Policy Gradient (DDPG) [23] which was then advanced by the Twin Delayed Deep Deterministic Policy Gradient (TD3) [12], Soft Actor-Critic (SAC) [19]).

3.2.2 Sparsity in DRL. Applications of sparse training in DRL have received increasing attention in the past few years [36], as a series of novel, high performance models started emerging. Notably, for agent training, these all employ dynamic sparse-to-sparse methods (broadly inspired by the work of Mocanu et al. [31] on SET), as Graesser et al. [17] proved their superior results on (up to 90%) sparse DRL models (namely DQN [28], PPO [35] and SAC [19]).

The idea was initially suggested in 2022, by Sokar et al. [36], who successfully integrated SET with 50% sparse TD3 [12] and SAC [19] models, into **DS-TD3** and **DS-SAC** - agents that require almost 50% less training steps and computation power to equate the results of the original, dense networks [36].

Within one year from the introduction, Tan et al. [38] proposed the **Rigged Reinforcement Learning Lottery (RLx2)** model, that achieved state-of-the-art results by training both TD3 [12] and SAC [19] entirely on (up to 99%) sparse networks, by using the RigL technique [8].

Finally, in 2023, Grooten et al. [18] presented the **Automatic Noise Filtering (ANF)** agent which extends previous research (particularly that of Sokar et al. [36]), by increasing the network's robustness to noise. For the purpose of testing, Grooten et al. [18] also designed the Extremely Noisy Environment (ENE) and instantiated it in four (MuJoCo Gym [2]) continuous control environments, by augmenting them with a fraction $n_f \in [0, 1)$ of Gaussian [15] noise (i.e. irrelevant features sampled from a random variable with Normal probability distribution [4]). These were used in an extensive study of ANF, which proved its outstanding performance in mediums with $n_f \geq 0.8$ (i.e. over 80%) noise features, where all other agents struggle to learn.

4 SPARSITY DISTRIBUTION ANALYSIS

4.1 Sparsity distributions in ANF

In order to answer the presented research question, the original ANF-SAC [18] algorithm has been used and partially modified to accommodate the selection and creation of various initial sparsity distributions (as presented in Figure 5), in the layers of the actor and critic networks. The algorithm (which will be made available online after its intended submission to an A* conference) has been schematically defined below, in Algorithm 1; the text written in black represents the original algorithm of Grooten et al. [18] and the text written in violet shows the novel components.

As partially seen in Algorithm 1, ANF-SAC [18] has dozens of parameters that can be adjusted at initialisation in order to control the structure and behaviour of the model, which rendered the creation of additional variables futile. While most of them were kept with the default values as set by Grooten et al. [18], a few have been adjusted throughout the training jobs, as listed below.

- **environment**: all experiments have been ran on Walker2d-v3 (one of the continuous control MuJoCo Gym environments [2]).
- **noise**: some study has been performed already by Grooten et al. [18], concerning the effect that the global sparsity level has on the returns of ANF, in ENE environments with $n_f = 0.9$ added noise. In line with their analysis, the base experiments in this research have also been conducted by augmenting the selected medium with 0.9 noise features.
- **sparsity distribution**: selected from {*normal*, *inverse_ER*, *new*, *ER*, *exp_input*, *exp_output*, *sparse_input*, *sparse_output*, *uniform*, *random*}, as shown in Figure 5 and presented in Subsection 4.2.
- **global sparsity**: incrementally set for each distribution, at values $s_g \in \{0.8, 0.9, 0.95, 0.97\}$, as done by Grooten et al. [18].
- **output layer sparsity**: $s_o \in \{sparse, dense\}$.

Algorithm 1: Sparsity distribution analysis in ANF

Data: topology-change period ΔT, drop fraction d_f, initial collect steps b_{init}, train every k env steps, minibatch size n, learning rate λ, target smoothing coefficient τ, max env steps T, global sparsity level s_g, sparsity distribution s_d, output layer sparsity s_o

Initialize the actor network π and two critic networks $Q1$, $Q2$, with weights ϕ, θ_1, θ_2.

Assign sparsity level s_l to each layer according to distribution s_d, so that the average network sparsity == s_g.

if s_o == *dense* **then**

  s_l of the output layer $\leftarrow 0$.

end

Randomly prune the layers in each network from $\{\pi, Q_1, Q_2\}$ to their sparsity level s_l.

Duplicate the two critics to create two target networks, with weights $\overline{\theta}_1 = \theta_1, \overline{\theta}_2 = \theta_2$.

Initialize the replay buffer $\mathscr{B}$ with b_{init} random actions.

for *t=1, t<T* **do**

  Sample action a from the policy (actor network) based on current state s: $a \sim \pi_\phi(\cdot|s)$.

  Take a step in the environment and observe reward r and new state s'.

  Store transition tuple (s, a, r, s') in $\mathscr{B}$.

  if t *mod* k == *0* **then**

    Sample minibatch of n transitions from $\mathscr{B}$.

    Update the weights according to SAC's objective functions J_Q, J_π:

$$\theta_i \leftarrow \theta_i - \lambda \hat{\nabla}_{\theta_i} J_Q(\theta_i) \text{ for } i \in \{1, 2\}$$

$$\phi \leftarrow \phi - \lambda \hat{\nabla}_{\theta_i} J_\pi(\phi)$$

    Update the target networks:

$$\overline{\theta}_i \leftarrow \tau\theta_i + (1-\tau)\overline{\theta}_i \text{ for } i \in \{1, 2\}$$

  end

  if t *mod* ΔT == *0* **then**

    Update the topology of the networks:

      Prune fraction d_f of the smallest magnitude weights. Grow fraction d_f new weights randomly, initialize at value 0. Mask Adam's running avg. of the 1st & 2nd raw moment of the gradient for pruned weights: $m \leftarrow 0$, $v \leftarrow 0$.

  end

end

4.2 Types of sparse initialisation

A selection of inexhaustive initial sparsity distributions have been devised for analysis, by creating associated probability functions. These are shown in Figure 5, where k is the index of the layer, n^k is the number of neurons on layer k, $\varepsilon = 0.3$ as suggested in Sundar and Dwaraknath [37], $\alpha = 0.5$ as proposed in Grooten et al. [18], σ and μ are the variance and mean of the normal distribution. They were chosen such that out of the ten distributions: one (*uniform*) has been studied before, by Grooten et al. [18], in both ANF and Sparser-ANF; two (*new* and *ER*) have been inspired by the work of Grooten et al. [18], but their ε value has been changed; one (*random*), used as benchmark, was previously suggested by Liu et al. [25]. The rest of six are novel and motivated as follows: one (*normal*) - by the (Gaussian [15]) noise distribution employed in tests; four - by existing mathematical (probability) functions, namely Linear [5] (in the case of *sparse_input* and *sparse_output*) and Power law [1] functions (*exp_input* and *exp_output*); one (*inverse_ER*, which is one of the main contributions of this thesis) - by SET [31], as its probability of connection on each layer is $1 - p_{ER}$, where p_{ER} is the probability of connection in SET [31]. This latter proposal sets a higher sparsity for layers with less neurons and its suitability for high noise levels could be explained, in the context of ANF, by the consequent increased sparsity of the input layer, acting as a filter.

4.3 Global versus local sparsity

Another relevant, analysed aspect is the individual influence of the **global** (i.e. the average sparsity level of the entire network) versus **local** (i.e. output layer) sparsity, on the performance of ANF [18]. Therefore, for each of the global sparsity levels presented in Subsection 4.1, the various distributions have been given a specific output layer setting between *sparse* and *dense*, illustrated in Figure 5 by the continuous and respectively, dashed lines. Only three of the distributions have been tested with a *dense* output layer, corresponding to the investigation of Grooten et al. [18].

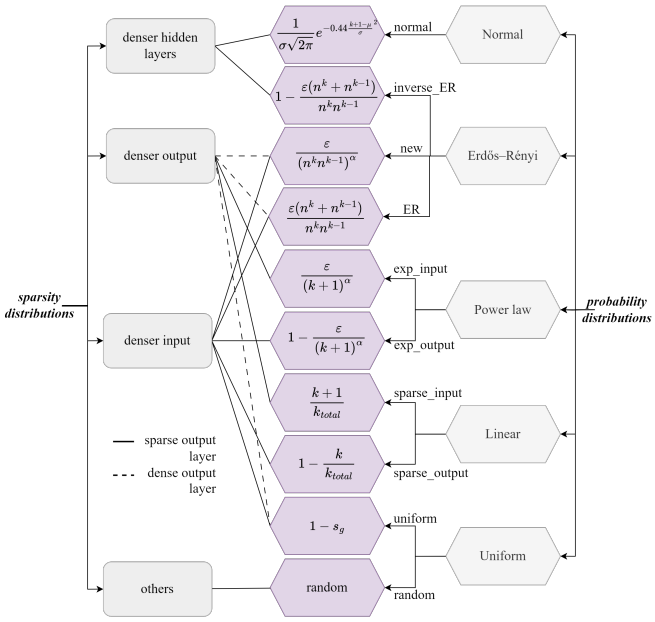

Fig. 5. Tested distribution functions and their algorithms (named as indicated on the arrows)

5 RESULTS

5.1 Evaluation metrics

Treasured as it is nowadays, one can fail to notice that the data itself, without those that can make sense of it, is meaningless; it is rather the manner in which it supports further conclusions that renders it so precious. Consequently, it is important to set clear

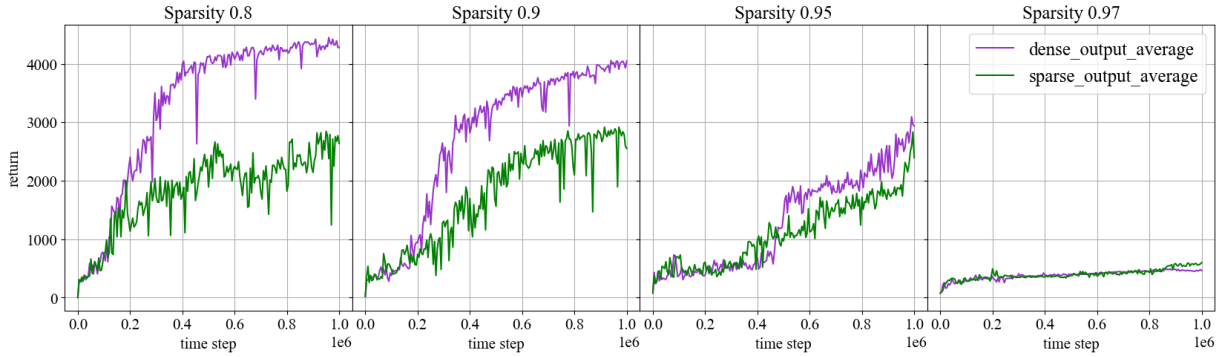

Fig. 3. Comparison of average return of methods *ER*, *new* and *uniform*, when using a dense versus a sparse output layer

variables that are to be analysed, as well as relevant metrics for them, before discussing the results. In the context of the present thesis, two characteristics of the algorithm will be studied. Namely, these are: the learning **efficacy**, depicted by contrasting the plotted returns, and the learning **speed**, measured by using (1) the average ratio between real (i.e. relevant) and fake (i.e. noisy) connections (which aims to depict how accurately the algorithm discerns among features) and (2) the average slope of the real features curve (which shall determine the speed of the model).

These results will be evaluated and presented in the context of three influencing variables: **global sparsity**, **output layer sparsity** and **sparsity distribution**.

5.2 Learning efficacy

5.2.1 Global sparsity. Given the clear indication (i.e. on the graphs) that regardless of the model, the returns decrease considerably as the global sparsity increases past 0.9, it can be concluded that extremely high sparsity levels are detrimental to the performance of DRL networks. However, it is important to note that almost all models can handle up to 0.9 sparsity with minimal loss (Figure 4).

5.2.2 Output layer. Figure 3 plots the average results of distributions *ER*, *new* and *uniform* given a dense output layer against the average results of the same distributions given a sparse output layer. Thus, it can be easily observed that the performance of DRL agents improves by maintaining a **dense setting**, which validates the choice of Grooten et al. [18] to design a partially, rather than fully sparse architecture in their ANF model.

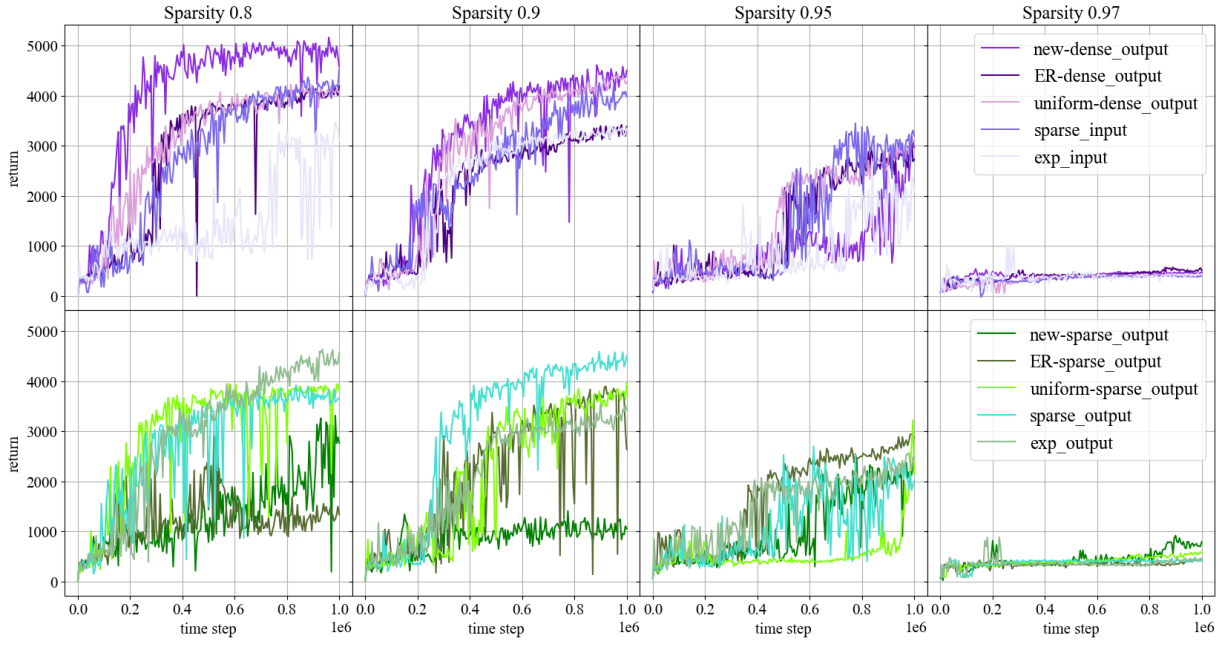

Fig. 4. Comparison of return from sparser input and respectively sparser output distributions

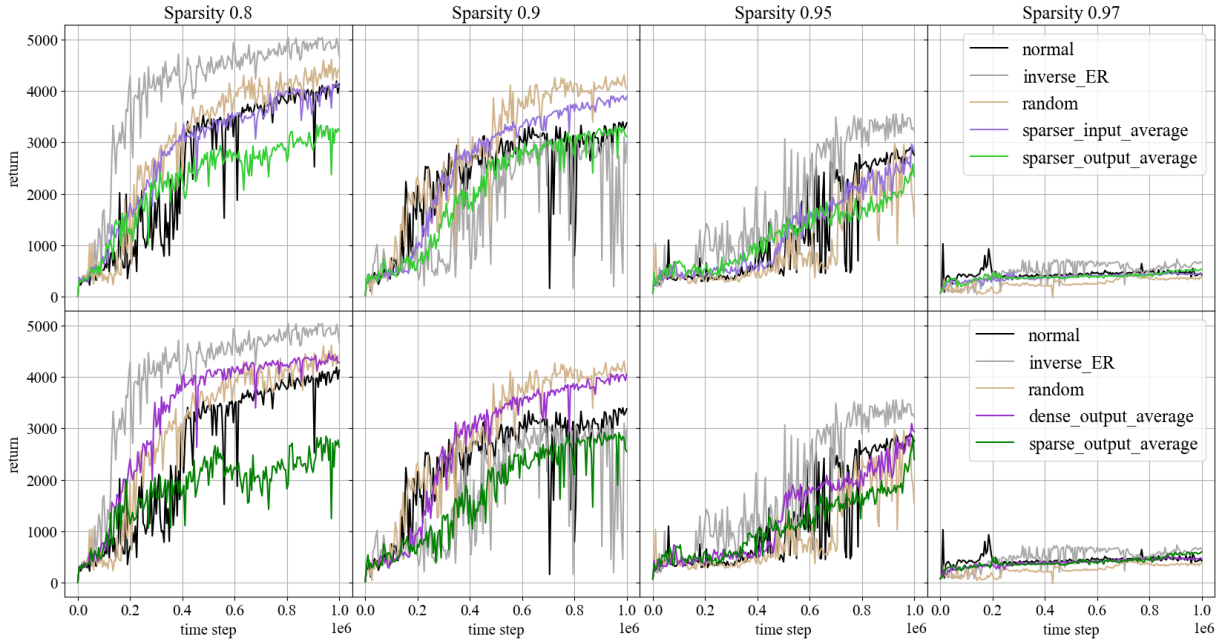

Fig. 6. Comparison of return of methods *normal*, *inverse_ER* and *random* with the average return of sparser input and respectively sparser output distributions

5.2.3 Sparsity distribution. Figure 6 depicts the performance of distributions *normal*, *inverse_ER* and *random*, in contrast with two averages: the first set of subplots shows the average return as estimated from all distributions with a sparser input, respectively all distributions with a sparser output (further outlined in Figure 4, in similar colours); the second set of subplots uses the same averages as shown in Figure 3.

Based on the results, several interesting observations can be made:

- *ER*, *new* and *uniform* perform considerably better than their Linear [5] and Power law [1] function distributed counterparts. This can be deduced from the decrease in returns when the latter two are included in the average (first row of plots in Figure 6);
- *inverse_ER* surpasses all other distributions in returns, in three out of four cases (*inverse_ER* underperforms in 0.9 global sparsity). Moreover, when compared to the results of Grooten et al. [18] in 0.8 and, respectively, 0.95 noise augmented, Walker2d-v3 environment, this distribution applied to ANF-SAC yields over 20% higher returns than distribution *uniform* on ANF-TD3 [18];
- *normal* and *inverse_ER* are both quite robust to increases in sparsity level;
- *random* is unexpectedly efficient, sometimes outperforming all other distributions. Although highly stochastic, this idea was previously suggested within the research of Gadhikar et al. [13] and its importance comes from the following: since a randomly sparse network can yield greater results than all other distributions, it is fair to assume that there must be a more suitable sparse architecture for DRL agents than what has been studied so far.

5.3 Learning speed

5.3.1 Global sparsity. Although not obvious from the graphs discussed in the previous section, an analysis of Table 1 shows that all sparsity distributions seem to have a spike in learning speed (i.e. both connections ratio and learning slope) at 0.9 global sparsity level. This points to the idea that 0.9 sparsity level is the optimum value for accelerating the learning process.

5.3.2 Output layer. Investigating the effects of using a dense output layer setting has only been conducted for distributions *ER*, *new* and *uniform* (as they have been previously tested with a dense output layer by Grooten et al. [18]), but it is still relevant to contrast the results. It can be observed that, in the context of learning velocity, having a dense output layer is also desirable over a sparse one, similarly to the information presented above.

5.3.3 Sparsity distribution. Even though a comparison across distributions is somewhat inconclusive, it is interesting to note that for the Linear [5] and Power law [1] function based models, sparser output distributions appear to learn quicker than the sparser input ones.

6 CONCLUSIONS

Although arduous work still awaits humanity in its pursuit of creating noise-indifferent intelligent entities, considerable advances have been made in this direction lately and, with the present findings, two more (research) questions (laid out in Section 1) have been answered. Concisely, I have observed that concrete information concerning the distribution of anatomical connections in the brain is as sparse as the studied neural networks and that fields such as

Sparsity distribution	Algorithmic method	Comparison parameter	Output layer setting for each global sparsity							
			0.8		0.9		0.95		0.97	
			sparse	dense	sparse	dense	sparse	dense	sparse	dense
Erdős-Rényi	ER	speed (10^{-5})	1.94	2.60	4.29	3.70	2.49	2.71	0.14	0.20
		connections ratio	1.75	1.89	3.42	3.40	5.63	5.66	1.96	2.86
	new	speed (10^{-5})	2.52	3.75	3.63	4.91	2.41	2.42	0.16	0.24
		connections ratio	1.90	2.21	3.31	4.14	5.47	5.39	2.02	3.37
	inverse_ER	speed (10^{-5})	3.44		3.91		2.51		2.78	
		connections ratio	2.28		3.49		6.19		0.16	
Power law	exp_input	speed (10^{-5})	2.06		2.87		2		0.18	
		connections ratio	1.96		3.44		5.19		2.88	
	exp_output	speed (10^{-5})	2.91		4.69		3.11		0.23	
		connections ratio	1.64		3.06		5.07		2.06	
Linear	sparse_input	speed (10^{-5})	2.59		3.54		1.29		0.08	
		connections ratio	2.29		4.88		4.77		1.99	
	sparse_output	speed (10^{-5})	2.90		4.69		2.54		0.12	
		connections ratio	1.95		3.75		5.13		1.90	
Normal	normal	speed (10^{-5})	3.28		4.43		2.59		0.18	
		connections ratio	1.95		3.64		5.15		2.82	
Uniform	uniform	speed (10^{-5})	3.35	2.76	3.98	4.33	2.28	2.26	0.153	0.118
		connections ratio	2.15	2.02	3.55	3.85	5.35	5.48	2.31	2.24
	random	speed (10^{-5})	1.98		4.10		1.88		0.118	
		connections ratio	1.88		4.08		4.89		2.27	

Table 1. Comparison of average learning speed (10^{-5}) and average ratio of real:fake connections in the actor network, given the sparsity distribution ($s_d = algorithmicMethod$), global sparsity ($s_g \in \{0.8, 0.9, 0.95, 0.97\}$) and output layer setting ($s_o \in \{sparse, dense\}$)

probability and graph theory can serve as fruitful inspiration for the architecture of performant deep reinforcement learning models.

Moreover, by contrasting a series of sparsity distributions on ANF-SAC, in the ENE Walker2d-v3 environment [18], the following conclusions have been reached:

- using a dense output layer appears to yield superior results (in both learning efficacy and speed) to employing a sparse one;
- distribution *inverse_ER* shows the highest returns for three out of four global sparsity levels (being closely followed by distribution *new*) and outperforms Sparser-ANF-TD3 [18] in the Walker2d-v3 environment, by over 20%;
- random distributions, although stochastic, sometimes perform better than all other models. Notably, this entails the relevance of a further study of the topic (i.e. in the pursuit of a truly optimal initial sparsity distribution);
- Normal [4] or nearly Normal distributions are surprisingly robust, to the extent where they outperform their counterparts in extremely noisy environments (e.g. 95% noise);
- the learning speed of the studied examples generally peaks at 0.9 sparsity level. However, equally important to note is the fact that returns do not follow the same pattern and rather decrease with the network's density.

6.1 Limitations

For various reasons, the discussed findings bear a series of limitations, as listed within the section. First of all, as previous research in the matter is rather scarce, the selection of sparsity distributions (and their functions' parameters) is inexhaustive and somewhat arbitrary. Second of all, the data validation was considerably constricted by time; thus, another insufficiency are the test setting and its selection: one model, environment and noise level value have been verified, to the degree of obtaining conclusive information.

6.2 Further directions

Work in the field of dynamic sparse training, especially in the context of reinforcement learning, is at an incipient phase and much exploration is left to be done in this direction. Hopefully, the present results will serve as an inspiration for an arguably needed further study of the issue. While the topics of analysis are vast, potential directions include probing other distributions or function parameters, validating the findings on different environments (e.g. Humanoid-v3 ENE, HalfCheetah-v3 ENE [18]), models (e.g. ANF-TD3 [18]) or noise levels and, ultimately, creating a (dynamic) sparsity distribution for optimal results, regardless of the agent's setting.

7 ACKNOWLEDGEMENTS

As the contribution of a few persons has been crucial to the outcome of this present thesis, I would like to thank the following, for their invaluable help: my supervisor, Elena Mocanu, for her extensive feedback and support; the lead creator of ANF [18], Bram Grooten, for his insightful suggestions and enthusiasm; Valentin Dragomir, for his skillful assistance in setting up the training environment.

REFERENCES

[1] Albert-László Barabási. 2016. *Network Science* (first ed.). Cambridge University Press.

[2] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. 2016. OpenAI Gym. (2016). arXiv:1606.01540

[3] Andy Clark. 2013. Whatever next? Predictive brains, situated agents, and the future of cognitive science. *Behavioral and Brain Sciences* 36 (2013). Issue 3.

[4] Abraham de Moivre. 1756. *The Doctrine of Chances: Or, A Method of Calculating the Probabilities of Events in Play.* Chelsea Publishing Company.

[5] René Descartes. 1886. *La géometrié.* A. Hermann, Librairie Scientifique.

[6] P. Erdős and A. Rényi. 1959. On random graphs. I. *Publicationes Mathematicae Debrecen* 6 (1959). Issue 3-4.

[7] Paul L. Erdős and Alfréd Rényi. 1984. On the evolution of random graphs. *Trans. Amer. Math. Soc.* 286 (1984).

[8] Utku Evci, Trevor Gale, Jacob Menick, Pablo Samuel Castro, and Erich Elsen. 2020. Rigging the Lottery: Making All Tickets Winners. In *Proceedings of the 37th International Conference on Machine Learning (ICML'20)*. JMLR.org.

[9] Farzad V Farahani, Waldemar Karwowski, and Nichole R Lighthall. 2019. Application of Graph Theory for Identifying Connectivity Patterns in Human Brain Networks: A Systematic Review. *Frontiers in Psychology* 13 (2019).

[10] Józef Fiser and Richard N Aslin. 2002. Statistical learning of new visual feature combinations by infants. *Proceedings of the National Academy of Sciences of the United States of America* 99 (2002). Issue 24.

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

[12] Scott Fujimoto, Herke van Hoof, and David Meger. 2018. Addressing Function Approximation Error in Actor-Critic Methods. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*. PMLR.

[13] Advait Gadhikar, Sohom Mukherjee, and Rebekka Burkholz. 2023. Why Random Pruning Is All We Need to Start Sparse. In *Proceedings of the 40th International Conference on Machine Learning*. arXiv:2210.02412

[14] Trevor Gale, Erich Elsen, and Sara Hooker. 2019. The State of Sparsity in Deep Neural Networks. (2019). arXiv:1902.09574

[15] Carl Friedrich Gauss. 1857. *Theoria Motus Corporum Coelestium.* Little, Brown and company.

[16] Michael S. Gazzaniga, Richard B. Ivry, and George R. Mangun. 2014. *Cognitive neurosience. The biology of the mind.* W. W. Norton.

[17] Laura Graesser, Utku Evci, Erich Elsen, and Pablo Samuel Castro. 2022. The State of Sparse Training in Deep Reinforcement Learning. In *Proceedings of the 39th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 162)*. PMLR.

[18] Bram Grooten, Ghada Sokar, Shibhansh Dohare, Elena Mocanu, Matthew E. Taylor, Mykola Pechenizkiy, and Decebal Constantin Mocanu. 2023. Automatic Noise Filtering with Dynamic Sparse Training in Deep Reinforcement Learning. In *AAMAS '23*. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS).

[19] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018 (Proceedings of Machine Learning Research, Vol. 80)*. PMLR.

[20] Demis Hassabis, Dharshan Kumaran, Christopher Summerfield, and Matthew Botvinick. 2017. Neuroscience-Inspired Artificial Intelligence. *Neuron* 95 (2017).

[21] Torsten Hoefler, Dan Alistarh, Tal Ben-Nun, Nikoli Dryden, and Alexandra Peste. 2021. Sparsity in Deep Learning: Pruning and growth for efficient inference and training in neural networks. *Journal of Machine Learning Research* 22, 241 (2021).

[22] E. W. Lang, A. M. Tome, I. R. Keck, J. M. Gorriz-Saez, and C. G. LastNamePuntonet. 2012. Brain Connectivity Analysis: A Short Survey. *Computational Intelligence and Neuroscience* (2012).

[23] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2019. Continuous control with deep reinforcement learning. (2019). arXiv:1509.02971

[24] Marcus Lindskog, Pär Nyström, and Gustaf Gredebäck. 2021. Can the Brain Build Probability Distributions? *Frontiers in Psychology* (2021).

[25] Shiwei Liu, Tianlong Chen, Xiaohan Chen, Li Shen, Decebal Constantin Mocanu, Zhangyang Wang, and Mykola Pechenizkiy. 2022. The Unreasonable Effectiveness of Random Pruning: Return of the Most Naive Baseline for Sparse Training. (2022). arXiv:2202.02643

[26] Warren S. McCulloch and Walter Pitts. 1943. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics* (1943).

[27] Dick Meijer. 2020. *Probability Theory for Engineers.*

[28] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing Atari with Deep Reinforcement Learning. (2013). arXiv:1312.5602

[29] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin A. Riedmiller, Andreas Kirkeby Fidjeland, Georg Ostrovski, Stig Petersen, Charlie Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. 2015. Human-level control through deep reinforcement learning. *Nature* 518 (2015).

[30] Decebal Constantin Mocanu, Elena Mocanu, Tiago Pinto, Selima Curci, Phuong H. Nguyen, Madeleine Gibescu, Damien Ernst, and Zita A. Vale. 2021. Sparse Training Theory for Scalable and Efficient Agents. In *AAMAS '21*. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS).

[31] 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).

[32] Michael C. Mozer and Paul Smolensky. 1989. Skeletonization: A Technique for Trimming the Fat from a Network via Relevance Assessment. In *Advances in Neural Information Processing Systems 1*. Morgan Kaufmann Publishers Inc.

[33] Stuart Russell and Peter Norvig. 2022. *Artificial Intelligence: A Modern Approach* (fourth ed.). Pearson.

[34] John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. 2015. Trust Region Policy Optimization. In *Proceedings of the 32nd International Conference on Machine Learning*, Vol. 37. PMLR.

[35] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. (2017). arXiv:1707.06347

[36] Ghada Sokar, Elena Mocanu, Decebal Constantin Mocanu, Mykola Pechenizkiy, and Peter Stone. 2022. Dynamic Sparse Training for Deep Reinforcement Learning. (2022). arXiv:2106.04217

[37] Varun Sundar and Rajat Vadiraj Dwaraknath. 2021. [Reproducibility Report] Rigging the Lottery: Making All Tickets Winners. In *ML Reproducibility Challenge 2020*.

[38] Yiqin Tan, Pihe Hu, Ling Pan, Jiatai Huang, and Longbo Huang. 2023. RLx2: Training a Sparse Deep Reinforcement Learning Model from Scratch. In *The Eleventh International Conference on Learning Representations*.