Procedural Content Generation: Better Benchmarks for Transfer Reinforcement Learning

Matthias Müller-Brockhausen, Mike Preuss, Aske Plaat
Leiden Institute of Advanced Computer Science
Leiden University
The Netherlands
m.f.t.muller-brockhausen@liacs.leidenuniv.nl

***Abstract*—The idea of transfer in reinforcement learning (TRL) is intriguing: being able to transfer knowledge from one problem to another problem without learning everything from scratch. This promises quicker learning and learning more complex methods. To gain an insight into the field and to detect emerging trends, we performed a database search. We note a surprisingly late adoption of deep learning that starts in 2018. The introduction of deep learning has not yet solved the greatest challenge of TRL: generalization. Transfer between different domains works well when domains have strong similarities (e.g. MountainCar to Cartpole), and most TRL publications focus on different tasks within the same domain that have few differences. Most TRL applications we encountered compare their improvements against self-defined baselines, and the field is still missing unified benchmarks. We consider this to be a disappointing situation. For the future, we note that: (1) A clear measure of task similarity is needed. (2) Generalization needs to improve. Promising approaches merge deep learning with planning via MCTS or introduce memory through LSTMs. (3) The lack of benchmarking tools will be remedied to enable meaningful comparison and measure progress. Already Alchemy and Meta-World are emerging as interesting benchmark suites. We note that another development, the increase in procedural content generation (PCG), can improve both benchmarking and generalization in TRL.**

***Index Terms*—Transfer, Reinforcement Learning, Benchmarks, Procedural Content Generation**

I. Introduction

The need for Transfer in Reinforcement Learning (TRL) is growing through increased usage of deep reinforcement learning (RL), as shown in Figure 1. Deep networks are expensive to train [1], so cutting down the learning time by re-using previously gained knowledge is desirable. Computer games are currently often used as benchmarks or challenging test systems, and here it is especially the case that changes of different amplitude (e.g. patches) happen regularly. Recent AI successes on well-known complex games such as Dota2 [2] and StarCraft II [3] show that constant retraining is necessary as the underlying systems evolve on the same time scale as the trained AI systems.

Although the need for knowledge transfer in a reliable manner is clear, most experiments show limited generalization, and progress in TRL is limited. Our main goal in this paper is to detect why this is the case and how it can be cured. It turns out that a major problem lies in the sparsity of suitable benchmarks, and we see the use of Procedural Content Generation (PCG) as a recommended solution to this problem.

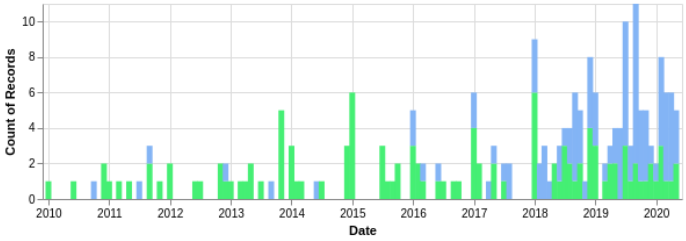

Fig. 1: Number of published papers per month. Light blue indicates entries that make use of deep neural networks, and green ones do not. The years 1985 to 2009 are cut off for readability, but the full graph is available at [4].

This paper has the following contributions.

1) We categorize the literature of Transfer in Reinforcement Learning (TRL), finding many different approaches and applications
2) The absence of clear benchmarks and a clear research agenda is noted
3) We provide a research agenda in which we stress the need for a clear measure of success, clear benchmarks, and suggest that Procedural Content Generation is ideally suited to provide such benchmarks for transfer reinforcement learning

Section II introduces related work (a meta-survey). To gain an unbiased insight into TRL, we scrape through a dataset (Section III). We explain the experimental parameters and decisions involved in TRL and identify trends in their usage (Section IV). After categorizing a large number of transfer experiments, we report the generalization capabilities of TRL (Section IV-C). We draft a research agenda for TRL (Section V) that lists current limitations and identifies directions that the field is likely headed to.

II. Related Work

Some summaries on individual aspects of Transfer in Reinforcement Learning (TRL) exist, and we will introduce them in chronological order. The first one was written by Bone in

978-1-6654-3886-5/21/$31.00 ©2021 IEEE

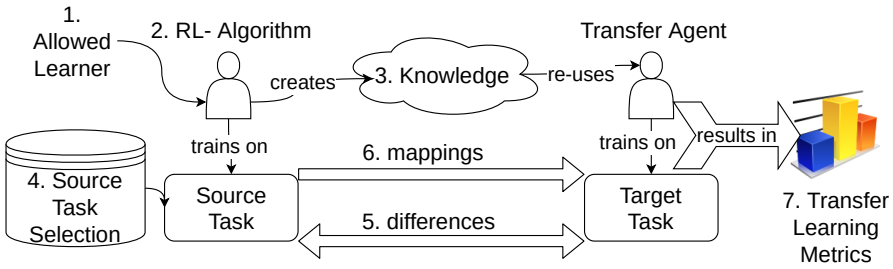

Fig. 2: A visualization of the transfer learning process. The chosen allowed learner (1) / algorithm (2) combination generates knowledge (3) while training on a selected source task (4). Source and target task have to differ (5), and depending on how large the differences are, task mappings (6) might be required. By using previously gained knowledge, the transfer agent then trains on the new task. Gathered data from source and target training can be distilled into the transfer learning metrics (7).

2008 [5], still pre-deep learning. A year later, Taylor & Stone followed [6] with a comprehensive survey. Its authors are the two most recurring names in the field in our data set. Two years later, they published a second survey with a focus on inter-task transfer [7]. In 2012, Lazaric formulated a framework [8] that enables the categorization of TRL experiments similar to [6]. Seven years later, in 2019, Da Silva & Costa published a survey focused on multiagent RL [9]. In 2020, multiple surveys appeared. One on curriculum learning in RL [10], one about multi-task transfer [11], and one about transfer in deep reinforcement learning [12]. Multiple surveys in one year might seem peculiar, but a look at Figure 1 shows a large increase in publications starting in 2018.

These surveys formulate frameworks to encompass the different TRL literature they encounter [6], [8], or analyze specific sub-fields of TRL [7], [9]–[12]. We aim for a comprehensive overview of all aspects of TRL. We perform scraping through the Microsoft Academic Graph [13], which contains over 209 Million papers, and include papers based on keywords in title and abstract. This yields some interesting statistics. For example, $\approx 74.8\%$ of the 270 relevant papers in our data set have not been included in any previously mentioned survey [5]–[12]. Moreover, it enables visualizations such as a publication timeline (Figure 1), a social network graph (Figure 3), and the creation of an interactive web tool [4] that facilitates the re-use of the data. Automatic quantitative analysis is to be seen as an addition to produce insightful figures and tools. It also served as tool to gain insight into the state of the field. But to form a true vision for our research agenda (Section V) we still relied on manual research.

III. Method

We search through a snapshot [14] of the Microsoft Academic Graph (MAG) [13] for entries that contain the three words "transfer," "reinforcement" and "learning" in either their abstract or title. We create a spreadsheet using the terminology introduced by [6]: transfer dimension, allowed task differences, source task selection, task mappings, transferred knowledge, allowed learners, transfer metrics. We also record which RL algorithm was used (e.g., Q, DQN, PPO, DDPG), whether the paper publishes additional resources such as source code, and we check if the links work [15]. All data and resources, including the spreadsheet, and an interactive data viewer, are openly available at [4].

Figure 2 provides a visualization of the process of a TRL experiment. For terminology, we stay as close as possible to [6]. The RL-algorithm (2) provides more details about the learning algorithm that is used (1) and how transferred knowledge is re-used. We briefly review which information was gathered from the papers. The content in parentheses behind keywords refers to the transfer process step in Figure 2 if numerical, or otherwise the abbreviation used in the spreadsheet. By explaining the Figure, we also follow the workflow of setting up a typical TRL experiment, and we describe essential choices of the authors.

The allowed learner (1) places restrictions on how transfer is approached and influences experimental parameters such as the pool of available reinforcement learning methods (2). The options are temporal difference learning (TD), model-based learning (MB), relational learning (RRL), hierarchical learning (H), batch learning (Batch), Bayesian methods (bayes), and case-based reasoning (CBR). Because many different algorithms exist per allowed learner method, we also note which RL-algorithm was used (2). The next step is to determine the type of knowledge (3) to transfer. The easiest methods re-use what was learned, e.g., the action-value function (Q), policy (π), task model (model), found prior distributions (pri), or experience instances (I).

However, higher-level knowledge can also be used in a variety of ways. One can extract partial policies (π_p) or options (options) for guidance. Rules or advice from experts (ra), be it human demonstration or successfully trained agents, can guide the training process. This advice can manifest itself in a shaped reward (R). Some algorithms can identify and learn important features (fea), autonomously find sub-task definitions (sub), or build a proto-value function (pvf). Other methods to transfer knowledge are a Variational auto-encoder (VAE) [16] or policy distillation [17].

To gather the re-usable knowledge,[1] an agent needs to train

[1]Simplified for the Figure. Knowledge used to train the transfer agent can stem from anywhere, like human made demonstrations or rules.

on a selected source task (4). Either all previously seen tasks are used (all), one is selected by the author (h), a library of tasks to choose from is defined by the author (lib), or the agent has to modify the source task to gain the required knowledge autonomously (mod). There are two important factors on transfer between the source and target task. The first are differences (5) an agent has to handle between tasks. These can be small, such as an alternating start (s_i) or end (s_f) position, another level layout, or a different number of encountered objects (#). Changes can also affect the number of involved objects (#), transition function (t), state variables (v), the action set (a), or reward function (r). Secondly, mapping between tasks (6) could be required. The agent can get no mapping (N/A) or learn it from experience (exp). The mapping can also be provided as higher-level knowledge (T), created by humans (sup), derived from action mapping (M_a), or a grouping of state variables (sv_g).

The transfer experiment results in metrics (7) that indicate success. One can measure an improvement in the Time to Threshold (tt), so how many fewer steps did the agent have to train to reach a previously specified reward threshold. If the transfer agent training starts at a higher reward than an agent that started from scratch, one has measured a Jumpstart (j). The transfer agent can also achieve a higher total reward (tr), and the difference between transfer agent and training from scratch is called transfer ratio[2]. Lastly, Asymptotic Performance (ap) indicates whether the final learned performance has improved.

IV. Analysis

To get an overview of the dataset, the social network structure of the citation data is visualized (section IV-A), and insights from the categorization are presented (section IV-B). The main goal of transfer learning is generalization. We analyzed the papers for the strength of generalization that has been achieved in TRL (section IV-C).

Many more insights have been found, and we refer the reader to discover the full data on an interactive website [4].

A. Social Network Structure

Figure 3 shows a visualization of citations between authors. Colors indicate different communities, as identified by the community detection algorithm [19]. Note that due to noise and missing values in the base data, 63 ($\approx 23\%$) of the entries are missing reference data and are therefore not present in Figure 3.

Eight connected components have been identified. Only one of these contains most entries. The other seven are independent. The independent components are not included in Figure 3. The largest component consists of nine individual communities. For each community, we attempt to identify common aspects and succeeded for four of them. Two communities focus on the different types of allowed learners. Red at the far left contains mostly case-based reasoning (CBR), and purple at the bottom left hierarchical (H) and Bayesian RL (bayes). The dark green community at the far right contains mostly Simulation to Reality (Sim2Real) experiments. From here on out, the similarities are already declining. As for the black group, the only link we could find is that they all applied Q learning at some point.

[2]We omit the transfer ratio as performance improvement measurement in our data and in Figure 4, as it is an extension of the total reward [18].

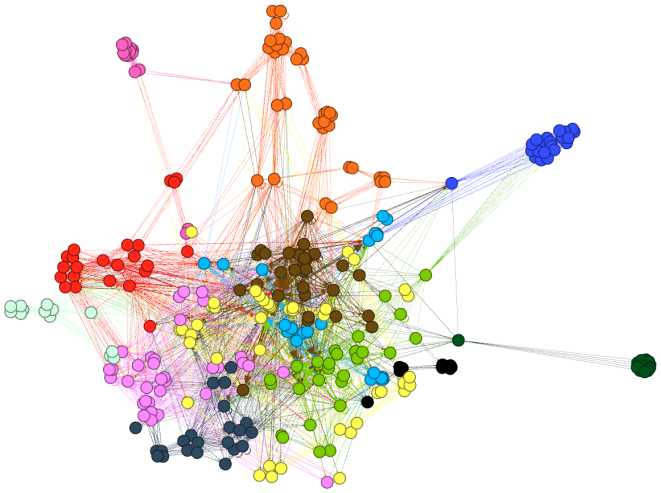

Fig. 3: Directed Social Network Graph of Authors citing each other. The red group contains mostly case-based reasoning learners, purple contains hierarchical algorithms and Bayesian RL, dark green is the Sim2Real community, black papers all applied Q-learning, dark blue contains reward shaping, orange reports mostly on applications (such as energy consumption in buildings). Layout determined by Force Atlas [20].

In addition to hierarchical learners, purple also contains numerous 2D navigation experiments, but not exclusively. Dark blue at the bottom contains many reward shaping experiments that sample from a (human) demonstration or world model prediction.

For the remaining five communities, no real common aspects could be found, except that $\approx$ 44% in the orange community focus on real-world engineering problems such as optimizing power usage for buildings [21], air conditioners [22], or collision avoidance for autonomous vehicles [23].

B. Category Data

Of the 270 entries, 202 ($\approx 74.8\%$) have not been in previous surveys of the field [5]–[12]. In the following, parentheses will indicate the number of papers in the dataset relating to a specific message, for example 10% (27) entries have not been approved through peer-review (23 arxiv, 4 rejected on openreview). Text in brackets refers to variable abbreviation presented in Section III. Furthermore, one entry may contain multiple tags per category. We follow the order of the steps of the transfer learning process of Figure 2) in presenting the data.

First, the allowed learner is chosen. The majority of papers uses regular temporal difference methods (241) [TD], followed by hierarchical learning (46) [H], model-based learning (31) [model], Bayesian learning (20) [Bayes], batch learning (13),

relational learning (11) [RRL], policy search learning (10) [PS], case-based reasoning (9) [CBR], and one linear programming entry. TRL transfers knowledge between different tasks, and it is no surprise that hierarchical learning is the second most applied learner type for TRL because the multiple tasks might be hierarchically related. Moreover, singular large tasks can be decomposed into multiple (hierarchically ordered) subtasks for transfer [6].

The allowed learner narrows the pool of RL algorithms that can be chosen. The most popular algorithms are tabular Q-Learning (124), DQN (36), SARSA (28), DDPG (11), PPO (10), FQI (10), DDQN (8), A3C (7), LSPI (7), and Policy Gradient (6). The high occurrence of tabular Q-Learning could give the impression that deep learning is not prevalent in TRL yet, but $\approx 36\%$ (99) already use deep neural networks. Moreover, Figure 1 depicts a clear trend towards deep learning that started in 2018.

Although the number of 3D environments that deep neural networks (DNN) (17) are applied to are almost the same as for tabular algorithms (14), their complexity differs. Tabular algorithms focus mostly on balancing problems, such as Mountain Car 3D (11) or controlling joints in a real-world RoboCup robot [24], [25]. The 3D environments that DNN's are applied to are more complex. AirSim requires full free 3D navigation through flying, plus the policy is transferred to a real drone [26], and Mujoco combines controlling multiple joints with moving in a 3D World [27].

The third step (see Figure) is to decide what knowledge should be extracted or transferred. The most popular methods are also the easiest to transfer, namely just re-using the previously learned action-value function (72) [Q] or policy (64) [π]. All other methods require more sophistication, such as extracting and re-using relevant features (38) [fea] or guiding training via Advisor / Teacher data (22) [advisor]. Shaping the reward (20) [R] is also an often-used means to transfer knowledge. Less often used are approaches like collecting experience instances (14), building a task model (14) or sub-task definition (8), defining rules (14), finding options (12), or collecting distribution priors (9). There are also new ways to transfer knowledge between tasks. One of these is policy distillation (4). Most entries used it to summarize multiple learned policies into a single one [17], [28], [29], but it can also be used for simulation to reality transfer [30]. The distillation process and the advisor method have one element in common: They both re-use the same type of saved knowledge. Another transfer method that was introduced through deep neural networks is the Variational Auto Encoder (VAE). These networks can help to automatically identify relevant features in the latent space and thus allow for universal control policies [16]. Moreover, Q-functions can be generated algorithmically [31]. This is related to hyper-networks, where a neural network outputs the weights used in another network [32].

The fourth step is determining the source task from which knowledge should be transferred. Here most approaches perform hand-selection (137) [h]. 48 entries let the algorithm use all tasks (all), 32 a library of selected tasks (lib), and two times, the agent automatically modifies a provided source task (mod).

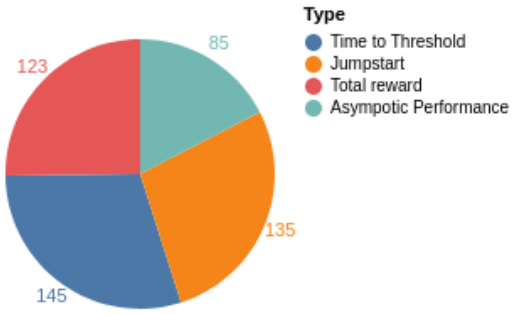

Fig. 4: Number of times papers reported that transfer performance improvements (Section III) were measured by category.

The fifth and most important question to answer is: What kind of differences in tasks will the learner have to handle? The most frequently occurring differences are in transition dynamics (115) [t] of the environment. Different transition dynamics refer to changes in parameters between tasks that influence how the world changes per timestep. For example, if the agent were to control an airplane, changes in gravity, weight, or friction would count as transition dynamic differences. The second to fourth place goes to navigation-related tasks, namely changes in the goal- (95) [s_f] or start- (91) [s_i] position or the level layout (56). Values that the agent receives are also well suited to accelerate the transition. For that, the received observation (52) [v], the action set (43) [a], the number of objects encoded in the observation (32) [#], or the reward (29) [r] given to the agent can be adjusted.

Based on the task differences, the sixth step, namely mappings between tasks, must be determined. They are not often necessary as 164 do not use any mapping between tasks. Nevertheless, when they are used, the majority of algorithms try to learn them from experience (40) [exp], or they are given manually (25) [sup]. The less-used methods are action mappings (8) [M_a] or groupings of state variables [sv_g]. One interesting approach for mappings in image-based domains is the Generative Adversarial Network (GAN) [33].

The last step in transfer learning, and the most important to get an idea of the experimental success, are the associated metrics (Figure 4). 146 entries achieved a decrease in the time to reach a threshold (tt), while 135 were able to jump-start (jp) the reward in a new setting. 123 entries achieved a higher total reward (tr) compared to no transfer at the end of training, and 85 entries trained agents that show asymptotic performance (ap) after transfer. Many papers measured multiple metrics of success. 91 achieved two, 38 three, and 27 all four metrics.

We also looked at what kind of problem TRL is mostly applied to. The most often recurring applications are navigation (122), robotics (56), classic control (42), and games (26). The navigation domain is the most diverse. The majority of experiments inspect 2D (110) instead of 3D (12) worlds. To further simplify 2D worlds, 79 entries use a grid instead

of continuous navigation. Most entries formulate their own problem, but there are also some recurring standardized environments such as the Taxi world from Dietterrich [34] or the blocks world by Langley [35]. Although most 2D grid-level layouts do not cite anyone, there is one recurring citation: the three-room grid-level by Thrun [36]. One entry also uses a slightly adjusted version of the three-room grid-level [37]. The goal of these papers is to test if transfer is possible. We would expect test domains to be different, challenging, and standarized. The state of affair that we encountered is lacking in this respect. While some problems repeat, there is no unified benchmark, and the few existing benchmarks are not dynamic in the sense that they could adapt their difficulty or similarity (Section V). Given the large number of entries on the topic, we were surprised that there is no real benchmark to assess the planning capabilities of an RL algorithm in the navigation domain. The ProcGen environment is such a benchmark for maze navigation [38]. However, ProcGen is an environment with procedurally generated levels, and most entries here use one (or more) static levels. And [39] has found that while (deep) RL can learn to generalize to generated levels within the same distribution, it can not handle arbitrary level layouts.

We encountered 27 game-related entries, of which 11 focus on Atari and only 4 on board games. Few complex games are approached with TRL, like Unreal Tournament [40], StarCraft [41], or GVGAI [42]. Simpler games include Sonic 3 [43] or Pinball [44].

The growing use of neural networks comes with a drawback, namely the reproducibility of results. The ICLR Reproducibility Challenge [45] from 2018 underlines the problem, as less than 33% of papers are rated as properly reproducible [46]. The most straightforward way to make a code-based science experiment reproducible is by publishing the source code. Only 15 entries did this, but at least 148 contain pseudo-code. Moreover, 31 entries link additional resources. 10 of these are websites, but 8 of them are not available anymore. The two websites that are still reachable summarize different short videos of robotics tasks on one page. The remaining 21 links are videos.

Another important aspect related to reproducibility is the software libraries used in the experiment code. Even when closed source, some entries do share details about used libraries. As machine learning backend, TensorFlow (20) and 7 PyTorch (7) are popular. Only 4 of the TensorFlow uses are from DeepMind, so the library's popularity seems community-driven. Regarding RL environments, 24 use OpenAI's Gym [47], 5 the proprietary physics playground Mujoco [48], 4 the unreal engine based 3D navigation simulator AirSim [49], and 2 the continuous control benchmark RLLab [50].

C. Generalization Capabilities

Achieving generalization by transferring knowledge from one task to another remains challenging. The more different tasks are, the harder generalization becomes. One algorithm, such as AlphaZero, may be able to learn to play at world champion level in three different board games [51]. The limitation is that one network has to be trained again for each game. Unlike humans, the AlphaZero AI can not yet generalize and transfer knowledge from one domain to another similar domain, even when the internal network architecture is identical. The field of TRL revolves around finding algorithms to enable this transfer between different tasks in the same domain. Transfer between differing domains works better the more similar they are, and when only the transition dynamics change.

For example, transferring a Q-learner from MountainCar to Cartpole works well [18]. Also, transfer from CartPole to Bicycle works well [52] or a three-linked CartPole to the Quadrotor [53] control tasks. Another popular transfer domain is RoboCup, with 24 entries. Many of the experiments focus on increasing the number of players involved from 3v2 to 4v3 [54] or 3v2 to 6v5 [55] in KeepAway. Others explore multi-task experiments such as MoveDownfield to BreakAway [56].

Although the RoboCup challenge could be seen as 2D navigation, it does not involve the same amount of planning as is required to navigate through a 2D maze, whether it is grid-based or continuous. For 2D maze navigation, which is intuitive for most humans, reinforcement learning needs special help. For example, repositioning doors in a level whose layout has not changed requires advisors [57]. Picking up a sequence of keys and then moving through doors can be solved by adding options [58]. For regular search algorithms such as A* these task changes would be easy to solve. However, for large, complicated 2D and 3D navigation domains, one has to incorporate planning into RL. There already exist two great examples of this. First of all, Go-Explore [59]. Many of the Atari games that [59] tackles can be viewed as multi-level 2D navigation tasks, such as Montezuma's Revenge, Berzerk, and Private Eye. Go-Explore effectively combines planning with regular reinforcement learning and is good at these navigation and planning Atari games. Other promising approaches exist such as MuZero [60] or MCTSnet [61].

Achieving reliable transfer of knowledge gained from perfect simulations to the real-world is another unsolved transfer problem. We found 23 entries in this sub-field (Sim2Real). Many of these focus on moving joints (3), which could be compared to classic control tasks. It can also be extended to multiple joints that form a robotic arm (12) to interact with objects. To narrow the gap between simulation and reality, in many papers, noise is applied for regularization or smoothing, e.g., Gaussian Noise [62], uniform random noise [63], the Simulation Optimization Bias (SOB) [64]). There are also efforts to make the noise obsolete [65].

The findings underline that TRL only works well when some similarity between source and target tasks can be found. In this sense, generalization is in the eye of the beholder, and there is a long way to go. Nevertheless, one paper's generalization capabilities are impressive: By encoding the Video and Audio output of an Atari game into a multi-modal latent space, a policy was trained on video-only that can

transfer its performance to audio-only input [66].

V. Research Agenda

In this literature review, we have categorized around 300 papers on transfer reinforcement learning. We have seen many different approaches trying to transfer knowledge between many different applications. The measure of success is generalization: how well knowledge can be transferred between different applications. We note (1) in supervised learning, transfer has achieved more success [67] than in reinforcement learning. TRL is still a young field. The first deep learning TRL papers appeared around 2010, but only in 2018 did the field really start adopting it. Deep learning methods can be expected to continue to yield good performance. Transfer reinforcement learning should continue to focus on transfer of network parameters. (2) The large diversity in applications and methods makes progress comparisons difficult. Also, we noted a lack of dynamic benchmarks (Section IV-B). (3) Generalization is hard, except when applications are clearly related. These observations bring us to the following research agenda.

1) A clear measure of transfer capabilities is needed in transfer reinforcement learning [68], [69]. This implies a universal measure of similarities between tasks/domains.
2) The combination of planning and learning can be expected to improve (as already shown by Go-Explore [59] and Mu-Zero [60]). Transfer reinforcement learning should focus on general planning methods [61].
3) Benchmarks are needed that are standardized, challenging, and dynamic (Section IV-B). Procedural Content Generation can be leveraged to enable fine-grained control on the different levels of difficulty and task similarity.

Specifically, in Section IV-C, we briefly mentioned the inability of TRL to generalize to procedurally generated levels in 2D navigation [39]. Although it can generalize to different levels from one distribution, it can not handle arbitrary levels. One trend that could help overcome this problem, at least for navigation-related tasks, is the fusion of learning and planning, as in model-based reinforcement learning [70]. Nevertheless, it is still an active research field with contributions such as a framework trying to unify the two [71]. LSTMs also play an increasingly important role in improving generalization, as they can already enable the adaptability to different layouts of 2D navigation levels [72], [73]. Furthermore, we view curriculum learning as a form of planning, as the creation of curricula inherently requires planning, and it has shown success as a TRL method [41], [74].

Unfortunately, using benchmarks to compare novel approaches is not the norm in TRL yet. Contrary to Supervised Learning, where achieved accuracy percentages on well-known data sets can be perfectly compared, each sub-field and application would require different benchmarks to assess specific transfer capabilities of varying algorithms. But as we have shown (Section IV), there are many experiments in similar fields like robotics or navigation that could profit from each other. There are already interesting simulators like Mujoco [48] for continuous control, ProcGen [38] for generalization capabilities, Meta-World [75] and Alchemy [76] for meta-TRL. However, there is still a plethora of other applications missing benchmarks to assess, e.g., game-playing AI [77] or 2D navigation. ProcGen does feature 2D Maze levels, but no benchmark verifies whether an algorithm can adapt to different movement styles (grid vs. continuous) or movement types (top-down vs. side-scroller).

Most experiments we encountered only transfer between fixed sets of tasks. As PCG has proven to be a reliable tool to improve generalization performance in RL [78], [79], the adoption of PCG is the logical next step for TRL. One could generate a seemingly infinite amount of different transfer tasks, and in game-like environments this is presumably relatively easy. Furthermore, PCG benchmarks would enable agents to control generation parameters that influence the difficulty, resulting in a dynamic curriculum that improves learning performance [74]. Moreover, the generation parameters could be chosen to influence task similarity. Such a quantifiable similarity control could be used as a benchmark metric to determine how much tasks may differ until the tested algorithm can not transfer efficiently anymore.

An ideal tool for the future would be a database, similar to OpenML [80], that contains machine processible information on all available TRL tasks/experiments. When approaching a new task, the trove of data could be used to cluster similar domains via task similarity metrics [68], to identify promising source tasks to transfer from. This would also allow a leaderboard style comparison of how well which algorithm transfers between what tasks, as has been done in the GVGAI [81] per game (set). While new task similarity metrics are still developed [69], the problem identified by [68] persists, that there is no one universal metric to encompass all similarity dimensions. The described database would enable the combination of all existing similarity metrics and the performance of different algorithms in transferring from one task to another to train a supervised network that outputs a singular numerical transfer success probability.

VI. Conclusion

By borrowing methods and a dataset from the field of social network analysis (Section III), we have created a unique survey about Transfer in Reinforcement Learning (TRL). We have collected tabular data about transfer-related metrics similar to [6] but on a larger scale. We verified that out of 270 unique TRL entries, $\approx 74.8\%$ have not been included in any of the previous surveys [5]–[12]. Because of the large scope, in which not every single entry can be mentioned textually, we created a website [4] that gives a better overview of the dataset with more graphs and the ability to interactively filter through the data. With this data, we have underlined the large diversity of applications for TRL, which shows a focus on navigation, robotics, classic control, and games. We find that transfer, at least in RL, has a hard time generalizing to different problem variations (Section IV-C). The transfer

that works best is to problems that are similar. In tasks that involve planning, such as routes through a 2D levels, TRL lacks as it can not generalize to arbitrary layouts yet [39]. We do see an increase in methods that try to merge planning with learning (Section V) to overcome this limitation. Another issue is the comparability of algorithms. Most approaches define their own slightly different version of known problems and compare their results to self-defined baselines. The field requires more benchmarks like Alchemy [76], Meta-World [75], or ProcGen [38] to quantify transfer performance properly and compare different algorithms. These benchmarks will increasingly include more procedural content generation to challenge generalization capabilities further. We provide a research agenda outlining how to achieve this goal. In our view, the game AI field is especially well suited to pursue this research as it already employs the necessary algorithms and an abundance of configurable problems.

REFERENCES

[1] D. Patterson, J. Gonzalez, Q. Le, C. Liang, L.-M. Munguia, D. Rothchild, D. So, M. Texier, and J. Dean, "Carbon emissions and large neural network training," *arXiv preprint arXiv:2104.10350*, 2021.

[2] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Debiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, R. Józefowicz, S. Gray, C. Olsson, J. Pachocki, M. Petrov, H. P. de Oliveira Pinto, J. Raiman, T. Salimans, J. Schlatter, J. Schneider, S. Sidor, I. Sutskever, J. Tang, F. Wolski, and S. Zhang, "Dota 2 with large scale deep reinforcement learning," *CoRR*, vol. abs/1912.06680, 2019. [Online]. Available: http://arxiv.org/abs/1912.06680

[3] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev *et al.*, "Grandmaster level in starcraft ii using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.

[4] M. Müller-Brockhausen. (2021) Interactive website to explore the data presented in this paper. [Online]. Available: https://hizoul.github.io/trlsnap/

[5] N. Bone, "A survey of transfer learning methods for reinforcement learning," 2008.

[6] M. E. Taylor and P. Stone, "Transfer learning for reinforcement learning domains: A survey." *Journal of Machine Learning Research*, vol. 10, no. 7, 2009.

[7] ——, "An introduction to intertask transfer for reinforcement learning," *Ai Magazine*, vol. 32, no. 1, pp. 15–15, 2011.

[8] A. Lazaric, H. I. Hal, and A. Lazaric, "Transfer in reinforcement learning: a framework and a survey, in "reinforcement learning: State of the art," in *Reinforcement Learning*. Springer, 2012, pp. 143–173.

[9] F. Silva and A. Costa, "A survey on transfer learning for multiagent reinforcement learning systems," *J. Artif. Intell. Res.*, vol. 64, pp. 645–703, 2019.

[10] S. Narvekar, B. Peng, M. Leonetti, J. Sinapov, M. E. Taylor, and P. Stone, "Curriculum learning for reinforcement learning domains: A framework and survey," *Journal of Machine Learning Research*, vol. 21, no. 181, pp. 1–50, 2020.

[11] N. Vithayathil Varghese and Q. H. Mahmoud, "A survey of multi-task deep reinforcement learning," *Electronics*, vol. 9, no. 9, p. 1363, 2020.

[12] Z. Zhu, K. Lin, and J. Zhou, "Transfer learning in deep reinforcement learning: A survey," 2020.

[13] A. Sinha, Z. Shen, Y. Song, H. Ma, D. Eide, B.-J. P. Hsu, and K. Wang, "An overview of microsoft academic service (mas) and applications," in *Proceedings of the 24th International Conference on World Wide Web*, ser. WWW '15 Companion. New York, NY, USA: Association for Computing Machinery, 2015, p. 243–246. [Online]. Available: https://doi.org/10.1145/2740908.2742839

[14] Michael Färber, "The Microsoft Academic Knowledge Graph: A Linked Data Source with 8 Billion Triples of Scholarly Data," in *Proceedings of the 18th International Semantic Web Conference*, ser. ISWC'19, 2019, pp. 113–129. [Online]. Available: https://doi.org/10.1007/978-3-030-30796-7_8

[15] J. Zittrain, K. Albert, and L. Lessig, "Perma: Scoping and addressing the problem of link and reference rot in legal citations," *LIM*, vol. 14, p. 88, 2014.

[16] J. Yang, B. Petersen, H. Zha, and D. Faissol, "Single episode policy transfer in reinforcement learning," 2020.

[17] M. Barekatain, R. Yonetani, and M. Hamaya, "Multipolar: Multi-source policy aggregation for transfer reinforcement learning between diverse environmental dynamics," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, C. Bessiere, Ed. International Joint Conferences on Artificial Intelligence Organization, 7 2020, pp. 3108–3116, main track. [Online]. Available: https://doi.org/10.24963/ijcai.2020/430

[18] H. B. Ammar, *Automated Transfer in Reinforcement Learning*. Citeseer, 2013.

[19] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of statistical mechanics: theory and experiment*, vol. 2008, no. 10, p. P10008, 2008.

[20] M. Jacomy, T. Venturini, S. Heymann, and M. Bastian, "Forceatlas2, a continuous graph layout algorithm for handy network visualization designed for the gephi software," *PloS one*, vol. 9, no. 6, p. e98679, 2014.

[21] E. Mocanu, P. H. Nguyen, W. L. Kling, and M. Gibescu, "Unsupervised energy prediction in a smart grid context using reinforcement cross-building transfer learning," *Energy and Buildings*, vol. 116, pp. 646–655, 2016.

[22] P. Lissa, M. Schukat, and E. Barrett, "Transfer learning applied to reinforcement learning-based hvac control," *SN Computer Science*, vol. 1, no. 3, pp. 1–12, 2020.

[23] K. Miriti, "Integrating policy transfer, policy reuse and experience replay in speeding up reinforcement learning of the obstacle avoidance task," Ph.D. dissertation, University of Nairobi, 2014.

[24] S. Barrett, M. E. Taylor, and P. Stone, "Transfer learning for reinforcement learning on a physical robot," in *Ninth International Conference on Autonomous Agents and Multiagent Systems-Adaptive Learning Agents Workshop (AAMAS-ALA)*, vol. 1, 2010.

[25] L. A. Celiberto, J. P. Matsuura, R. López de Mántaras, and R. Bianchi, "Using cases as heuristics in reinforcement learning: a transfer learning application," 2011.

[26] I. Yoon, M. A. Anwar, R. V. Joshi, T. Rakshit, and A. Raychowdhury, "Hierarchical memory system with stt-mram and sram to support transfer and real-time reinforcement learning in autonomous drones," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 9, no. 3, pp. 485–497, 2019.

[27] S. Xie, A. Galashov, S. Liu, S. Hou, R. Pascanu, N. Heess, and Y. W. Teh, "Transferring task goals via hierarchical reinforcement learning," 2018. [Online]. Available: https://openreview.net/forum?id=S1Y6TtJvG

[28] H. Yin and S. Pan, "Knowledge transfer for deep reinforcement learning with hierarchical experience replay," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.

[29] I. Xiao, "A distributed reinforcement learning solution with knowledge transfer capability for a bike rebalancing problem," 2018.

[30] R. Traoré, H. Caselles-Dupré, T. Lesort, T. Sun, N. Díaz-Rodríguez, and D. Filliat, "Continual reinforcement learning deployed in real-life using policy distillation and sim2real transfer," 2019.

[31] I. Arnekvist, D. Kragic, and J. A. Stork, "Vpe: Variational policy embedding for transfer reinforcement learning," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 36–42.

[32] K. O. Stanley, D. B. D'Ambrosio, and J. Gauci, "A hypercube-based encoding for evolving large-scale neural networks," *Artificial life*, vol. 15, no. 2, pp. 185–212, 2009.

[33] S. Gamrian and Y. Goldberg, "Transfer learning for related reinforcement learning tasks via image-to-image translation," in *International Conference on Machine Learning*. PMLR, 2019, pp. 2063–2072.

[34] T. G. Dietterich, "Hierarchical reinforcement learning with the maxq value function decomposition," *Journal of artificial intelligence research*, vol. 13, pp. 227–303, 2000.

[35] P. Langley, *Elements of Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1995.

[36] A. Schwartz and S. Thrun, "Finding structure in reinforcement learning," *Advances in neural information processing systems*, vol. 7, pp. 385–392, 1995.

[37] Z. Arabasadi and N. Didkar, "Learning transfer automatic through data mining in reinforcement learning," *International Journal of Computer Applications*, vol. 88, no. 13, 2014.

[38] K. Cobbe, C. Hesse, J. Hilton, and J. Schulman, "Leveraging procedural generation to benchmark reinforcement learning," *arXiv preprint arXiv:1912.01588*, 2019.

[39] N. Justesen, R. R. Torrado, P. Bontrager, A. Khalifa, J. Togelius, and S. Risi, "Illuminating generalization in deep reinforcement learning through procedural level generation," *arXiv preprint arXiv:1806.10729*, 2018.

[40] Y. Hou, Y.-S. Ong, L. Feng, and J. M. Zurada, "An evolutionary transfer reinforcement learning framework for multiagent systems," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 4, pp. 601–615, 2017.

[41] K. Shao, Y. Zhu, and D. Zhao, "Starcraft micromanagement with reinforcement learning and curriculum transfer learning," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 3, no. 1, pp. 73–84, 2018.

[42] K. Narasimhan, R. Barzilay, and T. Jaakkola, "Grounding language for transfer in deep reinforcement learning," *Journal of Artificial Intelligence Research*, vol. 63, pp. 849–874, 2018.

[43] N. Hamilton, L. Schlemmer, C. Menart, C. Waddington, T. Jenkins, and T. T. Johnson, "Sonic to knuckles: evaluations on transfer reinforcement learning," in *Unmanned Systems Technology XXII*, vol. 11425. International Society for Optics and Photonics, 2020, p. 114250J.

[44] T. Yang, J. Hao, Z. Meng, Z. Zhang, Y. Hu, Y. Cheng, C. Fan, W. Wang, W. Liu, Z. Wang *et al.*, "Efficient deep reinforcement learning via adaptive policy transfer," *arXiv preprint arXiv:2002.08037*, 2020.

[45] J. Pineau, K. Sinha, G. Fried, R. N. Ke, and H. Larochelle, "Iclr reproducibility challenge," *ReScience C*, vol. 5, no. 2, May 2019. [Online]. Available: https://zenodo.org/record/3158244/files/article.pdf

[46] J. Pineau. (2020-01-18) Reproducibility, reusability, and robustness in deep reinforcement learning. International Conference on Learning Representations (ICLR). [Online]. Available: https://www.youtube.com/watch?v=Vh4H0gOwdIg

[47] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," 2016.

[48] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 5026–5033.

[49] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and Service Robotics*, 2017. [Online]. Available: https://arxiv.org/abs/1705.05065

[50] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel, "Benchmarking deep reinforcement learning for continuous control," in *International conference on machine learning*. PMLR, 2016, pp. 1329–1338.

[51] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel *et al.*, "Mastering atari, go, chess and shogi by planning with a learned model," *Nature*, vol. 588, no. 7839, pp. 604–609, 2020.

[52] G. Joshi and G. Chowdhary, "Cross-domain transfer in reinforcement learning using target apprentice," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 7525–7532.

[53] H. B. Ammar, E. Eaton, P. Ruvolo, and M. Taylor, "Unsupervised cross-domain transfer in policy gradient reinforcement learning via manifold alignment," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 29, no. 1, 2015.

[54] V. Soni and S. Singh, "Using homomorphisms to transfer options across continuous reinforcement learning domains," in *AAAI*, vol. 6, 2006, pp. 494–499.

[55] I. Partalas, G. Tsoumakas, K. Tzevanidis, and I. P. Vlahavas, "Transferring experience in reinforcement learning through task decomposition." in *AAMAS (2)*, 2009, pp. 1193–1194.

[56] L. Torrey, J. Shavlik, T. Walker, and R. Maclin, "Skill acquisition via transfer learning and advice taking," in *European Conference on Machine Learning*. Springer, 2006, pp. 425–436.

[57] H. Plisnier, D. Steckelmacher, D. M. Roijers, and A. Nowé, "Transfer reinforcement learning across environment dynamics with multiple advisors." in *BNAIC/BENELEARN*, 2019.

[58] G. D. Konidaris and A. G. Barto, "Building portable options: Skill transfer in reinforcement learning." in *IJCAI*, vol. 7, 2007, pp. 895–900.

[59] A. Ecoffet, J. Huizinga, J. Lehman, K. O. Stanley, and J. Clune, "First return, then explore," *Nature*, vol. 590, no. 7847, pp. 580–586, 2021.

[60] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel *et al.*, "Mastering atari, go, chess and shogi by planning with a learned model," *Nature*, vol. 588, no. 7839, pp. 604–609, 2020.

[61] A. Guez, T. Weber, I. Antonoglou, K. Simonyan, O. Vinyals, D. Wierstra, R. Munos, and D. Silver, "Learning to search with mctsnets," in *International Conference on Machine Learning*. PMLR, 2018, pp. 1822–1831.

[62] I. Higgins, A. Pal, A. Rusu, L. Matthey, C. Burgess, A. Pritzel, M. Botvinick, C. Blundell, and A. Lerchner, "Darla: Improving zero-shot transfer in reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1480–1490.

[63] A. Anwar and A. Raychowdhury, "Autonomous navigation via deep reinforcement learning for resource constraint edge nodes using transfer learning," *IEEE Access*, vol. 8, pp. 26 549–26 560, 2020.

[64] F. Muratore, M. Gienger, and J. Peters, "Assessing transferability from simulation to reality for reinforcement learning," *IEEE transactions on pattern analysis and machine intelligence*, 2019.

[65] M. Kaspar, J. D. M. Osorio, and J. Bock, "Sim2real transfer for reinforcement learning without dynamics randomization," *arXiv preprint arXiv:2002.11635*, 2020.

[66] R. Silva, M. Vasco, F. S. Melo, A. Paiva, and M. Veloso, "Playing games in the dark: An approach for cross-modality transfer in reinforcement learning," *arXiv preprint arXiv:1911.12851*, 2019.

[67] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *Journal of Big data*, vol. 3, no. 1, pp. 1–40, 2016.

[68] J. L. Carroll and K. Seppi, "Task similarity measures for transfer in reinforcement learning task libraries," in *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, vol. 2. IEEE, 2005, pp. 803–808.

[69] H. B. Ammar, E. Eaton, M. E. Taylor, D. C. Mocanu, K. Driessens, G. Weiss, and K. Tuyls, "An automated measure of mdp similarity for transfer in reinforcement learning," in *Workshops at the Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

[70] R. S. Sutton, "Dyna, an integrated architecture for learning, planning, and reacting," *ACM Sigart Bulletin*, vol. 2, no. 4, pp. 160–163, 1991.

[71] T. M. Moerland, J. Broekens, and C. M. Jonker, "A framework for reinforcement learning and planning," 2020.

[72] Y. Duan, J. Schulman, X. Chen, P. L. Bartlett, I. Sutskever, and P. Abbeel, "Rl^2: Fast reinforcement learning via slow reinforcement learning," 2016.

[73] A. Y. Sorokin and M. S. Burtsev, "Episodic memory transfer for multi-task reinforcement learning," *Biologically inspired cognitive architectures*, vol. 26, pp. 91–95, 2018.

[74] M. C. Green, B. Sergent, P. Shandilya, and V. Kumar, "Evolutionarily-curated curriculum learning for deep reinforcement learning agents," 2019.

[75] T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine, "Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning," in *Conference on Robot Learning*. PMLR, 2020, pp. 1094–1100.

[76] J. X. Wang, M. King, N. Porcel, Z. Kurth-Nelson, T. Zhu, C. Deck, P. Choy, M. Cassin, M. Reynolds, F. Song, G. Buttimore, D. P. Reichert, N. Rabinowitz, L. Matthey, D. Hassabis, A. Lerchner, and M. Botvinick, "Alchemy: A structured task distribution for meta-reinforcement learning," 2021.

[77] V. Volz and B. Naujoks, "Towards game-playing ai benchmarks via performance reporting standards," *2020 IEEE Conference on Games (CoG)*, pp. 764–771, 2020.

[78] L. Gisslén, A. Eakins, C. Gordillo, J. Bergdahl, and K. Tollmar, "Adversarial reinforcement learning for procedural content generation," 2021.

[79] B. Baker, I. Kanitscheider, T. Markov, Y. Wu, G. Powell, B. McGrew, and I. Mordatch, "Emergent tool use from multi-agent autocurricula," 2020.

[80] J. N. Van Rijn, B. Bischl, L. Torgo, B. Gao, V. Umaashankar, S. Fischer, P. Winter, B. Wiswedel, M. R. Berthold, and J. Vanschoren, "Openml: A collaborative science platform," in *Joint european conference on machine learning and knowledge discovery in databases*. Springer, 2013, pp. 645–649.

[81] D. Perez-Liebana, J. Liu, A. Khalifa, R. D. Gaina, J. Togelius, and S. M. Lucas, "General video game ai: A multitrack framework for evaluating agents, games, and content generation algorithms," *IEEE Transactions on Games*, vol. 11, no. 3, pp. 195–214, 2019.