Evaluation of Curriculum Learning Algorithms using Computational Creativity Inspired Metrics
Paper type: Study Paper

Benjamin Fele, Jan Babič, Senja Pollak, Martin Žnidaršič
Jožef Stefan Institute
Jamova cesta 39
1000 Ljubljana
{benjamin.fele, jan.babic, senja.pollak, martin.znidarsic}@ijs.si

Abstract

Curriculum learning, especially in robotics, is an active research field aiming to devise algorithms that speed up knowledge acquisition by proposing sequences of tasks an agent should train on. We focus on curriculum generation in reinforcement learning, where various methods are currently compared based on the agent's performance in terms of rewards on a predefined distribution of target tasks. We want to extend this singular characterization of existing algorithms by introducing metrics inspired by notions from the field of computational creativity. Namely, we introduce surprise, novelty, interestingness, and typicality that quantify various aspects of tasks stochastically proposed by the curriculum learning algorithms for the learner to train on. We model proposed tasks with Gaussian mixture models which enable their probabilistic interpretation, and use Hellinger distances between distributions and training rewards in formulation of the proposed metrics. Results are presented for eight curriculum learning algorithms showcasing differences in prioritization of various aspects of task creation and statistically different mean metric values when comparing agent's best and worst training runs. The latter finding is not only useful for analysis of existing algorithms, but potentially also provides guidance for design of future curriculum learning methods.

Introduction

The idea of introducing tasks of increasing difficulty from the perspective of a student has a history in human learning (Oudeyer, Kaplan, and Hafner 2007) and teaching (Prideaux 2003), but similar ideas in machine learning have recently gained in popularity due to their ability to reduce the number of samples necessary for training or improvement in the final performance. Reinforcement learning algorithms often prohibit real-world applications due to inability to solve complex tasks from small number of agent's interactions with the environment, which is precisely why use of curricula has been popular in that domain.

While many practical advancements have been made in this field (Portelas et al. 2020; Gupta, Mukherjee, and Najjaran 2022), not many authors provide a theoretical analysis of their work. Some ideas about how curriculum learning works have been proposed by Bengio et al. (2009), suggesting that curriculum enables learning of smoothed convex functions first, allowing to reach a dominant (and possibly global) minimum before the loss function one is optimizing grows more complex. Kroemer, Niekum, and Konidaris (2021) also add that in practice, curriculum might enable the reinforcement learning agent to collect informative non-sparse rewards and thus aid training. Xu and Tewari (2021) on the other hand argue that in addition to above optimization benefits, statistical benefits are also important — curriculum algorithms can control the amount of variance the learning algorithm experiences, leading to faster convergence.

Analysis from the perspective of optimization theory provides an useful insight into inner-workings of the current curriculum learning algorithms, but are not the only way one should try to improve understanding of their characteristics. While theoretical analyses such as those in the previous paragraph are rare, we have not come across any works in the field of reinforcement learning pertaining evaluation and comparison of existing curriculum learning methods in terms other than accumulated agent's reward in the environment. Approaching evaluation by quantifying properties of generated curricula could provide insight into which approaches for curriculum generation work better and why, in addition to providing grounds for design of future algorithms.

Metrics introduced in this paper aim to fill the aforementioned gap. We do not analyse existing algorithms in terms of optimization theory, but instead evaluate them using metrics inspired by work from the field of computational creativity. Contributions of this paper are the following:

- We introduce surprise, novelty, typicality and interestingness for evaluation of curriculum learning algorithms in the scope of reinforcement learning. The metrics evaluate the tasks proposed by the algorithms from the perspective of the learning agent and are formulated using probabilistic measures ensuring their interpretability.

- We evaluate existing state-of-the-art curriculum learning approaches using our metrics. This provides insight into differences in prioritization of different aspects of curriculum generation and possible basis for future algorithms.

Related Work

In the scope of computational creativity, many frameworks and specific metrics have been proposed for evaluating creativity in the past. Computer generated artefacts that are evaluated using computational creativity metrics can be characterized across multiple dimensions and the exact formulations depend on their context. While the definitions differ across the field, there are existing metrics such as novelty (Boden 2004; Ritchie 2007; Elgammal and Saleh 2015; Canaan et al. 2018), surprise (Maher 2010; Grace and Maher 2014; Canaan et al. 2018), quality (Ritchie 2007), value (Boden 2004; Maher 2010; Elgammal and Saleh 2015; Canaan et al. 2018) or interestingness (Schmidhuber 2009; Canaan et al. 2018), among others, that each aim to provide means for evaluation of creative artefacts.

Boden (2004) introduced criteria describing new, surprising and valuable ideas as creative. She differentiated between what newness is to one person (P-creativity) or the whole human history (H-creativity), where the former guides the definition of our metrics. Many subsequent authors formulated some of their metrics based on her definition (Maher 2010; França et al. 2016). Wiggins (2006) bases his computational creativity formulation on Boden's work, but argues that the notion of surprise is redundant. Ritchie (2007) proposed assessment of creativity through novelty and quality in addition to matching the criteria of typicality. In our work, we take his idea of the latter as a measure of how well the artefact class in question is represented by the produced items. His definition of novelty is also useful for our formulation, since it describes produced item's dissimilarity to the already known artefacts. This definition of novelty is also similar to the one in Maher (2010).

While Wiggins (2006) doesn't differentiate between novelty and unexpectedness, some authors find it useful to separate the two. Unexpectedness, or surprise, can be defined as change of the generated artefacts compared to the recent past (Maher 2010; Grace and Maher 2014). Canaan et al. (2018) also differentiate between novelty and surprise, describing the former as a dissimilarity between collection of artefacts (distance-based novelty) and the latter as a measure of how much a generated sample differs from model's expectation. Surprise is also termed learning-based novelty in their work.

Schmidhuber (2009) describes a comprehensive theory of subjective beauty, interestingness, surprise and novelty providing a formulation of creativity. He starts by defining beauty from the perspective of an agent, describing it as a signal compressible to a large degree — in other words, finding it simple — and goes on to outline interestingness as a change in the perceived beauty. This is a template for our definition of interestingness as well. The formulation from Canaan et al. (2018) is also relevant, where the measure is defined as a specific value range of novelty. They stress the idea behind Wundt curve (Wundt 1874), explaining that too little or too much novelty might lead to uninteresting artefacts. Lastly, Reehuis et al. (2013) relate interestingness to learning progress, which is related to our formulation of this metric.

In the scope of practical applications, França et al. (2016) and Varshney et al. (2019) implement novelty as a Bayesian surprise. Bayesian surprise is large whenever the impact of new data on the prior distribution is large (Franceschelli and Musolesi 2021). Novelty can also be based on simple distance measures like in Morris et al. (2012) and Maher (2010). Quality and value can on the other hand be evaluated using artificial neural networks (Morris et al. 2012), distance between nodes in a graph (Elgammal and Saleh 2015) or associations of an artwork with its description (Norton, Heath, and Ventura 2010).

Curriculum Generation in Reinforcement Learning

The use case of our proposed metrics is in the scope of automatic curriculum generation. While curricula can be used in many machine learning fields, we focus specifically on its use in reinforcement learning. Before continuing to our proposed metrics, we therefore introduce the framework within which they are utilized.

Reinforcement Learning

Reinforcement learning (RL) is defined in the scope of Markov Decision Process (MDP) denoted by a tuple $M_{RL} = (\mathcal{S}_{RL}, \mathcal{A}_{RL}, R_{RL}, \mathcal{P}_{RL}, \gamma_{RL})$, with $\mathcal{S}_{RL}$ being state space, $\mathcal{A}_{RL}$ action space, R_{RL} reward function, $\mathcal{P}_{RL}$ the state transition probabilities (i.e. environment dynamics) and γ_{RL} the discount factor prioritizing long-term planning (Sutton and Barto 2018). In a reinforcement learning algorithm, an agent performs an action $a \in \mathcal{A}$ upon experiencing state $s_t \in \mathcal{S}$, receiving a new state s_{t+1} and a reward $r = R_{RL}(s_t, a_t, s_{t+1})$ in return. The goal is to find the optimal policy π^* that maximizes the expected reward for every possible state (Sutton and Barto 2018). In addition to searching for the optimal policy π^*, solutions to auxiliary objectives have to be found: state value function $V_\pi(s_t)$ and state-action value function $Q_\pi(s_t, a_t)$, that estimate achieved reward under current policy until the end of an episode. These functions are used to search for the optimal policy, for example by greedy selection of actions leading to highest expected rewards, or as a guiding signal for policy gradient algorithms.

Curriculum Learning

As implied in the previous paragraph the agent interacts with the environment by performing actions. However, in order to learn the optimal policy, it has to randomly explore its actions and the state space. Curriculum learning (CL) aims to narrow down this exploration space by bounding it to subtasks that an agent should master first (Narvekar et al. 2020). Automatic curriculum learning, to which the methods evaluated in this paper belong, generates curricula algorithmically depending on agent's progress. The problem of finding the sequence of tasks that lead to fastest learning and best performance can be seen as finding the optimal policy for a secondary MDP $M_{CL} = (\mathcal{S}_{CL}, \mathcal{A}_{CL}, R_{CL}, \mathcal{P}_{CL}, \gamma_{CL})$, where $\mathcal{S}_{CL}$ are representations of agent's policy or knowledge, $\mathcal{A}_{CL}$ control task difficulty, R_{CL} is the reward function, e.g. the accumulated agent's reward on a target task

distribution P_{target} (Narvekar and Stone 2019). The target task distribution P_{target} entails all possible subtasks that we want our agent to generalize over. Within M_{CL}, γ_{CL} prioritizes immediate versus long-term consequences of the chosen tasks.

Above definition describes the problem to be solved in curriculum learning, but says little about actual implementations. In practice, exhaustively searching for an optimal curriculum is often intractable and can take longer than training the agent without curriculum to achieve the same performance (Narvekar and Stone 2019). Researchers thus utilize various heuristics to simplify search for good solutions. In this regard, the notion of learning progress has been a valuable measure for choosing suitable subtasks to train from in the past (Baranes and Oudeyer 2010; Moulin-Frier, Nguyen, and Oudeyer 2014; Portelas et al. 2020; Colas et al. 2019). Learning progress is defined as the (absolute) difference of collected rewards over time in some part of the task space $\mathcal{T}_{env}$. For example, RIAC (Baranes and Oudeyer 2010), Covar-GMM (Moulin-Frier, Nguyen, and Oudeyer 2014) and ALP-GMM (Portelas et al. 2020) all use such criteria in their frameworks, varying how they split the task space — across one (RIAC) or multiple (Covar-GMM and ALP-GMM) dimensions — or how they implement learning progress — through covariance between rewards and time (Covar-GMM) or absolute difference between new and old rewards (RIAC and ALP-GMM). However, not all methods rely on learning progress for selection of task parameters. ADR (Akkaya et al. 2019) expands the distribution from which the task parameters are sampled by some δ depending on whether the agent achieved sufficient performance on current subtasks. Klink et al. (2020) on the other hand propose Self-Paced curriculum learning algorithm, taking a probabilistic approach to gradually expand the task parameter distribution towards target one by weighting the loss term of their algorithm according to the agents performance. With the increasing popularity of generative adversarial networks, GoalGAN (Florensa et al. 2018) and Setter-Solver (Racaniere et al. 2019) algorithms take advantage of competition between subtask proposition and subtask suitability estimation modules. We evaluate most of the approaches outlined in this paragraph using our proposed metrics.

A common trait of the previously outlined algorithms is the mechanism by which they generate a curriculum. They all control the environment difficulty through subtask selection. This is performed by sampling task parameters for environment initialization from the task space $\mathcal{T}_{env}$. That being said, other ways of generating curricula also exist (Schaal 2006; Andrychowicz et al. 2017; Zhou et al. 2019; Zhang, Abbeel, and Pinto 2020). The way task parameters are sampled during evaluation is determined by P_{target} which is usually normally (Klink et al. 2020) or uniformly (Portelas et al. 2020) distributed.

Framing in the Context of Computational Creativity

In order to bridge the gap between curriculum learning and computational creativity, it is useful to frame the former in the context of the latter. For this reason, we turn to Wiggins (2006), who formalized Boden's (2004) model of computational creativity. He defines it as a tuple $(\mathcal{U}, \mathcal{L}, [[\cdot]], \langle\langle\cdot,\cdot,\cdot\rangle\rangle, \mathcal{R}, \mathcal{T}, \mathcal{E})$, where $\mathcal{U}$ is the universe of all concepts, $\mathcal{L}$ is the language used for expressing acceptable concept space $\mathcal{R}$, concept search algorithm $\mathcal{T}$ and concept evaluation function $\mathcal{E}$. Functions $[[\cdot]]$ and $\langle\langle\cdot,\cdot,\cdot\rangle\rangle$ are functions that apply ruleset $\mathcal{R}$ and generate concepts using $\mathcal{R}$, $\mathcal{T}$ and $\mathcal{E}$, respectively.

Within the above formulation, space of all subtasks in a curriculum learning framework can be framed as $\mathcal{U}$, while $\mathcal{R}$ bounds this space to the target task distribution or subtasks achievable by the agent. $\mathcal{T}$ can then be an algorithm modelling the curriculum, and $\mathcal{E}$ is the curriculum evaluation metric. The latter is usually the agent's reward achieved during testing, but can also be learning progress or criteria of validity, feasibility and coverage (Racaniere et al. 2019). Metrics, proposed in the next section can also belong to the set $\mathcal{E}$. Lastly, function $[[\cdot]]$ filters tasks not conforming to $\mathcal{R}$ thus bounding possible subtasks to some subspace, and function $\langle\langle\cdot,\cdot,\cdot\rangle\rangle$ generates the actual curriculum. Language $\mathcal{L}$ is in our case a set of real numbers describing specific subtasks.

Proposed Metrics

We define four metrics for evaluation of curriculum generation algorithms in relation to the agents learning to perform a specified task. For three out of four metrics we use Hellinger distance (Hellinger 1909) between various distributions to capture the changing nature of the underlying task sampling process. The distance is defined as:

$$H(f,g) = \sqrt{\frac{1}{2}\int \left(\sqrt{f(x)} - \sqrt{g(x)}\right)^2 dx}, \qquad (1)$$

where f and g are probability density functions corresponding to the two distributions we want to measure the distance between. Hellinger distance has some advantageous properties compared to other probability-based distance measures: it is for example symmetric and bound to an interval $0 \leq H(f,g) \leq 1$, which simplifies the comparison and interpretation of our results. We split the raw task parameters into windows of fixed sizes as described in the Task Parameter Prepocessing section, forming the basis for computation of the following metrics.

Surprise, novelty and typicality are all based on the aforementioned distance measure. The central idea behind surprise and novelty revolves around measuring short- and long term changes of proposed tasks, while typicality aims to capture their overlap in regards to the target task distribution. Our formulation of surprise and novelty is largely inspired by Maher (2010) by taking an idea of surprise as a measure of change in the distribution expectation compared to the recent past, and novelty describing the difference between the new and already existing data.

We define *surprise* as:

$$S_t = H(P_t, P_{t-1}), \qquad (2)$$

and *novelty* as:

$$N_t = \frac{1}{t}\sum_{k=0}^{t-1} H(P_t, P_k), \tag{3}$$

where P above denotes probability density functions of distributions fitted at specified time-points. Surprise captures changes between two sequential distributions at time t and $t-1$, while for novelty, we compute mean changes between tasks in t-th window compared to the distributions of all windows prior to it. At $t = 1$, the definition of novelty is equal to surprise, but they measure a different quantity as t grows larger. Note, that above definitions are only sensible for $t > 0$.

For *typicality*, we turn to Ritchie (2007), who formulated it as a measurement of the extent the produced item is an example of the artefact class in question. With slight abuse of this notion, we take the task distribution P_{target} that delimits the scope of problems we want our agent to be able to solve and measure its distance from distribution of tasks P_t:

$$T_t = 1 - H(P_t, P_{target}). \tag{4}$$

This formulation yields high typicality when the distance between some task parameter and target distribution is low. In our experiments, task parameters underlying the distribution P_t are bounded by P_{target}. The latter is in our case uniformly distributed and in such settings this metric measures what might be better denoted as *coverage*. However, in general, P_{target} could follow any other distribution, so we keep referring to it as *typicality* in this paper.

Lastly, the *interestingness* is defined according to the reasoning behind such metric in Schmidhuber (2009). This is the only metric that does not look at the proposed task parameter distributions themselves, but is instead computed from agent's received rewards during training. It measures the change in agent's collected rewards in a particular period of training, assuming that the underlying task parameter distribution is not exhibiting sudden changes. This way, our proposed measure is related to the change of simplicity from the perspective of an agent, as proposed by Schmidhuber (2009).

A naive approach entailing simple subtraction of rewards in the first and second halves of the window has problems with taking into account changing number signs and is not bounded to any specific interval. In order to remedy this issue we use cumulative density function to bound the interestingness values. First, let's assume that the rewards collected in a specified time period are normally distributed. Cumulative density function Φ is then defined as (Walck 2007):

$$\Phi\left(\frac{x-\mu_t}{\sigma_t}\right) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{-\frac{x^2}{2}}, \tag{5}$$

where x is an input variable, and μ_t and σ_t are in our case the mean and standard deviation of the rewards collected during training within a particular period t. After splitting that period in half, we obtain $\mu_{1/2}$ and $\mu_{2/2}$ corresponding to respective means of the two halves. These statistics are finally used to compute interestingness:

$$I_t = \phi\left(\frac{\mu_{2/2}-\mu_t}{\sigma_t}\right) - \phi\left(\frac{\mu_{1/2}-\mu_t}{\sigma_t}\right). \tag{6}$$

The use of cumulative density function ensures that each of the above terms is bound to an interval $[0, 1]$ and supports the notion that large deviations of $\mu_{1/2}$ and $\mu_{2/2}$ from the mean μ_t will result in larger absolute value of interestingness. When $\mu_{2/2} > \mu_{1/2}$ the interestingness will be positive, while in the other case its value will be negative, bounding the metric to an interval $[-1, 1]$.

Experiments

The metrics proposed above are used for evaluation of results of various curriculum learning algorithms. This section describes the benchmark setup from which the results were obtained, in addition to necessary prepossessing steps enabling treatment of task parameters guiding agent's training in a probabilistic manner.

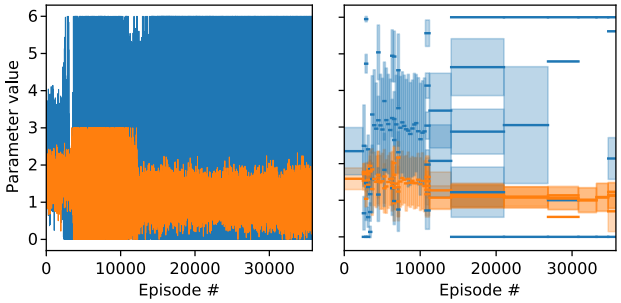

Figure 1: An example of raw task parameters (left), and GMM means and standard deviations after preprocessing (right) for every episode of training. Values of obstacle spacings are in blue and values for their heights are in orange.

Task Parameter Prepocessing

As outlined in the Curriculum Learning section, curriculum learning methods evaluated using proposed metrics operate by proposing task parameters used for environment initialization and thus controlling its difficulty. As the agent progresses, various approaches estimate distributions from which these initialization parameters are sampled. Some algorithms sample from uniform (Akkaya et al. 2019) or (multiple) Gaussian (Moulin-Frier, Nguyen, and Oudeyer 2014; Portelas et al. 2020; Klink et al. 2020) distributions, while others might use a more complex sampling scheme (Florensa et al. 2018; Racaniere et al. 2019). To enable computation of our metrics, we model the distributions of sampled task parameters during agent's learning using Gaussian mixture models (GMMs). Without knowledge of the real underlying distributions, this provides a reasonable estimate and a basis for probabilistic interpretation of the task parameter sampling process. This way, the process is not oversimplified as it would be if the normal distribution was assumed across all evaluated task distributions. This allows our use of Hellinger distance as a measure of change between two algorithms and subsequent analysis.

To capture the underlying distribution from which task parameters are sampled at different time-points during training, we split them into smaller windows w_t ($t \in [0, 40]$).

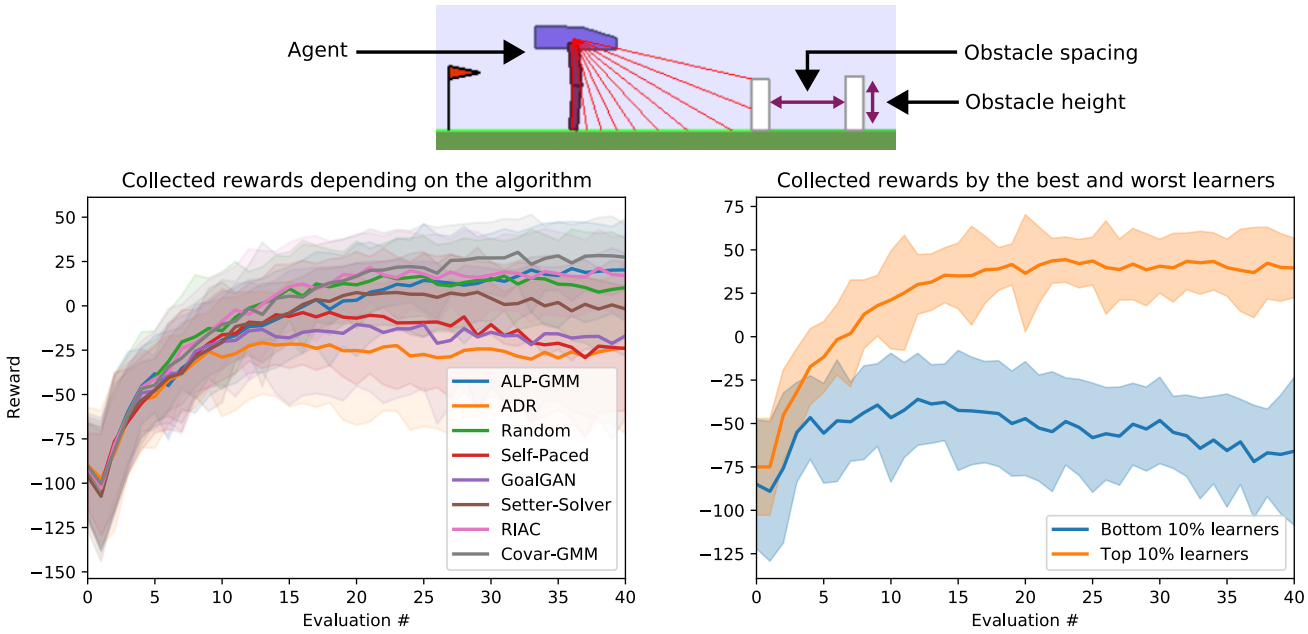

Figure 2: Top: an image of the agent in the environment and the two parameters controlling its difficulty. Bottom left: rewards collected during periodic testing while training. Bottom right: testing rewards for the highest and lowest $10\ \%$ of learners.

An example of raw task parameter data is seen in Figure 1 (left) and respective splits with fitted GMMs are seen in Figure 1 (right). Figures show the evolution of two task parameters used for controlling the difficulty of the agent moving through the environment. These two parameters control obstacle heights and spacings and are visualized in Figure 2 (top).

Each window size in Figure 1 corresponds to the time it took an agent to perform 500000 steps, resulting in a varying number of episodes in each window. Since the task parameters are sampled per episode, this also results in windows apparently varying in size in Figure 1 (right). GMMs with up to 5 components are fitted on the task parameters in each window and the one with the largest Akaike information criterion (Sakamoto, Ishiguro, and Kitagawa 1986) is kept for further analysis. This is a metric that quantifies how well the GMMs fit the underlying data.

Experimental Setup

TeachMyAgent benchmark (Romac et al. 2021) provides a framework for evaluation of various curriculum and reinforcement learning algorithms in two environments in multiple training configurations. It tests 7 curriculum learning algorithms in addition to the random baseline. The curriculum generation algorithms available in the TeachMyAgent benchmark and also used for evaluation of our metrics are ALP-GMM (Portelas et al. 2020), ADR (Akkaya et al. 2019), Self-Paced (Klink et al. 2020), GoalGAN (Florensa et al. 2018), Setter-Solver (Racaniere et al. 2019), RIAC (Baranes and Oudeyer 2010) and Covar-GMM (Moulin-Frier, Nguyen, and Oudeyer 2014). They are already briefly outlined in Curriculum Learning section. We take their results from the StumpTracks environment (Portelas et al. 2020), which is illustrated in the Figure 2 (top). The environment consists of an agent being tasked to learn to walk in environments with varying obstacle heights and spacings.

TeachMyAgent authors also introduced multiple agent embodiments and training configurations. We evaluate our metrics on results using a bipedal walker with an uniform target task distribution P_{target} bounded to the interval $[0, 3]$ for obstacle height and $[0, 6]$ for obstacle spacing. Data used in our analysis entails task parameters used for initialization of environments for each episode and respective accumulated rewards in addition to cumulative rewards obtained during each testing period.

As mentioned before, the agent's performance is periodically tested in the environment initialized with the parameters from the target task distribution P_{target}. Figure 2 shows the average performance during testing phases of the learning agent depending on curriculum learning algorithm used (bottom left) or its overall performance (bottom right). This illustrates that the differences between each curriculum generation method are in this case relatively small, but best and worst performances vary more substantially. Notice, how some algorithms in Figure 2 (bottom left) perform worse than the random baseline; this is in line with the results presented by Romac et al. (2021).

Each curriculum learning method in Figure 2 and the Results section was evaluated on results obtained from 64 experiment runs each using a different random seed. The points at which the agent is tested serve for splitting the task initialization parameters proposed by the curriculum learning algorithms into smaller windows. The best and worst

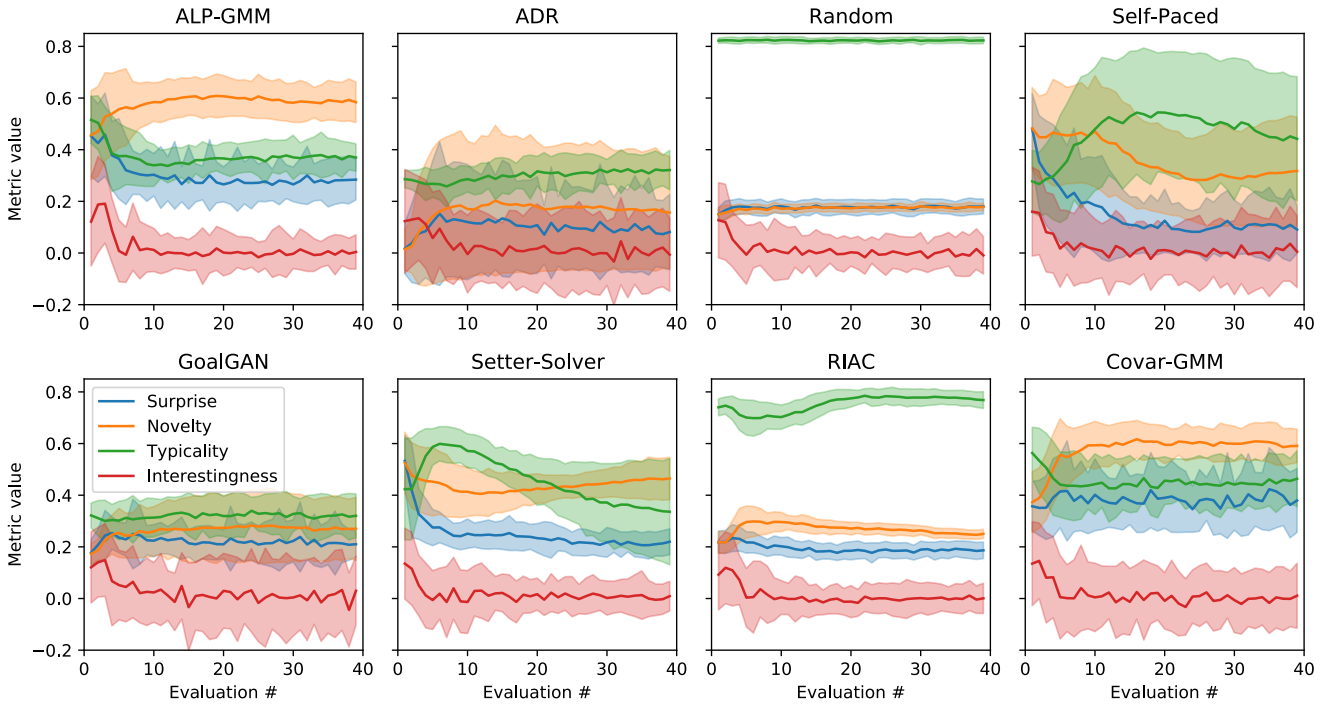

Figure 3: Metric values through agent training for all tested algorithms. It can be seen that every algorithm yields a different progression of metrics.

10 % of experiment runs used for visualization in Figures 2 and 5 are obtained by taking the distribution of mean collected test rewards and extracting bottom and top performers. This results in 52 samples in each group. We use Welch's t-test with $\alpha = 0.05$ with Bonferroni correction to evaluate statistically significant differences in metric values. For computation of Hellinger distances we use Monte-Carlo integration with 1000 samples, yielding a reasonably low error.

Results

Our metrics can be evaluated from two viewpoints presented in the following subsections. One perspective takes evolution of the metric values over various curriculum learning algorithms and thus provides the means for their comparison, while the other concentrates on agent's performance regardless of the underlying curriculum generator, highlighting changes between better and worse training runs.

Comparison of Curriculum Learning Algorithms

Figure 3 shows resulting values of our metrics depending on the algorithm they were evaluated on. At first glance, there are considerable differences between algorithms giving each of them a particular *silhouette*. Metrics usually vary the most at the beginning of training, and later stabilize at some value. We speculate their fast convergence is a consequence of the fact that over time the subset of tasks to be mastered gets smaller which is mirrored in the differences in their distributions. Self-Paced and Setter-Solver curriculum generation algorithms are an exception to this rule where novelty and typicality metrics don't converge like described. Results obtained from training with random curriculum show metric values when task parameters are uniformly distributed throughout training. Typicality, measuring the similarity between proposed and target task distributions, is in this case consistently the highest, but not equal to 1 due to inability of Gaussian mixture models to accurately capture uniform target task distribution. This is also the reason why the typicality for random curriculum is consistently high — there, the subtasks are sampled uniformly from the target task distribution.

Regardless of the algorithm they are evaluating, novelty and surprise hold similar values at the beginning of training, and grow more dissimilar later. Since the distribution underlying random curriculum generation doesn't change in the course of training, its surprise and novelty hold consistent values in the lower end of the spectrum. Interestingly, as seen in the Figure 4, ADR holds similar or lower values in this metrics and Self-Paced curriculum learning algorithm starts with a relatively high value of surprise but converges to around 0.1. In general, RIAC seems to hold values close to the ones obtained by the random curriculum and also leads to similar agent's performance as seen in the Figure 2. ALP-GMM and Covar-GMM show the highest novelty of the proposed subtasks, and generally hold similar metric values.

Comparing standard deviations in Figure 3, some algorithms (ALP-GMM, Setter-Solver, RIAC and randomly generated curriculum) exhibit smaller diversities of the com-

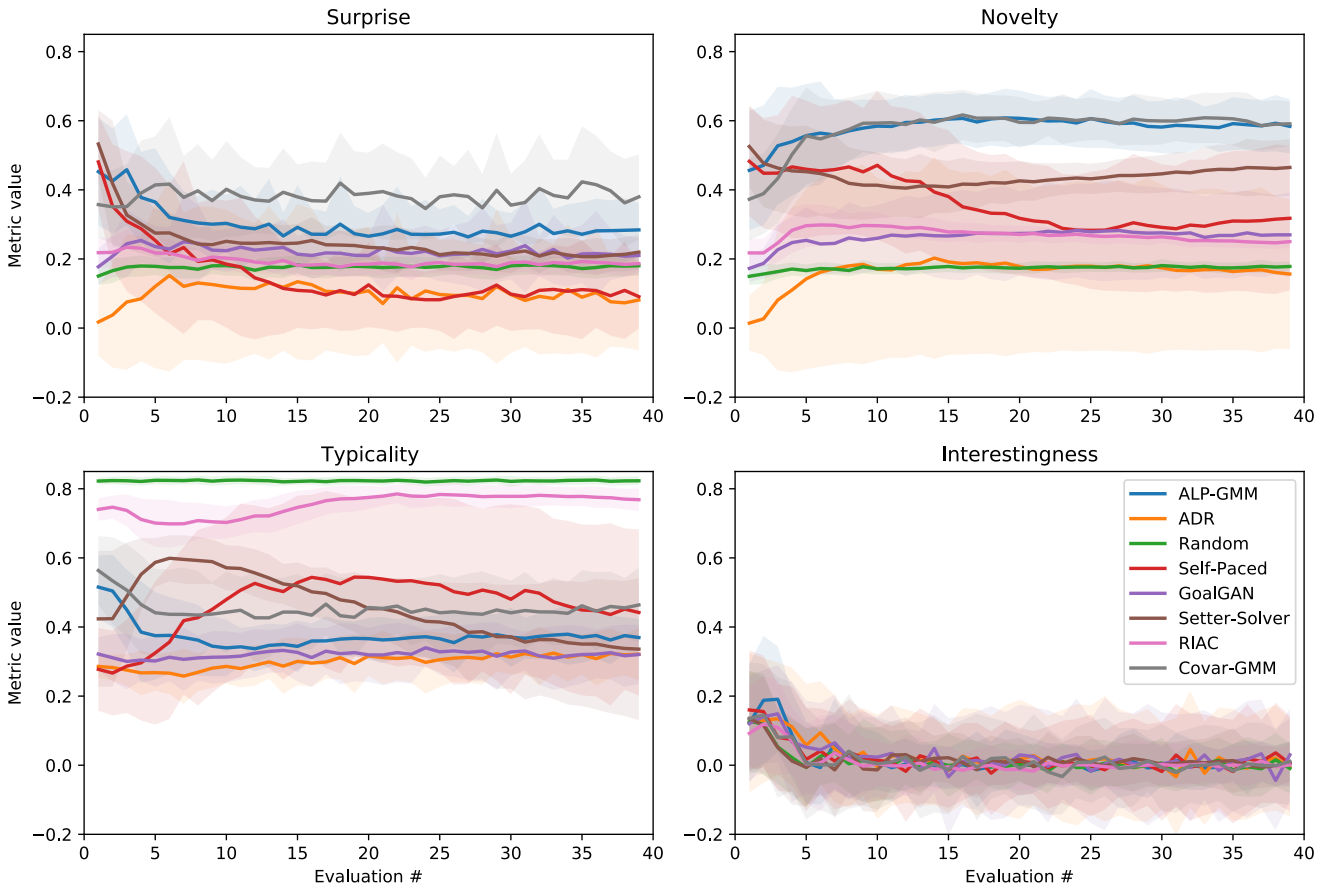

Figure 4: Results measuring tested algorithms shown by the metrics used. It can be seen that interestingness has the least variation across all curriculum learning algorithms.

puted measures, while others are more variable across experiment runs. Overall, interestingness seems to change between evaluated algorithms the least, which is clearly seen in Figure 4. From this perspective, it is not as useful for evaluation of curriculum learning algorithms as the other metrics. Most variability in this metric comes at the beginning of learning, when the agent's knowledge consistently starts improving. When comparing metrics other than interestingness between each other, it can be seen that they take distinctly different values and are in this sense not redundant.

Best- and Worst-performing Experimental Runs

As the differences between performances depending on the chosen algorithm are relatively small, this is not a suitable viewpoint for evaluation of curricula characterized by our metrics in regards to agent's performance. As shown in Figure 5, all metrics except interestingness consistently exhibit statistically different means when evaluated in regards to the best and worst training runs: $2.017 \times 10^{-16} < p < 4.306 \times 10^{-4}$ for surprise, $2.36 \times 10^{-15} < p < 4.728 \times 10^{-4}$ for novelty and $6.272 \times 10^{-11} < p < 4.584 \times 10^{-4}$ for typicality. Interestingness is not consistently significant with p-values $1.693 \times 10^{-2} < p < 36.056$. Surprise, novelty and typicality exhibit higher values with better learners. Visible trend in evolution of surprise and typicality is not obvious, but it is more clearly present when measuring novelty. Namely, with better performing learners it starts at a lower value and stabilizes around 0.45, a trend not present with evaluation of the bottom $10\ \%$ of the learners.

The lack of perceived trend might on one hand come from large diversity of surprise, novelty and typicality across training runs, also visible in relatively large variability of these metrics across algorithms in Figure 4. On the other hand, the metrics in general seem to stabilize at some value, which could also provide an explanation for the lack of trends on the graphs.

Interestingness is an exception in regards to patterns described in the above paragraphs. As was already seen in Figure 4, the metric shows low variability between curriculum generation algorithms, and is also not consistently significantly different in Figure 5. At the beginning of training, the interestingness stays similar across the two groups, but later settles at lower values for both groups. The values of better performing learners turn out to stabilize lower and have smaller standard deviation compared to the worse performing ones.

Since interestingness measures agent's progress during training, larger values correspond to faster learning of the

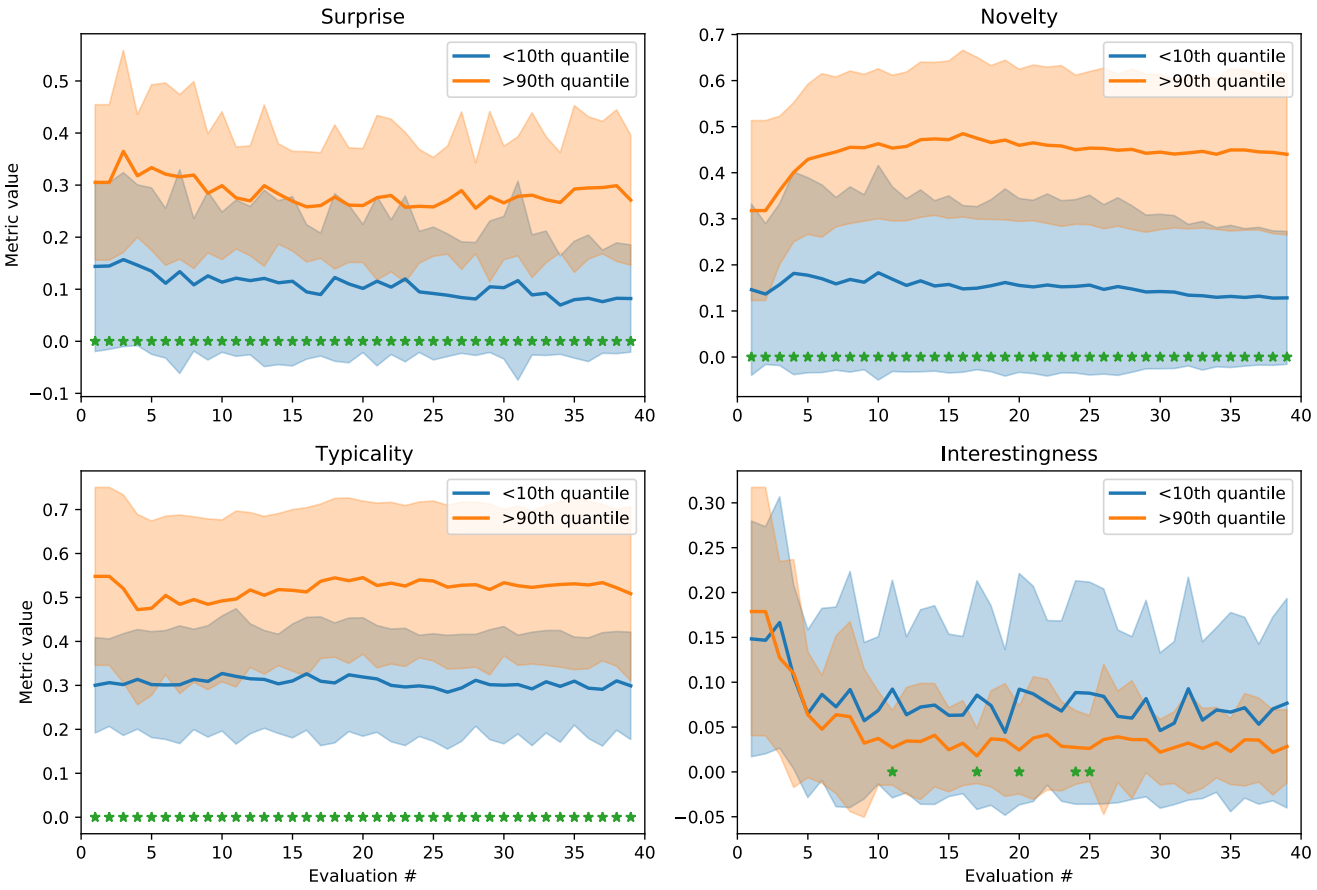

Figure 5: Mean metric values and standard deviations during agent's training for best and worst 10 % performing learners. Stars at the bottom of plots denote statistically significant changes.

tasks given in a particular time-period. This shows that fast improvements on specific subtasks during training don't translate into better performing agents on the target task distribution used during evaluation — our findings show that the opposite is true. This would imply that the tasks, even though they are labeled as more interesting by our metric, are perhaps less relevant for agent's progress on the target task.

Conclusion

This paper formulates metrics inspired by notions from the field of computational creativity and uses them for evaluation of curriculum learning algorithms. Results show that our metrics exhibit informative characteristics from two points of view: (i) as the means to differentiate and characterize curricula generated by different algorithms and (ii) distinguish more successful training runs from the less successful ones.

The differences between best and worst performing learners highlight that higher values of surprise, novelty and typicality, and lower values of interestingness, are generally beneficial for learning and its overall performance. Higher surprise signifies that more sudden changes in task distributions are beneficial, which also holds for coverage of the target task distribution implied by results for novelty and typicality. Interestingness results are less interpretative in our case, but suggest that proposing tasks resulting in larger values of this metric doesn't translate into better overall performance.

The property of interestingness to unsuccessfully capture what it was intended for is one of the shortcomings of our work. Furthermore, more tests should be conducted to determine how the proposed metrics correlate with actual underlying tasks that we are trying to model; notice how our approach is not concerned with mechanisms behind curriculum generation, subtask order or learner choice. Some of these issues could be remedied by conducting more detailed analysis using all results provided by the TeachMyAgent benchmark, and also obtaining some of our own.

Shortcomings aside, we want to stress that the results still provide a good starting point for design of future curriculum learning algorithm; for example, the selection of subtasks could be guided by balancing the values of the proposed metrics. This way, the utilization of our metrics could serve as a guiding criteria for determination of suitable subtasks that the agent should train on and contribute to the future of algorithmic curriculum generation.

Author Contributions

B. Fele carried out the analysis and wrote the core of the paper. B. Fele, S. Pollak and M. Žnidaršič conceptualized the metrics presented in this paper. B. Fele, S. Pollak, M. Žnidaršič and J. Babič worked on amending and correcting the text of the paper.

Acknowledgments

This work was supported by the Slovenian Research Agency (research core funding no. P2-0076 and P2-0103).

References

Akkaya, I.; Andrychowicz, M.; Chociej, M.; Litwin, M.; McGrew, B.; Petron, A.; Paino, A.; Plappert, M.; Powell, G.; Ribas, R.; et al. 2019. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*.

Andrychowicz, M.; Wolski, F.; Ray, A.; Schneider, J.; Fong, R.; Welinder, P.; McGrew, B.; Tobin, J.; Abbeel, P.; and Zaremba, W. 2017. Hindsight experience replay. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 5055–5065.

Baranes, A., and Oudeyer, P.-Y. 2010. Intrinsically motivated goal exploration for active motor learning in robots: A case study. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1766–1773. IEEE.

Bengio, Y.; Louradour, J.; Collobert, R.; and Weston, J. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, 41–48.

Boden, M. A. 2004. *The creative mind: Myths and mechanisms*. Routledge.

Canaan, R.; Menzel, S.; Togelius, J.; and Nealen, A. 2018. Towards game-based metrics for computational co-creativity. In *2018 IEEE conference on Computational Intelligence and Games (CIG)*, 1–8. IEEE.

Colas, C.; Fournier, P.; Chetouani, M.; Sigaud, O.; and Oudeyer, P.-Y. 2019. Curious: intrinsically motivated modular multi-goal reinforcement learning. In *International conference on machine learning*, 1331–1340. PMLR.

Elgammal, A., and Saleh, B. 2015. Quantifying creativity in art networks. In *Proceedings of the Sixth International Conference on Computational Creativity June*, 39.

Florensa, C.; Held, D.; Geng, X.; and Abbeel, P. 2018. Automatic goal generation for reinforcement learning agents. In *International conference on machine learning*, 1515–1528. PMLR.

França, C.; Góes, L. F. W.; Amorim, A.; Rocha, R.; and Da Silva, A. R. 2016. Regent-dependent creativity: A domain independent metric for the assessment of creative artifacts. In *Proceedings of the Seventh International Conference on Computational Creativity*, 68–75. Citeseer.

Franceschelli, G., and Musolesi, M. 2021. Creativity and machine learning: A survey. *arXiv preprint arXiv:2104.02726*.

Grace, K., and Maher, M. L. 2014. What to expect when you're expecting: The role of unexpectedness in computationally evaluating creativity. In *ICCC*, 120–128. Ljubljana.

Gupta, K.; Mukherjee, D.; and Najjaran, H. 2022. Extending the capabilities of reinforcement learning through curriculum: A review of methods and applications. *SN Computer Science* 3(1):1–18.

Hellinger, E. 1909. Neue begründung der theorie quadratischer formen von unendlichvielen veränderlichen. *Journal für die reine und angewandte Mathematik* 1909(136):210–271.

Klink, P.; D' Eramo, C.; Peters, J. R.; and Pajarinen, J. 2020. Self-paced deep reinforcement learning. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M. F.; and Lin, H., eds., *Advances in Neural Information Processing Systems*, volume 33, 9216–9227. Curran Associates, Inc.

Kroemer, O.; Niekum, S.; and Konidaris, G. 2021. A review of robot learning for manipulation: Challenges, representations, and algorithms. *J. Mach. Learn. Res.* 22:30–1.

Maher, M. L. 2010. Evaluating creativity in humans, computers, and collectively intelligent systems. In *Proceedings of the 1st DESIRE Network Conference on Creativity and Innovation in Design*, 22–28. Citeseer.

Morris, R. G.; Burton, S. H.; Bodily, P.; and Ventura, D. 2012. Soup over bean of pure joy: Culinary ruminations of an artificial chef. In *ICCC*, 119–125.

Moulin-Frier, C.; Nguyen, S. M.; and Oudeyer, P.-Y. 2014. Self-organization of early vocal development in infants and machines: the role of intrinsic motivation. *Frontiers in psychology* 4:1006.

Narvekar, S., and Stone, P. 2019. Learning curriculum policies for reinforcement learning. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, 25–33.

Narvekar, S.; Peng, B.; Leonetti, M.; Sinapov, J.; Taylor, M. E.; and Stone, P. 2020. Curriculum learning for reinforcement learning domains: A framework and survey. *arXiv preprint arXiv:2003.04960*.

Norton, D.; Heath, D.; and Ventura, D. 2010. Establishing appreciation in a creative system. In *ICCC*, 26–35.

Oudeyer, P.-Y.; Kaplan, F.; and Hafner, V. V. 2007. Intrinsic motivation systems for autonomous mental development. *IEEE transactions on evolutionary computation* 11(2):265–286.

Portelas, R.; Colas, C.; Hofmann, K.; and Oudeyer, P.-Y. 2020. Teacher algorithms for curriculum learning of deep rl in continuously parameterized environments. In *Conference on Robot Learning*, 835–853. PMLR.

Prideaux, D. 2003. Curriculum design. *Bmj* 326(7383):268–270.

Racaniere, S.; Lampinen, A.; Santoro, A.; Reichert, D.; Firoiu, V.; and Lillicrap, T. 2019. Automated curriculum generation through setter-solver interactions. In *International Conference on Learning Representations*.

Reehuis, E.; Olhofer, M.; Emmerich, M.; Sendhoff, B.; and Bäck, T. 2013. Novelty and interestingness measures for design-space exploration. In *Proceedings of the 15th annual conference on Genetic and evolutionary computation*, 1541–1548.

Ritchie, G. 2007. Some empirical criteria for attributing creativity to a computer program. *Minds and Machines* 17(1):67–99.

Romac, C.; Portelas, R.; Hofmann, K.; and Oudeyer, P.-Y. 2021. Teachmyagent: a benchmark for automatic curriculum learning in deep rl. In *International Conference on Machine Learning*, 9052–9063. PMLR.

Sakamoto, Y.; Ishiguro, M.; and Kitagawa, G. 1986. Akaike information criterion statistics. *Dordrecht, The Netherlands: D. Reidel* 81(10.5555):26853.

Schaal, S. 2006. Dynamic movement primitives-a framework for motor control in humans and humanoid robotics. In *Adaptive motion of animals and machines*. Springer. 261–280.

Schmidhuber, J. 2009. Simple algorithmic theory of subjective beauty, novelty, surprise, interestingness, attention, curiosity, creativity, art, science, music, jokes. *Journal of SICE* 48(1).

Sutton, R. S., and Barto, A. G. 2018. *Reinforcement learning: An introduction*. MIT press.

Varshney, L. R.; Pinel, F.; Varshney, K. R.; Bhattacharjya, D.; Schörgendorfer, A.; and Chee, Y.-M. 2019. A big data approach to computational creativity: The curious case of chef watson. *IBM Journal of Research and Development* 63(1):7–1.

Walck, C. 2007. Hand-book on statistical distributions for experimentalists. *University of Stockholm* 10:112–113.

Wiggins, G. A. 2006. A preliminary framework for description, analysis and comparison of creative systems. *Knowledge-Based Systems* 19(7):449–458.

Wundt, W. M. 1874. *Grundzüge der physiologischen Psychologie*, volume 1. Wilhelm Engelmann.

Xu, Z., and Tewari, A. 2021. On the statistical benefits of curriculum learning. *arXiv preprint arXiv:2111.07126*.

Zhang, Y.; Abbeel, P.; and Pinto, L. 2020. Automatic curriculum learning through value disagreement. *Advances in Neural Information Processing Systems* 33.

Zhou, B.; Zeng, H.; Wang, F.; Li, Y.; and Tian, H. 2019. Efficient and robust reinforcement learning with uncertainty-based value expansion. *arXiv preprint arXiv:1912.05328*.