Work-In-Progress: Enhanced Learning from Multiple Demonstrations with a Two-level Structured Approach

Su Zhang
Washington State University
su.zhang2@wsu.edu

Matthew E. Taylor
Washington State University
taylorm@eecs.wsu.edu

ABSTRACT

Learning from demonstrations (LfD) has been successfully leveraged to speed up reinforcement learning techniques. This paper investigates how to mitigate, or at least reduce, the impact of noisy or poor demonstrations. In particular, we consider how heterogeneous demonstrators may provide inconsistent and/or even conflicting information, and noise may occur in otherwise high-quality demonstrations, both of which can significantly affect the policy generalization and hurt learning performance. To address the above challenge, this paper develops a Flexible Two-level Structured Approach (FTSA) inspired by the existing Human Agent Transfer algorithm and the multi-armed contextual bandit problem. Our FTSA approach can mitigate the effects of bad demonstrations while retaining the benefits provided by good ones. Our preliminary experimental results in the Mario domain show that this approach holds promise and may be able to successfully leverage demonstrations of different quality.

KEYWORDS

Learning from demonstration; Multi-Armed Bandit; Reinforcement Learning; Transfer Learning

1 INTRODUCTION

Reinforcement learning [18] (RL) has had many successes in sequential decision tasks, where an agent learns to maximize a real-valued reward. However, learning tabula rasa can be slow, particularly in difficult domains. Learning from demonstration [1] (LfD) is an alternative formulation where an agent typically learns to mimic a human demonstrator. Following this general idea, a rich set of LfD techniques have been developed. However, the majority of such techniques are limited by the demonstrator's performance: an LfD agent's goal is typically to mimic the human, while an RL agent's goal is to maximize total reward.

In this paper, we are concerned with how to best combine LfD with RL so that we can reap the benefits of fast learning while not being limited by the demonstrator's ability. In particular, we focus on cases of heterogeneous demonstrations. One could consider combining demonstrations from sources with different average performances, or one demonstrator with a high variance in performance. In both cases, we would like to maximize how much we can learn from the demonstrators while minimizing how much poor demonstrations hurt the learner.

Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), M. Dastani, G. Sukthankar, E. Andre, S. Koenig (eds.), July 2018, Stockholm, Sweden

https://doi.org/doi

In dealing with this problem, several straightforward solutions have been developed, including removing unnecessary or inefficient examples from the demonstration [9], requesting additional clarifications [6], etc. However, those ideas try to eliminate the effects of heterogeneous and noisy demonstrations by removing them instead of trying to extract useful information and learn from them. The challenge of how to effectively learn from heterogeneous demonstrations, making use of the beneficial ones while avoiding the influence of bad ones, still exists.

To address this challenge, we draw inspiration from the multi-armed contextual bandit problem and propose a flexible two-level structure based on the probability policy reuse scheme of HAT (detailed in Section 2.2): level 0 uses multiple classifiers to summarize demonstrations and level 1 takes advice from the low-level classifiers and combines their opinions with the embedded decision algorithm (e.g., majority voting).

Our preliminary experimental results in Mario show that the two-level structure can improve the overall performance (total reward) and initial performance (jumpstart), while minimizing the effects of bad demonstrations (relative to the existing HAT algorithm).

2 BACKGROUND AND RELATED WORKS

This section briefly introduces the techniques needed to understand this paper and related works in RL, LfD, and contextual multi-armed bandit problem.

2.1 Reinforcement Learning

Reinforcement learning is an approach for an agent to learn from experience through the interaction with the environment [18]. Markov Decision Processes (MDPs) are often used to formalize RL tasks, where A is the set of available actions, S is the set of states, the reward function $R : S \times A \rightarrow \mathbb{R}$ is to be maximized, and the transition function $T : S \times A \rightarrow S$ defines how the state changes over time. Action-value function $Q : S \times A \rightarrow \mathbb{R}$ provides the estimated value of state-action pairs. At each time step, the agent will select an action to execute and receive the states and reward determined by the environment. The selection of actions follows a policy π, which could be approximated with the estimated value of a certain state-action pair Q(s, a). During the updating of Q(s, a), the agent gradually improves the policy and maximizes the expected reward. This could be done with different algorithms like SARSA, Q-learning [21] (which is used in this paper), etc.

2.2 Learning from Demonstration

LfD is a useful technique to deal with the poor initial performance of *tabula rasa* agents [1]. Demonstrations are usually trajectories of state-action pairs performed by human (or, sometimes, other agents). With demonstrations recorded from same or similar tasks,

the agent could transfer the learned feature of neural networks [22], learn heuristics for reward shaping [4], construct skill trees with experience replay [12], etc.

Among those related works, the Human Agent Transfer [20] (HAT) algorithm first summarizes the policies with a rule-based learner (e.g., a Decision Tree) then uses the rule transfer [19] technique of transfer learning in a probabilistic policy reuse [8] (PPR) manner, to speed up and improve the performance of the RL agent. This paper also adopts the PPR technique, while focusing on making use of multiple demonstrations regardless of their qualities.

There are also works that follow the Reinforcement Learning with Expert Demonstrations (RLED) framework, which try to extract optimal policies by directly applying iterations on the demonstration data [5, 11, 16], identify relevant task features through demonstrations and apply RL with the abstract state space [7], etc. Different from those solutions, our approach focuses on guiding the learning process using policies generalized from demonstrations instead of explicitly using the demonstration examples as an additional source for updating value estimates.

2.3 Learning from Multiple Experts

Online learning with expert advice could be viewed as contextual multi-armed bandit problem [14], where the goal of the agent is to compete against the best expert (that has the highest expected reward). At each time step, an expert will generate advice about the arm selection based on the current context, and the agent will build a strategy of pulling the arm, considering all the given advice, and dynamically update its belief of each expert [23]. The most relevant work is the Exponential-weight Algorithm for Exploration and Exploitation using Expert advice (EXP4) algorithm [2], and similar ideas are also adopted in this paper. Additionally, such modeling could be applied to applications like recommendation [13], selecting state machine policies for robotic systems [15], etc.

3 A FLEXIBLE TWO-LEVEL STRUCTURED APPROACH TO LEARNING FROM MULTIPLE DEMONSTRATIONS

In this section, we propose a Flexible Two-level Structured Approach (FTSA) to address the problem of combining multiple demonstrations. This approach integrates the multi-armed bandit approach with the probability policy reuse scheme of HAT and allows the agent to take advantage of multiple demonstrations regardless of their quality.

3.1 Problem Statement

We formalize the problem of learning from multiple demonstrations as a contextual multi-armed bandit problem with following notation:

- K number of actions
- N number of classifiers/experts/oracles, where each is trained on a demonstration
- D demonstration data set, each d_i is a recording of an episode, and contains state/action pairs
- t time step, where the agent will select actions and execute at each t
- ξ policy advice vector: $\xi^i_{t,j}$ is demonstrator i's recommended probability of choosing arm (action) j at round t
- r reward vector with dimension of K, all the elements in r are zero, except the one corresponds to the selected arm (action)
- x_t context vector, used by classifiers to generate advice, could be the state vector from task environment, etc.

Generally, each classifier is trained with a demonstration and at round t, classifier i will give its estimation of probability $\xi^i_{t,j}$ of selecting action (arm) j based on the context vector x_t, the agent will then decide to trust and follow a classifier's advice or not, pick one action (arm), and continue the RL steps.[1]

3.2 Method Description

The entire procedure is built on the framework of HAT, and the multi-armed bandit selection could be captured with this two-level structure:

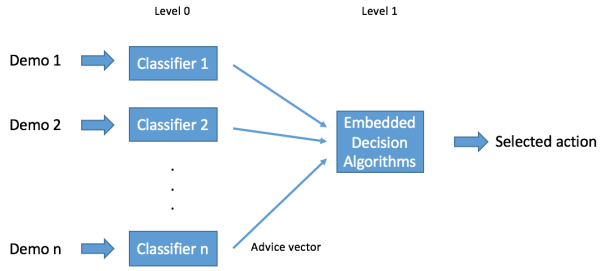

Figure 1: Two-level Structure of bandit selection

With the HAT framework, the agent will adopt the advice from classifiers with a certain probability ϕ (which decays over time). Each level-0 classifier is trained with one demonstration,[2] and will generate advice vector according to different context vectors; level-1 could embed with any algorithms to combine all the advice from level-0 classifiers and provide the final suggestion for the agent to select. With this two-level structure, the agent could then make use of knowledge from multiple different demonstrations regardless of their quality. Various algorithms and schemes could be adopted into level-1 as the decision component. In this work, we pick majority voting, Exponential Weighted Algorithm (EWA), and a meta-classifier as examples to show how this approach works.

3.2.1 Majority Vote. When combining the advice with majority voting, the entire structure will become similar to the Bagging ensemble method [3]. The difference is that the training data are not generated with bootstrap sampling, but directly use the data from different demonstrators to build each base classifier. And this voting scheme may not be able to reduce the "variance," but only to combine multiple advice and get a final output suggestion. Since each classifier is treated equally, this method works well when the quality of most demonstrations are good or similar to each

[1] It is possible the expected regret for each classifier could be defined and calculated, but we leave this to future work.
[2] Future work will consider training each classifier per demonstrator.

Algorithm 1: Flexible Two-level Structured Approach for LfD

Input: Demonstration data set $D = \{d_0; d_1; :::; d_n\}$, reuse probability Φ_0, decay rate Φ_D

```
Φ ← Φ_0
for demonstration d_i ∈ D do
    Train classifier c_i with d_i          . Build level-0 classifiers
end
Initialize level-1 algorithms
. Update weights for each classifier, build a meta-classifier, etc.
for each episode do
    Initialize state s_0
    for step t do
        Get state vector s_t, reward r_t
        a ← φ
        if rand() ≤ Φ then
            Construct context vector x_t with s_t
            for each classifier c_i do
                Get ξ_t^i with x_t
            end
            Apply level-1 algorithms to select action a
            . majority vote, value estimation, classification, etc.
        else
            if rand() ≤ ε then
                a ← random action
            else
                a ← argmax_a Q
        Execute action a
        Update Q-value (with SARSA, Q-Learning, etc.)
        Φ ← Φ * Φ_D
    end
end
```

other, as the advice from those few classifiers trained with bad demonstrations will be ignored.

3.2.2 Exponential Weighted Algorithm (EWA). When combining the advice with EWA, the structure will be similar to the EXP3 and EXP4 bandit algorithms. In contrast to EXP3 or EXP4, the exploration/exploitation balancing will not be handled by EWA itself, but by the HAT framework. A weight will be initialized equally and assigned to each classifier, and then the algorithm will exponentially decay the weights of classifiers based on their advice during iterations. The update of the weights could use the reward from the environment or other loss functions. In each round, the agent could estimate the reliability of each classifier based on the advice they give and choose which classifier to listen to. In this way, the EWA method could handle the blending of good and bad demonstrations.

3.2.3 Meta-Classifier. We could also build a meta-classifier on level-1 based on the advice of level-0, similar to the Stacking (stacked generalization) ensemble method. The difference is that the stacking method is built in a cross-validation manner, and here the level-1 classifier is trained with the same context of demonstration data but replaces the original label with predictions of level-0 classifiers. The level-1 classifier could use either the same base classification method as level-0 does, or different methods.

4 EVALUATION

This section discusses the empirical evaluation of the proposed methods.

4.1 Experimental Setting

Our approach is evaluated with the Mario simulator released by Karakovskiy [10], based on the Nintendo's platform game Super Mario Bros. The agent is trained to get as high a score as possible. The state is composed of 27 state variables and includes both numeric and Boolean variables; there are $3{:}65 \times 10^{10}$ different discrete states in total (although many are unreachable in practice). The agent can choose 12 different actions with the combinations of three sets: {left, right, no action}, {jump, do not jump}, {run, do not run}. The agent will receive +10 for jumping on an enemy, +58 for eating a mushroom, +64 for eating a fire-flower, +16 for collecting a coin, +24 for finding a hidden block, +1024 for clearing the level, -42 for getting hurt by an enemy, and -512 for dying [17].

To investigate whether our approach could learn reasonable performance with multiple demonstrations, we conduct experiments with 10 good demonstrations (23,259 state/action example pairs in total), 10 good demonstrations and 20 bad demonstrations (48,852 examples in total), and 20 bad demonstrations (25,594 examples in total).[3] We use the J48 decision tree to build base classifiers in level-0 and the following algorithms are embedded in level-1:

(1) Majority vote
(2) Exponential Weighted Algorithm, where $\eta = 0{:}001$, iterate for 100 rounds;
(3) Meta-classifier, using J48 as level-1 classifier.

All "good" demonstrations are recorded with a well-trained Q-learning agent playing Mario, with an average score of 3,408.4; bad demonstrations are recorded with a simple agent, the average score is -267.6. The baseline is the Q-learning agent without any prior knowledge. We also adopt the HAT agent with J48 as a benchmark method (which uses a merged data file of multiple demonstrations as input).

For the evaluation metrics, we consider 1) *Jumpstart*, the improvements of initial performance vs. the benchmark agent; and 2) *Total Reward*, the accumulated reward achieved by an agent (i.e., the area under the learning curve).

4.2 Results and Discussion

To present the results of the experiments, we plot the learning curves with the averaged score over 10 runs of 50,000 episodes with a sliding window of size 3,000.

According to Figure 2, when using 10 good demonstrations, the performance of the proposed approach is much better than the baseline, and could converge to a similar performance level of HAT; at the beginning stage, HAT has the highest jumpstart, and FTSA with majority vote has the lowest jumpstart, while the jumpstarts

[3] We choose the combination of 10 good demonstrations and 20 bad demonstrations to present approximately the same number of examples from good and bad demonstrations.

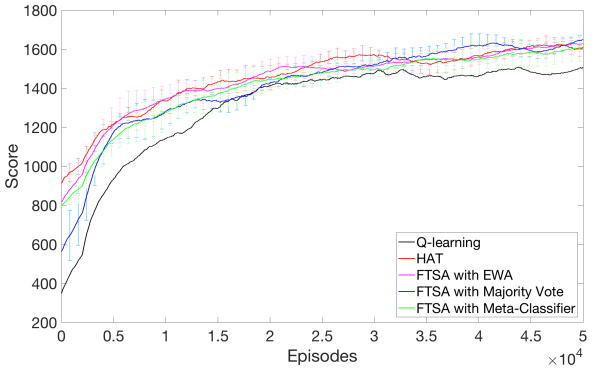

Figure 2: With 10 good demos

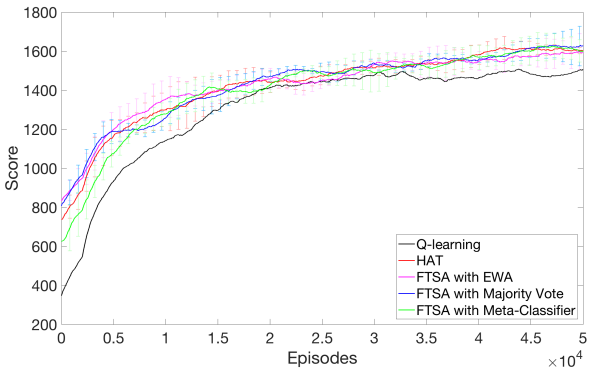

Figure 3: With 10 good demos and 20 bad demos

of using EWA and the meta-classifier lie between. However, even with a low jumpstart, the performance of FTSA with all different methods could catch up with the performance of the HAT within 5,000 episodes.

Figure 3 shows the performance of training an agent with the proposed approach using 10 good demonstrations and 20 bad demonstrations. With the blending of good and bad demonstrations, both HAT and FTSA with all the methods could outperform the baseline; the jumpstart of HAT is lower than the approach with EWA and majority vote but higher than FTSA with meta-classifier; the performance level of all the methods will converge to a similar level between 15,000 to 20,000 episodes.

From Figure 4, we could observe that, with 20 bad demonstrations, FTSA with majority vote and EWA could still provide some benefit to the jumpstart, while HAT and FTSA with meta-classifier could not. With the bad jumpstarts, the performance of HAT and FTSA with the meta-classifier could converge to a higher level than the baseline after 10,000 episodes, even though they could not benefit from the demonstrations at the beginning stage.

Table 1 shows the jumpstart and total reward of each method when experimenting with different blendings of demonstrations. The jumpstart is the subtraction of the initial reward of the baseline from the average initial reward of each method (the jumpstart of

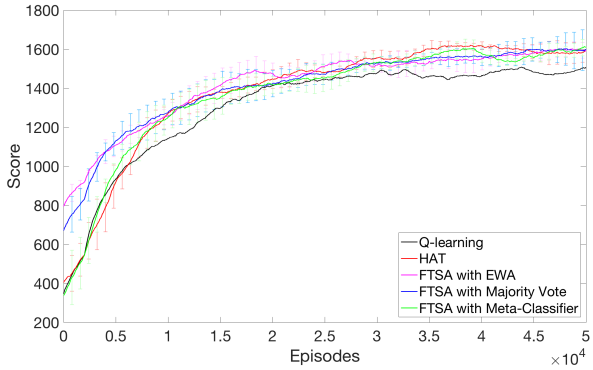

Figure 4: With 20 bad demos

the baseline is defined as 0). From the table, we could tell that with the increasing amount of bad demonstrations, the jumpstart and total reward of HAT declines. However, our FTSA with EWA and Majority Vote have a robust performance on those two metrics. Also, note that FTSA with the Meta-classifier also seems more vulnerable to the bad demonstrations than FTSA embedding with the other two algorithms—this may be related to the selection of meta-classifier and other factors. We also conduct Welch's t-test on the experiment results. With $p < 0.05$, we can say that the FTSA with Majority Vote and FTSA with EWA could significantly outperform HAT when provided 10 good demonstrations and 20 bad demonstrations, as well as when provided 20 bad demonstrations.

According to the experimental results, this 2-level structure could efficiently capture the knowledge from multiple demonstrations. Embedded with different bandit algorithms, it could eliminate the effects of bad demonstrations better than the classical HAT approach and enable the agent to make better usage from the blending of demonstrations without the assumption of using good or expert level demonstrations.

5 CONCLUSION AND FUTURE WORK

In this paper, we introduced a Flexible Two-level Structured Approach (FTSA) to address the task of combining policies learned from multiple demonstrations with different quality. With the probabilistic policy reuse scheme and algorithms from multi-armed contextual bandit problems, this approach could summarize and then combine advice from different demonstrations. With the experimental results in the Mario domain, particularly in scenarios where there are bad demonstrations, this approach could outperform the HAT in both overall performance (learning curve), initial performance (jumpstart), and total reward. We can therefore conclude that this two-level structure enables the agent to benefit from multiple demonstrations efficiently while eliminating the effects of bad demonstrations.

To improve this idea, first we would like to conduct more experiments: Try embedding more different algorithms other than those mentioned in this paper, theoretically analyze the pros, cons and robustness of each algorithm; Evaluate this approach with different combinations of demonstrations (i.e., a skewed combination of 95 bad demonstrations and 5 good demonstrations); Adopt other LfD

Table 1: Jumpstart and Total Reward of different methods with blending of demonstrations

Demonstration	10 good demos		10 good demos + 20 bad demos		20 bad demos	
	Jumpstart	Total Reward ($\times 10^7$)	Jumpstart	Total Reward ($\times 10^7$)	Jumpstart	Total Reward ($\times 10^7$)
Q-learning	0	6.541	0	6.541	0	6.541
HAT	565.020	7.266	390.344	7.116	57.629	6.922
FTSA with EWA	469.113	7.210	485.940	7.130	447.808	7.099
FTSA with Majority Vote	216.315	7.067	460.404	7.144	324.007	7.046
FTSA with Meta-Classifier	450.646	7.053	277.669	7.022	- 8.873	6.855

algorithms other than HAT as baseline method, like DPID [5], etc. Second, we are going to investigate the possibility of combining a confidence-based decision scheme into this approach. Third, we would like to further explore the performance of our FTSA scheme in both online (i.e., iterating with real-time state-action sequences) and offline (iterating with the demonstration examples) settings and analyzing the difference, if there is any. Fourth, considering the noise and irregularity of the collected demonstration data, more analysis of the demonstrations could also help with the performance and explainablity. For example, looking into the context of good and bad examples or extracting higher-level information with unsupervised methods might help in seperating out the good examples (sequences) from the bad demonstrations and vice versa.

REFERENCES

[1] Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. 2009. A survey of robot learning from demonstration. *Robotics and autonomous systems* 57, 5 (2009), 469–483.

[2] Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire. 2002. The nonstochastic multiarmed bandit problem. *SIAM journal on computing* 32, 1 (2002), 48–77.

[3] Leo Breiman. 1996. Bagging predictors. *Machine learning* 24, 2 (1996), 123–140.

[4] Tim Brys, Anna Harutyunyan, Halit Bener Suay, Sonia Chernova, Matthew E Taylor, and Ann Nowé. 2015. Reinforcement Learning from Demonstration through Shaping. In *IJCAI*. 3352–3358.

[5] Jessica Chemali and Alessandro Lazaric. 2015. Direct Policy Iteration with Demonstrations. In *IJCAI*. 3380–3386.

[6] Sonia Chernova and Manuela Veloso. 2007. Confidence-based policy learning from demonstration using gaussian mixture models. In *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*. ACM, 233.

[7] Luis C Cobo, Peng Zang, Charles L Isbell Jr, and Andrea L Thomaz. 2011. Automatic state abstraction from demonstration. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, Vol. 22. 1243.

[8] Fernando Fernández and Manuela Veloso. 2006. Probabilistic policy reuse in a reinforcement learning agent. In *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*. ACM, 720–727.

[9] H Friedrich and R Dillmann. 1995. Obtaining good performance from a bad teacher. In *Workshop: Programming by Demonstration vs Learning from Examples; International Conference on Machine Learning*.

[10] Sergey Karakovskiy and Julian Togelius. 2012. The Mario AI benchmark and competitions. *IEEE Transactions on Computational Intelligence and AI in Games* 4, 1 (2012), 55–67.

[11] Beomjoon Kim, Amir-massoud Farahmand, Joelle Pineau, and Doina Precup. 2013. Learning from limited demonstrations. In *Advances in Neural Information Processing Systems*. 2859–2867.

[12] George Konidaris, Scott Kuindersma, Roderic Grupen, and Andrew G Barto. 2010. Constructing skill trees for reinforcement learning agents from demonstration trajectories. In *Advances in neural information processing systems*. 1162–1170.

[13] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. 2010. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*. ACM, 661–670.

[14] Tyler Lu, Dávid Pál, and Martin Pál. 2010. Contextual multi-armed bandits. In *Proceedings of the Thirteenth international conference on Artificial Intelligence and Statistics*. 485–492.

[15] Pyry Matikainen, P Michael Furlong, Rahul Sukthankar, and Martial Hebert. 2013. Multi-armed recommendation bandits for selecting state machine policies for robotic systems. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 4545–4551.

[16] Bilal Piot, Matthieu Geist, and Olivier Pietquin. 2014. Boosted bellman residual minimization handling expert demonstrations. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 549–564.

[17] Halit Bener Suay, Tim Brys, Matthew E Taylor, and Sonia Chernova. 2016. Learning from demonstration for shaping through inverse reinforcement learning. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 429–437.

[18] Richard S Sutton and Andrew G Barto. 1998. *Reinforcement learning: An introduction*. Vol. 1. MIT press Cambridge.

[19] Matthew E. Taylor and Peter Stone. 2007. Cross-Domain Transfer for Reinforcement Learning. In *Proceedings of the Twenty-Fourth International Conference on Machine Learning (ICML)*.

[20] Matthew E Taylor, Halit Bener Suay, and Sonia Chernova. 2011. Integrating reinforcement learning with human demonstrations of varying ability. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*. International Foundation for Autonomous Agents and Multiagent Systems, 617–624.

[21] Christopher JCH Watkins and Peter Dayan. 1992. Q-learning. *Machine learning* 8, 3-4 (1992), 279–292.

[22] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks?. In *Advances in neural information processing systems*. 3320–3328.

[23] Li Zhou. 2015. A survey on contextual multi-armed bandits. *arXiv preprint arXiv:1508.03326* (2015).