SPAM: Biology-Inspired Starting Point Method for Transfer Learning Using Autoencoder Compression

Beom Jin Lee
University of California, Berkeley
cluesbj@berkeley.edu

Kyung Geun Kim
University of California, Berkeley
kyung4952@berkeley.edu

Abstract

Transfer learning is a promising area of developing methods to transfer knowledge and features learned in source tasks to improve the efficiency and performance in a related target task. However, the performance of transfer learning algorithms is currently severely limited. One of the causes for poor performances of transfer learning is in the inherent difficulty of capturing the main features of a model expert of the source task. In the evolutionary process, the highly complex structure of the animal brain is compressed and encoded into DNA to be passed to the next generation, yet it still preserves the highly efficient and robust learning behaviors. Inspired by this observation, in this paper, we propose SPAM, a starting point transfer learning method that compresses the neural network itself from the source task to be given as a starting point of the target task. The results are promising, as between OpenAI Gym's Inverted Pendulum and CartPole environments, we have observed the reduction in time to reach the maximum returns suggesting more efficient learning.

1 Introduction

Transfer learning is a promising area of developing tools and methods to transfer "knowledge" learned in one or more source tasks and use it to improve learning in a related target task. The idea is particularly compelling, because of its relevance to the ways in which humans learn. Human learners seem to be able to inherently transfer knowledge between tasks, so learners recognize and apply relevant knowledge from previous learning experiences to solve new tasks. While transfer learning is an exciting field of research, it has not been so successful. For example, in recent OpenAI's Retro Contest, even the best transfer learning agents could not surpass (or come close to) human level performance, even though humans have only trained for a few hours.

One cause for poor performances of transfer learning algorithms is that it is very difficult to capture the main features of the model expert from the source task, or they require domain knowledge. To illustrate, Taylor, et al. [1] use a starting-point method for transfer in temporal-difference RL. However, this method requires a mapping of features and actions between the tasks, and Taylor, et al. [1] provide a mapping based on their domain knowledge of the source and target tasks.

To investigate these causes, we turn to our biological roots. While an animal's brain is a very complex structure, it is extremely fast and robust in learning to solve new tasks. The efficiency and robustness for certain tasks such as leaning how to walk or how to speak, it is known that there exists a highly structured, dedicated region in the brain which animals are born with from the beginning. Zador [2] suggests that these complex structure of the brain is encoded into the genome of animals, so that animals are able to learn rapidly from birth. In a way, by encoding natural neural network structures onto the genomes, animals are able to transfer their remarkable learning algorithm from generation to generation.

These findings from biology suggests a potential leap forward for transfer learning by exploiting the general structure and key features of the model network of a source task. To that extent, we explore a starting-point transfer learning algorithm that requires very little domain expertise by mimicking how humans pass their learning algorithm to next generations. In order to transfer knowledge encoded in the highly complex structure of the brain, humans compress the structure onto a limited information space of the DNA. These biological conclusions from human brains suggest that we may be able to improve transfer learning algorithms by compressing our models to determine the dominant structure of the models, and use these compressed models as a basis for transfer learning. Indeed, model compression is a widely studied field in both reinforcement learning and computer vision, and a wide number of tools are available to compress models.

In this paper, we introduce Starting Point Autoencoder-Compressed Method (SPAM) where we compress the network generated from training on the source task using autoencoders [3], and use the corresponding weights as a starting point for training the target task. Throughout the paper, we used two similar OpenAI Gym environments [4], Inverted Pendulum and CartPole, and show successfully that our algorithm allows faster convergence to the optimal solution.

In Section 2, we discuss some related works, including those on model compression and those on starting point transfer learning. In Section 3, we present the architectural details of the general algorithm from a high-level view, so that the algorithm may be adopted to other domains. In Section 4, we then present the details and results of conducting transfer learning within and between Inverted Pendulum and CartPole Gym environments.

2 Related Work

Model compression is a widely studied field to compress large neural networks into smaller networks that have same or similar probability distributions. While we attempt to also use SPAM to output similar probability distributions, the bulk of our work is on determining a structure to the neural network, and differs from these work on model compression due to the difference in the goal of compressing networks.

Distillation Distillation is a model compression technique proposed by [5]. It uses two networks: the teacher network and the student network, where the teacher network is the model expert network and the student network is the compressed network to be returned. Intuitively, student network is trained such that it outputs the same values as the teacher network.

AutoML for Model Compression [6] AutoML for Model Compression (AMC) leverages reinforcement learning as its model compression policy and concludes that it outperforms conventional rule-based compression policy and shows that it achieves higher compression ratio, better preserving the accuracy.

3 Architectural Details

At a high level, our transfer learning method applies to general reinforcement learning techniques. As such, we outline here a general algorithm that applies to all deep reinforcement learning techniques.

3.1 Preliminaries

We let $f(\cdot,\cdot)$ be the reinforcement learning algorithm that we use to train either our model expert or our resulting expert using SPAM, where the second argument is optional, and represents the starting point (without the parametrization, we assume random initializaiton). We will denote the different reinforcement learning algorithms with subscripts. Also, let g represent the autoencoding step where we use autoencoders to compress our network's weights. Let $w(\cdot)$ represent the function that flattens neural network weights, into a single vector in $\mathbb{R}^d$, where d is the number of weights in the neural network. Similarly, let its inverse represent the function transforming the single vector in $\mathbb{R}^d$ into a neural network. We also let $\mathcal{S}$ be our source task and let $\mathcal{T}$ be our target task. Finally, let $\mathcal{M}$ represent the neural network models.

3.2 Autoencoder

Autoencoders are neural networks used to learn efficient data encoding using unsupervised learning. Though autoencoders are most often used to compress the data, like with images for convolutional neural networks, in our algorithm, we use autoencoders to compress the model expert of the source task $\mathcal{S}$, $\mathcal{M}^*$. Autoencoders are particularly compelling in our algorithm, as they impose bottleneck layers in the network that forces a compressed knowledge representation of the original input, which most closely resembles the way humans compress knowledge to DNA for transfer.

To compress the model expert $\mathcal{M}^*$, we use a 2-layer autoencoder neural network with sizes 256 and 128. The autoencoder accepts as input the weights $w(\mathcal{M}^*)$ of the model expert $\mathcal{M}^*$, and outputs new set of weights $W_{SPAM} = w(\mathcal{M}_{SPAM})$ for the new model $\mathcal{M}_{SPAM}$. The loss function for this autoencoder is the l-2 norm between $w(\mathcal{M}^*)$ and $w(\mathcal{M}_{SPAM})$.

3.3 Algorithm

The transfer learning architecture begins with creation of a model expert using f_1 on source task $\mathcal{S}$. Then, we run our autoencoder neural network g on the corresponding model $\mathcal{M}^*$. Finally, we take the compressed model and use it to train our target task $\mathcal{T}$ using a potentially different learning algorithm f_2, and return the resulting model. Formally, our algorithm is defined as the following:

Algorithm 1 Transfer Learning using Autoencoders

```
1: procedure SPAM($f_1$, $f_2$, $g$, $\mathcal{T}$, $\mathcal{S}$)
2:     $\mathcal{M}^* \leftarrow f_1(\mathcal{S})$
3:     $W_{SPAM} \leftarrow g(w(M^*))$
4:     $\mathcal{M}_{SPAM} \leftarrow w^{-1}(W_{SPAM})$
5:     $\mathcal{M}_{TL} \leftarrow f_2(\mathcal{T}, \mathcal{M}_{SPAM})$
6:     return $\mathcal{M}_{TL}$
```

4 Experiments

We will separate the discussion of our results into two parts: model compression and transfer learning between two different tasks. At a high level, in discussing both of these tasks, we look at the performance of the compressed model and the sparsity of the compressed model before and after it is trained again.

Throughout, our networks were of size 64, with two layers, and using policy gradients with rewards-to-go. As our model examples, we investigated the OpenAI's Inverted Pendulum and CartPole environments.

4.1 Model Compression

We ran the algorithm outlined in Section 3 on the same task of Inverted Pendulum and CartPole Environment to investigate model compression. We drive most of the discussion below with Inverted Pendulum, but the same observations hold for the CartPole Environment. While model compression is a modestly different field to transfer learning, since the bulk of the algorithm relies on successful model compression, we confirm that our algorithm using autoencoders is successful. Figure 1 shows the performance of the compressed model and expert model averaged across multiple seeds for the Inverted Pendulum Environment.

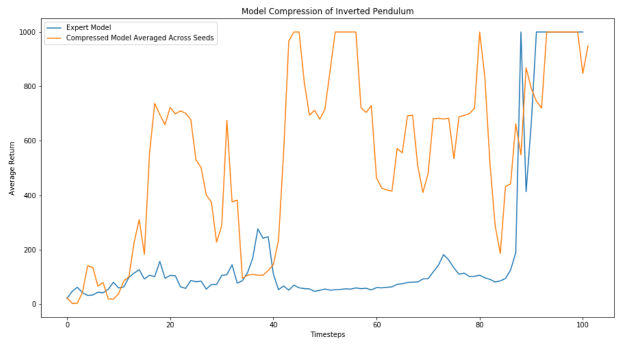

Figure 1: Performance of Compressed Model $\mathcal{M}_{cmp}$ and Model Expert $\mathcal{M}^*$ for Inverted Pendulum Environment

Likewise, figure 2 shows the performance of the compressed model and expert model averaged across multiple seeds for the CartPole Environment.

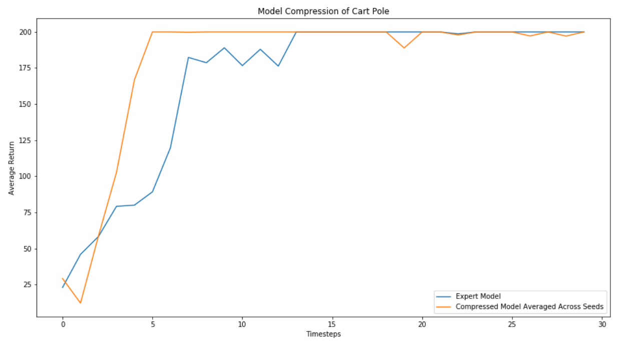

Figure 2: Performance of Compressed Model $\mathcal{M}_{cmp}$ and Model Expert $\mathcal{M}^*$ for CartPole Environment

4.1.1 Faster convergence

In pursuing model compression, one of the key features to look out for is faster convergence. The above graph and the quantitative data below achieves faster convergence to the maximum reward.

Model	Timestep to Convergence
Expert Model: Inverted Pendulum	88
Compressed Model: Inverted Pendulum	43
Expert Model: CartPole	13
Compressed Model: CartPole	4

While in the CartPole environment, the expert model is also very fast to converge to the maximum rewards, our results are the average across multiple seeds, so we only reach the maximum reward if and only if all the rewards of the multiple seeds at that timestep are maximized. This ensures that our results are not coincidental, but we guarantee convergence to the maximum reward with high probability.

4.1.2 Dependence on the Reinforcement Learning Algorithm

Noticeably, in this set of experiments, we see that after the algorithm reaches the maximum rewards, the average returns begin to fluctuate. This is in large part because of the reinforcement learning algorithm pursued in our experiments: policy gradients, which are known to be unstable. In fact, our next analysis with the model expert, by pushing the model expert beyond 100 timesteps, show that the same fluctuation holds for model expert too:

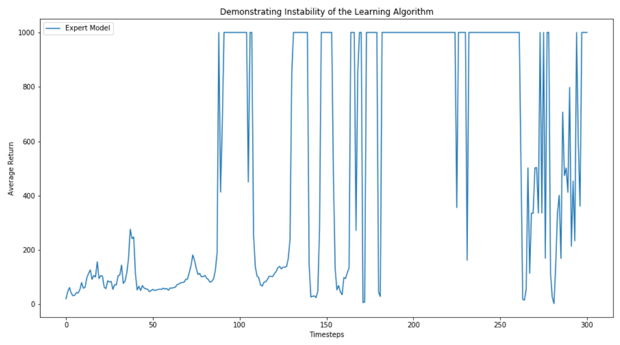

Figure 3: Performance of Model Expert for Extended Timesteps Beyond Convergence

Thus, the fluctuation in the rewards is not within our method, but due to the instability of the learning algorithm of choice in our experiments.

4.1.3 Sparsity of the Resulting Model

We begin with two definitions:

Definition 4.1. Zero Ratio. The zero ratio measures how sparse the weights are directly from the compression step of the model expert. Following notational conventions outlined in Section 2, this is measured as:

$$\frac{\sum_{i=1}^{d} \mathbf{1}\{w(g(\mathcal{M}^*))_i = 0\}}{d}$$

Definition 4.2. Sparsity Ratio. The sparsity ratio measures the extent to which sparsity is maintained after the compressed model is trained again. Again, following notational conventions outlined in Section 2, this is measured as:

$$\frac{\sum_{i=1}^{d} \mathbf{1}\{w(\mathcal{M}_{SPAM})_i = 0\}}{\sum_{i=1}^{d} \mathbf{1}\{w(g(\mathcal{M}^*))_i = 0\}}$$

Below we have the sparsity and zero ratios for both Inverted Pendulum and CartPole environments:

Environment	Zero Ratio	Sparsity Ratio
Inverted Pendulum	0.62	0.79
CartPole	0.61	0.82

The sparsity of the resulting model is very close to the compressed model resulting directly from the autoencoding step. As our goal is to use the compressed models to train new experts, it confirms that these neural networks have a dominant structure that can be exploited to generalize across multiple environments.

4.2 Transfer Learning from CartPole to Inverted Pendulum Environment

Confirming our transfer learning between the same tasks, we apply our algorithm on two different, but similar tasks of CartPole and Inverted Pendulum. Similar to the performance of compressed model and the model expert, seen above in Figure 1, transfer learning between Cartpole and Inverted Pendulum works similarly well. Below, Figure 4 shows the performance of transfer learning:

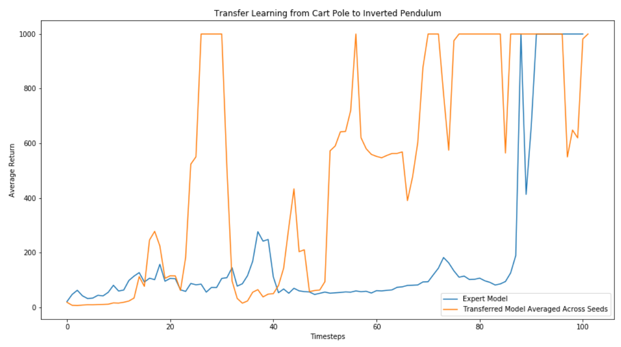

Figure 4: Transfer Learning from CartPole Environment to Inverted Pendulum Environment

4.2.1 Convergence Rate

In pursuing model compression, one of the key features to look out for is faster convergence. The above graph and the quantitative data below achieves faster convergence to the maximum reward.

Model	Timestep to Convergence
Expert Model	88
SPAM Model	25

Again, our results are the average across multiple seeds, so we only reach the maximum reward if and only if all the rewards of the multiple seeds at that timestep are maximized. This ensures that our results are not coincidental, but we guarantee a faster convergence to the maximum reward with high probability.

4.2.2 Sparsity of the Resulting Model

The zero ratio and sparsity ratio of the resulting model of transfer learning through SPAM is as follows:

Zero Ratio	Sparsity Ratio
0.61	0.71

Similar to our results with model compression, we see that the sparsity ratio is high. However, there is still a drop in the sparsity ratio compared to model compression. We predict that this is due to the model beginning to overfit to the new task it has been given, and given more iterations, we would see an even bigger drop in the sparsity ratio. Previously, the model was given the same task it had already been trained on.

4.3 Transfer Learning on Suboptimal Models

Above in figure 4, we showed transfer learning across two model experts. We show in figure 5 that given a model expert of the source task, even if we have sub-optimal policies to train our target task, the good starting point of the model expert will allow the tasks to converge to high average returns quickly.

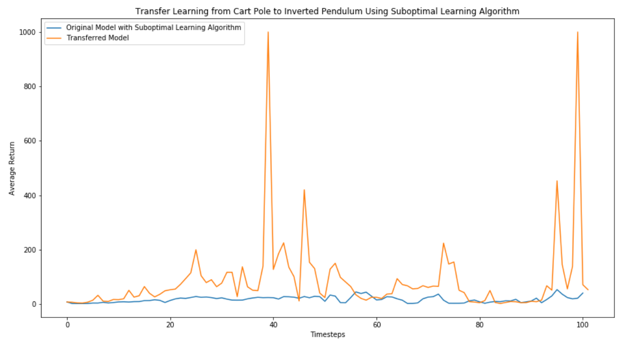

Figure 5: Transfer Learning from CartPole Environment to Inverted Pendulum Environment

The results here are more dramatic than those above, as the results above are bounded by the maximum rewards. Instead, here, as we have suboptimal policies, we see that SPAM clearly outperforms the suboptimal model expert.

5 Discussion

5.1 Autoencoder as Compression Algorithm

While there exists many efficient neural network compression algorithms in supervised learning including computer vision [7], in our case, we use an autoencoder as our compression algorithm to specifically mimic the evolutionary process in biology. Conclusions from biology research suggests that there exists a dominant structure of a natural neural network for many learning tasks. In the case of animals, animals compress the complex structure of the brain into the space-limited DNA, and future generations use these genes to learn, robustly and efficiently, new tasks. Indeed, we argue that the compression of the complex structure of the brain into a space-limited DNA requires animals to find the dominant structure of the complex brain to pass onto future generations. We exploit this observation to be used in the case of transfer learning algorithm using deep reinforcement learning because we believe the notion of dominant structure exists in artificial neural networks as well. In particular, we use autoencoder as our compression scheme as it most resembles the way the structure of the brain is encoded into the DNA, then propagated back into future generation's brains.

5.2 Model Compression to Transfer Learning

We started with model compression to verify our hypothesis that the dominant structure of the neural networks exists and it can be found using autoencoders. As our conclusions above, we verified that the compressed network very rapidly converges to the expert level when trained with the same policy gradient used to train the initial expert agent. We also verified that at the start of the learning, the compressed agent is no better than the agent with randomly initialized agent. We finally observed that most of the sparsity of the neural network is conserved before and after the training of the compressed agent. With these three results, we conclude that there indeed exists the dominant neural network structure for each of the tasks and it can be found by compressing the network using autoencoders. Applying the same idea to transfer learning is not a big leap. Since we have already concluded that dominant structure of an artificial neural network can be discovered and exploited using autoencoders, we were able to transfer the compressed agent to be trained in new, but similar, task from CartPole to Inverted Pendulum.

6 Conclusion

We present Starting Point Autoencoder-Compressed Method (SPAM), a starting point transfer learning algorithm compressing the model network generated from the source task as a starting point for the target task. Conclusions about the way humans transfer brain structure across generations inspired our compression scheme. The initial results from applying SPAM onto transferring within the same tasks showed promising results that achieves faster convergence and sparsity in the resulting model. Applying SPAM to transfer from the CartPole Environment to the Inverted Pendulum environment has been equally promising, and shows similar rate of convergence and resulting level of sparsity in the models. Finally, we applied SPAM to do transfer learning task but train using a suboptimal set of hyperparameters, and we saw even more dramatic increases in the performances. Unfortunately, our algorithm suffers from heavy dependence on the training algorithm. If the training algorithm is suboptimal, we should not expect the resulting model to be optimal, though we have demonstrated that it outperforms the suboptimally trained agent. As the goal of SPAM is to make the models robust to changes in task by extracting the main structure, an exciting avenue would be to apply similar methods as SPAM to other tasks where robustness is not guaranteed.

Contributions

Kyung Geun provided the initial biological inspiration and the use of autoencoder for the algorithm. Beom Jin and Kyung Geun designed and implemented the experiments and carried out the initial experiments. Kyung Geun ran the remaining experiments and collected data presented in our paper. Beom Jin carried out the analysis and formalized the algorithm used in the paper.

References

[1] Matthew E Taylor, Peter Stone, and Yaxin Liu. Value functions for rl-based behavior transfer: A comparative study. 2005.

[2] Anthony M. Zador. A critique of pure learning: What artificial neural networks can learn from animal brains. *bioRxiv*, 2019.

[3] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[4] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016.

[5] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[6] Yihui He, Ji Lin, Zhijian Liu, Hanrui Wang, Li-Jia Li, and Song Han. Amc: Automl for model compression and acceleration on mobile devices. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 784–800, 2018.

[7] Yu Cheng, Duo Wang, Pan Zhou, and Tao Zhang. A survey of model compression and acceleration for deep neural networks, 2017.