Contributions to Deep Transfer Learning

From Supervised to Reinforcement Learning

Matthia Sabatelli
Montefiore Institute, Department of Electrical Engineering and Computer Science, Université de Liège, Belgium

March 30th 2022

Presentation outline

1. Part I: Preliminaries
   - Deep Transfer Learning
2. Part II: Supervised Learning
   - On the Transferability of Convolutional Neural Networks
   - On the Transferability of Lottery Winners
3. Part III: Reinforcement Learning
   - A Novel Family of Deep Reinforcement Learning Algorithms
   - On the Transferability of Deep-Q Networks

PART I

Deep Transfer Learning

PART II

We start by defining the components of Supervised Learning:

- An input space $\mathcal{X}$,
- An output space $\mathcal{Y}$,
- A joint probability distribution $P(X, Y)$.
- A loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$

The goal is to find a function $f : \mathcal{X} \to \mathcal{Y}$ that minimizes the expected risk:

$$R(f) = \mathbb{E}_{(\vec{x},y)\sim P(X,Y)}\left[\ell(y, f(\vec{x}))\right]$$

On the Transferability of Convolutional Neural Networks

Typically, the only information available to build this function f is a learning sample of input-output pairs $LS_T = \{(x_i, y_i) | i = 1, \ldots, N_T\}$ drawn independently from $P_T(x, y)$.

However, when it comes to Transfer Learning we have an additional dataset LS_S that can be used for finding a better f than when only LS_T is used for training.

On the Transferability of Convolutional Neural Networks

For our first study ...

- We consider f to come in the form of a Convolutional Neural Network
- We define the source domain $\mathcal{D}_S$ to be that of natural images
- We define the target domain $\mathcal{D}_T$ to be that of Digital Heritage
- We assume that labels are available in both the source and target data and that the input spaces $\mathcal{X}_T$ and $\mathcal{X}_S$ match: Inductive Transfer Learning

On the Transferability of Convolutional Neural Networks

We have a CNN pre-trained on the ImageNet dataset as source-task $\mathcal{T}_S$

We have three target-tasks $\mathcal{T}_T$ defined by the Rijksmuseum collection

On the Transferability of Convolutional Neural Networks

On the Transferability of Convolutional Neural Networks

On the Transferability of Convolutional Neural Networks

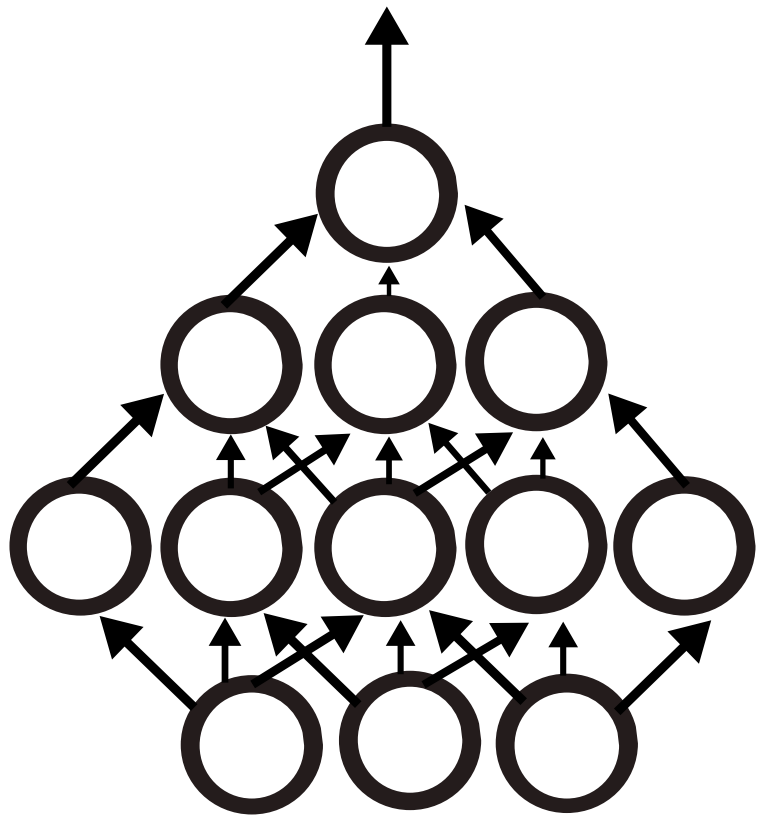
On the Transferability of Convolutional Neural Networks

On the Transferability of Lottery Winners

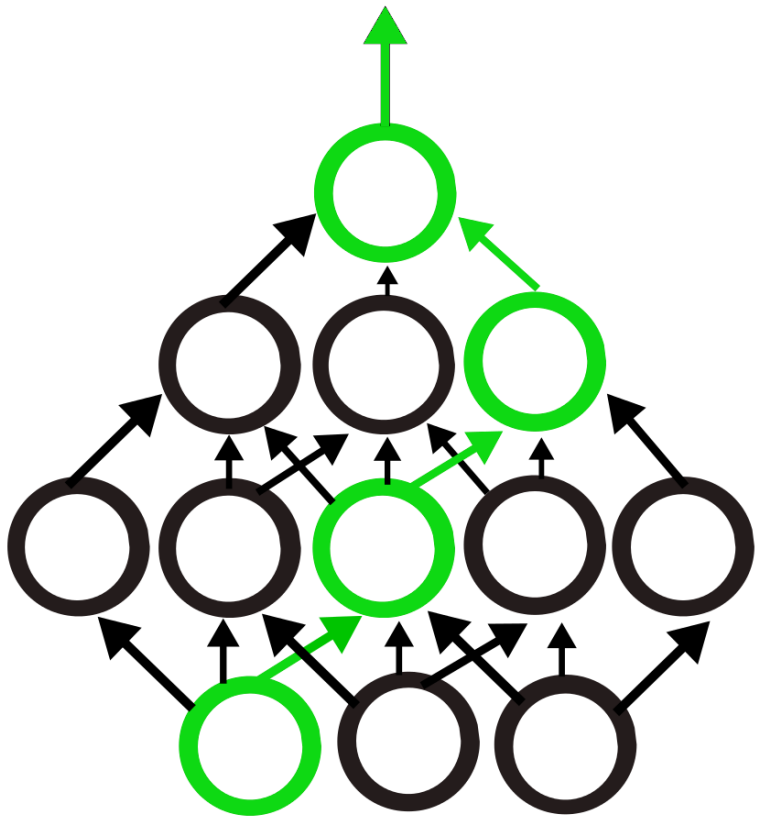
The Lottery Ticket Hypothesis: *"A randomly-initialized dense neural network contains a subnetwork that is initialized such that when trained in isolation it can match the test accuracy of the original network after training for at most the same number of iterations.*[1]*"*

[1] Frankle, Jonathan, and Michael Carbin. "The lottery ticket hypothesis: Finding sparse, trainable neural networks." (2018)

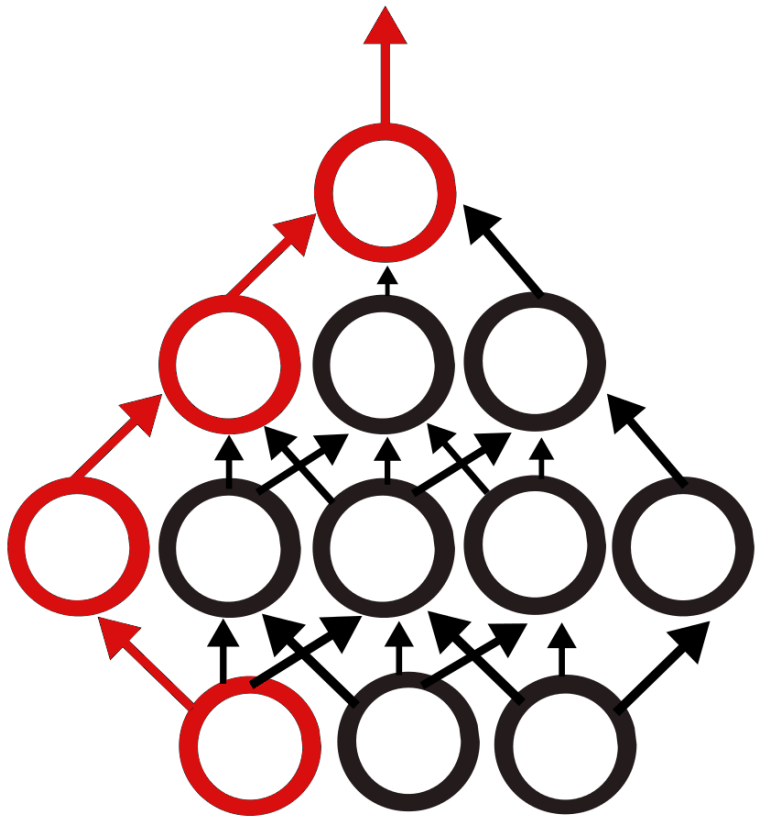
On the Transferability of Lottery Winners



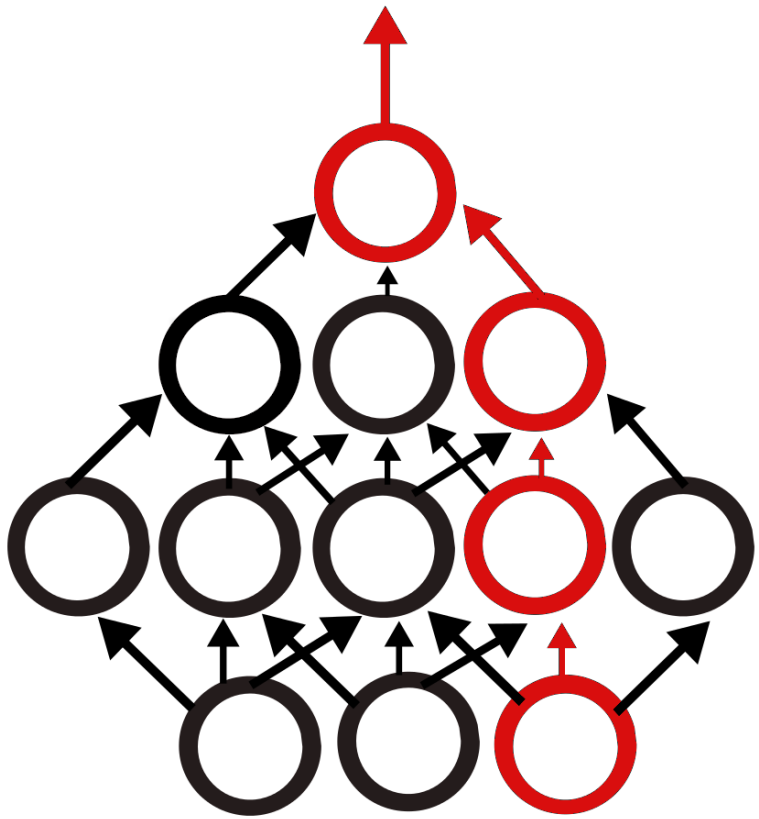
On the Transferability of Lottery Winners



On the Transferability of Lottery Winners



On the Transferability of Lottery Winners



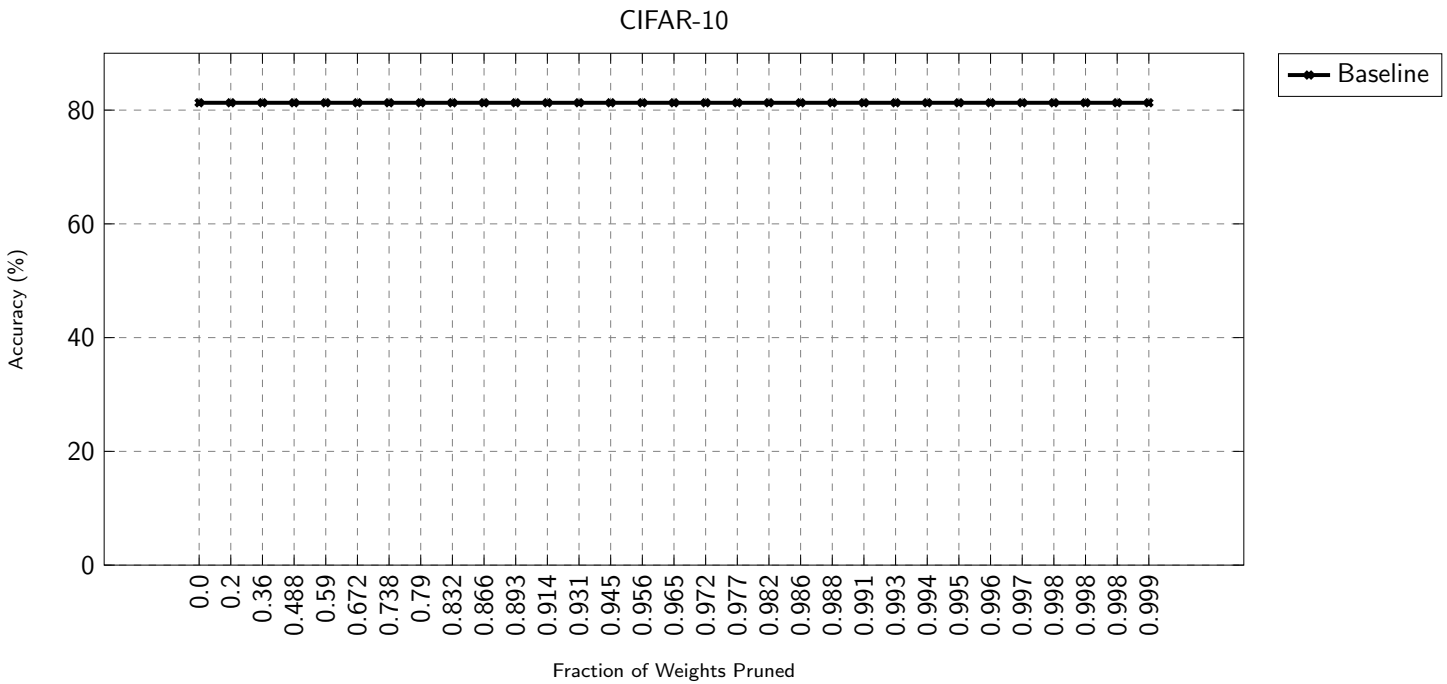
On the Transferability of Lottery Winners

$\Rightarrow$ How does one find a winning ticket?

- Randomly initialize a network $f(x; \theta_0)$ where $\theta_0 \sim \mathcal{D}_\theta$
- We train the network for j iterations
- We prune $p\%$ of the parameters in θ_j creating a mask m
- Reset the remaining parameters to their values at θ_0, creating a winning ticket $f(x; m \odot \theta_0)$

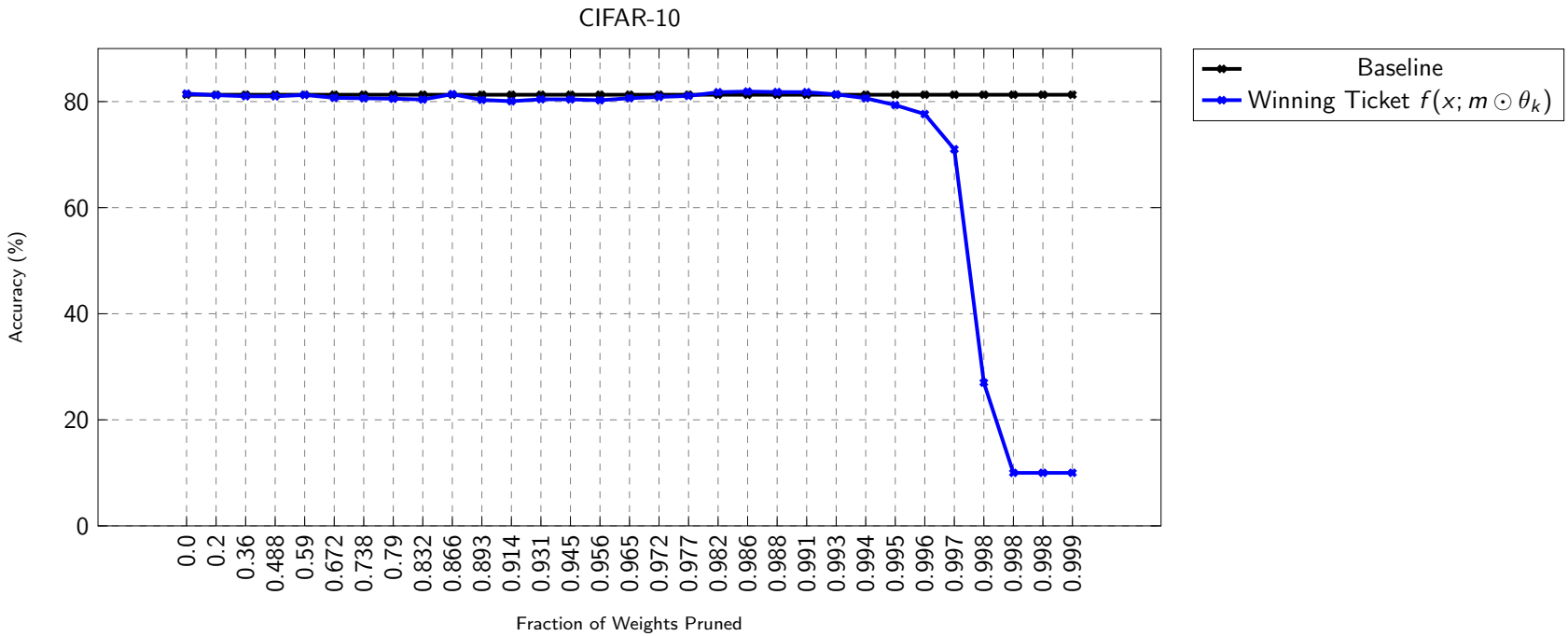
On the Transferability of Lottery Winners

The Lottery Ticket Hypothesis in practice ...



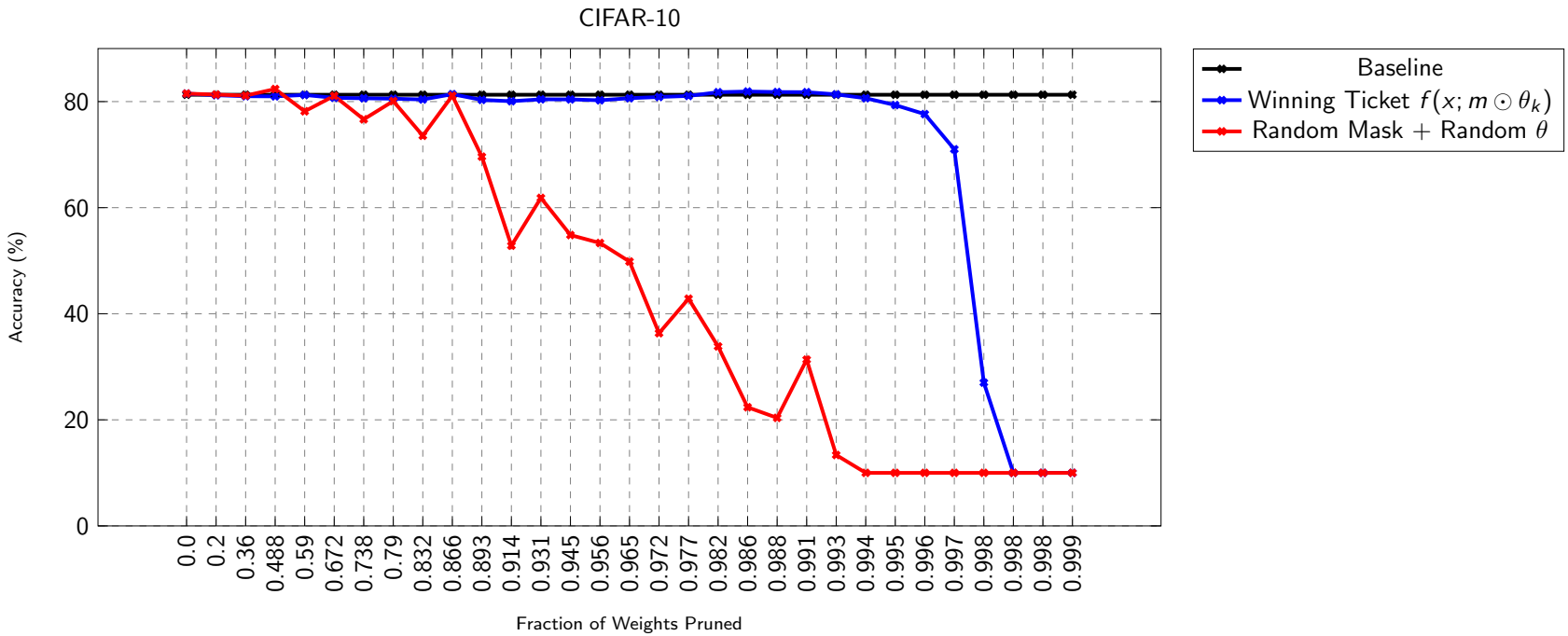
On the Transferability of Lottery Winners

The Lottery Ticket Hypothesis in practice ...



On the Transferability of Lottery Winners

The Lottery Ticket Hypothesis in practice ...



On the Transferability of Lottery Winners

Why are lottery tickets $f(x; m \odot \theta_0)$ so special?

- Train faster
- Faster Inference
- (Sometimes) obtain a better final performance

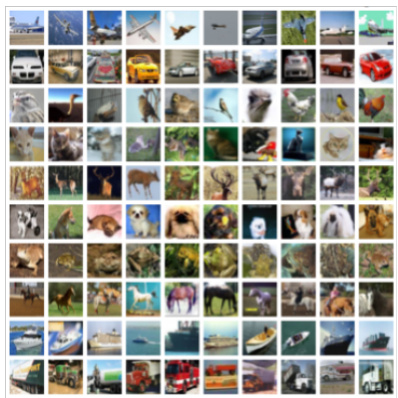
However ...

- Identifying a winning ticket is computationally expensive
- We do not know how and why lottery winners appear throughout learning

On the Transferability of Lottery Winners

$\Rightarrow$ Therefore we study whether winning tickets $f(x; m \odot \theta_0)$ found on natural image datasets can get transferred to the non-natural realm

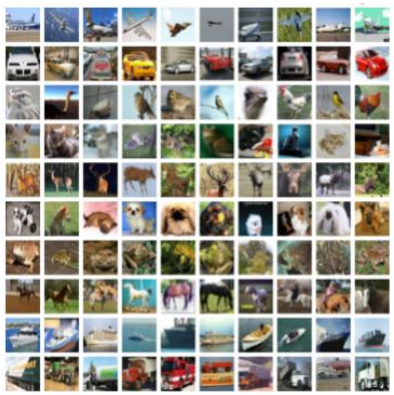
We use three popular Computer Vision datasets as source domains $\mathcal{D}_S$:



On the Transferability of Lottery Winners

$\Rightarrow$ Therefore we study whether winning tickets $f(x; m \odot \theta_0)$ found on natural image datasets can get transferred to the non-natural realm

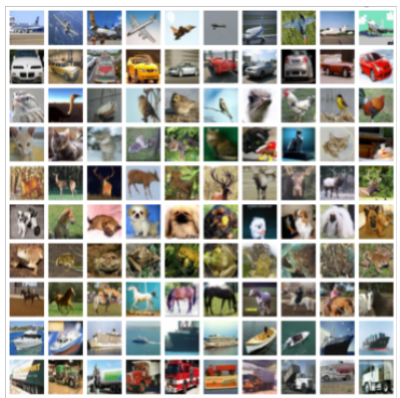
We use three popular Computer Vision datasets as source domains $\mathcal{D}_S$:



On the Transferability of Lottery Winners

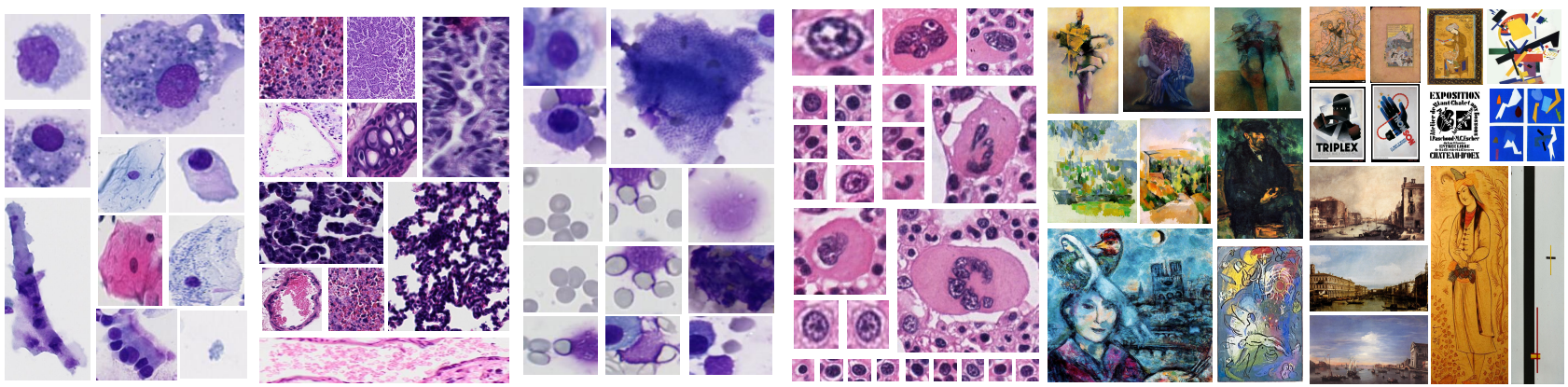
$\Rightarrow$ Therefore we study whether winning tickets $f(x; m \odot \theta_0)$ found on natural image datasets can get transferred to the non-natural realm

We use three popular Computer Vision datasets as source tasks $\mathcal{T}_S$



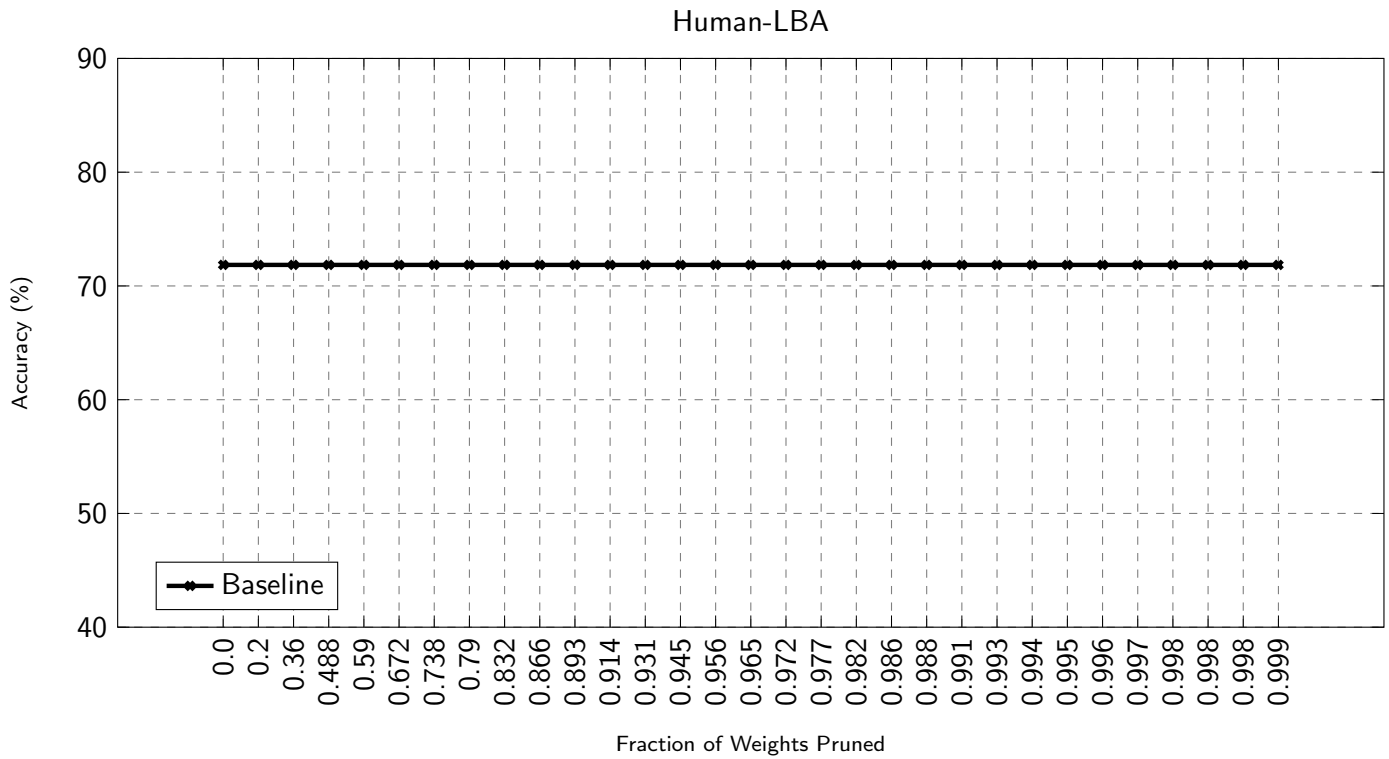
On the Transferability of Lottery Winners

And we use seven datasets of non-natural images as target tasks $\mathcal{T}_T$ coming from the fields of Digital Pathology and Digital Heritage



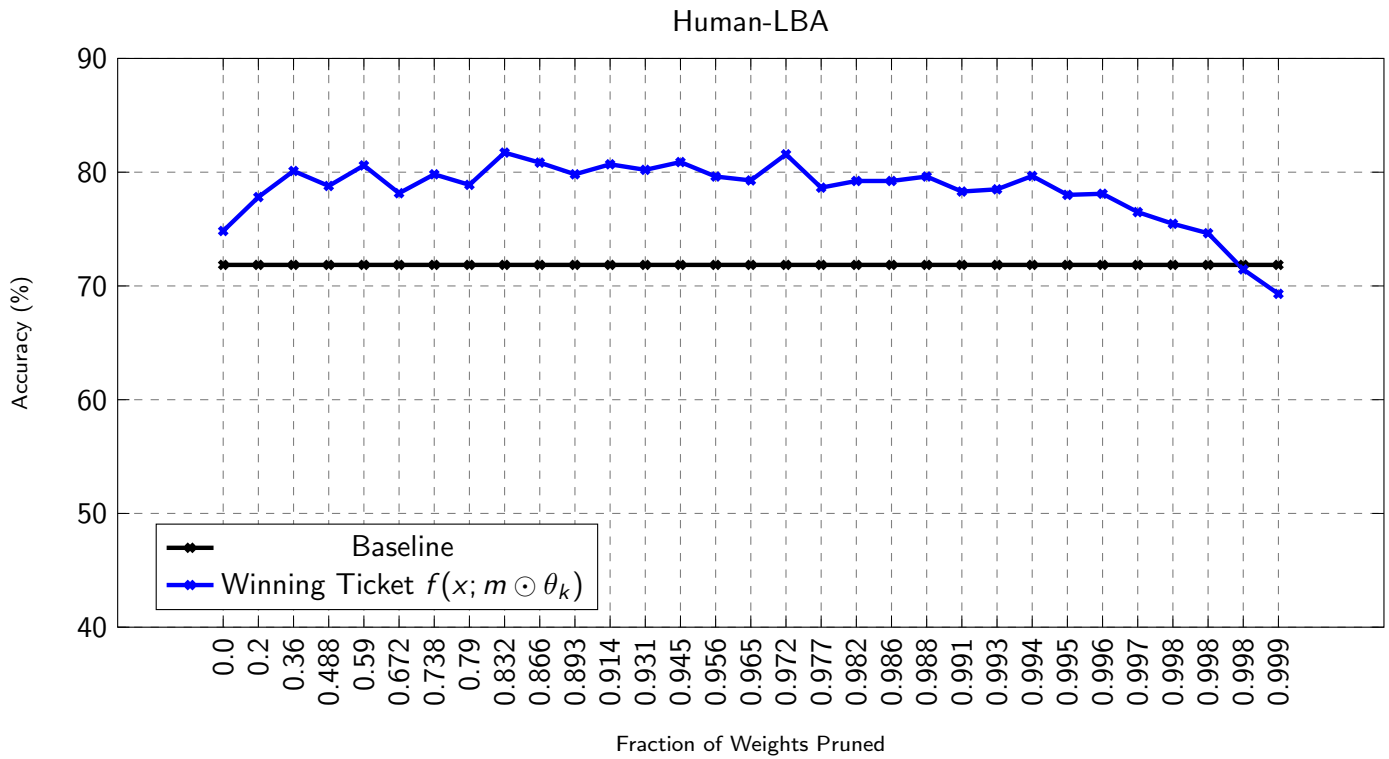
On the Transferability of Lottery Winners

Transferring lottery winners to the Digital Pathology data ...



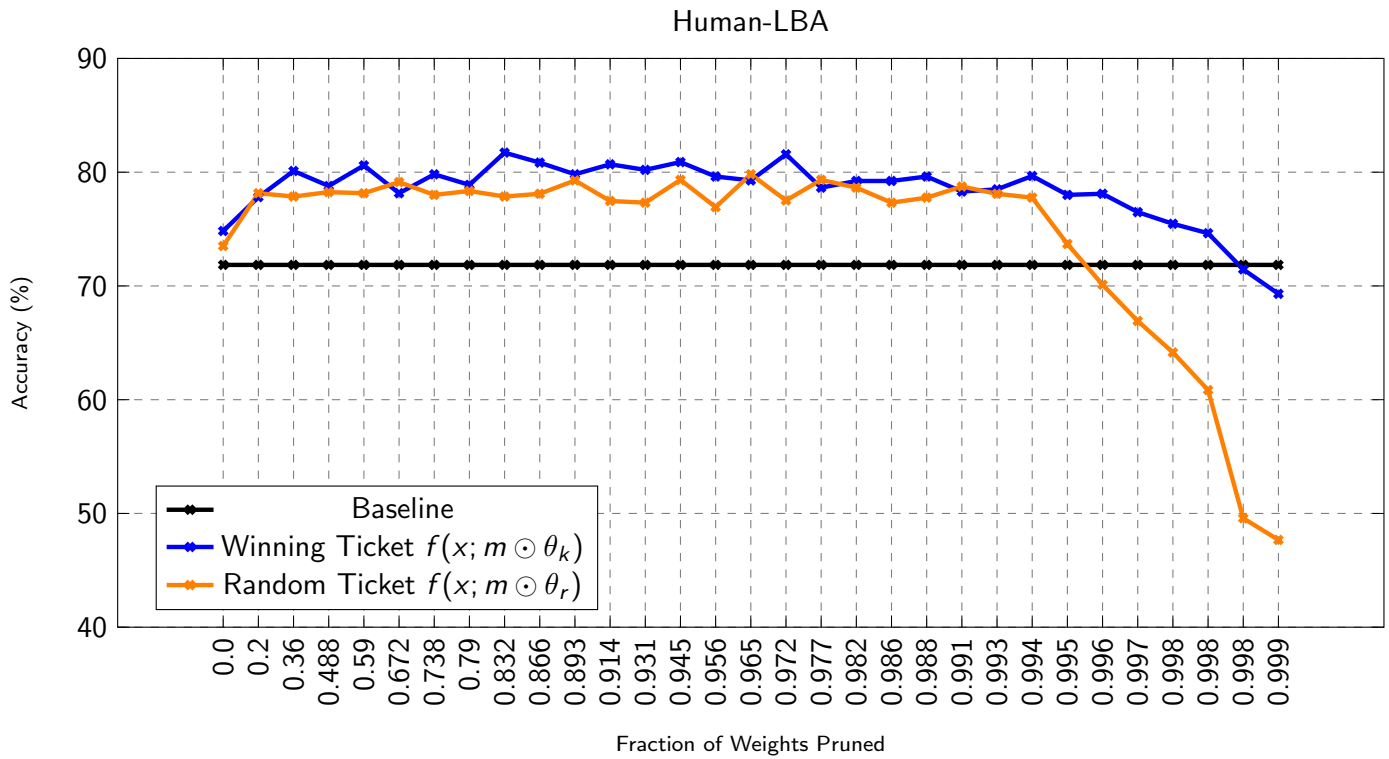
On the Transferability of Lottery Winners

Transferring lottery winners to the Digital Pathology data ...



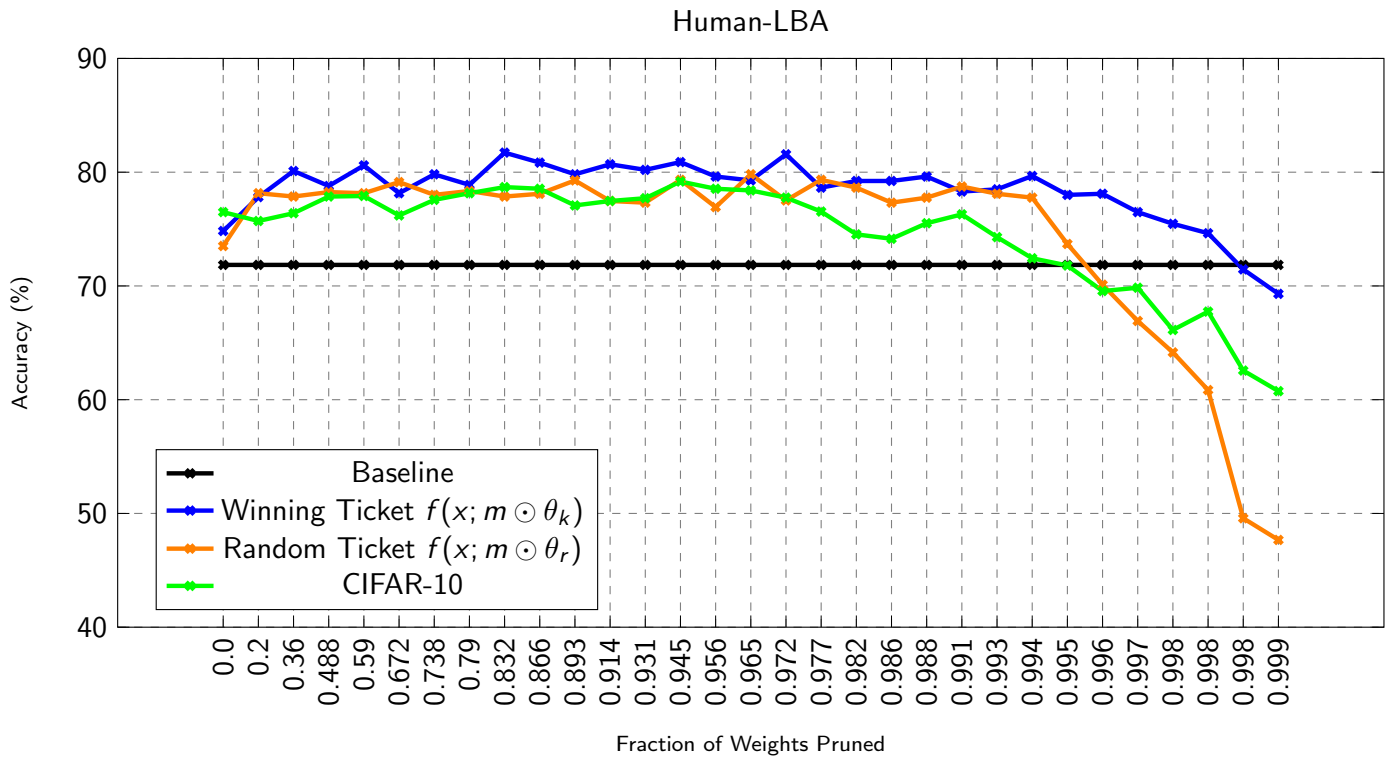
On the Transferability of Lottery Winners

Transferring lottery winners to the Digital Pathology data ...



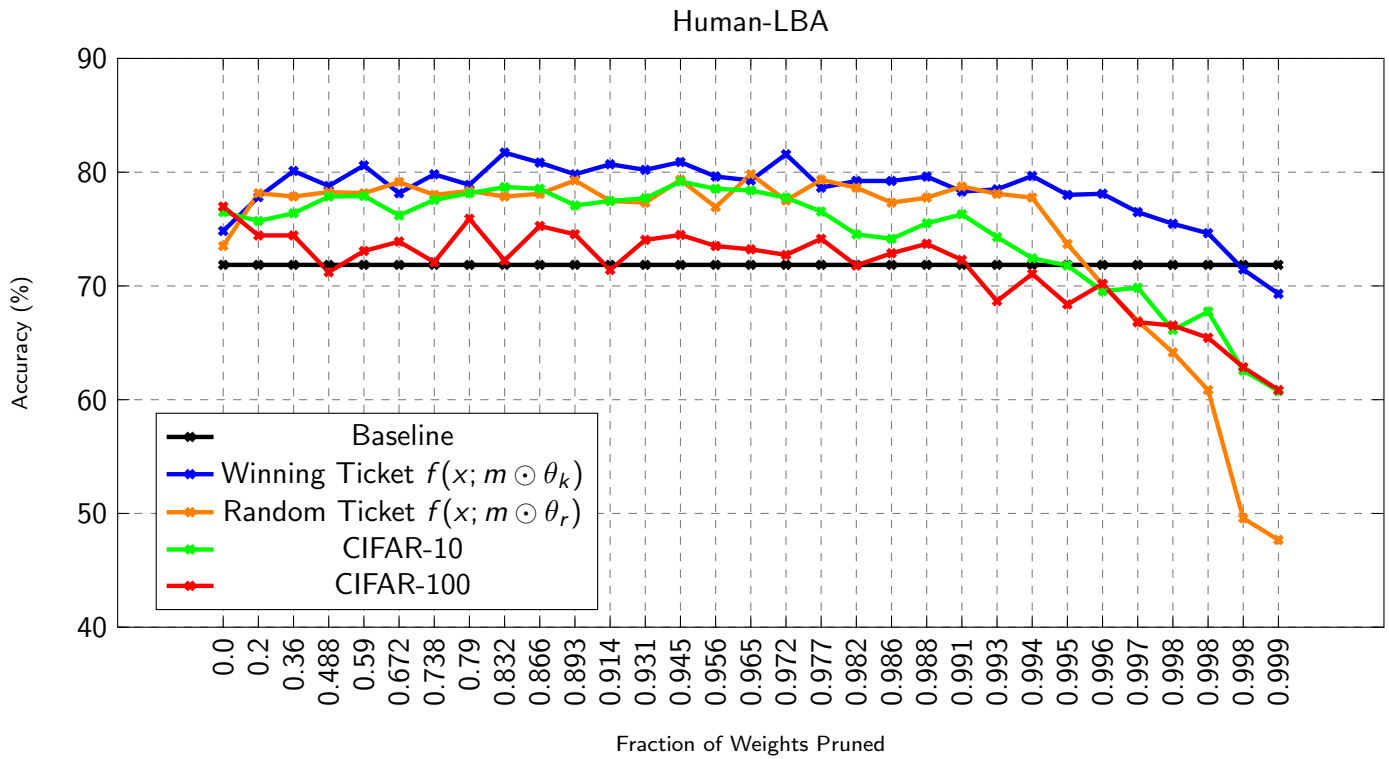
On the Transferability of Lottery Winners

Transferring lottery winners to the Digital Pathology data ...



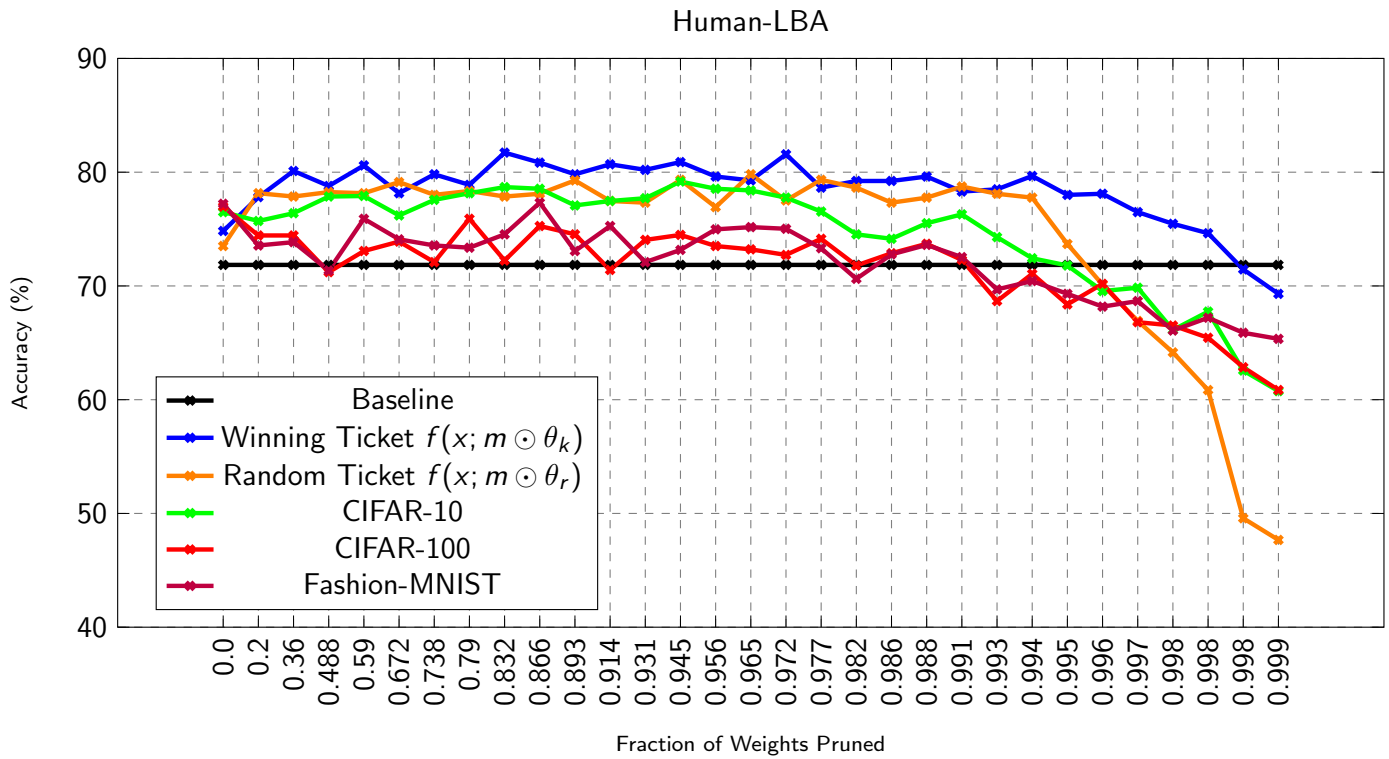
On the Transferability of Lottery Winners

Transferring lottery winners to the Digital Pathology data ...



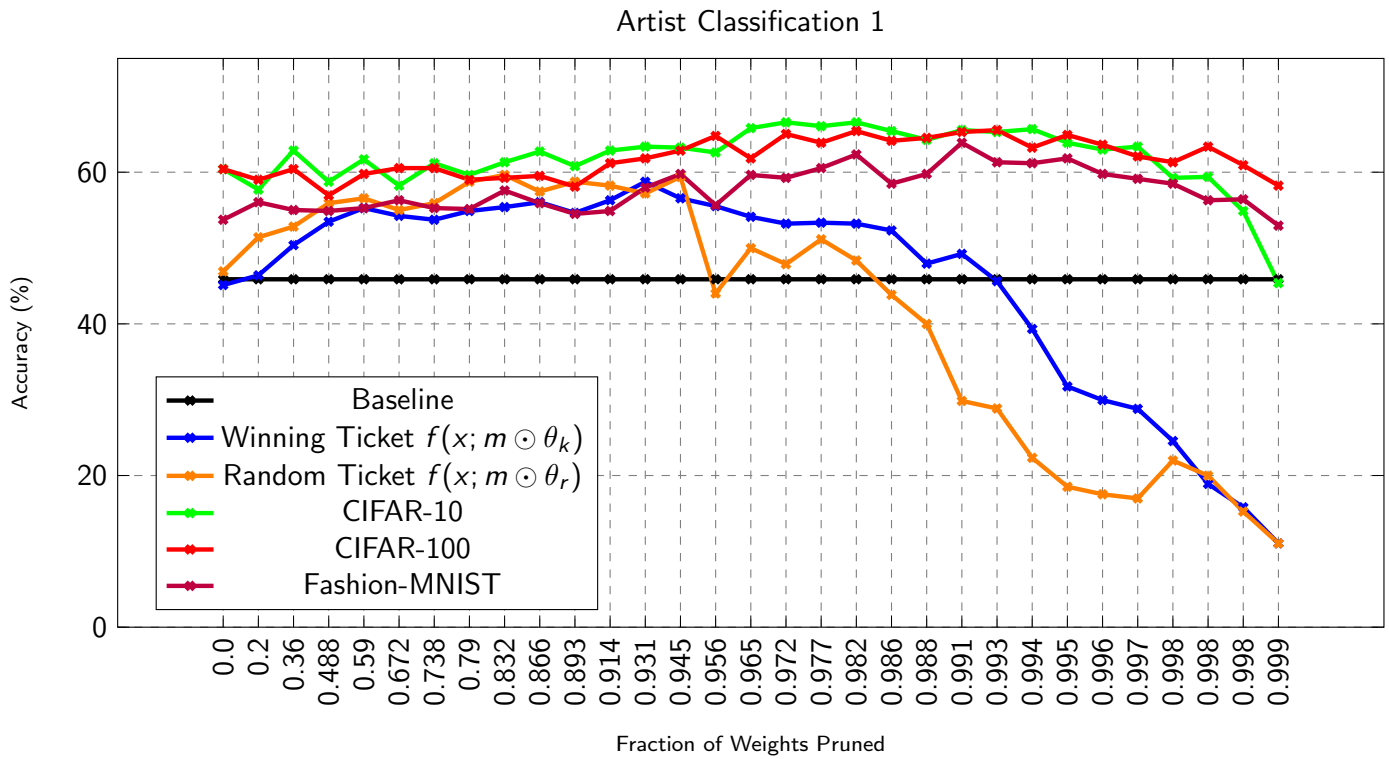
On the Transferability of Lottery Winners

Transferring lottery winners to the Digital Pathology data ...



On the Transferability of Lottery Winners

Transferring lottery winners to the Digital Heritage data ...



On the Transferability of Lottery Winners

Our main findings show that on Digital Pathology data

- All lottery winners significantly outperfom unpruned models
- Natural lottery winners contain a generic inductive bias (to some extent)
- Best performance is obtained by identifying a winning ticket directly on the target task $\mathcal{T}_T$

whereas on Digital Heritage data

- Natural lottery winners transfer much better
- They can even outperform target task $\mathcal{T}_T$ tickets

On the Transferability of Lottery Winners

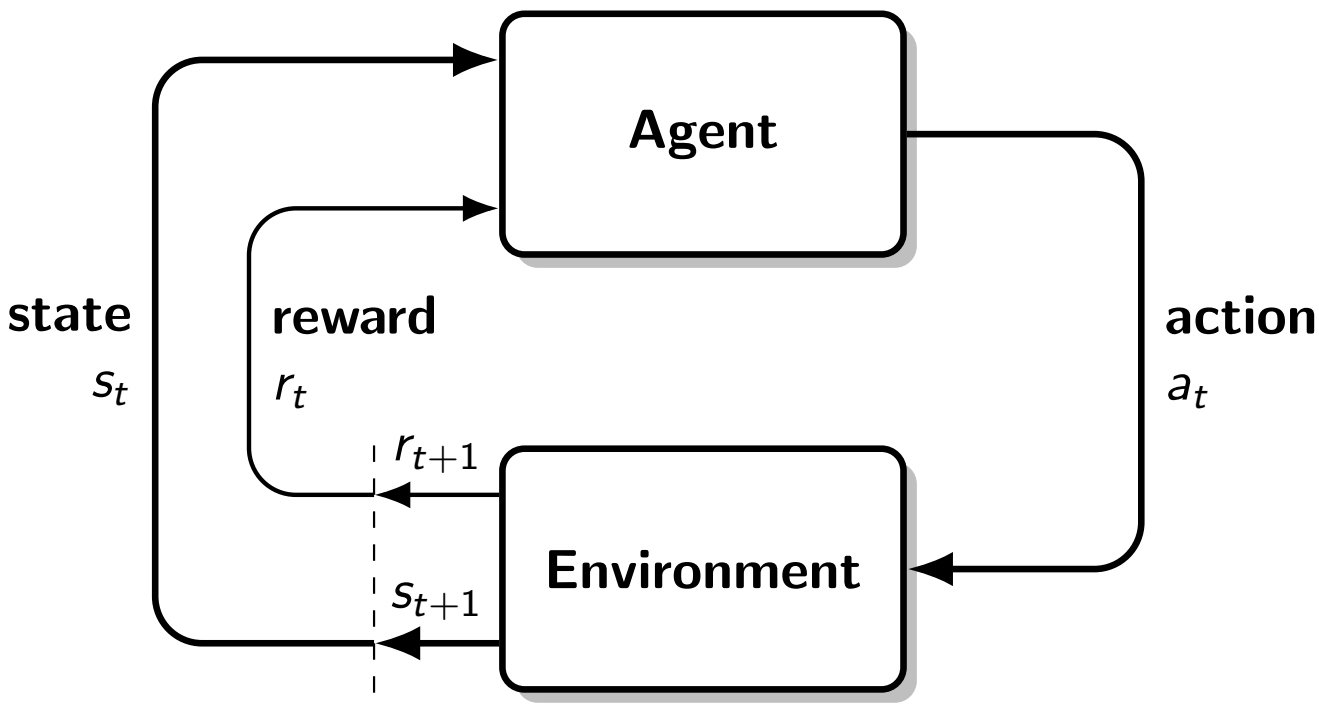
We also provide additional empirical insights into the Lottery Ticket Hypothesis:

- We show that completely pre-trained winning tickets $f(x; m \odot \theta_i)$ overfit on the source task $\mathcal{T}_S$
- The presence of winning tickets does not depend on the size of the training data
- The closer the source task $\mathcal{S}_T$ and the target task $\mathcal{T}_T$ the better the transferability of $f(x; m \odot \theta_k)$

PART III

A Novel Family of Deep Reinforcement Learning Algorithms

We now consider a different machine learning paradigm: Reinforcement Learning (RL), where an agent needs to learn how to interact with its environment



A Novel Family of Deep Reinforcement Learning Algorithms

Such interaction is modeled as a Markov Decision Process (MDP) consisting of the following elements:

- A set of possible states $\mathcal{S}$,
- A set of possible actions $\mathcal{A}$,
- A transition function $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$,
- A reward function $\mathfrak{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$,
- A discount factor denoted as $\gamma \in [0, 1]$.

Therefore we can define an MDP as $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathfrak{R}, \gamma \rangle$.

A Novel Family of Deep Reinforcement Learning Algorithms

The interaction between the agent and the environment is given by the agent's policy π, a probability distribution over $a \in \mathcal{A}(s)$ for each $s \in \mathcal{S}$:

$$\pi(a|s) = \Pr\{a_t = a | s_t = s\}, \text{ for all } s \in \mathcal{S} \text{ and } a \in \mathcal{A}.$$

The goal of the agent is to maximize the expected discounted return as:

$$\begin{aligned} G_t &= r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots \\ &= \sum_{k=0}^{\infty} \gamma^k r_{t+k}. \end{aligned}$$

A Novel Family of Deep Reinforcement Learning Algorithms

In RL some components of the MDP are unknown

$$\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle,$$

which in the case of model-free RL can be overcome by learning either the state-value function

$$V^{\pi}(s) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k} \middle| s_t = s, \pi \right],$$

or the state-action value function

$$Q^{\pi}(s, a) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k} \middle| s_t = s, a_t = a, \pi \right].$$

A Novel Family of Deep Reinforcement Learning Algorithms

- Model-free algorithms are implemented in a tabular fashion, meaning that the state values, or state-action values, are stored within tables of sizes $|\mathcal{S}|$ and $|\mathcal{S} \times \mathcal{A}|$ respectively
- For many problems we typically seek to learn an approximation of the value functions

$$V^{\pi}(s) \approx V^{\pi}(s;\theta) \text{ and } Q^{\pi}(s,a;\theta) \approx Q(s,a;\theta)$$

A Novel Family of Deep Reinforcement Learning Algorithms

This approximation can be represented by a Convolutional Neural Network

A Novel Family of Deep Reinforcement Learning Algorithms

Typical DRL algorithms aim at only learning an approximation of the state-action value function $Q^{\pi}(s, a) \approx Q^{\pi}(s, a; \theta)$

- A policy π can only be derived from $Q^{\pi}(s, a)$
- It is hard to combine RL algorithms with neural networks
- Dealing with one value function is enough

However ...

- Training can be very slow
- Algorithms are prone to diverge
- Do not correctly estimate $Q^{\pi}(s, a)$

A Novel Family of Deep Reinforcement Learning Algorithms

$\Rightarrow$ Therefore we suggest to jointly approximate the state-value function $V^{\pi}(s;\phi)$ alongside the state-action value function $Q^{\pi}(s,a;\theta)$

We can do this by either learning $V^{\pi}(s;\phi)$ with

$$L(\phi) = \mathbb{E}_{\langle s_t, a_t, r_t, s_{t+1}\rangle \sim U(D)}\left[\left(r_t + \gamma V(s_{t+1};\Phi^{-}) - V(s_t;\Phi)\right)^2\right],$$

and $Q^{\pi}(s,a)$ with:

$$L(\theta) = \mathbb{E}_{\langle s_t, a_t, r_t, s_{t+1}\rangle \sim U(D)}\left[\left(r_t + \gamma V(s_{t+1};\Phi^{-}) - Q(s_t,a_t;\theta)\right)^2\right],$$

A Novel Family of Deep Reinforcement Learning Algorithms

... or by learning $V^{\pi}(s;\phi)$ with

$$L(\phi) = \mathbb{E}_{\langle s_t, a_t, r_t, s_{t+1}\rangle \sim U(D)}\left[\left(r_t + \gamma \max_{a \in \mathcal{A}} Q(s_{t+1}, a; \theta^{-}) - V(s_t; \Phi)\right)^2\right],$$

and $Q^{\pi}(s,a)$ with:

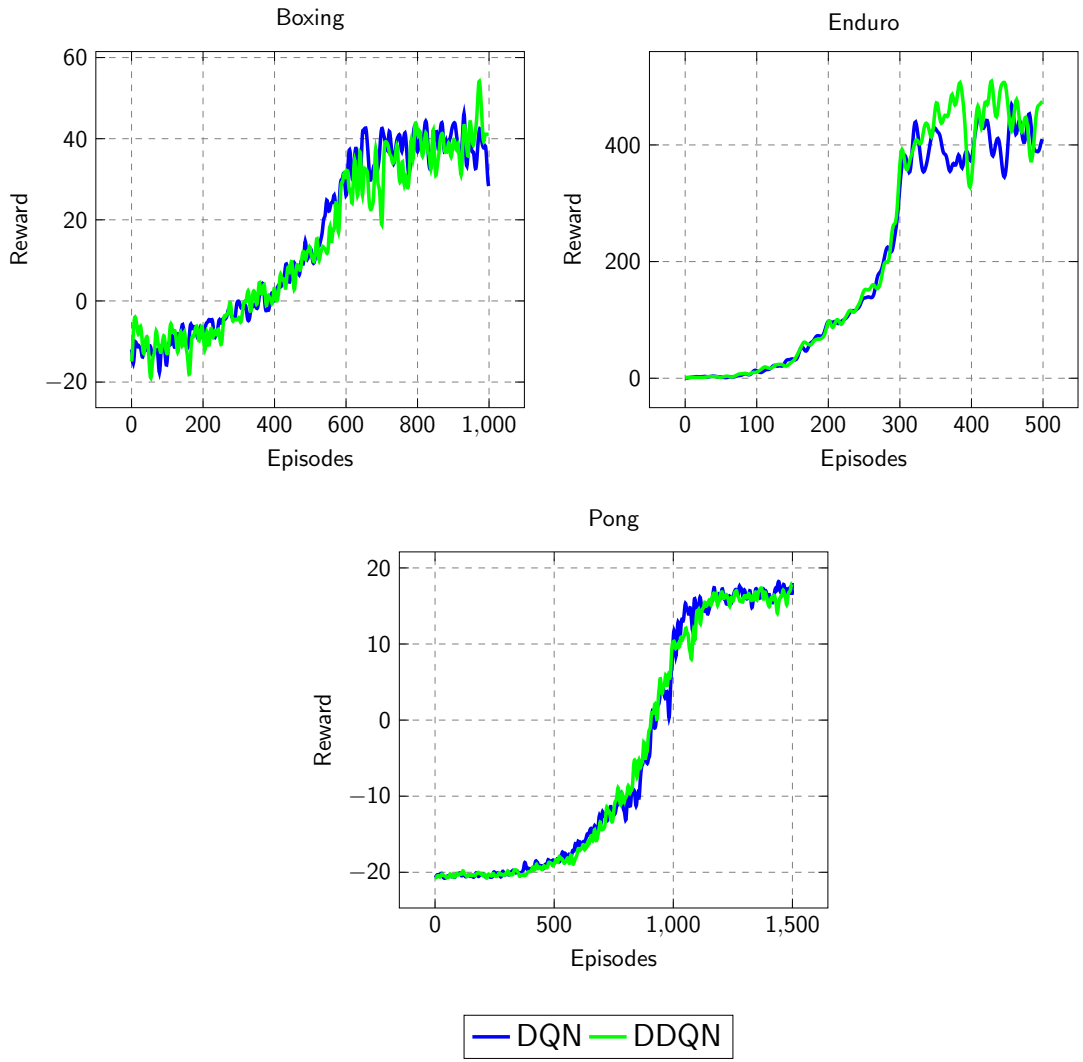
$$L(\theta) = \mathbb{E}_{\langle s_t, a_t, r_t, s_{t+1}\rangle \sim U(D)}\left[\left(r_t + \gamma V(s_{t+1}; \Phi) - Q(s_t, a_t; \theta)\right)^2\right].$$

A Novel Family of Deep Reinforcement Learning Algorithms

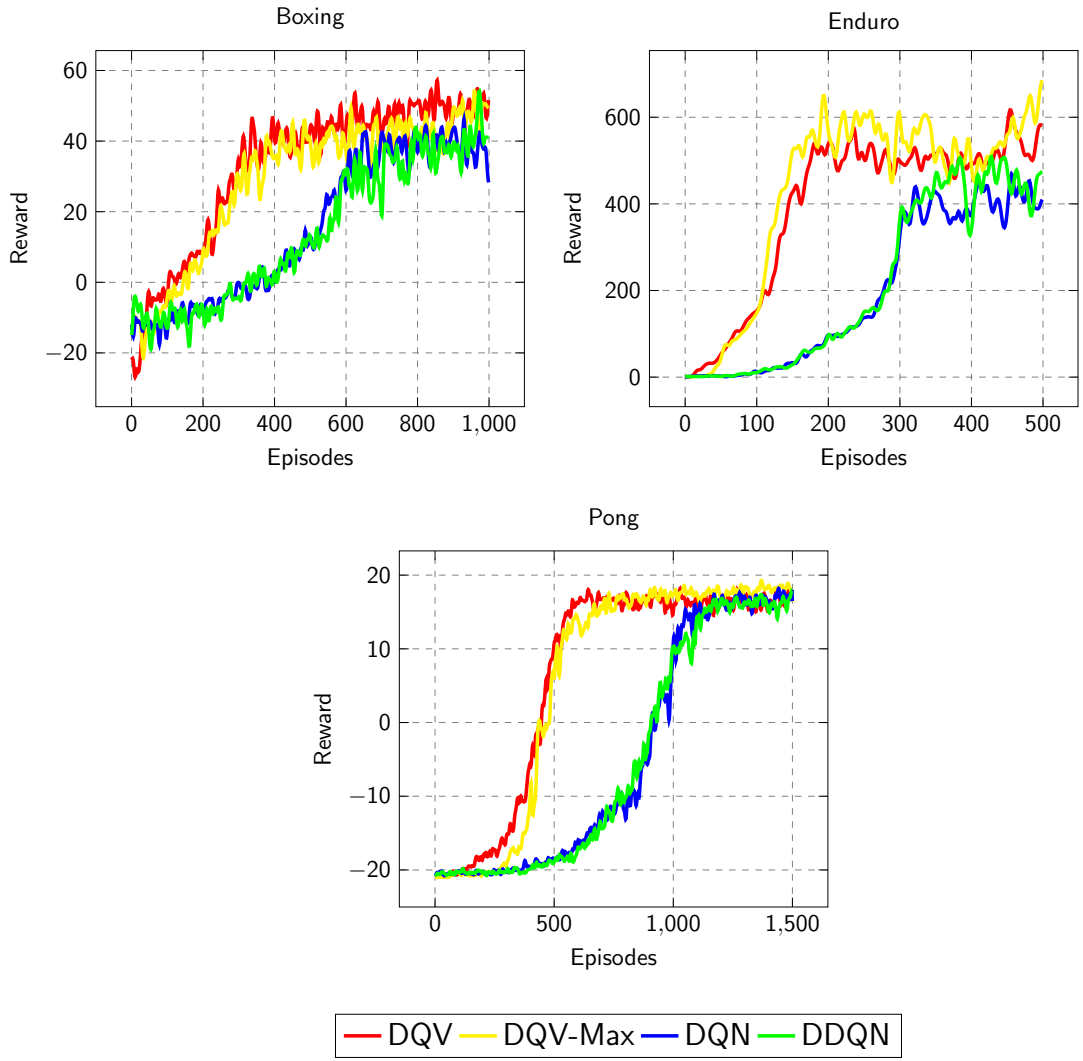
The first two objective functions constitute the **DQV-Learning** algorithm, whereas the second two objective functions define the **DQV-Max Learning** algorithm.

We compare their performance to algorithms which only learn an approximation of the state-action value function $Q^{\pi}(s, a)$

A Novel Family of Deep Reinforcement Learning Algorithms

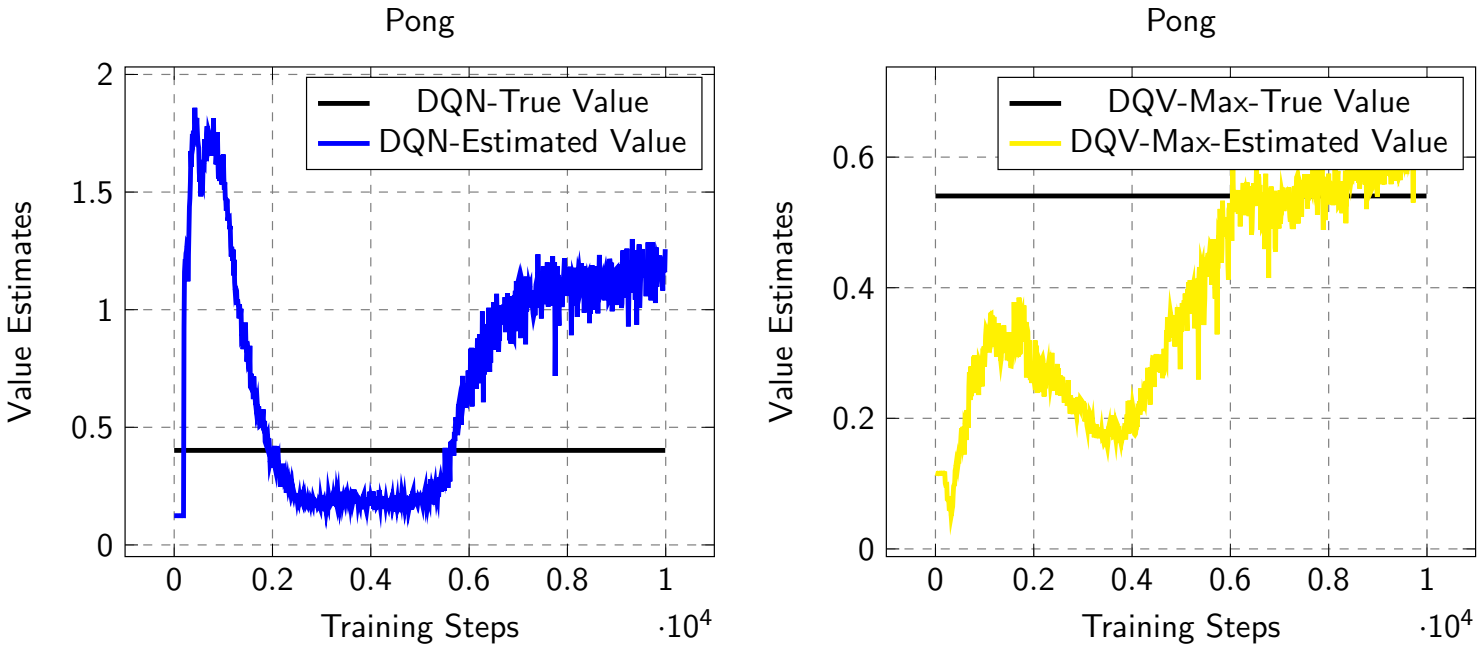


A Novel Family of Deep Reinforcement Learning Algorithms



A Novel Family of Deep Reinforcement Learning Algorithms

$\Rightarrow$ Jointly approximating two value functions over one results in faster convergence, but also in accurate value estimates:



A Novel Family of Deep Reinforcement Learning Algorithms

To conclude DQV-Learning and DQV-Max Learning

- Result in faster learning
- Are robust to the "Deadly Triad of DRL"
- Suffer less from the overestimation bias of the Q function
- Scale well to the multi-agent setting [2]

However ...

- Memory wise are twice as expensive as DQN and DDQN
- Do not always result in better policies

[2] Leroy, Pascal, et al. "QVMix and QVMix-Max: Extending the Deep Quality-Value Family of Algorithms to Cooperative Multi-Agent Reinforcement Learning." (2021).

Final References