AFRL-RI-RS-TR-2019-125

DEEP LIFELONG REINFORCEMENT LEARNING FOR RESILIENT CONTROL AND COORDINATION

TRUSTEES OF THE UNIVERSITY OF PENNSYLVANIA

JUNE 2019

FINAL TECHNICAL REPORT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

STINFO COPY

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE

■ **AIR FORCE MATERIEL COMMAND** ■ **UNITED STATES AIR FORCE** ■ **ROME, NY 13441**

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09. This report is available to the general public, including foreign nations. Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-RI-RS-TR-2019-125 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE CHIEF ENGINEER:

/ S /
WARREN GEILER
Work Unit Manager

/ S /
JULIE BRICHACEK
Chief, Information Systems Division
Information Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
JUNE 2019	FINAL TECHNICAL REPORT	SEP 2016 – DEC 2018

4. TITLE AND SUBTITLE

DEEP LIFELONG REINFORCEMENT LEARNING FOR RESILIENT CONTROL AND COORDINATION

5a. CONTRACT NUMBER
N/A

5b. GRANT NUMBER
FA8750-16-1-0109

5c. PROGRAM ELEMENT NUMBER
62788F

6. AUTHOR(S)

Seungwon Lee, James Stokes, Eric Eaton

5d. PROJECT NUMBER
S2MY

5e. TASK NUMBER
RA

5f. WORK UNIT NUMBER
SP

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Trustees of the University of Pennsylvania
Office of Research Services
3451 Walnut Street, Room P-221
Philadelphia, PA 19104-6205

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Research Laboratory/RISC
525 Brooks Road
Rome NY 13441-4505

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/RI

11. SPONSOR/MONITOR'S REPORT NUMBER
AFRL-RI-RS-TR-2019-125

12. DISTRIBUTION AVAILABILITY STATEMENT

Approved for Public Release; Distribution Unlimited. This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The objective of this effort was to develop deep lifelong learning methods that can successfully handle sequential decision making in complex dynamic environments, focusing on multi-agent intelligence, surveillance, and reconnaissance (ISR) scenarios. We developed a novel architecture for deep convolutional neural networks that supports lifelong learning via deconvolutional factorization (DF-CNN), explored a combination of policy distillation via Distral and Sobolev training, and developed a hybrid controller for applying deep learning to ISR agents. Our approaches were evaluated on standard benchmark deep learning datasets, the DOOM environment, and on ISR scenarios in the ATE3 simulation environment.

15. SUBJECT TERMS

Machine learning, lifelong learning, deep learning, reinforcement learning, convolutional neural networks

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			**WARREN GEILER**
U	U	U	UU	30	19b. TELEPHONE NUMBER (*Include area code*) **N/A**

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1.0 SUMMARY

The objective of this effort was to develop deep lifelong learning methods that can successfully handle sequential decision making in complex dynamic environments, focusing on multi-agent intelligence, surveillance, and reconnaissance (ISR) scenarios. We developed a novel architecture for deep convolutional neural networks that supports lifelong learning via deconvolutional factorization (DF-CNN), explored a combination of policy distillation via Distral and Sobolev training, and developed a hybrid controller for applying deep learning to ISR agents. Our approaches were evaluated on standard benchmark deep learning datasets, the DOOM environment, and on ISR scenarios in the ATE^3 simulation environment.

Our primary contribution is the Deconvolutional Factorized Convolutional Neural Network (DF-CNN). The DF-CNN framework adapts the standard convolutional neural network (CNN) framework to enable transfer between tasks. It maintains a shared knowledge base at each CNN layer, and facilitates transfer between different task-specific CNNs via this shared knowledge. The individual filter layers for each specific task's CNN model are reconstructed from this shared knowledge base, which is adapted over time as the network is trained over multiple tasks. The DF-CNN represents the generalization of the ELLA lifelong learning framework to deep networks.

Experiments showed that the DF-CNN outperformed other approaches (including single-task learning, hard-parameter sharing of the lower layers, and progressive neural networks) on benchmark recognition tasks in lifelong scenarios. Moreover, the framework is resistant to catastrophic forgetting while still permitting reverse transfer to previously learned models from future learning.

For deep reinforcement learning, we investigated the integration of Sobolev training into the Distral multi-task framework in an effort to improve transfer and training, explored the use of the DF-CNN for deep RL, and developed a hybrid controller that combined locally learned deep RL policies together to complete ISR scenarios in the ATE^3 simulation environment.

2.0 INTRODUCTION

Recent progress combining deep learning (DL) with reinforcement learning (RL) has achieved several groundbreaking results in artificial intelligence, including deep Q-learning that can achieve human-level performance in Atari games [Mnih et al., 2015], and AlphaGo [Silver et al., 2016] winning against a top-ranked human Go player. However, despite their empirical success, current DRL methods require extensive amounts of offline training through repeated interaction with the environment (e.g., over a millions hours of parallel compute time for one Atari game [Mnih et al., 2015]) to learn high performance policies. In operational environments, such repeated interactions to learn an optimal policy are simply impossible—it is feasible to learn on an (imperfect) simulated environment, but any bootstrapped policy would still need to be adapted rapidly online when deployed in a novel real scenario, making direct usage of DRL methods impractical for all but the simplest tasks in real environments.

Lifelong transfer learning provides a mechanism to avoid this problem by efficiently and rapidly reusing previous experience. This process of knowledge transfer is inherent in human and animal learning: we rapidly learn novel tasks, often with only a few repetitions, by building upon a lifetime of experience and acquired skills. Recent advances in lifelong machine learning [Ruvolo & Eaton, 2013; Bou Ammar et al., 2014] have enabled this same ability in automated systems, enabling the rapid learning of multiple, consecutive tasks in classification, regression, and reinforcement learning domains. However, these lifelong learning methods were limited at the start of this project to relatively simple models (e.g., linear or logistic regression) and control policies (e.g., parameterized Gaussian policies). Consequently, such lifelong learning systems cannot handle complex sequential decision making environments.

The objective of this effort was to develop deep lifelong learning methods that can successfully handle sequential decision making in complex dynamic environments, focusing on multi-agent intelligence, surveillance, and reconnaissance (ISR) scenarios. By incorporating lifelong knowledge transfer in deep reinforcement learning, our approach can acquire deep hierarchical knowledge representations over multiple, consecutive tasks. When faced with a novel ISR scenario, the resulting deep lifelong RL methods will rapidly learn a policy for the new scenario, reusing knowledge at the appropriate level of abstraction to minimize the amount of new data needed from the environment. The developed methods were applied to and evaluated on simulated multi-agent scenarios in the ATE^3 simulation environment.

2.1 Related Work

This section briefly surveys related work on deep multi-task and lifelong learning that we will build upon in the remainder of this report.

Deep multi-task and lifelong learning

Previously proposed deep learning work for multi-task and lifelong learning can be classified into four categories. Figure 1 shows some representative models of such previous work. These four categories include:

Explicit weight sharing. Hard parameter sharing (HPS) is simple but basic idea to share knowledge across tasks in deep learning [Caruana 1997; Ranjan et. al. 2015; Huang et. al. 2013; Bell & Renals 2015]: explicitly sharing lower layers of the network for feature extraction and building higher layers of the network to be task-specific. The explicit sharing of lower layers of the network forces them to learn universal features for multiple tasks and task-specific layers learn mapping from the universal features to the output of each task. The lowest layers can also be task-specific to allow the network dealing with various input domain of tasks, but the core idea of sharing some layers of the network across tasks is still invariant.

One variant of this method not only used shared lower layers of the network but correlated task-specific layers by a tensor normal distribution to learn the relations between tasks [Long et. al. 2017]. Furthermore, there are methods which automatically add new hidden units, split them into disjoint groups for different feature spaces, and merge groups of hidden units to encourage transfer between tasks [Lu et. al. 2017; Yoon et. al. 2018]. These methods allows more flexible transfer between tasks than HPS, but they still explicitly share lower layers across all or partial sets of tasks, which only support restricted form of task relationships.

Pipelined transfer. Instead of using tree-structured networks for multi-task and lifelong learning by sharing layers, another approach trains task-specific networks which have lateral connection from networks of other tasks to utilize learned features for those tasks [Misra et. al. 2016; Gao et. al. 2018; Pinto & Gupta 2017; Liu et. al. 2017a; Rusu et. al. 2016]. This architecture enables networks to learn and maintain low- and high-level features for their own tasks, so it is more robust to handling diverse tasks and catastrophic forgetting than explicit weight sharing. Despite these benefit, this approach can also learn restricted forms of task correlation because it reuses only features of previous tasks. Moreover, the size of the total network and number of cross-task lateral connections increase at most quadratically with the number of tasks.

Shared knowledge base. This approach correlates task-specific networks through shared matrices or tensors which are serve as a knowledge base. Sharable detectors for face alignment use the sparse representation of a shared basis to learn both a universal basis and weights of networks for tasks [Liu et. al. 2017b]. Mathematical methods of tensor decomposition is also a possible method to define the knowledge base and the relation between knowledge base and weights of task-specific networks [Yang & Hospedales 2017].

Dynamic filters. Dynamic filters method trains one independent neural network which generates weights of a network for the task according to the given input [Brabandere et. al. 2016; Ha et. al. 2017]. This approach is able to learn more abstract relationship between multiple tasks than the approach of using a shared knowledge base because of the expressive power of the weight-generating network.

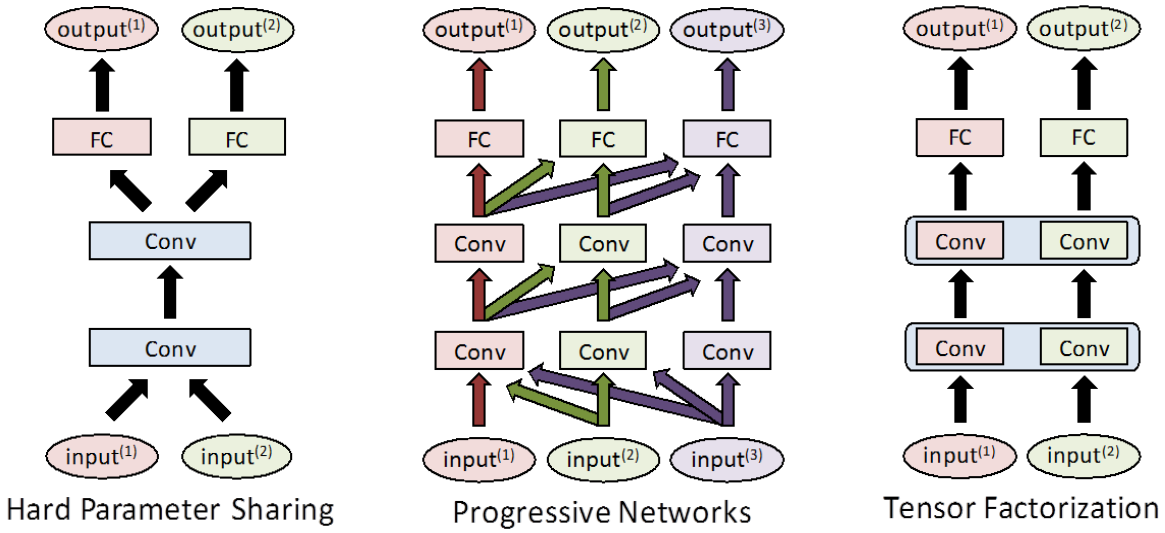


Figure 1 - Architectures for Deep Multi-task and Lifelong Learning

Policy Transfer for Multi-task Deep Reinforcement Learning

Transfer of knowledge between tasks in the area of deep reinforcement learning can follow the approaches introduced in the previous section, which share knowledge of useful features. However, it is also possible to constrain the action behavior of task-specific networks to be correlated to each other and make these learned policies to be similar. This approach shows improvements in performance of the learned policies when tasks are less common in the observation space but require similar control, such as driving a car in a desert and a city.

Policy distillation [Rusu et. al. 2015; Parisotto et. al. 2016] applies the technique of knowledge distillation in the area of deep learning to train one network for multiple tasks from supervision of

multiple other task networks. This method reduces the size of a neural network and is also able to improve the performance of policies encoded in the final network, but these achievements are sub-optimal with respect to multi-task learning because it is not possible to boost the speed of training teacher policies, which must be learned independently. Also, the teacher policy itself cannot take advantage of the knowledge of other tasks for better performance during its training. Therefore, Distral (**dis**till and **t**ransfer **l**earning) [Teh et. al. 2017] modifies policy distillation into multi-task learning by training both a central policy and policies for individual tasks simultaneously.

3.0 METHODS, ASSUMPTIONS, PROCEDURES

Our approach is divided into three methods: a novel architecture for deep convolutional neural networks that supports lifelong learning via deconvolutional factorization (DF-CNN), a combination of policy distillation via Distral and Sobolev training, and a hybrid controller for applying deep learning to ISR agents.

3.1 Sharing Learned Knowledge via Deconvolution Networks

A multi-task and lifelong learning system faces a set of tasks in batch or sequentially, and must train a model for each task such as a classifier [Chen and Liu 2016]. For tasks of visual perception, a convolutional neural network (CNN) is widely applied to learn useful features. The related work described previously are designed to transfer learned features or knowledge across a model for each task, but they are general methods which ignore unique characteristics of the convolutional layer.

In this section, we propose an approach to lifelong learning using convolutional neural networks. Our proposed approach, called a deconvolutional factorized CNN (DF-CNN), seeks to address the lifelong learning problem using deep convolutional networks with a shared knowledge base to enable transfer between the tasks (Figure 2). In the DF-CNN, each learning task admits an associated convolutional neural network that is independently trained on labeled data for that task. Recall that a CNN is composed of multiple layers of stacked filters, each of which is parameterized. To facilitate transfer between tasks, our architecture maintains a shared latent knowledge base that connects the various layers across the task-specific CNNs. The filters of the CNNs are formed by applying the deconvolution operator (transposed convolution) to the learned latent knowledge base, followed by a tensor contraction. Unlike previous methods that involve tensor factorization to achieve sparsity, our proposal is naturally sparse by virtue of the deconvolution operator.

Factorized transfer via deconvolution

For each task-specific convolutional network with L layers, the l-th convolutional layer has the filter ${W_t}^{(l)}$ of size $h \times w \times c_{in} \times c_{out}$ where h and w are height and width of the filter, and c_{in} and $c_{out}$are the numbers of input and output channels. A collection of filters of the l-th convolutional layer of$T$ task-specific networks, $[{W_1}^{(l)}, {W_2}^{(l)}, \ldots, {W_T}^{(l)}]$, is a 5th-order tensor, so the tensor decomposition approach [Liu et. al 2017; Yang & Hospedales 2017] factorizes this aggregated filters into a tensor shared across tasks ($L^{(l)}$) and a set of task-specific tensors $({S_1}^{(l)}, \ldots, {S_T}^{(l)})$ such as ${W_t}^{(l)} = L^{(l)} {S_t}^{(l)}$.

Instead of using a general mathematical tool such as tensor decomposition, we utilize a deconvolutional mapping and tensor contraction to factorize the task-specific filter into the

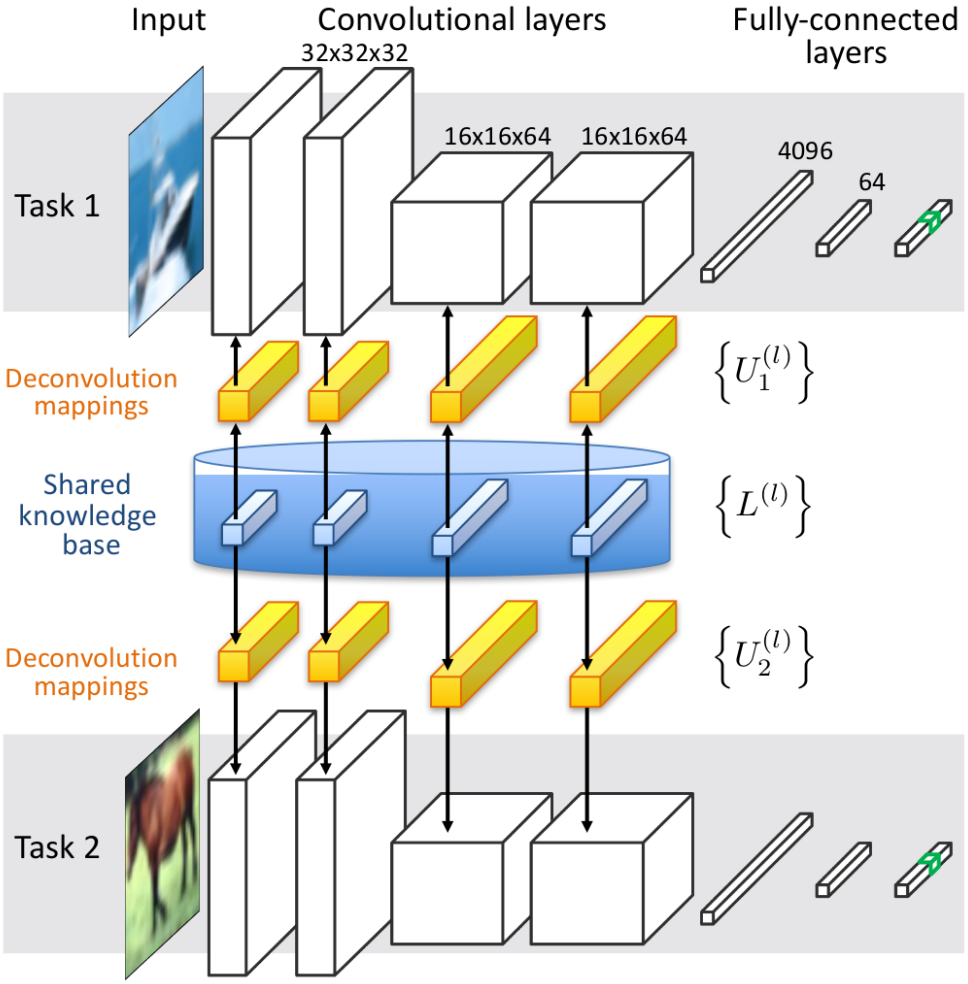


Figure 2 - Our Deconvolutional Factorized CNN Architecture

shared knowledge base ($L^{(l)}$), which is a 3rd-order tensor of size $\hat{h} \times \hat{w} \times \hat{c}$. We first deconvolve the shared knowledge base into

$$D_t^{(l)} = Deconv\left(L^{(l)}; V_t^{(l)}\right) \quad (1)$$

where $D_t^{(l)}$ is a 3rd-order tensor of size $h \times w \times c$, $V_t^{(l)}$is the task-dependent deconvolutional filter of size $p \times p \times \hat{c} \times c$, and p is the spatial size of the deconvolutional filter. We then apply tensor contraction to construct each convolutional filter $W_t^{(l)}$based on $D_t^{(l)}$:

$$W_t^{(l)} = D_t^{(l)} \cdot V_t^{(l)} = \sum_{k=1}^{c} D_{t,(\cdot,\cdot,k)}^{(l)} \; U_{t,(k,\cdot,\cdot)}^{(l)} \quad (2)$$

where $U_t^{(l)}$ is a 3rd-order tensor of size $c \times c_{in} \times c_{out}$, and both subscripts $(k,\cdot,\cdot)$ and $(\cdot,\cdot,k)$ express the elements' index in the tensor. The tensor contraction formulize the filter as a linear combination of the basis vectors $D_t^{(l)}$ by changing the size of channels.

The shared knowledge base is a tensor with small size compared to the filters of task-specific convolutional layers, both in terms of the spatial axis ($h \times w$) and channel ($c_{in} \times c_{out}$). Rather than applying the same type of operation to expand the knowledge into a large task-specific filter, two-staged expansion of the knowledge base by deconvolution and tensor contraction distinguishes between the transfer process along the spatial axis of images and along the channels of images.

Training the DF-CNN

Our proposed architecture must learn both the shared knowledge bases $L^{(l)}$ and task-specific transformation$V_t^{(l)}$ and $U_t^{(l)}$ from data of each task in a lifelong learning setting. The architecture can be trained end-to-end via gradient-based optimization.

The shared knowledge bases $\{L^{(l)}\}_{l=1}^{L}$ are randomly initialized prior to training on the first task, while task-specific transformations $\{(V_t^{(l)}, U_t^{(l)})\}_{l=1}^{L}$are initialized randomly when the training data for the task (labeled t) is first provided. While training on the task T, the knowledge bases and knowledge transformations for the task T are updated according to the observed training instances, but transformations of previously observed tasks $t < T$ are held fixed. Since the convolutional filters for each task-specific networks are generated dynamically from the shared knowledge base, update of the knowledge base can affect the performance of previously trained networks, which is known as reverse transfer [Ruvolo & Eaton 2013]. Catastrophic forgetting which is severe negative reverse transfer commonly occurs in deep lifelong learning. There are no explicit mechanisms designed to prevent catastrophic forgetting (such as [Rusu et. al. 2016]) in our architecture, but deconvolutional factorization of the task-specific models' parameter space empirically avoids catastrophic forgetting and exhibits positive reverse transfer. While we focused on developing the DF-CNN for supervised settings, due to its simplicity, it can also operate in reinforcement learning settings, which is a focus of our current work.

3.2 Sobolev Training for Multi-task Reinforcement Learning

In addition to developing the DF-CNN method, we also investigated mechanisms for improving transfer and learning speed by combining policy distillation in Distral and the use of Sobolev training, which incorporates derivatives of the target output into the training of deep networks.

Distral

Distral [Teh et. al. 2017] is a framework for multi-task reinforcement learning by enforcing the similarity of action policies between tasks on the assumption of tasks having the same state S and action A spaces. This method trains both task-specific policies and the central policy. The central policy distills common action behaviors from task-specific policies, and the task-specific policies maximize reward in tasks via interaction with the associated environment and knowledge transferred from other tasks via the central policy.

Let π_0 and π_i be the distilled policy (central policy) and task-specific policy for task i, respectively. If each task i has transition dynamics $p_i(s_{t+1}|\, s_t, a_t)$ and reward functions $R_i(s_t, a_t)$, the Distral mechanism optimizes the following objective:

$$J(\pi_0, \{\pi_i\}_{i=1}{}^{n}) = \sum_i p_i(s_{t+1}|\, s_t, a_t) \left[\sum_t \gamma^t R_i(s_t, a_t) - c_{KL}\, \gamma^t \log \frac{\pi_i(a_t|s_t)}{\pi_0(a_t|s_t)} - c_{ent}\, \gamma^t \log \pi_i(a_t|s_t) \right] \quad (3)$$

where γ is discount factor, c_{KL} and c_{ent} are coefficient weighting the Kullback-Leibler (KL) divergence between the central and task-specific policies and the entropy of task-specific policies. In this objective, KL divergence is minimized to enforce the similarity of policies while entropy is maximized to encourage exploration.

Sobolev training

Sobolev training [Czarnecki et. al. 2017] assumes that learning of a function f has access to the value of derivatives of multiple orders with respect to the input, $D_x{}^j f(x_i)$, as well as the output values $f(x_i)$ for training points x_i. The typical learning of the function optimizes a neural network model according to a set of pairs $\{(x_i, f(x_i))\}_{i=1}{}^{N}$, but the Sobolev training optimizes the model according to a set of $K+2$ tuples $\{(x_i, f(x_i), D_X{}^1 f(x_i), \cdots, D_X{}^K f(x_i))\}_{i=1}{}^{N}$. The cost function of learning in Sobolev spaces sums loss functions of the function f and derivatives.

This method showed empirical improvement in the trained model for both regression tasks and distillation of reinforcement learning tasks. It achieved less error than a training mechanism which uses only the input-output pairs with relatively small amounts of training data.

Combining Distral and Sobolev training

Incorporating Sobolev training into Distral is straightforward because the only change from the original Distral method is the addition of new loss terms related to the derivatives into the objective function of Distral (Equation 3). Instead of training only action policies for tasks, we applied both Distral and Sobolev approaches to actor-critic method for reinforcement learning which learns optimal policy and value function simultaneously. For the optimization, we introduced the error of the first-order derivative of the policy and value functions into the objective function; the type of error function for the value functions and derivatives was the sum-of-squares error (L2 loss).

In addition to the aforementioned changes for combining Distral and Sobolev training, we developed a variant of Distral which applies regularization on task-specific policies at the level of the last hidden layers of neural networks rather than the output layers. This alternative is able to circumvent the major assumption of Distral method requiring all tasks to have same action spaces.

3.3 Hierarchical Control of Multiple Agents in Environments with Sparse Reward

To apply our developed deep learning agents to the ATE[3] simulator, which provides a very sparse reward signal, we developed a hybrid hierarchical controller for controlling the unmanned aerial vehicles (UAVs). This section describes the hybrid controller for multiple drones using a finite state automaton and a deep neural network in Figure 3.

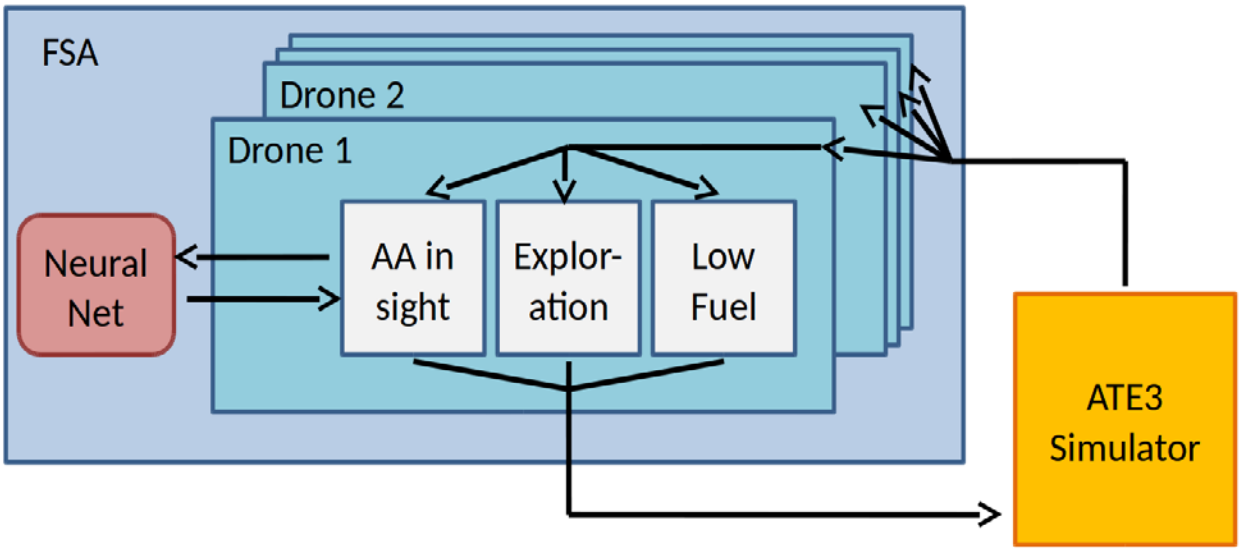


Figure 3 - The Hybrid Controller for the ISR UAV Asset, Which Consists of a Finite State Automaton and a Deep RL Agent

Hierarchy of control agents

Each neural network learns a single policy and functions best when applied to a focused scenario. Consequently, it is not currently feasible to train a single deep network to complete the entire ISR scenario, which is large and complex. Instead we trained neural networks for multiple local control policies, and designed a finite state automaton for abstract and global control to coordinate across these policies. An alternative approach would be to use a hierarchical deep reinforcement learning agent, such as a FeUdal network [Vezhnevets et. al. 2017], which we leave to future work.

The deep neural network controller

We trained a deep neural network to avoid hostiles in the vicinity of mobile anti-aircraft (AA) in the simulation. The observation space of the drone is sparse in comparison with video games, which are typical benchmarks of deep reinforcement learning, because the observation space in ATE^3 consists of the list of observed entities, such as a intel target, a jammer or a mobile AA. Because of the characteristics of observation space, we firstly processed the observation of each drone to a vector of pre-specified numbers of intel targets, jammers, and mobile AAs. We set a positive reward for the case of detecting the intel target and a negative reward for the case of losing the drone. When running the scenarios, even aggregating the rewards across multiple drones could not obtain a frequent reward signal, and the deep neural network easily failed to learn optimal control. To compensate for this problem, we used intrinsic reward [Pathak et. al. 2017] to adapt to the sparse reward from the environment. The intrinsic reward provides a 'curiosity' signal, defined by the discrepancy between the environment and the learned model of it, so the intrinsic reward encourages the exploration of the agent in the early phase of training.

This local controller is trained on the simplified scenario of 10 minute-lengths shown in Figure 4. This scenario has only drones and mobile AAs, and each drone receives positive reward when it reaches the goal region while getting a negative reward on the loss of the drone. Because of the time limit of the scenario, the neural network agent must learn the policy which makes the drone move around the hostile and reach the opposite side of the AA region as fast as possible.

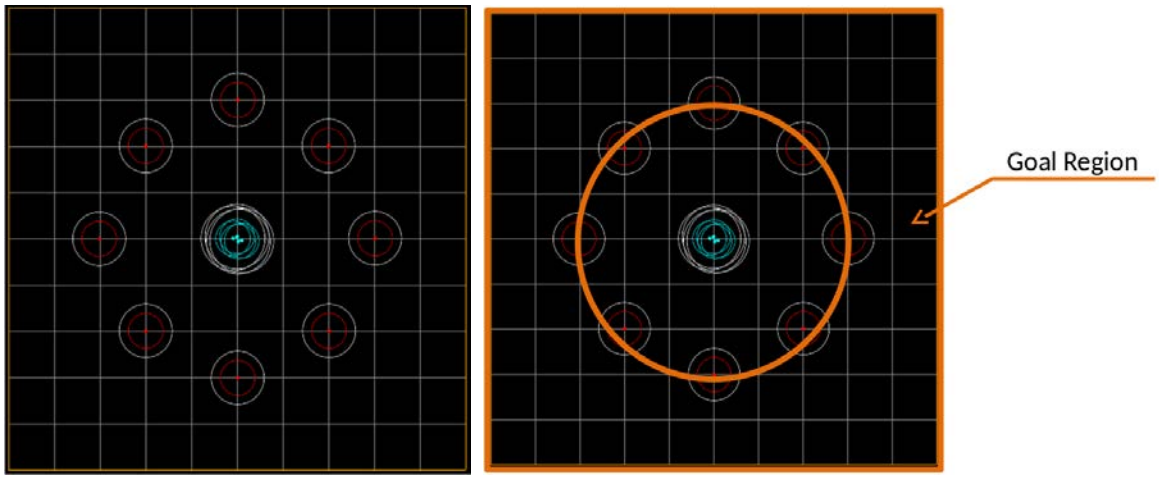


Figure 4 - The Simplified Scenario for Training the Neural Network Controller

Finite State Controller

We used a finite state controller to govern the high-level behavior of the drone, which employs local policies trained via deep learning to handle different situations. To maximize the speed of exploration at the beginning, the controller sends all drones in equal directions with a randomly chosen distance to move. After the spread of drones, the controller moves them in a counter-clockwise direction to explore the unobserved area. Whenever the level of fuel of a drone goes 30% or below, the controller sets home base as the goal location of the drone to refuel it. Any local situations the drone encounters are governed by the trained policies.

4.0 RESULTS AND DISCUSSION

This section presents our evaluation and results on each of the methods described above.

4.1 Evaluation of DF-CNN on Lifelong Learning Scenarios

We evaluated our DF-CNN against a variety of alternative methods in lifelong learning scenarios using two visual recognition data sets: CIFAR-100 [Krizhevsky & Hinton 2009] and Office-Home [Venkateswara et. al. 2017]. Due to its simplicity, we started with evaluating these methods on classification scenarios, moving later to deep reinforcement learning.

Baseline Approaches

We compared our proposed approach to the alternative methods described in Section 2.1:

Single-task learning (STL) trains a neural network for each task that is independent from networks for all other tasks. This method has a clear disadvantage below transfer methods in the few-data or noisy data regime.

Hard parameter sharing (HPS) [Caruana 1997] shares the lower layers of the network across tasks, with separate task-specific layers for the output. A heuristic which we follow is that all convolutional layers are shared while all fully-connected layers are task-specific. This model is expected to show its strength when tasks share a common set of useful features, but may fail when tasks are sufficiently dissimilar.

Progressive neural networks (ProgNN) [Rusu et. al. 2016] enable each task model to exploit learned features of its predecessors. This approach was firstly designed for knowledge transfer in reinforcement learning, but it was evaluated in supervised setting in our experiments.

Experimental Setup

We built two lifelong learning problems using the CIFAR-100 and Office-Home data sets. For the CIFAR-100, we created 10 image classification tasks of ten distinct classes. To follow the assumption of limited training data [Chen & Liu 2016], we sampled only 4% of the available training data, and split it into training and validation sets in the ratio of 5.6:1 (170 training and 30 validation instances per task). We used all test images in the CIFAR-100 for the test set of the lifelong learning tasks (1,000 instances per task).

The Office-Home dataset originally has four different domains with the same 65 classes of images, and we focused on two of these domains: Product images and Real-World images. The Product domain has each object at the center with white background, while the Real-World domain has each object in various background. We created 5 image classification tasks from each domain, resulting in 10 tasks with 13 distinct classes of images per task. The original dataset has no pre-specified training/validation/test split, so we randomly split the data into those with a 60% : 10% : 30% ratio (approximately 550 training, 90 validation and 250 test instances).

All models were trained end-to-end on only one task at any moment, and the task was switched to the next one after every 2,000 (CIFAR-100) and 1,000 (Office-Home) training epochs, regardless of the model's convergence. The optimal hyper-parameters for each approach were determined by the accuracy on the validation sets. All baselines and our proposed models have the same architectural hyper-parameters, such as the number of convolutional layers and the size of convolutional filters. Note that the regular STL has 3.28M parameters for CIFAR-100 and 26.8M parameters for Office-Home, and the regular DF-CNN has 7.96M parameters for CIFAR-100. We also evaluated a larger STL, which has 9.35M parameters for CIFAR-100 and 129M parameters for Office-Home in total, and a reduced-size DF-CNN with 2.8M parameters for CIFAR-100.

We assessed performance of these models by measuring the following metrics on the held-out test set for all tasks:

- *Peak Per-Task Accuracy*: The best test accuracy of each task during its training phase. This metric focuses on the model's performance on the currently learned task.
- *Catastrophic Forgetting Ratio*: The ratio of a task's test accuracy after training on subsequent tasks to its peak per-task accuracy. This ratio shows how much the model maintain its performance on older tasks.
- *Convergence*: The number of training epochs for the test accuracy to reach 98% of the peak per-task accuracy of the task. This number of epochs shows the effect of knowledge transfer from previously learned tasks.

Results

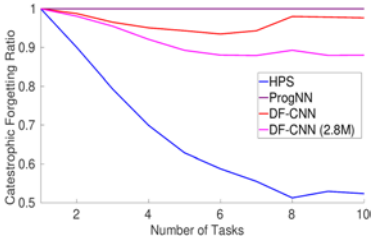
The performance of all approaches is summarized in Figures 5-6 (CIFAR-100) and Figure 7 (Office-Home). For the CIFAR-100, the test accuracy of each task model over training epochs, averaged over 5 trials, is visualized in Figure 6. To explore the effect of learning subsequent tasks on previous task models, we repeatedly evaluated performance on the previous task. Significant decreases in task performance after switching from the current task indicates catastrophic forgetting; increases in performance indicate positive reverse transfer.

We can clearly see that HPS suffers from catastrophic forgetting as the shared layers were adapted to new tasks, as shown by the rapid decline in performance once learning on each task finishes

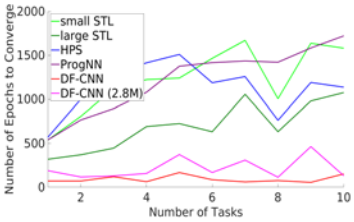
(Figure 5(b), 6(a) and 7(b)). Additionally, HPS could not achieve a peak per-task accuracy comparable to or better than that of STL consistently, and it converged slowly to its peak per-task accuracy. This means that the adaptation of the shared layers of HPS is not guaranteed to have a positive effect on training to both current and previous tasks.

Model	Peak Acc.	Time (10k sec)
STL (small)	31.2%±2.2	4.99±0.007
STL (large)	34.3%±1.1	5.46±0.005
HPS	24.7%±0.6	5.14±0.014
ProgNN	31.3%±1.1	9.21±0.019
DF-CNN	37.2%±1.3	6.66±0.013
DF-CNN (2.8M)	36.2%±3.5	5.34±0.008

(a) Peak per-task accuracy and training time with 95% confidence intervals

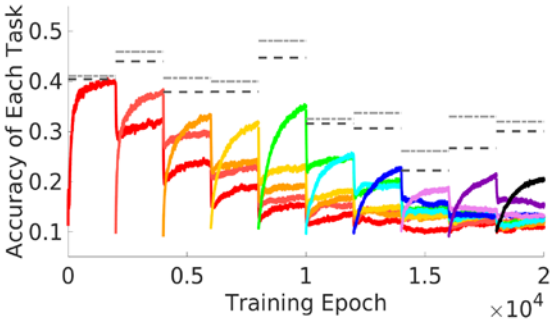


(b) Catastrophic forgetting ratio

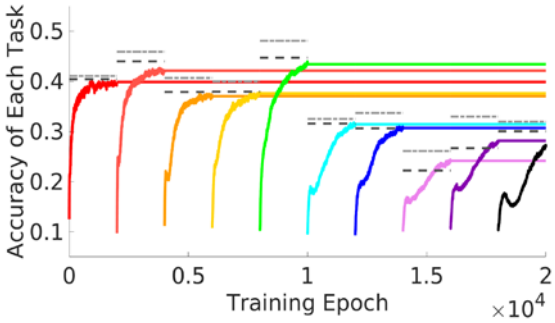


(c) Speed of convergence

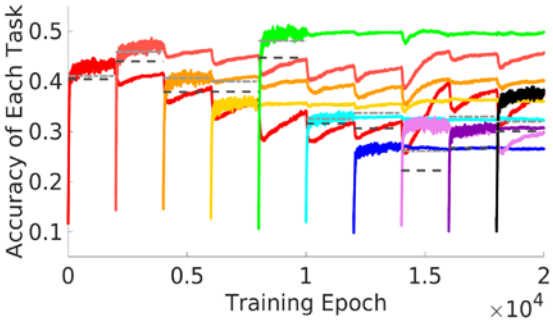
Figure 5 - Performance Metrics of Models on CIFAR-100 Lifelong Learning Tasks, Averaged Over All Tasks



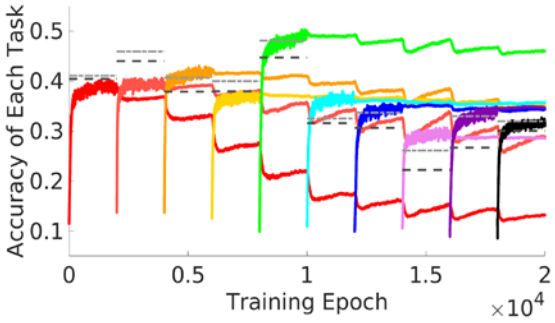
(a) Hard Parameter Sharing (2.69M parameters total)



(b) Progressive Neural Net (3.51M parameters total)



(c) DF-CNN (7.96M parameters total)

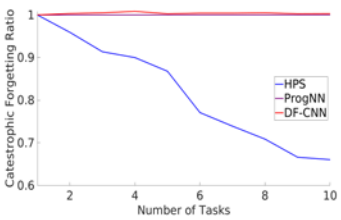


(b) DF-CNN (2.8M parameters total)

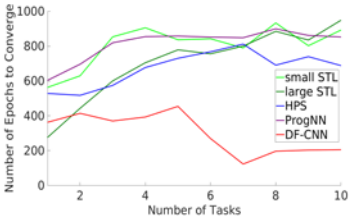
Figure 6 - Mean Test Accuracy in Lifelong Learning on CIFAR-100. Each Color Corresponds to One Task by Presentation Order

Model	Peak Acc.	Time (10k sec)
STL (small)	45.5%±0.5	3.79 ± 0.009
STL (large)	51.9%±0.9	6.09 ± 0.005
HPS	52.0%±0.7	3.79 ± 0.012
ProgNN	46.4%±1.0	11.7 ± 0.003
DF-CNN	49.1%±0.6	4.11±0.004

(a) Peak per-task accuracy and training time with 95% confidence intervals



(b) Catastrophic forgetting ratio



(c) Speed of convergence

Figure 7 - Performance Metrics of Models on Office-Home Lifelong Learning Tasks, Averaged Over All Tasks

In contrast to HPS, ProgNN was able to preserve its performance on previous tasks after learning new tasks by virtue of its design. Lateral connections of ProgNN that reuse previously learned features caused improvement of learning speed and test accuracy in a few tasks, such as the 7th -- 9th tasks of the CIFAR-100 experiment, but the benefit of the lateral connections was marginal in comparison with STL. Furthermore, training the ProgNN takes approximately twice as much training time as others.

DF-CNN showed significant improvement in peak per-task accuracy over STL, HPS, and ProgNN for the CIFAR-100 experiment, and achieved peak per-task accuracy better than STL for the Office-Home experiment. Moreover, DF-CNN converged to its peak per-task accuracy more than twice as fast as other models. These improvement in peak per-task accuracy and speed of convergence show the positive effect of knowledge transfer within the DF-CNN.

Previous task models of DF-CNN deteriorated slightly as it trained on new tasks because of the update of the shared knowledge base without consideration to previous tasks. However, the rate of performance loss on the earliest task models is much slower than HPS, and DF-CNN recovered its performance for those tasks over time. Especially, in the CIFAR-100 experiment, the performance of the earliest task models has the most degradation, and the performance on the later tasks maintains almost constant post-training accuracy when the shared knowledge base became mature, such as for 4th -- 10th tasks. Additionally, during the training on the 8th task of the CIFAR-100 experiment, we can find positive reverse transfer from new to old tasks, which had not been observed in the training of other approaches.

The reduced-size DF-CNN lost performance on the first task catastrophically, because of its reduced capacity of the shared knowledge base. Even with the limited capacity of both the shared knowledge and task-specific knowledge transformation, the reduced-size DF-CNN still showed improvement in accuracy, speed of convergence, and robust retention of performance on previous tasks as compared to the baselines. These results support the benefit of knowledge transfer through the shared knowledge and deconvolutional mapping.

4.2 Evaluation of Distral and Sobolev Training

We evaluated our combination of Distral and Sobolev training using the Doom environments of OpenAI Gym. Doom is a challenge for reinforcement learning because the agent must learn visual features and the optimal behavior according to the partial observation of the environment. Moreover, Doom scenarios have diverse observation and action spaces as well as goals to achieve, so transfer of knowledge across tasks by multi-task learning may cause interference.

The scenarios we used are MyWayHome, Corridor, and DefendTheLine (Figure 8). The first scenario, MyWayHome, requires the agent to reach the goal location in a labyrinth. The second scenario, Corridor, is similar to the first scenario, but there are hostiles and the agent can shoot a weapon to eliminate them. The last scenario, DefendTheLine, is restricted in a single square-shaped room, and hostiles are in the scenario. Valid actions of each scenario are 'Move Forward', 'Turn Right' and 'Turn Left' (MyWayHome); 'Attack', 'Move Right', 'Move Left', 'Move Forward', 'Turn Right' and 'Turn Left' (Corridor); and 'Attack', 'Turn Right' and 'Turn Left' (DefendTheLine).

Evaluated Approaches

We compared our proposal to the following methods:

Single-task learning (STL) trains separate neural network model for each scenario, without transfer from other tasks.

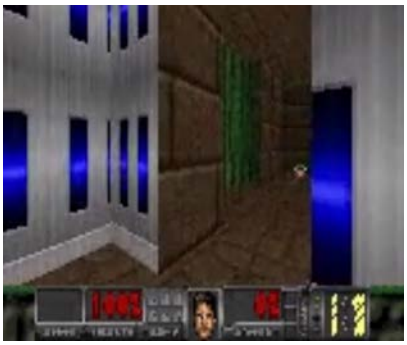
Hard parameter sharing (HPS) shares feature spaces of tasks rather than behavior spaces, in contrast to Distral. HPS shows its strength when tasks have common visual features but no common action behavior.

Distral transfers action policies to regulate them to remain similar to each other. Since the scenarios described above have different action spaces, we modify these scenarios to have same number of possible actions to evaluate the Distral model. The newly introduced actions for the uniform action space are processed as 'No Operation'.

Results

Figure 9 shows the performance of STL, HPS, and our proposed method (Distral and Sobolev on the last hidden layer) on three Doom environments with their own action spaces. Our method boosts the speed of training in the MyWayHome scenario while achieving similar performance to STL method in other two scenarios. HPS shows similar positive effects of transfer in the MyWayHome and DefendTheLine scenarios, however HPS learns nothing in the Corridor

scenario---the reason for this is because these scenarios have similar goal of actions, such as eliminating hostiles, but differ in visual features. In contrast, our method is able to maintain the performance of STL when no positive transfer from other tasks exists and improve otherwise.

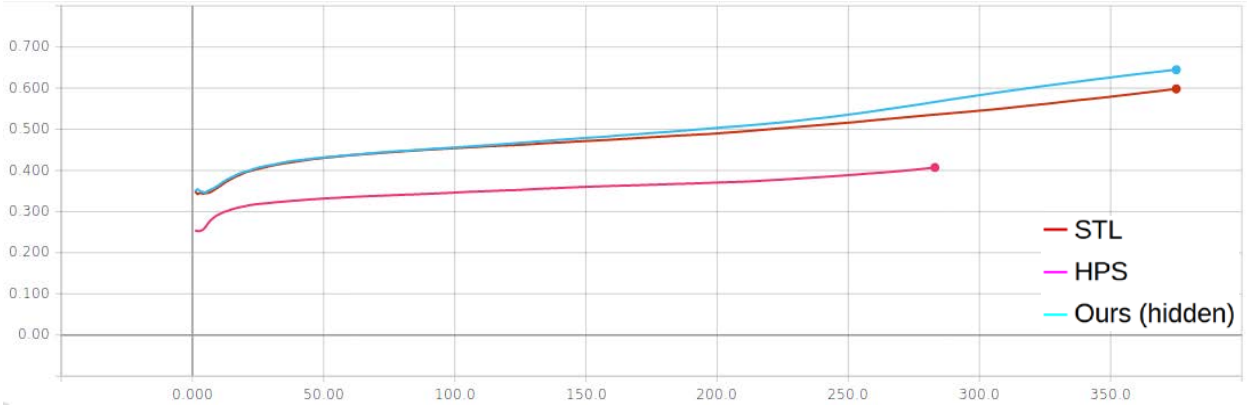


(a) Doom MyWayHome scenario

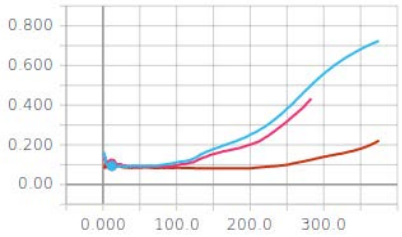
(b) Doom Corridor scenario

(c) Doom DefendTheLine scenario

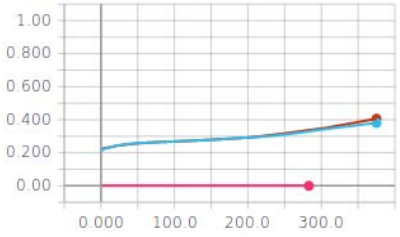
Figure 8 - Doom Environments Used to Compare Performance of Reinforcement Learning Methods



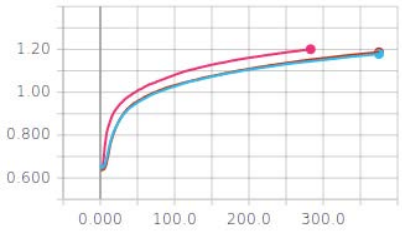
(a) Averaged reward over tasks.



(b) Reward of MyWayHome.



(c) Reward of Corridor.



(d) Reward of DefendTheLine.

Figure 9 - Performance on Doom Environments When Each Task Has Its Own Action Space

Figure 10 compares the performance of the multi-task models according to two possible choices of task action spaces. Interestingly, Distral with an extended action space learns policies outperforming all others, because newly introduced actions provides space for task-specific policies to match the central policy while avoiding any harm to the valid actions of each task. On

the other hand, our proposed method performs better when no redundant actions exist in the action space. This is because regularization on the last hidden layer already gives enough flexibility to the task-specific policies, so the addition of redundant actions makes the model less sample efficient, in contrast to using Distral only.

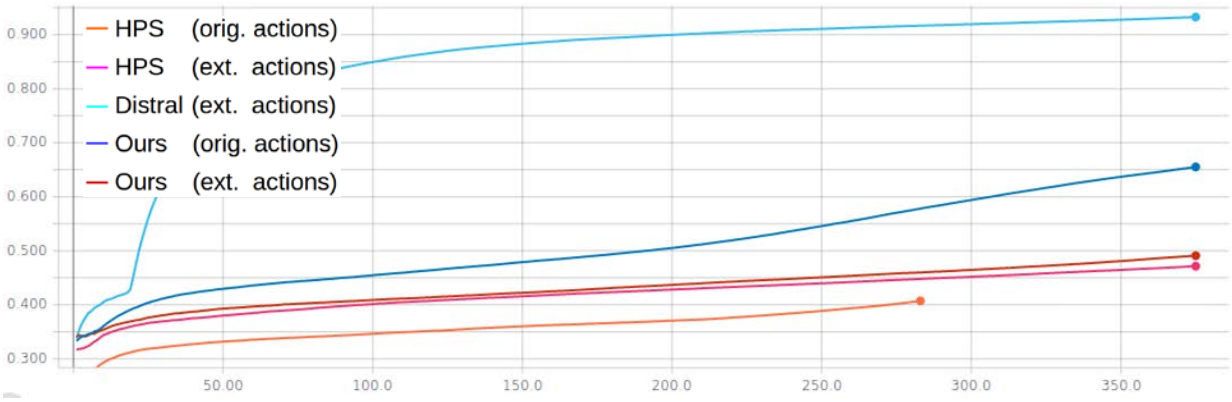


Figure 10 - Multi-task Methods on Doom Environments with Different Action Spaces

4.3 Results on ISR Scenarios in the ATE[3] Simulator

We evaluated the performance of our approach on the ATE[3] simulator developed by Embry-Riddle Aeronautical University, controlling a team of unmanned aerial vehicles in simulated intelligence, surveillance, and reconnaissance (ISR) scenarios. There are six ISR scenarios in total which have three different maps and two sets of hostile agents per map. The smallest map (scenario 1), 6km by 6km, has 15 ISR drones and a homebase at the center of the map. The second smallest map (scenario 2), 10km by 10km, has 40 ISR drones and a homebase at the location 3km away from the center in both longitudinal and latitudinal direction. The largest map (scenario 3), 16km by 16km, has 40 ISR drones and a homebase at the location 4km away from the center similar to the map of scenario 2. One of two hostile groups of each map has fewer mobile anti-aircraft weapons (AA) than the other group, and it is named as 'permissive' case while the other group is named as 'A2AD' case.

Statistics of the scenarios and the performance of our agent in these scenarios are summarized in Tables 1, 2, and 3. First of all, our agent was able to make more drones return to the homebase safely in permissive cases rather than in A2AD cases because of the small number of mobile AAs. However, in scenario 3, it lost more drones in permissive cases. We observed that the majority of the lost ISR drones in scenario 3 were due to low fuel rather than hostiles, so this loss happened because of the fixed criterion (30%) to return to the homebase regardless of the map size.

We found one limitation of the learned neural network controller that it prefers moving drones toward AA as a byproduct of how it was trained, anticipating that a region of interest lies beyond the AA. In the scenario for training the neural network, the goal region lies behind AAs from the view of drones, so we conjectured that this behavior of the neural network in ISR scenarios originated from the characteristics of training scenario. Using different training scenarios would make the neural network learn better policies to avoid these hostile.

Table 1 - Performance of our Hierarchical Agent on ATE3 ISR Scenario 1

Types	Total Number in Map	A2AD Scenario	Permissive Scenario
Survived ISR Drones	15	0	8
Observed Intels	20	11	19
Observed Jammers	5	4	5

Table 2 - Performance of our Hierarchical Agent on ATE3 ISR Scenario 2

Types	Total Number in Map	A2AD Scenario	Permissive Scenario
Survived ISR Drones	40	0	3
Observed Intels A	20	8	8
Observed Intels B	20	7	10
Observed Jammers	10	4	6

Table 3 - Performance of our Hierarchical Agent on ATE3 ISR Scenario 3

Types	Total Number in Map	A2AD Scenario	Permissive Scenario
Survived ISR Drones	40	21	10
Observed Intels A	20	6	13
Observed Intels C	20	2	5
Observed Intels D	10	0	2
Observed Jammers	10	3	3

5.0 CONCLUSIONS

The development of deep neural networks capable of lifelong learning is still in its infancy, and since the start of this project, we have seen increasing interest in the broader literature of this problem [Chen and Liu, 2018]. The methods developed under this project, especially the DF-CNN, represent a huge step toward the development of lifelong deep learning. Indeed, the DF-CNN is the first deep neural network architecture capable of supporting scalable learning to numerous tasks over its lifetime. Its behavior during experimentation is characteristic of lifelong learning, adapting knowledge over time, recovering learning performance on the earliest tasks, and avoiding catastrophic forgetting. Future work is needed to develop further the idea of lifelong deep learning, with our results being early indications of its feasibility and promise.

Our experiments also reveal the limitations of deep reinforcement learning, requiring numerous interactions to learn scenarios. Training first in simulation and then deploying to the real world would reduce the burden of live training, but methods that exploit such simulation-to-real transfer need further development. In the longer term, to solve problems at scale with rapid live learning, we need improved lifelong learning systems that move beyond current reinforcement learning. Learning hierarchical composable knowledge via lifelong learning is potentially one solution. Such composable knowledge could correspond to skills, which could then be dynamically integrated together live via planning algorithms to solve larger problems. Both these skills and the ability to plan with them could be refined over time, with knowledge reused across multiple scenarios. Lifelong skill learning and planning could potentially solve many of the application issues experienced in this project, and would represent a feasible next step toward developing lifelong learning systems that are capable of operating live.

6.0 REFERENCES

Bell, P., Renals, S., "Regularization of context-dependent deep neural networks with context-independent multi-task training," *IEEE International Conference on Acoustics, Speech and Signal Processing*, Apr 2015, pp. 4290-4294.

Bou Ammar, H., Eaton, E., Ruvolo, P., and Taylor, M.E. "Online multi-task learning for policy gradient methods." In *Proceedings of the 31st International Conference on Machine Learning* (ICML-14), June 2014.

Brabandere, B. D., Jia, X., Tuytelaars, T., Gool, L. V., "Dynamic Filter Networks," *In Advances in Neural Information Processing Systems*, 2016, pp. 667-675.

Caruana, R., "Multitask learning," *Machine learning*, **28(1)**, Jul 1997, pp. 41-75.

Yim, J., Jung, H., Yoo, B., Choi, C., Park, D., Kim, J., "Rotating your face using multi-task deep neural network," *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun 2015, pp. 676-684.

Chen, Z., Liu, B., "Lifelong machine learning, second edition," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 2018.

Czarnecki, W. M., Osindero, S., Jaderberg, M., Swirszcz, G., Pascanu, R., "Sobolev training for neural networks," *In Advances in Neural Information Processing Systems*, 2017, pp. 4278-4287.

Gao, Y., She, Q., Ma, J., Zhao, M., Liu, W., Yuille, A. L., "NDDR-CNN: layer-wise feature fusing in multi-task cnn by neural discriminative dimensionality reduction," *arXiv preprint arXiv:1801.08297*, 2018.

Ha, D., Dai, A. M., Le, Q. V., "HyperNetworks," *In Proceedings of the International Conference on Learning Representations*, 2017.

Huang, J., Li, J., Yu, D., Deng, L., Gong, Y., "Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers," *IEEE International Conference on Acoustics, Speech and Signal Processing*, May 2013, pp. 7304-7308.

Krizhevsky, A., Hinton, G., "Learning multiple layers of features from tiny images," *Technical report, University of Toronto*, **1(4)**, Apr 2009.

Liu, P., Qiu, X., Huang, X., "Adversarial multi-task learning for text classification," *In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, **1**, Jul 2017, pp. 1-10.

Liu, H., Lu, J., Feng, J., Zhou, J., "Learning deep sharable and structural detectors for face alignment," *IEEE Transactions on Image Processing*, **26(4)**, Apr 2017, pp. 1666-1678.

Long, M., Cao, Z., Wang, J., Yu, P. S., "Learning multiple tasks with multilinear relationship networks," *In Advances in Neural Information Processing Systems*, 2017, pp. 1594-1603.

Lu, Y., Kumar, A., Zhai, S., Cheng, Y., Javidi, T., Feris, R., "Fully-adaptive feature sharing in multi-task networks with applications in person attribute classification," *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5334-5343.

Misra, I., Shrivastava, A., Gupta, A., Hebert, M., "Cross-stitch networks for multi-task learning," *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3994-4003.

Mnih, V., et al. "Human-level control through deep reinforcement learning." *Nature* 518(7540):529–533, February 2015.

Parisotto, E., Ba, J., Salakhutdinov, R., "Actor-mimic deep multitask and transfer reinforcement learning," *In Proceedings of the International Conference on Learning Representations*, 2016.

Pathak, D., Agrawal, P., Efros, A. A., Darrell, T., "Curiosity-driven exploration by self-supervised prediction," *In Proceedings of the International Conference on Machine Learning*, 2017.

Pinto, L., Gupta, A., "Learning to push by grasping: using multiple tasks for effective learning," *In Proceedings of the IEEE International Conference on Robotics and Automation*, May 2017, pp.2161-2168.

Ranjan, R., Patel, V., Chellappa, R., "HyperFace: a deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **41(1)**, Dec 2017, pp. 121-135.

Rusu, A. A., Colmenarejo, S. G., Gulcehre, C., Desjardins, G., Kirkpatrick, J., Pascanu, R., Mnih, V., Kavukcuoglu, K., Hadsell, R., "Policy distillation," arXiv preprint arXiv:1511.06295, 2015.

Rusu, A. A., Rabinowitz, N. C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., Hadsell, R., "Progressive neural networks," *arXiv preprint arXiv:1606.04671*, 2016.

Ruvolo, P., Eaton, E., "ELLA: an efficient lifelong learning algorithm," *In Proceedings of the International Conference on Machine Learning*, 2013.

Silver, D., et al. "Mastering the game of go with deep neural networks and tree search." *Nature* 529:484–503, 2016.

Teh, Y. W., Bapst, V., Czarnecki, W. M., Quan, J., Kirkpatrick, J., Hadsell, R., Heess, N., Pascanu, R., "Distral: robust multitask reinforcement learning," *In Advances in Neural Information Processing Systems*, 2017, pp. 4496-4506.

Venkateswara, H., Eusebio, J., Chakraborty, S., Panchanathan, S., "Deep hashing network for unsupervised domain adaptation," *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5018-5027.

Vezhnevets, A. S., Osindero, S., Schaul, T., Heess, N., Jaderberg, M., Silver, D., Kavukcuoglu, K., "Feudal networks for hierarchical reinforcement learning," *In Proceedings of the International Conference on Machine Learning*, **70**, Aug 2017, pp. 3540-3549.

Yang, Y., Hospedales, T., "Deep multi-task representation learning: a tensor factorisation approach," *In Proceedings of the International Conference on Learning Representations*, 2017.

Yoon, J., Yang, E., Hwang, S., "Lifelong learning with dynamically expandable networks," *In Proceedings of the International Conference on Learning Representations*, 2018.

7.0 LIST OF SYMBOLS, ABBREVIATIONS AND ACRONYMS

CNN	Convolutional Neural Network
DF-CNN	Deconvolutional Factorized CNN
Distral	Distillation and Transfer Learning
FSA	Finite State Automaton
HPS	Hard Parameter Sharing
ProgNN	Progressive Neural Network
RL	Reinforcement Learning