Multi-Agent Behavior-Based Policy Transfer

Sabre Didi and Geoff Nitschke

Department of Computer Science, University of Cape Town,
7700 Rondebosch, Cape Town, South Africa
sabredd0@gmail.com,gnitschke@cs.uct.ac.za

Abstract. A key objective of transfer learning is to improve and speed-up learning on a target task after training on a different, but related, source task. This study presents a neuro-evolution method that transfers evolved policies within multi-agent tasks of varying degrees of complexity. The method incorporates behavioral diversity (novelty) search as a means to boost the task performance of transferred policies (multi-agent behaviors). Results indicate that transferred evolved multi-agent behaviors are significantly improved in more complex tasks when adapted using behavioral diversity. Comparatively, behaviors that do not use behavioral diversity to further adapt transferred behaviors, perform relatively poorly in terms of adaptation times and quality of solutions in target tasks. Also, in support of previous work, both policy transfer methods (with and without behavioral diversity adaptation), out-perform behaviors evolved in target tasks without transfer learning.

Keywords: Multi-agent Learning, Evolutionary Algorithms, Transfer Learning, Behavioural diversity adaptation

1 Introduction

Transfer learning[1] is a technique that attempts to improve learning a task by leveraging knowledge from learning a related but simpler task [1]. Specifically, transfer learning is the process of reusing learned information across tasks, where information is shared between a source and target task. Transferring knowledge that is learned on a source task accelerates learning and increases solution quality in target tasks by exploiting relevant prior knowledge.

Transfer learning has been widely studied in the context of *Reinforcement Learning* (RL) [2], for various single-agent tasks including pole-balancing [3], game-playing [4], robot navigation as well as multi-agent tasks including predator-prey [5]. For such single and multi-agent tasks, policy (behavior) transfer is typically done within the same task domain for varying task complexity [2]. To facilitate the learning of generalized problem solving behavior various agent (controller) representations have been used including *Decision Trees* [4], *Artificial Neural Networks* (ANNs), *Cerebellar Model Arithmetic Computer*, and *Radial Basis Functions* [6]. Such representations are typically selected as they are amenable to decomposition for transfer of partial policies between source

[1] *Transfer learning* and *policy transfer* are used interchangeably in this paper.

and target tasks as well as further adaptation in target tasks [4]. In multi-agent transfer learning, policies learned in source tasks are often shared between agents and used as a starting point for learning new policies in target tasks [5]. A popular multi-agent test-bed is *RoboCup Keep-Away Soccer* [7], which has received significant attention in multi-agent transfer learning research [6].

Recently, in addition to RL, there has been an increasing amount of work on transfer learning using evolutionary algorithms to adapt policies with various representations. For example, Doncieux [8] used *neuro-evolution* [9] to search for effective ANN controllers in a simulated robot ball collecting task, and investigated methods for extracting behavioral features shared between versions of the task. These extracted features were then used as stepping stones to shape rewards in the evolution of controllers transferred to more complex versions of the ball collecting task.

In related research, Moshaiov et al. [10] used *Multi-Objective Evolutionary Algorithms* [11] to devise a *Family Bootstrapping* method that evolved groups of complementary ANN controllers to robot navigation tasks. These controllers were then used as an evolutionary starting point for controller evolution in robot navigation tasks with different objectives. Taylor et al. [12] used the NEAT neuro-evolution method [13] to further evolve a population of ANN controllers already evolved for a source keep-away soccer task. The authors demonstrated that biasing and further evolving a fittest population of controllers for more complex versions of keep-away significantly decreased evolution time.

Verbancsics et al. [14] used an indirect encoding neuro-evolution method (HyperNEAT [15]) to facilitate evolved solutions encoding the geometry of the keep-away soccer task. HyperNEAT facilitated the transfer of evolved multi-agent behaviors between source and target tasks with varying numbers of agents and soccer field sizes, without the need for any further adaptation. The authors also used HyperNEAT to demonstrate successful transfer of multi-agent behaviors between the *Knight's Joust* (a multi-agent predator-prey task variant) [6] and keep-away soccer tasks. The efficacy of this approach was further supported by improved task performance on target tasks after further neuro-evolution and evolved behaviors that were comparable to RL derived policies [16], [17]. In support of this approach, related work [18], [14] has also highlighted the effectiveness of indirectly coded representations for facilitating transfer learning between multi-agent task variants as well as between different multi-agent tasks.

A key challenge in transfer learning is to ensure that a policy, learned in a source task can be meaningfully transferred to a policy in a target task, with a typically more complex representation [6]. Hence a mapping function is required in order that learned policies are transferable between tasks with different numbers of state and action variables. For example, tasks of increasing complexity or different but related tasks such as keep-away soccer [7] and Knight's Joust [6]. To address this, Taylor et al. [6], devised a *inter-task mappings for policy search* method to transfer a population of control policies (ANN controllers) between keep-away soccer, knight's joust and *Server Job Scheduling* tasks [17]. This method was successfully applied with full (hand-coded) inter-task mapping

functions, where inter-task mapping functions were only partially available or where inter-task mapping functions had to be learned prior to policy transfer.

This study combines and extends previous work on *inter-task mappings for policy search* [6] and facilitating transfer learning with HyperNEAT [14]. Specifically, we investigate the adaptation of multi-agent behaviors in the keep-away soccer task domain with the *Novelty Search* [19] behavioral diversity mechanism. Whilst many studies support the efficacy of objective-based (fitness function) search approaches in transfer learning [17], [6], [18], [14], the impact of behavioral diversity maintenance on transfer learning remains unexplored. This study also investigates the benefits of behavioral diversity and objective based search with policy transfer using direct (NEAT) and indirect encoding (HyperNEAT) methods to evolve behaviors in target keep-away tasks.

First, we hypothesize that if behavioral diversity maintenance is used in multi-agent behavior evolution, this will yield a higher *task performance* than objective-based evolution in all tasks tested. Second, we hypothesize that NEAT and HyperNEAT are appropriate policy (multi-agent behavior) search methods for enabling policy transfer where transferred behaviors yield a *higher task performance* and *efficiency* compared to those without policy transfer. Efficiency refers to the average number of generations until the *average maximum fitness* (for the given method) was attained.

These hypotheses were devised given related research results [20], [21], [22], and were tested by a comparison of keep-away behaviors evolved with NEAT and HyperNEAT (using either objective-based search or behavioral diversity maintenance) in keep-away target tasks with and without policy transfer.

2 Methods

2.1 NEAT: Neuro-Evolution of Augmenting Topologies

This research uses *Neuro-Evolution of Augmenting Topologies* (NEAT) [13] as the direct encoding policy search method. NEAT evolves both connection weights and ANN topologies, and applies three key techniques to maintain a balance between performance and diversity of solutions. First, it assigns a unique historical marking to every new gene so as crossover can only be performed between pairs of matching genes. Second, NEAT speciates the population so as ANNs (genotypes) compete primarily within their own niches (identified by historical markings) instead of competing with the population at large. Third, NEAT begins evolution with a population of simple ANNs with no hidden nodes but gradually adds new topological structure (nodes and connections) using two special mutation operators called *add hidden node* and *add link*.

NEAT was selected as this study's direct encoding method as it has been successfully used for a broad range of multi-agent control tasks [12], [2], [23], [24], [10]. However, there has been relatively little research as to efficacy of NEAT as a policy search method for multi-agent transfer learning [12].

2.2 HyperNEAT: Hypercube-based NEAT

Hypercube-based NEAT (HyperNEAT) [15] is an indirect (generative) encoding neuro-evolution method that extends NEAT and uses two networks, a *Composite*

Pattern Producing Network (CPPN) [25] and a *substrate* (ANN). The CPPN is the generative encoding mechanism that indirectly maps evolved genotypes to ANNs and encodes pattern regularities, symmetries and smoothness of the geometry of a given task in the form of the substrate. This mapping functions via having coordinates of each pair of nodes connected in the substrate fed to the CPPN as inputs. The CPPN outputs a value assigned as the synaptic weight of that connection and a value indicating whether that connection can be expressed or not. HyperNEAT uses the evolutionary process of NEAT to evolve the CPPN and determine ANN fitness values. The main benefit of HyperNEAT is scalability as it exploits task geometry and thus effectively represents complex solutions with minimal genotype structure [15]. This makes HyperNEAT an appropriate choice for evolving complex multi-agent solutions [14], [26].

HyperNEAT was selected as this study's indirect encoding neuro-evolution method since previous research indicated that transferring the *connectivity patterns* [27] of evolved behaviors is an effective way for facilitating transfer learning in multi-agent tasks [18], [14]. HyperNEAT's capability to evolve controllers that account for task geometry also makes HyperNEAT appropriate for deriving controllers that elicit behaviors robust to variations in state and action spaces [28] as well as noisy, partially observable environments of multi-agent tasks. Also, it has been demonstrated that HyperNEAT evolved multi-agent policies can be effectively transferred to increasingly complex versions of keep-away soccer [7] without further adaptation [14] and that transferred behaviors often yield comparable task performance to specially designed learning algorithms [16].

2.3 Behavioral Diversity

Encouraging behavioral diversity is a well studied concept in neuro-evolution and has been used to discover novel solutions, increase solution performance in a wide range of tasks as well as out-perform controller evolution approaches that encourage genotypic diversity [20], [29], [30], [31],

One such approach is *Novelty search* (NS) [19], that is not driven by a fitness (objective) function but rather rewards evolved phenotypes (behaviors) based on their novelty. Thus, a genotype is more likely to be selected for reproduction if its encoded behavior is sufficiently different from all other behaviors produced thus far in an evolutionary run. Recent results indicate that controllers evolved with a NS metric attained some degree of generality. For example, in a maze solving task, controllers evolved to solve one maze were successfully transferred to solve different mazes [32]. Also, NS has been demonstrated as yielding solutions that out-perform objective based search in various tasks [20], [22] including complex multi-agent tasks with large numbers of agents [21]. Hence, NS was selected as the behavioral diversity mechanism to be applied to our selected policy search methods (NEAT and HyperNEAT).

In this study, the function of NS is to consistently generate novel team (keep-away) behaviors. Hence, we define team behavior in terms of properties that potentially influence team behavior but are not directly used for task performance evaluation. For the keep-away task, the behavioral properties we use are

the *average number of passes*, *average dispersion of team members*, and *average distance of the ball to the center of the field*.

In line with previous research on hybrid NS and fitness metrics supporting performance gains in various tasks [33], including multi-agent tasks [34], we use a behavioral diversity metric that linearly combines NS with objective-based search (NEAT and HyperNEAT), in order to improve keep-away policy search.

Several hybrid metrics have been proposed including fitness sharing and linear combination [21], restarting converged evolutionary runs using NS [33], a minimal criteria NS (for genotype survival and reproduction) [35], and a progressive minimal criteria (incrementing reproduction requirements throughout evolution) [34]. Here we use a linear combination [21] (Equation 1):

$$score_i = \rho \cdot \overline{fit_i} + (1 - \rho) \cdot \overline{nov_i} \tag{1}$$

Where, $\overline{fit_i}$ and $\overline{nov_i}$ are normalized fitness and novelty of i^{th} genotype respectively. Then $\rho \in [0, 1]$ is a parameter selected by the experimenter ($\rho = 0.4$, in this study) to control the relative contribution of each metric to the selection pressure. To measure novelty we use normalized task specific behavioral vectors: *Average number of passes*, *Mean team mates dispersion*, and *Average distance of ball to the center of the field*.

This team level behavioral characterization has been used previously [21] and out-performs individual behavioral characterizations and fitness based search. Behavioral distance is computed as a Euclidean distance (Equation 2):

$$\delta(x, y) = \|x_i - y_i\| \tag{2}$$

Where, x_i and y_i are normalized behavioral characterization vectors of genotype x and y. The novelty is then quantified by equation 3:

$$nov_x = \frac{1}{3k} \sum_{i=1}^{k} \sum_{j=1}^{3} \delta(x_j, y_{ij}) \tag{3}$$

Where, x_j is the j^{th} behavioral property of genotype x, y_{ij} is the j^{th} behavioral property of the i^{th} nearest neighbour of genotype x and δ is the behavioral distance between two genotypes x and y computed in equation 2 which is based on the behavioral characterization vector. The nov_x is then derived from the mean of behavioral distance of an individual with k nearest neighbors. The parameter k is specified by the experimenter to represent the number of nearest neighbors, where $k = 15$ has been widely used in NS experiments [22]. A few researchers have used $k = 20$ [36], [22] and k in the range of $[3, 10]$ though it is unclear if such k values were derived experimentally. Gomes et al. [22] discovered that the choice of k value heavily depended on the type of novelty archive used and that $k = 15$ yielded relatively good performance across all tested archive types. Hence in this study we use $k = 15$.

2.4 Policy Transfer Method

For both NEAT and HyperNEAT, and their non-objective (novelty) and objective based search variants, we tested three policy transfer approaches. First, the entire evolved population was transferred from the source task (at the final generation of neuro-evolution) and set as the initial population for neuro-evolution in the target task. Second, target population was seeded with the fittest genotype in the source task and used as a bias for initialising the remainder of the target population. Third, the fittest 50% of the population evolved for the source task was selected to seed and bias initialization of the rest of the starting population in the target task. The first approach was found to be the most effective for all methods and tasks tested in this case study and was thus used in company with the selected mapping function for policy transfer (Algorithm 1). Algorithm 1 is a transfer mapping function that is an extension of that proposed by Taylor et al. [6] and used is used in this study's keep-away policy transfer experiments.

Algorithm 1: Transfer Mapping Function

Generate a network with same number of inputs and outputs as in the Π_{source}
Add the same number of hidden nodes to Π_{target} as in Π_{source}
Repeat
For each pair of nodes (n_i,n_j) in Π_{target} **do**
  If $\exists$ link $L_{i,j} \in \Pi_{source}$ **then**
    add link $L_{i,j}$ to Π_{target} with ${w^t}_{i,j} = {w^s}_{i,j}$ in Π_{source}
  Else
    If $\nexists$ nodes(n_i,n_j) $\in \Pi_{source}$
      add link $L_{i,j}$ to Π_{target} with ${w^t}_{i,j}$ = random weights
Until all pairs of nodes are visited

3 Experiments

Experiments test this study's research objectives (Section 1). First, to test the impact of using a non-objective (behavioral diversity) versus objective (fitness) based search approach for two given policy search methods (NEAT and HyperNEAT). Second, to test the efficacy of NEAT and HyperNEAT as appropriate methods for yielding task performance and efficiency boosts after policy transfer.

Experiments are run in a source keep-away task (using NEAT or HyperNEAT to evolve multi-agent keep-away behavior), where populations evolved after 20 generations, are transferred to a target task, and evolved for a further 50 generations (Table 2). Results are compared to those where no policy transfer takes place, that is where NEAT and HyperNEAT are used to evolve keep-away behaviors from *scratch* in the target tasks. For both NEAT and HyperNEAT experiments, each genotype (agent team) is evaluated over 30 task trials per generation, where each task trial tests different (random) agent positions. The ball always starts in the possession of a (randomly selected) keeper. Average fitness (task performance) per genotype is computed over these 30 task trials. Table 2 specifies the neuro-evolution and simulation parameters for these experiments.

Sensory Inputs	Description
dist(K_b, C), dist(K_{t1}, C), dist(K_{t2}, C)	Distance of each keeper to field center
dist$(T1, C)$, dist$(T2, C)$	Distance of each taker to field center
dist(K_b, K_{t1}), dist(K_b, K_{t2})	Distance of each taker to keeper 1
dist$(K_b, T1)$, dist$(K_b, T2)$	Distance of each taker to keeper 1
$min_{j\in1,2}dist(K_{t1}, T_j)$, $min_{j\in1,2}dist(K_{t2}, T_j)$	Distance of closest taker to keeper 1
$min_{j\in1,2}angle(K_{t1}, T_j)$, $min_{j\in1,2}angle(K_{t1}, T_j)$	Angle of closest keeper, taker, keeper 1
Motor Outputs	
Hold	Do not pass ball
Pass to K_{t1}, Pass to K_{t2}	Pass to keeper 2, keeper 3

Table 1. Sensory inputs (13 input nodes) and motor outputs (three outputs) for a team's ANN controller in the *3vs2 keep-away task*. Keeper 1 is the agent with the ball.

The efficacy of policy transfer was evaluated in terms of time (genotype evaluations) taken to attain a policy transfer threshold, with and without policy transfer. The threshold was the *average maximum fitness* attained after applying NEAT and HyperNEAT to evolve behaviors *from scratch* in each target task. Policy transfer occurs between source and incrementally complex target tasks. That is, first we evolve keep-away behavior for three keepers versus two takers (denoted as *3vs2*) in a 20 x 20 virtual field[2] (Table 2). Evolved behaviors (policies) are then transferred (and neuro-evolution continued) in one of three keep-away target tasks, four keepers versus three takers (*4vs3*), five keepers versus three takers (*5vs3*) or six keepers versus four takers (*6vs4*).

3.1 NEAT Experiments

Table 1 describes the 13 sensory input nodes in a team's ANN controller for the *3vs2 keep-away* task. The output nodes represent an agent's decision to *hold* the ball, *pass to keeper 2* or *pass to keeper 3*, where *keeper 1* has the ball. At any task trial iteration, the output with the highest activation is the action selected.

NEAT is direct encoding method, so the genotype representation (encoding sensory-motor elements of a keep-away team's controller) needs to change as task complexity and the number of agents changes. For example, as task complexity increases, from *3vs2* to *4vs3 keep-away*, an ANN topology with 19 input nodes and 4 output nodes is required. The additional output node represents the decision of *keeper 1* to *pass to keeper 4*. The extra six input nodes represent: 1) distance of *keeper 4* from the field's center, 2) distance of *taker 3* from the field's center, 3) distance of *keeper 1* from *taker 3*, 4) distance between *keeper 4* and the closest taker, 5) angle formed between keeper 1 and the closest keeper and taker, and 6) distance of *keeper 1* to *keeper 4*. Similarly, for the *5vs3 task* an ANN with 27 inputs and six outputs is needed.

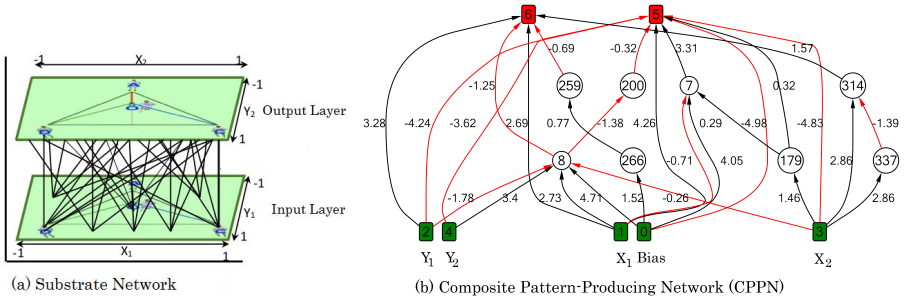
However, for all keep-away tasks tested (*3vs2*, *4vs3*, and *5vs4*) the ANN sensory-motor layer topology was kept static (13 sensory inputs and three motor

[2] All experiments were run in *RoboCup Keep-Away version 6* [6]. Source code and executables can be found at: http://people.cs.uct.ac.za/~gnitschke/EvoStar2016/

NE / NS Parameters	Setting
Population Size	150
Generations (Source task)	20
Generations (Target task)	50
Maximum number of species	10
Maximum species population	30
Weight mutation	± 0.01
NEAT Weight value range	[-5.0, 5.0]
HyperNEAT Weight value range	[-5.0, 5.0]
Mutation rate	0.05
Survival threshold	0.2
NS nearest neighbor k	15
Maximum archive size	1000
Compatibility threshold	3
Behavioral threshold	0.03

HyperNEAT CPPN	Functions
Identity	x
Gaussian	$e^{-2.5x^2}$
Bipolar Sigmoid	$\frac{2}{1+e^{-4.9x}} - 1$
Absolute value	$\|x\|$
Sine	$\sin(x)$
Simulation Parameters	**Setting**
Number of Runs	20
Iterations per task trial	4500
Trials per generation	30
Agent positions	Random
Environment size	20x20 grid
Agent speed (per iteration)	1 grid cell
Ball speed (per iteration)	2 grid cells

Table 2. Left: *Neuro-Evolution* (NE), *Novelty Search* (NS) parameters (final three rows). **Right:** CPPN (HyperNEAT) activation Functions and simulation parameters.



(a) Substrate Network

(b) Composite Pattern-Producing Network (CPPN)

Fig. 1. Left: Substrate encoding the virtual field (20 x 20 grid of inputs and outputs). Connection values ([-1.0, 1.0]) between these input-output nodes represent positions of agents relative to the keeper with the ball. **Right:** Connections from pairs of nodes in the substrate are sampled and the coordinates passed as inputs to the CPPN, which then outputs the synaptic weight of each sampled connection.

outputs) in order to facilitate transfer across tasks of increasing complexity. Thus, as the number of agents increased with task complexity, a heuristic selected which agents in the environment would be processed by the ANN's 13 sensory input nodes. At each sensory-motor cycle (task trial iteration), the heuristic selected the closest two keeper and taker agents to be processed by the ANN, but had the potential to process any agent as sensory input. In keep-away task simulation this was tantamount to noise preventing the keeper with the ball from processing agents too far away and thus accounting for them in action selection.

Experiment	4vs3 Keep-Away	5vs3 Keep-Away	6vs4 Keep-Away
No Policy Transfer			
NEAT	0.438 (0.037)	0.473 (0.052)	0.419 (0.057)
HyperNEAT	0.587 (0.059)	0.765 (0.050)	0.533 (0.044)
Fitness-Based Policy Transfer			
NEAT	0.482 (0.059)	0.580 (0.069)	0.464 (0.033)
HyperNEAT	0.729 (0.089)	0.873 (0.089)	0.632 (0.038)
Fitness + NS Policy Transfer			
NEAT	0.545 (0.047)	0.638 (0.0048)	0.520 (0.036)
HyperNEAT	0.752 (0.054)	0.943 (0.029)	0.697 (0.032)

Table 3. Average normalized maximum fitness (over 20 runs) for the three experimental setups. Values are portions of the maximum possible hold time (possession of the ball) for the team of keepers. Standard deviations are shown in parentheses.

3.2 HyperNEAT Experiments

HyperNEAT uses indirect encoding and can thus represent changes in task complexity without changing genotype representation [14]. In this experiment *Bird's Eye View* (BEV) representation [14] is used to encode keep-away's physical state (layout of the field and locality of agents) and actions onto a substrate network. The virtual keep-away soccer field is divided into a 20 x 20 grid world, where each agent can occupy one grid cell per task trial iteration. The input and output layers of the substrate network are two dimensional, with coordinates in the x, y plane in the range of $[-1.0, 1.0]$. Each grid cell in the virtual space is represented by a node in the substrate network layer, so the 20 x 20 grid world is represented by 400 nodes in the substrate network. Hence, the layout of nodes in the substrate network (network geometry) directly maps to the tasks geometry and this enables HyperNEAT to exploit the task's geometric regularities and relationships. The position of each agent is marked on the substrate input layer, where each position of the keeper is marked by a value 1.0, and takers by -1.0. Physical paths between agents are drawn. Each direct path from a keeper with a ball to another keeper is marked by a value 0.3 and to a taker by a value -0.3. The region to pass the ball to is highlighted on the substrate output by activating the node with the highest output.

The CPPN queries each connection between input and output layers of the two dimensional substrate network taking coordinates $(x1, y1)$ and $(x2, y2)$ as input. The CPPN output represents the weight of that connection and the connection expression value. The connection weights are then produced as a function of their endpoints. The functions used are listed in Table 2 (right).

4 Results and Discussion

Policy transfer was applied between the source *3vs2* keep-away task and incrementally complex *4vs3*, *5vs3* and *6vs4* keep-away tasks (section 3). Keep-away behaviors were evolved for 20 generations with NEAT or HyperNEAT (using

either the novelty-objective hybrid or objective-based search) in the source task, transferred to the target task and then further evolved for 50 generations. For policy transfer, three population initialization methods in the target task were tested (Section 2.4). However, the transfer of the *entire population* (from generation 20 in the source task) to target tasks best facilitated policy transfer. Hence only results for this population initialization method are presented here.

Table 3 presents the average normalized maximum fitness (attained during each run and averaged over 20 runs) for the three experimental setups. Experiment 1 (*No Policy Transfer*) presents results from evolving keep-away behaviors in each of the target tasks from *scratch* (without policy transfer). Experiment 2 (*Fitness-Based Policy Transfer*) presents results from evolving keep-away behaviors using objective (fitness) based NEAT and HyperNEAT in the source and then in target tasks (after policy transfer). Experiment 3 (*Fitness + NS Policy Transfer*) presents results from evolving keep-away behaviors with NEAT and HyperNEAT using the novelty-objective hybrid based search. In experiments 2 and 3, NEAT or HyperNEAT is applied in the source task for 20 generations and thereafter for 50 generations in the target task.

Results data was found to be non-parametric using the *Kolmogorov-Smirnov* normality test with *Lilliefors* correction [37]. The *Mann-Whitney U* test [38] was then applied in a series of pair-wise comparisons to gauge if there was a statistically significant difference between corresponding result sets of the three experiments (Table 3). Pair-wise comparisons were conducted between average results data for NEAT or HyperNEAT (for a given experiment). The null hypothesis stated that two comparative data sets were not significantly different, and $\alpha = 0.05$ was selected as the significance threshold.

4.1 Policy versus No-Policy Transfer: Performance Comparisons

First, statistical tests indicated that for all policy transfers (from *3vs2* to *4vs3*, *5vs3* and *6vs4*), there was a statistically significant difference (p-value < 0.05) between the *novelty-objective hybrid* and *objective-based NEAT*. That is, NEAT with behavioral diversity maintenance yielded a significantly higher average maximum task performance for all policy transfer cases (Table 3).

Second, statistical tests indicated that for all policy transfers transfer cases, both NEAT and HyperNEAT using behavioral diversity maintenance yielded a higher average maximum task performance compared to objective-based NEAT and HyperNEAT (Table 3).

This result supports this study's first hypothesis (Section 1), that encouraging behavioral diversity facilitates the evolution of higher performance keep-away behaviors in all tasks tested, compared to keep-away behavior adaptation without behavioral diversity maintenance.

Statistical tests also indicated that NEAT (using either the novelty-objective or objective-based search) yielded a higher average task performance (with statistical significance) for all policy transfers, compared to objective-based NEAT without policy transfer. That is, where NEAT was applied to evolve keep-away behavior (*from scratch*) in each of the target tasks (*4vs3*, *5vs3* and *6vs4*). Similarly, statistical tests indicated that HyperNEAT (*with* and *without* behavioral

diversity maintenance), yielded significantly higher task performances in all target tasks compared to HyperNEAT without policy transfer (Table 3).

These results partially support this study's second hypothesis, that NEAT and HyperNEAT are appropriate as policy search methods where policy transfer enables the evolution of significantly higher performance keep-away behaviors in all target tasks tested (compared to keep-away behaviors evolved *from scratch*). These results are further supported by previous work demonstrating that transfer learning enables multi-agent behavior adaptation with significantly higher task performances compared to adaptation without transfer learning [12], [14], [5].

4.2 Policy versus No-Policy Transfer: Efficiency Comparisons

To further support this study's second hypothesis, the efficiency of NEAT and HyperNEAT (*with* and *without* behavioral diversity maintenance) is compared in the target tasks where policy transfer was applied versus where keep-away behaviors were evolved in the target tasks *from scratch*.

Results (*performance threshold*) of applying NEAT and HyperNEAT to adapt keep-away behaviors *from scratch* (without policy transfer) in the target tasks (*4vs3*, *5vs3* and *6vs4*) were used as a benchmark for comparisons with the same methods applied with policy transfer. This threshold was the *average maximum task performance* of NEAT and HyperNEAT in the target tasks, where keep-away behavior was evolved without policy transfer (Table 3).

First, for *objective-based NEAT without policy transfer*, an average maximum task performance of 0.443 (as a portion of maximum task performance) for all three target tasks was attained after approximately 40 generations[3]. After 40 generations negligible task performance increases were observed. Additional experiments that used relatively few runs, but 100 generations of evolution indicated that objective-based NEAT, without policy transfer, gets stuck in a local optima. However, this is not the case when policy transfer is used (for both objective and hybrid objective-novelty based variants of NEAT).

Comparatively, results from *objective-based NEAT with policy transfer* indicated efficiency gains for all target tasks. Objective-based NEAT with policy transfer yielded an average maximum task performance of 0.482, 0.580 and 0.464 for the *4vs3*, *5vs3* and *6vs4* keep-away tasks, respectively. These tasks performances were attained after approximately 48 generations. However, additional experiments using relatively few runs but 100 generations indicated that the task performances yielded by objective-based NEAT with policy transfer continued to increase. Also, all task performances yielded by objective-based NEAT *with policy transfer* were significantly higher (Mann-Whitney test, p-value < 0.05) compared to those yielded by NEAT without policy transfer in the same tasks.

Objective-based HyperNEAT with policy transfer also yielded greater efficiency for all target tasks. That is, objective-based HyperNEAT with policy transfer resulted in an an average maximum task performance of 0.632 for *6vs4* keep-away after 38 generations, *with policy transfer*. This was compared to the

[3] NEAT and HyperNEAT average maximum task performance progression graphs can be found at: http://people.cs.uct.ac.za/~gnitschke/EvoStar2016/

significantly lower 0.533 (Mann-Whitney test, p-value < 0.05) average maximum performance after 49 generations, *without policy transfer* in the same *5vs3* keep-away task.

Task performance in *5vs3* keep-away steadily reached an average maximum of 0.873 after 50 generations, *with policy transfer*. This was compared to the significantly lower 0.765 (Mann-Whitney test, p-value < 0.05) average maximum performance after 48 generations, *without policy transfer* in the same task.

Task performance in *4vs3* keep-away steadily reached an average maximum of 0.729 after 47 generations, *with policy transfer*. This was compared to the significantly lower 0.587 (Mann-Whitney test, p-value < 0.05) average maximum performance after 45 generations, *without policy transfer* in the same task. Also, additional experiments using relatively few runs but 100 generations indicated that the task performances yielded by objective-based HyperNEAT with policy transfer continued to increase.

Second, for *novelty-objective based NEAT with policy transfer*, average maximum task performances of 0.545, 0.638 and 0.520 were attained for tasks *4vs3*, *5vs3* and *6vs4*, after 48, 49 and 48 generations, respectively. This compared to the significantly lower task performances (Mann-Whitney test, p-value < 0.05) of *novelty-objective based NEAT without policy transfer* for the same tasks. That is, 0.443, 0.473, and 0.473 for the *4vs3*, *5vs3* and *6vs4* tasks, attained after 40, 37 and 43 generations respectively.

Also, *novelty-objective based HyperNEAT with policy transfer* yielded average maximum task performances of 0.752, 0.943 and 0.697 for tasks *4vs3*, *5vs3* and *6vs4*, after 45, 49 and 48 generations, respectively. This compared to the significantly lower task performances (Mann-Whitney test, p-value < 0.05) of *objective based HyperNEAT without policy transfer* for the same tasks (Table 3), yielded in a comparable number of generations (45, 48 and 49 generations, for the *4vs3*, *5vs3* and *6vs4* tasks, respectively).

Hence, these results further support this study's second hypothesis, that NEAT and HyperNEAT are appropriate policy search methods, where policy transfer enables a higher efficiency in the target tasks tested. Thus, NEAT and HyperNEAT with policy transfer (using objective-based search or behavioral diversity maintenance) converge to a higher task performance faster compared to the same methods without policy transfer.

4.3 Behavioral Diversity Maintenance and Policy Transfer

The results of this study have important implications for current policy transfer research, specifically multi-agent policy transfer where neuro-evolution is used for policy search (agent behavior adaptation).

First, the results indicated significant *task performance* and *efficiency* (speed-up of evolution) benefits of policy transfer in a multi-agent task (*Keep-away RoboCup Soccer*) where team behavior was evolved with NEAT or HyperNEAT in a source task and then further evolved in more complex target tasks. This was compared to the same methods for evolving keep-away behavior *from scratch* in the target tasks. These results are also supported by related policy transfer research that used neuro-evolution for policy search [12], [14], [8], [10].

Second, results indicated that HyperNEAT with behavioral diversity maintenance yielded the greatest benefits for policy transfer overall. These transferred behaviors leveraged the most benefits of behaviors evolved in the source task such that further evolution in target tasks yielded the highest overall task performances. This was compared to NEAT with behavioral diversity maintenance, objective-based NEAT and HyperNEAT and the same methods without policy transfer. Such benefits of behavioral diversity maintenance coupled with objective-based search is supported by related work [35], [33], [34]. Also, advantages of indirect encoding neuro-evolution methods such as HyperNEAT have been highlighted in a broad range of task domains[15], [14], [26], [28].

However, a key contribution of this study is that this is the first time (to the authors' knowledge), the benefits of behavioral diversity maintenance coupled with neuro-evolution, have been demonstrated in multi-agent policy transfer.

The significantly higher performance of HyperNEAT with behavioral diversity maintenance across all tasks is theorized to be a result of beneficial interactions between a more effective search for high performance behaviors (aided by behavioral diversity maintenance) and HyperNEAT's indirect encoding of agent behaviors. Consider that in the source task the novelty-objective hybrid based search employed by HyperNEAT facilitated an effective exploration versus exploitation trade-off in the search for high-performance keep-away behaviors. This is supported by results from previous work [35], [34], [33], [21] that similarly report the benefits of hybrid novelty-objective based search approaches (including task performance advantages over objective based search approaches).

Also, when effective high-performance behaviors are discovered as solutions to the source task (*3vs2 keep-away*), HyperNEAT's indirect encoding of such behaviors and the spatial geometry of the keep-away task facilitates more effective policy transfer to incrementally complex target tasks. That is, HyperNEAT evolves CPPNs that are able to represent complex ANN controllers with their own symmetries and regularities and exploit the sensory-motor geometry of multi-agent tasks [15], [26]. This controller representation significantly impacted the efficacy of evolved keep-away behavior across all tested target tasks.

Thus, we hypothesize that adapting controllers with HyperNEAT in company with the aid of behavioral diversity maintenance allows first, for the discovery of novel robust and effective multi-agent behaviors (that might not have otherwise been discovered with pure objective-based search). Second, HyperNEAT encodes team behaviors that do not rely upon specific sensory-motor mappings in the agent team controller and thus set task environment configurations (such as specific agent and ball positions and numbers of agents). That is, HyperNEAT evolves *connectivity patterns* [27] that are broadly applicable to tasks of varying complexity (in keep-away, numbers of agents). This is supported by related research that similarly demonstrates the robustness of HyperNEAT evolved controllers in tasks of varying complexity [28].

The performance of HyperNEAT evolved teams was contrasted to the significantly lower task performance of NEAT (with and without behavioral diversity maintenance). In NEAT, team behaviors were directly encoded with an ANN

with fixed sensory-motor layers (13 sensory inputs and three motor outputs), where the number of hidden nodes and connections were evolved. This static sensory-motor layer ANN topology prevented a smooth and effective transfer from the source task to the more complex target tasks. However, behavioral diversity maintenance did boost the task performance and efficiency of NEAT evolved behaviors in all target tasks after policy transfer (Table 3).

The lower performance of both NEAT and HyperNEAT (with and without behavioral diversity maintenance) in the *6vs4* target task (Table 3) remains the subject of current research. Though this is hypothesized to be a result of the increased complexity of four takers on the same sized virtual field, making taker interception of ball passes more likely. Also this increases the required complexity of evolved keep-away behaviors, meaning evolved behaviors must effectively scale to coordinate larger numbers of keepers while accounting for more takers, but with the same spatial constraints on the virtual field as the *4vs3* and *5vs3* tasks.

5 Conclusion

This study investigated methods for improving the current state of the art in multi-agent transfer learning. That is, improving task performance and efficiency (speed of adaptation) of *Keep-away RoboCup Soccer* behaviors evolved in a source task but then further evolved on more complex versions of the same task. Experiments compared two neuro-evolution methods, NEAT and HyperNEAT, applying them to evolve keep-away behaviors. This study's main contribution was elucidating that behavioral diversity maintenance coupled with these methods yielded increased task performance in increasingly complex keep-away tasks.

Results indicated that behavioral diversity maintenance used in company with NEAT and HyperNEAT is an appropriate approach for increasing task performance and efficiency in keep-away tasks of increasing complexity. Using behavioral diversity maintenance enabled NEAT and HyperNEAT to out-perform objective-based NEAT and HyperNEAT with and without policy transfer in all tested target keep-away tasks. Also, results indicated that HyperNEAT using behavioral diversity maintenance yielded the highest overall task performance and efficiency. This was theorized to be a result of HyperNEAT's indirect encoding of keep-away behaviors, facilitating effective transfer of evolved behaviors between tasks of varying complexity.

Future work will further investigate the efficacy of indirect encoding methods for facilitating effective policy transfer between similar but related multi-agent tasks (for example, keep-away to multi-agent predator-prey [26]), thus addressing the larger goal of devising controller design methods capable of producing generalized problem solving behaviors.

References

1. Pan, S., Yang, Q.: A survey on transfer learning. IEEE Transactions on Knowledge and Data Engineering **22(10)** (2010) 1345–1359

2. Torrey, L., Shavlik, J.: Transfer learning. In: Handbook of Research on Machine Learning Applications. IGI Global, Hershey, USA (2009) 17–23
3. Ammar, H., Tuyls, K., Taylor, M., Driessens, K., Weiss, G.: Reinforcement learning transfer via sparse coding. In: Proceedings of the eleventh international conference on autonomous agents and multiagent systems, Valencia, Spain, AAAI (2012) 4–8
4. Ramon, J., Driessens, K., Croonenborghs, T.: Transfer learning in reinforcement learning problems through partial recycling. In: Proceedings of the 18th European Conference on Machine Learning, Warsaw, Poland, Springer (2007) 699–707
5. Boutsioukis, G., Partalas, I., Vlahavas, I.: Transfer learning in multi-agent reinforcement learning domains. In: Recent Advances in Reinforcement Learning. Springer, Berlin, Germany (2012) 249–260
6. Taylor, M., Stone, P., Liu, Y.: Transfer learning via inter-task mappings for temporal difference learning. Journal of Machine Learning **8(1)** (2010) 2125–2167
7. Stone, P., Kuhlmann, G., Taylor, M., Liu, Y.: Keepaway soccer: From machine learning testbed to benchmark. In: Proceedings of RoboCup-2005: Robot Soccer World Cup IX, Berlin, Germany, Springer (2006) 93–105
8. Doncleux, S.: Knowledge extraction from learning traces in continuous domains. In: AAAI 2014 fall Symposium on Knowledge, Skill, and Behavior Transfer in Autonomous Robots, Arlington, USA, AAAI Press (2014) 1–8
9. Floreano, D., Dürr, P., Mattiussi, C.: Neuroevolution: from architectures to learning. Evolutionary Intelligence **1**(1) (2008) 47–62
10. Moshaiov, A., Tal, A.: Family bootstrapping: A genetic transfer learning approach for onsetting the evolution for a set of realated robotic tasks. In: Proceedings of the Congress on Evolutionary Computation, IEEE Press (2014) 2801–2808
11. Deb, K.: Pareto based Multi-objectives optimization using evolutionary algorithms. John Wiley and Sons, New York, USA (2001)
12. Taylor, M., Whiteson, S., Stone, P.: Transfer learning for policy search methods. In: ICML 2006: Proceedings of the Twenty-Third International Conference on Machine Learning Transfer Learning Workshop, Pittsburgh, USA, ACM (2006) 1–4
13. Stanley, K., Miikkulainen, R.: Evolving neural networks through augmenting topologies. Evolutionary Computation. **10**(2) (2002) 99–127
14. Verbancsics, P., Stanley, K.: Evolving static representations for task transfer. Journal of Machine Learning Research **11**(1) (2010) 1737–1763
15. Stanley, K., D'Ambrosio, D., Gauci, J.: A hypercube-based indirect encoding for evolving large-scale neural networks. Artificial Life **15**(2) (2009) 185–212
16. Stone, P., Sutton, R., Kuhlmann, G.: Reinforcement learning for robocup-soccer keepaway. Adaptive Behavior **13(3)** (2006) 165–188
17. Whiteson, S., Stone, P.: Evolutionary function approximation for reinforcement learning. Journal of Machine Learning Research **7(1)** (2006) 877–917
18. Bahceci, E., Miikkulainen, R.: Transfer of evolved pattern-based heuristics in games. In: Proceedings of the IEEE Symposium On Computational Intelligence and Games, Perth, Australia, Morgan Kaufmann (2008) 220–227
19. Lehman, J., Stanley, K.: Abandoning objectives: Evolution through the search for novelty alone. Evolutionary computation **19(2)** (2011) 189–223
20. Mouret, J., Doncieux, S.: Encouraging behavioral diversity in evolutionary robotics: An empirical study. Evolutionary Computation **20(1)** (2012) 91–133
21. Gomes, J., Mariano, P., Christensen, A.: Avoiding convergence in cooperative coevolution with novelty search. In: Proceedings of the International conference on Autonomous Agents and Multi-Agent Systems, ACM (2014) 1149–1156

22. Gomes, J., Mariano, P., Christensen, A.: Devising effective novelty search algorithms: A comprehensive empirical study. In: Proceedings of the Genetic Evolutionary Computation Conference, Madrid, Spain, ACM (2015) 943–950
23. Degrave, J., Burm, M., Kindermans, P., Dambre, J., Wyffels, F.: Transfer learning of gaits on a quadrupedal robot. Adaptive Behavior **23** (2015) 9–19
24. Knudson, M., Tumer, K.: Policy transfer in mobile robots using neuro-evolutionary navigation. In: Proceedings of the Genetic and Evolutionary Computation Conference, Philadelphia, USA, ACM Press (2012) 1411–1412
25. Stanley, K.: Compositional pattern producing networks: A novel abstraction of development. Genetic Programming and Evolvable Machines **8**(2) (2007) 131–162
26. D'Ambrosio, D., Stanley, K.: Scalable multiagent learning through indirect encoding of policy geometry. Evolutionary Intelligence Journal **6**(1) (2013) 1–26
27. Gauci, J., Stanley, K.: A case study on the critical role of geometric regularity in machine learning. In: Proceedings of the AAAI Conference on Artificial Intelligence, Menlo Park, USA, AAAI Press (2008) 628–633
28. Risi, S., Stanley, K.: Confronting the challenge of learning a flexible neural controller for a diversity of morphologies. In: Proceedings of the Genetic and Evolutionary Computation Conference, ACM (2013) 255–261
29. Gomes, J., Christensen, A.: Generic behavior similarity measures for evolutionary swarm robotics. In: Proceedings of the Genetic and Evolutionary Computation Conference, Amsterdam, the Netherlands, ACM Press (2013) 199–206
30. Urbano, P., Georgiou, L.: Improving grammatical evolution in santa fe trail using novelty search. In: Proceedings of the 12th European Conference on Artificial Life, Taormina, Italy, MIT Press (2013) 917–924
31. Lehman, J., Stanley, K.: Efficiently evolving programs through the search for novelty. In: Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation, Portland, USA, ACM (2010) 837–844
32. Velez, R., Clune, J.: Novelty search creates robots with general skills for exploration. In: Proceedings of the Genetic and Evolutionary Computation Conference, Vancouver, Canada, ACM (2014) 737–744
33. Cuccu, G., Gomez, F., Glasmachers, T.: Novelty-based restarts for evolution strategies. In: Proceedings of the Congress on Evolutionary Computation, New Orleans, USA, IEEE Press (2011) 158–163
34. Gomes, J., Urbano, P., Christensen, A.: Progressive minimal criteria novelty search. In Pavn, J., Duque-Mndez, N., Fernndez, R., eds.: Advances in Artificial Intelligence. Springer, Berlin, Germany (2012) 281–290
35. Lehman, J., Stanley, K.: Revising the evolutionary computation abstraction: minimal criteria novelty search. In: Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation, ACM (2010) 103–110
36. Liapis, A., Yannakakis, G., Togelius, J.: Constrained novelty search: A study on game content generation. Evolutionary Computation **23(1)** (2015) 101–129
37. Ghasemi, A., Zahediasl, S.: Normality tests for statistical analysis: A guide for non-statisticians. Int J Endocrinol Metab. **10(2)** (2012) 486–489
38. Flannery, B., Teukolsky, S., Vetterling, W.: Numerical Recipes. Cambridge University Press, Cambridge, UK (1986)