DOI: http://dx.doi.org/10.7551/978-0-262-32621-6-ch075

Evolution, development and learning with predictor neural networks

Konstantin Lakhman and Mikhail Burtsev

Neuromorphic Cognitive Systems Lab, Kurchatov NBICS-Centre,
National Research Center "Kurchatov Institute", 1 Kurchatov sqr., Moscow, Russia
klakhman@gmail.com

Abstract

Study of mechanisms, that can make possible effective learning of artificial systems in complex environments, is one of the key issues in the adaptive systems research. In this paper we make an attempt to implement and test a number of ideas motivated by brain theory. Proposed model integrates evolutionary, developmental and learning phases. The main concept of this paper is the notion of predictor neural network which provide distributed evaluation of the effectiveness of goal-directed behavior on the neuronal level. We also propose learning mechanism based on gradually inclusion of new neuronal functional groups in case when the existing behavior fails to deliver adaptive result. We performed basic computational study of the model to investigate some of its' core properties such as evolution of innate and learned behavior and dynamics of the learning process.

Introduction

The problem of adaptation in multi-goal environments is a great challenge for state of the art machine learning. To solve it we may need to combine best practices from different frameworks as well as novel hypothesis from theoretical neuroscience.

There is a very little effort to integrate evolutionary and learning approaches to solve the problem of adaptive behavior in the field of machine learning despite the significant independent progress in both fields (e.g. neuroevolution (Stanley and Miikkulainen, 2002) and reinforcement learning (Sutton et al., 2011) frameworks). Roughly speaking most of the existing attempts to combine evolution and learning can be divided in two main categories: 1) evolution serves only for optimization of the initial controller, e.g. (Risi et al., 2010), or 2) evolution serves for generation of various kinds of "evaluation" functions, e.g. (Singh et al., 2009). In the latter case "evaluation" guides learning process during an agent's life by contributing to the prediction of the future states of the agent-environment system. From the standpoint of prediction mechanism it is possible to distinguish several approaches: prediction of the expected reward (Singh et al., 2009; Whiteson and Stone, 2006); explicit prediction of the expected parameters of an environment (Nolfi et al., 1994); generation of a separate "teacher" structure (Nolfi and Parisi, 1996). Prediction in terms of the expected reward is not always possible in biological systems since in most cases there is no access to the reinforcement signals. Explicit prediction of an environment's parameters usually performed on the level of the whole controller. Therefore it is difficult to localize individual structural elements of the controller to be modified during learning.

At the same time within the field of theoretical neuroscience there are several suggestions for possible mechanisms underlying distributed formation of prediction in biological systems. However most of this approaches require carefully designed topology of neural microcircuits and give no clue how this topology could emerge in the evolution (Bastos et al., 2012; Hawkins et al., 2011).

Numerous researches in the artificial life field investigates interaction between evolution and learning (Ackley and Littman, 1991; Suzuki and Arita, 2003), including notable Baldwin effect. These works provide great insight into the theoretical understanding of the problem, however they usually utilize synthetic learning models that are not capable to scale well to the real-world problems.

What is required for the neural network learning algorithm to adapt successfully in a multi-goal environment? Starting from the theories of functional systems (Anokhin, 1974) and neuronal group selection (Edelman, 1993) we suggest the following logic to derive such requirements. The neural network should be able to produce initial or *primary* repertoire of basic behaviors. When during lifetime the agent starts to encounter problems in achieving goals with the primary behaviors then this initial repertoire should be extended to allow mission completion for a spectrum of environmental variations. These particular solutions acquired by learning constitute the *secondary* repertoire of behaviors. Primary behaviors can be generated by evolutionary and developmental algorithms or pre-specified by hand. During life-time adaptation learning algorithm should detect failures to execute existing behaviors and generate actions to create alternative solution. We suggest that failure detection should be distributed over the whole neural network. Our hypothesis is that such detection is possible when neu-

rons not only activate but also predict future activity of each other. If some neurons detect mismatch between predicted and actual activities then the learning starts. We require that creation of a new action sequence should not disrupt existing behavior. This can be achieved by integration of new neurons in the network during learning process without changing already existing part of the network. Compared to the previous attempts to formalize principles of the functional systems theory (Red'ko et al., 2004; Komarov et al., 2010) our current model implements adaptive learning by distributed prediction on the level of individual neurons, so we call this architecture *predictor neural network*. In this paper we present computational study of the model consisting of evolutionary and developmental phases for generation of primary behavioral repertoire as well as life-time phase with learning controlled by continuous distributed prediction on the neuronal level.

The Model

Overview

According to the neuronal group selection theory (Edelman, 1993) developmental algorithm in our model targets two different objectives: 1) generation of a primary repertoire of behaviors and 2) generation of *"diversity of repertoires"* required for further self-learning during the life. During development an agent's genotype translates into the neural network controlling the agent's behavior in the environment. Diversity required for the selection of neuronal groups is implemented by simulation of the brain's anatomical regions with *neuronal pools*. Neurons in the same pool have similar but not identical connections with the rest of the network. During development endogenous stochastic activations of neurons lead to selection of highly connected subnetworks. These subnetworks produce primary behaviors of the agent. Remaining neurons form a set of *silent neurons* that are used for learning.

In the theory of functional systems (Anokhin, 1974) every goal-directed behavior serving some adaptive function unfolds in three stages: 1) generation of an action and prediction of its' results; 2) evaluation of results after action completion; 3) formation of a new functional system of neurons if additional leaning is needed. To implement both generation of action and evaluation of results neurons in our model have two types of synapses namely *effector* and *predictor*. Effector synapse is the same as in the standard artificial neural network. Predictor synapse does not propagate excitation but contributes to prediction of future activity of the neuron. Moments of the *mismatch* between expected results and state of the environment are detected in the neurocontroller as neurons calculate discrepancy between predicted and observed activity of each other. Detected mismatch initiates activation of silent neurons to modify agent's behavior. Agent's ability to generate meaningful and useful predictions at the neuronal level is subject to evolutionary selection of predictor connections in the network.

Multi-goal Environment and Neural Net

In the model population of agents evolves on the hypercube. State of the environment is represented by a binary string and at each discrete time step the agent can change the state of one bit. A goal in this environment is defined as the consecutive changes of a particular bits of the state vector. Set of such a goals in the environment forms a branched hierarchy. One goal could be nested in another one, i.e. be a beginning of it, and different goals could have identical starting nested goal. Detailed description of the environment can be found in (Lakhman and Burtsev, 2013).

The agent operates in the environment for a fixed amount of time. Fixed reward is associated with each goal and a value of reward accumulated over life-time affects the reproductive success of an agent at the evolutionary phase of the simulation. However, in contrast to the paper (Lakhman and Burtsev, 2013) we have changed how reward is recovered in the case when the agent attempts to reach the same goal during its' recovery period. Previously, the reward was reseted to zero on the goal's completion and then linearly recovered to the initial value. The agent had no information about accumulated reward. As we introduce learning in the current study there should be some feedback from the environment to the agent that completion of the goal failed. In the current version of the model completion of the goal during value recovery is impossible. If the agent tries to perform an action that would lead to the finalizing of the unrestored goal then the change of the bit is blocked and the state vector of the environment remained the same. Thus, the agent can perceive failure to complete target actions sequence by the neurons predicting the environmental change after successful action. Note that in the current scheme of agent-environment interaction the agent also has no direct information about the value of reward.

The structure of the agent's neurocontroller is determined by a processes of evolution, development and learning. The controller consists of three sets of neurons: input, output and interneurons. Number of input and output neurons is fixed and determined by a dimension n^{env} of the hypercube ($n^{\mathrm{env}} = 8$ for all simulations). Input neurons represent current state vector with one neuron for each bit. Output neurons encode action that agent performs on a current time step. Each pair of output neurons is responsible for turning on or off a particular bit of the state vector. Current agent's action are selected according to the pair of the most active output neurons.

Interneurons of a neurocontroller are organized in layers with no restriction on connectivity. This allows to form both direct and recurrent effector synapses. Recurrent effector synapses propagate excitation with a unit time delay.

Evolutionary Algorithm

Genotype of an agent is defined by a tuple: $G = (P, EC, PC)$, where $P = \{P_\alpha\}$ is a set of neuronal pools, $EC = \{ec_{\gamma\alpha}\}$ is a set of effector connections between pools, $PC = \{pc_{\gamma\alpha}\}$ is a set of predictor connections. During development this genotype is translated to the neural network. Each neuronal pool P_α of the genotype corresponds to a group of neurons with similar connectivity topology and has the following parameters: pool's size c_α, i.e. the number of neurons that will be formed from this pool during development; mean m_α and standard deviation σ_α of the biases of the neurons, that corresponds to a particular pool.

Each effector connection $ec_{\gamma\alpha}$ between pools α and γ has the following parameters: mean $m_{\gamma\alpha}$ and standard deviation $\sigma_{\gamma\alpha}$ of the weights of the synapses, that will be formed between neurons belonging to the corresponding pools; probability of the synapse development $p^{\mathrm{dev}}\left(ec_{\gamma\alpha}\right)$.

The only parameter for predictor connections $pc_{\gamma\alpha}$ is probability of connection development $p^{\mathrm{dev}}\left(pc_{\gamma\alpha}\right)$.

Reproductive success of the agent, i.e. the probability to be selected as a parent for the next population, is directly proportional to the value of reward accumulated over fixed amount of time. An offspring inherits parent's genotype transformed by mutations, including all numerical parameters of the genome introduced above. The structure of the network is modified by *duplication pool* mutation described in detail in the paper (Lakhman and Burtsev, 2013). This mutation substitute a single "parent" pool with two "offspring" pools with the same connectivity structure and half the size of the ancestor. We also used addition and deletion of effector and predictor connections as additional structural mutations. Full list of values of parameters that have been used for the simulations is provided in the Appendix.

Developmental Algorithm

Generation of neuronal pools Development starts with translation of the agent's genotype $G = (P, EC, PC)$ into complete network. Each neuronal pool P_α is filled with c_α neurons. Bias b_i for the neuron v_i is drawn from a normal distribution: $b_i \sim \mathcal{N}\left(m_\alpha, \sigma_\alpha^2\right),\ \forall v_i \in P_\alpha$. Effector synapse is created between neurons $v_i \in P_\alpha$ and $v_j \in P_\beta$ according to the pool connectivity coded in the genome: $w_{ji} \sim \mathcal{N}\left(m_{\beta\alpha}, \sigma_{\beta\alpha}^2\right)$ with probability $p^{\mathrm{dev}}\left(ec_{\beta\alpha}\right)$, if $ec_{\beta\alpha} \in EC$; and $w_{ji} = 0$ otherwise (the synapse is not developing). As a result every neuron in the net has unique connectivity structure both in terms of topology and weights distribution. Development of predictor synapses occurs similarly with the only exception that predictor synapse has no weight.

Similar model of neuroevolution with pools encoded in the genome was proposed in the framework of *Enforced Sub-Populations* evolutionary algorithm (Gomez and Miikkulainen, 1999). However, according to this algorithm selection process was applied directly to neurons and only one neuron was selected from each pool in the development. Therefore this mechanism introduces competition between neurons with different specialization from the same pool. In the our model a similar approach is used to generate a number of variations of the same specialization within single pool.

Selection of primary neuronal groups The next phase of neurocontroller development has a goal to select neuronal groups for the primary behavioral repertoire. To perform this developmental selection we utilize competitive principle proposed in the theory of neuronal group selection (Edelman, 1993) which states that neuronal groups with highest endogenous stochastic activity are selected. As a result network of active neurons constitute neurocontroller at the beginning of agent's life. Less active neurons became silent but form required for further learning set of various local modifications of the network and can be recruited later.

Selection of primary neuronal groups takes place in the isolation from the model environment for a fixed number of time steps T^{sys}. At each time step each neuron of the network produces spontaneous activation with probability p^{spon}. Outputs of the remaining neurons are calculated in the standard way using truncated positive sigmoid activation function (truncation implies zero output in case of the negative neuron's potential). Spontaneous activations of the neurons are designed to provide signal flow in the network in the absence of information about external environment. Total signed value of activation potential received by each neuron is calculated during simulation of endogenous network activity. Within each pool fraction p^{act} of the neurons is selected to participate in primary neuronal groups and remaining neurons become silent. The probability for the neuron to be selected is directly proportional to the corresponding value of accumulated potential.

The role of predictor synapses in the network is prediction of future activations of a neuron. If there is a predictor synapse between neurons v_i and v_j then pre-synaptic neuron's activity predicts that post-synaptic neuron will be active on the next time step (and vice versa in case of absence of activity). During development the predictor synapses between active neurons with prediction rate less then some threshold value L^{sig} (0.5 in the current study) are being deleted from the network. Predictor synapses connecting silent neurons remained intact.

It should be noted that the outputs of silent neurons during the agent's life are always set to zero, regardless of the value of potential they receive by incoming effector synapses.

Learning Algorithm

Innate behavior produced by initial neural network right after development is not optimal and should be complemented by learning.

It is necessary to answer two basic questions when designing a learning algorithm: *1)* When should learning begin? *2)* How to perform learning if it is needed?

Mismatch detection To solve the first problem it is necessary to introduce mechanism for detection of the moments in which additional learning is needed. These are the moments when the agent performs actions that earlier led to adaptive result but now failed. Theory of functional systems (Anokhin, 1974) solves this problem by postulating that at the beginning of goal-directed behavior *functional system* of neurons (neural subnetwork responsible for some adaptive function) generates both the program of actions and prediction of expected outcomes so-called *acceptor of action results* (AAR). If AAR is not consistent with observed features of the environment, obtained immediately after the performance of the planned action, then functional system enters so-called *mismatch state* and initiates learning. Functional systems interact with an environment as well as with other functional systems. In our model predictor synapses make possible distributed evaluation of goal-directed behavior on the level of individual neurons. Several theoretical papers concluded that the biological neurons are able to provide similar functions (Fiorillo, 2008).

Let consider single neuron as a functional system. Signals from other neurons constitute its "environment". At a given time step the neuron can be excited or nonactive. In the model the neuron is in the excited state when its output is greater than zero. Each neuron forms prediction about its own future activity based on predictor synapses from other neurons. Each presynaptic neuron makes its contribution to the total prediction in accordance with the following scheme: if the presynaptic neuron was excited at time step t then it predicts that the postsynaptic neuron will be excited at time step $t+1$, and vice-versa if the presynaptic neuron was in the ground state. Thereby each neuron could calculate probability distribution of its own activity on the next time step based on predictor signals:

$$\begin{gathered} p^{\mathrm{exc}}\left(v_j,t\right)=\frac{\left|\{v_i|\ ps_{ji}\in PS,\, o_i(t-1)>0\}\right|}{\left|PS_j^{\mathrm{act}}\right|} \\ p^{\mathrm{sil}}\left(v_j,t\right)=1-p^{\mathrm{sil}}\left(v_j,t\right) \end{gathered}, \qquad (1)$$

where $p^{\mathrm{exc}}\left(v_j,t\right)$ is the probability of neuron v_j of being active at time step t, PS is a set of network's predictor synapses, o_i is the output of neuron v_i, PS_j^{act} is a set of incoming predictor synapses of neuron v_j that are coming from active neurons. Neuron makes prediction based on this distribution only if one of these probabilities exceeds a fixed threshold $L^{\mathrm{pred}}\in(0.5,1]$. Eq. 1 implies that only active neurons affect prediction.

Described procedure assumes two types of the possible mismatch situations: I-type mismatch implies the absence of the neuron's activity when it was predicted and II-type mismatch implies the presence of the activity when it was not predicted.

Learning via specialization of "silent" neurons According to the systems-selection theory (Shvyrkov, 1986) learning at the neuronal level consists of neuronal specializations for the problem situation. This specialization occurs via selection of neurons from the "reserve" of low active cells or, in our case, from silent neurons.

Mismatch detection on the level of neurons allows effectively locate a specific place in a neural net where it is necessary to make modifications during learning. Thus for each neuron with I-type mismatch a silent neuron from the same pool with highest effector input is found and added to the set of network's active neurons. This just activated neuron is *specialized* on solving current problem. Effector synapses of this neuron are pruned to maximize recognition of the current neuronal input. This implies deletion of effector synapses from neurons that have not been excited on the current time step. Additionally we add a strong excitatory synapse $w\sim\mathcal{U}\left(0.5,1\right)$ from activated neuron to the mismatched one. This synapse could potentially lead to the elimination of the mismatch on the postsynaptic neuron in the similar behavioral situations in the future. We also add additional predictor synapses from the neurons that predicted activation of the mismatched neuron to the set of predictor synapses of the activated neuron.

Learning for the II-type mismatch occurs in exactly the same manner except that we add strong inhibitory connection from activated to mismatched neuron in order to avoid mismatch in the similar behavioral situations.

As a result of learning the initial neural network is expanded by distributed integration of neurons specialized on solution of the current problem. This scheme fits into the conceptual framework considering learning as following the same principles as evolutionary process: generation of diversity and selection (Burtsev, 2008). The specialization of the silent neurons could be interpreted as a local mutations of neurocontroller, which can potentially lead to successful behavior. Nonetheless, learning terminates only if there is no mismatch between organism's expectations and actual state of environment detected at the neuronal level.

Experimental Results

As the first step we have studied how learning algorithm affects efficiency of an agents in terms of accumulated reward. 10 environments with different goals structure were generated randomly. Two modifications of the model – with or without learning (only with developmental phase) were run in every environment 5 times (50 runs in total for each version). Then we detected the best population in terms of the average accumulated reward in each run. For all agents in the best population development was performed 5 times with different random seeds. Finally, we run these 5 differ-

ent controllers from all $2^{n^{\text{env}}} = 2^8 = 256$ initial states of the environment. Resulting values of average reward are shown on the Fig. 1A. We have not found statistical difference between efficiency of the agents evolved with and without learning – t-test showed $p-\text{value} = 0.23$. However, if learning is switched off for the agents evolved with it then average reward decreases significantly indicating importance of learning in this case. Effect of learning is detailed on the Fig. 1B where majority of runs show lower reward without learning. As one can see several runs do not suffer from learnings shutdown, however this doesnt mean that learning didnt play significant role in these runs during the evolution.

We have also tested our learning mechanism against simple random activation of silent neurons (Fig. 1A). Random learning implies that silent neuron has a fixed probability of integration into the network at a given time step. We randomized learning of the agents evolved with learning. Several activation probabilities were examined from the interval $[0.01, 0.1]$, but results are presented only for the best value of 0.5. Random specialization of neurons do not increase reward compared to normal learning and even has some tendency to decrease efficiency (paired samples t-test with $p-\text{value} = 0.08$) comparing to the results of the same agents but without any learning.

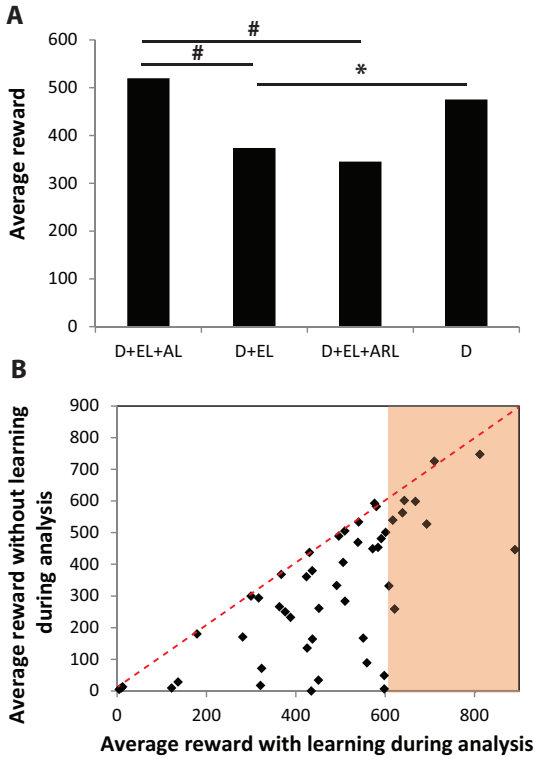

Figure 1: Effects of developmental and learning components of the model on reward. **A)** Average reward for the different types of agents (D – development, EL – learning during evolution, AL – learning during an analysis phase, ARL – "random learning" during an analysis phase). $\#$ denotes paired samples t-test with $p-\text{value} < 0.001$, $*$ denotes t-test with $p-\text{value} = 0.012$. **B)** Comparison of the efficiency of the agents that evolved with learning and the same agents without learning. Each dot represents averaged value for the best population of corresponding evolutionary run. The range of reward values of the best 20 runs is highlighted.

Results suggest that evolutionary algorithm alone is able to make non-learning agents almost as effective as learning ones. We found that after evolution the non-learning agents have in the neural net more pools and smaller pool sizes compared to the agents with learning (data is not shown). Larger pool sizes of learning agents indicate that selection support mechanism for variation of neuronal groups during learning.

As the next step we have studied evolution of the best run among those that use both developmental and learning phases in more details. For the best agent of each generation 5 different phenotypes were produced by randomly initialized development and then tested with and without learning. Resulting dynamics of innate and learned behaviors are presented on the Fig. 2A. Over the period of evolution an efficiency of learned behavior usually grows first and innate behavior follows. The increase of the best agent's reward after learning can occur both while efficiency of the innate behavior is decreasing (Fig. 2B) or remains unchanged (Fig. 2C).

As we have showed earlier (Lakhman and Burtsev, 2013) behavior evolved in the similar hypercubic environment consists of two phases: convergence phase when the sequence of actions depends on the particular initial state and stable repeating cycle of actions. We will call the latter behavioral cycle. Analysis of behavior cycles evolution for the best run shows that significant growth of learning efficiency is accompanied by explosion in variability (up to 200 variations) of behavioral cycles and subsequent change of the dominant behavioral strategy (data is not shown). The number of observed behaviors are sharply reduced when we analyze the

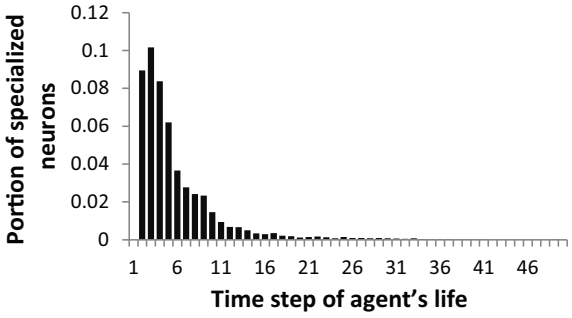

Figure 3: Dynamics of silent neurons specialization during an agent's life (averaged over the best 10 evolutionary runs).

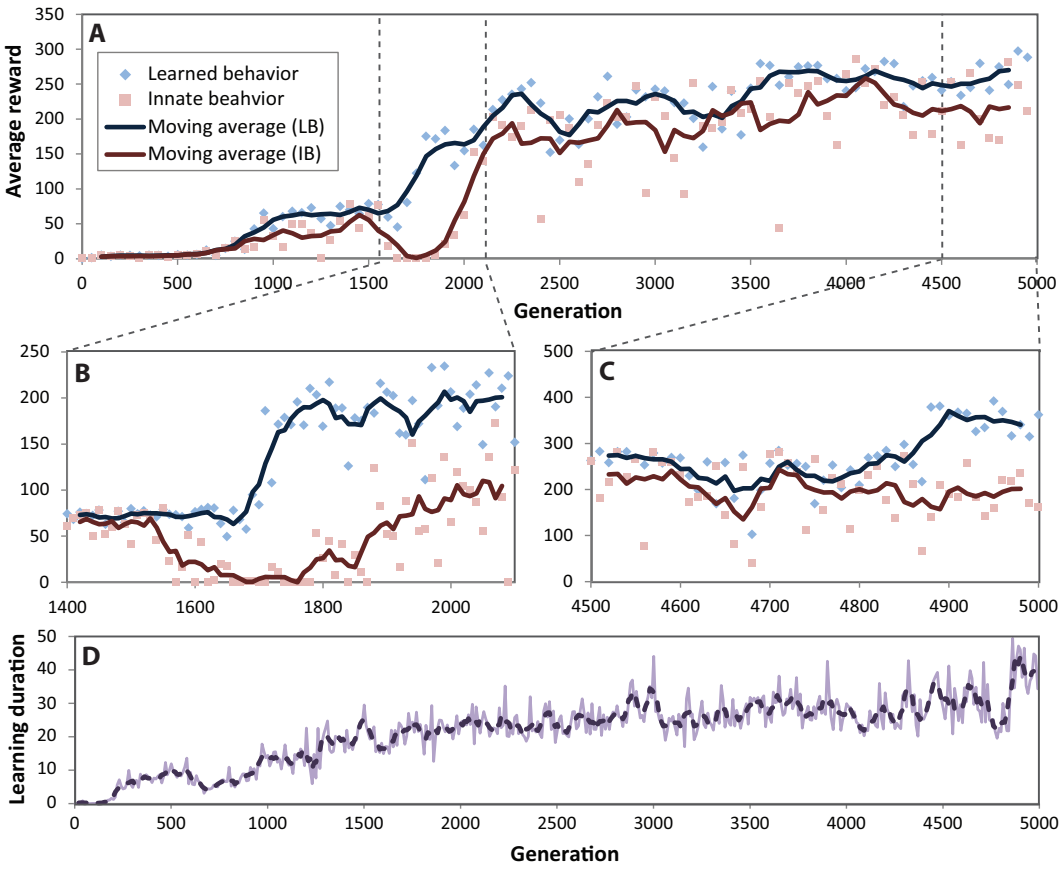

Figure 2: Evolutionary dynamics of innate and learned behaviors. **A)** Dynamics of the average rewards for the innate (pink) and learned (blue) behaviors for the best evolutionary run (agents were evaluated in the environment for 100 time steps and therefore rewards is roughly 2.5 times less than for the best run on the Fig. 1). The values are calculated for the best agent of the each 50-th generation. **B-C)** Periods when average reward after learning increases. The values are calculated for the best agent of the each 10-th generation. **D)** Averaged duration of learning. Duration of learning is calculated as the last time step of the agent's life when silent neurons specialize.

same agents but without learning. This might represent a general scenario when learning guide evolution of complex adaptive behavior in natural and artificial systems.

Reward accumulated by the agent during learning sometimes increases together with the average learning time, i.e. the period of silent neurons specialization(for example, starting from 4850-th generation on the Fig. 2D). Dynamics of the specialization of silent neurons during the learning process averaged over the best 10 evolutionary runs is shown on the Fig. 3. On the average about 50% of all silent neurons are turned on lifelong, but most of them are specialized at the beginning of life (up to 10-th time step). However, sometimes learning takes place even as late as 100-th time step for some of the evolutionary runs.

We found that there are the periods of evolution when success of learning significantly correlates with the number of specialized silent neurons (Fig. 4). This happens when behavior of the agents is not fully formed (i.e. the dominant strategy is not stable) and learning can play significant role in improvement of partial actions sequences. Later in the course of evolution (for example in the 3500-th generation on the Fig. 2) this correlation disappears and roughly the same number of silent neurons are being activated regardless of the difference between learned and innate behavior (data is not shown). It seems that the role of learning is changing here, it mainly "corrects errors" that were made in the developmental process. On this stage the development can generate neurocontrollers that accumulate the same reward with or without learning.

To investigate how the neural network can generalize after learning we put the agent in each one of the states of the environment to learn then this agent with neural net after learning was tested starting from all the other states (Fig. 5A). For this analysis we have chosen the agent from the later stage of evolution (the best agent of 4950-th generation) and generated neurocontroller that shows big difference between re-

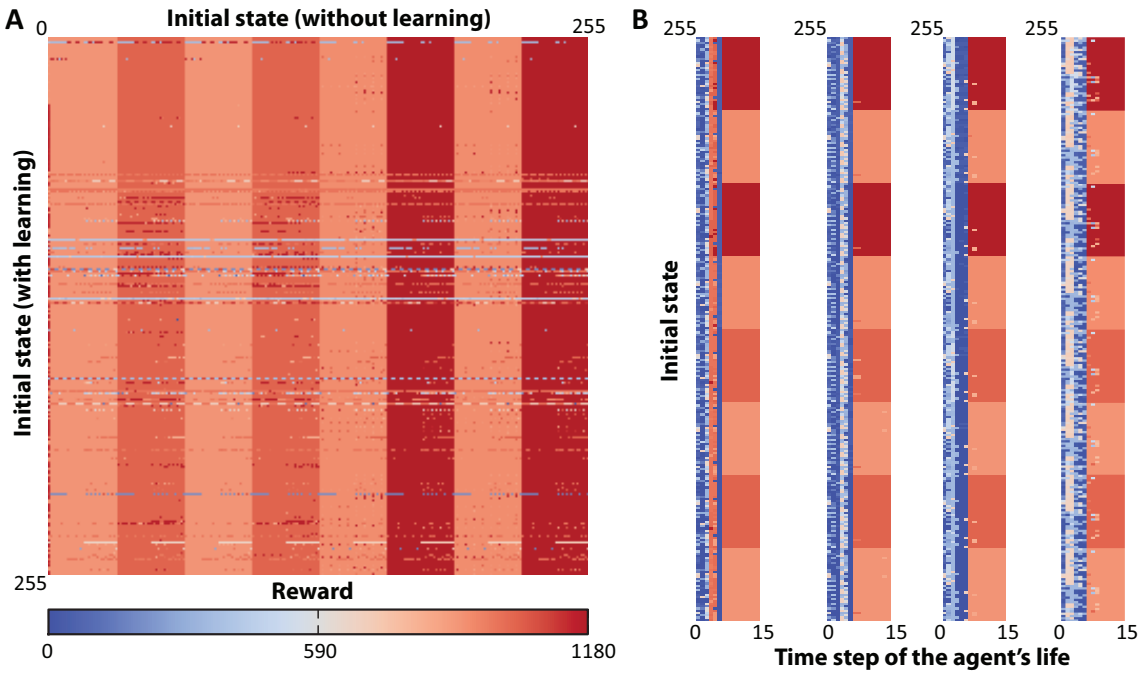

Figure 5: Neurocontroller performance after learning. **A)** After learning from each state of the environment the same learned neural network was started from each other state with the learning turned off. **B)** Performance of neurocontroller on the different stages of learning for a 4 random initial states(see text for the details).
All results presented on the figure are for the best agent of the 4950-th generation from the Fig. 2 (see text for more details).

wards with and without learning. Fig. 5A demonstrates that the neurocontroller after learning from one state of the environment is also efficient in the most cases when the agent starts life from the different state. "Stripes" on the figure correspond to the sets of initial states with different accumulated reward for the same neural net. This means that the agent achieves the same goals in the same order, but can be more or less optimal in terms of the number of actions performed.

The same agent that was used to produce Fig. 5A was utilized to study dynamics of the learning process (Fig. 5B). The agent was started with learning from a few states and then after each time step we evaluated efficiency of the controller, obtained so far, by running its non-learning version from all states of the environment. The results demonstrate that learning occurs only at the earliest stage of agent's life during the first 6–10 time steps. This is also consistent with the average dynamics of the specialization of silent neurons presented on the Fig. 3. Important feature of the learning process is that it is non gradual. Learning can even lead to temporary degradation of the intermediate controller performance. However when learning is finished the agent behavior abruptly becomes adaptive. It is important to notice that behavioral policies produced by intermediate controllers can be found in the behavior of the agents from earlier generations.

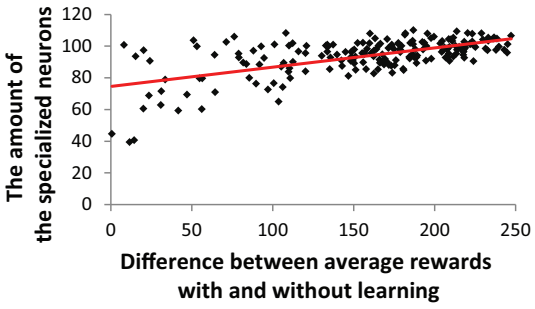

Figure 4: The number of specialized silent neurons positively correlates with the quality of learning. Each point represents value of reward averaged over all initial states of the environment for one development of the best agent of 1781-th generation from the Fig. 2. Spearman correlation coefficient $\rho = 0.52$ with $p-\text{value} < 0.001$

Conclusions

In this paper we presented a novel model of adaptive behavior that combines evolutionary, developmental and learning phases. In the model we introduced for the first time a number of principles inspired by brain theory: 1) selection of neuronal groups during development to build primary net-

work structure; 2) evolution of neuronal anatomy instead of exact neuronal connectivity to create diversity of subnetworks for learning during life-time; 3) distributed prediction mechanism that makes possible to detect mismatch between expected and perceived states of the environment for the initiation of learning on the neuronal level; 4) learning by specialization of "silent" neurons that produces non-disruptive modification of the network structure suitable for the acquisition of alternative behaviors in multi-goal environments.

In computational study we attempted to reveal some basic properties of this model, such as dynamics of the innate and learned behavior in evolution, dynamics of learning in terms of controller and behavior modification. Despite the fact that evolutionary algorithm "finds way" to make non-learning agents almost as effective as learning ones, our results demonstrate the important role of learning in evolution.

Acknowledgements

This work was partially supported by RFBR grant #13-04-01273.

References

Ackley, D. H. and Littman, M. L. (1991). Interactions between learning and evolution. volume 10, pages 487–509.

Anokhin, P. (1974). *Biology and Neurophysiology of the Conditioned Reflex and Its Role in Adaptive Behavior*. Pergamon, Oxford.

Bastos, A., Usrey, M. W., Adams, R., Mangun, G., Fries, P., and Friston, K. (2012). Canonical microcircuits for predictive coding. *Neuron*, 76(4):695 – 711.

Burtsev, M. (2008). Basic principles of adaptive learning through variation and selection. In Bullock, S., Noble, J., Watson, R., and Bedau, M. A., editors, *Proceeding of the ALIFE XI*, ALIFE XI, pages 88–93. MIT Press, Cambridge, MA.

Edelman, G. M. (1993). Neural darwinism: Selection and reentrant signaling in higher brain function. *Neuron*, 10(2):115 – 125.

Fiorillo, C. D. (2008). Towards a general theory of neural computation based on prediction by single neurons. *PLoS ONE*, 3(10):e3298.

Gomez, F. J. and Miikkulainen, R. (1999). Solving non-markovian control tasks with neuroevolution. In *Proceedings of the IJCAI'99*, volume 2 of *IJCAI'99*, pages 1356–1361, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Hawkins, J., Ahmad, S., and Dubinsky, D. (2011). Htm cortical learning algorithms (white paper).

Komarov, M., Osipov, G., and Burtsev, M. (2010). Adaptive functional systems: Learning with chaos. *Chaos*, 20(4):045119.

Lakhman, K. and Burtsev, M. (2013). Neuroevolution results in emergence of short-term memory in multi-goal environment. In *Proceeding of the GECCO '13*, GECCO '13, pages 703–710, New York, NY, USA. ACM.

Nolfi, S., Elman, J. L., and Parisi, D. (1994). Learning and evolution in neural networks. *Adaptive Behavior*, 3(1):5–28.

Nolfi, S. and Parisi, D. (1996). Learning to adapt to changing environments in evolving neural networks. *Adaptive Behavior*, 5(1):75–98.

Red'ko, V. G., Prokhorov, D. V., and Burtsev, M. S. (2004). Theory of functional systems, adaptive critics and neural networks. In *Proceeding of the IEEE International Joint Conference on Neural Networks*, volume 3, pages 1787–1792. IEEE.

Risi, S., Hughes, C. E., and Stanley, K. O. (2010). Evolving plastic neural networks with novelty search. *Adaptive Behavior*, 18(6):470–491.

Shvyrkov, V. B. (1986). Behavioral specialization of neurons and the system-selection hypothesis of learning. In *Human Memory and Cognitive Capabilities*, pages 599–611. Elsevier, Amsterdam.

Singh, S., Lewis, R., and A.G., B. (2009). Where do rewards come from? In Taatgen, N. and van Rijn, H., editors, *31st Annual Conference of the Cognitive Science Society*, pages 2601–2606, Austin, TX.

Stanley, K. O. and Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10:99–127.

Sutton, R. S., Modayil, J., Delp, M., Degris, T., Pilarski, P. M., White, A., and Precup, D. (2011). Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction. In *Proceeding of the AAMAS '11*, volume 2, pages 761–768, Richland, SC.

Suzuki, R. and Arita, T. (2003). The baldwin effect revisited: Three steps characterized by the quantitative evolution of phenotypic plasticity. In *Advances in Artificial Life*, volume 2801 of *Lecture Notes in Computer Science*, pages 395–404. Springer Berlin Heidelberg.

Whiteson, S. and Stone, P. (2006). Evolutionary function approximation for reinforcement learning. *The Journal of Machine Learning Research*, 7:877–917.

Appendix

We used the following parameters of the artificial environment in all simulations: dimensionality of the hypercube environment n^{env} = 8 bits; agent lifetime duration T^{life} = 250 time steps; recovery time of a goal reproductive value T^{rec} = 30 time steps; probability of random change of the state vector's bit p^{stoch} = 0.0085 (for each bit); reproductive value, associated with every goal R = 20. Evolutionary algorithm was run with parameters: population N^{p} = 250 agents; period of evolution T^{ev} = 5000 generations; probability of the effector connection weight mutation p^{weight} = 0.6 (for each synapse); variance of the mutation of mean of effector connection weight $d^{\mathrm{weight-mean}}$ = 0.08; variance of the mutation of variance of effector connection weight $d^{\mathrm{weight-variance}}$ = 0.08; probability of adding an effector connection $p^{\mathrm{add-con}}$ = 0.1 (for the whole network); probability of deleting an effector connection $p^{\mathrm{del-con}}$ = 0.05 (for the whole network); probability of adding an predictor connection $p^{\mathrm{add-pred-con}}$ = 0.1; probability of deleting an predictor connection $p^{\mathrm{del-pred-con}}$ = 0.05; probability of pool's duplication p^{dup} = 0.007 (for the whole network). Developmental algorithm was run with parameters: duration T^{sys} = 200 time steps; probability of neuron's spontaneous activation p^{spon} = 0.1; fraction of activated neurons p^{act} = 0.4 (for each pool); minimum prediction rate L^{sig} = 0.5. The mismatch threshold L^{pred} used in the learning algorithm was set to 0.5.