



The latest from Google Research

# Permutation-Invariant Neural Networks for Reinforcement Learning

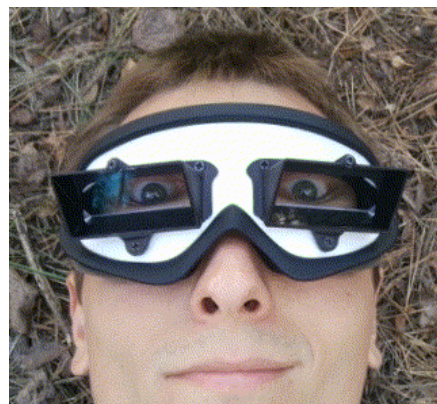
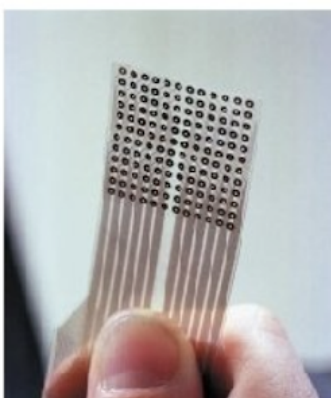
Thursday, November 18, 2021

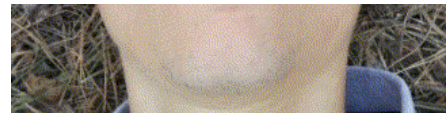
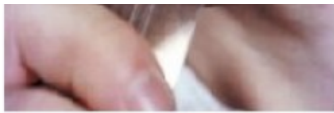
Posted by David Ha, Staff Research Scientist and Yujin Tang, Research Software Engineer, Google Research, Tokyo

*"The brain is able to use information coming from the skin as if it were coming from the eyes. We don't see with the eyes or hear with the ears, these are just the receptors, seeing and hearing in fact goes on in the brain."*

— [Paul Bach-y-Rita](#), quoted in [Livewired](#)

People have the amazing ability to use one sensory modality (e.g., touch) to supply environmental information normally gathered by another sense (e.g., vision). This adaptive ability, called [sensory substitution](#), is a phenomenon well-known to neuroscience. While difficult adaptations — such as adjusting to seeing things [upside-down](#), learning to ride a "[backwards](#)" bicycle, or learning to "see" by interpreting visual information emitted from a grid of electrodes placed on one's tongue — require anywhere from weeks, months or even years to attain mastery, people are able to eventually adjust to sensory substitutions.

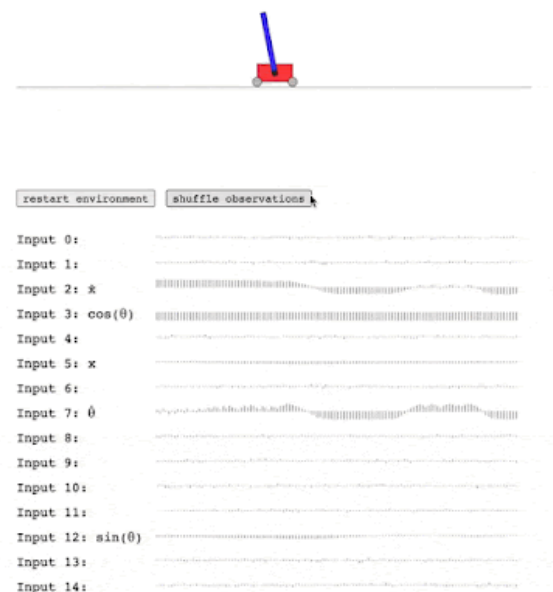
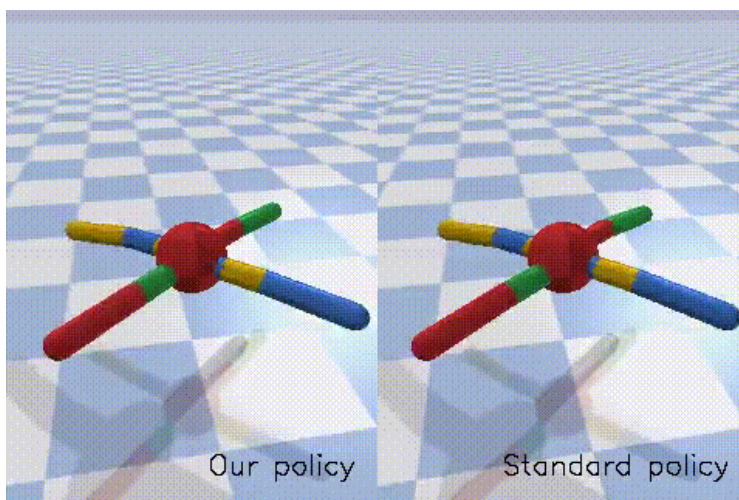




Examples of Sensory Substitution. Left: [Tongue Display Unit](#) ([Maris and Bach-y-Rita, 2001](#); Image: [Kaczmarek, 2011](#)). Right: “Upside down goggles” initially conceived by Erismann and Kohler in 1931. (Image [Wikipedia](#)).

In contrast, most neural networks are not able to adapt to sensory substitutions *at all*. For instance, most [reinforcement learning](#) (RL) agents require their inputs to be in a pre-specified format, or else they will fail. They expect fixed-size inputs and assume that each element of the input carries a precise meaning, such as the pixel intensity at a specified location, or state information, like position or velocity. In popular RL benchmark tasks (e.g., [Ant](#) or [Cart-pole](#)), an agent trained using current [RL algorithms](#) will fail if its sensory inputs are changed or if the agent is fed additional noisy inputs that are unrelated to the task at hand.

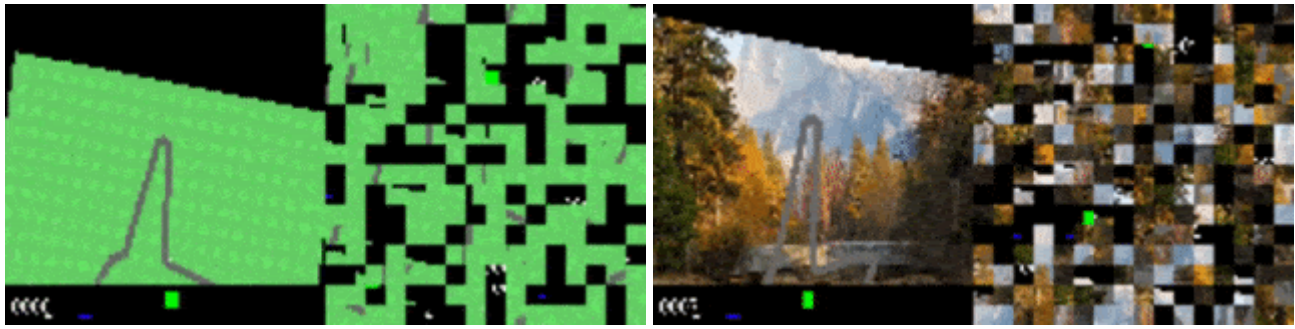
In “[The Sensory Neuron as a Transformer: Permutation-Invariant Neural Networks for Reinforcement Learning](#)”, a [spotlight paper](#) at [NeurIPS 2021](#), we explore *permutation invariant* neural network agents, which require each of their sensory neurons (receptors that receive sensory inputs from the environment) to figure out the meaning and context of its input signal, rather than explicitly assuming a fixed meaning. Our experiments show that such agents are robust to observations that contain additional redundant or noisy information, and to observations that are corrupt and incomplete.



*Permutation invariant* reinforcement learning agents adapting to sensory substitutions. Left: The ordering of the ant’s 28 observations are randomly shuffled every 200 time-steps. Unlike the standard policy, our policy is not affected by the suddenly permuted inputs. Right: Cart-pole agent given many redundant noisy inputs (Interactive [web-demo](#)).

In addition to adapting to sensory substitutions in state-observation environments (like the ant and cart-pole examples), we show that these agents can also adapt to sensory substitutions in complex

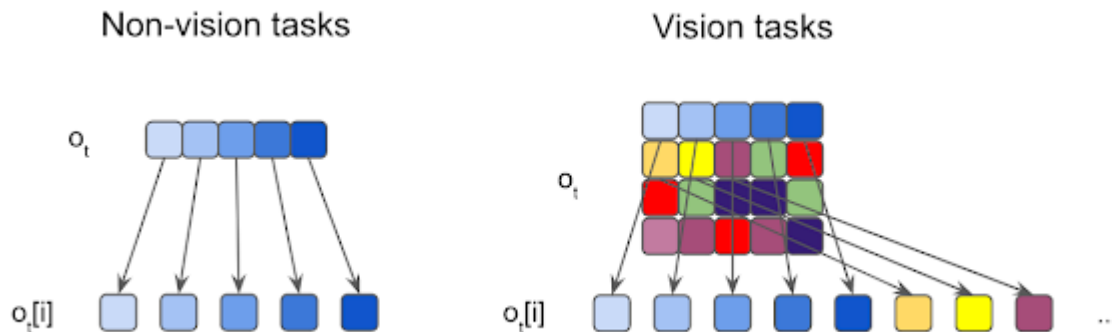
visual-observation environments (such as a [CarRacing](#) game that uses only pixel observations) and can perform when the stream of input images is constantly being reshuffled:



We partition the visual input from CarRacing into a 2D grid of small patches, and shuffled their ordering. Without any additional training, our agent still performs even when the original training background (left) is replaced with new images (right).

## Method

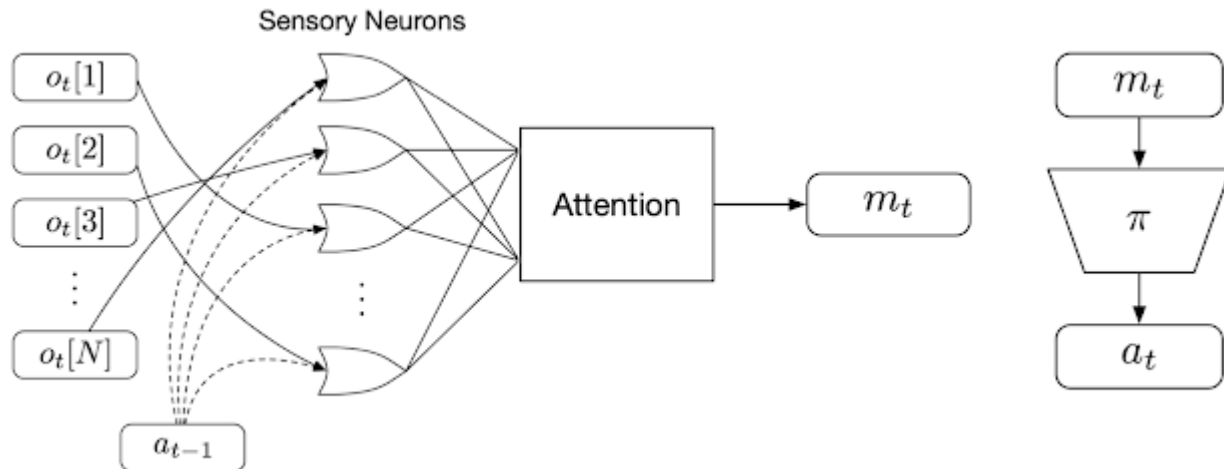
Our approach takes observations from the environment at each time-step and feeds each element of the observation into distinct, but identical neural networks (called “sensory neurons”), each with no fixed relationship with one another. Each sensory neuron integrates over time information from only their particular sensory input channel. Because each sensory neuron receives only a small part of the full picture, they need to [self-organize](#) through communication in order for a global coherent behavior to [emerge](#).



**Illustration of observation segmentation.** We segment each input into elements, which are then fed to independent sensory neurons. For non-vision tasks where the inputs are usually 1D vectors, each element is a scalar. For vision tasks, we crop each input image into non-overlapping patches.

We encourage neurons to communicate with each other by training them to broadcast messages. While receiving information locally, each individual sensory neuron also continually broadcasts an output message at each time-step. These messages are consolidated and combined into an output vector, called the *global latent code*, using an attention mechanism similar to that applied in the [Transformer](#) architecture. A policy network then uses the global latent code to produce the action that the agent will use to interact with the environment. This action is also fed back into each

sensory neuron in the next time-step, closing the communication loop.



Overview of the permutation-invariant RL method. We first feed each individual observation ( $o_t$ ) into a particular sensory neuron (along with the agent's previous action,  $a_{t-1}$ ). Each neuron then produces and broadcasts a message independently, and an attention mechanism summarizes them into a global latent code ( $m_t$ ) that is given to the agent's downstream policy network ( $\pi$ ) to produce the agent's action  $a_t$ .

Why is this system permutation invariant? Each sensory neuron is an identical neural network that is not confined to only process information from one particular sensory input. In fact, in our setup, the inputs to each sensory neuron are not defined. Instead, each neuron must *figure out* the meaning of its input signal by paying attention to the inputs received by the other sensory neurons, rather than explicitly assuming a fixed meaning. This encourages the agent to process the entire input as an **unordered set**, making the system to be permutation invariant to its input. Furthermore, in principle, the agent can use as many sensory neurons as required, thus enabling it to process observations of arbitrary length. Both of these properties will help the agent adapt to sensory substitutions.

## Results

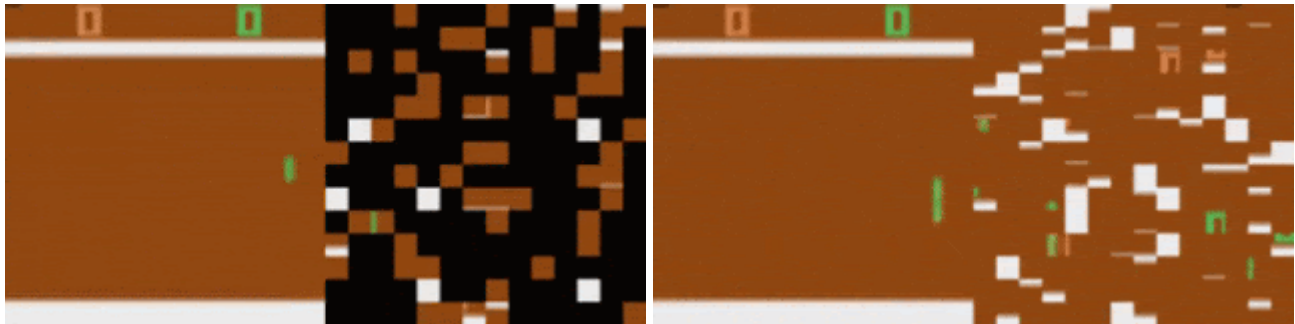
We demonstrate the robustness and flexibility of this approach in simpler, state-observation environments, where the observations the agent receives as inputs are low-dimensional vectors holding information about the agent's states, such as the position or velocity of its components. The agent in the popular **Ant** locomotion task has a total of 28 inputs with information that includes positions and velocities. We shuffle the order of the input vector several times during a trial and show that the agent is rapidly able to adapt and is still able to walk forward.

In **cart-pole**, the agent's goal is to swing up a cart-pole mounted at the center of the cart and balance it upright. Normally the agent sees only five inputs, but we modify the cartpole environment to provide 15 shuffled input signals, 10 of which are pure noise, and the remainder of which are the actual observations from the environment. The agent is still able to perform the task, demonstrating the system's capacity to work with a large number of inputs and attend only to channels it deems useful. Such flexibility may find useful applications for processing a large unspecified number of signals, most of which are noise, from ill-defined systems.

We also apply this approach to high-dimensional vision-based environments where the observation is a stream of pixel images. Here, we investigate screen-shuffled versions of vision-based RL



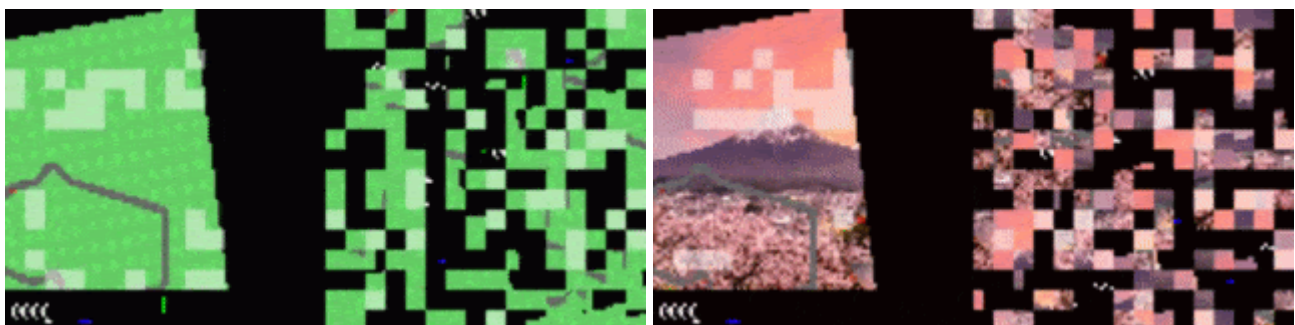
environments, where each observation frame is divided into a grid of patches, and like a puzzle, the agent must process the patches in a shuffled order to determine a course of action to take. To demonstrate our approach on vision-based tasks, we created a shuffled version of Atari Pong.



Shuffled Pong results. Left: Pong agent trained to play using only 30% of the patches matches performance of Atari opponent. Right: Without extra training, when we give the agent more puzzle pieces, its performance increases.

Here the agent's input is a variable-length list of patches, so unlike typical RL agents, the agent only gets to "see" a subset of patches from the screen. In the puzzle pong experiment, we pass to the agent a random sample of patches across the screen, which are then fixed through the remainder of the game. We find that we can discard 70% of the patches (at these fixed-random locations) and still train the agent to perform well against the built-in Atari opponent. Interestingly, if we then reveal additional information to the agent (e.g., allowing it access to more image patches), its performance increases, even without additional training. When the agent receives all the patches, in shuffled order, it wins 100% of the time, achieving the same result with agents that are trained while seeing the entire screen.

We find that imposing additional difficulty during training by using unordered observations has additional benefits, such as improving generalization to unseen variations of the task, like when the background of the [CarRacing](#) training environment is replaced with a novel image.



Shuffled CarRacing results. The agent has learned to focus its attention (indicated by the highlighted patches) on the road boundaries. Left: Training environment. Right: Test environment with new background.

## Conclusion

The permutation invariant neural network agents presented here can handle ill-defined, varying

observation spaces. Our agents are robust to observations that contain redundant or noisy information, or observations that are corrupt and incomplete. We believe that permutation invariant systems open up numerous possibilities in reinforcement learning.

If you're interested to learn more about this work, we invite readers to read our [interactive article](#) ([pdf](#) version) or watch our [video](#). We also released [code](#) to reproduce our experiments.



Google

[Google](#) · [Privacy](#) · [Terms](#)