Introduction

Group 10 (Abdullah Al Afique, Tanha Ahmed, Nijhum, Israt Asha)

Abdullah Al Afique

2031117642

Israt Asha

1921758042

Tanha Ahmed Nijhum

2031523642

Project

AI Self-Driving Car Racing Game

Group 10 (Abdullah Al Afique, Tanha Ahmed, Nijhum, Israt Asha)

Objective

The goal of this project is to create an autonomous self-driving car simulation that learns to navigate a 2D track without human input. Using virtual sensors, the car improves its driving decisions—such as steering and speed control—through trial and error, aiming to complete tracks efficiently while avoiding collisions.

Group 10 (Abdullah Al Afique, Tanha Ahmed, Nijhum, Israt Asha)

Problem statement

Manually collecting labeled training data for self-driving behavior is time-consuming, rigid, and often impractical. This project addresses the challenge of enabling an autonomous car to learn how to navigate complex tracks without supervision, using adaptive neural networks that evolve through interaction with the environment.

Challenges in Learning to Drive Autonomously

Designing a self-driving car requires real-time decision-making in a dynamic environment. The car must learn to navigate unpredictable tracks, avoid collisions, and adapt to various conditions without predefined rules or external guidance. These challenges make it difficult to apply traditional logic-based approaches effectively.

Why Traditional Methods Fall Short

Rule-based systems and supervised learning depend on predefined logic or large labeled datasets, which are impractical in this setting. They fail to generalize across different tracks and require extensive human input. As environments become more complex, such methods become rigid and ineffective for developing adaptive driving behavior.

Our approach

We used an evolutionary learning approach where each car is controlled by a neural network that takes distance sensor inputs and outputs driving actions like steering or acceleration. These networks are evaluated based on how far the car travels without crashing. Over generations, the best-performing networks are selected, mutated, and recombined to evolve smarter driving behavior without any manual training data.

Implementation of NEAT

Group 10 (Abdullah Al Afique, Tanha Ahmed, Nijhum, Israt Asha)

What is NEAT

NEAT, or **NeuroEvolution of Augmenting Topologies,** is an algorithm that evolves neural networks over time, **inspired by natural selection.**

- Instead of training with labeled data, it uses a genetic approach
- Survival of the fittest. Networks that perform better are selected to reproduce, just like in nature

Why Use NEAT Instead of Supervised Learning?

Unlike supervised learning,

- NEAT doesn’t require labeled datasets or predefined input-output mappings.
- In a self-driving simulation, collecting accurate training data is time-consuming and often unrealistic.
- NEAT learns purely through interaction with the environment

Components of NEAT

Fitness

Measures how well a neural network performs (e.g., how far the car drives without crashing).

Mutation

Randomly alters network connections or structure to introduce new behaviors.

Crossover

Combines parts of two parent networks to create a new child network.

Speciation

Groups similar networks into species to preserve innovation and maintain diversity.

Fitness

- Each car's performance is measured by how far it travels without crashing.
- This distance is used to calculate the fitness score

Fitness Function:

$$\text{fitness} = \frac{\text{distance}}{\text{CAR\_SIZE\_X}/2}$$

A higher fitness score increases the likelihood of a network being selected for reproduction.

Mutation

Mutation introduces random changes in the neural networks to promote exploration and innovation.

- New connections forming between sensor inputs and outputs
- Small tweaks in weight values affecting how the car turns or accelerates
- Entirely new hidden nodes being added, allowing more complex driving logic

For example, a car that initially veers into a wall might, through mutation, gain a connection that helps it respond more effectively to radar input — keeping it on track longer in the next generation.

Crossover

- **Crossover is like breeding in nature**

- NEAT takes two high-performing neural networks and combines their structures to produce a new child network. This allows the offspring to potentially inherit the best traits of both parents.

In the car simulation, one parent might be good at turning, while the other might be better at controlling speed. Through crossover, a new network may emerge that can do both — improving performance in the next generation of races.

Speciation

When NEAT evolves neural networks, speciation is like creating small groups or teams of networks that are similar to each other.

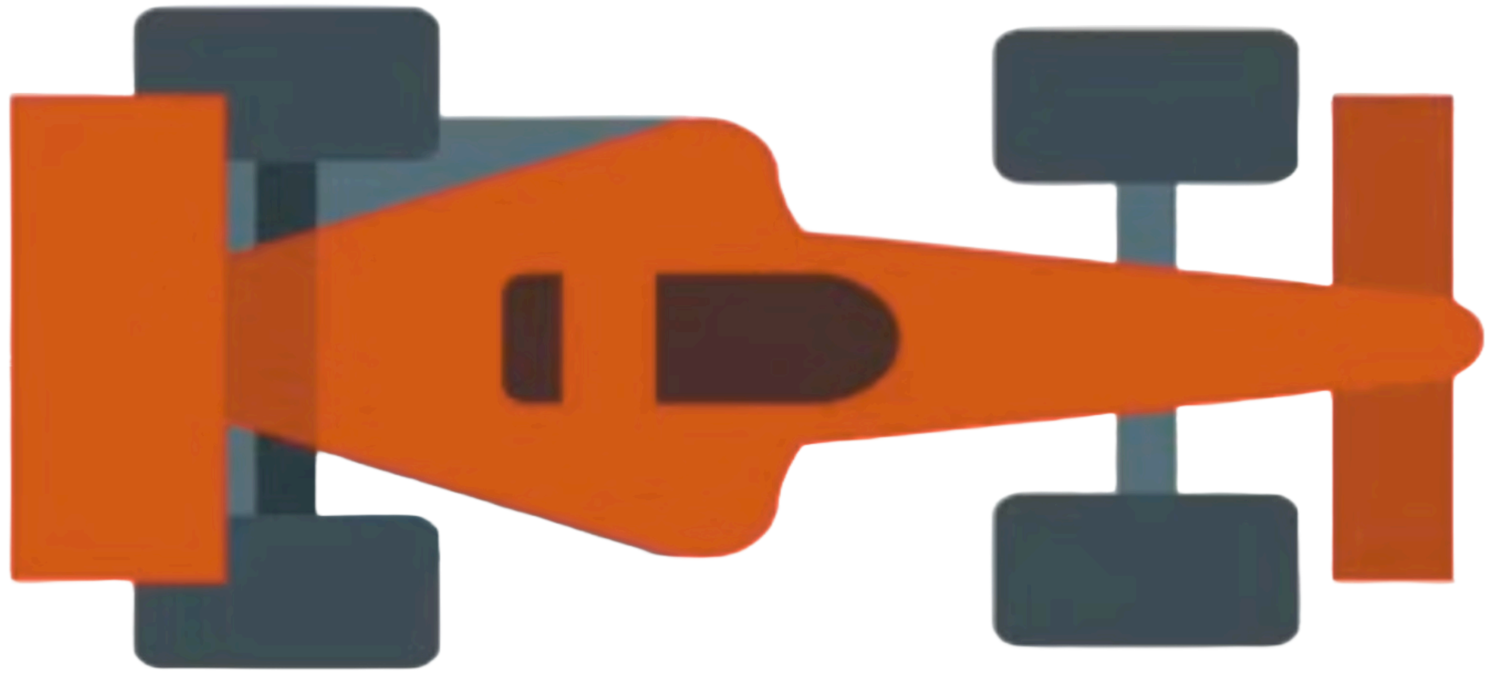
By letting each group evolve separately, NEAT keeps a variety of strategies alive. This helps it find better solutions overall and avoids getting stuck with just one "okay" way of doing things.

System Architecture

The system is composed of two core components: the **autonomous car** and the simulated **track environment**. The car interacts with the map using sensor inputs and neural network outputs, while the environment provides the structure, constraints, and feedback necessary for learning. Together, they form a closed loop of perception, decision-making, and real-time response.

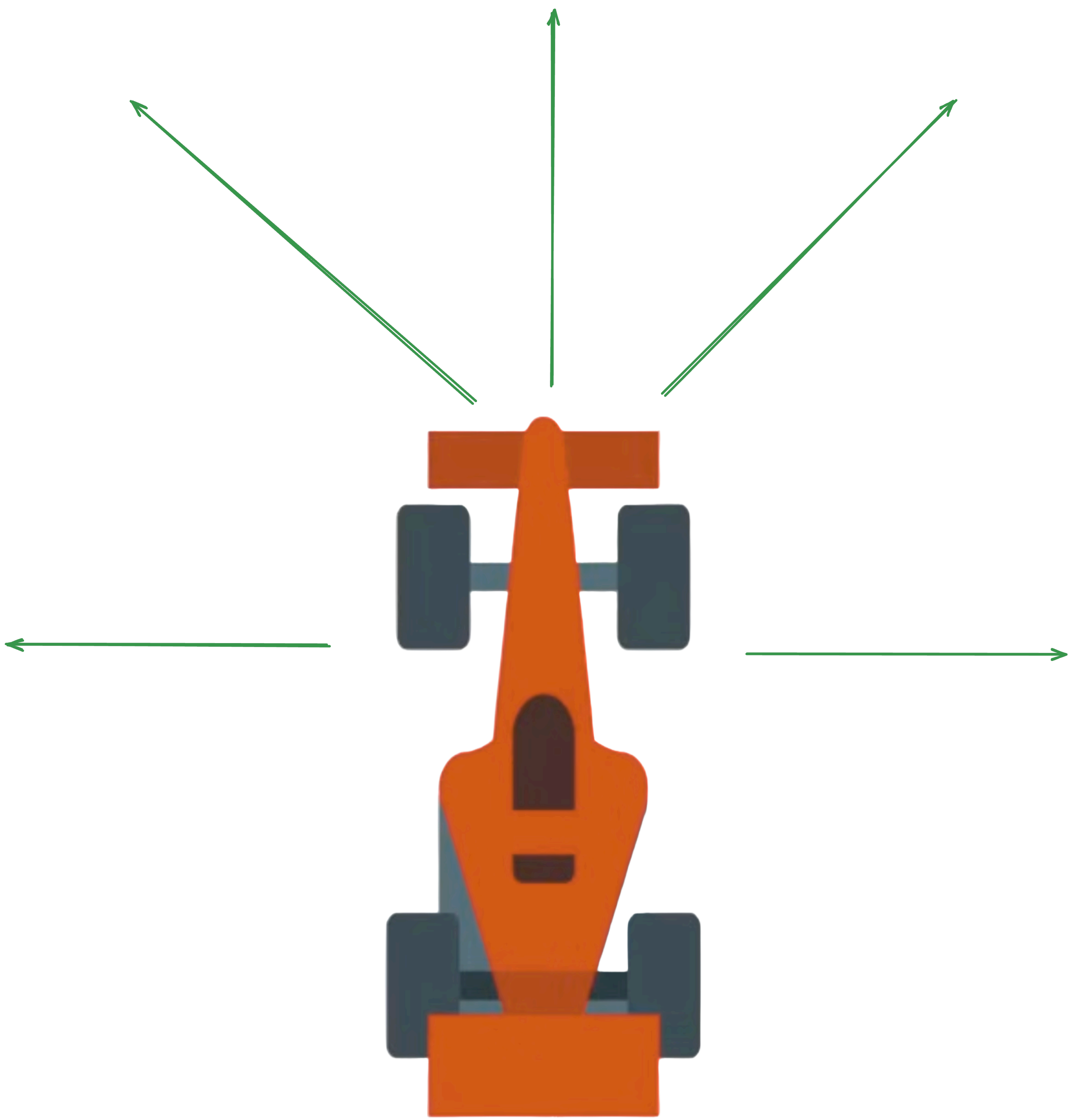
Car

Each car is an autonomous agent equipped with virtual sensors that measure distances to nearby walls. These sensor readings act as inputs to a neural network that determines the car's driving behavior. The car maintains information about its position, angle, speed, and collision status. As it moves through the environment, it constantly updates its center position, checks for collisions, and recalculates sensor data in real-time.



Radars sensors

Each car is controlled by a neural network that takes in inputs from 5 radar sensors, which detect distances to nearby walls at various angles.



Right

Moves to the right

Left

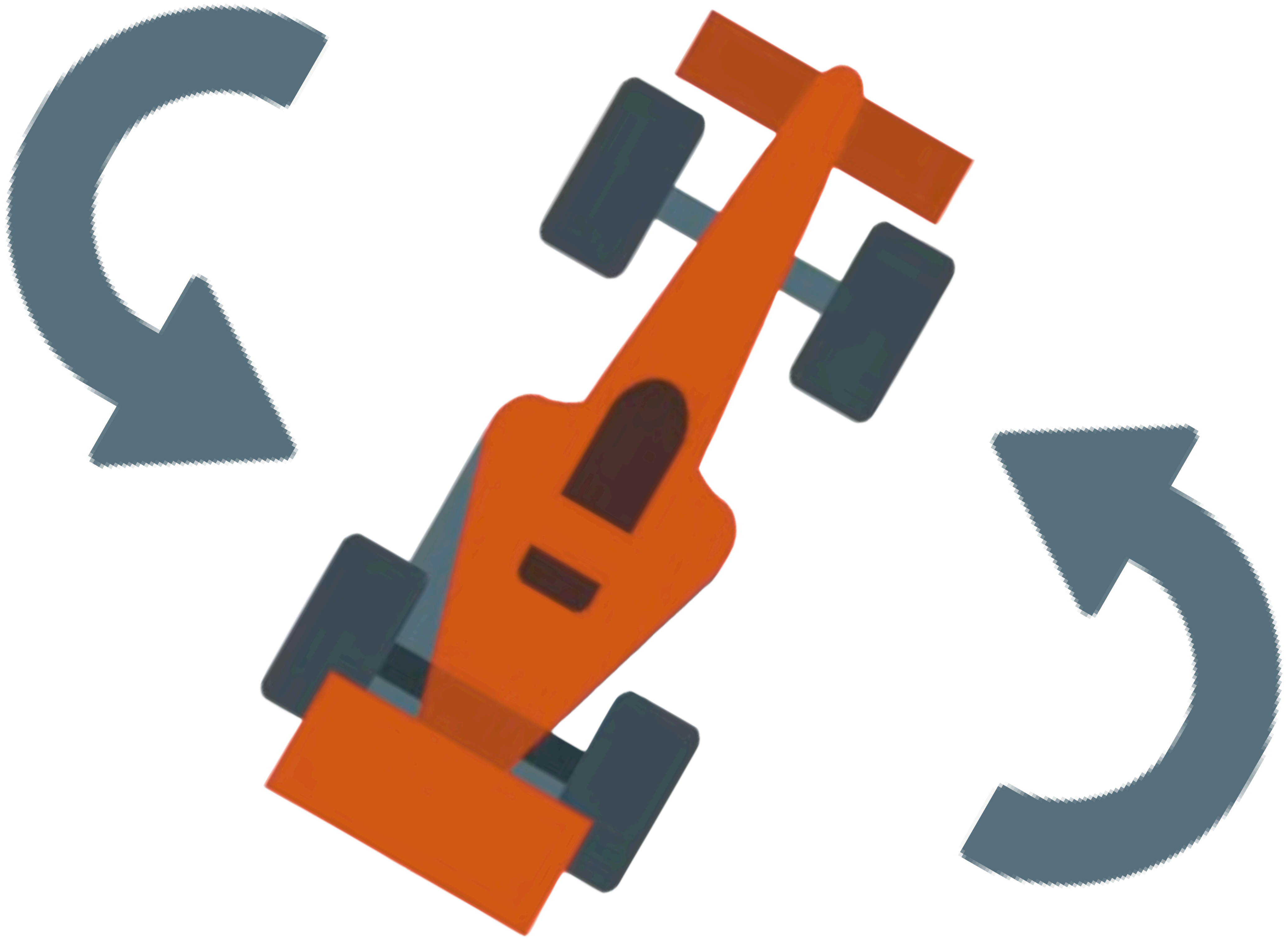
Moves to the left

Speed +

Accelerate and increases the velocity

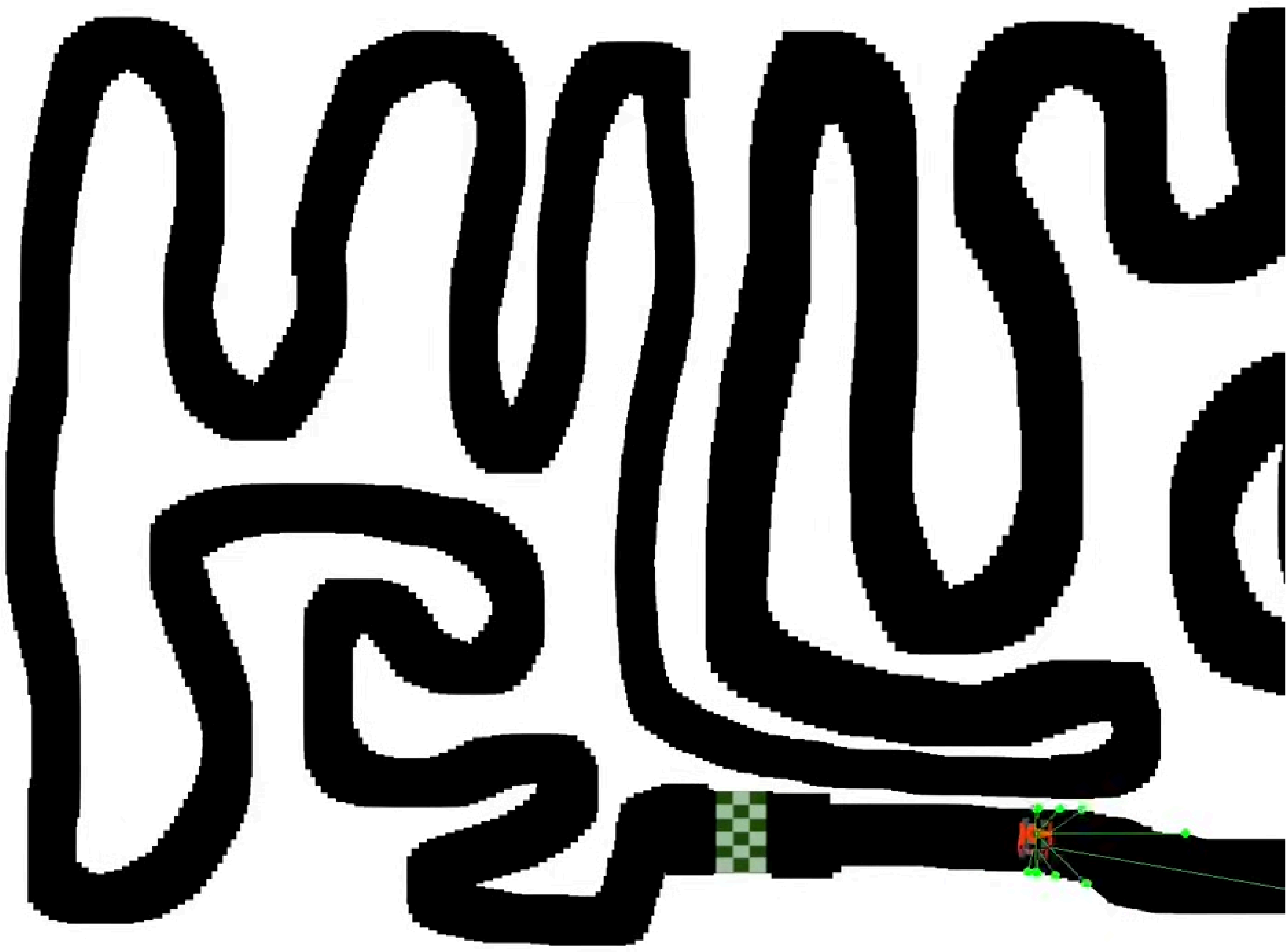
Speed -

Deacelerate and decreases the velocity



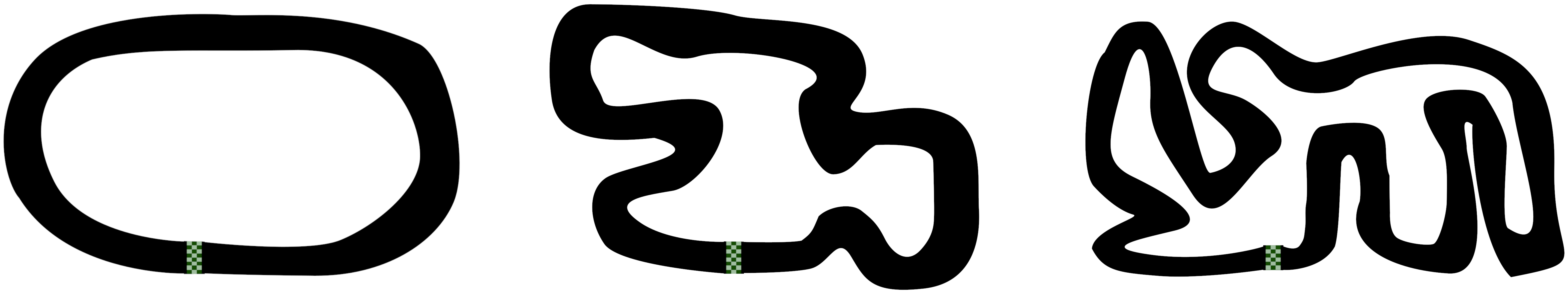
Map

The environment is a 2D track loaded from an image file, where white pixels represent wall boundaries. The simulation uses this image to detect collisions: if any corner of the car touches a white pixel, the car is marked as crashed. The map serves as the fixed layout against which all cars are evaluated. It plays a critical role in shaping the difficulty of the task and acts as the challenge the neural networks must learn to overcome through navigation and survival.



Levels

To introduce progressive difficulty and encourage more generalized learning, the simulation includes different track layouts across levels.



Group 10 (Abdullah Al Afique, Tanha Ahmed, Nijhum, Israt Asha)

Methodology

How NEAT works

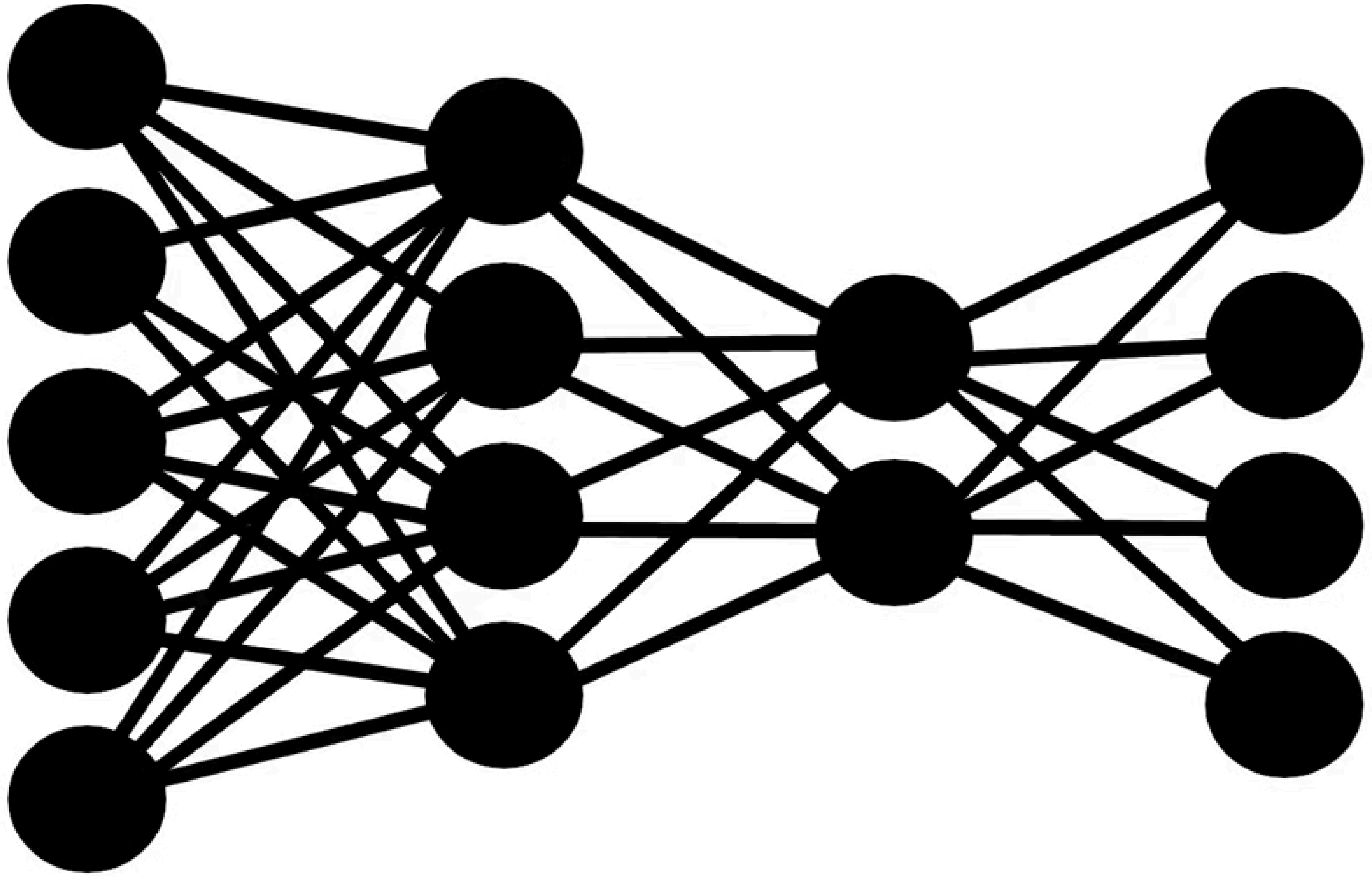
Group 10 (Abdullah Al Afique, Tanha Ahmed, Nijhum, Israt Asha)

Methodology

NEAT allows the car to learn through experience by evolving neural networks over time. Each car is driven by a neural network that makes decisions based on sensor input. After evaluating performance, NEAT evolves the best-performing networks using principles inspired by natural selection.

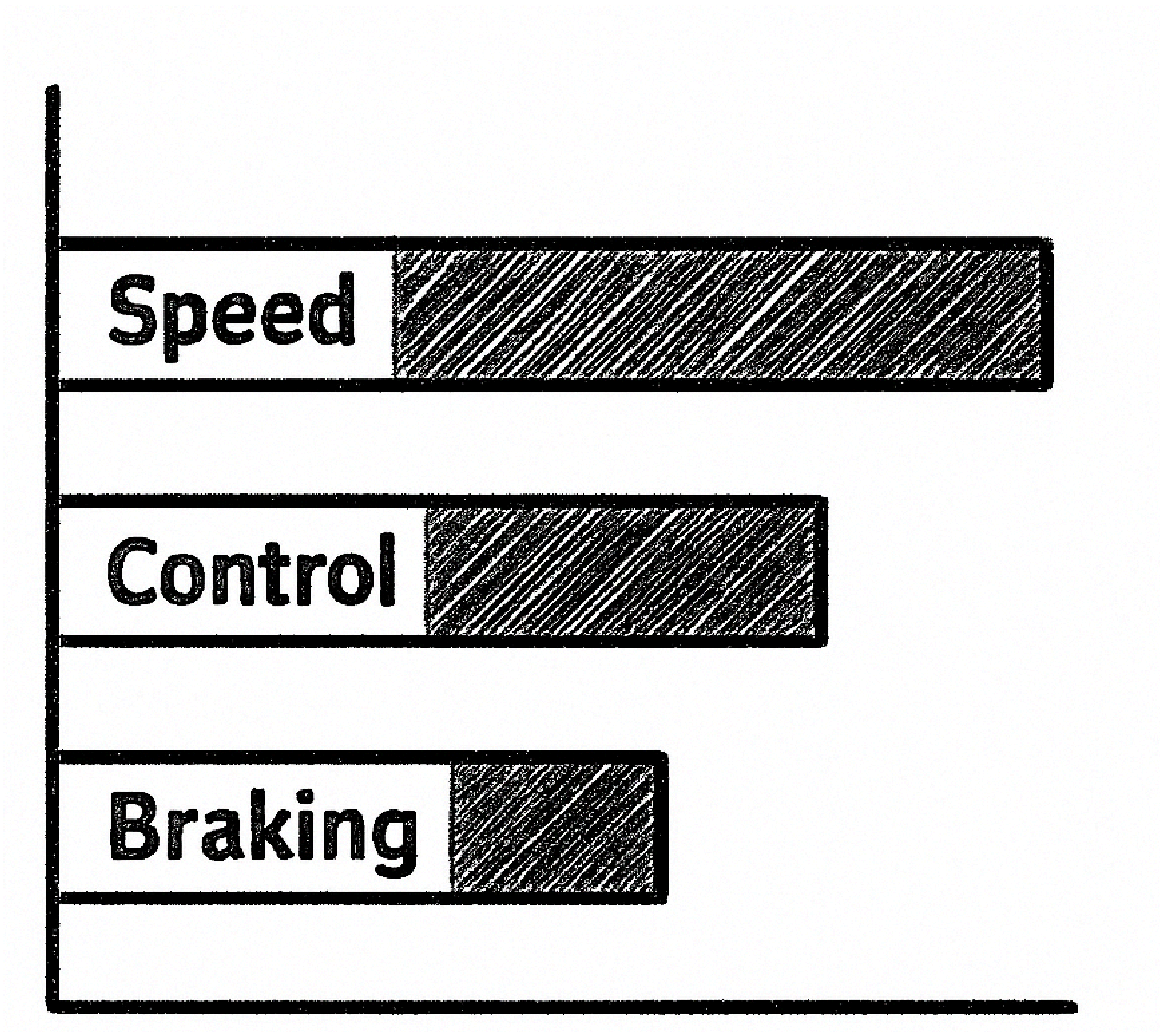
NEAT Starts with Simple Neural Networks

NEAT begins with a population of minimal neural networks, each containing only input and output layers with no hidden nodes. In our simulation, these networks receive 5 radar sensor inputs and produce 4 outputs: steer left, steer right, accelerate, and brake. All connections are initially sparse and random, allowing the system to grow complexity only as needed.



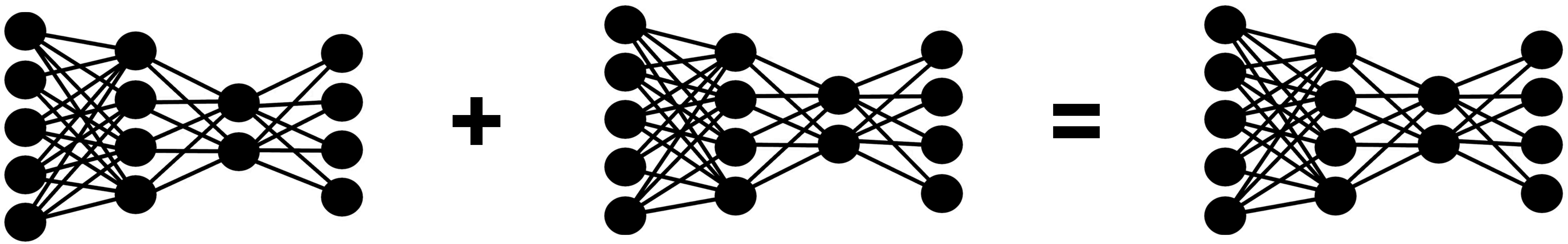
Evaluating Each Car's Performance

Each neural network controls a car in the simulation. The car's survival depends on how well it can drive without hitting a wall. The performance is measured using a fitness function based on distance traveled. Networks that result in cars driving farther without crashing receive higher fitness scores and are more likely to be selected for the next generation.



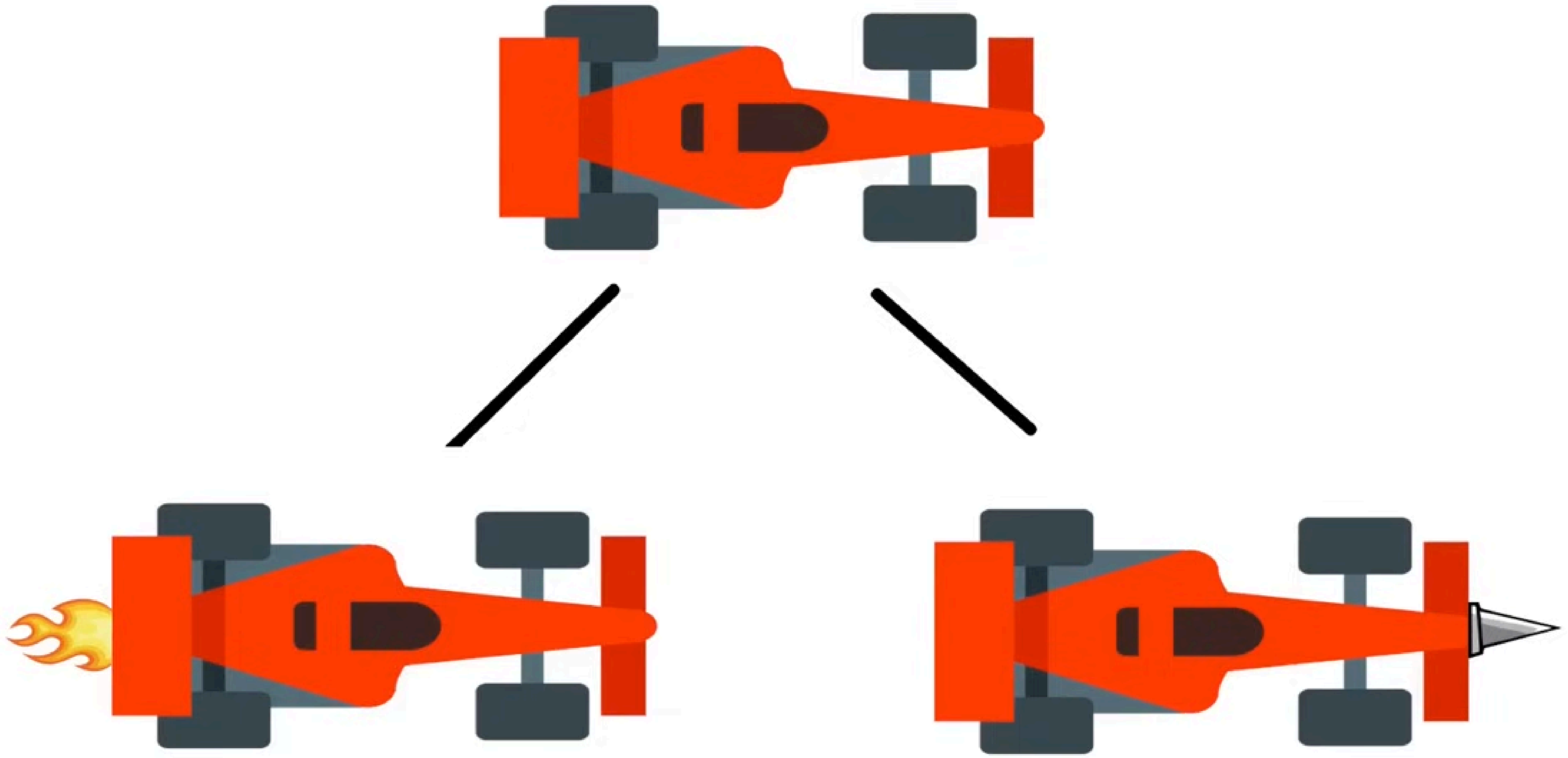
Selecting and Reproducing the Fittest Networks

After evaluation, the networks with the highest fitness scores are selected as parents. NEAT uses a selection process to ensure the best-performing networks have a greater chance of producing offspring, while poorly performing networks are discarded. This encourages the retention and improvement of successful driving behaviors.



Crossover: Combining Two Networks to Create Offspring

Crossover merges two parent networks by combining their structure and weights. Matching connections (based on innovation numbers) are inherited from either parent, while excess or disjoint genes are usually inherited from the more successful one. This creates a new child network that may inherit useful traits from both parents—for example, better turning from one and better speed control from the other.

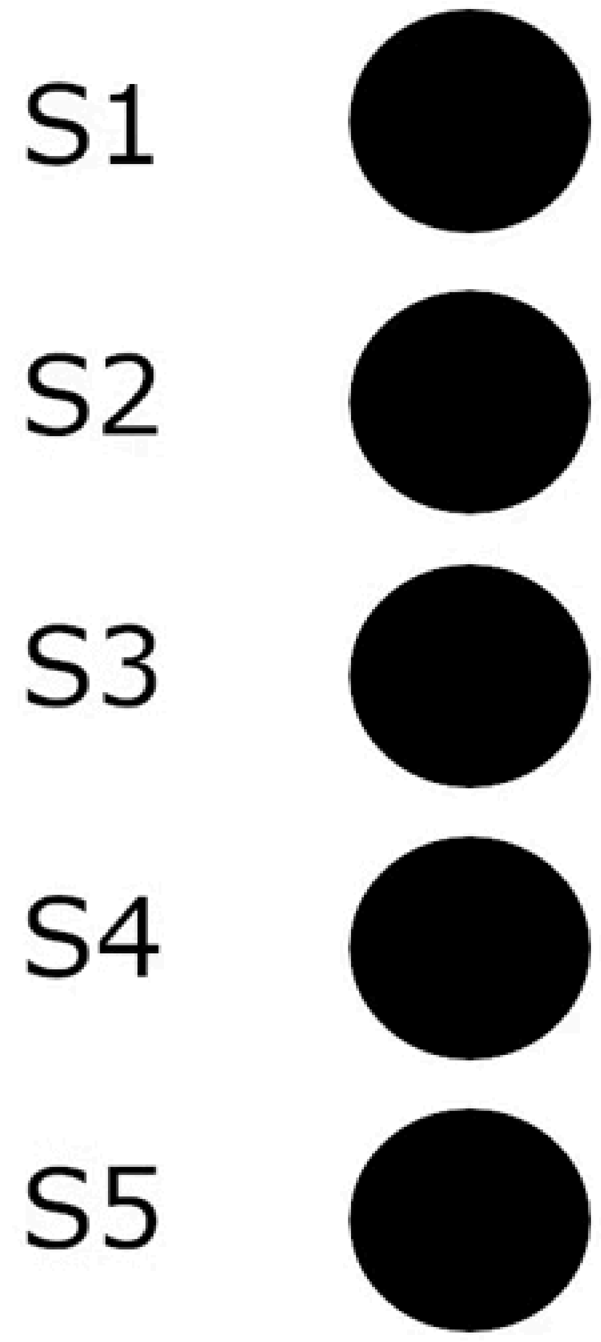


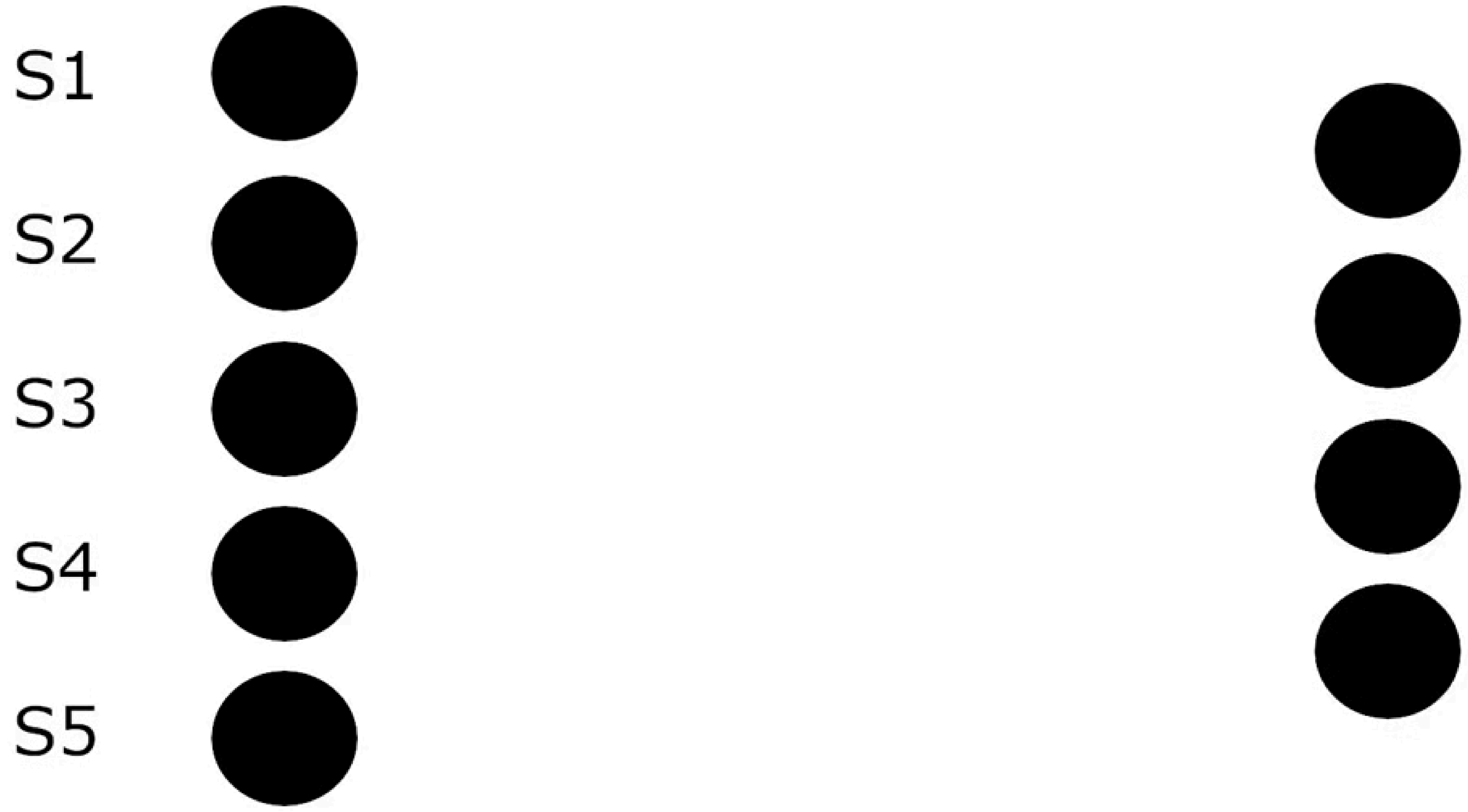
Deep dive

Diving into the Neural Network

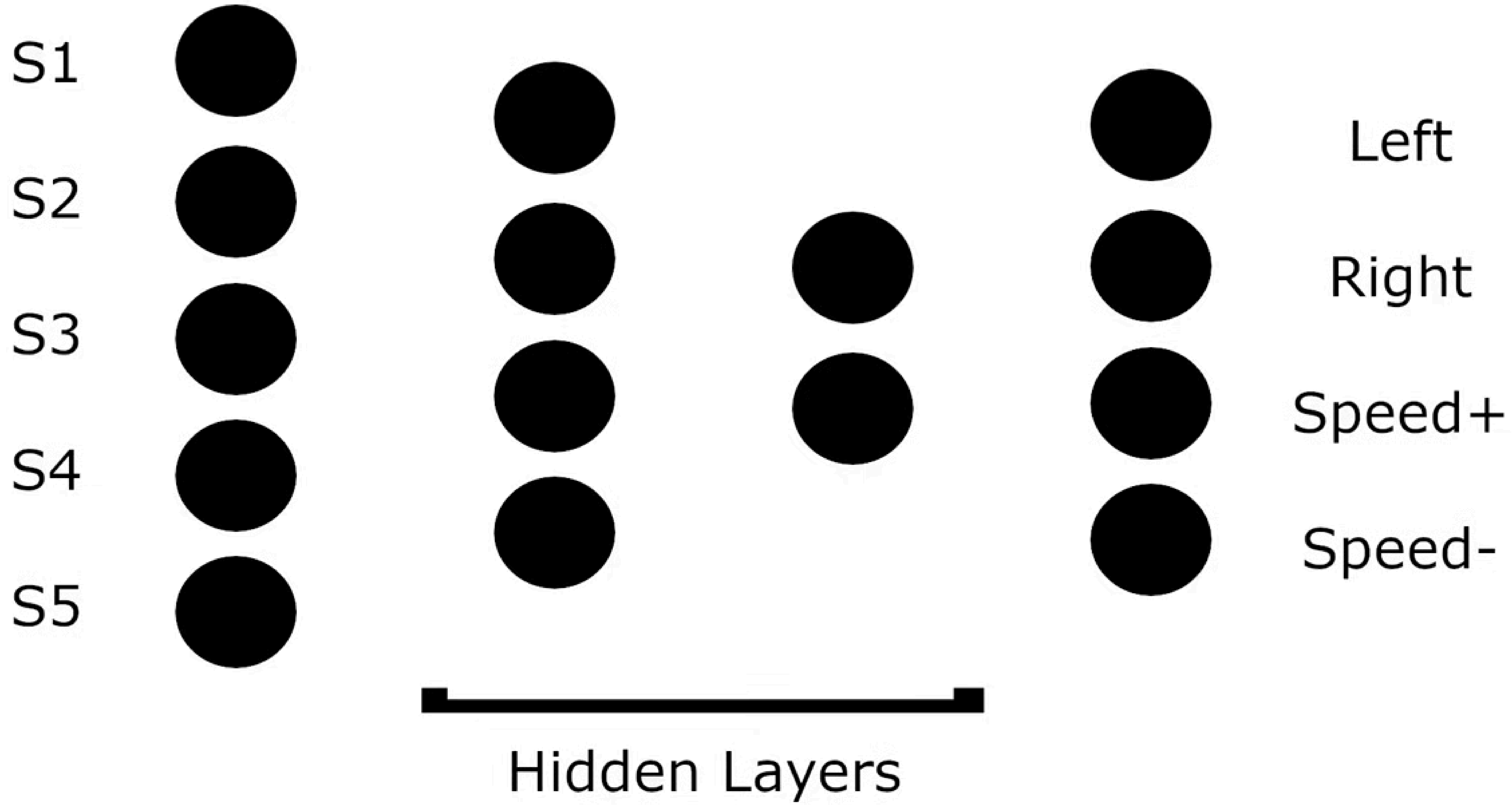
Group 10 (Abdullah Al Afique, Tanha Ahmed, Nijhum, Israt Asha)

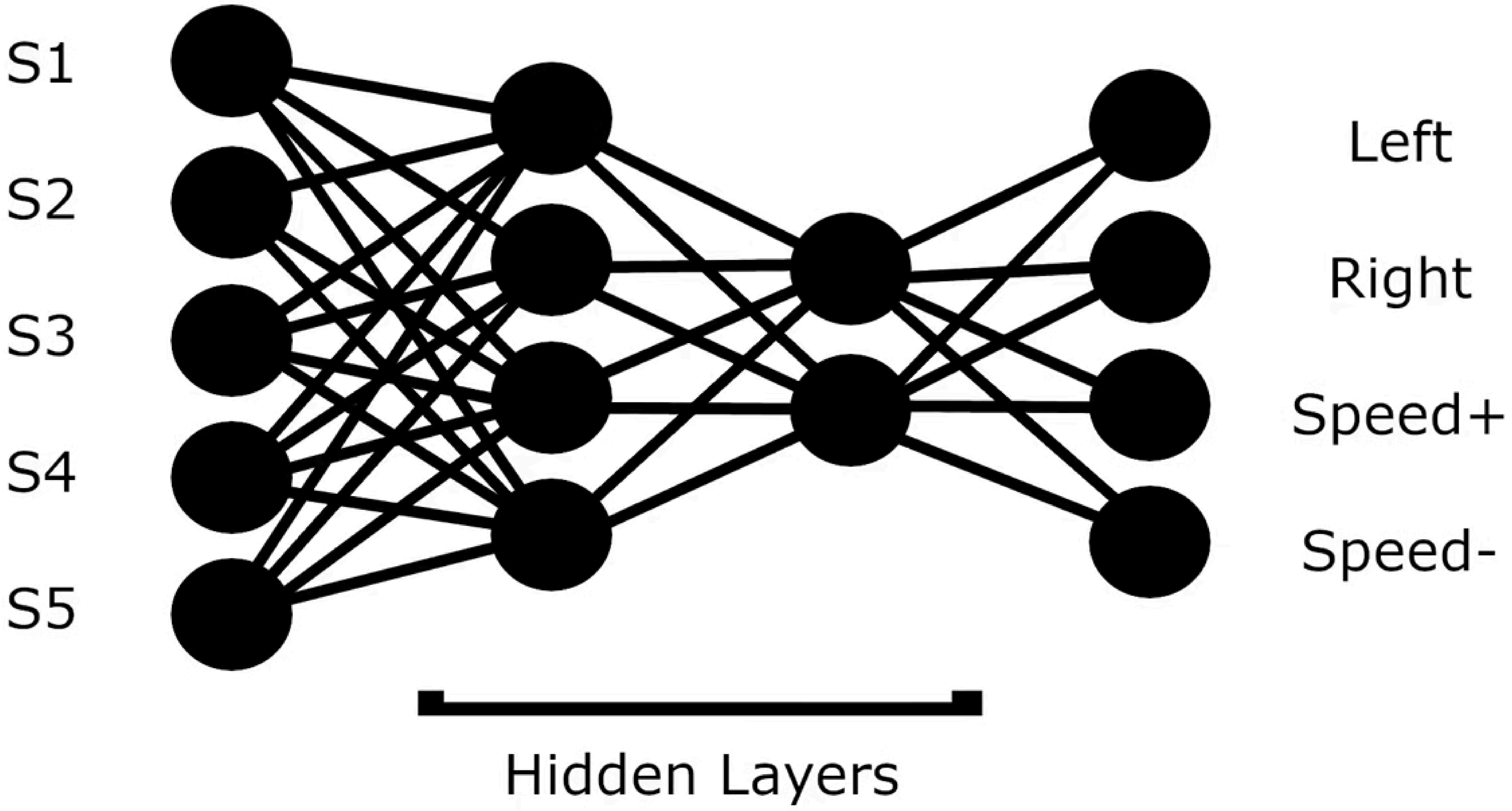
Group 10 (Abdullah Al Afique, Tanha Ahmed, Nijhum, Israt Asha)

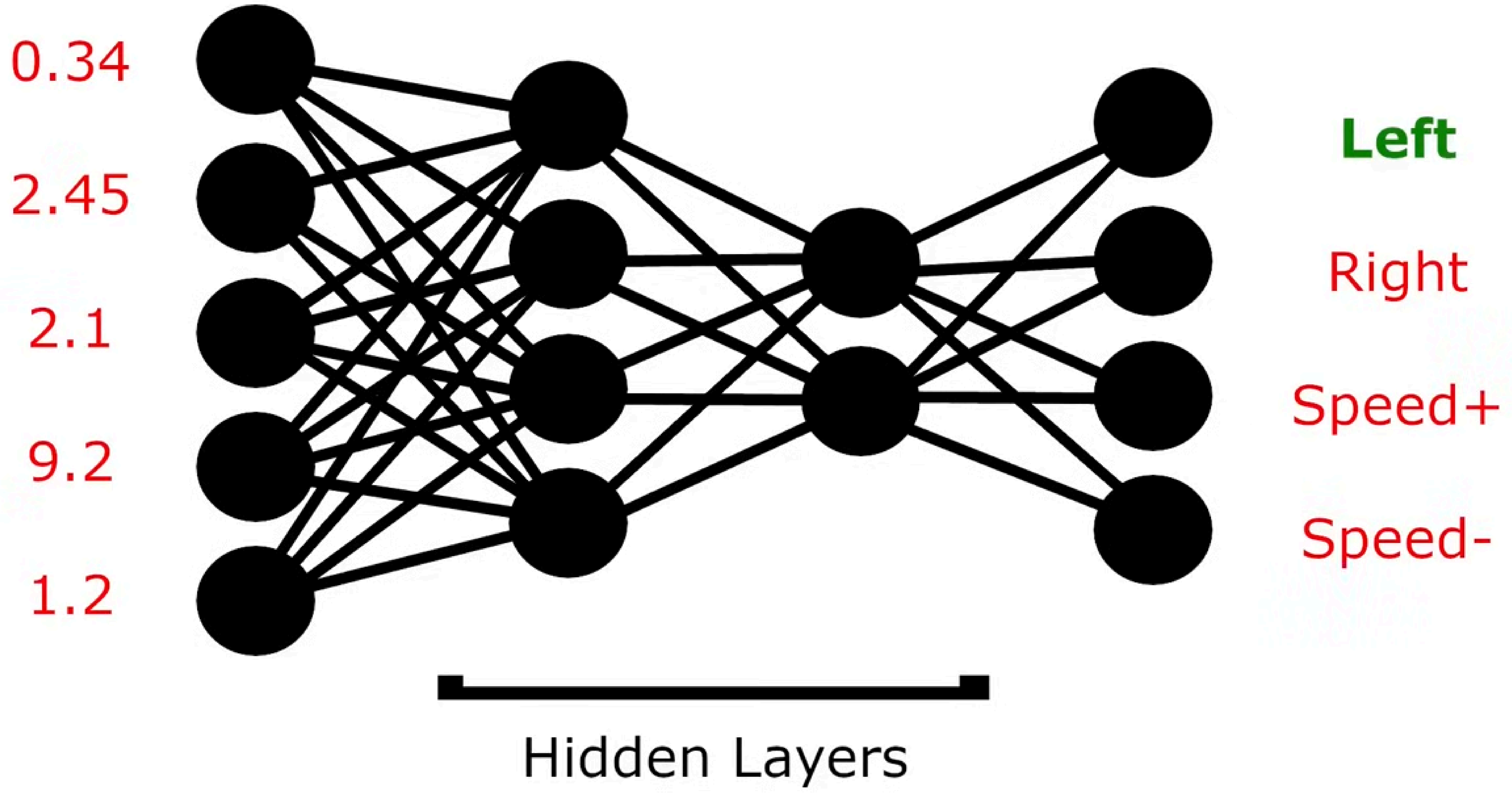

S1
S2
S3
S4
S5


S1
S2
S3
S4
S5

Group 10 (Abdullah Al Afique, Tanha Ahmed, Nijhum, Israt Asha)


S1
S2
S3
S4
S5
Left
Right
Speed+
Speed-
Hidden Layers


S1
S2
S3
S4
S5
Left
Right
Speed+
Speed-
Hidden Layers


0.34
2.45
2.1
9.2
1.2
Left
Right
Speed+
Speed-
Hidden Layers

Let’s watch the AI Learn to Drive

This demo showcases how the AI car improves over generations—from crashing early on to successfully navigating the track with smooth steering, speed control, and collision avoidance. It highlights the power of neural evolution and the effectiveness of learning through interaction with the environment.

https://www.loom.com/share/40853bc7ec8d4da2a355e12201a5ce50?sid=ececffff-3eaf-449c-8866-50ea5f12581d

https://drive.google.com/file/d/11RerMiTTyfYPRrwaJoeut4c0kK51gZ4C/view?usp=sharing

Thank you!

Group 10 (Abdullah Al Afique, Tanha Ahmed, Nijhum, Israt Asha)