

죽림고수에 유전 알고리즘 및 인공 신경망 적용하기

2427 조성민

목차

이 주제를 선택한 계기

이 주제를 선택한 이유를 책의 내용과 크롬 공룡 게임과 연관 지어 설명합니다.

죽림고수 만들기

Javascript로 죽림고수를 만들면서 사용한 스택들과 방법을 소개합니다.

유전 알고리즘 적용하기

유전 알고리즘을 죽림고수에 적용하면서 겪었던 어려움을 소개합니다.

인공 신경망 적용하기

인공 신경망을 적용하면서 겪었던 어려움과 해결과정을 소개합니다.

이 주제를 선택한 계기

이기적 유전자

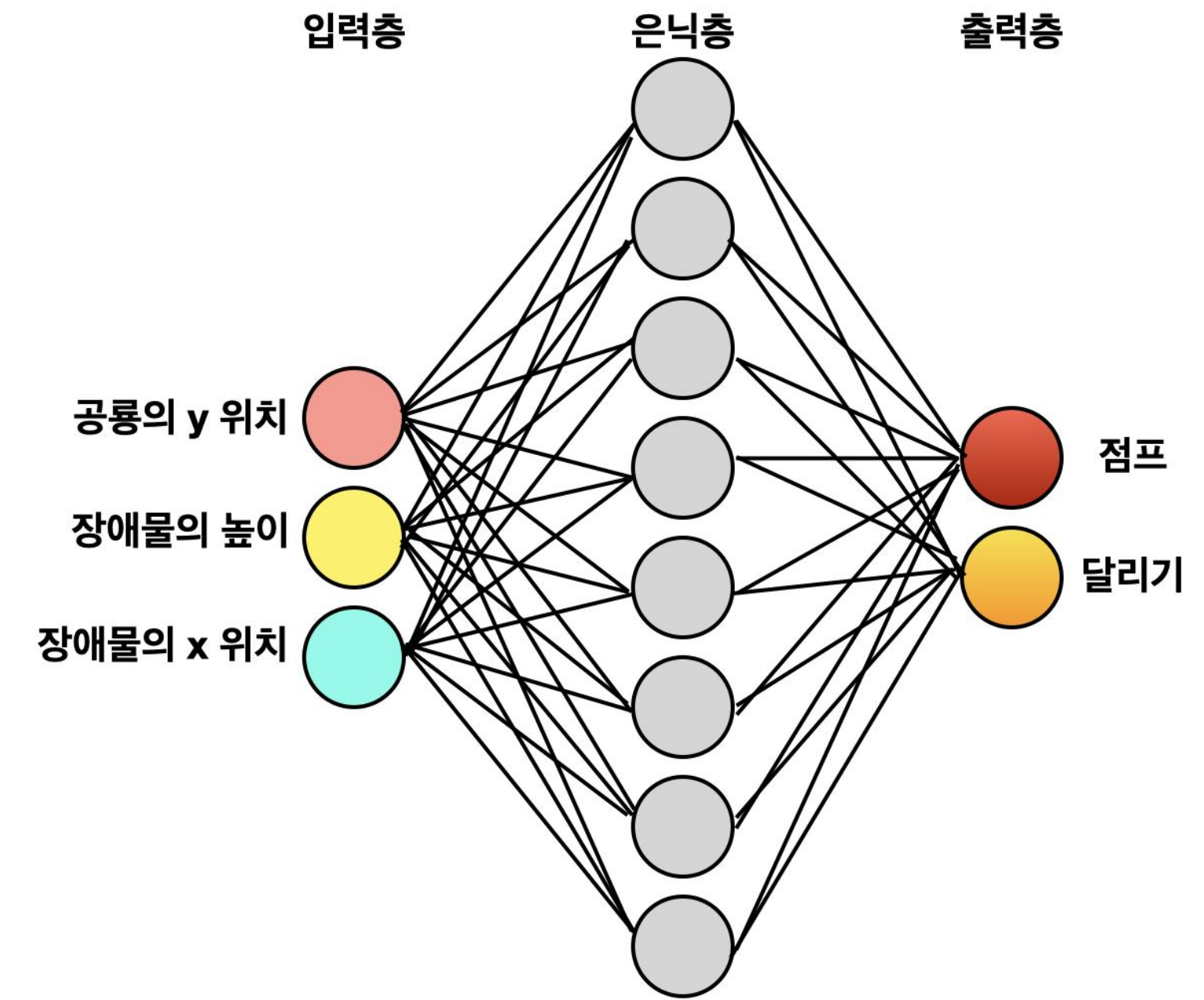
이기적 유전자는 유전자의 특성들을 설명하는데, 유전자의 특성들 중 다음 세대에도 남으려고 하는 과정과 유사한 유전 알고리즘을 사용하여 어떤 문제를 해결해보면 좋겠다고 생각했습니다.

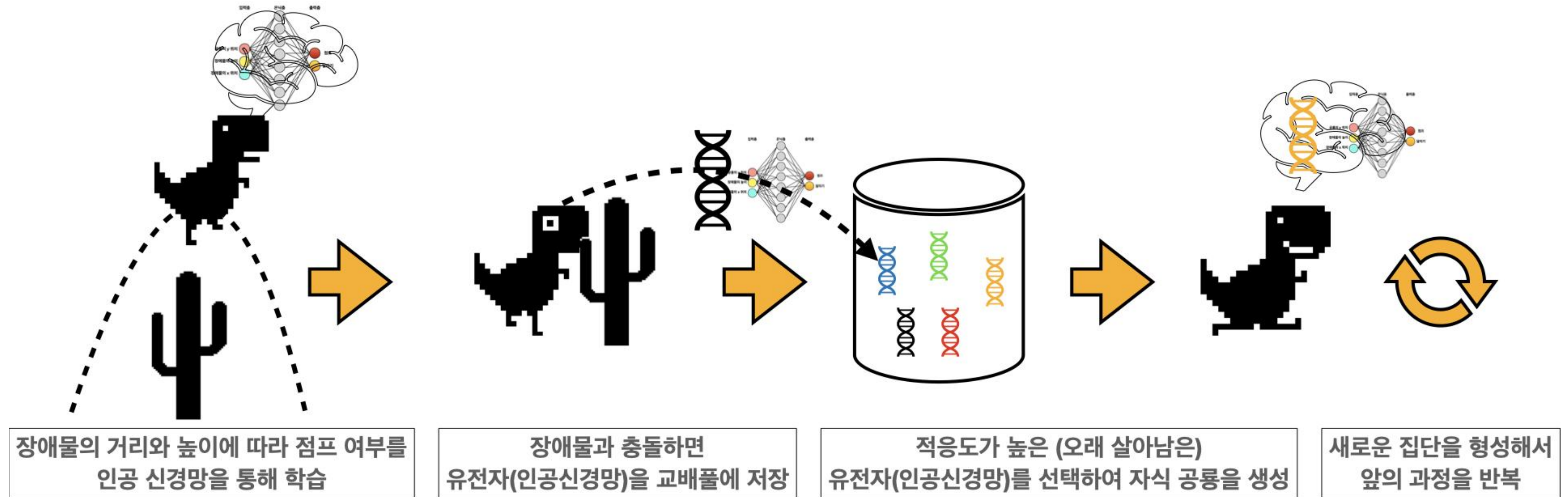
서울대학교 사범대학 부속고등학교 물리 수업

유전 알고리즘을 사용한 문제 해결 예제를 찾던 중 우연히 서울대학교 사범대학 부속고등학교 물리 수업에서 크롬 공룡 게임을 해결하는 예제를 발견했습니다. 이를 보고 석준이가 좋아하는 죽림고수에 적용해보면 재밌겠다고 생각했습니다.

유전 알고리즘을 적용한 공룡 게임

1. 50마리의 공룡으로 이루어진 집단을 만듭니다.
2. 공룡의 유전자는 인공 신경망 객체입니다.
3. 인공 신경망은 다음과 같이 구성됩니다.
4. 각 공룡마다 장애물을 뛰어 넘으면서 인공 신경망을 통해 언제 점프할 것인지를 학습합니다.
5. 장애물과 충돌한 공룡들은 적응도가 낮아서 퇴화되고 오랫동안 살아남은 공룡, 즉 적응도가 높은 공룡은 교배풀에 더 많이 배정되고 선택될 확률이 높아집니다.
6. 교배풀에서 임의의 공룡을 뽑아 인공신경망 객체를 유전자로 자식을 생성합니다. 이때 유전자끼리 교차는 하지 않고 돌연변이만 일정확률로 발생시킵니다.
7. 새로운 집단을 구성하여 이전 세대에서 넘겨 받은 학습된 인공신경망 유전자를 계속 학습시켜 나가면서 진화합니다.





Source: SNU High School Physics Online Classroom

죽림고수 만들기

Webpack 설정

웹 페이지를 빌드할 수 있도록 번들러 중 Webpack을 사용하여 설정합니다.

p5.JS 설정

canvas를 사용하기 위해 p5.JS를 설치하고 sketch()를 통해 기본 함수들을 설정해줍니다.

캐릭터 움직임 구현

고양이 캐릭터가 방향키 입력에 따라 움직일 수 있도록 함수를 추가합니다.

화살 추가

일정 시간마다 랜덤하게 화살이 생성되도록 화살 생성 함수를 추가합니다.

Source: <https://github.com/SID12g/bamboo-master>

Post: <https://post.sid12g.dev/tech/posts/bamboo-master>

유전 알고리즘 적용하기

초기 개체 생성

우선 무작위로 초기 개체를 생성합니다. 이 개체들은 진화 과정에서 유전자에 해당합니다.

선택

다음 세대 개체를 만들기 위해 적합도 평가가 우수한 개체들을 선택합니다.

적합도 평가

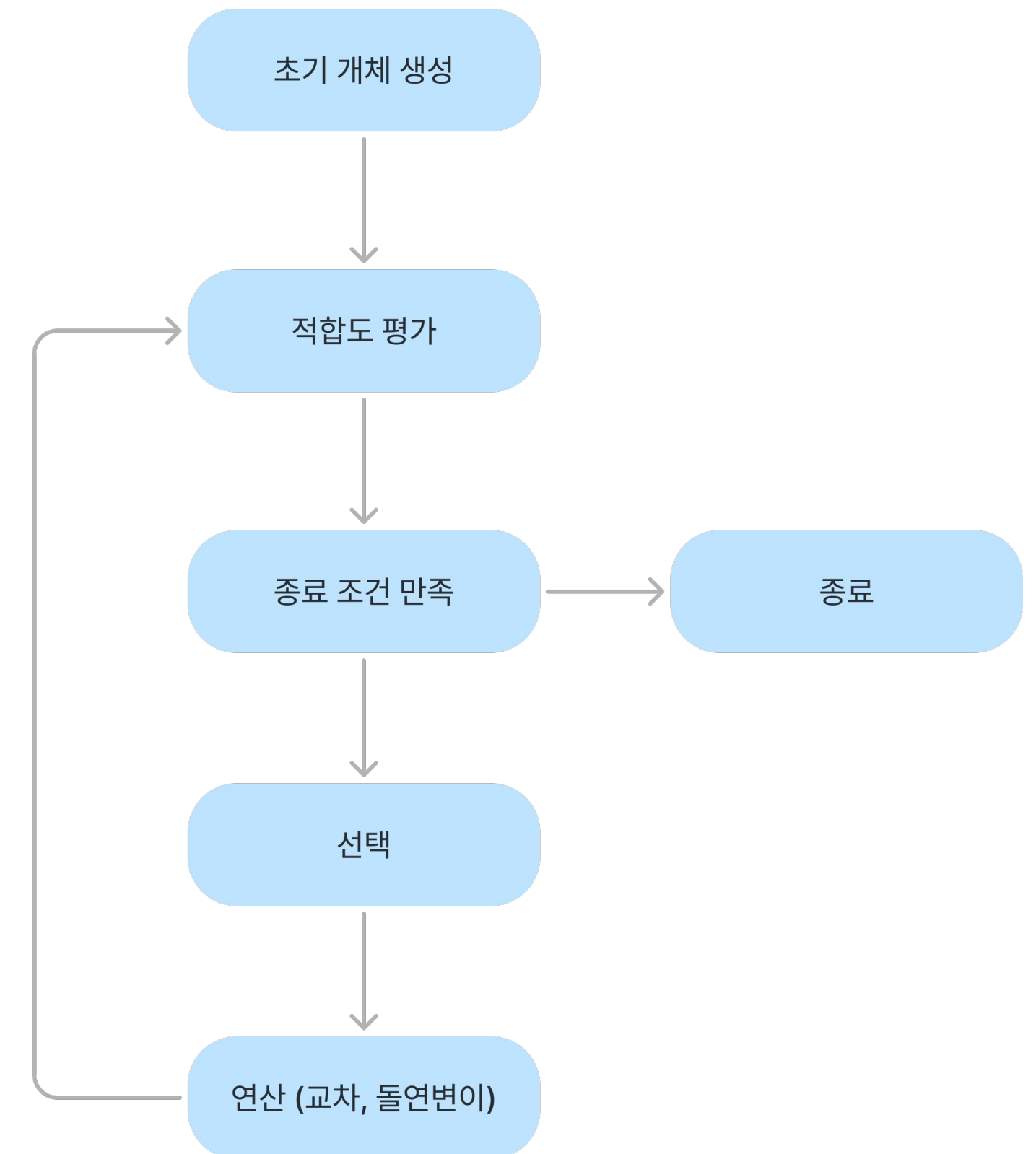
현재 세대의 개체를 보통 적합도 함수를 사용하여 평가합니다.

연산

선택된 개체들을 교차 연산하여 다음 세대의 개체를 만듭니다. (부모가 자식을 낳는 과정과 유사합니다.)

종료 조건 검사

현재 세대의 개체가 종료 조건을 만족하는지 확인합니다. 이를 만족하는 경우 알고리즘을 종료하고, 만족하지 않는 경우 선택 단계로 이동합니다.



Source: <https://github.com/SID12g/genetic-bamboo-master>

Post: <https://post.sid12g.dev/tech/posts/genetic-network>

유전 알고리즘 적용하기

```
const crossover = (brain1, brain2) => {
  const newBrain = setupNeuralNetwork();
  newBrain.layers.hidden.forEach((layer, i) => {
    layer.list.forEach((neuron, j) => {
      Object.values(neuron.connections.projected).forEach((connection, k) => {
        const parentConnection =
          Math.random() > 0.5
            ? Object.values(
                brain1.layers.hidden[i].list[j].connections.projected
              )[k]
            : Object.values(
                brain2.layers.hidden[i].list[j].connections.projected
              )[k];
        connection.weight = parentConnection.weight;
      });
    });
  });

  return newBrain;
};
```

crossover()는 상위 2개 개체 (고양이 캐릭터) 중 랜덤하게 하나를 골라 상속하는 함수입니다.

유전 알고리즘 적용하기

```
const mutate = (network) => {  
  network.layers.hidden.forEach((layer) => {  
    layer.list.forEach((neuron) => {  
      Object.values(neuron.connections.projected).forEach((connection) => {  
        if (Math.random() < 0.1) {  
          connection.weight += Math.random() * 0.2 - 0.1;  
        }  
      });  
    });  
  });  
};
```

그리고 mutate()를 통해 돌연변이를 생성해줍니다. 돌연변이는 weight 값에 영향을 줍니다.

유전 알고리즘 적용하기

```
const generateCats = (brain1, brain2) => {  
  generation++;  
  cats = [];  
  
  for (let i = 0; i < 10; i++) {  
    let newBrain;  
    if (brain1 && brain2) {  
      newBrain = crossover(brain1, brain2);  
      mutate(newBrain);  
    } else {  
      newBrain = setupNeuralNetwork();  
    }  
  
    const newCat = new Cat(281, 281, i);  
    newCat.brain = newBrain;  
    cats.push(newCat);  
  }  
};
```

다음으로 위 함수들을 기반으로 10마리 모두 죽었을 때 새로운 집단을 생성합니다.

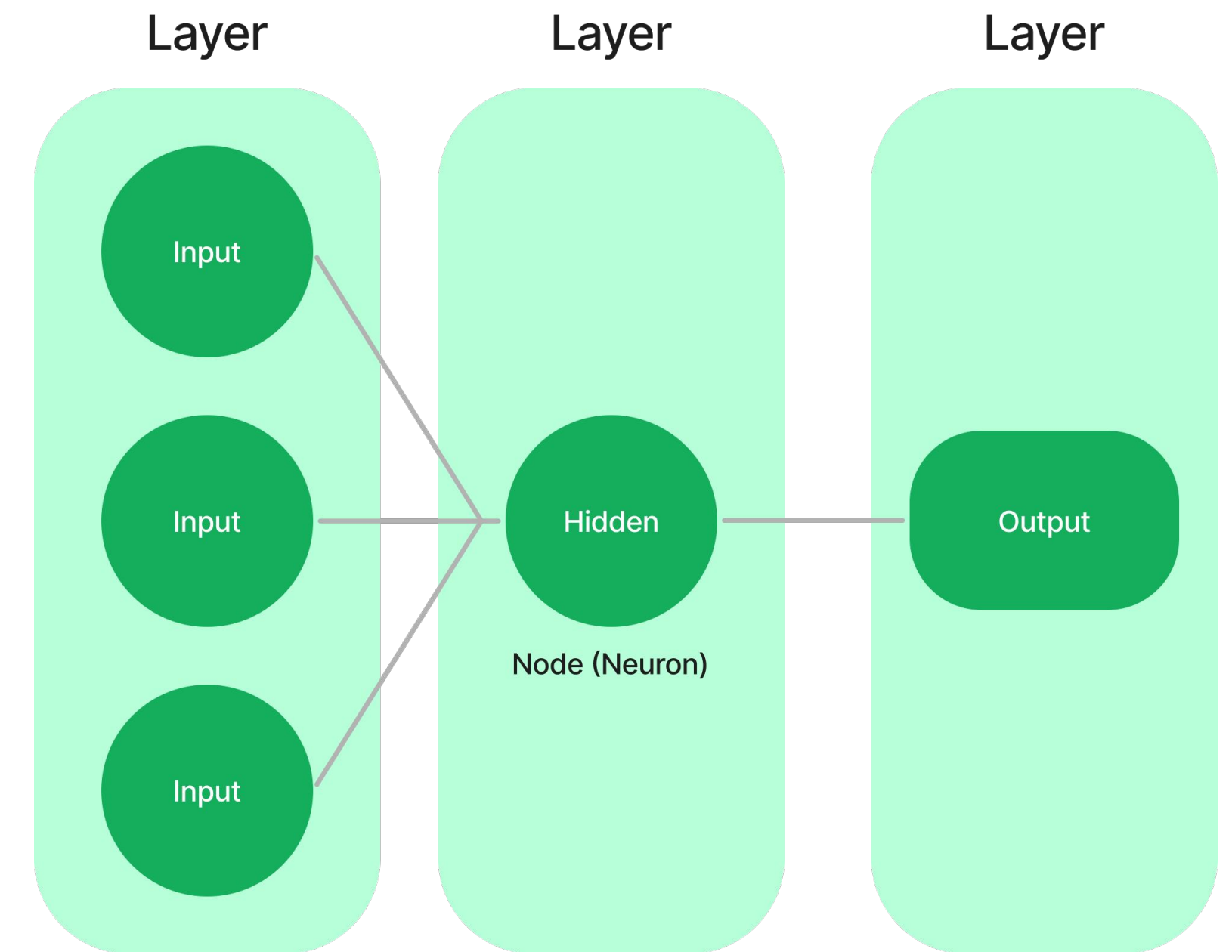
인공 신경망 적용하기

노드 (Neuron)

인공신경망의 기본 단위로
각 노드는 입력을 받아 처리
하고, 출력값을 생성합니다.
노드는 생물학적 신경세포와
유사하게 작동합니다.

층 (Layer)

- 입력층 (Input Layer): 외부 데이터 (인풋 값)를 받아들이는 층으로 입력값의 수는 이 층의 노드 수에 따라 결정됩니다.
- 은닉층 (Hidden Layer): 입력층과 출력층 사이에 위치한 층으로, 데이터의 복잡한 패턴을 학습하는 역할을 합니다. 여러 개의 은닉층을 사용할 수 있으며, 이는 신경망의 깊이를 결정합니다.
- 출력층 (Output Layer): 신경망의 최종 결과를 출력하는 층으로 예측하려는 결과의 수에 따라 노드 수가 결정됩니다.

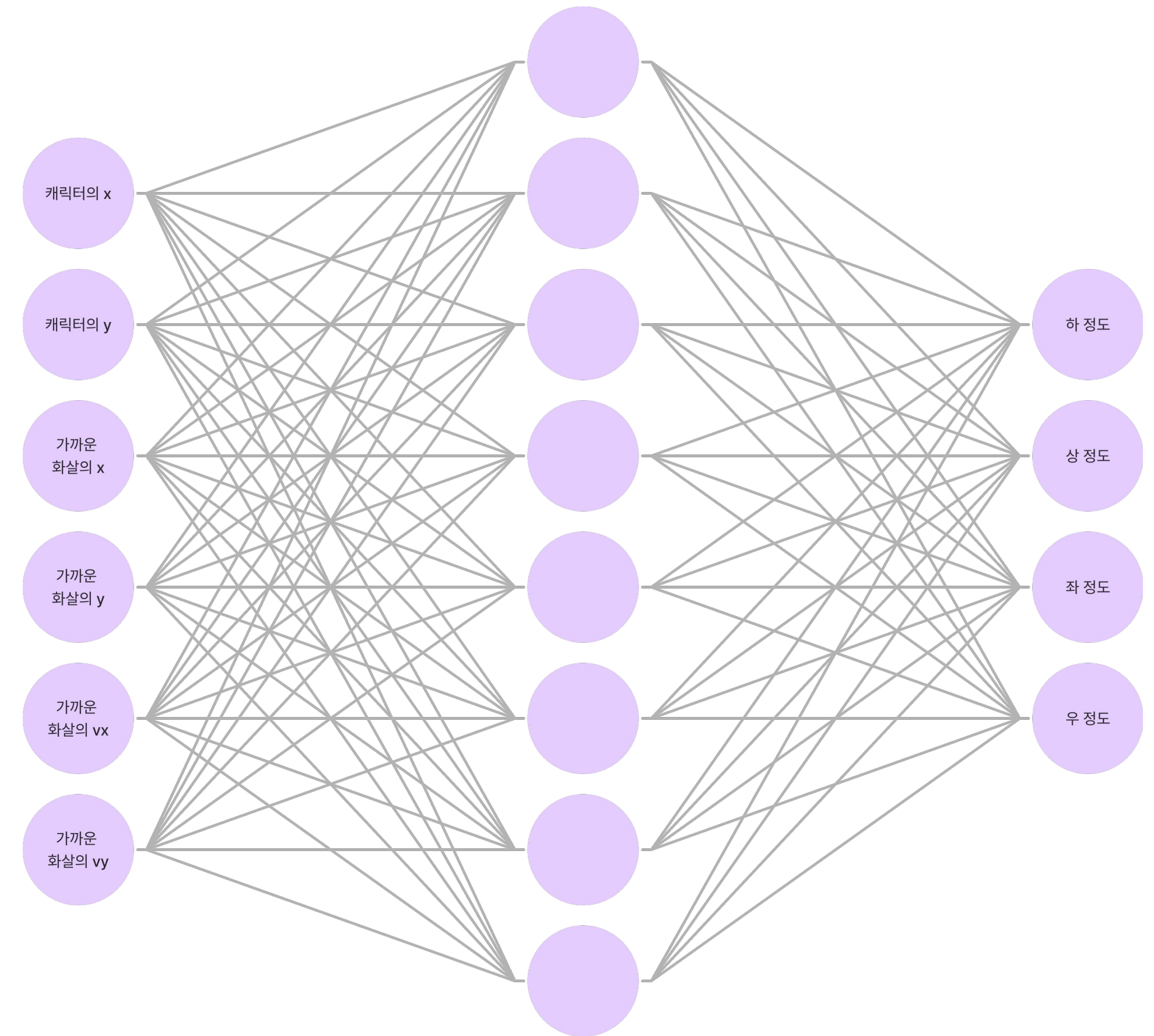


Source: <https://github.com/SID12g/genetic-bamboo-master>

Post: <https://post.sid12g.dev/tech/posts/genetic-network>

인공 신경망 적용하기

아래는 죽림고수에 적용할 인공 신경망입니다. Input Layer에는 캐릭터의 위치 (x, y), 화살의 위치(x, y), 화살의 속도 (vx, vy)로 이루어져 8개의 Hidden Layer를 거쳐 움직임에 해당하는 Output Layer가 출력됩니다.



Source: <https://github.com/SID12g/genetic-bamboo-master>

Post: <https://post.sid12g.dev/tech/posts/genetic-network>

인공 신경망 적용하기

```
const setupNeuralNetwork = () => {  
  const inputLayer = new Layer(6);  
  const hiddenLayer = new Layer(8);  
  const outputLayer = new Layer(4);  
  
  inputLayer.project(hiddenLayer);  
  
  hiddenLayer.project(outputLayer);  
  
  const network = new Network({  
    input: inputLayer,  
    hidden: [hiddenLayer],  
    output: outputLayer,  
  });  
  
  return network;  
};
```

Source: <https://github.com/SID12g/genetic-bamboo-master>

Post: <https://post.sid12g.dev/tech/posts/genetic-network>

위 사진과 같은 구조를 synaptic을 통해 적용합니다.

결과



Source: <https://www.youtube.com/watch?v=aMuCE-CbLeQ>