NÖROEVRİM ALGORİTMASIYLA OYUN OYNAYABİLEN YAPAY ZEKA GELİŞTİRME

BİTİRME PROJESİ 2

Furkan Kaya

191216002

DANIŞMAN

Öğr. Gör. Ezgi Özer

5 Mayıs 2021

Özet

Bu raporda, genetik algoritması kullanılarak oluşturulan yapay zeka kütüphanesi anlatılmıştır ve bu kütüphane ile geliştirilen projeler tanıtılmıştır. Bunlar;

- Perceptron'a doğrusal denklem öğretmek
- Neural Network ile XOR geliştirmek
- Flappy Bird oyununa yapay zeka eklemek
- Doodle Jump oyununa yapay zeka eklemek
- Circle oyununa yapay zeka eklemek

Geliştirilen yapay zeka kütüphanesinin ve örnek uygulamalarının kaynak kodlarına https://github.com/Wijt/graduation-project-2 Linki üzerinden erişilebilmektedir.

Neural Network Kütüphanesi

Bir önceki raporda anlatılan Circle, Flappy Bird ve Doodle Jump oyunlarına entegre edilebilecek bir yapay zeka kütüphanesi geliştirilmiştir.

Perceptron Sınıfı:

Perceptron sınıfı aşağıdaki birimlerden oluşmaktadır.

```
13 başvuru
public class Perceptron
{
    public float[] weights;
    float bias;

    System.Func<float, float> activationFunc;

    2 başvuru
    public Perceptron(int inputCount)...

    1 başvuru
    public Perceptron(int inputCount, System.Func<float, float> activationFunc)...

    3 başvuru
    public float Fire(float[] inputs)...

    1 başvuru
    public Perceptron Copy()...

    1 başvuru
    public void Mutate(float mutateRate)...

    1 başvuru
    public void Train(float[] inputs, float[] outputs)...
}
```

Neural Network Sınıfı:

Neural network sınıfı aşağıdaki resimde görülen birimlerden oluşmuştur.

```
7 başvuru
public class NeuralNetwork
{
    int[] sizes;
    Perceptron[][] perceptrons;

    2 başvuru
    public NeuralNetwork(int[] sizes)...

    6 başvuru
    public float[] FeedForward(float[] inputs)...

    1 başvuru
    public NeuralNetwork Copy()...

    1 başvuru
    public void Mutate()...
}
```

SmartObject Sınıfı:

Bu sınıf, kütüphaneyi kullananların kendi ödül/ceza algoritmalarını geliştirebileceği sınıftır. Birimlerini yandaki görselde görebilirsiniz.

```
Unity Betiği | 13 başvuru
public abstract class SmartObject : MonoBehaviour
{
    public NeuralNetwork brain;
    public float fitness = 0;

    public bool isActive = true;
}
```

Evolution Manager Sınıfı:

En iyi yapay zekayı elde edene kadar SmartObject'leri üretip, üretilen SmartObject'lerin hepsi ölünce ortama en iyi adaptasyonu sağlayanın seçilerek yeni jenerasyonların üretildiği sınıftır. İçerdiği birimleri aşağıdaki görselde görebilirsiniz.

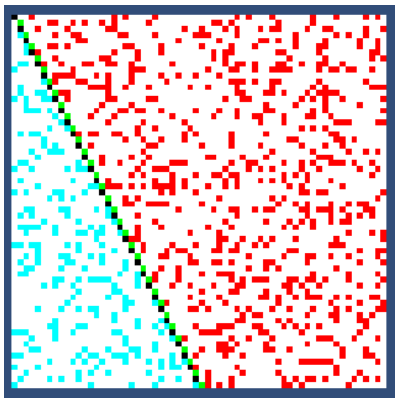
```
Unity Betiği | 1 başvuru
public class EvolutionManager : MonoBehaviour
{
    public string id;
    public Transform spawnPoint;
    public GameObject smartObject;
    public int[] networkSize;

    [SerializeField]
    public List<SmartObject> population;

    public int totalPopulationSize;
    [Range(0, 100)]
    public int newBornRatio;

    public GameObject bestObject;

    public UnityEvent ResetFunctions;
    public UnityEvent LeaveBestFunctions;

    bool shouldReset = false;

    public bool debug;

    string dataPath;

    Unity İletisi | 0 başvuru
    private void Start()...
    Unity İletisi | 0 başvuru
    private void Update()...
    0 başvuru
    public void LeaveBest()...
    1 başvuru
    public List<SmartObject> GetPopulation(bool isDeath)...
    2 başvuru
    public void CreatePopulation(NeuralNetwork brain)...
    2 başvuru
    public NeuralNetwork GetFittestObjectBrain()...
    2 başvuru
    public SmartObject GetFittestObject()...
    1 başvuru
    public GameObject GetFittestGameObject()...
    0 başvuru
    public void CheckFittestandTempSave()...
}
```

Perceptron Geliştirilmesi:

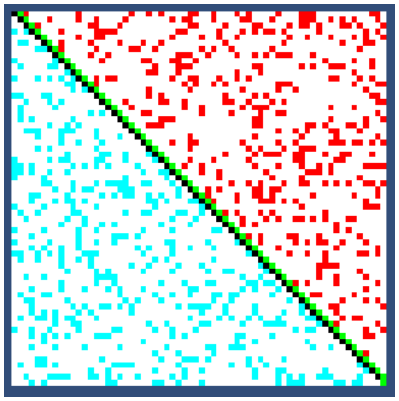
Perceptron Trainer Sınıfı:

Kütüphanenin doğru çalıştığını test edebilmek için ilk olarak bu proje geliştirilmiştir. Tek bir perceptron eğitilerek doğrusal denklemi anlayabilmesi sağlanmıştır ve doğrusal düzlemdeki rastgele noktaları sınıflandırması istenmiştir. Sonucu ve kodları aşağıdaki görsellerde görebilirsiniz.

```
public class P_Trainer : MonoBehaviour
{
    Texture2D plane;
    public Perceptron p;
    public int dotCount = 250;
    1 başvuru
    public float CalculateY_AI(float x) { return (-p.weights[2] - p.weights[0] * x) / p.weights[1]; }
    2 başvuru
    public float CalculateY(float x) { return 2 * x; }
    // Start is called before the first frame update
    Unity İletisi | 0 başvuru
    void Start()
    {

        p = new Perceptron(2, Constants.Sign);

        plane = Resources.Load<Texture2D>("Plane-64");
        Sprite s = GetComponent<SpriteRenderer>().sprite;
        s = Sprite.Create(plane, new Rect(0, 0, 64, 64), Vector2.one * 0.5f);

        Train();
    }
    1 başvuru
    public void Train()
    {
        //create random dots
        Vector2[] dots = new Vector2[dotCount];
        for (int i = 0; i < dotCount; i++)
        {
            dots[i] = new Vector2(Random.Range(0,plane.width), Random.Range(0, plane.height));
        }
        ClearPlane(Color.white);

        for (int i = 0; i < 1000; i++)
        {
            foreach (Vector2 dot in dots)
                p.Train(new float[] { dot.x, dot.y }, new float[] { dot.y < CalculateY(dot.x) ? 1 : -1 });
        }

        foreach (Vector2 dot in dots)
        {
            float answer = p.Fire(new float[] { dot.x, dot.y }); /*   dot.y < CalculateY(dot.x) ? 1 : -1;*/
            plane.SetPixel((int)dot.x, (int)dot.y, answer == 1 ? Color.red : Color.cyan);
        }
        DrawLine(CalculateY, plane, Color.black);
        DrawLine(CalculateY_AI, plane, Color.green);
        plane.Apply();
    }
```



y=2*x grafiği



y=x grafiği

XOR Geliştirilmesi:

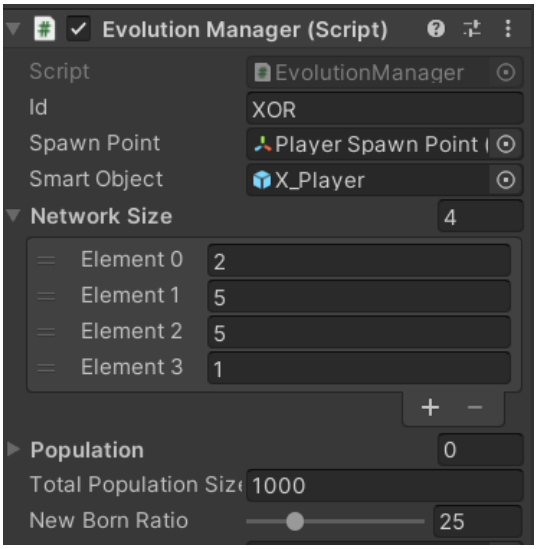
```
public class X_Player : SmartObject
{
    public int maxThinkCount = 100;

    int[,] truthTable;

    public int rowID;

    public float output;
    public bool calculate = false;

    // Start is called before the first frame update
    Unity İletisi | 0 başvuru
    void Start()
    {
        truthTable = new int[,]
        {
            {0,0,0},
            {0,1,1},
            {1,0,1},
            {1,1,0}
        };
        isActive = true;
        fitness = 0;
        for (int i = 0; i < maxThinkCount; i++)
        {
            int truthRowID = Random.Range(0, truthTable.GetLength(0));
            float[] inputs = new float[2];
            inputs[0] = truthTable[truthRowID, 0];
            inputs[1] = truthTable[truthRowID, 1];
            float[] outputs = brain.FeedForward(inputs);
            float diffToTruth = Mathf.Abs(outputs[0] - truthTable[truthRowID, 2]);
            if (diffToTruth < 0.50f)
            {
                fitness += 100;
            }
            else
            {
                fitness -= 100;
            }
            //break;
        }
        isActive = false;
    }
```

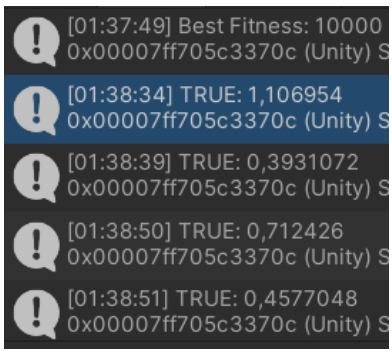
X_Player Sınıfı:

SmartObject sınıfından türetilen bu sınıf her bir ayrı yapay zeka beynini temsil etmektedir. Yapay zekalar doğdukları anda belirlenen düşünme sayısı kadar düşünür ve sonucunda fitlik değeri elde ederler.

Evolution Manager Parametreleri:

XOR çözebilen yapay zeka için ayarlanan parametreler sağdaki görselde görülebilir. Eğitimin sonuçları ise aşağıda bulunmaktadır. Ortalama 4 jenerasyonda mükkemel derecede XOR çözebilen yapay zeka üretilebilmektedir.

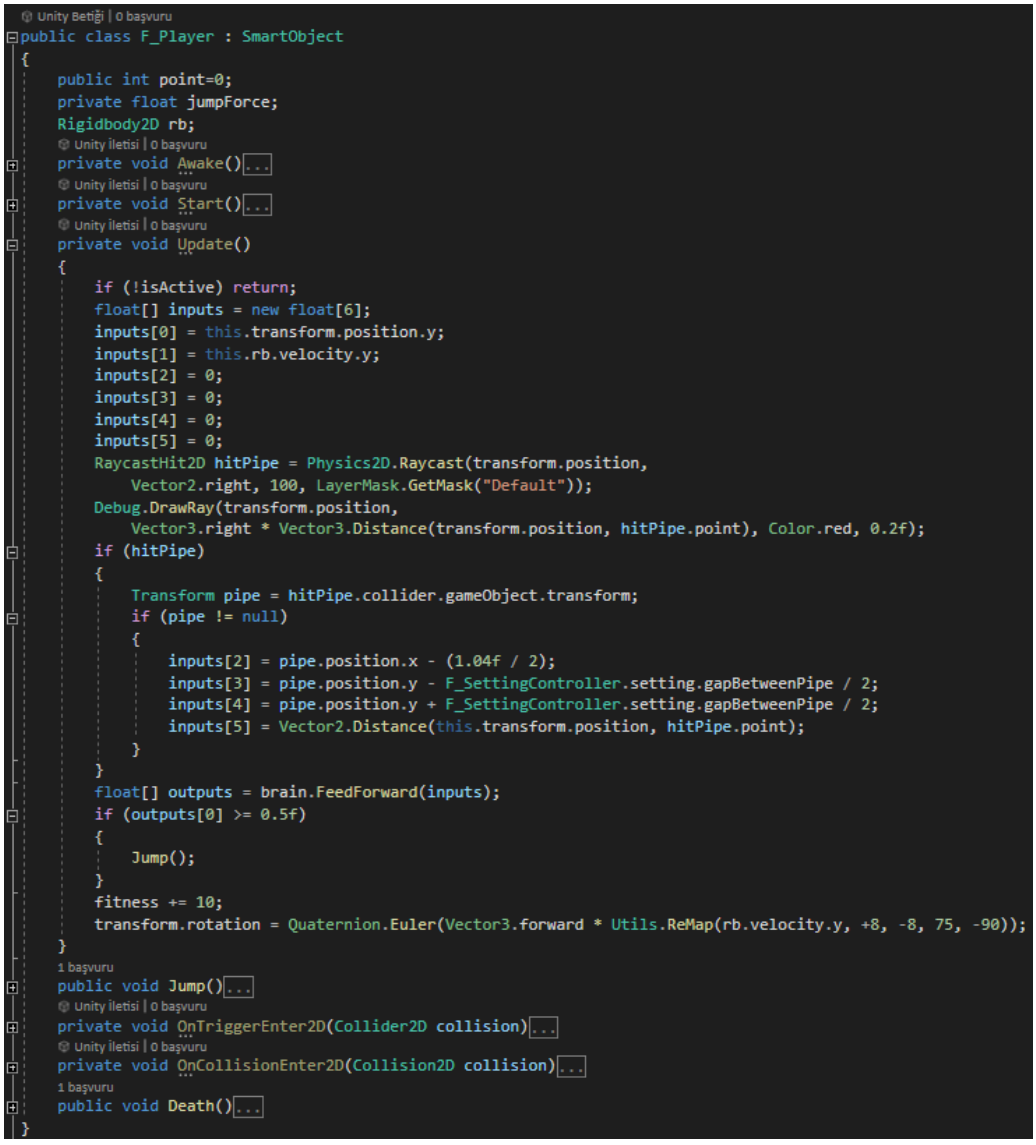




Flappy Bird Geliştirilmesi:

F_Player Sınıfı:

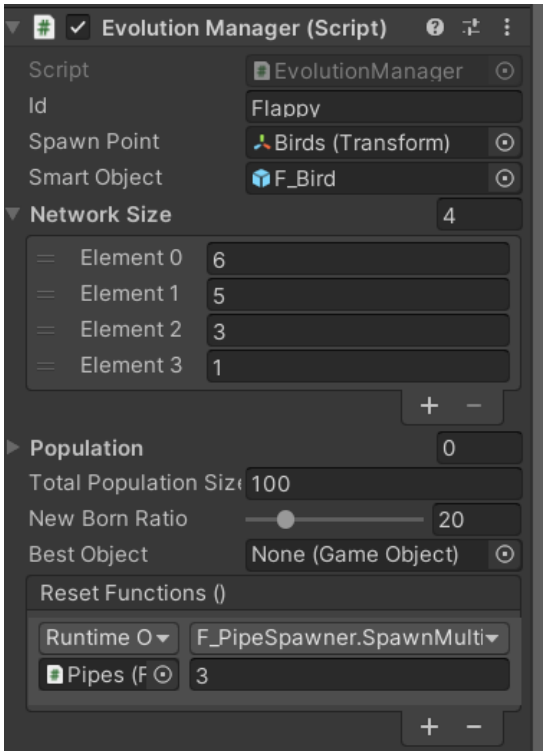
Bu sınıf SmartObject sınıfından türetilmiştir ve çevreden aldığı verileri beyne iletip gelen sonuca göre kuşu zıplatmaktadır. Ayrıca kuşun yaşadığı süre boyunca fitlik değeri artmaktadır.

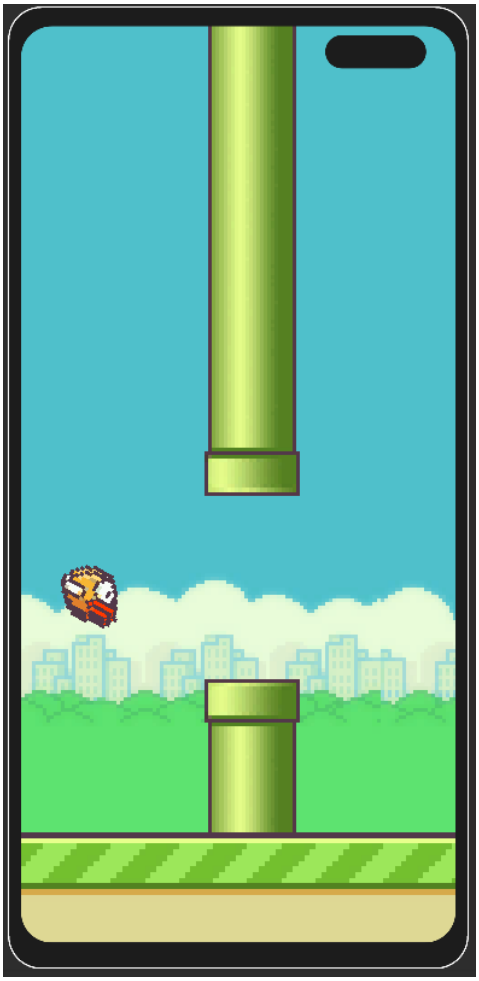


```
Unity Betiği | 0 başvuru
public class F_Player : SmartObject
{
    public int point=0;
    private float jumpForce;
    Rigidbody2D rb;
    Unity iletisi | 0 başvuru
    private void Awake()...
    Unity iletisi | 0 başvuru
    private void Start()...
    Unity iletisi | 0 başvuru
    private void Update()
    {
        if (!isActive) return;
        float[] inputs = new float[6];
        inputs[0] = this.transform.position.y;
        inputs[1] = this.rb.velocity.y;
        inputs[2] = 0;
        inputs[3] = 0;
        inputs[4] = 0;
        inputs[5] = 0;
        RaycastHit2D hitPipe = Physics2D.Raycast(transform.position,
            Vector2.right, 100, LayerMask.GetMask("Default"));
        Debug.DrawRay(transform.position,
            Vector3.right * Vector3.Distance(transform.position, hitPipe.point), Color.red, 0.2f);
        if (hitPipe)
        {
            Transform pipe = hitPipe.collider.gameObject.transform;
            if (pipe != null)
            {
                inputs[2] = pipe.position.x - (1.04f / 2);
                inputs[3] = pipe.position.y - F_SettingController.setting.gapBetweenPipe / 2;
                inputs[4] = pipe.position.y + F_SettingController.setting.gapBetweenPipe / 2;
                inputs[5] = Vector2.Distance(this.transform.position, hitPipe.point);
            }
        }
        float[] outputs = brain.FeedForward(inputs);
        if (outputs[0] >= 0.5f)
        {
            Jump();
        }
        fitness += 10;
        transform.rotation = Quaternion.Euler(Vector3.forward * Utils.ReMap(rb.velocity.y, +8, -8, 75, -90));
    }
    1 başvuru
    public void Jump()...
    Unity iletisi | 0 başvuru
    private void OnTriggerEnter2D(Collider2D collision)...
    Unity iletisi | 0 başvuru
    private void OnCollisionEnter2D(Collision2D collision)...
    1 başvuru
    public void Death()...
}
```

Evolution Manager Parametreleri:

Her kuş 6-5-3-1 katman büyüklüklerine sahip bir Nöral ağa sahiptir. Aralarından en iyileri seçilip New Born Ratio'ya göre tamamen rastgele nöral ağlar da oluşturulup yeni jenerasyonlar oluşturulmaktadır.





Sonuç:

Ortalama 8 jenerasyonda Flappy Bird oynayabilen yapay zeka ortaya çıkartılabilmiştir.

Doodle Jump Geliştirilmesi:

D_Player Sınıfı:

Bu sınıf SmartObject sınıfından türetilmiştir ve çevreden aldığı verileri beyne iletip gelen sonuca göre oyuncuyu sağa yada sola ilerletmektedir. Ayrıca oyuncu yukarı doğru çıktığı sürece fitlik değeri artmaktadır.

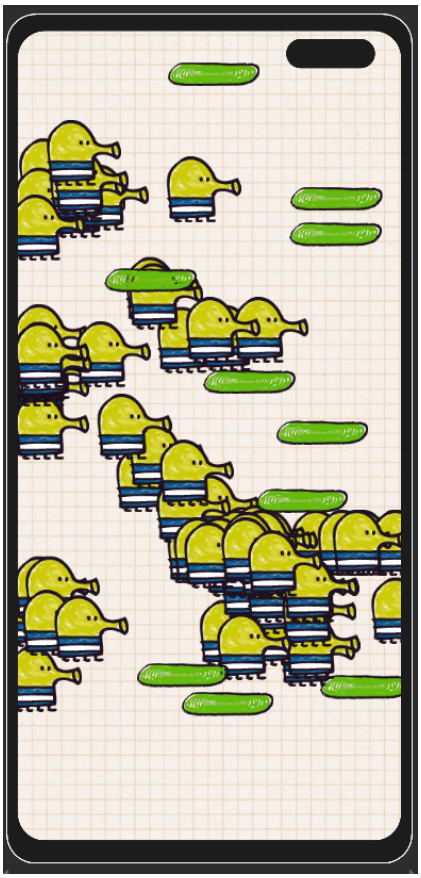
```
void Update() {
    if (!isActive) return;

    inputs = new float[15];
    inputs[0] = this.transform.position.x;
    inputs[1] = this.transform.position.y;
    inputs[2] = this.rb.velocity.y;

    RaycastHit2D hitUpCenter = Physics2D.Raycast(transform.position,
        Vector2.up, 10, LayerMask.GetMask("Default"));
    if (hitUpCenter)...

    RaycastHit2D hitUpRight = Physics2D.Raycast(transform.position,
        new Vector2(1, 1), 10, LayerMask.GetMask("Default"));
    if (hitUpRight)...

    RaycastHit2D hitUpLeft = Physics2D.Raycast(transform.position,
        new Vector2(-1, 1), 10, LayerMask.GetMask("Default"));
    if (hitUpLeft)...

    RaycastHit2D hitCenterDown = Physics2D.Raycast(transform.position,
        new Vector2(0, -1), 10, LayerMask.GetMask("Default"));
    if (hitCenterDown)...

    float output = brain.FeedForward(inputs)[0];
    output = Mathf.Min(output, 1);
    Vector3 newPos = Vector3.Lerp(transform.position,
        new Vector3(Utils.ReMap(output, 0, 1, -2.3f, 2.3f), transform.position.y, 0),
        Time.deltaTime * 10f);

    this.transform.position = newPos;

    if (Camera.main.transform.position.y < transform.position.y) {
        point += 1;
        fitness += 10;
    }
}
```

Evolution Manager Parametreleri:

Her Doodle 15-10-10-5-1 katman büyüklüklerine sahip bir Nöral ağa sahiptir. Aralarından en iyileri seçilip New Born Ratio'ya göre tamamen rastgele nöral ağlar da oluşturulup yeni jenerasyonlar oluşturulmaktadır.

Sonuç:

Ortalama 5 jenerasyonda Doodle Jump oynayabilen yapay zeka ortaya çıkartılabilmiştir.



Circle Geliştirilmesi:

C_Player Sınıfı:

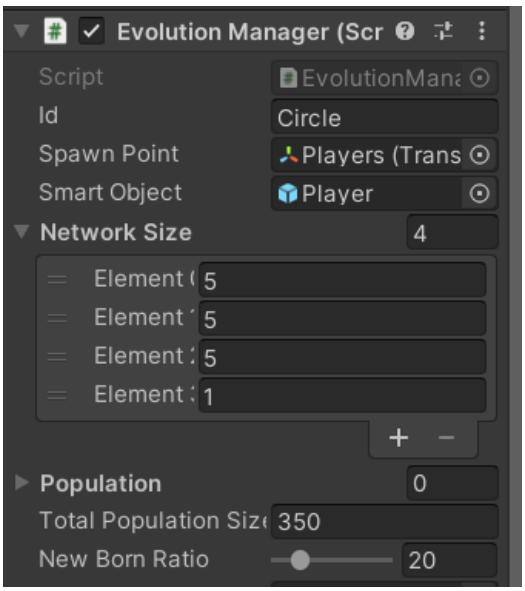
Bu sınıf circle oyuncularının beyinlerine verilerin iletilmesi ve beynin çıktısının işlenmesinden sorumludur. 5 farklı veri beyne iletilmektedir. Bunlar;

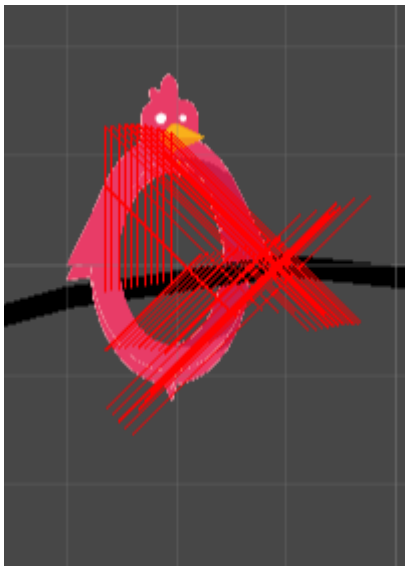
- Oyuncu y pozisyonu
- Oyuncunun y yönündeki hızı
- Oyuncunun üst bölgesinin yola olan uzaklığı
- Alttan ve üstten 45 derece açıyla yola doğru uzanan çizgilerin uzunluğu

```
void Update() {
    if (!isActive) return;
    inputs = new float[5];
    inputs[0] = this.transform.position.y;
    inputs[1] = this.rb.velocity.y;

    Vector3 originTop = transform.position + new Vector3(0, 0.9f, 0);
    Vector3 originBottom = transform.position + new Vector3(0, -1.14f, 0);

    RaycastHit2D hitCenter = Physics2D.Raycast(originTop, Vector2.down, 10, LayerMask.GetMask("Default"));
    Debug.DrawRay(originTop, Vector3.down * Vector3.Distance(originTop, hitCenter.point), Color.red, 0.2f);
    inputs[2] = Vector3.Distance(originTop, hitCenter.point);

    RaycastHit2D hitBottom = Physics2D.Raycast(originBottom, new Vector2(1, 1), 10, LayerMask.GetMask("Default"));

    Debug.DrawRay(originBottom, new Vector2(1, 1) * Vector2.Distance(originBottom, hitBottom.point), Color.red, 0.2f);
    inputs[3] = Vector3.Distance(originBottom, hitBottom.point);

    RaycastHit2D hitTop = Physics2D.Raycast(originTop, new Vector2(1, -1), 10, LayerMask.GetMask("Default"));

    Debug.DrawRay(originTop, (new Vector2(1, -1) * Vector3.Distance(originTop, hitTop.point)), Color.red, 0.2f);
    inputs[4] = Vector3.Distance(originTop, hitTop.point);

    float output = brain.FeedForward(inputs)[0];
    if (output > 0.5f) {
        Jump();
    }
    rb.AddRelativeForce(Vector2.right * speed - rb.velocity);
}
```

Evolution Manager Parametreleri:

Circle oyunundaki her bir oyuncu 4 katmanlı, 16 perceptronlu bir beyne sahiptir ve her jenerasyonda toplam 350 adet oyuncu bulunmaktadır. Bunların %20'si tamamen rastgele bir şekilde oluşmaktadır.





Sonuç:

Ortalama 3 jenerasyonda rastgele oluşan yolu baştan sona bitirebilen yapay zeka elde edilebiliyor.