Yapay Zeka 2021-2022
Bahar Dönemi
Dönem Projesi

Video Link

Ömer Buğrahan Çalışkan - 17011076

Ömer Aras Kaplan - 17011039

Konu: Genetik algoritma kullanarak programa Flappy Bird oyununun öğretilmesi.

Projede Neuro Evolution of Augmenting Topologies(NEAT) yaklaşımını tercih ettik. Bu yaklaşım, yapay sinir ağları ve genetik algoritmaların bir füzyonu olarak düşünülebilir. Çeşitli giriş değerleri, belli fonksiyonlar içerisinde puanlanıp, genetik olarak elde edilen ağırlıklarına göre bir puan alır. Elde edilen puana göre bir karar verilir. Algoritma çeşitli hiperparametrelere sahiptir. Mutasyon oranı, popülasyon sayısı, karar eşiği, önyargı(bias) değerleri, iki pipe arasındaki yatay boşluklar, iki pipe arasındaki düşey boşluklar, programda kullanılan birer hiperparametredir. Ara nöronlarda kullanılan formüller de birer hiperparametre sayılabilir.

Geliştirme Süreci:

Projenin başındaki fikrimiz, programı sabit bir harita üzerinde sadece genetik algoritma kullanarak geliştirmekti. Bunu, belli zaman aralıkları sonrasında bireyin genetik ünitesindeki binary string üzerindeki değerleri kullanarak yapmayı düşündük. Her 0.1 saniyede bir string içerisindeki değeri okuyup, 1 ise zıpla 0 ise zıplama mekanizmasını yapmayı düşündük.

Konu örneklerini araştırırken, NEAT algoritması ile yapılan programların basarisini gördük. Ardından kendi projemizi de bu yöntem ile yapmaya karar verdik.

Projemizi processing dili kullanarak yaptık. Processing dili Java tabanlı, tasarımcılar ve kodlamaya yeni başlayan kişiler için geliştirilmiş bir dildir. Animasyon ve akan görüntü barındıran programlar için kullanışlıdır.

Projemiz 4 sınıftan oluşmaktadır. Bunlar sırasıyla Main, Brain, Pipe ve Bird'dür.

Main:

Simulasyon Main sınıfı içerisinde yapılır. Setup fonksiyonu içerisinde popülasyonun oluşturulması ve haritanın yüklenmesi yapılır.

Simulasyonun devamlılığı ve fonksiyonalitesi draw fonksiyonu içerisindedir. Duvara ya da yere çarpan kuşların silinmesi, duvarların oluşturulması, ara nöron formülleri, bu formüllerin hesaplanıp puana dönüştürülmesi, skor hesaplanması bu fonksiyon içinde gerçekleşir.

Genetik algoritma için crossover ve mutasyon fonksiyonları da Main içindedir. Projede kullandığımız crossover fonksiyonu tek noktadan kırılım yapmaktadır. Mutasyon fonksiyonumuz ise birden fazla noktaya etki edebilmektedir.

Pipe:

Duvarlar ile ilgili bilgilerin ve fonksiyonların bulunduğu sınıftır. Duvarların ekranda gözükmesi, yerinin güncellenmesi, ekrandan silinmesi ve kuşların duvara vurma durumunun kontrolü bu sınıf içerisinde yapılır.

Brain:

Ağırlık katsayıları, tahmin ve aktivatör değerlerinin ve fonksiyonlarının bulunduğu sınıftır. Bias ve ağırlık değerlerinin ilk değerlerinin atanması burada yapılır.

Bird:

Kuşların ekranda gözükmesi, zıplama fonksiyonu, hareket sonrası güncelleme ve x-y pozisyon getter fonksiyonları bu sınıf içerisindedir.

Hiperparametreler:

Mutasyon Orani: Jenerasyonlar arasında genetik tekdüzeliğin engellenmesi için kullanılan değer. Jenerasyonlar ilerledikçe bireyler birbirine çok benzemeye başladığı için projemiz için bu değerin 0.2 ve üzerinde olması gerektiğini düşünmekteyiz.

Fitness: Flappy Bird oyunundaki basari gidilen yatay mesafe ile ölçülebileceği için skor fonksiyonu dışında bir fitness fonksiyonu yazılmasına gerek duyulmamıştır.

Selection: Simülasyon içerisindeyken, çoğu kuş duvarın yan kısımlarına çarptığı ve bu alanların pozisyonları ayni olduğu için bir sonraki jenerasyona sadece en iyi skora sahip iki kusun seçilmesini mantıklı bulduk.

Spacing: Duvarlar arasındaki düşey ve yatay mesafeleri temsil eder. Azaltılması oyunu zorlaştırır.

Popülasyon: Jenerasyon içerisindeki bireylerin sayısını temsil eder. Artırılması doğruluk oranını artıracak fakat simülasyonların daha yavaş yapılmasını sağlayacaktır.

Karar Eşiği: Ara nöron puanları sonucu çıkan değerin, kusun atlaması için gereken eşiktir. Artırıldığında kuşlar daha aşağıda bulunacak ve genel olarak daha yükseğe zıplama yapmak istemelerine neden olacaktır.

Onyargi: Ağırlıklar eşitken ya da simülasyonun ilk jenerasyonları içerisindeyken, kuşların zıplamaya olan isteğidir.

Ara Nöron Formülleri:

```
float envStatus[] = new float[6];
envStatus[0] = birds.get(j).retY() / height;                    //distance of bird to ground
envStatus[1] = birds.get(j).vel;                                //bird's velocity
envStatus[2] = (237.5-birds.get(j).retY())/ height;             //distance of bird to center

Pipe p = returnNearestPipe(pipes,birds.get(j));                 //Find nearest pipe
envStatus[3] = p.x / width;                                     //pipe's x position
envStatus[4] = (birds.get(j).retY() - ( -p.h + p.spacing)) / height;   //distance of bird to top pipe
envStatus[5] = (birds.get(j).retY() - ( height/2 - p.h)) / height;     //distance of bird to bottom pipe

float result = brains.get(j).predict(brains.get(j).coeffs,envStatus);   //Calculate the result of the neural network
```

envStatus[0]-> Kusun yere olan uzaklığını temsil eder.

envStatus[1]-> Kusun düşey yönde hızını temsil eder.

envStatus[2]-> Kusun ufuk çizgisinden(Ekranın ortası) düşey olarak uzaklığını temsil eder.

envStatus[3]-> En yakın duvarın kuşlara yatay olarak uzaklığını temsil eder.

envStatus[4]-> Kuşların en yakın duvarın üst kısmına olan uzaklık farkını temsil eder.

envStatus[5]-> Kuşların en yakın duvarın alt kısmına olan uzaklık farkını temsil eder.

Farklı Çalıştırma Örnekleri ve Hiperparametre Denemeleri

Projemizin çıktısını etkileyen Mutasyon oranı, Popülasyon, Neural Networkte karar mekanizması için threshold değeri ve iki boru arasındaki mesafede deney yaparak çeşitli çıktılara ulaştık.

Tablolarda sol üst ilk hücrede girilen bilgiler incelenen hiperparametrenin haricindeki hiperparametrelerin değerini temsil etmektedir.

Flappy Bird Pipe Gap HyperParameter

Pop: 300 / Mut: 0.2 NN Threshold: 0.5	Pipe Gap: low	Pipe Gap: medium	Pipe Gap: high
Gen1	0	0	2
Gen2	1	1	0
Gen3	6	11	3
Gen4	2	9	50
Gen5	17	72	44
Gen6	12	100+	200+
Gen7	20	100+	200+
Gen8	13	100+	200+
Gen9	18	100+	200+
Gen10	16	100+	200+

Yukarıdaki tabloda görüldüğü üzere borular arasındaki boşluk arttığında kuşların geçme oranı artmaktadır fakat açıklığın çok olması da gerçekçilik açısından mantıklı bir seçim olmayacaktır. Orijinal oyundaki boşluğu göz önüne alarak benzer olması açısından orta seviyede bir boşluğu seçmeye karar verdik.

Tabloda görülen 100+ / 200+ gibi değerler oyunun 100/200 puan aşamasına gelindiğinde sonlandırıldığını ifade etmektedir bunun sebebi büyük ihtimalle kuşların sonsuza kadar gideceğini öngörmemizden kaynaklanmıştır.

Flappy Bird Mutation, Population HyperParameter

Number of generations required for more than 100 score, Pipe Gap: Medium NN Threshold: 0.5	Pop: 100	Pop: 300	Pop: 500
MutateRate: 0.2	10th Gen	5th Gen	4th Gen
MutateRate: 0.3	9th Gen	7th Gen	5th Gen
MutateRate: 0.4	7th Gen	4th Gen	4th Gen

2. İncelememizde popülasyon ve mutasyon hiperparametrelerini değiştirerek deneyimizi tamamladık. Deneyimiz belirtilen hiperparametrelerde 100 puanına kaçıncı jenerasyonda ulaştığını test ettik. Açıkça görüldüğü üzere popülasyon sayısı arttığında doğrudan başarı artmaktadır. Fakat performans da göz önüne alındığında 1000-2000 gibi değerler verilmesi çalışmayı oldukça yavaşlatmaktadır. Mutasyon oranında ise tutarsızlıklar bulunsa da çeşitlilik oluşturulabilmesi açısından yüksek bir oran verilmesinin öğrenmeyi kolaylaştırdığı varsayımında bulunabilir.

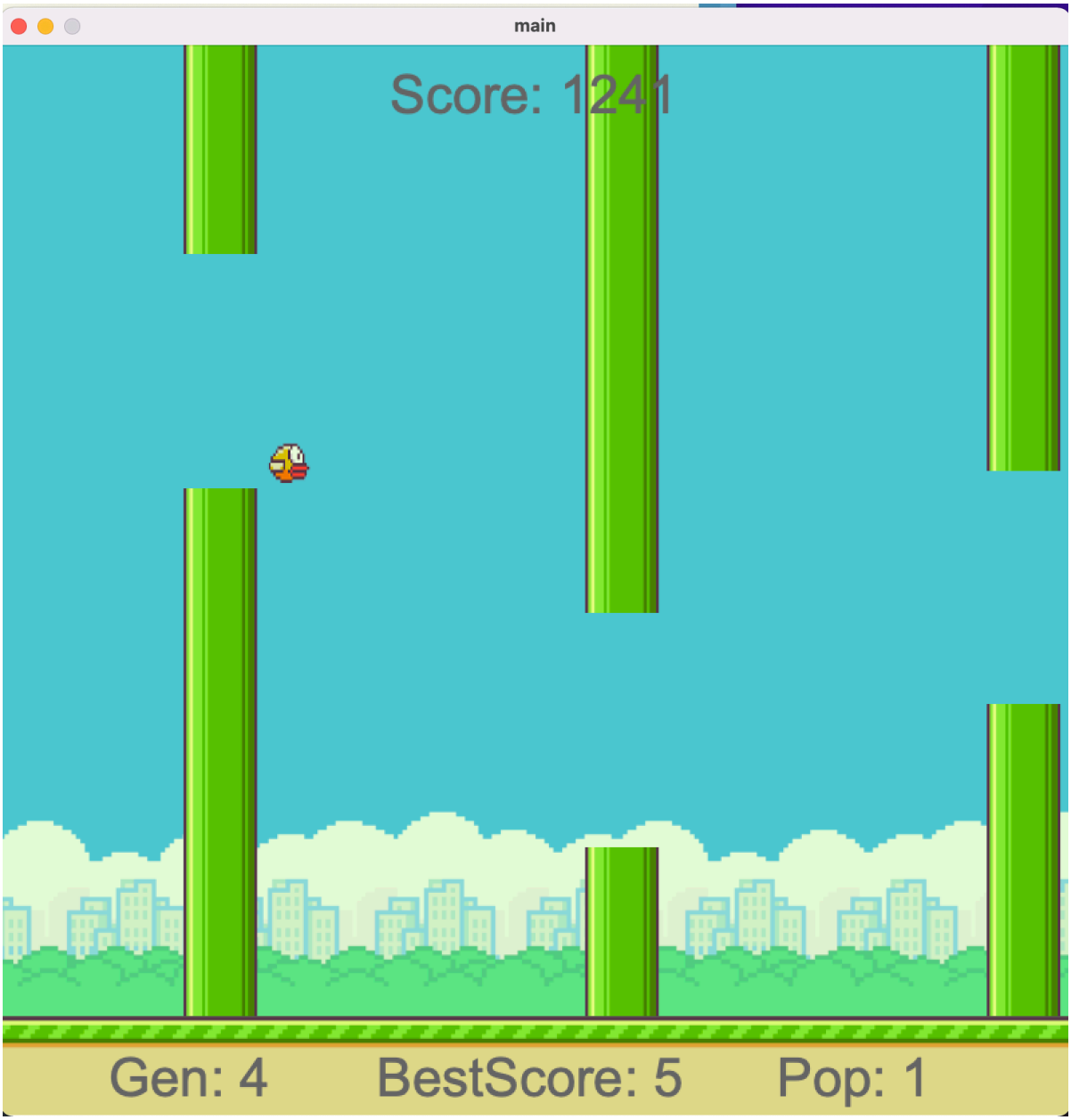
Flappy Bird NN Threshold HyperParameter

Pop: 300, Mut: 0.2, Pipe Gap: medium	Best of 5th Gen
0.2	Cannot reach even 1
0.4	2
0.5	100+
0.6	100+
0.7	3

Bir diğer parametremiz neural network sonucumuzla alakalı, neural network yukarıdaki anlatılmış olan ara nöron değerlerinin ağırlığına göre bir sonuç hesaplar. Bu sonuç kuşun zıplaması veya zıplamaması gerektiğine karar verir. Tablomuzda görüleceği üzere 0.2 ve 0.4 değerlerinde kuşlar sürekli zıpladığı için yanlış karar vermiş olur ve borular arasından geçmesi gerektiğine karar veremez. 0.5-0.6 değerlerinde ise optimum sonuçlar elde edilmiş olur ve kuşlar zıplayıp-zıplamama kararını doğru vermiş olur. 0.7 değerinde ise başarı görüldüğü üzere bir hayli düşmüştür bunun sebebi ise kuşların zıplama kararı veremediği için düşüp ölmesinden kaynaklanmaktadır.

Tablolarımız incelendiğinde ve alınan sonuçlar göz önüne alındığında optimum ve başarılı sonuçlar alınabilmesi için boruların orta boşlukta olması, 300 popülasyon, 0.2 mutasyon oranı ve 0.5 eşik değeri verilmesi kararlaştırılmıştır.

Bu çıkarımlara dayanarak çalıştırılan ve uzun bir süre beklenen :) çalıştırma örneği aşağıda gösterilmektedir. Kuşlarımız sadece 4. Jenerasyonda ve bir önceki maksimumu 5 olmasına rağmen kalan son kuş ile 1241 e kadar gidebilmiştir ve daha fazlasına da gidecektir.



Burada değinmek istediğimiz bir farklı konu ise genetik algoritmalarda ve neural networklerde şans faktörü. Boruların randomlarından gelen pozisyonlar örneğin çok aşağıdan çok yukarıya çıkması gerekmesi gibi faktörler öğrenme hızını/ başarısını değiştirmektedir. Hiperparametre inceleme tablosuna 300 popülasyon, 0.2 mutasyon oranı için 100 puanın üzerine çıkması için 5 jenerasyon gerekirken gösterilen örnekte bu değere 4. Jenerasyonda ulaşmaktadır.

Neural network işleminde incelenmesi gereken değerleri düşünürken birçok farklı etmen düşündük örneğin alt ve üst boruların yerden yüksekliğinin de zıplama kararında etki etmesi gerektiğini düşünmüştük fakat sonrasında aslında boruların pozisyonundan bağımsız olarak kuşun o boruların pozisyonuna olan uzaklığının daha önemli olduğuna karar verdik ve bu nedenle yukarıdaki belirlemiş olduğumuz etmenler ile karar vermesine gerektiğine karar verdik. Aşağıda 52 puana ulaşmış en başarılı kuşun ağırlık değerlerini görebilirsiniz, bir sonraki jenerasyonda 200+ puana ulaştığı için ağırlık değerlerini göremedik :)

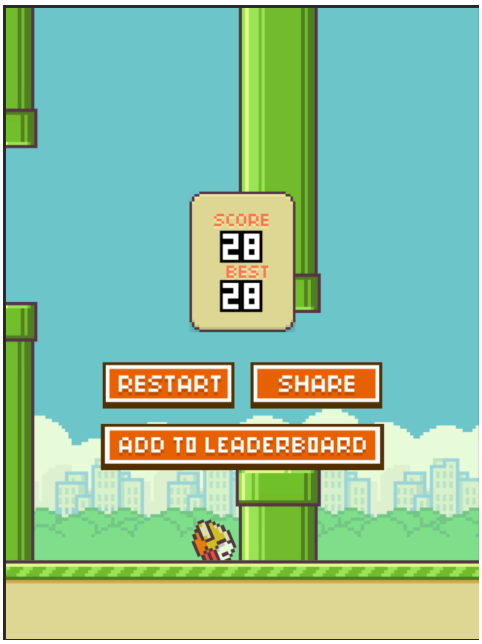
```
[0] 0.26534128
[1] 0.02196565
[2] 0.0055652554
[3] 0.14585945
[4] 0.089268334
[5] 0.9110322
```

Sistemin Başarı Oranı

Öncelikle öneri raporunda belirttiğimiz toplam boru sayısını geçilen boru sayısına oranını ölçme düşüncemiz, yaptığımız proje sonucunda mantıklı bir yaklaşım olmadı çünkü random dinamik bir şekilde üretilen boruların toplam sayısını bilememekteyiz.

Bundan dolayı ikinci düşüncemiz olan ortalama bir insanın oyunda aldığı skorla kıyaslama yapmayı düşündük. Kendi oynayışımızda 10 oyunda maksimum 28 skoruna ulaşabilirken, internet üzerinden oyunu oynatabilen bir platformda aşağıda görülen en yüksek skorların elde edildiğini gördük.

http://flappybird.io/leaderboard/



Top 10 scores in the last hour

ez	155
be a men not a man	103
EliteTierGamer	95
Angel:)	82
hugoguapooo	79
grant	73
mason hana	68
Mason	62
Just a random try	56
poopyinmypantsgross on roblox	56

Skorlar göz önüne alındığında (ve büyük ihtimalle hile yapılabileceğini de göz önüne alarak) maksimum elde edilebilen skorun ortalama 100-150 olduğunu görmekteyiz. Programımızda ise optimum hiperparametrelerde ve 4. Jenerasyonda 1241 skoruna ulaştığını görmüştük. Burada tabiki genetik algoritmada 300 popülasyon faktörü de göz önüne alınmalı normal oyunda aynı anda 1 kuş uçarken bizim programımızda öğrenme aşamasında 300 kuş uçmaktadır. Fakat yukarıda verilen 1241 puanındaki ekran görüntüsünde görülebileceği üzere belirli bir yerden sonra diğer kuşların ölmesiyle sona kalan tek kuşun ölmeden yüksek skorlara ulaşabildiği görülmektedir.

Kaynak Listesi:

Projemize başlamadan önce daha önceden yapılan örnek projelere ve videolara göz attık bunlardan başlıcası YouTube üzerinden Tech with Tim kanalında yayınlanmış olan “Python Flappy Bird AI Tutorial (with NEAT) - Creating the Bird” videosudur.

Çalışma algoritması olarak buradaki temel yapıyı göz önünde bulundurduk sonrasında neural network nedir nasıl çalışır karar mekanizmaları nasıl oluşturulur sorularına cevap bulabilmek adına birkaç video ve yazı/makale araştırması yaparak belirli bir seviyede öğrenmiş olduk.

Son olarak daha önce deneyimlemediğimiz bir dil olduğundan processing dili üzerinde yapılmış birkaç projeyi inceledik projeye başladık.