Q-LEARNING İLE YOL PLANLAMASI

Şevki Karagöl
Bilgisayar Mühendisliği Bölümü
Kocaeli Üniversitesi
170201009

Mustafa Yiğit
Bilgisayar Mühendisliği Bölümü
Kocaeli Üniversitesi
180201108

ÖZET- Bu projede, projeyi yapan kişiler için derin öğrenmenin temellerinden biri olan Q-Learning algoritmasının mantığını anlaması, bunun ardından pekiştirmeli öğrenme (Reinforcement Learning) yöntemleri hakkında bilgi edinmesi, bu yöntemler ile bir masaüstü uygulaması geliştirilmesi ve derin öğrenme temellerini yazma becerisinin geliştirilmesi amaçlanmaktadır. Bu amaç doğrultusunda öğrencilerden, bir Q-Learning algoritması ile 50x50'lik bir matris içerisindeki engeller arasından belirli konumlar arasındaki optimum yolu bulması bazı isterler doğrultusunda istenmiştir.

Bu amaçlar ve isterler doğrultusunda Python programlama dili ve görselleştirme aracı olarak Pygame, Tkinter modülleri kullanılarak proje isterlerini uygulayan bir masaüstü uygulaması geliştirilmiş, test çalışmaları yapılmış ve kullanıma hazır hale getirilmiştir.

Anahtar kelimeler-

Q-Learning, python, pygame, tkinter, masaüstü uygulaması, derin öğrenme, pekiştirmeli öğrenme, engel, matris

I.GİRİŞ

Bu projede ilk olarak açılan arayüzde kullanıcıdan başlangıç ve bitiş konumlarına atama yapmak için veri alındı. Daha sonra ise alınan bu veriler doğrultusunda rastgele engellerden ve geçişlerden oluşan bir çevre (matris) tanımlandı ve bunun ardından rastgele oluşturulan çevreye göre bir ödül tablosu (reward table) oluşturuldu. Elde ettiğimiz çevre ve ödül tablosu üzerinden Q-Learning algoritması kullanılarak Q-table dolduruldu. Algoritma, geliştiriciler tarafından belirlenen süre boyunca çalıştırıldı ve doldurulan Q-table doğrultusunda optimum yol (rota) bulunarak arayüz üzerinde bu yol gösterildi. Ardından da bölüm başına elde edilen ödül ve atılan adım sayısına bağlı olarak iki adet grafik çizdirildi. En son ise engellerin, geçişlerin ve başlangıç-bitiş noktalarının konumları "engel.txt" içerisine yazdırıldı.

Bu projede Python dili, Pygame ve Tkinter modüllerinin ve Q-Learning algoritmasının bir arada kullanımına yönelik bir çalışma gerçekleştirilmiştir. Aynı zamanda öğrencilerin, proje isterlerlerinin çözümününe yönelik araştırdığı algoritmalar IDE aracılığıyla bilgisayar ortamına aktarılmıştır.

II.TEMEL BİLGİLER

Bu proje Python dili ile geliştirilmiş olup, geliştirme ortamı olarak "Visual Studio Code" ve “Spyder 4” kullanılmıştır. İlk etapta proje için bir yol haritası çıkarılarak ön hazırlık sürecine girilmiştir. Bu aşamada projenin isterlerine yönelik araştırmalar gerçekleştirilmesi adına grup içerisinde bir iş bölümü yapılmış olup elde edilen veriler doğrultusunda projenin ana hatları ortaya çıkarılmış ve büyük ölçüde karşılaşılabilecek problemler saptanıp çözümlendirilmeye çalışıldıktan sonra IDE ortamında projenin ilk adımları atılmıştır.

Yapılan ön hazırlık sürecinde bir masaüstü uygulamasının nasıl oluşturulacağı, proje isterlerine cevap verecek modüllerin nasıl yükleneceği, yüklenen modüller üzerinden Q-Learning işlemi yapacak algoritmaların nasıl yazılacağı, arayüz tasarımının nasıl olacağı, arayüz üzerinden nasıl veri alınacağı gibi problemler üzerinde durulmuştur. Bu konulara ve problemlere yönelik gerekli araştırmalar yapıldıktan sonra projeye şekil verme aşamasına gidilmiştir.

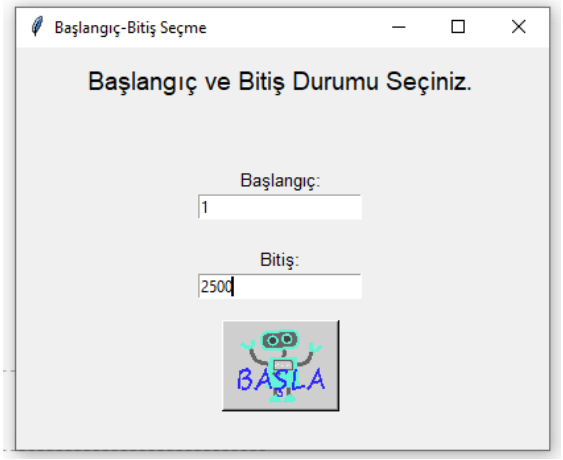
Proje ön hazırlık süreciyle birlikte yaklaşık on günlük bir süreçte tamamlanmıştır.

Not: Proje görselleştirilmesi 50x50 üzerinden yapılmıştır. Boyut değişikliği durumunda “gridSize” değeri de yeni boyuta göre ayarlanmalıdır.

III.YÖNTEM

Bu projede izlenilen yol aşağıda anlatılmıştır:

İlk olarak “Tkinter” modülü kullanılarak başlangıç-bitiş noktalarının konumları kullanıcıdan alınmıştır.



Başla butonuna basıldığında girilen konum değerleri doğrultusunda uygulama çalışmaya başlayacaktır.

Daha sonra Q-Learning gerçekleştirilmesi için rastgele bir çevre oluşturulur.

```
# Çevre tanımlama----------------------------------------------------
cevreMatrisi = np.ones(boyut*boyut)
cevreMatrisi[:int(boyut*boyut*0.3)] = 0

np.random.shuffle(cevreMatrisi)
cevreMatrisi = (np.random.rand(boyut*boyut) > 0.3).astype(int)
cevreMatrisi = np.where(cevreMatrisi == 1, 3, cevreMatrisi)
cevreMatrisi = np.where(cevreMatrisi == 0, -5, cevreMatrisi)
cevreMatrisi = cevreMatrisi.reshape((boyut,boyut))
```

Oluşturulan çevreye alınan girdi doğrultusunda başlangıç-bitiş noktaları eklenir.

```
cevreMatrisi[int(bitisDurumu/boyut)][bitisDurumu % boyut]=999
cevreMatrisi[int(baslangicDurumu/boyut)][baslangicDurumu % boyut]=0
```

Hemen ardından her şeyiyle tamamlanmış çevre baz alınarak bir ödül tablosu (reward table) oluşturulur.

```
# Ödül tablosu (reward table) tanımlama
rewards = np.array(np.zeros([boyut*boyut, boyut*boyut]))

for i in range(0, boyut):
    for j in range(0, boyut):
        if(i != boyut-1):
            rewards[i*boyut+j][(i+1)*boyut+j] = cevreMatrisi[i+1][j]  # alt
        if(i != 0):
            rewards[i*boyut+j][(i-1)*boyut+j] = cevreMatrisi[i-1][j]  # üst
        if(j != 0):
            rewards[i*boyut+j][i*boyut+(j-1)] = cevreMatrisi[i][j-1]  # sol
        if(j != boyut-1):
            rewards[i*boyut+j][i*boyut+(j+1)] = cevreMatrisi[i][j+1]  # sağ
        if(j != boyut-1 and i != 0):
            rewards[i*boyut+j][(i-1)*boyut+(j+1)] = cevreMatrisi[i-1][j+1]  # sağ üst
        if(j != 0 and i != 0):
            rewards[i*boyut+j][(i-1)*boyut+(j-1)] = cevreMatrisi[i-1][j-1]  # sol üst
        if(j != boyut-1 and i != boyut-1):
            rewards[i*boyut+j][(i+1)*boyut+(j+1)] = cevreMatrisi[i+1][j+1]  # sağ alt
        if(j != 0 and i != boyut-1):
            rewards[i*boyut+j][(i+1)*boyut+(j-1)] = cevreMatrisi[i+1][j-1]  # sol alt
```

Elde edilen çevre ve ödül tablosu doğrultusunda

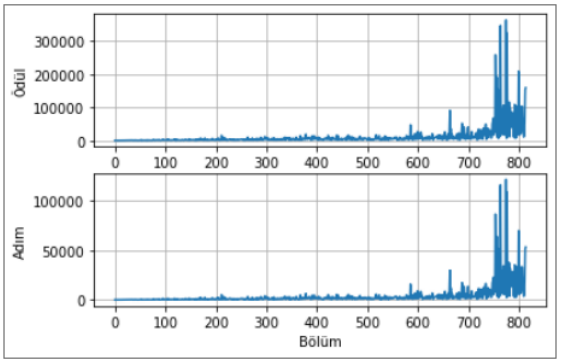
$$Q(durum, aksiyon) = R(durum, aksiyon) + \gamma \times Max\{Q(sonraki\ durumlar, tüm\ aksiyonlar)\}\ \gamma$$

şeklinde olan Q-Learning formülü kullanılarak algoritma çalışmaya başlar.

Algoritma sonucu bölüm başına elde edilen ödül ve atılan adım sayısı

```
# Grafik işlemleri------------
fig ,axs = plt.subplots(2)
plt.figure(figsize=(20,100))
axs[0].plot(odulListesi)
axs[0].set_xlabel("Bölüm")
axs[0].set_ylabel("Ödül")

axs[1].plot(adımListesi)
axs[1].set_xlabel("Bölüm")
axs[1].set_ylabel("Adım")

axs[0].grid(True)
axs[1].grid(True)

plt.show()
```

üstte görüldüğü gibi tasarlanmış olup

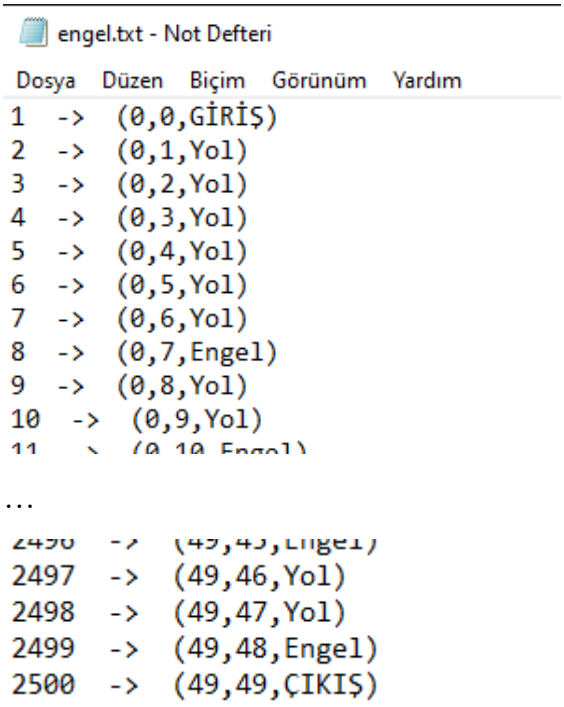


şeklinde görülmektedir.

İsterlerden biri olan “engel.txt” içerisine yazdırma işlemi

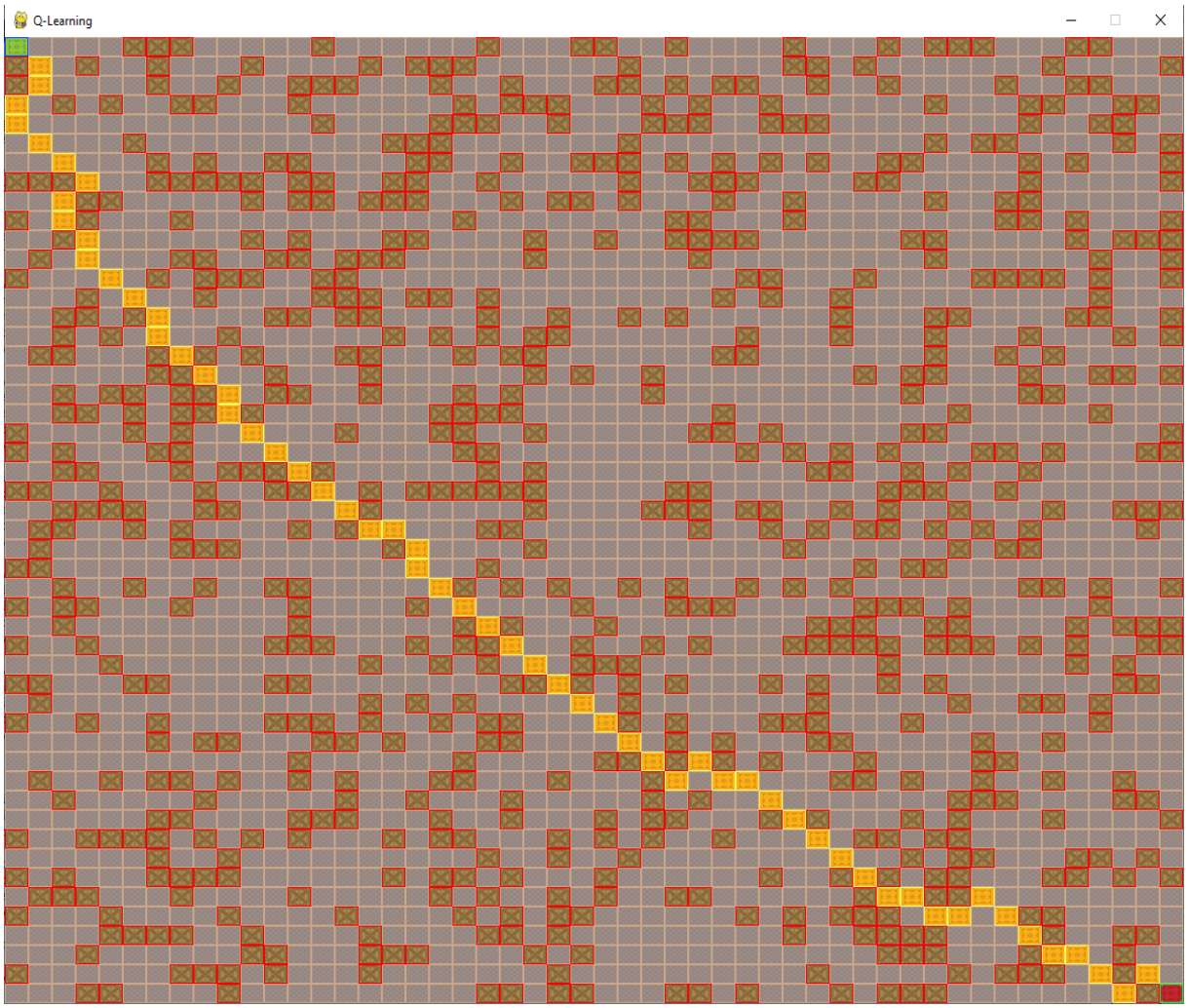
```
# engel.txt oluşturma----------------------------------------------------------
# Oluşturulan çevrenin tüm özelliklerini bir txt dosyasına yazdırır.
def engelYazdirma(cevre):

    engelDosyasi = open("engel.txt", "w")
    a=1
    for i in range(boyut):
        for j in range(boyut):

            if cevre[i, j] == -5:
                engelDosyasi.write(str(a)+"  ->  ("+str(i) + "," + str(j) + ",Engel)\n")
            elif cevre[i, j] == 0:
                engelDosyasi.write(str(a)+"  ->  ("+str(i) + "," + str(j) + ",GİRİŞ)\n")
            elif cevre[i, j] == 999:
                engelDosyasi.write(str(a)+"  ->  ("+str(i) + "," + str(j) + ",ÇIKIŞ)\n")
            else:
                engelDosyasi.write(str(a)+"  ->  ("+str(i) + "," + str(j) + ",Yol)\n")
            a+=1

    engelDosyasi.close()
```

şeklinde yapılmakta ve



…

şeklinde görülmektedir.



En sonunda ise Q-Learning algoritması ile elde edilen optimum yol (rota) Pygame modülü kullanılarak görselleştirilmiş hali üstte gösterildiği gibi kullanıcıya sunulur.

IV.KABA KOD

1) Kullanıcı metin kutularını doldurdu ve "Başla" butonuna tıkladı.

2) Ajanın öğrenme işlemini yapacağı ortam (çevre) oluşturuldu.

3) "engel.txt" dosyası oluşturuldu ve doldurduldu.

4) Ajan Q-Learning algoritmasını kullanarak hareket etmeye başladı ve bu doğrultuda Q-table içeriğini doldurdu.

5) Ajan engele çarptı ve başa döndü (bu adım öğrenme işlemi tamamlanana kadar sürekli tekrarlandı).

6) Yapılan hareketler sonucu optimum rota (yol) elde edildi.

7) Plot table ve optimum yolun gösterildiği arayüz kullanıcıya sunuldu.

V.REFERANSLAR

[1]https://medium.com/deep-learning-turkiye/python-i%CC%87le-q-learning-ef6413aa896e

[2]https://medium.com/@sddkal/python-ve-makine-%C3%B6%C4%9Frenmesi-q-learning-temelleri-181d29326782

[3]https://towardsdatascience.com/a-beginners-guide-to-q-learning-c3e2a30a653c

[4]https://knowledge.dataiku.com/latest/kb/analytics-ml/reinforcement-learning-visual/reinforcement-learning-q-learning.html

[5] https://www.udemy.com/course/python-ile-yapay-zeka-adan-zye-reinforcement-learning/

[6]https://www.youtube.com/watch?v=qhRNvCVVJaA

[7] https://stackoverflow.com/

[8]https://www.geeksforgeeks.org/

[9]https://www.geeksforgeeks.org/python-add-image-on-a-tkinter-button/

[10]https://pythonbasics.org/tkinter-image/

[11]https://www.pygame.org/docs/

[12]https://docs.python.org/3/library/tk.html