Q-learning ile Yol Planlaması

Ertuğrul Demir

Bilgisayar Mühendisliği Bölümü
Kocaeli Üniversitesi
170201052
ertugrulbusiness@gmail.com

Özet

Proje, labirent benzeri 50 satır ve 50 stundan oluşan bir harita üzerinde her hücre için rastgele oluşttulan ödülleri ve engelleri hesaba katarak başlangıç noktasından bitiş noktasına giden en verimli yolu Qlearning path planning yaklaşımını kullanarak bulmayı amaçlamaktadır.

Bu projeyi gerçekleştirmek için kullanılan programlama dili python'dur. Python sahip olduğu açık kaynak imkanları ve beraberinde gelen teknoleji desteğinin sağladığı yararlar göze alındığında projede tercih edilmesinde etkili olmuştur.

Projede kullanılan kütüphaneler ise matematiksel işlemler için Numpy, verilerin görselleştrilmesi için Matplotlib, kullanıcı arayüzü için ise Tkinter kullanılmıştır.

Uygulama haritanın oluşturulması ile başlar. Qlearning algoritması oluşturulan harita üzerinde eğitilir. Algoritma, eğitimleri sonucunda hangi yolların daha fazla ödül getirdiğini tutan tablolar hazırlar. Bu tablolar harita üzerinde en verimli yolun bulunmasında daha önceden elde ettiği deneyimlerden yararlanır. Bu deneyimler algoritmanın haritayı ne kadar doğru iyi öğrendiğini gösterir.

En verimli yolun bulunulacağı harita, kullanıcıdan alınan girdilere göre oluşturulmaktadır. Bu harita 50x50 boyutundadır. Haritanın her hücresinin bir ödül değeri vardır. Bu ödül değerleri algoritmanın sahip olduğu parametrik değerlere göre işleme alınmaktadır. Alınan her değer yol bulmada yöntem geliştirmek için tablolar üzerinde tutulur. Bu tablolar daha sonra en verimli yolun bulunması için algoritmanın hafızası olarak görev yapacaktır.

Algoritma tarafından oluşturulan tablolar projede verilen kurallar doğrultusunda yorumlanarak en verimli yolu bulmak için defalarca test edilir.

Testlerin sonuçları kendi aralarında karşılaştırılarak en karlı yolu çizen test sonucu alınır. Testler sonucu elde edilen yol daha sonra kullanıcı arayüzünden ekrana bir harita olarak bastırılmaktadır. Haritanın görsel olarak çizilmesinin yanında elde edilen test bilgileride grafik şekşinde kullanıcı arayüzüne yansıtılır.

1.Giriş

Projede programlama dili olarak, yapay zeka alanında kullanımıyla ünlü olmasının yanı sıra bilimsel hesaplamalarda da sıkça kullanılan Python dili tercih edilmiştir.

Projeyi geliştirmede kullanılan geliştirme ortamı ise Anaconda navigator ekosistemi içerisinde bulunun Spyder IDE'sidir. Anaconda sayesinde birbirinden bağımsız birçok farklı geliştirme ortamı kurabilmesinin yanında bu ortamların kullanılmasında paketlerin doğrudan kontrol edilebilmesi sayesinde farklı koşullarada hızlıca geliştirme yapılabilmektedir.

Projedeki Qlearning algoritması, geliştirici tarafından elle yazılarak uygulanmıştır. Algoritmanın uygulanmasında aritmetik işlemlerin optimize olması için numpy kütüphanesi kullanılmıştır. Numpy kütüphanesi kullanılarak oluşturulan veriyapıları ve gerçekleştrilen aritmetik işlemler geliştirici tarafından elle implemente edilmiş olan Qlearning algoritmasına girdi olarak verilir.

Qlearning algoritması sonraki bölümde daha ayrıntılı olarak ele alınmıştır. Qlearning algoritması kısaca defalarca deneme yanılmayla elde edilen sonuçları tablolar vasıtasıyla kayıt altına alır. Tablodaki değerler önemlerine göre ödül dağılımı ayarlanır. Kullanılan tablolar yol bulabilmek için en verimli hale getirmek amacıyla eğitilir. Eğitimle elde edilen tablo en verimli

yolu bulmak amacıyla teste tabi tutulur ve test sonuçlarından en karlısı seçilerek yol bulunur .

Harita üzerinde rastgele atanmış engeller bulunmaktadır. Bu engellerin haritadaki hücre sayısına oranı kullanıcı tarafından verilmektedir. Verilen oran doğrultusunda haritadaki engellerin sayısı değişmektedir. Değişen engel sayısı haritada yol bulmayı zorlaştırmaktadır.

Algoritma engellerin etrafından dolaşarak hedefine gitmeli ve çizdiği yolu da aldığı ödüllere göre şekillendirmelidir.Yolun oluşturulmasında proje kapsamında verilen kurallara dikkat edilerek harita üzerinde ilerlenmelidir. Kurallara uyularak en verimli yolun bulunması için kullanıcıdan alınan girdiler doğrultusunda oluşturulan haritada Qlearning algoritmasının eğitilmesi gerekmektedir. Eğitim derinliği ne kadar artarsa elde edilecek sonuç o kadar daha iyi olacaktır.

Qlearning algoritması vasıtasıyla elde edilen tablo temel alınarak harita üzerinde yol planlaması yapılır. En verimli yol için algoritma tablosu kullanılarak defalarce test yapılır ve en verimlisi kabul edilerek yol bulur.

Yolu elde ederken yapılan test sonuçları grafik olarak matplotlib kütüphanesi vasıtasıyla ekrana yansıtılır. Bununşa birlikte harita kullanıcının daha iyi anlayabilmesi için kullanıcı arayüzünde çizdirilir.

2.Qlearning Algoritması

Qlearning algoritması , belirli bir durumda bir eylemin değerini öğrenmek için model içermeyen, pekiştirmeli bir öğrenme algoritmasıdır.

2.1 Pekiştirmeli Öğrenme (Reinforcement learning)

Pekiştirmeli öğrenme, bir temsilci, bir dizi durum S ve durum başına bir dizi A eylemi içerir. Ajan, A dizisinin elemanın olan bir a eylemini gerçekleştirerek durumdan diğer durumlara geçiş yapar. Belirli bir durumda bir eylemi yürütmek, temsilciye bir ödül (sayısal bir puan) sağlar.

Temsilcinin amacı toplam ödülünü maksimize etmektir. Bunu, gelecekteki durumlardan elde edilebilecek maksimum ödülü mevcut durumuna ulaşma ödülüne ekleyerek, mevcut eylemi potansiyel gelecekteki ödülle etkili bir şekilde etkileyerek yapar. Bu potansiyel ödül, mevcut durumdan başlayarak gelecekteki tüm adımların ödüllerinin beklenen değerlerinin ağırlıklı toplamıdır.

2.2 Qlearning

Q-learning algoritmasının girdileri kazanç matrisi olarak adlandırılan R matrisidir. Bu matristeli değerlere göre ileride atılacak adımların belirlenir.

Q-learning algoritmasının çıktısı ise öğrenmenin kalitesini gösteren Q matrisidir. Q-learning iteratif bir algoritmadır ve tüm değerleri başlangıçta 0 olan Q matrisi optimal değerlere yakınsadığı da sona erer. Algoritma her iterasyonda rastgele bir durumdan öğrenmeye baslar, A'ya göre durum değiştirir ve Q matrisini günceller. A'ya göre hedef duruma ulaşıldığında iterasyon sona erer. A'ya göre bir durumdan birden fazla duruma geçis olabilir. Boyle bir durumda, olası geçişler den biri rastgele seçilir. Eğer seçilen durum hedef duruma ulaştırmıyorsa, durum rastgele olacak durum olarak belirlenir. Hedef duruma ulaşılanakadar iterasyon devam eder
Q matrisi aşağıdaki formüle göre güncellenir:

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

γ ogrenme katsayısıdır ve 0 ile 1 arasında bir değer alır. Algoritma hedefe ulaştığında yol sona erer.

3.YALANCI KOD

3.1 Qlearning Algorithm

1. Start
2. For(Episode)
    a. Qtable=Train(Start,Stop,Matrix)
        i. FirstAssigments()
            1. For (step)
            2. FindAvailableAction()
                a. SelectAction()
                b. If(randomRate > Rate)
                    i. Random
                c. Else
                    i. Greedy
            3. R=TakeRewards(a,s)
            4. MaxQ = FindMaxQ(s+1,a+1)
            5. Update
                a. Q(s,a) = R + LearningRate*MaxQ
            6. NextState = step(action)
            7. Return Qtable
    b. Path, Results = Test(Qtable,Start,Stop)
        i. While (not isCaptured)
            1. FirstAssigments()
            2. While (isEndStep)
                a. SelectAction(Qtable)
                b. NextState = step(action)
                c. Path.append(step)
3. Return (path,testResults)

3.2 Main

1. Start
2. İnputs = Input(matrix prams)
   a. Control inputs and exceptions
   b. Create random values
   c. Assign and reshape values
   d. Start matrix
3. Maze = CreateMaze(İnputs)
   a. Control matrix
   b. Construct varaibles
   c. Generate and return maze
4. Qtable = Qlearning.train(Maze)
   a. Take shape of maze
   b. Create Qtable Matrix
   c. For (episode)
      i. Select Action
      ii. Update Qtable
      iii. NextState
5. Path,Results = Qlearning.test(Qtable)
   a. While (not isTargetCaptured)
      i. SelectActionUsingQtable(Qtable)
      ii. NextState
      iii. Path.append(nextState)
6. GUI.start(Path)
   a. DrawMatirx()
   b. Paint Cells ()
   c. Assign values to cells()
   d. Write Text
7. PlotGraph(Results)
   a. Plot(results)
8. Stop

4.YAKLAŞIM

Proje deneme yanılma ve bu tecrübeler üzerinden eldeki ürünü geliştirme stratejisini izlemiştir.

Algoritma uygulaması için literatür taraması ve makale derlemeleri yapılmıştır.

Denemeler sırasında karşılaşılan hatalar ilgili teknolejiyi sunan firmaların dökümantasyonları aracılığıyla çözülmeye çalışılmıştır.

Dökümantasyonlardaki alternatif yöntemler projeye uygulanmaya çalışılmıştır.

Vaar olan sınıflar override edilerek tekrardan isteklere cevap verilbilmesi için çalışmalar yürütülmüştür.

Çözülen ve elde edilen kodlar daha sonrasında gerekli isterleri karşılabildiklerinden emin olunmak amacıyla teste tabi tutulmuşlardır.Kullanılan kodların eksik tarafları üzerine test senaryoları oluşturularak program sınanmıştır.

5.KAZANIMLAR

- Reinforcement learning
- Qlearning
- Yapay Zeka kavramları
- Nesne yönlimli programlama
- Masaüstü uygulama oluşturma
- Matematiksel Denklemlerden uygulama oluşturma
- Test ortamı oluşturma ve test etme
- Test senaryosu oluşturma
- Sınıf oluşturma
- Kalıtım
- Kullanıcı arayüzü için Component kullanımı
- Ekrana görsel çıktı verme
- Çoklu dosyaların birbirlerine bağlanması ve aralarındaki versiyon sorunlarının giderilmesi
- Geliştrime ortamlarının seri şekilde değiştirilebilmesi ve isterlere uygun senaryoların yürütülmesini sağlamak.

6.EKSİKLİKLER

- Projenin zaman yönetimi
- Kullanıcı arayüzü tasarımı
- Çıktının otomatik olarak Frame üzeinde gösterilmesinin sağlanamaması.
- Efektif olmayan Kullanıcı etileşimleri
- Verilerin uygulama içerisinde kullanılamaması
- Donanımsal isterlerin karşılanması
- Yazılımsal bağımlılıkların giderilebilmesi için gerekli ortamların stabil şekilde sağlanmasındaki zorluklar.
- Elde edilen sonuçların iyileştrilmesi
- En verimli yol için eğitim aşamasının derinliğinin en verimli yolu bulabilmesini sağlayacak seviyede uygulanamaması

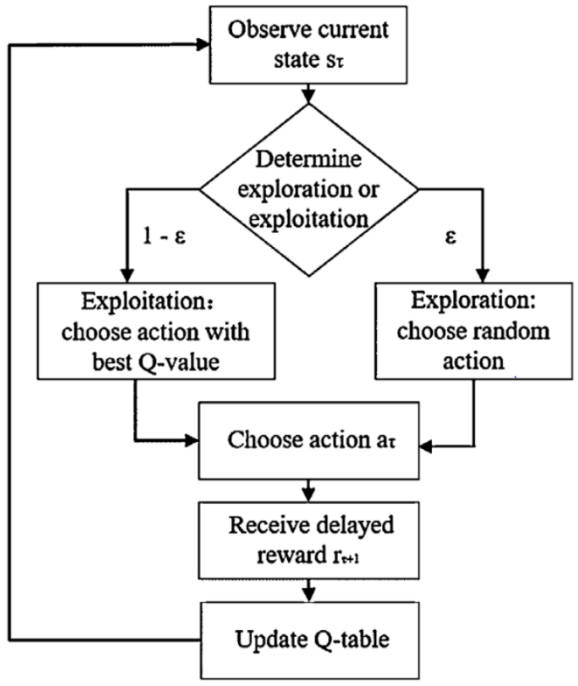
7.SONUÇ

Program Kullanıcı arayüzü üzerinden kontrol edilebilir ve çıktıları konsoldan alınabilir.

Açılan ekranlar vasıtasıyla kullanıcı teknik bilgi gereksinimi duymadan uygulama içerinde ihtiyacı olan çözümlere erişebilir ve uygulayabilir.

Bunlar ile birlikte Kullanıcı verilerini uygulama içerisinde gerçek zamanlı izleyemez ve Google Map üzerinde görüntüleyemez.

8. FLOW CHART



9.Kaynakça

1) https://docs.python.org/3/reference/
2) https://docs.python.org/3/
3) https://numpy.org/doc/
4) https://pub.dev/documentation/google_maps_flutter/latest/
5) https://matplotlib.org/
6) https://www.sciencedirect.com/
7) https://www.researchgate.net/
8) https://stackoverflow.com/
9) https://github.com/flutter/flutter
10) https://www.tutorialspoint.com/index.htm
11) https://www.w3schools.com/