

**Doç. Dr. Ekin EKİNCİ**

**Arş. Gör. Emin GÜNEY**

**Yapay Zekâ ve İlkeleri Dersi**

**Teknoloji Fakültesi**

**Bilgisayar Mühendisliği 2.sınıf**

**OgrenciNo =**

**“B210109048,22010903049”**

**AdSoyad =**

**“Furkan Duran, Ahmet VATANSEVER”**

Öncelikle Q learning ve yapay öğrenme hakkında bazı bilgilere yer vererek kodlarımızın ve raporumuzun daha anlamlı hale gelmesini istedik.

Q-Learning Algoritması

Q-Learning algoritması pekiştirmeli öğrenmenin en çok bilinen algoritmalarından biridir. Algoritmadaki temel amaç bir sonraki hareketleri inceleyerek yapacağı hareketlere göre kazanacağı ödülü görerek bu ödülü MAXİMİZE etmeye çalışarak hareket etmektir. Ajanın ödül haritası, gitmesini ya da gitmesi istenilmeyen yerler daha önceden ona öğretilerek ödül tablosuna yazılır. Ajanın tecrübeleri de bu tabloya göre şekillenecektir. Biz oluşturduğumuz ajanda bunları X ajanın beyni ya da O ajanın beyni olarak adlandırdık. Öncesinde yapılacak olan tekrarların sayısı arttıkça ajanın Q-Tablosu dolarak daha da güçlü kararlar vermesini sağlayacak. Tabii ki ajanımızın en başta hiçbir tecrübesi olmadığı için Q Tablomuz bomboş olacak ve ajan rastgele hareket ederek edecektir. Bu rastgelelik ajanın ilk ödülü bulmasına kadar sürecektir. Ajan ödülün olduğu bir yere geldiği anda ödüle gelmeden önceki yerini bilir ve bu yerin değerini kendi tecrübelerini biriktirdiği Q-Tablosuna yazar. Bu yazma işleminin belirli bir algoritması bulunmaktadır.

metin, yazı tipi, beyaz, hat sanatı, kaligrafi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Q(s,a) -> Q(state , action) yani Q(bulunduğumuz konum, gideceğimiz konum) manasına gelmektedir. “lr” dediğimiz değer öğrenme katsayısı (learning rate),“r(s,a)” dediğimiz değer bizim {bulunduğumuz, gideceğimiz} ödül tablomuzdaki ödül değerimiz, “Y” değeri gamma, “max (Q (s’,a’))” değeri ise gidebileceğimiz {gideceğimiz,gideceğimiz yerden gidebileceğimiz} yerlerin en yüksek Q değeridir.

Her seferinde ajanımız bu algoritmaya göre hareket ederek bir ilerisini tahmin ederek hareketlerini devam ettirir ve ödüle ulaşır ve sonrasında tekrardan rastgele bir yerden harekete başlar ve tekrardan ödülü bulmaya çalışır. Bu işlemler devam ettikçe ajan hangi durumda nasıl karar vermesi gerektiğini daha da iyi bilmeye başlar. Yani biz kodumuzda da kullandığımız ajanı eğittiğimiz kısımda oyun tekrarını ne kadar arttırırsak ajanımız o kadar güçlü olacak demektir veya ne kadar oyun tekrarını azaltırsak ajanın kazanma olasılığı o kadar düşecek demektir.

Projemizi Çalıştırabilmemiz için Q learning algoritmasına sahip kodlar TicTacToe oyunu oynamak için kodlar ve bu ikisini entegre ederek yapay öğrenme ile ajanlarımızı yapay öğrenme sağlayacak ve oyunu oynamamızı gerçekleştirecek kodlar olmak üzere 3 Python dosyası olarak işlemlerimizi gerçekleştirdik, sırasıyla her .py dosyası için açıklama yapılacaktır. İlk olarak TicTacToe oyunun temel kodlarının bulunduğu TicTacToe dosyasından daha sonra Q learning algoritmamızın ve yapay öğrenmeyi gerçekleştireceğimiz ve

1.TicTacToe Oyununu Oluşturan Temel Kodların Oluşturulması(xox.py)

metin, yazı tipi, ekran görüntüsü, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturulduÖncelikle oyunu oynayabilmemiz için bazı temel bileşenlere ihtiyacımız var. Bu temel bileşenler belirli bir durumda kontrol edilmek ve değiştirilmek üzere değerleri değişebilir ve bunları fonksiyonlar içerisinde çağırıp kullanabilmemiz için global değişkenler olarak tanımlamamız lazım. Örnek olarak kodumuz da;

board (X ve O ların yerleşeceği kısım), currentPlayer (Sırası gelen = X ya da O), winner (kazanan), gameRunning(Oyun devam ediyor mu kontrolu)

Bu değişkenlerimiz fonksiyonlar içerisinde kullanmamız gerekecek bundan dolayı bu değişkenlerimizi global olarak tanımlamamız gerekecek.

Şimdi ise oyun durum kontrollerini fonksiyonlara atayarak işlemlere devam edeceğiz. Öncelikle

def printGameState(board): Oyun tahtamızı görebilmek adına oluşturduğumuz bu fonksiyon 3x3 bir matris formatında tahtamızın mevcut durumunu yazdırmamızı sağlıyor.

def get\_empty\_cells(board): Bu fonksiyon ise bir for döngüsü kullanılarak tablomuzdaki boş indexleri görmemizi sağlayacak bir işleve sahip

def get\_cells\_state(board): Oyun tahtamızın mevcut durumunu görmemizi sağlayan fonksiyondur.

def get\_current\_game\_tuple(board): Tahtanın mevcut durumu liste halinde görmemizi sağlar. (Q learning algoritmalarında tüpleler değiştirilmez oldukları için kullanılır.)

def playerInput(board, inp): bu fonksiyonumuz ise sıradakinin bir hücreye hamle yapmasını sağlayacak olan fonksiyondur. Burada eğer doldurulacak hücre dolu ise kullanıcıdan tekrar değer girmesi istenir.

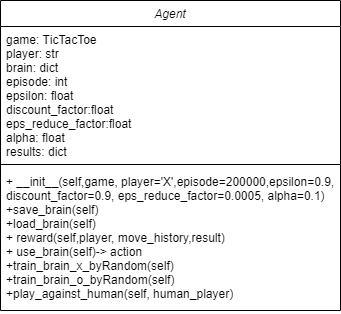
Şimdi ise hangi durumlarda oyununun kazanılmış olacağını belirlemek adına global değişkenizim olan winner değişkenini çağırarak fonksiyonlar içerisinde kullanacağız.

def checkHorizontal(board), def checkRow(board), def checkDiagonal(board) :bu fonksiyonlar sırasıyla yatay , dikey ve çapraz olarak üç hücreyi kontrol eder ve bu kontrol sonrasında hücreler aynı ise winner değişkenine bu atanarak True döndürülür. def checkWin(): bu fonksiyonumuz ise bu uç durumu kontrol ederek oyunun kazanılıp kazanılmadığını döndüren fonksiyondur. 1 X için kazanma durumu -1 O için 0 ise berabere sonucu durumu için döndürür.

def checkforTie(board): bu fonksiyonun tahtada boş hücre kalmadıysa global gameRunning değerini döndürerek beraberlik kontrolu sağlar.

def switchPlayer(): yine global olarak tanımladığımız currentPlayer değişkenini çağırarak mevcut oyuncuyu değiştirmemizi sağlayan fonksiyondur.

2.Q-Learning Algoritması ile TicTacToe Oynayan ve Öğrenen Ajan Tanımlama (Agent.py)

Öncelikle oyunu oluşturduğumuz python dosyasını sonrasında Q-learning algoritmasını kullanırken ajanın öğrenmesini sağlayacak olan verileri Q-Tabloda saklayabilmesi ve bu verileri kullanabilmesi için python un kütüphanelerinden olan pickle kütüphanesini import ediyoruz ve tabii ki yapay öğrenmeyi sağlayacak olmazsa olmaz random kütüphanesini dosyamıza import ediyoruz. Daha sonra ise ajanımız için bir sınıf tanımlıyoruz ve bu sınıfı tanımlarken geçen dönem nesneye dayalı programlama dersinde gördüğümüz constructor (Yapıcı) fonksiyonuna yer vererek işlemlere devam edeceğiz. Yapıcı fonksiyonlar, ajan sınıfından nesne oluşturduğumuzda yapıcı fonksiyon içerisine atanan değerlerde bir nesne oluşturur ve bu değerler ilk değer olarak atanmıştır yani default değerlerimizdir. Yine geçen dönemden öğrendiğimiz anlaşılabilirliği daha da arttırmak için Agent sınıfımızın UML diyagramını koyuyoruz:

\_\_init\_\_ Consturucter (Yapıcı) Fonksiyonu: Bu, ajan sınıfının başlatıcı fonksiyonudur ve çeşitli parametrelerle birlikte ajanın başlangıç değerlerini ayarlar.

game: TicTacToe oyununun dosyasını temsil eder.

player: Ajanın hangi oyuncu olduğunu belirler (ajana varsayılan olarak X koyduk).

episode: Ajanı eğitirken ki oynanacak oyun miktarıdır.

epsilon: Epsilon-greedy stratejisi için rastgelelik miktarıdır. Greedy dersimizde gördüğümüz (aç gözlü) olarak adlandırılan algoritmadır. Bu başlangıç değeri her 10 oyununun 9 unda randomize olarak atama yapar.

discount\_factor: Gelecek ödüllerin bugünkü değeri için indirim oranıdır. Ajanın daha mantıklı kararlar vererek ilerlemesini sağlar.

eps\_reduce\_factor: Epsilon değerinin her bir döngüde ne kadar azaltılacağını belirler. Q-Tablomuz zamanla dolacağından randomizelik azaltılır.

alpha: Öğrenme oranıdır. (Yani 1 – epsilon) Q-Tablodan oynamak diyebiliriz.

results: Oyun sonuçlarının tutulduğu X O D. D beraberliktir.

def save\_brain(self), def load\_brain(self): Bu fonksiyonlar pickle kütüphanemizin parametreleri kullanılarak ajanımızın Q-Tablosunu yani bir nevi beynini tutmamızı , bu dosyalara yazma ve okuma işlemlerini yapmamızı sağlayacak olan fonksiyonlardır.

def reward(self, player, move\_history, result): Bu fonksiyon ise ajanımızın ödüllendirileceği fonksiyondur. Ajanın kazanması durumunda +1, kaybetmesi durumunda -1 ve bu fonksiyonda Q değerleri ödüle göre güncellenir.

def use\_brain(self): Ajanın pickle dosyasını yani Q-Tablosuna göre en mantıklı kararı vererek hamle yapmasını sağlayan fonksiyondur.

def train\_brain\_x\_byRandom(self), def train\_brain\_o\_byRandom(self): Ajanımıza random hareketler yaptırarak sırasıyla X ve O için eğitiminin gerçekleştirildiği fonksiyonlardır.

def play\_against\_human(self, human\_player): Ajanımızın bir insana karşı oynamasını gerçekleştiren fonksiyondur. Ajan bu kısımda öğrendiği bilgileri kullanarak oynar. Oyunun sonucun ekrana yazdırıldığı kod parçası da bu kısımda yer almaktadır.

3. Ajanı Eğitme ve Oyunun Oynanacağı Dosya (Training.py)

Bu kısımda xox.py ve agent.py dosyalarımızı import ederek işlemlere devam edeceğiz. Bu dosyanın genel amacı bir ana fonksiyon içerisinde bir ajan sınıfının nesnesini üreterek kullanıcında input girişine göre X ya da O eğitimi ve buna göre de oyunun gerçekleşmesi.

Öncelikle kullanıcıdan alınan input değerine göre (X ya da O) hangi ajanın eğitileceği burada gerçekleşir. Ajan sınıfından bir nesne oluşturduğumuzda 2.python dosyası olan agent.py dosyasındaki default değerleri değiştirmemize olanak sağlanır. Bu kısımda yapıcı fonksiyon içerisindeki her değeri değiştirebiliriz ve bu da bizim oyunumuzu oradaki değerlere göre şekillenmesine olanak sağlar. Bizim kodlarımızda eğitim ve oyunun kullanıcıya karşı oynanması aynı komut ile gerçekleşiyor. Öncelikle bizim seçimimize göre ajan nesnesi oluşturup iç parametrelerine göre eğitiminin gerçekleşmesi sağlanıyor ve elindeki verileri pickle dosyası oluşturarak bir nevi beyin dediğimiz Q-Tablosuna kaydeder. episode sayısına göre ajan eğitimi gerçekleşir. Sonrasında kullanıcı ve eğitilmiş ajanımız arasında oyun başlar ve yüksek olasılıkla ajanımız kazanacaktır (Episode miktarına bağlıdır.).

Genel hatlarıyla özetleyecek olursak Q-Learning algoritması ile XOX oyununa tanımlanan ajanların eğitilmesiyle gerçekleştirdiğimiz kullanıcıya karşı oynanan bu proje ödevimizde sırasıyla xox.py, agent.py ve training.py dosyalarının birleşmesiyle gerçekleştirmiş olduk.

*Kaynaklar:*

[*https://medium.com/deep-learning-turkiye/q-learninge-giri%C5%9F-6742b3c5ed2b*](https://medium.com/deep-learning-turkiye/q-learninge-giri%C5%9F-6742b3c5ed2b) *,* [*https://medium.com/deep-learning-turkiye/python-i%CC%87le-q-learning-ef6413aa896e*](https://medium.com/deep-learning-turkiye/python-i%CC%87le-q-learning-ef6413aa896e) *,* [*https://www.geeksforgeeks.org/deep-q-learning/*](https://www.geeksforgeeks.org/deep-q-learning/) *, YouTube, ChatGPT.*