T.C.
YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

VİDEO OYUNLARINDA DERİN ÖĞRENME

Alihan ÖZTÜRK

YÜKSEK LİSANS TEZİ
İstatistik Anabilim Dalı
İstatistik Programı

Danışman
Prof. Dr. Fatma NOYAN TEKELİ

Temmuz, 2022

T.C.
YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

VİDEO OYUNLARINDA DERİN ÖĞRENME

Alihan ÖZTÜRK tarafından hazırlanan tez çalışması 06.07.2022 tarihinde aşağıdaki jüri tarafından Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü İstatistik Anabilim Dalı İstatistik Programı **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Prof. Dr. Fatma NOYAN TEKELİ
Yıldız Teknik Üniversitesi
Danışman

Jüri Üyeleri

Prof. Dr. Fatma NOYAN TEKELİ, Danışman
Yıldız Teknik Üniversitesi _____

Prof.Dr. Gülhayat GÖLBAŞI ŞİMŞEK, Üye
Yıldız Teknik Üniversitesi _____

Prof.Dr. Hakan YILDIRIM, Üye
Marmara Üniversitesi _____

Danışmanım Prof. Dr. Fatma NOYAN TEKELİ sorumluluğunda tarafımca hazırlanan Video Oyunlarında Derin Öğrenme başlıklı çalışmada veri toplama ve veri kullanımında gerekli yasal izinleri aldığımı, diğer kaynaklardan aldığım bilgileri ana metin ve referanslarda eksiksiz gösterdiğimi, araştırma verilerine ve sonuçlarına ilişkin çarpıtma ve/veya sahtecilik yapmadığımı, çalışmam süresince bilimsel araştırma ve etik ilkelerine uygun davrandığımı beyan ederim. Beyanımın aksinin ispatı halinde her türlü yasal sonucu kabul ederim.

Alihan ÖZTÜRK

İmza

Bu çalışma, Yıldız Teknik Üniversitesi Bilimsel Araştırma Proje Koordinatörlüğü' nün FYL-2021-4388 numaralı projesi ile desteklenmiştir.

Hocalarıma

TEŞEKKÜR

Aldığım ilk dersinden itibaren desteklerini esirgemeyen, bilgi ve tecrübelerinden faydalandığım danışman hocam sayın Prof. Dr. Fatma NOYAN TEKELİ'ye, bana çok kıymetli tecrübeler yaşatan ve RoboGör Şirketinde yanında çalışma fırsatı bulduğum hocam sayın Prof. Dr. Şeref Naci ENGİN'e, beni şaşırtan ve her zaman yeni bakış açıları sunan sevgili arkadaşlarım Fatma KURŞUN, Tolga KURTULUŞ, Hamit KAVAS, Hürcan Samet ÇAKIR'a yoluma ışık oldukları için teşekkürü borç bilirim.

Hayatım boyunca destek olan, beni bugünlere getiren ve varlığını arkamda hissettirip bana daima güç veren haklarını ödeyemeyeceğim sevgili annem ve babama minnettarım.

Ek olarak tez sürecimde beni destekleyen Yıldız Teknik Üniversitesi Bilimsel Araştırma Proje Koordinatörlüğü'ne katkılarından dolayı teşekkür ediyorum.

Alihan ÖZTÜRK

İÇİNDEKİLER

SİMGE LİSTESİ viii

KISALTMA LİSTESİ ix

ŞEKİL LİSTESİ x

TABLO LİSTESİ xii

ÖZET xiii

ABSTRACT xiv

1 GİRİŞ 1

1.1 Literatür Araştırması . . . 1
1.2 Tezin Amacı . . . 3
1.3 Hipotez . . . 3

2 Pekiştirmeli Öğrenme 5

2.1 Ödül . . . 6
2.2 Markov Zinciri . . . 7
2.3 Politika ve Değer Fonksiyonları . . . 7
2.4 Bellman Beklenti Denklemleri . . . 7
2.5 Değer Yineleme Metodu . . . 11
2.6 Monte Carlo Methodu . . . 11
2.7 Zamansal Fark Metotları . . . 13
2.7.1 SARSA . . . 14
2.7.2 Q Öğrenme . . . 15

3 Gözetimli Öğrenme 17

3.1 Yapay Sinir Ağları . . . 17
3.1.1 Algılayıcı . . . 17
3.1.2 Aktivasyon Fonksiyonu . . . 18
3.1.3 Çok Katmanlı Algılayıcı . . . 19
3.1.4 Geri yayılım . . . 19

3.1.5 Derin Öğrenme 21
3.1.6 Evrişimli Sinir Ağı (Convolutional Neural Network) 21
3.2 Önyargı-Varyans Değiş Tokuşu 23
3.3 Taklit Öğrenme 24

4 Derin Pekiştirmeli Öğrenme **27**
4.1 Derin Q Öğrenme 28
4.1.1 Düello Mimarisi 30
4.1.2 Öncelikli Deneyim 31

5 SONUÇ VE ÖNERİLER **32**
5.1 Grid World 32
5.2 Pacman 39
5.3 Lunar Lander 40
5.4 Doom 2 41

KAYNAKÇA **44**

TEZDEN ÜRETİLMİŞ YAYINLAR **46**

SİMGE LİSTESİ

a	Aksiyon
s	Anlık Durum
E	Beklenen Değer
$s^{'}$	Gelecek Durum
G_t	İndirimli Toplam Ödül ($\sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$)
r	Ödül
α	Öğrenme Oranı
γ	Ödül Oranı ($0 \leq \gamma \leq 1$)
q_*	Optimal Aksiyon Beklenen Değer Fonksiyonu
v_*	Optimal Durum Beklenen Değer Fonksiyonu
π	Politika
q_π	Politikanın Aksiyon Beklenen Değer Fonksiyonu
v_π	Politikanın Durum Beklenen Değer Fonksiyonu
$A(s)$	s Anındaki Mümkün Aksiyon Kümesi
$p(s', r\|s, a)$	s Durumunda ve a Aksiyonu Gerçekleştiği Bilindiğinde Bir Sonraki Durumun $s^{'}$ Olması ve r Ödül Alma Olasılığı
A_t	t Zamanda Aksiyon
S_t	t Zamanda Durum
R_t	t Zamanda Ödül
A	Tüm Aksiyonlar Kümesi
S	Tüm Durumlar Kümesi
R	Tüm Ödüller Kümesi

KISALTMA LİSTESİ

ANN	Artifical Neural Networks
CNN	Convolutional Neural Networks
DQN	Deep Q Network
DRL	Deep Reinforcement Learning
RL	Reinforcement Learning

ŞEKİL LİSTESİ

Şekil 2.1 Çevre 6
Şekil 2.2 Markov Zinciri 7
Şekil 2.3 Durum-Aksiyon İkilisi 8
Şekil 2.4 Aksiyon-Durum İkilisi 8
Şekil 2.5 Durum-Aksiyon İlişkisi 9
Şekil 2.6 Aksiyon-Durum İlişkisi 9
Şekil 3.1 Algılayıcı ve Bileşenleri 18
Şekil 3.2 Çok Katmanlı Algılayıcı 19
Şekil 3.3 Geri Yayılımın Mimari Üzerindeki yayılımı 20
Şekil 3.4 Kayıp Fonksiyonu Eğrisi 21
Şekil 3.5 D Birim Girdili ve C Birim Çıktılı Çok Katmanlı Algılayıcı 21
Şekil 3.6 Konvolüsyon 22
Şekil 3.7 Birden Fazla Konvolüsyon Çıktısı 22
Şekil 3.8 Önyargı-Varyans İlişkisi 23
Şekil 3.9 Kayıp Fonksiyonunun Eğitim ve Test Örnekleri İlişkisi 24
Şekil 3.10 Aynı Sona Ulaşan Politikalar 25
Şekil 3.11 Birden Fazla Sona Ulaşan Politikalar 25
Şekil 3.12 Eğitim ve Beklenti Aksiyonlarının İzi 25
Şekil 4.1 Durum Beklenen Değer Fonksiyon Yaklaşımı 27
Şekil 4.2 Aksiyon Beklenen Değer Fonksiyon Yaklaşımı 27
Şekil 4.3 DQN Mimarisi 30
Şekil 5.1 Bölüm 1 Grid World 33
Şekil 5.2 Deney 1 34
Şekil 5.3 Deney 2 34
Şekil 5.4 Deney 3 35
Şekil 5.5 Deney 4 35
Şekil 5.6 Deney 5 36
Şekil 5.7 Deney 6 37
Şekil 5.8 Bölüm 3 Discount Grid 37
Şekil 5.9 Pacman Small Grid 39
Şekil 5.10 Pacman Discount Grid 40

Şekil 5.11 Lunar Lander 41
Şekil 5.12 Doom 2 Kutu 42
Şekil 5.13 Doom 2 Can Toplama 43

TABLO LİSTESİ

Tablo 5.1 Farklı hedefler için oluşturulan deneyler 38
Tablo 5.2 Deneylerin parametre değerleri 39

ÖZET

Video Oyunlarında Derin Öğrenme

Alihan ÖZTÜRK

İstatistik Anabilim Dalı
Yüksek Lisans Tezi

Danışman: Prof. Dr. Fatma NOYAN TEKELİ

Pekiştirmeli öğrenme şu anda yapay zeka araştırmalarındaki en dinamik alanlardan biridir. Otonomi zamanla daha çok hayatımıza girmekte ve üzerine yoğunlaşılmaktadır. Tezde bahsedilen Pekiştirmeli Öğrenme algoritmaları insan gücüne ihtiyaç duymadan çevre ile etkileşime girerek, deneme-yanılma yoluyla ardışık seçimler yapmayı ve amaca optimum yoldan ulaşmayı öğrenir ayrıca Derin Öğrenme ile birlikte kullanıldığında genelleme yeteneği kazanır. Bu sayede karmaşık ve belirsiz dünya uygulamalarında da kullanılabilir.

Tezde Pekiştirmeli Öğrenme algoritmalarındaki gelişmeler incelenerek bir araya getirilmiştir. Pekiştirmeli öğrenmede durum değer ve aksiyon değer fonksiyonları için Derin Öğrenme fonksiyon yaklaşımı kullanılmıştır. Pekiştirmeli Öğrenme algoritmaları, fonksiyon yaklaşımı, fonksiyon yaklaşımında kullanılan mimariler ve hiper parametrelerin ajana etkisi araştırılmış ve testlerle ortaya konmuştur.

Çeşitli ödül fonksiyonları pekiştirmeli öğrenme algoritmalarında denenmiş ve çıkarımlar yapılmıştır. Ayrıca ödül fonksiyonu önerileri oluşturulmuştur.

Anahtar Kelimeler: Derin öğrenme, pekiştirmeli öğrenme, q öğrenme, derin q öğrenme, video oyunları

YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ABSTRACT

Deep Learning in Video Games

Alihan ÖZTÜRK

Department of Statistics
Master of Science Thesis

Supervisor: Prof. Fatma NOYAN TEKELİ

Reinforcement learning is currently one of the most dynamic areas in AI research. Autonomy comes into our lives more and more over time and we focus on it. Reinforcement Learning algorithms mentioned in the thesis learn to make sequential choices through trial and error and reach the goal in the optimum way by interacting with the environment without the need for human power, and also gain the ability to generalize when used with Deep Learning. In this way, it can also be used in complex and uncertain world applications.

In the thesis, the developments in Reinforcement Learning algorithms were examined and brought together. Deep Learning function approach is used for state value and action value functions in reinforcement learning. Reinforcement Learning algorithms, function approach, architectures used in the function approach and the effect of hyperparameters on the agent were investigated and demonstrated by tests.

Various reward functions have been tested in reinforcement learning algorithms and inferences have been made. In addition, reward function suggestions were created.

Keywords: Deep learning, reinforcement learning, q learning, deep q learning, video games

YILDIZ TECHNICAL UNIVERSITY
GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

1
GİRİŞ

1.1 Literatür Araştırması

Yapay zekanın farklı şekillerde tanımı yapılmaktadır. Bir dizi doğru kararları nasıl yapay zekaya öğretebiliriz? Yapay zekayı nasıl kendi kendine eğitebiliriz? Bu sorular için cevap Pekiştirmeli Öğrenme (Reinforcement Learning) olacaktır. Pekiştirmeli Öğrenme, ödül sinyalini kümülatif olarak maksimuma çıkarmak için hangi aksiyonlar alınacağını, eylemler ile öngörüleri nasıl eşleştireceğinizi öğrenmektir. Ajana(Agent) hangi eylemleri gerçekleştireceği bilgisi verilmez, bunun yerine hangi eylemleri deneyerek keşfetmesi gerekir. Deneme-yanılma, sıralı seçim ve kümülatif ödül tahmini, Pekiştirmeli Öğrenmenin en önemli ayırt edici özelliğidir.

Pekiştirmeli Öğrenme yeni bir alan değildir. Bu alandaki temel fikirlerin çoğu, son yetmiş yılda dinamik programlama ve optimum kontrol bağlamında tanıtılmıştır [1]. Bununla birlikte, Derin Öğrenmedeki atılımlar ve daha güçlü hesaplama kaynakları sayesinde son zamanlarda Derin Pekiştirmeli Öğrenmenin popülaritesi artmıştır.

Eski Çin oyunu Go, uzun zamandır yapay zeka araştırmacılarının uğraştığı bir oyundur. Diğer oyunlarda insan üstü performans gösteren yöntemler yakın zamana kadar Go oyununda usta seviyesine ulaşamamıştır. DeepMind takımı AlphaGo programını geliştirerek Avrupa şampiyonunu 5-0 skorla yenmiştir. Kısa bir süre sonra AlphaGo nun güncellenmiş versiyonu AlphaGo Zero ile 18 kez dünya şampiyona karşı 5-4 skorla galip gelip dünya çapında sesini duyurmuştur. AlphaGo Pekiştirmeli Öğrenme ile eğitilmesinin yanı sıra Gözetimli Öğrenme (Supervised Learning) ile usta oyuncuların oyunları ile katkı sağlanarak geliştirilmiştir. Sonrasında geliştirilen ve daha yüksek performans gösteren sürümü sadece Pekiştirmeli Öğrenme ile insan verisi veya rehberlik eden algoritmalar kullanılmadan eğitilmiştir [2, 3].

Pekiştirmeli Öğrenmein etkileyici uygulamalarından biri Tavla oyununda başarı göstermesidir. Tavla diğer uygulamalara göre çok az bilgi gerektirir ve şans oyunudur, yine de Derin Pekiştirmeli Öğrenme ajanı en büyük tavla ustalarına yakın, son derece

iyi oynamayı öğrenmiştir [4].

Google araştırma ekibi DeepMind, Atari oyunlarının otomatikleştirilebileceğini göstermek için Pekiştirmeli Öğrenme Algoritmasını Pong, Breakout, Space Invaders, Asteroids ve birçok oyunları içeren Atari 2600'ün üzerinde kullanmıştır. Bu oyunlar insan oyuncular hala için zorlayıcıdır. DeepMind, Pekiştirmeli Öğrenme'in metotlarından Derin Q Öğrenme (Deep Q Learning) her oyunda eğitilmiştir. Uygulama, 46 oyundan 29 unda insan performansı ya da daha yüksek performans gösteren çalışmalar gerçekleştirmiştir [5].

Open AI şirketinin çalışmalarından biri rubik küpü robot el ile çözmesidir. Sanal ortamda farklı koşullarda eğitilen Pekiştirmeli Öğrenme algoritmasını insan eli benzeri bir robotik kol üzerinde manipüle ederek rubik küpü çözmüşlerdir [6].

Literatürde ilginç örneklerden biri Hide and Seek oyunu olmuştur. Bu oyunda 2 takım vardır. Bir takım saklanırken diğer takım onları bulmaya çalışır. Saklanan takımın çevreye dağılmış nesneleri tutup yakalayan takımın yollarını tıkaması gerekmektedir. Yakalayan takımın ise kazanmak için bazı objeleri araç olarak kullanarak kendilerine yeni yollar üretmesi gerekmektedir. Ödül fonksiyonu sadece kazanan takıma +1 ödül kaybeden takıma -1 ödül seçilmiştir. Ajanlar oyunu oynadıkça karşılıklı takımlar birbirlerine karşı taktik geliştirirler. 6 adede kadar farklı stratejiler ortaya çıkmıştır. Her yeni strateji, bir sonraki aşamaya geçmeleri için yeni bir baskı yaratır. Başlangıçta yeni strateji aramak için hiçbir teşvik sağlamayan ödül fonksiyonu eğitim ilerledikçe farklı stratejiler ortaya çıkmasını sağlamıştır [7].

Ödül fonksiyonunun tasarlamak zordur. Araştırmacı ödül fonksiyonunu ajanı amaca doğru bir şekilde ulaşmayı teşvik edecek şekilde ayarlamalıdır. Ödül fonksiyonunu daha iyi belirlemek için DeepMind, ödül fonksiyonunu otomatik oluşturan bir algoritma önermiştir. Bu algoritma sayesinde gelecekteki ödülü daha yüksek yapacak adımlara erken ulaşmasını engelleyen yanlış tasarlanmış ödül fonksiyonu yerine kullanılabileceğini öne sürmüşlerdir. Oluşturdukları oyunda optimal çözüme ulaşamamışlardır ancak bu alanda adım kat etmiştir [8]. Başka bir çalışma ise ödül fonksiyonu yerine arama algoritması geliştirerek tüm politikaları inceleyip uygun kararın seçileceğini öne sürmüştür. Bazı video oyunlarında Derin Q Öğrenmeye göre daha yüksek performans göstermiştir [9].

Karmaşık çevrelerde sezgisel olarak optimal ödül fonksiyonu ayarlamak imkânsız olabilir. Karmaşıklık ödül fonksiyonu oluşturmayı zorlaştırır ve geribildirim süresini uzatır.

Farklı birçok kaynakta belirtildiği üzere optimum ödül fonksiyonu ayarlamak çok

zordur. Bununla birlikte ödül fonksiyonları basitleştikçe eğitim süreleri artmaktadır. Yanlış verilen ödül fonksiyonları politika arayışında aksamalara sebep olur ve yanıltıcı seçimler yaptırabilir. Sezgisel seçilen ödül fonksiyonları eğitimde yanlış yönlendirmelere ve Pekiştirmeli Öğrenme uygulamasının istenilen performansı gösterememesine sebep açabilmektedir. Gerçekleştirilen birçok uygulamada bazı ödül fonksiyonları dışına çıkılamamasının sebebi projelerin genellenememesidir. Ayrıca gerçek ortamdaki eğitimler çok daha maliyetli ve zaman alıcı olduğundan farklı ödül fonksiyonlarını denemek mümkün olmamaktadır.

1.2 Tezin Amacı

Literatürdeki birçok Pekiştirmeli Öğrenme örnekleri basit ve sezgisel ödül fonksiyonları ile eğitildiğinden seyrek ve aldatıcı geribildirimler ile mücadele eder. Ayrıca deneme-yanılma için farklı yolların araştırılması ve bu yolları keşfettikten sonra bazen önceki adımlara dönmesi gerekebilir bu yüzden doğru arama yöntemlerinin kullanılması önemlidir. Eğitimde tüm yolların keşfedilmesi alanın zorluklarından biri olmaya devam etmektedir.

Pekiştirmeli Öğrenmede gerçekleşmesini istediğimiz aktivitelere yüksek değerler vermek veya istenmeyen aktivitelere ceza sistemi uygulamak optimum çözüm olmayabilir. Bununla birlikte her durum için uygun ödül fonksiyonunu seçen kanıtlanmış bir yöntem yoktur. Kullanılan ödül fonksiyonları sezgiseldir. Literatür çalışmalarında her ne kadar bilgiye dayalı olsa da ödül fonksiyonu seçme sezgisel ve deneme yanılma yoluyla ilerlemiştir.

Bu tezde, çeşitli ödül fonksiyonları Pekiştirmeli Öğrenme algoritmalarında denenerek çıkarımlar yapılacak ve bir takım ödül fonksiyonları önerilecektir. Pekiştirmeli Öğrenme algoritmalarındaki gelişmeler incelenerek bir araya getirilecektir. Pekiştirmeli Öğrenme ve Derin Öğrenme konularının ajanın eğitiminde birlikte ve ayrı kullanılmaları incelenecektir. Pekiştirmeli Öğrenmede durum değer ve aksiyon değer fonksiyonları için Derin Öğrenme fonksiyon yaklaşımı kullanılacaktır. Pekiştirmeli Öğrenme algoritmaları, fonksiyon yaklaşımı, fonksiyon yaklaşımında kullanılan mimariler ve hiper parametrelerin video oyunları üzerinde ajana etkisi araştırılacak ve ortaya konacaktır.

1.3 Hipotez

Pekiştirmeli Öğrenme ödül fonksiyonuna ihtiyaç duyar. Ödül fonksiyonunun tam olarak istenilen görevi karakterize etmesi gerekir. Aksi taktirde ajan eğitim sırasında

amaçtan sapacaktır. Bazı Video oyunları ve masa oyunları Pekiştirmeli Öğrenmenin varsayımı ve ödül fonksiyonu için uygun bir ortamdır.

Literatürde yapılan çalışmalar genellikle halihazırda bulunan çevreden gelen skora eşitlenmiştir ya da amacın gerçekleştirilmesi durumu için +1 aksi durum için -1 değeri veren elle oluşturulan seyrek bir ödül fonksiyonu kullanılmıştır. Ancak ajan son durumda alacağı ödül ile bulunduğu durum arasındaki zamanın fazla olması sebebiyle ilişkilendirmeyebilir ve sorun oluşturabilir.

Zorlu ortamlarda ödül fonksiyonu tasarımına ilham sağlamak için yönlendirecek hem olumlu hem de olumsuz örnekler olmalıdır. Ajanın performansı toplam ödül değerine bağlıdır. Farklı bir ödül fonksiyonları ile eğitilen ajanlar toplam ödül değerine bakılarak karşılaştıramazlar. Çünkü toplam ödül değeri ödül fonksiyonuna göre ölçeklenmemiştir. Performans karşılaştırması yapabilmek için ajan eğitim sonunda incelenmeli ve çıkarımlar yapılmalıdır. Çeşitli ödül fonksiyonları Pekiştirmeli Öğrenme algoritmalarında denenerek çıkarımlar yapılacak ve bir takım ödül fonksiyonu önerilecektir.

Pekiştirmeli Öğrenme algoritmaları incelenecektir. Genelleme yapabilmesi için gerekli olan fonksiyon yaklaşımında Derin Öğrenme mimarileri kullanılacak ve çıktılar karşılaştırılacaktır. Ajanın politikasını belirlemede önemli olan noktalar video oyunları ile birlikte vurgulanacaktır.

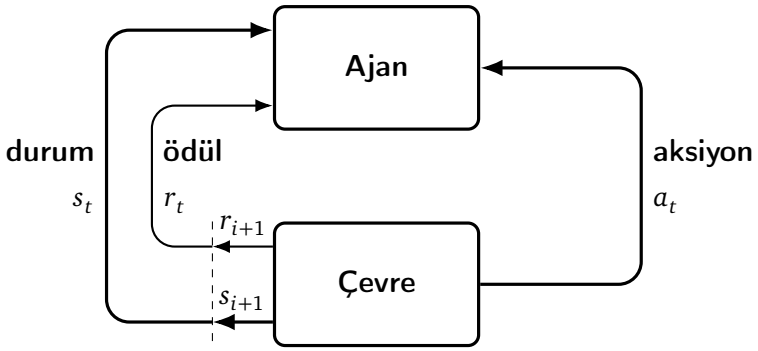
2
Pekiştirmeli Öğrenme

Çeşitli ardışık seçimler yaparak görevleri çözebilen akıllı sistemler geliştirmek uzun süredir devam eden bir zorluk olmuştur. Bu görevleri çözmeye yönelik bir yaklaşım, amaçlanan hedefe götürecek dijital sinyaller üreten bir fonksiyon oluşturmak ve bu görevin sonundaki sinyallerin toplamı maksimum olacak şekilde pekiştirmeli öğrenme aracısını doğru eylemleri seçmesi için eğitmektir [10].

Pekiştirmeli öğrenmede, ajanın eğitimi için belirlenen etkinliklere yüksek ödül veya yüksek ceza veren işlev uygun çözüm olmayabilir. Ajan, sıralı seçimler yaparak ödülünü maksimize etmeye çalışır. Ödül fonksiyonu oluşturulurken süreci düşünmeden hareket etmek, bu fonksiyondan yararlanılacağı için ajanın amaçtan sapmasına neden olacaktır.

Ayrıca tüm olasılıkların aranması pratik kullanım için çok verimsiz olacaktır. Öte yandan, satranç gibi sistemler ya da matematikte çözülmesi kolay olmayan hesaplamalar için gereklidir. Tam bir analiz çok uzun zaman aldığından bir miktar deneme-yanılma aksatılabilir. Dama oyunu yaklaşık 10^{40}, satranç oyunu 10^{120} adet durum içerir [11, 12] .

Pekiştirmeli öğrenmede ortam olası S durumlardan oluşur. Ajan belirli t zamanında s_t durumundadır. Ajan olası A aksiyonlardan olasılığı p(a_t, s_t) olan bir a_t seçer. Zaman geçtikçe, verdiği karara bağlı olarak ortam değişir ve pekiştirme ödülü olarak bilinen bir r_{t+1} alır [1]. Ödül sayısal değerlerdir. Aynı zamanda ajan s_{t+1} durumuna girer. Şekil 2.1'de ajanın çevre ile etkileşimi görselleştirilmiştir.



Şekil 2.1 Çevre

Tezde bahsi geçen pekiştirmeli öğrenme algoritmaları ayrık durum ve eylem alanı ortamlarında çalışır. Aksi takdirde, bu yöntemin kullanılabilmesi için sürekli durum veya eylem uzayının ayrık uzaya dönüştürülmesi gerekir. Ortamı ve eylemleri ayrık değişkenler yapmak niyeti olumsuz etkileyebilir. Gereksiz fazla veya az ayrıştırılmış ortamlar ve eylemler yüksek bellek boyutu oluşturarak eğitim sürecini artırırken, yetersiz dönüşüm yanlış veya eksik eğitilmesine neden olur.

2.1 Ödül

Literatürde ödül ve ceza değerleri aynı anlamda kullanılmaktadır. Richard Bellman ödül olarak, Lev Pontryagin ceza olarak notasyon kullanmıştır. Anlık ödül pozitif ya da negatif olmak zorunda değildir. Bazı durumlarda ödül 0 da olabilir. Toplam ödül aşağıdaki eşitlikte verilmiştir.

$$G_t = r_{t+1} + r_{t+2} + ... + r_T = \sum_{n=t+1}^{T} r_x \tag{2.1}$$

Toplam ödülü yukarıdaki eşitlikteki şekilde seçmek süre bakımından anlamlı değildir. Çünkü toplam ödüle baktığımızda bitişe t_∞ sürede ulaşmak ile t_1 sürede ulaşmak arasında hiç bir fark yoktur. Bu nedenle, indirim faktörü γ'ya ihtiyaç vardır. İndirim faktörü büyüdükçe zamanla üstel olarak toplam ödül azalacaktır, bu nedenle ajan hedefe mümkün olan en kısa sürede ulaşmak isteyecektir. Bunu sağlamak için aşağıdaki denklem kullanılmaktadır.

$$G_t = \sum_{k=0}^{T} \gamma^k r_{t+k+1} \tag{2.2}$$

Ödül γ parametresini kullanarak ağırlıklandırılır. Bu parametre değeri 0 ile 1 arasında

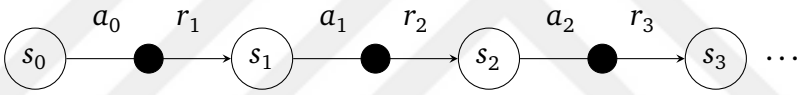
seçilir. 1'e yakın değerler, ajanın uzaktaki ödülleri daha çok önemsemesine 0'a yakın değerler uzaktaki ödüllere aldırış etmemesini sağlar.

2.2 Markov Zinciri

Bir Markov zinciri veya Markov süreci, her bir olayın olasılığının yalnızca önceki olayda elde edilen duruma bağlı olduğu olası olaylar dizisini tanımlayan stokastik bir modeldir. Gelecek, şimdiki zamanda geçmişten bağımsızdır. Markov zinciri varsayımının sağlandığı eşitlik aşağıda verilmiştir.

$$P(s_{t+1}|s_t) = P(s_{t+1}|s_t, a_{t-1}, s_{t-1}, ..., a_0, s_0) \tag{2.3}$$

Her bir anda sistem belirli bir olasılık dağılımına bağlı olarak kendi durumunundan başka bir duruma geçebilir ya da aynı durumda kalabilir. Durumda olan değişiklikler geçiş olarak bilinir ve çeşitli durum değişmeleriyle ilişkili olasılıklar da geçiş olasılıkları olarak adlandırılır. Şekil 2.2 de ajanın durum, aksiyon ve ödül sırası görselleştirilmiştir.



Şekil 2.2 Markov Zinciri

2.3 Politika ve Değer Fonksiyonları

Ajanın maksimum ödüle ulaşması için verdiği kararların ne kadar iyi olduğunu bilmesi gerekir. Politika π ajanın strateji haritasıdır. Bulunduğu durumun beklenen değeri $v_\pi(s)$ ve mümkün aksiyonların beklenen değeri $q_\pi(s,a)$ ajanın politikasını dikkate alarak hedefe ulaşmak için alacağı aksiyonların getireceği ödüle göre belirlenir.

$$v_\pi(s) = E_\pi(G_t|s_t) \tag{2.4}$$

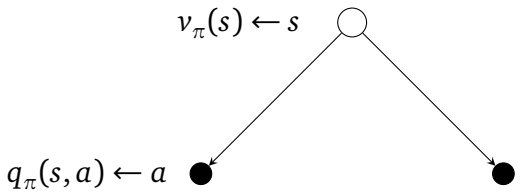
$$q_\pi(s,a) = E_\pi(G_t|s_t, a_t) \tag{2.5}$$

2.4 Bellman Beklenti Denklemleri

Her bir eylem için bir aksiyon beklenen değeri $q_\pi(s,a)$ vardır. Belirli bir durumun ne kadar iyi getirisi olduğunu söyleyen durum beklenen değeri $v_\pi(s)$, mümkün

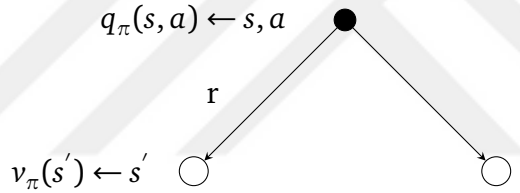
aksiyonların beklenti değerlerinin ortalamasıdır. Şekil 2.3'te olası 2 aksiyon vardır. Çevre Markov Zinciri Varsayımını sağlamıyorsa ajan aksiyon alana kadar anlık alacağı ödülü bilemez.

$$v_\pi(s) = \sum_{a \in A} p(a|s) q_\pi(s,a) \tag{2.6}$$



Şekil 2.3 Durum-Aksiyon İkilisi

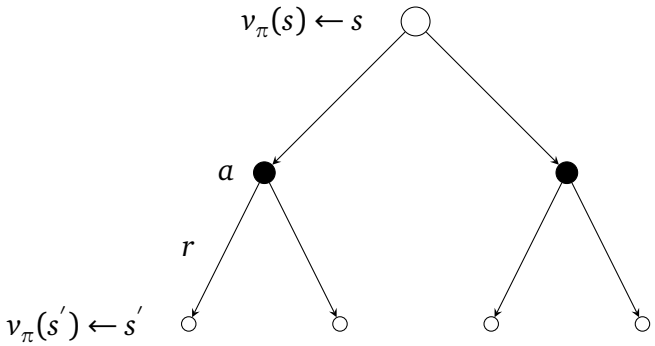
Aksiyon beklenen değeri aksiyon gerçekleştirildikten sonra alınan ödül ile yeni durumun beklenen değeri toplamıdır. Şekil 2.4'te p(a|s) olasılıkla aksiyon seçilmiş ve ajan s' durumuna geçmiştir.



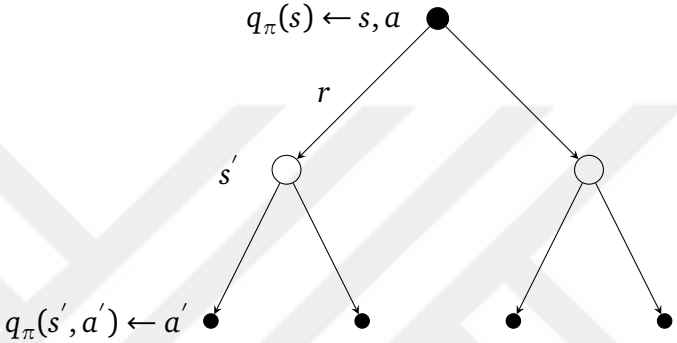
Şekil 2.4 Aksiyon-Durum İkilisi

$$q_\pi(s,a) = r + \gamma \sum_{s' \in S} p(s'|s,a) v_\pi(s') \tag{2.7}$$

Durum ve aksiyon beklenen değer fonksiyonları birbirlerine bağlıdır. Her durumun ve aksiyonun değeri bulunduğu konumdan bitişe kadar hesaplamak yerine sonraki beklenen değer fonksiyonları ile ilişkilendirilebilir. Bellman beklenti denklemleri bu ilişki kullanarak her değerin sıfırdan hesaplanması yerine bu fonksiyonları iteratif olarak hesaplayabilmektedir [1]. Şekil 2.5 ve 2.6 da Durum beklenen değeri ile aksiyon beklenen değeri arasındaki ilişki sergilenmiştir.



Şekil 2.5 Durum-Aksiyon İlişkisi



Şekil 2.6 Aksiyon-Durum İlişkisi

Aşağıdaki eşitliklerde değer fonksiyonlarının Bellman Denklemleri kullanılarak türetilmiş fonksiyonları bulunmaktadır. Bellman Beklenti Denklemlerinini kullanmak için Markov Zinciri Varsayımı sağlanmalıdır.

$$
\begin{aligned}
V_\pi(s) &= E_\pi\Big[G_t|s_t\Big] \\
&= E_\pi\Big[\sum_{k=0}^{\infty}\gamma^k r_{t+k+1}|s_t\Big] \\
&= E_\pi\Big[r_{t+1}+\gamma\sum_{k=0}^{\infty}\gamma^k r_{t+k+2}|s_t\Big] \\
&= \underbrace{\sum_a \pi(a|s)\sum_{s'}\sum_r p(s',r|s,a)}_{\text{Mümkün tüm aksiyonların ödül getirilerinin tolamı}} \\
&\quad \Big[r+\gamma\underbrace{E_\pi\Big[\sum_{k=0}^{\infty}\gamma^k R_{t+k+2}|S_{t+1}=s'\Big]}_{s_{t+1}\text{'in beklenen değeri}}\Big] \\
&= \sum_a \pi(a|s)\sum_{s'}\sum_r p(s',r|s,a)\Big[r+\gamma V_\pi(s')\Big]
\end{aligned}
\tag{2.8}
$$

$$
\begin{aligned}
q_\pi(s,a) &= E_\pi\Big[G_t|S_t=s,A_t=a\Big] \\
&= E_\pi\Big[\sum_{k=0}^{\infty}\gamma^k R_{t+k+1}|S_t=s,A_t=a\Big] \\
&= E_\pi\Big[R_{t+1}+\gamma\sum_{k=0}^{\infty}\gamma^k R_{t+k+2}|S_t=s,A_t=a\Big] \\
&= \sum_{s',r} p(s',r|s,a)\Big[r+\gamma E_\pi\Big[\sum_{k=0}^{\infty}\gamma^k R_{t+k+2}|S_{t+1}=s'\Big]\Big] \\
&= \sum_{s',r} p(s',r|s,a)\Big[r+\gamma V_\pi(s')\Big]
\end{aligned}
\tag{2.9}
$$

Optimal politika için durum ve aksiyon beklenen değer fonksiyonu aşağıda verilmiştir.

$$
v_*(s) = \max_{a\in A(s)}\sum_{s'\in S,r\in R} p(s',r|s,a)(r+\gamma v_*(s')) \tag{2.10}
$$

$$
q_*(s,a) = \sum_{s'\in S,r\in R} p(s',r|s,a)(r+\gamma\max_{a'\in A(s')} q_*(s',a')) \tag{2.11}
$$

2.5 Değer Yineleme Metodu

Değer yineleme metodu ortamın Markov Zinciri Varsayımına uyduğunu sayar. Yani geçiş olasılıkları ve durumların ödülleri önceden bilinir. Böylelikle ajanın ortamı keşfetmesine gerek yoktur ancak en iyi politikanın bulunabilmesi için durumların beklenen değerleri bilinmelidir. Değer yineleme metodu Bellman Beklenti Denklemlerini kullanarak iteratif bir şekilde durum beklenen değerlerini hesaplar. İterasyon durum beklenen değerlerinin güncellenme farkının istenilen hatadan küçük olmasına kadar devam eder.

Değer yineleme metodu oldukça hızlıdır. Ortamın geçiş olasılıkları bulunduğundan aksiyon beklenen değerlerinin hesaplanmasına ihtiyaç yoktur. Optimal politikada ajan bulunduğu durumdan geçiş olasılığı olan en yüksek beklenen değerli duruma geçiş yapar. Böylelikle ajan son duruma geldiğinde maksimum ödül toplamış olur.

Birçok ortam için değer yineleme metodunun varsayımları sağlanmaz. Ancak Pekiştirmeli Öğrenmenin mantığının en basit şekilde uygulandığı algoritmalardandır.

Algorithm 1: Değer Yineleme Metodu

Input: MDP, small positive number θ

Output: policy $\pi \approx \pi_*$

Initialize V arbitrarily (e.g., $V(s) = 0$ for all $s \in S^+$)

repeat

  $\Delta \leftarrow 0$

  for $s \in S$ **do**

    $v \leftarrow V(s)$

    $V(s) \leftarrow \max_{a \in A(s)} \sum_{s' \in S, r \in R} p(s', r|s, a)(r + \gamma V(s'))$

    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$

  end

until $\Delta < \theta$;

$\pi \leftarrow$ **Policy_Improvement**(MDP, V)

return π

2.6 Monte Carlo Methodu

Ajan ortamla ilk kez karşılaştığında herhangi bir politikası yoktur. Politika oluşturmaya yönelik deneysel yaklaşımlardan biri, n kez rastgele eylemlerde bulunmak ve durumların ve eylemlerin beklenen değerlerini tahmin etmeye çalışmaktır. Monte Carlo yönteminde, politika her denemede rastgele oluşturulur ve ajanın eğitimi tamamlanana kadar rastgeledir, başka bir deyişle eylemlerin olasılık dağılımı düzgün dağılımdır. Ajan, bu politikayı izleyerek çevre ile etkileşime girer. Her deney için

Bellman Beklenti Denklemleriyle durum-eylem çiftinin beklenen değerlerini hesaplar. Eğitimde aynı durumdaki eylem birden fazla seçilebilir. Bu durumlarda, ortalamaları veya ilk beklenen değerleri kullanılır. Eğitim istenilen tekrar sayısına ulaştığında biter [13, 14].

Çevrenin geçiş olasılıkları bulunduğunda Değer Yineleme Metoduna benzer olarak durum beklenti değerlerini hesaplamak yeterlidir. Geçiş olasılıkları bulunmadığı taktirde durum-aksiyonların beklenen değerlerinin eşleştirildiği Q tablo kullanılır. Ajan bulunduğu durumun aksiyonlarının değerlerini bu tablodan bakarak karşılaştırır. Eğitim sonunda politika her durumun maksimum aksiyonunu seçmektir.

Ajan deneme-yanılma yaparak politikasını iyileştirmeye çalışır. Bu durum basit ortamlarda işe yarayabilir, ancak karmaşık ortamlardaki keşif sürecini olumsuz yönde etkileyecektir. Bunun nedeni, ajan hareket halindeyken öğrenilen bilgileri kullanmadan sadece rastgele hareketlerle ortamı keşfetmesidir. Karmaşık ortamlarda belleğin kullanılması, ajanın yeni durumları keşfetmesine katkıda bulunacaktır.

Algorithm 2: Monte Carlo Metodu (*durum beklenen değerleri için*)

Input: policy π, positive integer $num_episodes$

Output: value function V ($\approx v_\pi$ if $num_episodes$ is large enough)

Initialize $N(s) = 0$ for all $s \in S$

Initialize $returns_sum(s) = 0$ for all $s \in S$

for $i \leftarrow 1$ ***to*** $num_episodes$ **do**

  Generate an episode $S_0, A_0, R_1, \ldots, S_T$ using π

  for $t \leftarrow 0$ ***to*** $T-1$ **do**

    if S_t *is a first visit (with return* G_t*)* **then**

      $N(S_t) \leftarrow N(S_t) + 1$

      $returns_sum(S_t) \leftarrow returns_sum(S_t) + G_t$

  end

end

$V(s) \leftarrow returns_sum(s)/N(s)$ for all $s \in S$

return V

Algorithm 3: Monte Carlo Metodu (*aksiyon beklenen değerleri için*)

Input: policy π, positive integer $num_episodes$

Output: value function Q ($\approx q_\pi$ if $num_episodes$ is large enough)

Initialize $N(s,a) = 0$ for all $s \in S, a \in A(s)$

Initialize $returns_sum(s,a) = 0$ for all $s \in S, a \in A(s)$

for $i \leftarrow 1$ ***to*** $num_episodes$ **do**

  Generate an episode $S_0, A_0, R_1, \ldots, S_T$ using π

  for $t \leftarrow 0$ ***to*** $T-1$ **do**

    if (S_t, A_t) *is a first visit (with return* G_t*)* **then**

      $N(S_t,A_t) \leftarrow N(S_t,A_t) + 1$

      $returns_sum(S_t,A_t) \leftarrow returns_sum(S_t,A_t) + G_t$

  end

end

$Q(s,a) \leftarrow returns_sum(s,a)/N(s,a)$ for all $s \in S$, $a \in A(s)$

return Q

2.7 Zamansal Fark Metotları

Zamansal fark yöntemlerinde öğrenme çevrimiçi olduğundan ajanın başarılı ya da başarısız olmasına gerek yoktur. Ajan ilk durumundan itibaren sonraki her aksiyonda öğrenir. SARSA ve Q Öğrenme Zamansal Fark Metotlarıdır. Aradaki fark SARSA bir sonraki durumda yapacağı harekete göre eylem değerini güncellerken, Q Öğrenme, bir sonraki aksiyonlardan maksimum beklenen değerine göre güncellemektedir. Her iki yöntem de ajanın bulunduğu durum ile sonraki durumu arasındaki farkı kullanarak durum-aksiyon beklenen değerlerini günceller. Güncelleme değerleri zamanla eğer varsa optimal politikaya yaklaşır.

Ajan çevre ile ilk karşılaştığında bilgisi yoktur. Başlangıçta politikası rastgele oluşturulur. Ajanın politikasını takip etmesine sömürü, politikasının dışına çıkmasına keşif denir. Ajan keşif yaptıkça optimal politikayı öğrenme olasılığı artar ya da aynı kalır. Yeni durumlar keşfetmek her zaman politikanın güncellenmesini sağlamaz. Bir durumun aksiyon değerlerinden maksimum olan güncelleme sonrasında maksimum değilse o zaman politika değişir. Güncelleme değeri öğrenme oranından etkilenir. Düşük öğrenme oranı beklenen değerlerin değişim aralığını daha stabil hale getirir ancak öğrenme süresi artar. Yüksek öğrenme oranı keşiflerin politikayı daha hızlı değiştirmesini sağlar ancak durum-aksiyon beklenen değerlerinin doğru değerlere yakınsamasını zorlaştırır [14].

Yeterli keşif yapmayan ajan optimal politikayı öğrenemez. Eğitim süreci aynı kalırken keşif sürecini arttırmak durum-aksiyon beklenen değerlerinin doğruluğunu düşürebilir

böylelikle ajanın yeterliliği azalır. Bu duruma keşif-sömürü takası denir.

Keşif-sömürü arasında seçim yapmak için 0 ile 1 değeri arasında olan ϵ değeri kullanılır. Genellikle eğitim başlangıcında bu değer 1 olarak belirlenir ve zamanla azalır. Ajan ϵ olasılıkla keşif yapar. Başlangıçta çevre hakkında bilgisi olmayan ajan yüksek olasılıkla keşif yaparak yakın çevreyi öğrenir. Zamanla azalan ϵ değeri keşiften çok sömürü yaparak ile mevcut politikasını sağlamlaştırmaya çalışır. Sağlamlaşan politika sayesinde ajan başlangıç durumundan daha uzaklara gidebilir. Çevrenin tam keşifi için epsilon değerini 0'a kadar azaltmak yerine 0'a yakın bir değere kadar azaltılabilir. Ajan başlangıçta sadece rastgele keşif yaparken zamanla bilgisini kullanarak keşif yapmış olur. Eğitimin sonlarında doğru ϵ değerinin 0'a yakın olması yeni yerler keşfetmek için yeterli bilgiye sahip olmasına ve bu bilgiyi kullanarak keşfetmesine olanak sağlar.

2.7.1 SARSA

Ajan bulunduğu s durumundan ϵ değeri ve politikasına göre a aksiyonu alır. Sonucunda r ödülünü alır ve ajan $s^{'}$ durumuna gelir. Önceki durumda olduğu gibi $a^{'}$ aksiyonu alır. Ajan s durumunun a aksiyon beklenti değerini güncellemek için $a^{'}$ aksiyonun beklenti değerini kullanır. Güncellemeler politikaya ve ϵ değerine bağlı olduğundan politikaya dayalı (on policy) bir algoritmadır. Algoritmanın ismi "s, a, r, $s^{'}$ $a^{'}$" sıralamasından gelir.

Algorithm 4: SARSA

Input: policy π, positive integer $num_episodes$, small positive fraction α, GLIE $\{\epsilon_i\}$

Output: value function Q ($\approx q_\pi$ if $num_episodes$ is large enough)

Initialize Q arbitrarily (e.g., $Q(s,a) = 0$ for all $s \in S$ and $a \in A(s)$, and $Q(terminal\text{-}state, \cdot) = 0$)

for $i \leftarrow 1$ ***to*** $num_episodes$ **do**

  $\epsilon \leftarrow \epsilon_i$

  Observe S_0

  Choose action A_0 using policy derived from Q (ϵ-greedy)

  $t \leftarrow 0$

  repeat

    Take action A_t and observe R_{t+1}, S_{t+1}

    Choose action A_{t+1} using policy derived from Q (ϵ-greedy)

    $Q(S_t,A_t) \leftarrow Q(S_t,A_t) + \alpha(R_{t+1} + \gamma Q(S_{t+1},A_{t+1}) - Q(S_t,A_t))$

    $t \leftarrow t+1$

  until S_t *is terminal*;

end

return Q

2.7.2 Q Öğrenme

Q Öğrenme, SARSA Max olarak da bilinir. Aralarındaki fark SARSA $s^{'}$ durumunda seçtiği $a^{'}$ aksiyonuna göre güncelleme yaparken Q Öğrenme $s^{'}$ durumunun maksimum aksiyon değeri ile güncelleme yapar. Bu durum politikasına uymadığından politika dışı(off policy) bir algoritmadır [15].

Algorithm 5: SARSA-Max (Q Öğrenme)

Input: policy π, positive integer $num_episodes$, small positive fraction α, GLIE $\{\epsilon_i\}$

Output: value function Q ($\approx q_\pi$ if $num_episodes$ is large enough)

Initialize Q arbitrarily (e.g., $Q(s,a) = 0$ for all $s \in S$ and $a \in A(s)$, and $Q(terminal\text{-}state, \cdot) = 0$)

for $i \leftarrow 1$ ***to*** $num_episodes$ **do**

  $\epsilon \leftarrow \epsilon_i$

  Observe S_0

  $t \leftarrow 0$

  repeat

    Choose action A_t using policy derived from Q (ϵ-greedy)

    Take action A_t and observe R_{t+1}, S_{t+1}

    $Q(S_t,A_t) \leftarrow Q(S_t,A_t) + \alpha(R_{t+1} + \gamma \max_a Q(S_{t+1},a) - Q(S_t,A_t))$

    $t \leftarrow t+1$

  until S_t *is terminal*;

end

return Q

3
Gözetimli Öğrenme

Bir Y değişkeni tahmin etmek amacıyla aralarında ilişki bulunan X değişken/değişkenler kullanılmak istensin. Gözetimli öğrenme X ve Y örnekleri arasındaki bağlantıları keşfederek parametrelerini düzenler [16].

$$\hat{Y} = g(X|\theta) \tag{3.1}$$

Burada g (·) gözetimli öğrenme tahmin edicisi, θ onun parametreleri ve $\hat{Y}$ tahmin değerdir. Tahmin edici eşleştirilmiş X ve Y ikililerine göre aralarındaki ilişkiyi öğrenerek θ parametrelerini, en yakın tahminler yapabilmek için optimize etmeye çalışır.

3.1 Yapay Sinir Ağları

Yapay sinir ağları genellikle karmaşık ve doğrusal olmayan ilişkileri keşfetmek için kullanılmaktadır.

3.1.1 Algılayıcı

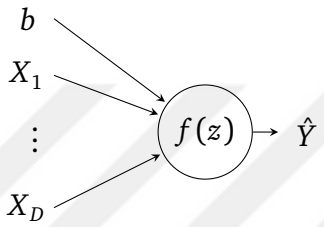
Yapay sinir ağının en küçük yapısı olan algılayıcılar girdileri parametreleriyle ağırlıklandırarak sabit katsayısı b ile birlikte toplarlar. Çıktı olarak sayısal bir değer elde edilir. Eğer X ve Y arasındaki ilişki doğrusal ise parametreler optimize şekilde ayarlandığında istenilen değerlere yakınsar.

$$\hat{Y} = w_1X_1 + w_2X_2 + w_3X_3 + ... + w_nX_n + b \tag{3.2}$$

3.1.2 Aktivasyon Fonksiyonu

Algılayıcı X ve Y arasındaki ilişki doğrusal olmadığı durumlarda kullanılan aktivasyon fonksiyonlarına ihtiyaç duyar. Aktivasyon fonksiyonu algılayıcı çıktısına uygulanır. Bu sayede doğrusal çıktılar doğrusal olmayan forma geçiş yapar. Geri yayılım ile parametrelerin güncellenebilmesi için türevlenebilir aktivasyon fonksiyonu kullanılmalıdır. Aktivasyon fonksiyonu f (·) ve algılayıcı çıktısı z ile Şekil 3.1'de gösterilmiştir.

$$\hat{Y} = f(z) \tag{3.3}$$



Şekil 3.1 Algılayıcı ve Bileşenleri

Birçok aktivasyon fonksiyonu vardır. İstenilen amaca göre bu fonksiyonlar seçilir. Aşağıda sırasıyla Sigmoid, Tanjant, Softmax, ReLU ve doğrusal aktivasyon fonksiyonları verilmiştir. Doğrusal aktivasyon fonksiyonu algılayıcı çıktısı ile aynıdır.

$$\sigma(z) = \frac{1}{1+e^{-z}} \tag{3.4}$$

$$tanh(z) = \frac{e^{z}-e^{-z}}{e^{z}+e^{-z}} = \frac{1-e^{-2z}}{1+e^{-2z}} \tag{3.5}$$

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad for\ i = 1, 2, \ldots, K \tag{3.6}$$

$$ReLU(z) = max(0, z) \tag{3.7}$$

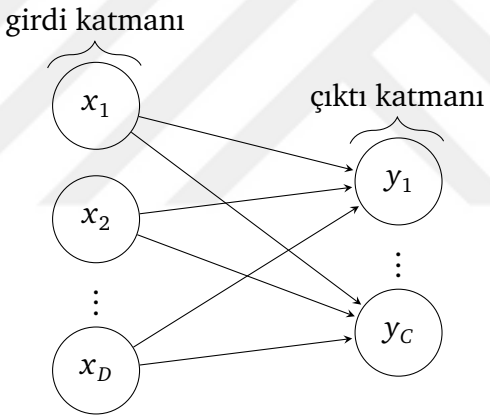
$$z = w_1X_1 + w_2X_2 + w_3X_3 + \ldots + w_nX_n + b \tag{3.8}$$

3.1.3 Çok Katmanlı Algılayıcı

X ve Y arasındaki karmaşık bağlantılar için tek bir algılayıcı yeterli olmayabilir. Bunun için farklı parametrelere sahip olan algılayıcı çıktıları aktivasyon fonksiyonundan geçirilerek başka algılayıcılara aktarılabilir. Bu sayede hem yeni öznitelikler keşfedilir hem de bu öznitelikler ile Y arasında bağ kurulabilir. Birden fazla algılayıcının art arda eklenmesi sonucunda çok katmanlı algılayıcı oluşur. Aynı girdiler ile sonraki katmanlara girdi sağlayan algılayıcılar yeni öznitelikler oluştururken art arda eklenen algılayıcılar yeni perspektifler keşfetmeye olanak sağlar.

Aktivasyon fonksiyonundan geçirilmeyen katmanlar ne kadar bir birine eklenirse eklensin tek bir algılayıcı olarak hesaplanabileceğinden tahmin ediciye katkı sağlamayacaktır.

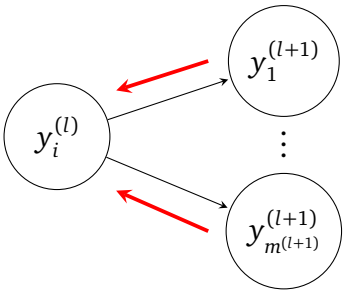
Yapay sinir ağları mimari konusunda esnektir. Birden fazla çıktı verilebilir. Bu sayede birden fazla boyutlu değerler tahmin edilebilir. Şekil 3.2'de D değişkenli ve C çıktılı çok katmanlı algılayıcı görselleştirilmiştir.



Şekil 3.2 Çok Katmanlı Algılayıcı

3.1.4 Geri yayılım

Özdeş parametrelere sahip algılayıcı ile doğrusal regresyon aynı işleve sahiptir ancak ağırlıklara ulaşma konusunda birbirinden ayrılırlar. Lineer regresyon ile parametreler tek bir hesaplamada bulunurken algılayıcıda bu durum sonsuza kadar bulunmayabilir. Bunun sebebi algılayıcının parametrelerini iteratif ve hiper parametrelere bağlı olarak aramasıdır. Doğru olmayan hiper parametreler arama süresini uzatabilir ya da tahmin edilmek istenen değerlere yakınsamamasına sebep olabilir. Şekil 3.3'te geri yayılımın yönü görselleştirilmiştir.



Şekil 3.3 Geri Yayılımın Mimari Üzerindeki yayılımı

.

Geri yayılma yapay sinir ağlarının öğrenme aşamasıdır. Tahmin edici parametreleri ile X değişken/değişkenlerini kullanarak tahminde bulunur. Tahmini ile Y değeri arasındaki uzaklığın parametrelere göre türevi alınarak parametrelerin ne yönde ve ne büyüklükte güncelleneceği tespit edilir. Buradaki uzaklık mutlak fark olarak düşünülmemelidir. Metrik kayıp fonksiyonu tarafından hesaplanır ve bağlama göre değişiklik gösterir. Başlangıçta optimal parametreler bilinmediğinden rastgele olarak atanır.

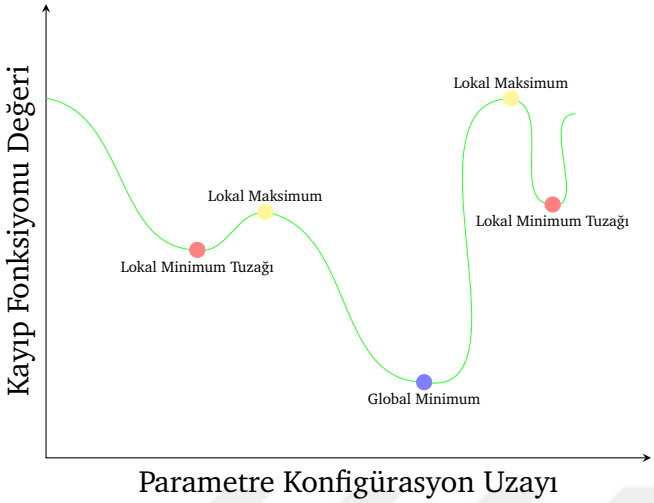
Kayıp fonksiyon değerini minimuma indirmek amacıyla uzaklığın parametrelere göre türevi, hiper parametre olan öğrenme oranı ile çarpılarak, parametreler negatif yönde güncellenir. Kayıp fonksiyonu J(w) azaldıkça tahminler Y değerlerine yaklaşır. Aşağıdaki eşitliklerde sırasıyla kayıp fonksiyonun ağırlıklara göre türevi ve öğrenme oranı α ile güncellenmesi verilmiştir.

$$\Delta_w J(w) = [\frac{\partial J(w)}{\partial w_1} \frac{\partial J(w)}{\partial w_2} \quad \frac{\partial J(W)}{\partial w_d}] \tag{3.9}$$

$$w \leftarrow w - \alpha \Delta_w J(w) \tag{3.10}$$

Sadece öğrenme parametresi kullanılarak güncelleme yöntemine Stokastik Gradyan İnişi denir. Her X-Y ikilisinden hesaplanan güncellemeler anında parametrelere etki eder. Bu yöntem karmaşık türev eğrileri ile mücadele edemez ve parametreler Şekil 3.4'te gösterilen lokal minimum tuzaklarına takılır. Yeni noktalar keşfetmek için ikililer güncellemeler birbirine eklenebilir. Böylelikle hareket yönünün kararsızlığının azalması ve bu tuzakların aşılmasına olanak sağlar. Stokastik Gradyan İnişi eğimlerde çok fazla salınım yapar. Minimum noktasında daha hızlı gidebilmesi için momentum metodu kullanılır. Geçmiş güncellemeler hesaba katılarak bir sonraki güncelleme miktarı belirlenir. Ek olarak adaptif öğrenme sağlamak, lokal minimum noktalarına takılmamak ve öğrenmenin hızlı olması amacıyla birden fazla ikililer güncellemeye

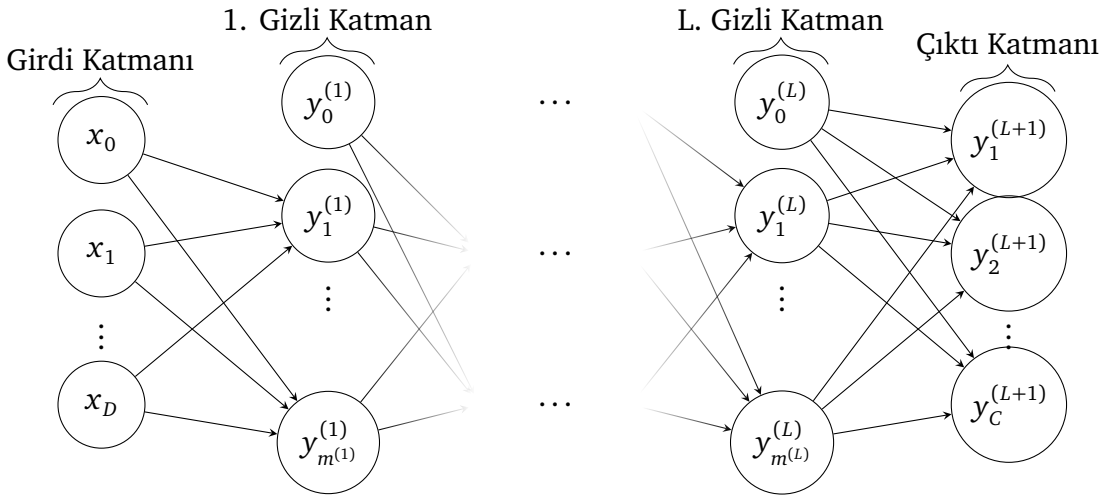
eklenir ve Adagrad, RMSprop, AdaDelta, Adam gibi optimize etme yöntemleri geliştirilmiştir.



Şekil 3.4 Kayıp Fonksiyonu Eğrisi

3.1.5 Derin Öğrenme

Derin öğrenme mimaride katman kullanımının 2 ya da daha fazla olduğu anlamına gelir. Doğru hiper parametre seçimleri ve uygun koşullar altında evrensel fonksiyon yaklaşımı olabilir. Şekil 3.5'te Yapay Sinir Ağlarından oluşan Derin Öğrenme mimarisi görselleştirilmiştir.



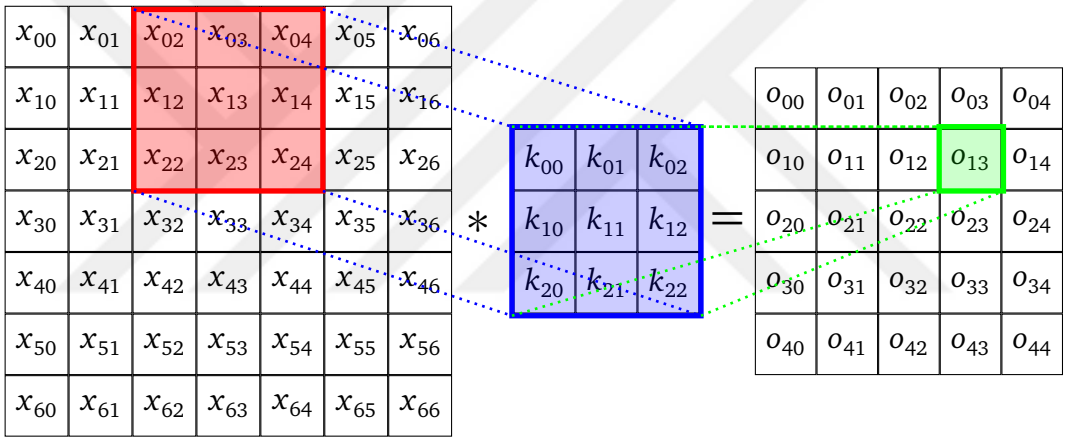
Şekil 3.5 D Birim Girdili ve C Birim Çıktılı Çok Katmanlı Algılayıcı

3.1.6 Evrişimli Sinir Ağı (Convolutional Neural Network)

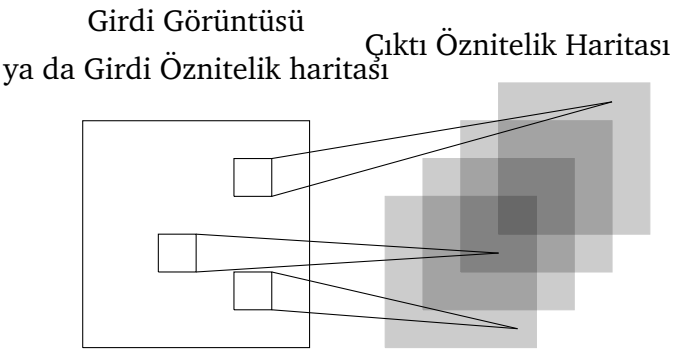
Yapay sinir ağlarında her girdi birlikte değerlendirilir. Ancak X değerleri içerisinde her zaman birbiriyle ilişkisi olmayabilir. Evrimsel sinir ağları değerleri belli bir

pencere içerisinde inceler. Böylelikle çıkarılan öznitelikler daha saflaşır ve ayırt etmek kolaylaşır.

Evrişimli sinir ağı katmanlarında parametreler kernelde bulunur. Kernel eksenlerine göre simetriği alındıktan sonra değişkenler üzerinde gezdirilir. Kernel üzerindeki ağırlıklar her yeni yüzeyde paylaşılır. Üst üste gelen değişkenler ile ağırlıklar çarpılarak toplanır. Bu işleme konvolüsyon denir. Mimariye göre aktivasyon fonksiyonundan geçirilir. Paylaşılan ağırlıklar sayesinde birbirinden bağımsız aynı türde değişkenler oluşturulur. Aynı değişkene birden fazla kernel ile konvolüsyon uygulanabilir. Art arda uygulanan konvolüsyonlar ile yeni perspektifler keşfedilir. Şekil 3.6'da 6x6 boyutunda bir matrisin konvolüsyon işlemi görselleştirilmiştir. Konvolüsyon sonrasında çıktı dolgulama (padding) kullanılmadığı için kernelin boyutundan kaynaklı küçülmüştür. Şekil 3.7'de farklı kerneller ile oluşturulan konvolüsyon çıktıları görselleştirilmiştir.



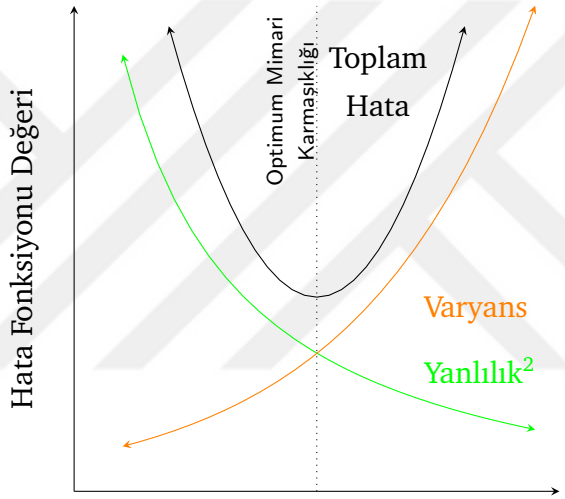
Şekil 3.6 Konvolüsyon



Şekil 3.7 Birden Fazla Konvolüsyon Çıktısı

3.2 Önyargı-Varyans Değiş Tokuşu

Önyargı, tahmin edicinin amacına ulaşmasında yetersiz kalmasından oluşan hatadır. Yüksek önyargı, tahmin edicinin öznitelikler ile tahmin edilmek istenen değer arasındaki ilişkileri kaçırmasına ve yetersiz kalmasına denir. Varyans, tahmin edicinin hassasiyetten kaynaklı tahmin edilmek istenen değerlerdeki rastgele gürültüye kadar öğrenmesinden, tahmin edici parametrelerinin aşırı eğilmesinden oluşan hatadır. Tahmin edicinin eğitim sürecinde gördüğü rasgele gürültüyü öğrenmesi yüksek varyans oluşabilir. Yüksek varyanslı bir tahmin edici parametrelerini eğitim örneklerine fazla uydurduğu için ilk defa göreceği örneklerde hatası yüksek olacaktır [17]. Önyargı-varyans değiş tokuşu tahmin edicilerde eğitim setlerinin ötesine genelleme yapmasını engelleyen bu iki hata kaynağını aynı anda en aza indirmeye çalışmaktaki ikilemdir. Önyargı-varyans ilişki grafiği Şekil 3.8'de verilmiştir.

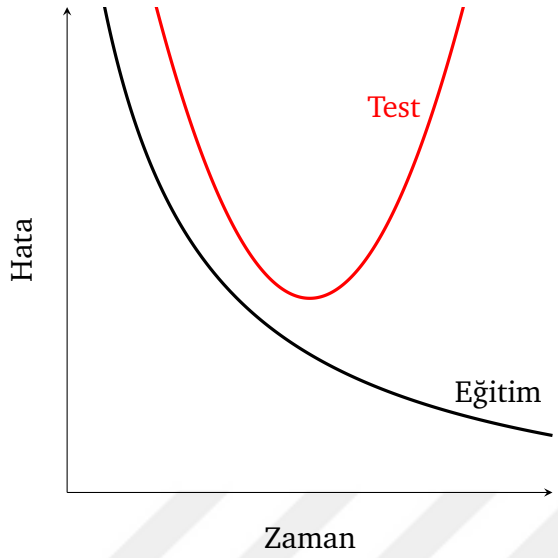


Şekil 3.8 Önyargı-Varyans İlişkisi

Gözetimli öğrenmede eğitim örnekleri üzerinde iyi tahminler yapmak başarı performansı için yeterli ölçüt değildir. Tahmin edici daha önce görmediği örnekler üzerinde yakın tahminlerde bulunmalıdır.

Yanlılık ve varyans hatası derin öğrenmede mimariye, öğrenme oranına, optimize etme yöntemine ve eğitim örneklerine bağlı olarak değişkenlik gösterecektir. Bu hiper parametrelerin belirlenme yöntemi için kesin bir yöntem olmasa da hata grafiklerine bakılarak üzerinde çıkarımlar yapılabilir. Bazı özniteliklerin keşfedilmesi için benzer ya da aynı örnekler eğitim sürecine dahil edilebilir. Ancak bu şekilde uzayan eğitim süresi tahmin edicinin hiç görmediği örnekler üzerinde hatasını arttıracaktır. Performans durumunu kontrol etmek için eğitim süresi boyunca model hiç görmediği veri seti ile birlikte test edilir. Eğitim test hatasının incelenmesi ile yönetilir. Şekil

3.9'da tahmin edicinin zamana karşı eğitim ve test örnekleri üzerindeki eğilimi verilmiştir.

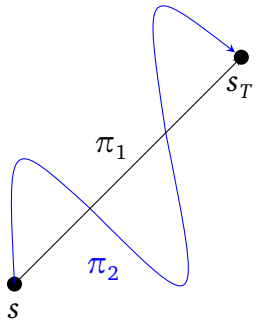


Şekil 3.9 Kayıp Fonksiyonunun Eğitim ve Test Örnekleri İlişkisi

3.3 Taklit Öğrenme

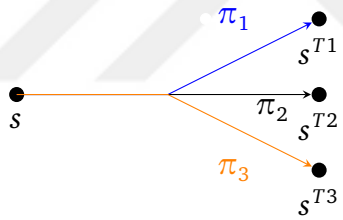
Taklit Öğrenmede uzman tarafından kontrol edilen eylemler kayıt altına alınır ve Gözetimli Öğrenme yöntemleri kullanılarak durum-eylem ikililerine göre eğitilir [18, 19]. Ajanın bulunduğu durumda aksiyon seçimi için sınıflama gerçekleştirir. Ancak eğitim sonunda ajanın performansı, uzmanı taklit etme yeteneğine göre değil toplam ödüle göre ölçeklendirilir. Bunun nedeni birden fazla seçim yaparak sonuca ulaşmasıdır. Sonuç olarak ilk bakışta çekici görünen bu yöntem aslında çözümü oldukça zor bir problemdir.

Şekil 3.10'da gösterildiği üzere amaca giden birden fazla yol olabilir. Gözetimli öğrenme yöntemi kullanılacağından birden fazla yol seçimi modeli başarısız göstermeye sebep olacaktır. Tahmin edicinin doğrulama aşamasında uzmanın aksiyonu ile kendi aksiyonu arasındaki farkı minimuma indirmeye çalışmasıdır. Tahmin edicinin seçtiği aksiyon amaca gitmesine sebep olabilir ancak uzmandan farklı seçtiği için cezalandırılır.



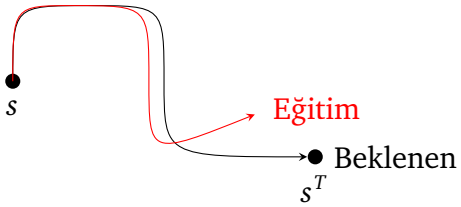
Şekil 3.10 Aynı Sona Ulaşan Politikalar

Şekil 3.11'de gösterildiği üzere hedef birden fazla olabilir. Ancak ajan sadece uzmanı örnek alarak öğrenmeye çalışır. Tek bir hedef için birden fazla yol olması problemi ile aynı problem yaşanacaktır. Buna ek olarak tahmin edicinin aksiyon seçimi uzmandan farklı ancak doğru olsa bile sonraki seçimler bu aksiyona bağlı olarak değişeceğinden ajan bu durumlarla ilk defa karşılaşacak ve doğru seçimler yapamayacaktır. Uzman her ne kadar yetkin olursa olsun insan hatası her zaman bulunacaktır. Eğitim sırasında yapılan seçimler ne kadar doğru olursa olsun önceki seçimlerden farklı olması durumunda tahmin edicinin öğrenmesini zorlaştıracaktır.



Şekil 3.11 Birden Fazla Sona Ulaşan Politikalar

Ajan hata yaptığı andan itibaren sonraki seçimleride etkilenecek, hata değeri katlanarak artacak ve amaçtan sapacak adeta kelebek etkisine dönüşecektir. Şekil 3.12'de ajanın hatasının etkisi gösterilmiştir.



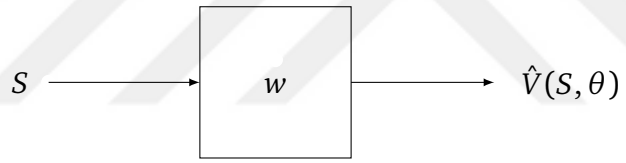
Şekil 3.12 Eğitim ve Beklenti Aksiyonlarının İzi

Gözetimli Öğrenmede yöntemleri karmaşıklığı ve uzman durum-aksiyon ikilileri üzerine fazla eğilebilir. Yanlış öznitelikler ile ilişkilendirilen aksiyonlar eğitim

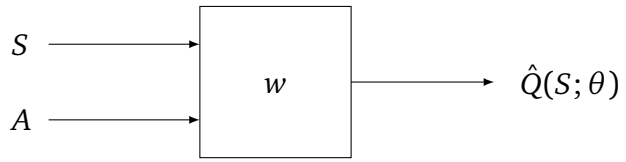
sonrasında performansının yetersiz kalmasına sebep olur. Gözetimli öğrenmenin en büyük zorluklarından olan bu durumun anlaşılması da oldukça güçtür.

4
Derin Pekiştirmeli Öğrenme

Pekiştirmeli Öğrenme algoritmaları durum ve aksiyonların beklenti değerlerini hesaplayıp hafızada tutarlar. Ajan bulunduğu durumda karar vermek için hafızasını kullanır. Durum değer ve aksiyon değer fonksiyonu işlevi, bir hafıza yerine Gözetimli Öğrenme yöntemleri kullanarak fonksiyon yaklaşımı olarak temsil edilebilir. Şekil 4.1 ve 4.2 de fonksiyon yaklaşımı görselleştirilmiştir. Özelliklerin doğrusal kombinasyonu [20], karar ağaçları, en yakın komşu gibi birçok gözetimli öğrenme yöntemi kullanılabilir. Bu yöntemlerin yorumlanması kolay olabilir ancak karmaşık ortamlarda fonksiyon yaklaşımı için yeterli olmayabilirler.



Şekil 4.1 Durum Beklenen Değer Fonksiyon Yaklaşımı



Şekil 4.2 Aksiyon Beklenen Değer Fonksiyon Yaklaşımı

Pekiştirmeli öğrenme algoritmaları durum ve aksiyon beklenen değerlerini hesaplamak için sadece ödülleri kullanır. Ajan aksiyon beklenen değerine bulunduğu duruma bakarak ulaşır ve hafızasında olmayan bir durumla karşılaştığında aksiyon alamaz. Bunun sebebi beklenti değerlerinin hesaplanmaması değil o durumun hafızada tanımlı olmamasıdır. Derin Pekiştirmeli Öğrenme durum ve aksiyon beklenen değerini tahmin etmek için durum özniteliklerini kullanır. Durumun öznitelikleri yapay sinir ağları ile çıkartılır. Hafıza bilgisi kullanmak yerine ortamın özniteliklerini işin içine katarak ajanı eğitmek, genelleme yapabilmesine ve herhangi bir durumda çıkarımda bulunmasına imkan sağlayacaktır [21].

4.1 Derin Q Öğrenme

SARSA ve Q Öğrenme yöntemlerini karmaşık ortamlarda kullanmak, optimum politikaya ulaşmak için çok zaman alacaktır. Bunun nedeni, ajanın her durum için durum ve aksiyon beklenen değerlerini ezberlemesidir. Uzun eğitim süresi ve genelleyememe sorunu, ajan için yetersiz örneklem olarak hissedilmektedir [22].

Bu sorunu çözmek için fonksiyon yaklaşımı kullanılır. Her durumu ve eylemi bellekte tutmak yerine, fonksiyon yaklaşımı yardımıyla genelleştirilecek olan durum değer ve aksiyon değer fonksiyonları genelleme yapma yeteneğine sahip olur. Böylece durum ve aksiyon uzayı ne kadar büyük olursa olsun değerler hafızada tutulmayacağından problem teşkil etmeyecektir.

Derin Q Öğrenme için kullanılacak derin öğrenme mimarisinin kayıp fonksiyonu, kayıp fonksiyonu, kayıp fonksiyonunun ağırlıklara göre türevi ve parametre güncellemesi aşağıdaki eşitliklerde sırasıyla verilmiştir.

$$J(w) = E_{\pi}[((q_{\pi}(S,A) - \hat{q}(S,A,w))^2] \tag{4.1}$$

$$\Delta_w J(w) = -2(q_{\pi}(S,A) - \hat{q}(S,A,w))\Delta_w \hat{q}(S,A,w) \tag{4.2}$$

$$\begin{aligned} \Delta_w &= -\alpha \tfrac{1}{2} \Delta_w J(w) \\ &= \alpha(q_{\pi}(S,A) - \hat{q}(S,A,w))\Delta_w \hat{q}(S,A,w) \\ &= \alpha(R + \gamma max_a \hat{q}(S^{'},a,w) - \hat{q}(S,A,w))\Delta_w \hat{q}(S,A,w) \end{aligned} \tag{4.3}$$

Zamansal fark metodu kullanılarak oluşturulan bu eşitlikler matematiksel olarak eğitimine katkı sağlamaz. Mevcut durum ve sonraki durum için tahminler yapan fonksiyon yaklaşımının parametreleri aynıdır. Her parametre güncellemesinde aralarındaki fark her zaman aynı kalacağından, fonksiyon yaklaşımında bir sonraki adım için kullanılan parametreler değerleri doğru temsil etmeli ve mevcut durum için kullanılan fonksiyon yaklaşımı parametrelerinden farklı olmalıdır. Bunu sağlamak için öncelikle aynı parametrelere sahip modeller oluşturulur. Hedef olarak adlandırılan sonraki durum için tahmin fonksiyonu parametreleri, yerel olarak adlandırılan anlık durum için tahmin fonksiyonu parametreleri ile τ parametresi kadar güncellenir. Her çevrimdeki hataya göre yerel model güncellenirken, hedef model yerel parametrelere göre güncellenir [5, 23, 24].

Sırayla çevreden gelen geri bildirimler yüksek oranda ilişkili olduğundan, fonksiyon yaklaşımındaki öğrenme parametreleri yanlış temsil edilecektir. Ajandan geri bildirimleri durum, aksiyon, ödül, yeni durum olarak ileride rastgele kullanmak üzere deneyim havuzunda depolar. Deneyim havuzundan geri bildirimler rastgele seçilir ve yerel model parametreleri güncellenir. Böylelikle geri bildirimler aynı dağılımdan ve bağımsız olduğundan aşırı eğilme durumundan kurtulur.

Algorithm 6: Derin Q Öğrenme

Input: positive integer $num_episodes$, small positive fraction γ, small positive fraction α, positive integer C, less than 1 positive float τ, GLIE $\{\epsilon_i\}$

Output: action value function weights θ

Initialize Q_{local} action value function with random weights θ

Initialize $Q_{targets}$ action value function with local action value weights θ^-

Initialize D experience replay memory

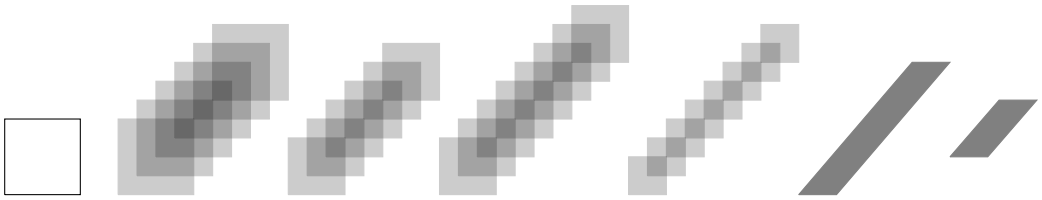
for $i \leftarrow 1$ ***to*** $num_episodes$ **do**

- $\epsilon \leftarrow \epsilon_i$
- Observe S_0
- $t \leftarrow 0$
- **repeat**
  - Choose action a_t using $argmax_a Q_{local}(s_t;\theta)$ (ϵ-greedy)
  - Take action a_t and observe r_{t+1}, s_{t+1}
  - Store $(s_t, a_t, r_{t+1}, s_{t+1})$
  - Sample minibatch from D
  - targets= $reward + \gamma Q_{target}(S_{Dt+1};\theta^-)$
  - expected $= Q_{local}(S_{Dt};\theta)$
  - perform a α size gradient descent step on (targets-expected)^2 with respect to network parameters θ
  - every C step $\theta^- = (1-\tau)\theta^- + \tau\theta$
  - $t \leftarrow t+1$
- **until** S_t *is terminal*;

end

return θ

Şekil 4.3'te DQN mimarisi görselleştirilmiştir. Mimari girdi olarak piksel değerlerini alıp çıktı olarak aksiyon beklenen değerlerini vermektedir.



Şekil 4.3 DQN Mimarisi

4.1.1 Düello Mimarisi

Düello mimarisi algoritmada değişiklik yapmadan derin öğrenme mimarisinde değişiklik yaparak eylemler arasında genelleştirmeyi amaçlar. Mimari sadece aksiyon beklenen değerlerini tahmin etmek yerine durum beklenen ve aksiyon beklenen değerlerinden yola çıkar [25]. Bellman beklenti denklemlerinde durum beklenen değeri aksiyon beklenen değerlerinin ortalamasıdır. Buradan hareketle avantaj değerleri aksiyon beklenen değerlerin ile durum beklenen değerinin arasındaki farktır. Aksiyon beklenti değerlerinin ortalama farkları olan avantaj beklenti değerleri 0 etrafında dağılır ve ortalaması 0'dır.

$$A_\pi(s_t, a_t) = Q_\pi(s_t, a_t) - V_\pi(s_t) \tag{4.4}$$

$$E\Big[A_\pi(s_t, a_t)\Big] = 0 \tag{4.5}$$

Avantaj değerleri aksiyonların bulunan durum için hangisinin en iyi olduğunu gösterir. Mimarinin amacı her durum için aksiyonların etkisini öğrenmek zorunda kalmadan hangi durumların avantajlı olduğunu öğrenmektir. Böylelikle mimari çok daha hızı eğitilebilecektir.

Derin Q öğrenme algoritması yapay sinir ağı fonksiyon yaklaşımı ile aksiyon beklenen değerlerini tahmin eder. Mimari algoritmada değişiklik yapılmaması için durum beklenti değerlerini ve avantaj değerlerini tahmin etmelidir. Ancak 4.4'teki eşitlikte eylem beklenen değeri bilindiğinde V(s) ve A(s,a) için sonsuz çözüm vardır. Düello mimarisi tek bir çözüm olması için avantaj değerleri ile ortalamasını da hesaba katarak tek çözüme indirmeye ve tahmin etmesine zorlar.

$$Q(s,a) = V(s) + (A(s,a) - \frac{1}{N_{action}} \sum (A(s,a)) \tag{4.6}$$

4.1.2 Öncelikli Deneyim

Öncelikli deneyimde temel amaç zamansal fark hatasının deneyimlerin yeniden eğitime katılma olasılığını arttırmaktır. Bu örnekleme yöntemi Derin Pekiştirmeli Öğrenme mimarisinin yapacağı hataya göre deneyim seçilimlerini düzenleyerek öğrenme hızını arttırır [26]. Bu fikirden yola çıkılarak farklı öncelikli deneyim örnekleme yöntemleri de geliştirilmiştir.

Lokal minimum tuzağına yakalanan bir mimari ϵ değeri azaldıkça öğrendiği politikaya fazla eğilecek ve ajanın belli aksiyonları gerçekleştirmesini azaltacaktır. Deneyimler benzer olmaya başladıkça genelleme azalarak keşif yerini rastgele aksiyonlar bırakacaktır. Bunu önlemek için deneyim havuzu aksiyonlara göre ya da durumlara göre ayrılır [27]. Eğitim sürecinde her havuzdan aynı örneklem sayısında deneyim seçilerek ajan eğitilir. Mimari ajana aynı politikaya devam etmeyi zorlasa da öncelikli deneyim kullanılarak bu durumdan kurtulmaya çalışılır.

Lokal minimum tuzağına yakalanan bir mimari ϵ değeri azaldıkça öğrendiği politikaya fazla eğilecek ve ajanın belli aksiyonları gerçekleştirmesini azaltacaktır. Deneyimler benzer olmaya başladıkça genelleme azalarak keşif yerini sömürüye bırakacaktır. Keşif asla bitmese dahi lokal minimum tuzağına yakalanan ajanın rastgele aksiyonları keşifte devamlılığı olmadığında katkı sağlamayacaktır. Bunu önlemek için deneyim havuzu aksiyonlara göre ya da durumlara göre ayrılır [27]. Eğitim sürecinde her havuzdan aynı örneklem sayısında deneyim seçilerek ajan eğitilir. Mimari ajana aynı politikaya devam etmeyi zorlasa da öncelikli deneyim kullanılarak bu durumdan kurtulmaya çalışılır.

5
SONUÇ VE ÖNERİLER

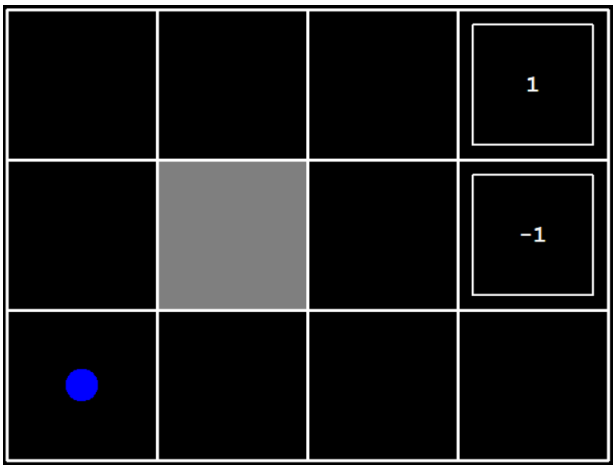
Çeşitli video oyunlarında Pekiştirmeli Öğrenme ve Derin Öğrenme kullanılarak amaçlanan karşılaştırmalar yapılmıştır.

Ödül fonksiyonu önerisi olarak istatistiksel dağılımlar uygun görülmüştür. İstatistiksel dağılımlar gerçek hayatta birçok olayı karakterize etmektedir. Ajanın durum bilgilerine ulaşıldığında bu dağılımlar ile bulunduğu durum değerlendirilebilir. Örnek olarak ajanın ve gitmek istediği yerin konum bilgisine sahip olduğumuzu varsayalım. Hedef ile ajan arasındaki mesafe ödül olarak düşünülebilir ve buradan yola çıkarak mesafe azaldıkça ödülün artacağı bir fonksiyon düşünülebilir. Ancak Ajan farklı şekillerde gitmek istediğinde bu ödül fonksiyonu yeterli olmayacaktır. İstatistiksel dağılım olan Çok Değişkenli Normal Dağılım ödül fonksiyonu olarak kullanıldığında ortalama ve standart sapma değerleri kullanılarak farklı şekillerde ödülleri ayarlayabilecektir. Ya da ajanın canının önemli olduğu durumlarda Poisson Dağılımı kullanılarak ödül fonksiyonu istenilen farklı şekilde özelleştirilebilecektir. Bu sayede geliştirilen ödül fonksiyonu dağılım parametreleri kullanılarak özelleştirilebilecektir.

5.1 Grid World

Değer yineleme algoritması Grid World [28] ortamında 3 farklı bölümde 11 farklı ödül fonksiyonu ile 100 iterasyon boyunca eğitilmiş ve incelenmiştir. Verilen her ödül değeri deneme yanılma ve sezgisel olarak seçilmiştir. Tezin amacına uygun olması için politikalar üzerinde büyük fark yaratan ödül fonksiyonları oluşturulmuştur.

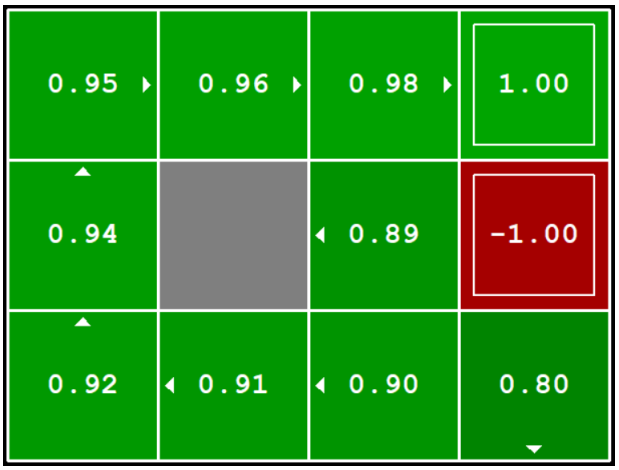
Grid World içerisindeki amaç oldukça basittir. Mümkün olan en yüksek puanı elde etmek oyunun tek amacıdır. Çevre Markov Zinciri Varsayımını içermektedir. Pekiştirmeli öğrenme ile özdeşleşen bu durumlar ödül fonksiyonu incelemek için oyunu oldukça iyi bir test ortamı oluşturmaktadır. Test için kullanılacak bölümlerden ilki Şekil 5.1'de sergilenmiştir.



Şekil 5.1 Bölüm 1 Grid World

Programdaki parametreler gürültü, indirim faktörü ve her adımda alınan ödüldür. Bir bölümün bitmesi için temsilci bölümde verilen belirli durum veya durumlar için ödüllere ulaşmalıdır. Pekiştirmeli öğrenmenin daha iyi anlaşılması için oyuna gürültü eklenmiştir. Bunun nedeni gürültü olmadığında problemin en kısa yol problemi olmasıdır. Bunun için dinamik programlama en kısa yol algoritmaları kullanmak çok daha mantıklıdır. Ayrıca takviyeli öğrenmenin bir amacıda ortama uyum sağlayabilmesidir bu yüzden gürültü 0'dan farklıdır yani ortam stokastiktir, ajan seçtiği aksiyondan farklı hareket edebilir.

İlk deney gürültü 0.2, indirim faktörü 1 ve her adımda alınan ödül -0.01 dir. Basitçe parametreler incelenmek istenirse gürültü temsilcinin 0.8 olasılıkla istediği yöne gitmesini sağlayacaktır. Örneğin ajan yukarı gitmek istediğinde 0.1 sağa ve 0.1 sola istemsiz gitme olasılığı vardır. Eğer sol tarafında duvar varsa ve istemsiz olarak sola gitme olasılığı denk gelmiş ise bulunduğu konumda kalacaktır. Durum beklenen değerleri Şekil 5.2'de gösterilmiştir. Her durum içerisinde bulunan beyaz oklar politikayı temsil etmektedir.



Şekil 5.2 Deney 1

Deney 1 incelendiğinde ajanın bitiş durumlarından birisi olan -1 duruma yakın olduğu durumlarda ödülle ilgilenmeyip amacının -1 bitiş değerinden kaçmak olduğunu görülmüştür. Bunun sebebi her adımda alınan ödül parametresinin çok düşük negatif bir değere sahip olmasıdır. Ajan bölümü -1 değeri ile kapatmamak için yakın olan durumlarda kaçma eylemini göstermiştir. Ayrıca -1 çıkışının solundaki durumda ajanın politikası yukarı gitmek yerine duvara doğru gitmektir. Bunun sebebi gürültünün yukarı gitmek istediğinde -1 ile gitme ihtimalidir. Politika oldukça risksizdir.

Deney 2 için Deney 1 den farklı olarak sadece her adımda alınan ödül -0.03 olarak değiştirilmiştir ve Harita 5.3'te durum değer fonksiyonu değerleri gösterilmiştir

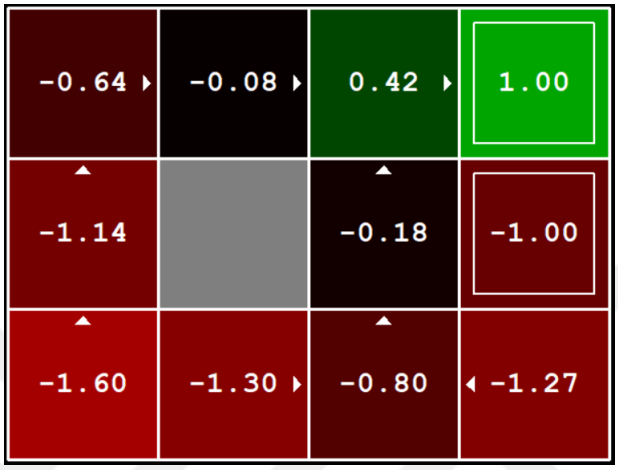


Şekil 5.3 Deney 2

Deney 2'nin politikası ile Deney 1'in politikası arasında küçük ancak etkili bir fark vardır. Politikada -1 çıkışının sol tarafında bulunan durumda temsilciyi sola değil yukarı yönlendirme durumu olduğunu görülmüştür. Deney 1'deki politika 0.8

olasılıkla aynı yerde kalmak ya da en iyi ihtimalle 0.1 olasılıkla +1 çıkışına yakın tarafa gitmektir. Ancak bu sefer temsilciyi riske atma söz konusudur. Yani 0.8 olasılıkla +1 çıkışına yakın olacaktır ancak 0.1 olasılık ile -1 ile de oyun sonlanabilir. Bu sefer temsilci daha uzak yolu seçmeyip toplam ödülü riske atma eylemi göstermiştir.

Deney 3 için parametreler yine farklı olarak her adımda alınan ödülün -0.4 olarak değiştirilmesi yapılmıştır. Durum değer fonksiyonu değerleri Şekil 5.4'te gösterilmiştir.



Şekil 5.4 Deney 3

Deney 3'te ajan artık hangi durumda bulunursa bulunsun -1 çıkışından kaçınmadan +1 çıkışına direkt ulaşmak isteyen bir politika oluşturmuştur. İlk 2 deneyden farklı olarak artık başlangıçta rastgele sağa doğru gittiğinde aşağıdaki yolu kullanma eğilimi göstermektedir. Politika en kısa yol problemine benzer bir tavır sergilemiştir.

Deney 4'te her adımda alınan ödül -2'ye azaltılmış ve durum değer fonksiyonu değerleri Şekil 5.5'te sergilenmiştir.



Şekil 5.5 Deney 4

Deney 4, önceki deneylerden farklı olarak artık -1 çıkışını seçmektedir. Bunun sebebi her adımda alınan -2 değerinin +1 çıkışına ulaşılmak istendiğinde toplam ödülü -1 çıkışına göre düşük olmasıdır. Artık temsilci istemsiz olarak yukarı gitmediği sürece -1 çıkışına gidip bölümü bitirecektir.

Deney 5 ve Deney 6, Bridge World bölümünde gerçekleştirilmiştir. Deney parametreleri indirim faktörü 0.9 ve her adımda alınan ödül 0 her iki deneyde de aynıdır. Deneylerde sadece gürültü parametresi incelenmiştir. Deney 5 için 0.2 olan gürültü parametresi Deney 6'da 0 dır. Bu parametrenin 0 olması gerçek hayatta karşılaşılacak bir durum değildir ancak politika değişimine etkisi ve risklerin nasıl indirildiğinin göstergesi olarak yer verilmiştir. Deney 5'in durum yinelemesi tablosu Şekil 5.6'da sergilenmiştir.



Şekil 5.6 Deney 5

Ajanın bölümü en yüksek puanla bitirmesi için sadece sağa doğru gitmesi gerekmektedir. Aksi taktirde +10 ödülüne oranla çok daha düşük bir değerle bitirebilir ya da bulunduğu konumdaki +1 ödülüne ulaşabilir. Hesaplanan durum değer fonksiyonuna göre oluşan politika orta nokta ve sonrası hariç başlangıç noktasına geri dönmektir. Ajanın politikası orta noktada başlangıç ve çıkış noktalarının uzaklığı aynı olacağından yüksek ödüle yönelmiştir

Deney 6'nın politikası Şekil 5.7'de sergilenmiştir.



Şekil 5.7 Deney 6

Gürültü 0 olduğundan ajan için herhangi bir risk söz konusu değildir. Politika her durumda sağa doğrudur.

Discount Grid bölümü Şekil 5.8'de sergilenmiştir. Bu bölüm için önce amaçlar belirlenmiş sonrasında bu görevleri yerine getirecek parametre değerleri deneme yanılma yöntemiyle bulunmuştur. Görevler Tablo 5.1'de verilmiştir. Discount Grid Bölümünde 3 farklı çıkış vardır. Bunlar +1, +10 ve -10 çıkışlarıdır.



Şekil 5.8 Bölüm 3 Discount Grid

Tablo 5.1 Farklı hedefler için oluşturulan deneyler

Deney İsmi	Hedef
Deney 7	Risk alarak +1 çıkışına git
Deney 8	Güvenli şekilde +1 çıkışına git
Deney 9	Risk alarak +10 çıkışına git
Deney 10	Güvenli şekilde +10 çıkışına git
Deney 11	Asla bitirme

Deney 7'de özellikle her adımda alınan ödülün bu hedef için oldukça etkisi olduğu görülmektedir. Bunun sebebi her adımda alınan ödülün negatif bir değer verilmesi ile yakında bulunan +1 çıkışa gideceği bellidir. Ancak çok küçük negatif olursa bu ödül temsilci +1 çıkışı yerine daha yakın olan -10 çıkışına gidebilir. Yol uzunluğuna göre her adımda alınan ödül parametresi belirlenmelidir. Gürültü faktörü başlangıçtaki şekilde değiştirilmemiştir. İndirim faktörünün 0'a yakın değer seçilmesi ile uzaktaki ödüllerin daha az çekici olması anlamına gelmektedir. Deney 8 için her adımda alınan ödülün çok düşük negatif değer olmaması güvenli olarak uzun yolu tercih etmesine sebep olacaktır. İndirim faktörünün 1'e yakın olmaması ise +1 çıkışına gitmesini teşvik edecektir. Deney 9 için indirim faktörü 1'e yakın seçilmelidir. Temsilci +10 çıkışı indirim faktörünün 1'e yakın olması ile +1 çıkışı yerine +10 çıkışına gidebilecektir. Her adımda alınan ödülün 0 ya da çok düşük olmayan negatif değer olması yine +10 çıkışına teşvik edecektir. Deney 10'da ise Deney 9'a göre indirim faktörünün daha büyük olması ajanın geçeceği durumlardan çok fazla etkilenmeden uzakta da olsa en büyük çıkışa gitmesini sağlayabilir. Gürültü ile her adımda alınan ödül arasında doğru orantılı bir ilişki olduğu düşünülebilir bunun sebebi gürültü azaldıkça riskin azalacağı yani kısa yolun seçileceği, her adımda alınan ödül azaldıkça daha kısa yolun seçileceğidir. Ancak her adımda alınan ödülün azalması temsilcinin en yakın çıkıştan (-10 çıkışı dahi olsa) bölümü bitirmesine teşvik ederken gürültünün azalması riskin azalacağı yani en kısa yoldan en yüksek değere ulaşabilmesini sağlayacaktır. Deney 11 için her adımda alınan ödülün pozitif yüksek bir değer verilmesi güvenli alanda gezmesini sağlayacaktır.

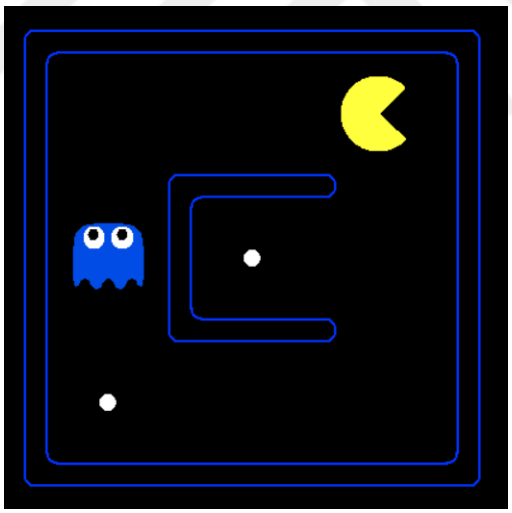
Tablo 5.2 Deneylerin parametre değerleri

Deney	Gürültü	İndirim Faktörü	Her adımda alınan ödül
Deney 7	0.2	0.2	-7
Deney 8	0.2	0.4	-1
Deney 9	0.2	0.8	0
Deney 10	0.1	0.9	-0.5
Deney 11	0.2	0.2	10

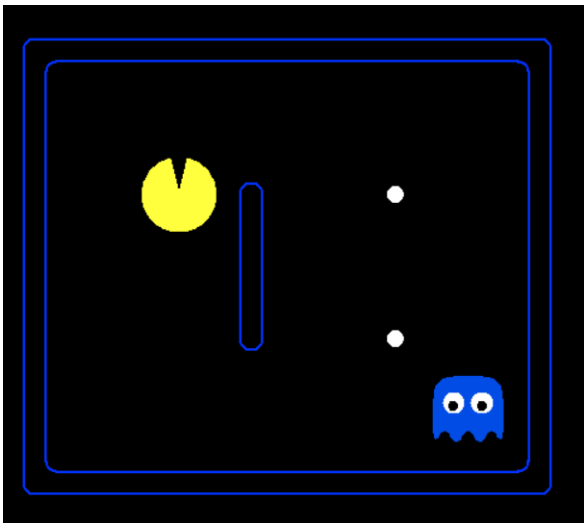
Tablo 5.2'de verilen parametre değerleri ilgili deneylerin hedeflerine ulaşmıştır. Birçok farklı parametre ile bu hedeflere ulaşılacağı deneme yanılma sırasında görülmüştür.

5.2 Pacman

Q Öğrenme algoritması Pacman [28] ortamında 2 farklı bölümde aynı ödül fonksiyonları ile incelenmiştir. Diğer bölüme göre daha küçük olan Small Grid bölümü Şekil 5.9'da ve Discount Grid Bölümü Şekil 5.10 da yer verilmiştir.



Şekil 5.9 Pacman Small Grid



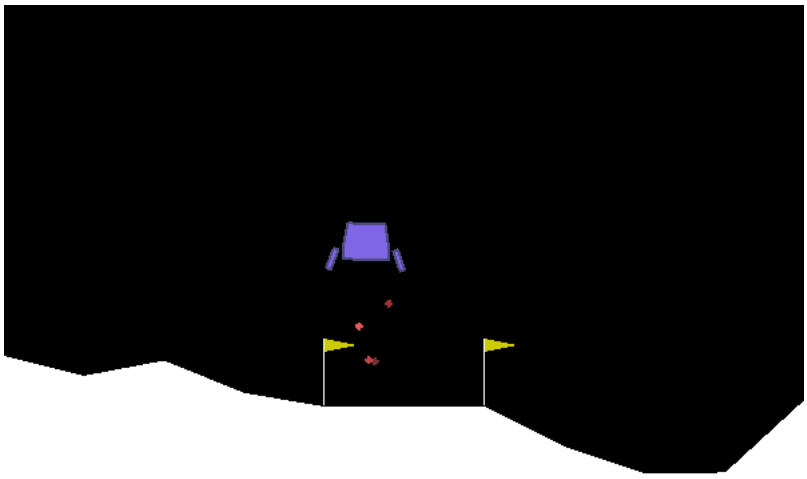
Şekil 5.10 Pacman Discount Grid

Q Öğrenme ajanları 2000 iterasyon eğitilmiş ve 10 defa her bölümün performansları kontrol edilmiştir. Ödül fonksiyonu her adımda -1,oyunu kazanılırsa +500 kaybedilirse -500 şeklindedir. Öğrenme oranı, indirim faktörü ve epsilon değerleri sırası ile 0.05, 0.8 ve 0.2'dir.

Small Grid bölümünde 10/10 başarılı olan temsilci Medium Grid bölümünde 0/10 başarılı olmuştur. Ajanın başarı sağlaması için iterasyon sayısının oldukça fazla arttırılması gerekmektedir. Bunun sebebi birçok önemli değişkenin sadece Q Öğrenme kullanarak göz ardı edilmesidir. Sadece ödüllere değil durumdan elle çıkartılan özellikler kullanıldığında (yiyeceğe uzaklık, hayalete uzaklık, hayalet sayısı v.b.) bu değişkenkerin doğrusal birleşimi ile eğitilen metodun kısa sürede başarı sağladığı [29] ve Derin Q Öğrenme ile çok daha iyi genelleştirileceği görülmüştür.

5.3 Lunar Lander

Derin Q Öğrenme algoritması Şekil A.11'de yer alan Lunar Lander [30] ortamında eğitilmiştir. Ortam durum bilgisini 8 değişkenle ifade etmektedir. Değişkenler Aracın koordinatları, 2 boyuttaki doğrusal hızları, açısı, açısal hızı ve her bacağın zeminle temas halinde olup olmadığını temsil eden 0-1 değerleridir. Ödül fonksiyonu burada çevrenin kendisi tarafından belirlenmiştir. Araç iniş alanına indiğinde yüksek ödül, iniş yapamadığında yüksek ceza ve havada kaldıkça düşük ceza alır. Ortam her denemede değişmektedir.



Şekil 5.11 Lunar Lander

Derin Q Öğrenme mimarisinde çok katmanlı algılayıcılar kullanılmıştır. Bunun sebebi durumların piksel değerleri yerine tanımlı değişkenler olmamasıdır. Çevrenin karmaşıklığı düşük olduğundan dolayı mimari oldukça sığdır (3 katmanlı). Aktivasyon fonksiyonu olarak ReLU kullanılmıştır. 1000 deneme sonrasında ajan optimal politikaya oldukça kolay yaklaşmıştır.

5.4 Doom 2

Lunar Lander'a göre daha karmaşık olan Doom 2 [31] Oyunu ilk 3 boyutlu video oyunlarından birisidir. 3 boyutlu bir ortamda piksel değerlerini kullanarak Derin Q Öğrenme Ajanları eğitilmiştir.

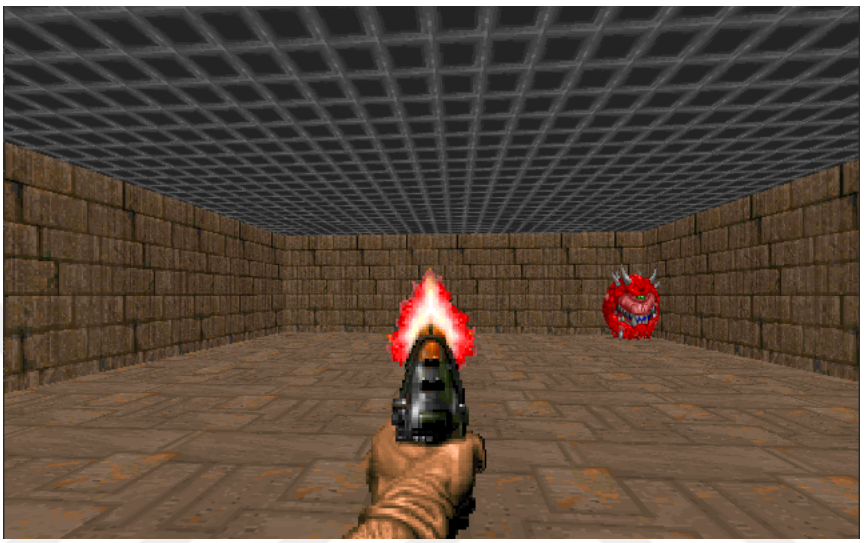
Şekil 5.12'de kutu bölümü sergilenmiştir. Bu bölümde amaç ajanın en kısa sürede canavarı öldürmesidir. Ödül fonksiyonu her yapılan aksiyon için -1 ve canavarın ölümü için +101 ödülü verir. Bu bölüm için ajana 2 farklı aksiyon seti belirlenmiştir. İlk aksiyonlar sağ, sol ve ateştir. Bu aksiyon seti için canavar her zaman ajanın karşısındadır. Ajanın anlık durumu bölümü geçmek için tüm bilgileri içermektedir. Markov Zinciri Varsayımının özelliklerinden birisi olan bu durum bahsi geçen algoritmaların hepsi için önemlidir. Çünkü ajan sadece anlık duruma bakarak tahminde bulunur.

İkinci aksiyon seti için ajanın kamera açılarını istediği şekilde hareket ettirebilir ancak bulunduğu konumu değiştiremez. Aynı ödül fonksiyonu bu deneme için de geçerlidir.

Ajanların eğitimi 5000 bölüm olarak ayarlanmıştır. Toplam ödüllerin eğitim süresi arttıkça birbirine yakınsayacağı aşikardır ancak ilk aksiyon setine sahip ajan çok daha

hızlı gelişme sürdürürken ikinci ajanın toplam ödül grafiği kararsız ve yavaş gelişim göstermiştir.

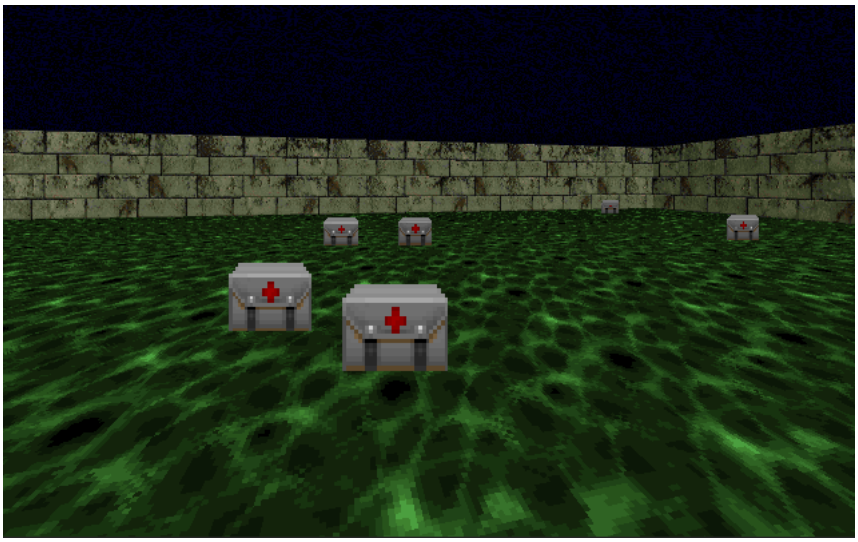
İkinci aksiyon setine sahip ajanın daha kararlı öğrenebilmesi için Markov Zinciri Varsayımı genişletilebilir. Bunun için ajanın son 4 durumu bir deneyim olarak kabul edilebilir. Böylelikle ajan yanlış bir hareket yapıp canavarı kaybetse de ondan önceki durumlarda canavarı gördüğünden bulunduğu durum hala amacı için işe yarar bilgi barındıracaktır.



Şekil 5.12 Doom 2 Kutu

Can Toplama bölümü Şekil 5.13'te sergilenmiştir. Ödül Fonksiyonu can miktarına bağlıdır. Ajan maksimum toplam ödül için en hızlı şekilde can paketlerini toplayıp hayatta kalmalıdır. Ajan 100 can ile başlar ve asit içerisinde bekledikçe canı azalır. Ajan bulunduğu durumda can bilgisini görememektedir. Ödül miktarını canları toplamakla ilişkilendirmesi muhtemeldir. Ancak böyle bir ödül fonksiyonu durum can paketlerinin yenilenmesinin yavaş olduğu durumlarda da hızlı toplamasına ve can değerinin belli bir sınırı aşamayacağından hayatta kalma süresinin azalmasına yol açacaktır.

Ajanın durum bilgisi pikselleri içermektedir. Ajan son 4 durumunun bilgisini anlık olarak kullanabilecek şekilde eğitilmiştir. Düello Mimarisinin DQN mimarisine göre daha hızlı öğrendiği gözlemlenmiştir.



Şekil 5.13 Doom 2 Can Toplama

KAYNAKÇA

[1] R. S. Sutton, A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[2] D. Silver *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[3] J. Schrittwieser *et al.*, "Mastering atari, go, chess and shogi by planning with a learned model," *Nature*, vol. 588, no. 7839, pp. 604–609, 2020.

[4] G. Tesauro, "Td-gammon, a self-teaching backgammon program, achieves master-level play," *Neural computation*, vol. 6, no. 2, pp. 215–219, 1994.

[5] V. Mnih *et al.*, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[6] I. Akkaya *et al.*, "Solving rubik's cube with a robot hand," *arXiv preprint arXiv:1910.07113*, 2019.

[7] B. Baker *et al.*, "Emergent tool use from multi-agent autocurricula," *arXiv preprint arXiv:1909.07528*, 2019.

[8] V. Krakovna, L. Orseau, R. Ngo, M. Martic, S. Legg, "Avoiding side effects by considering future tasks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 19 064–19 074, 2020.

[9] A. Ecoffet, J. Huizinga, J. Lehman, K. O. Stanley, J. Clune, "First return, then explore," *Nature*, vol. 590, no. 7847, pp. 580–586, 2021.

[10] D. Silver *et al.*, "Mastering the game of go without human knowledge," *nature*, vol. 550, no. 7676, pp. 354–359, 2017.

[11] D. Silver *et al.*, "A general reinforcement learning algorithm that masters chess, shogi, and go through self-play," *Science*, vol. 362, no. 6419, pp. 1140–1144, 2018.

[12] M. Minsky, "Steps toward artificial intelligence," *Proceedings of the IRE*, vol. 49, no. 1, pp. 8–30, 1961.

[13] S. P. Singh, R. S. Sutton, "Reinforcement learning with replacing eligibility traces," *Machine learning*, vol. 22, no. 1, pp. 123–158, 1996.

[14] E. Bilgin, *Mastering Reinforcement Learning with Python: Build Next-generation, Self-learning Models Using Reinforcement Learning Techniques and Best Practices*. Packt Publishing, 2020, ISBN: 9781838644147. [Online]. Available: `https://books.google.com.tr/books/about/Mastering_Reinforcement_Learning_with_Py.html?id=_g8FzgEACAAJ&redir_esc=y`.

[15] C. J. Watkins, P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3, pp. 279–292, 1992.

[16] E. Alpaydin, *Introduction to machine learning*. MIT press, 2020.

[17] U. Von Luxburg, B. Schölkopf, "Statistical learning theory: Models, concepts, and results," in *Handbook of the History of Logic*, vol. 10, Elsevier, 2011, pp. 651–706.

[18] M. Bojarski *et al.*, "End to end learning for self-driving cars," *arXiv preprint arXiv:1604.07316*, 2016.

[19] A. Giusti *et al.*, "A machine learning approach to visual perception of forest trails for mobile robots," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 661–667, 2015.

[20] K. Ranjan, A. Christensen, B. Ramos, "Recurrent deep q-learning for pac-man," *URL: https://pdfs. semanticscholar. org/70c2/265cf07405a1deecc7728def9711ff5127b6. pdf*, 2016.

[21] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[22] A. Irpan, *Deep reinforcement learning doesn't work yet*, `https://www.alexirpan.com/2018/02/14/rl-hard.html`, 2018.

[23] H. Van Hasselt, A. Guez, D. Silver, "Deep reinforcement learning with double q-learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 30, 2016.

[24] S. Thrun, A. Schwartz, "Issues in using function approximation for reinforcement learning," in *Proceedings of the 1993 Connectionist Models Summer School Hillsdale, NJ. Lawrence Erlbaum*, vol. 6, 1993.

[25] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, N. Freitas, "Dueling network architectures for deep reinforcement learning," in *International conference on machine learning*, PMLR, 2016, pp. 1995–2003.

[26] T. Schaul, J. Quan, I. Antonoglou, D. Silver, "Prioritized experience replay," *arXiv preprint arXiv:1511.05952*, 2015.

[27] R. Polvara *et al.*, "Autonomous quadrotor landing using deep reinforcement learning," *arXiv preprint arXiv:1709.03339*, 2017.

[28] *Berkeley pacman code*, `http://ai.berkeley.edu/project_overview.html`.

[29] A. Gnanasekaran, J. F. Faba, J. An, "Reinforcement learning in pacman," *See nalso URL http://cs229. stanford. edu/proj2017/final-reports/5241109. pdf*, 2017.

[30] G. Brockman *et al.*, "Openai gym," *arXiv preprint arXiv:1606.01540*, 2016.

[31] M. Kempka, M. Wydmuch, G. Runc, J. Toczek, W. Jaśkowski, "Vizdoom: A doom-based ai research platform for visual reinforcement learning," in *2016 IEEE conference on computational intelligence and games (CIG)*, IEEE, 2016, pp. 1–8.

TEZDEN ÜRETİLMİŞ YAYINLAR

Konferans Bildirisi

1. Alihan Öztürk, Fatma NOYAN TEKELİ - 8 th International Conference on Advances in Statistics 16-18 May 2022, BOOK OF ABSTRACTS, s. 29

Proje

1. "Video Oyunlarında Derin Öğrenme" BAP Yüksek Lisans Tez Projesi, FYL-2021-4388