T.C.
YALOVA ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
BİLGİSAYAR MÜHENDİSLİĞİ PROGRAMI

AKCİĞER RÖNTGEN GÖRÜNTÜLERİNDEN COVİD'19 VE ZATÜRRE HASTALIĞININ KUANTUM MAKİNE ÖĞRENMESİ YÖNTEMLERİ İLE TAHMİNİ

YÜKSEK LİSANS TEZİ

SEÇMEN ŞAHİN

DANIŞMAN: DR. ÖĞR. ÜYESİ GÜNEŞ HARMAN

YALOVA
KASIM 2023

T.C.
YALOVA ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
BİLGİSAYAR MÜHENDİSLİĞİ PROGRAMI

AKCİĞER RÖNTGEN GÖRÜNTÜLERİNDEN COVİD'19 VE ZATÜRRE HASTALIĞININ KUANTUM MAKİNE ÖĞRENMESİ YÖNTEMLERİ İLE TAHMİNİ

YÜKSEK LİSANS TEZİ

SEÇMEN ŞAHİN
205105001

DANIŞMAN: DR. ÖĞR. ÜYESİ GÜNEŞ HARMAN

YALOVA
KASIM 2023

ETİK BEYAN

Yalova Üniversitesi Lisansüstü Eğitim Enstitüsü Tez Yazım Kuralları'na uygun olarak hazırladığım "**AKCİĞER RÖNTGEN GÖRÜNTÜLERİNDEN COVİD'19 VE ZATÜRRE HASTALIĞININ KUANTUM MAKİNE ÖĞRENMESİ YÖNTEMLERİ İLE TAHMİNİ**" başlıklı bu tez çalışmasında; tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi, tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu, tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi, kullanılan verilerde herhangi bir değişiklik yapmadığımı, bu tezde sunduğum çalışmanın özgün olduğunu bildirir, aksinin tespiti halinde doğabilecek her türlü hukuki sorumluluğu kabul ettiğimi taahhüt ve beyan ederim.

Seçmen ŞAHİN

ÖNSÖZ

Lisans Eğitimime başladığım andan itibaren çalışma hayatımda manevi desteğini sürekli gösteren Prof. Dr. Doğan ŞENYÜZ, Prof. Dr. Zekeriyya ARI ve Prof. Dr. Halit AKER'e çok teşekkür ederim. Yüksek Lisans eğitimim boyunca bilgi ve tecrübelerinden yararlandığım, her zaman desteğini gördüğüm, benim için yeni bir ufuk çizmiş olan, saygı değer tez danışmanım Dr. Öğr. Üyesi Güneş HARMAN hocama minnet ve teşekkürlerimi sunarım.

Maddi ve manevi destekleri ile her koşulda yanımda olan sevgili aile üyelerime ayrı ayrı şükranlarımı sunarım.

Kasım – 2023 Seçmen ŞAHİN

İÇİNDEKİLER

KISALTMALAR LİSTESİ

GAA	: Grover'ın Arama Algoritması
MAA	: Makine Öğrenimi Algoritmaları
ML	: Makine Öğrenimi
NN	: Sinir ağları
QA	: Kuantum Tavlama
QAOA	: Kuantum Yaklaşık Optimizasyon Algoritması
QC	: Kuantum Hesaplama
QGD	: Kuantum Gradyan İnişi
QML	: Kuantum Makine Öğrenimi
QNN	: Kuantum Sinir Ağları
QP	: Kuantum Fiziği
QPCA	: Kuantum Temel Bileşen Analizi
QRL	: Kuantum Takviyeli Öğrenme
QSVM	: Kuantum Destek Vektör Makineleri
RL	: Pekiştirmeli Öğrenme
SL	: Denetimli Öğrenme
SSL	: Yarı Denetimli Öğrenme

ŞEKİLLER LİSTESİ

AKCİĞER RÖNTGEN GÖRÜNTÜLERİNDEN COVİD’19 VE ZATÜRRE HASTALIĞININ KUANTUM MAKİNE ÖĞRENMESİ YÖNTEMLERİ İLE TAHMİNİ

ÖZET

Kuantum evrişimli sinir ağları (QCNN'ler), kuantum hesaplamanın potansiyel olarak güçlü bazı yönlerinden yararlanarak CNN'lerin yeteneklerini genişletir. Bir dizi rastgele kuantum devresi kullanarak verileri yerel olarak dönüştürerek giriş verileri üzerinde çalışır. Klasik evrişimli sinir ağlarının verimliliğinden yola çıkarak, Evrişimli sinir ağını (QNN'ler) kullanarak veriler analiz edilmiş, tahminler yapılmış ve sonuçlar değerlendirilmiştir. Kuantum halinde kodlanmış covid’19 veri setinin ikili sınıflandırması gerçekleştirilmiştir. Ayrıca Pennylane'in "varsayılan qubit" cihazındaki farklı parametreleri de dikkate alarak performansı araştırılmıştır. Kullanılan veri seti modeli için 250 eğitim ve 65 test görüntüsü içermektedir. Veri setinde verilen görüntüler gerçek hayattaki göğüs röntgenidir ve önceden değiştirilmemiştir. Ancak hesaplama kaynaklarındaki bazı sınırlamalar nedeniyle bu çalışmada boyut 28x28 olarak tutulmuştur. Model-1, 'Normal Kişi' ve 'Covid’19/Viral Pnömoni' olmak üzere iki sınıf arasında sınıflandırma yapar. Model-2, 'Covid’19' ve 'Viral Pnömoni' olmak üzere iki sınıf arasında sınıflandırma yapar. Kuantum Sınıflandırıcısı 1'de, Temel Veri Analizi ile çıkarılan 256 öznitelik boyutlu girdi verisinden 11 öznitelik kullanılmıştır. Burada yaklaşık %70 doğruluk elde edilmiştir. Kuantum Sınıflandırıcısı 2’de TruncatedSVD yöntemini kullanarak her görüntünün 256 özniteliği 4'e indirilmiştir. Yaklaşık %72 doğruluk (accuracy) elde edilmiştir. Kuantum Sınıflandırıcısı 3’de verileri yalnızca 2 özniteliğe indirgenmiştir. Beklenmedik bir şekilde bu daha önce yaklaşılanların en yükseği olan %76 doğruluğu vermiştir.

Anahtar Kelimeler: Derin öğrenme, Makine Öğrenmesi, Kuantum makine öğrenmesi, Kuantum evrişimli sinir ağları, Covid’19

PREDICTION OF COVID'19 AND PNEUMATURE FROM LUNG X-RAY IMAGES USING QUANTUM MACHINE LEARNING METHODS

ABSTRACT

Quantum convolutional neural networks (QCNNs) expand the capabilities of CNNs by leveraging some of the potentially powerful aspects of quantum computing. It works on the input data by locally transforming the data using a series of random quantum circuits. Based on the efficiency of classical convolutional neural networks, using Quanvolutional neural networks (QNNs) data were analyzed, predictions were made and results were evaluated. Binary classification of the covid'19 data set encoded in quantum form was performed. In addition, the performance of Pennylane's "default qubit" device was investigated by taking into account different parameters. The dataset used contains 250 training and 65 test images for the model. The images provided in the dataset are real-life chest X-rays and have not been previously modified. However, due to some limitations in computational resources, the size is kept as 28x28 in this study. Model-1 classifies between two classes, 'Normal Person' and 'Covid'19/Viral Pneumonia'. Model-2 classifies between two classes, 'Covid'19' and 'Viral Pneumonia'. In Quantum Classifier 1, 11 features were used from 256 feature sized input data extracted by Fundamental Data Analysis. Here, approximately 70% accuracy has been achieved. Using the TruncatedSVD method in Quantum Classifier 2, 256 features of each image are reduced to 4. Approximately 72% accuracy (accuracy) was obtained. In Quantum Classifier 3, its data is reduced to only 2 features. Unexpectedly, this yielded an accuracy of 76%, the highest ever approached.

Keywords: Deep learning, Machine learning, Quantum machine learning, Quantum convolutional neural networks, Covid'19

1. GİRİŞ

Makine öğrenmesi yarım yüzyıldan fazla bir süredir geliştirilmektir. Hesaplama yeteneğinin de gelişmesiyle birlikte bilgisayar biliminin çok önemli bir parçası haline gelmiştir. Bilgi işlem gücü, teknolojideki hızlı gelişimle birlikte oldukça hızlı bir şekilde artmıştır. Sürekli olarak yeni algoritmalar ortaya çıkmış olsa da veri artış hızı bilgisayarların performansındaki artış hızından çok daha fazladır.

Artan veri ile birlikte hızlanan ve öğrenen bilgisayarların gelişmesi bugün çok daha değerli bir konuma gelmiştir. Geldiğimiz noktada klasik makine öğrenmesinin yanı sıra teknolojideki ilerlemeler sonucunda kuantum fiziğinin temellerinden faydalanan ve kuantum hesaplama yapabilen bilgisayarlar geliştirilmektedir. Kuantum hesaplama, süperpozisyon ve dolanıklık gibi kuantum mekaniği olgularına dayanmaktadır.

Makine öğrenmesinin tanımı göz ardı edildiğinde öğrenme "Denetimli öğrenme (supervised learning), denetimsiz öğrenme (unsupervised learning) ve pekiştirmeli öğrenme (reinforcement learning)" olarak üç ana kategoriye ayrılır.

Yüksek hızlı hesaplama için en önemli özellik olması nedeniyle, paralellik belirli sorunları çözmek için belirli algoritmalarda tasarlanabilir. Bu klasik problemler genellikle kuantum sisteminde olduğu kadar verimli şekilde çözülemez. Makine öğrenmesi, hesaplama gücü eksikliği nedeniyle baskı altında olduğundan ve kuantum hesaplama bu güçlü hesaplama yeteneğine sahip olduğundan, insanlar kuantum hesaplama ve makine öğrenimi kombinasyonunun olasılıklarını düşünmektedir.

Örneğin; Shor'un algoritması, kuantum hesaplamanın herhangi bir klasik yöntem kullanarak imkânsız olan büyük tamsayı çarpanlarına ayırma problemini çözmek için üstel bir hızlanma sağlayabildiğini göstermektedir (Zhang ve Ni, 2020). Bu algoritmadan sonra, belirli problemleri çözmek için çok sayıda kuantum algoritması önerilmiştir. Yine, Grover'ın algoritmasının yapılandırılmamış bir veri tabanında arama yaparken ikinci dereceden bir hızlanma sağlayabileceği kanıtlanmıştır(Zhang ve Ni, 2020).

Öte yandan, bazı şirketler ve araştırma kurumları, kuantum devre modeline dayalı evrensel kuantum bilgisayarların gerçek prototip makinelerini üreterek, bulut

platformları aracılığıyla az sayıda kübit (qubit) üzerinde kuantum hesaplama işlemleriyle deneyler yapılabilmesini sağlamıştır.

Günümüzde, genel büyük ölçekli kuantum bilgisayarı halen geliştirilmeye devam etmektedir. Bununla birlikte, potansiyel kuantum makine öğrenimi algoritmalarının araştırılmasında bazı ilerlemeler kaydedilmiştir (Schuld ve Sinayskiy, 2015). Kuantum makine öğrenmesi klasik makine öğrenemsinde olduğu gibi, kuantum algoritmalarının makine öğrenmesi programlarına entegre edilmesidir. Bugün yoğun bir şekilde kullanılan makine öğrenmesi algoritmaları, çok büyük miktarlarda veriyi hesaplamak için kullanılırken; kuantum makine öğrenmesi, algoritmaları yapılan uygulamada kullanılan algoritmalar tarafından yapılan hesaplama hızını ve veri depolamayı iyileştirmek için kübitler ile kuantum işlemlerini veya özel kuantum sistemlerini kullanır.

Bu durum, bilgisayara hesaplama açısından zor alt işlemlerin bir kuantum cihazına dış kaynak olarak verildiği hem klasik hem de kuantum veriyi işlemeyi içeren hibrit yöntemleri barındırır. Bu sureçler çalışma şekli olarak karmaşık olabilir ve klasik bilgisayara göre kuantum bilgisayarda daha hızlı yürütülebilir.

Makine öğreniminde iki temel vardır; "veri ve öğrenme süreci". Aynı şekilde kuantum alanı da bu iki parçayı içerir.

Kuantum hesaplamanın kuantum verileriyle uğraşması gerektiğinden, kuantum hesaplamanın hayal edildiği gibi çalışabilmesi için klasik verilerin kuantum verilerine önceden işlenmesi gerekir. Günlük hayatta, insanların uğraştığı çoğu bilginin klasik olduğuna inanılır, bu nedenle bu ön işlemeyi yapmak gereklidir. Bununla birlikte, kuantum verisinin doğrudan işlenebileceği özel bir durum vardır. Örneğin, kuantum iletişimi son yıllarda sıcak bir konu haline gelmiştir ve kuantum kanallarında, kuantum olan bazı gürültüler olabilir (Biamonte ve diğ, 2017).

Bugün dünyada üretilen veri miktarı üstel olarak artarken, klasik bilgisayarların bu veriyi işleyip anlamlı bilgi çıkarma süresi de artmaktadır. Büyük veri kavramının hayatımıza girdiği günden beri önişleme süreçlerinin fazlalığı ya da bu süreçlerin otomatik olarak gerçekleştirilmesi yapay zekâ yöntemlerinde kaynak ihtiyacı gerektiren faktörlerdir. Veriye olan ihtiyacının karşılanabilmesi için çok çekirdekli, çok işlemcili ve grafik işlemcili bilgisayarlar yoğun bir şekilde kullanılmaktadır. Makine öğrenmesi modellerinin zaman maliyetini minumum seviyeye indirmek için

duyduğu performans ihtiyacı şuanki teknolojilerle kısmen karşılanmaktadır. Günümüzde yüksek başarım ile çalışan makine öğrenmesi yöntem ve uygulamalarının, kuantum bilgisayarlar ile bahsedilen zaman ve doğruluk maliyetini çok daha iyi noktalara taşıyacağı görüşü baskındır (Schuld ve Killoran, 2018).

1.1. Literatür Taraması

Makine öğrenimi, veri işleme ve sınıflandırma için her alanda kullanılan ve etkili bir teknik haline gelmiştir. Ayrıca, birçok alanda kuantum hesaplamanın üstünlüğü ve ilerlemesi nedeniyle (örneğin, kriptografi, makine öğrenimi, sağlık hizmetleri vb.), klasik makine öğrenimi ve kuantum bilgi işleme kombinasyonu kuantum makinesi öğrenimi olarak adlandırılan yeni bir alan haline geldi.

Bu bölüm, QML (Quantum Machine Learning) ile ilgili kapsamlı bir incelemesini sunmaktadır. Son yıllarda ML (Machine Learning) tabanlı QC (Quantum Computing) dikkate değer bir evrime tanık oldu. Kuantum otomatik kodlayıcılar (Khoshan ve diğ., 2018, Pepper ve diğ., 2019, Romero ve diğ., 2017), kuantum biyomimetiği (Alvarez-Rodriguez vd diğ., 2018, Lamata, 2020), Kuantum İletişimi (Nawaz ve diğ., 2019, Sheng ve Zhou, 2017, Wallnöfer ve diğ., 2020), Kuantum Tavlama (Li ve diğ., 2018, Rieffel ve diğ., 2015), Hesaplama Kimyası (McArdle ve diğ., 2020, von Lilienfeld, 2018) ve Boltzmann makinesi (Amin ve diğ., 2018). Yapılmış olan çalışmada, QML algoritmaları ML ile kullanılan kuantum hesaplama fikrine göre üç kategoride düzenlenebilir ve sınıflandırılabilir. Tamamen QML (Beer ve diğ., 2020, Dunjko ve diğ., Levine ve diğ., 2019), hibrid klasik -quantum ML (Killoran, Bromley ve diğ., 2019, Mari ve diğ., 2020), kuantumdan ilham alan ML (Gao ve diğ., 2017, Pomarico ve diğ., 2021).

Huang ve diğ., 2021, ML görevlerinde yeni bir yaklaşım önermek için potansiyel kuantum avantajını kullandılar. Bu yaklaşım, giriş veri alanı aracılığıyla geometrik çekirdek işlevine dayanmaktadır. Ayrıca, klasik alanda "öngörülen kuantum çekirdekleri" (PQK) adı verilen kuantum ve klasik ML modellerini kullanarak bir kuantum çekirdeği sağladılar. Bu kuantum çekirdeği, veriler arasındaki benzerliği ölçer ve öğrenme görevlerinde katı kuantum hızlandırma sağlar. Burada kullanılan geometrik sabit, çeşitli çekirdek fonksiyonlarına sahip klasik ve kuantum ML

algoritmalarındaki geometrik farkı ölçer. Potansiyel kuantum avantajının veri miktarına dayandığını bildirdiler.

Schuld ve diğ., (2016), denetimli öğrenmeye dayanan örüntü tanıma için yeni bir kuantum algoritması önermişlerdir. Bu algoritma, kuantum doğrusal regresyon adı verilen doğrusal regresyonun bir versiyonudur. Yazarlar, kuantum durumundaki verileri dönüştürmek için genlik kodlama yöntemini kullandılar. Kuantum lineer regresyon, logaritmik zaman içinde özelliklerin n-boyutları ile kuantum verileri üzerinde çalışır. Rebentrost ve diğ., (2014), büyük verilerin sınıflandırılması için Destek Vektör Makinesi'nin (QSVM) kuantum versiyonunu sunmuşlardır. Kuantum SVM, büyük verileri eğitmek için iç ürünün matris inversiyonunu gerçekleştirmek için mükemmel olmayan bir matrise dayanmaktadır. Logaritmik karmaşıklığa sahip çok sayıda özellik ve örnekle çalışır. Büyük verilerin kuantum SVM'si, klasik SVM'ye kıyasla özellik boyutları dikkate alındığında üstel hızlandırma sağlar.

Klasik NN'lere göre QNNs avantajları Ezhov ve Ventura'da (2000) (örn. Kuantum paralellik, daha yüksek stabilite, daha yüksek bilgi işleme hızı ve bellek kapasitesi) tartışılmaktadır. Da Silva ve diğ., (2016), Kuantum Potansiyel Frekansları (QPF) üzerinde kuantum algılama adı verilen yeni bir QNS (Quantum Network Solution) ve öğrenme algoritması, süperpozisyon tabanlı mimari öğrenme (SAL) olarak adlandırdı. Sal algoritması bir üst üste binme özelliğine ve kuantum operatörüne dayanmaktadır. Ayrıca, NN mimarisini polinom zamanıyla işler. QPF, kuantum algılama modellerinin sınırlamalarının üstesinden gelir. Başka bir çalışmada, yazarlar (Schuld ve diğ., 2015), kuantum donanımı üzerinde kuantum faz tahmini kullanarak klasik algılama kuantum bir versiyonunu tanıttılar. Kuantum algılama algoritması, NNS'deki aktivasyon fonksiyonunu (adım işlevi) simüle eder.

Rekabetçi NN'lerde, Zhou (2010) iki ana parça sundu: birincisi, QC adı verilen Rekabetçi Öğrenme NN'lerine dayanan yeni bir model. QCNN modeli, kuantum paterni rekabetini kullanarak giriş modellerini sınıflandırır. İkinci bölümde Zhou, önerilen QCNN için bellek kapasitesi sağladı. QCNN, ağ ağırlığı olmadan bir kuantum kaydı kullanarak rekabetçi öğrenme elde eder. Kuantum dolaşımını ve Grover'ın algoritmasını kullanan bir başka QCNN modeli önerilmektedir (Zhong ve Yuan, 2012). Bu model, sahte desenler nedeniyle kuantum ilişkilendirici bellek kullandı. Ayrıca, bu model rekabet sürecinde sahte durumları eksik kalıplarda

hatırlar. Zidan ve diğ., (2019), QCPNN adı verilen ikili sınıflandırma için dolaşma önlemine dayanan başka bir QCNN önermişlerdir. QCPNN, giriş verilerini bir kuantum bilgisayardaki eksik desenlerde sınıflandırır.

Son zamanlarda, Abbas ve diğ., (2021) QNN'nin gücünü mevcut yakın vadeli kuantum donanımı ile tartışmıştır. Yazarlar, adlandırdıkları modelin kapasitesi, etkili boyut için yeni bir önlem önermişlerdir. Bu etkili boyut, modelin yeni/görünmeyen veriler üzerinde genelleme yeteneğini sınırlamak için kullanılır. Buna ek olarak, önlemlerinin bir Fisher bilgi matrisi ile veriye bağlı bir genelleme yöntemi olduğunu bildirdiler. Son olarak, yazarlar QNN'nin mevcut gürültülü kuantum cihazı ile klasik NN'ye kıyasla daha hızlı eğitim aldığını bildirmişlerdir. Ayrıca QNN'nin klasik NN'den daha yetenekli olduğunu gösterdiler. Chen ve Yoo (2021), hibrit kuantum - klasik ML'ye dayanan yeni bir eğitim modeli önerdi. Yazarlar kuantum donanımını (yani, cihaz veya simülatör okuma) yerel istemciler olarak kullandılar. Ayrıca, yazarlar özellik çıkarma için VGG16 ile klasik -quantum transfer öğrenimi kullanmışlardır. Önerilen çerçevenin avantajı klasik ve kuantum verileri üzerinde çalışır.

Dang ve diğ., (2018), görüntü sınıflandırması için kuantum KNN algoritması olarak adlandırılan yeni bir kuantum modeli önerdi. Kuantum KNN modeli iki bölümden oluşur: klasik ve kuantum bölümü. Yazarlar klasik bilgisayarı görüntülerin özelliklerini çıkarmak için kullandılar. Çıkarılan özellikler bir kuantum cihaz tarafından bir kuantum durumuna dönüştürülür. Ardından, kuantum devresi, görüntüler arasındaki benzerliği hesaplamak için kullanılır. Son olarak, sınıflandırma işlemi bir ölçüm devresi tarafından gerçekleştirilir. Kuantum KNN modeli, verimlilik ve sınıflandırma performansı açısından klasik modellerden daha iyi performans gösterir. Adhikary ve diğ., (2020), tek bir kuantum sistemi ile yeni bir varyasyonlu kuantum sınıflandırıcısı sunmak ve tek-atış eğitimi adı verilen bir eğitim algoritması ile n-boyutlu verileri kodlamak için bir kuantum devresi kullanmıştır. Ayrıca, yazarlar tüm veri kümesini tek bir kuantum durumuna kodladılar. Tek atış eğitim, eğitim için daha az parametre kullanır ve daha yüksek hassasiyet elde eder. Mitariai ve diğ., (2018), sınıflandırma, regresyon ve kümeleme, kuantum devre öğrenimi (QCL) gibi farklı görevleri yerine getirmek için hibrit bir klasik -quantum tekniği sundu. QCL, küçük ölçekli kuantum cihazlarda hareket eder. Yazarlar, QCL'nin

yüksek boyutlu sınıflandırma/regresyon görevleri ile performansını gözlemlemişlerdir.

Başka bir hibrit çalışmada, Henderson ve diğ., (2020) yazarları, görüntü sınıflandırması için standart evrişimsel sinir ağlarına sahip kuantum devreleri kullandılar. Yazarlar, mevcut küçük ölçekli ve NISQ kuantum donanımında uygulamak için küçük derinlikli bir kuantum devresi kullandılar. Kuantum devresi, bilgilendirici özellikleri çıkarmak için bir evrişim katmanı olarak uygulanır. Kuantum evrişim tabakasında üç aşama vardır: kodlama, kuantum devresi ve ölçüm. Başka bir mimaride Bausch (2020), QRNN adı verilen tekrarlayan sinir ağının (RNN) kuantum bir versiyonunu önerdi. QRNN'nin temel bileşeni kuantum bir nörondur. QRNN, rakam verilerini sınıflandırmak için kullanılır. Ayrıca, QRNN üretken bir model olarak kullanılır.

Benedetti ve diğ., (2019), veri odaklı kuantum devre öğrenimi (DDQCL) adı verilen çerçeve üretken bir model sunmuşlardır. Kuantum bilgisayarı kullanarak yazarlar (Zhao, Pozas-Kerstjens, Rebentrost ve Wittek, 2019) derin öğrenme için Bayes tekniğinin yeni bir versiyonunu önerdi. Bu tekniğin ana kısmı kuantum matris inversiyonudur. Bu teknik iki kuantum donanımında (Rigetti ve IBM) uygulanır.

QML yeni bir araştırma alanı haline geldi ve birçok uygulamada yer aldı. QC'nin ilerlemesi ve başarısı yaygın olarak görülmektedir. Bu nedenle, QM'nin avantajları ve özellikleri ML'ye uygulanmalıdır. Bildiğimiz kadarıyla, Chrisley'de (1995) uygulanan ilk QML kavramı. QC, Aïmeur, Brassard ve Gambs (2006) ve Lloyd, Mohseni ve Rebentrost'ta (2013) denetimli ve denetimsiz öğrenme ile karıştırılmıştır. Aïmeur ve diğerleri, 2006, Dunjko ve diğerleri, 2016 ve Schuld (2018) 'de QML algoritmalarını kuantum veya klasik algoritmanın entegrasyonuna bağlı olarak dört kategoriye ayırmıştır ve kuantum veya klasik verilerde gösterildiği gibi aşağıdaki Şekil 1.1 ‘de olduğu gibi tanımlanabilir.

		Algoritma Tipi	
		Klasik	Kuantum
Veri Tipi	Klasik	CC	CQ
	Quantum	QC	QQ

Şekil 1.1.Kuantum/klasik verilere ve kuantum/klasik algoritmaya dayalı kuantum makinesi öğrenme algoritmaları

- ✓ Kuantum -Quantum (QQ) kategorisi, bu kategori tamamen QML olarak da bilinir. QQ kategorisi kuantum algoritmaları ve verileri kullanır.
- ✓ Kuantum -Klasik (QC) kategorisi olan bu kategori, klasik ajanlardan öğrenmek için bir kuantum algoritması kullanır (Kuo, Fang ve Chen, 2021).
- ✓ Klasik -Quantum (CQ) kategorisi, CQ algoritmaları standart ML'nin kuantum sürümleridir ve bu algoritmalar gerçek bir kuantum cihazda yürütülebilir.
- ✓ Kuantumdan ilham alan ML kategorisi tarafından yaygın olarak kullanılan klasik-klasik (CC) kategorisi. İlham ile CC kategorisinde kuantum bilgi işlem özellikleri (yani kuantum bitleri, süperpozisyon ve dolaşma) kullanılır.

Kuantum bilgi işlem, parazit, süperpozisyon ve dolaşma gibi kuantum mekanik özelliklerini kullanarak bilgileri işler. Bu nedenle, kuantum bilgi işlem klasik algoritmaları geliştirmek için makine öğrenimi (ML) gibi çeşitli alanlarla entegre edilmiştir. Bu bölüm, kuantum makine öğrenimi (QML) paradigmaları (örn., Tamamen QML, hibrid klasik-quantum ML, kuantumdan ilham alan ML) hakkında kapsamlı bir literatür çalışması yapmak için düzenlenmiştir. Ayrıca, kuantum derin öğrenmesinde en son çalışmalar sunulmuştur. Hilbert alanında klasik verileri kuantum verilerine kodlamak için çeşitli yöntemler vardır. Klasik makine öğreniminin performansını artırmak için kuantum alt rutin kullanan birçok kuantum makine öğrenimi önerilmiştir. Bazı kuantum alt rutinlerini ve uygulamalarından bahsedildi.

Araştırmacılar için yeni yollar açmak için QML'nin gelecekteki perspektifleri ve zorlukları da ele alındı. Sınırlı kubit sayıları, küçük ölçekli kuantum donanımı ve kodlama yöntemleri nedeniyle QML tekniklerinin gerçek dünya sorunları ile uygulanması ve uygulanması için hala zorluklar vardır.

1.2. Çalışmanın Literatüre Katkısı

Klasik ve Kuantum Makine Öğrenmesi (Quantum Machine Learning - QML) şeklinde yapılan bu hibrit çalışmanın literatüre sağlıyacağı düşünülen katkısı aşağıda maddeler halinde verilmiştir.

- **Hız ve İşlem Kapasitesi:** Klasik bilgisayarlar, belirli problemleri çözmek için sınırlı işlem kapasitesine sahiptir. Kuantum bilgisayarlar ise belirli tipteki problemleri daha hızlı çözebilir. Hibrit bir sistem, klasik bilgisayarların genel hesaplama yeteneklerini kullanırken, özellikle kuantum avantajlarına sahip problemleri çözmek için kuantum bilgisayarları kullanabilir.
- **Veri İşleme ve Analizi:** Hibrit bir yaklaşım, büyük veri setlerinde daha hızlı ve etkili bir şekilde işlem yapabilir. Kuantum bilgisayarlar, belirli veri analizi problemlerinde paralel hesaplamaları kullanarak klasik bilgisayarlardan daha etkili olabilir.
- **Belirli Algoritmaların İyileştirilmesi:** Klasik algoritmaların belirli zorlu problemlerde yetersiz olduğu durumlar vardır. Kuantum algoritmaları, bu tür problemleri çözmek için özel olarak tasarlanabilir. Hibrit bir yaklaşım, belirli algoritmaların klasik versiyonlarına kuantum iyileştirmeleri ekleyerek performansı artırabilir.
- **Yeni Algoritmaların Geliştirilmesi:** Hibrit sistemler, kuantum ve klasik bilgisayarlar arasında etkileşim sağlayarak yeni ve daha etkili algoritmaların geliştirilmesine olanak tanır. Bu, belirli problemleri çözmek için özel olarak tasarlanmış algoritmaların oluşturulmasına imkan tanır.
- **Çeşitli Uygulama Alanları:** Hibrit bir yaklaşım, finans, sağlık, yapay zeka, optimizasyon problemleri gibi çeşitli uygulama alanlarında kullanılabilir. Bu, klasik ve kuantum bilgisayarların güçlü yanlarını birleştirerek daha geniş bir problem yelpazesine hitap edebilir.

Ancak, kuantum bilgisayarlar henüz geniş çapta ticari olarak kullanılabilir değillerdir ve belirli teknik zorluklarla karşılaşmaktadırlar. Bu nedenle, hibrit bir yaklaşımın pratik uygulama alanları ve gerçek dünya etkileri üzerindeki çalışmalar halen aktif araştırma konularıdır.

1.3. Tezin İçeriği

Yapılmış olan bu tez çalışmasının ilerleyen bölümleri şu şekilde devam etmektedir; Bölüm 2, Makine öğremesi hakkında ayrıntılı bilgi içermektedir. Bölüm 3, Derin öğrenme hakkında ayrıntılı bilgi içermektedir. Ayrıca bu bölümde çalışma kapsamında kullanılan Derin öğrenme modeli, hakkında bilgi içermektedir. Bölüm 4, çalışmanın ana amacı olan Kuantum Makine öğrenmesi ve çeşitleriyle ilgili bilgi ayrıntılı bilgi vermektir. Ayrıca bu bölümde kuantumun temel yapsınını oluşruran kübit kavramı, süper yoğun kodlama ve son olarak kuantum kapılarıyla ilgili bilgi içermektedir. Bölüm 5, Temel kuantum algoritmalarını hakkında bilgi vermektedir. Bölüm 6, IBM tarafından devre ve algoritma düzeyinde kuantum bilgisayarlarla çalışmak için oluşturulan bir yazılım geliştirme kiti olan Qiskit, hakkında bilgi vermektedir. Bölüm 7, tez çalışması kapsamında kullanılan veri seti ve önişleme, oluştuturan kuantum devresi, sınıflandırıcı model ve ortaya çıkan bulgular hakkında bilgi vermektedir. Son bölüm olan Bölüm 8, sonuçlar ve öneriler kısmını oluşturmaktadır.

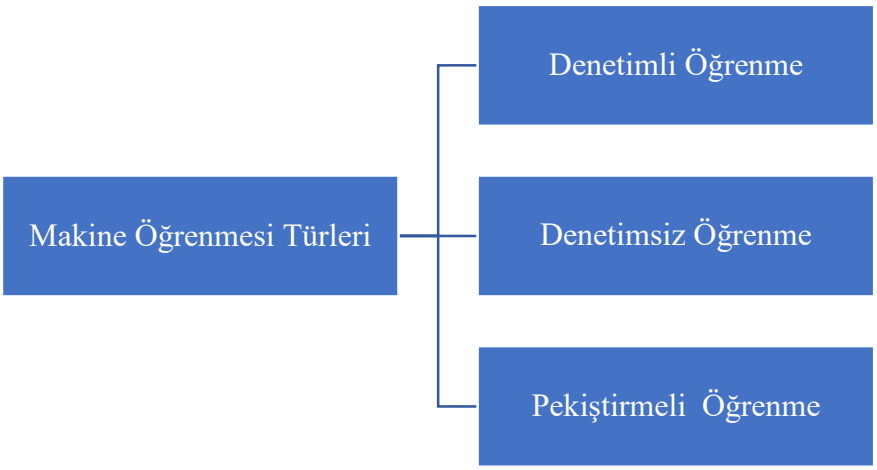
2. MAKİNE ÖĞRENMESİ

2.1. Makine Öğrenmesi

Makine öğrenimi, deneyimlerden "yapay" bilgi üretimi için kullanılan genel bir terimdir (Reitmaier, 2015). Buna göre, yapay bir sistem örneklerden öğrenir ve öğrenme aşaması tamamlandıktan sonra bunları genelleştirebilir. Bunu yapmak için, makine öğrenimi algoritmaları eğitim verilerine dayalı istatistiksel bir model oluşturur ve bu model test verilerine karşı test edilir. Bu, örneklerin basitçe ezbere öğrenilmediği, ancak öğrenme verilerinde kalıpların ve düzenliliklerin tanındığı anlamına gelir. Bu şekilde, sistem bilinmeyen verileri de değerlendirebilir (öğrenme transferi) veya bilinmeyen verileri öğrenmede başarısız olabilir (aşırı uyum). Olası uygulamaların geniş yelpazesinden şunlar sayılabilir: “otomatik teşhis prosedürleri, kredi kartı dolandırıcılığının tanınması, borsa analizleri, nükleotid dizilerinin sınıflandırılması, konuşma ve metin tanıma ve otonom sistemler” (Pierson, 2021).

Konu, "veri tabanlarında bilgi keşfi" ve "veri madenciliği" ile yakından ilgilidir, ancak esas olarak yeni kalıplar ve düzenlilikler bulmakla ilgilidir. Birçok algoritma her iki amaç için de kullanılabilir. "Veri tabanlarında bilgi keşfi" yöntemleri, "makine öğrenimi" için öğrenme verilerini üretmek veya önceden işlemek için kullanılabilir. Tersine, makine öğrenimi algoritmaları veri madenciliğinde kullanılır. Bu terim, yapay sinir ağlarını kullanan olası öğrenme çeşitlerinden yalnızca biri olan "derin öğrenme" teriminden de ayırt edilmelidir. Verilerden (varsayımsal) modellere yapılan çıkarıma istatistiksel çıkarım denir (Langley, 2011).

Makine öğreniminde, bilgi temsilinin türü ve gücü önemli bir rol oynamaktadır. Bilginin açıkça temsil edildiği sembolik yaklaşımlar ile hesaplanabilir bir şekilde davranmak üzere "eğitilen" ancak öğrenilen çözüm yollarının anlaşılmasına izin vermeyen sinir ağları gibi sembolik olmayan yaklaşımlar arasında bir ayrım yapılır. Burada bilgi örtük olarak temsil edilir. Sembolik yaklaşımlar, önerme mantığı ve yüklem mantığı sistemleri arasında ayrım yapar. İlkinin temsilcileri ID3 ve halefi C4.5'tir. İkincisi tümevarımsal mantıksal programlama alanında geliştirilmiştir. Pratik uygulama algoritmalar aracılığıyla yapılır. Makine öğrenimi alanındaki çeşitli algoritmalar kabaca üç gruba ayrılabilir: “denetimli öğrenme, denetimsiz öğrenme ve pekiştirmeli öğrenme” (Mikut, 2008). Bu sınıflandırma Şekil 2.1 ‘de gösterilmiştir.



Şekil 2.1. Makine öğrenmesi çeşitleri şeması

- **Denetimli öğrenme:** Algoritma kendisine verilen girdi ve çıktı çiftlerinden bir fonksiyon öğrenir. Burada amaç, ağı modelimizi girdi ve çıktılarla birkaç hesaplamadan sonra ilişkilendirme yapacak şekilde eğitmektir.
- **Denetimsiz öğrenme:** Belirli bir girdi kümesi için algoritma, girdileri tanımlayan ve tanınan kategorileri ve korelasyonları içeren istatistiksel bir model oluşturur ve böylece tahminleri mümkün kılar. Kümeleme yöntemleri verileri karakteristik örüntülerle birbirinden ayrılan çeşitli kategorilere ayırmak için kullanılır. Böylece ağ, girdi örüntülerini böldüğü sınıflandırıcıları bağımsız olarak oluşturur. Bu bağlamda önemli bir algoritma, bir modelin parametrelerini, görülen verileri en iyi şekilde açıklayacak şekilde iteratif olarak ayarlayan Beklenti Maksimizasyonu (BM) algoritmasıdır. Bunu, gözlemlenemeyen kategorilerin varlığını varsayarak ve dönüşümlü olarak verilerin kategorilerden birine üyeliğini ve kategorileri oluşturan parametreleri tahmin ederek yapar. BM algoritmasının bir uygulaması, örneğin Gizli Markov Modellerinde (GMM'ler) bulunabilir. Temel bileşen analizi gibi diğer denetimsiz öğrenme yöntemleri kategorizasyondan vazgeçer. Gözlemlenen verileri, büyük ölçüde azaltılmış bilgiye rağmen mümkün olduğunca doğru bir şekilde yansıtan daha basit bir temsile dönüştürmeyi amaçlamaktadırlar (Mahesh ve diğ., 2020).

- **Pekiştirmeli Öğrenme:** Bu öğrenme modelinde, bir ajanın çevresiyle etkileşime girerek belirli bir görevde en iyi performansı elde etmeye çalıştığı modelir. Ajan çevresinden gelen bildirimleri (ödüller veya cezaları) kullanarak öğrenir. Bu süreçte, ajan deneyimlerinden öğrenir, belirli bir durumda hangi aksiyonun daha iyi sonuç verdiğini anlamak için çeşitli stratejiler geliştirir. Bu öğrenme süreci, ajanın aldığı aksiyonlar ve elde ettiği ödüller arasındaki ilişkiyi optimize edene kadar devam eder.

2.2. Makine Öğrenmesi Uygulama Alanları

Bugün yapay zekanın bir alt kolu olan makine öğrenmesi, bilgisayarların donanımsal ve yazılımsal özelliklerinin artmasıyla birlikte hemen her sektörde yoğun bir şekilde kullanılmaya başlandı. Bu alanlara örnek olarak; eğitim, finans ve meteoroloji gibi birçok alanı göstermek mümkündür. Geçmişte elde edilen veriler üzerine uygulanan algoritmalar ve öğrenme modelleri ile gelecekte olabilecek durumların tahminleri ya da verilerin sınıflandırmaları yapılabilmektedir. Bu tahminlerin ve sınıflandırmaların en önemli özelliği, gerçekleşmesi uzun sürebilen ya da gerçekleştikten sonra geri dönülmez sonuçlar üretebilen olayların önceden tahmin edilmesidir. Makine Öğrenmesinin uygulandığı alanlara baktığımızda; Astronomiden hastalık tespitine, veri madenciliğinden enerji sistemlerine, doğal dil işlemeden kredi risk hesaplamasına kadar pek çok alanda kullanılmakta olduğu görülmektedir (Ünsal, 2011).

2.3. Makine Öğrenmesi Modeli

Makine öğrenmesi modelleri, verilerden öğrenme yeteneğine sahip algoritmaları ifade eder. Burada birkaç temel öğrenme modeli bulunur.

- **Regrasyon Modelleri:** Girdi verileri arasındaki ilişkiyi inceleyereksürekli bir çıktı tahimin eder. Bunlara lineer regrasyon ve polinomiyal regrasyon örnek olarak gösterilebilir.
- **Sınıflandırma Modelleri:** Girdi verilerini farklı sınıflara ayırarak, yeni veri noktalarını uygunsınıflara atar. Bu modele örnek olarak, destek vektör makineleri (SVM), karar ağaçları ve k-en yakın komşu (k-NN) gösterilebilir.

- **Kümeleme Modelleri:** Veri noktalarını benzerliklerine göre gruplayarak, içsel yapıları ortaya çıkarır. Bu modellere, k-ortalama kümeleme ve hiyerarşik kümeleme örnek olarak gösterilebilir.
- **Derin Öğrenme Modelleri:** Yapay sinir ağları kullanılarak çok katmanlı ve karmaşık ilişkileri öğrenir. Bu yapıda evrişimli sinir ağları görüntü işleme için ve yinelemeli sinir ağları sıralı veri analizi için sıklıkla kullanılır.

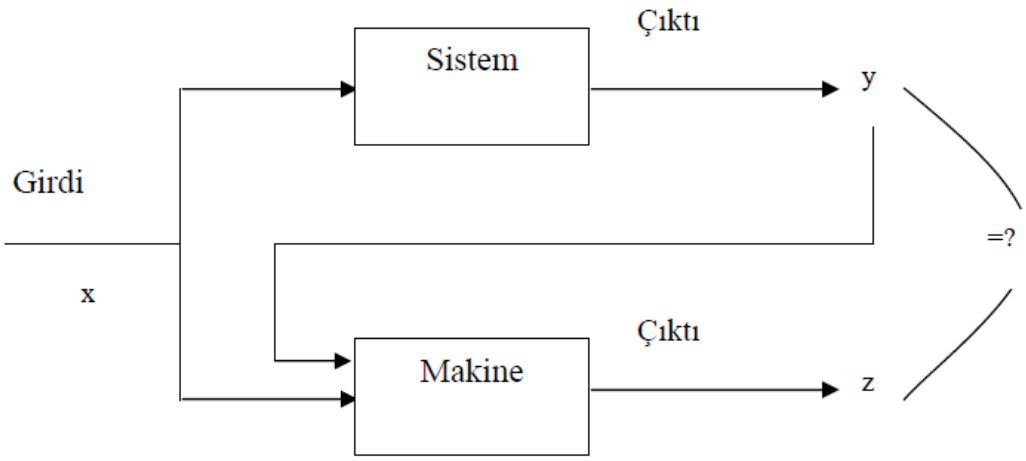
Bu modeller farklı veri türlerine, problemlere ve öğrenme süreçlerine uygun olarak seçilir ve kullanılır. Hangi modelin tercih edileceği, veri setinin yapısı, problem tipi ve kullanılabilir veri miktarı gibi faktörlere bağlıdır.

Şekil 2.2 'de yer alan modelin çalışması aşağıdaki gibi belirtilmiştir.

1. (x;y)'nin bir kümesi alınır, burada x bir girdi vektörü ve y uygun bir çıktıdır.

2. y=f(x) fonksiyonu, önceden bildirilen bir modelin oluşumudur;

   - Modelin kalite ölçümüne, bir kriter tanımlanır
   - Modelin kullanılacağı bir eğitim kümesi oluşturulur
   - Modelin kullanacağı geçerli bir test kümesi oluşturulur (Uzun,2005).



Şekil 2.2. Makine öğrenmesi seması (Uzun, 2005)

2.4. Makine Öğrenmesi Teknikleri

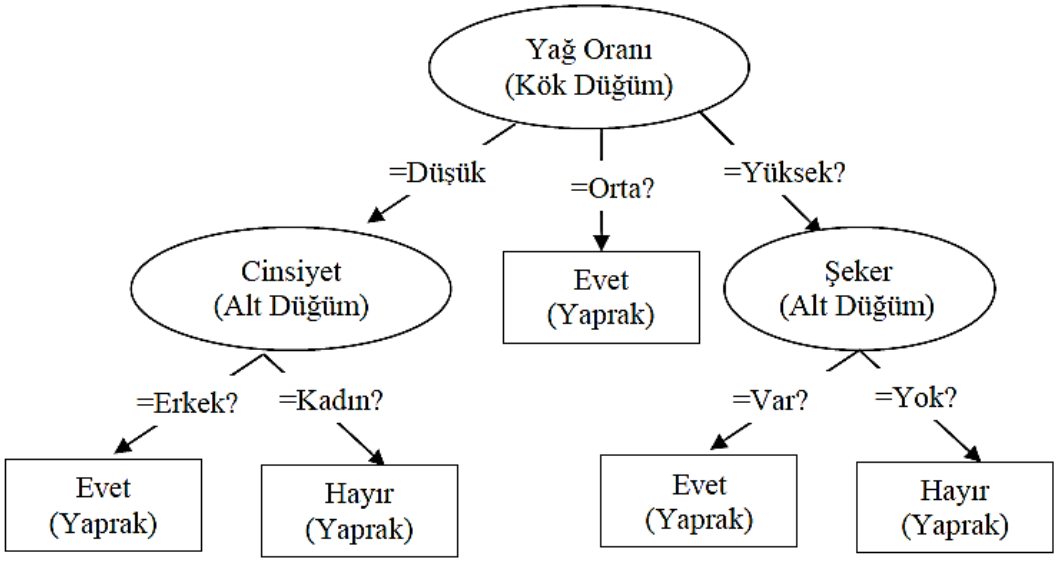
Makine öğrenmesi, var olan problemin çözümüne yönelik olarak bir bilgisayar programının belirli bir görevi yerine getirmek için verilen verileri ve uygun algoritmaları kullanarak kendisini geliştirip sonuçlar üretmesi anlamına gelir. Bu bölümde makine öğrenmesi tekniklerinden dolan ve günümüzde sıkça kullanılan sınıflandırma, kümeleme, birliktelik kuralları tekniklerinde yer alan makine öğrenmesi algoritmaları anlatılmıştır.

Sınıflandırma: Makine öğrenmesi, bilgisayar sistemlerinin verilerden öğrenmesine ve bu verilerden yararlı bilgiler çıkarmasına izin veren bir alanıdır. Bu alanda sınıflandırma, verileri farklı kategorilere ayırma işlemidir. Sınıflandırma, öğrenme algoritmalarının temel öğelerinden biridir ve öğrenme sürecinde en yaygın kullanılan yöntemlerden biridir.

Sınıflandırma, birçok farklı uygulama alanında kullanılır. Örneğin, spam filtreleme, müşteri segmentasyonu, tıbbi teşhis, yüz tanıma, doğal dil işleme gibi birçok uygulama alanında sınıflandırma yöntemleri kullanılır.

Sınıflandırma, öğrenme algoritmaları tarafından verilerin farklı kategorilere ayrılmasıyla gerçekleştirilir. Veriler genellikle etiketli veya etiketsiz veriler olarak iki kategoriye ayrılır. Etiketli verilerde, verilerin her bir örneği bir etiketle belirlenir. Bu etiketler önceden belirlenmiş bir kategoriye ait olabilir veya uzmanlar tarafından belirlenir. Etiketsiz verilerde ise verilerin her bir örneği belirli bir kategoriye ait değildir ve öğrenme algoritmalarının bu verileri analiz etmesi ve kategorilere ayırması gerekmektedir.

Sınıflandırma algoritmaları, verilerin farklı kategorilere ayrılması için farklı yöntemler kullanır. En yaygın kullanılan sınıflandırma yöntemleri arasında Karar Ağaçları, K-En Yakın Komşu (KNN), Destek Vektör Makineleri (SVM), Yapay Sinir Ağları (ANN) ve Doğrusal Regresyon yer almaktadır. Bu yöntemler, verilerin doğru bir şekilde sınıflandırılması için farklı kriterler kullanır ve farklı avantajlara sahiptir. Bahsedilen sınıflandırma modellerinden biri olan karar ağaçları modeli Şekil 2.3 ‘te gösterilmektedir.



Şekil 2.3. Karar ağacı modeli örneği

Kümeleme: Kümeleme modeli, veri madenciliği ve makine öğrenmesi alanlarında kullanılan bir tekniktir. Temel amacı, verileri belirli özellikleri temel alarak birbirine benzer gruplara (kümeler) ayırmaktır. Bu sayede verilerin daha anlaşılır hale gelmesi, benzer özelliklere sahip verilerin bir arada bulunması, veriler arasındaki ilişkilerin daha iyi anlaşılması gibi faydalar sağlanabilir.

Kümeleme modelleri, çeşitli yöntemlerle oluşturulabilir. Bu yöntemler arasında hiyerarşik kümeleme, k-ortalama kümeleme ve yoğunluk tabanlı kümeleme gibi yöntemler bulunur. Bu yöntemlerin her biri, verilerin farklı özelliklerine göre kümeleme yapar.

Örneğin, bir marketin müşteri verilerini kullanarak müşterileri belirli özelliklere göre kümelere ayırmak mümkündür. Bu sayede, belirli bir kümeye ait müşterilerin ortak özellikleri (yaş, cinsiyet, gelir seviyesi vb.) anlaşılabilir ve bu özelliklere göre pazarlama stratejileri belirlenebilir.

Birliktelik Kuralları: Birliktelik kuralı, bir veri kümesindeki öğeler arasındaki ilişkileri tespit etmeye çalışır. Bu teknik, genellikle alışveriş sepeti analizi gibi uygulamalarda kullanılır. Alışveriş sepeti analizi, müşterilerin birbirleriyle hangi ürünleri satın aldığını belirlemek için kullanılır. Bu analiz, birliktelik kuralı kullanılarak gerçekleştirilir.

Birliktelik kuralı, veri kümesindeki öğeler arasındaki ilişkiyi sıralar ve bu ilişkileri belirli bir sıklıkta gerçekleşen birliktelikler olarak ifade eder. Bu birliktelikler, "destek" ve "güven" olarak adlandırılan iki ölçüt kullanılarak değerlendirilir. Destek, birliktelik kurallarının kaç kez gerçekleştiğini belirtirken, güven, birliktelik kurallarının ne sıklıkla gerçekleştiğini belirtir.

Birliktelik kuralı, özellikle büyük veri kümelerindeki öğeler arasındaki ilişkileri keşfetmek için çok kullanışlı bir tekniktir. Bu teknik, müşteri davranışlarının analizi, web sayfası tavsiyeleri ve reklam hedefleme gibi birçok alanda kullanılmaktadır.

3. DERİN ÖĞRENME

Bugün dünyada teknoloji alanında çalışacak olan insanlarda en çok aranan yeteneklernden biri de yapay zekâ, makine öğrenmesi ve derin öğrenme bilgileridir. Çünkü çok fazla alanda kullanıyor ve kullanılan alanların artarak devam edeceği de veri bilimi uzmanları tarafından öngörülmektedir.

Derin öğrenmeyi anlayabilmek için öncelikle, yapay zekâ mantığını ve makine öğrenmesi ile olan temel farkları hakkında bilgi sahibi olmak gerekir. Bu bölümde sonraki bölümler için konunun temel taşları ele alınarak, derin öğrenmede nasıl kullanılabileceğinden bahsedilecektir.

3.1. Derin Öğrenme Nedir?

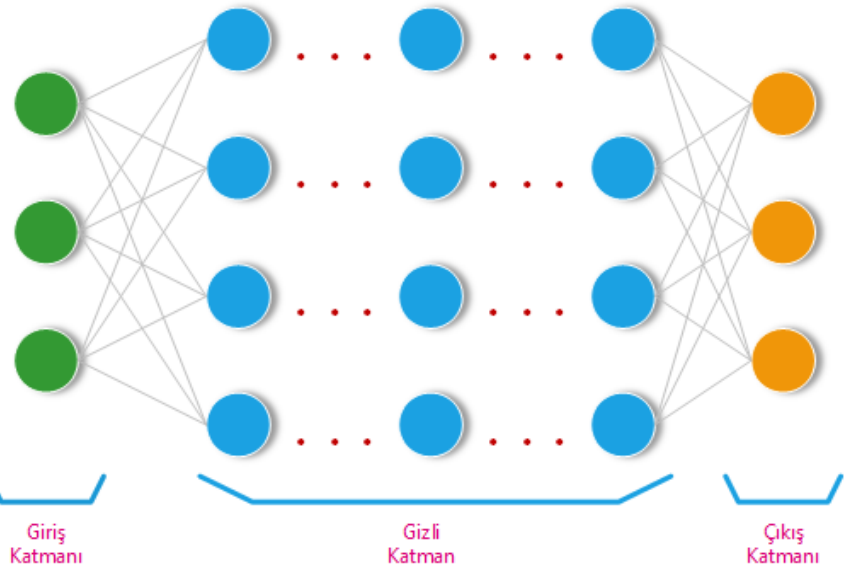
Bugün derin öğrenme kavramından bahsedildiğinde öncelikli olarak bir görüntü sınıflama veya nesne tanıma problemi gelmektedir. Örneğin elimizde kedi ve köpek olarak iki görselimiz olsun. Biz bu görsellere baktığımızda kedimi yoksa köpek mi olduğuna karar verirken, beynimizde kediye dair birtakım özellikleri (kullak tipi, tüy yapısı, yüz tipi vs.) düşünüp elimizdeki görüntüde bu özelliklerin olup olmamasını baz alarak karar veriyoruz. Aynı değerlendirmeyi köpek ya da başka nesneler içinde yaparak karar veriyoruz. Yapay sinir ağları ve makine öğrenmesi gibi modellerde bu özelliklere ihtiyazımız oluyordu.

Derin öğrenmede yapay sinir ağlarından farklı olarak katmanlar arasında bu öznitelikler kendiliğinden öğreniliyor. Öğrenme işlemi tasarladığımız modelimize uyguladığımız (3x3, 5x5 vs.) filtreler sayesinde olur. Modelimize uyguladığımız her filtre bir öznitelik çıkarıcı olarak işlem yapıyor. Bu işlem temelde elimizde bulunan verinin kendisinden öğrenmeye dayalı bir yapıdır. Örneğin elimizde bir görselimizde varsa, bu görsele ait kenar, köşe ve ışık bilgileri ya bizim tasarlağımız bir filtre ile ya da hazır filtrelerle (sobel, prewitt, robert, gabor vs.) öğrenilebilir.

3.2. Derin Sinir Ağları

Derin sinir ağları (deep neural networks), yapay sinir hücrelerinin (nöronlar) katmanlar halinde bir araya getirilerek oluşturulan bir tür yapay sinir ağı yapısıdır. Bu yapı, karmaşık problemleri çözmek ve veri üzerinde işlem yapmak için kullanılır.

Her katman, girdi verilerini işleyip daha soyut temsilleri çıkararak bilgiyi hiyerarşik bir şekilde öğrenir.

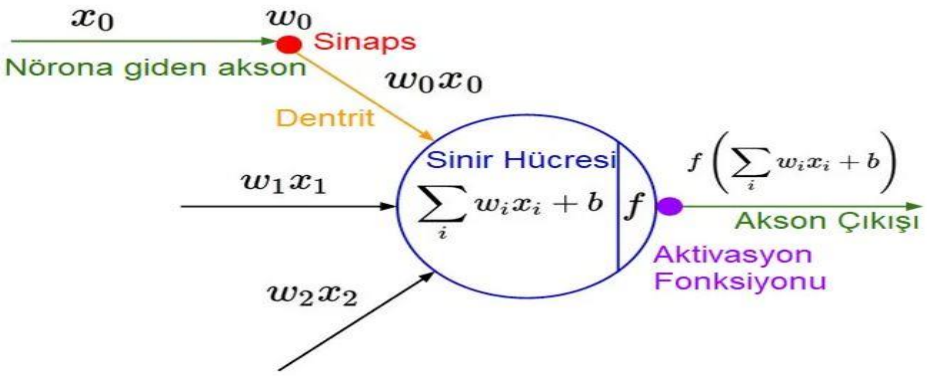


Şekil 3.1. Derin sinir ağı örneği

Burada:

- Giriş Katmanı, veri özelliklerini temsil eder.
- Gizli Katmanlar, verileri daha soyut temsiller haline getirerek öğrenir.
- Çıkış Katmanı, sonuçları üretir.
- Toplam katman sayısı hesabında giriş katmanı sayılmaz.

Her bir nöron, girdileri ağırlıklarla çarparak aktivasyon fonksiyonuna gönderir. Bu, nöronun belirli bir özellik veya deseni ne kadar "etkin" olarak temsil ettiğini belirler. Yapılan bu işlem sırasında aktivasyon fonkisonu kullanırız. Kullanılan fonkisyon doğrusal olmayan problemleri tanımalarına yardımcı olur. Bu sorunların çözümü için, Sigmoid, ReLU (Rectified Linear Activation), Tanh gibi farklı aktivasyon fonksiyonları kullanılabilir. Burada seçilen aktivasyon fonkiyonu zaman maliyetini minimuma indirmek için kolay türevlenebilir olmalıdır. Aktivasyon fonskiyonun çalışma prensibi Şekil 3.2 ’de verilmiştir.



Şekil 3.2. Aktivasyon fonkisyonunun genel yapısı

Derin öğrenme modellerinde öğrenme geri yayılım esnasında yapılır. Her iterasyondan sonra üretilen değer gerçek değerle karşılaştırılır ve bir kayıp (loss) değeri hesaplanır. Kayıp değerinin sıfır çıktığı döngülerde ya ağımız ezberleme yapıyor ya da optimizasyon sorunu yaşıyor olabilir. Optimizasyon sorunu yaşaması durumunda performansını artırmak için Gradien Descent gibi bir algoritma kullanılabilir. Derin sinir ağları ezberlemeye (overfitting) de yatkın olabilir. Bu sorunun çözümü içinse genelde veri bölme ya da azaltma tekniklerinden olan dropout gibi bir regularizasyon yöntemi kullanılarak ağın genelleme yeteneği artırılır.

3.3. Derin Sinir Ağlarında Parametre ve Hiper Parametre Kullanımı

Genel olarak parametreler, modellerde her zaman bulunması ve hesaplanması gereken ağırlıklar ve bias değerleridir. Hiper parametreler ise bizim karar vereceğimiz ve parametrelerin heseplanmasına katkı sağlayan, modelin performansını direk olarak etkileyen değerlerdir. Derin öğrenmede çok sayıda hiper parametre vardır. Literatürde hiperparametreleri bulan çok sayıda algoritmalar ve arayüzler var. Örneğin,

Parametreler:

$W^{[1]}$,$b^{[1]}$,$W^{[2]}$,$b^{[2]}$…

W:Ağırlıklar,

b:Bias değerleri

Hiper Parametreler:

- ✓ Öğrenme Oranı (Learning Rate)
- ✓ İterasyon sayısı
- ✓ Gizli katman sayısı
- ✓ Aktivasyon fonkisyonu seçimi
- ✓ Küme boyutu ve diğer düzenleme (regularization)yöntemleri

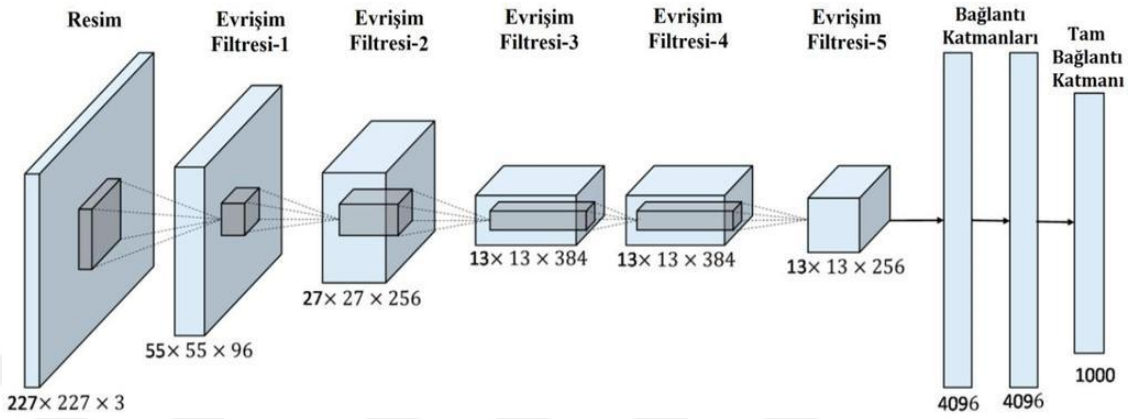
3.4. Evrişim İşlemi

Evrişim işlemi (convolution), özellikle görüntü ve sinyal işleme gibi alanlarda kullanılan matematiksel işlemdir. Evrşim işleminin genel yapısı şematik olarak Şekil 3.3 'te verilmiştir. Temel olarak, bir filtre veya kernel adı verilen bir matrisin, bir girdi matrisi (örneğin bir görüntü) üzerinde kaydırılması işlemidir. Evrişim işleminin amacı, verilerdeki desenleri tespit etmek, özellikleri çıkarmak ve önemli bilgileri vurgulamaktır. Özellikle görüntü işlemede, kenarlar, köşeler, doku gibi temel desenleri tespit etmek için kullanılır.

- ✓ **Özellik Çıkarma:** Evrişim işlemi, girdi verilerinden önemli özellikleri çıkararak veriyi daha temsilci hale getirir. Bu, daha sonra gelen katmanlarda daha etkili öğrenmeyi sağlar.
- ✓ **Parametre Paylaşımı:** Evrişim işlemi, aynı filtreleri farklı bölgelere uygulayarak parametre paylaşımını kullanır. Bu, ağın daha az parametre gerektirmesine ve daha etkili öğrenmesine yardımcı olur.
- ✓ **Davranış İnvariyansı:** Evrişim işlemi, girdi verilerindeki küçük kaymaları (translasyonlar) tolere edebilir. Bu sayede nesnelerin farklı pozisyonlarda bulunabileceği durumları daha iyi işleyebilir.
- ✓ **Hiyerarşik Temsil:** Evrişim işlemi, katmanlar halinde uygulanarak verinin farklı düzeylerdeki özellikleri daha soyutlamaya yardımcı olur. Bu da ağın daha karmaşık yapıları ve ilişkileri öğrenmesine olanak tanır.

Sonuç olarak, evrişim işlemi, derin öğrenme modellerinde veri işleme ve özellik çıkarma süreçlerini optimize etmek için kullanılır. Görüntü işleme, ses işleme ve

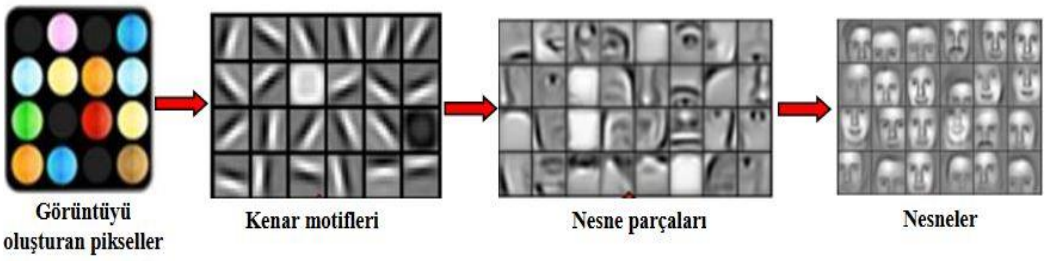
doğal dil işleme gibi alanlarda başarılı sonuçlar elde etmek için vazgeçilmez bir bileşendir.



Şekil 3.3. Evrişim işleminin genel yapısı

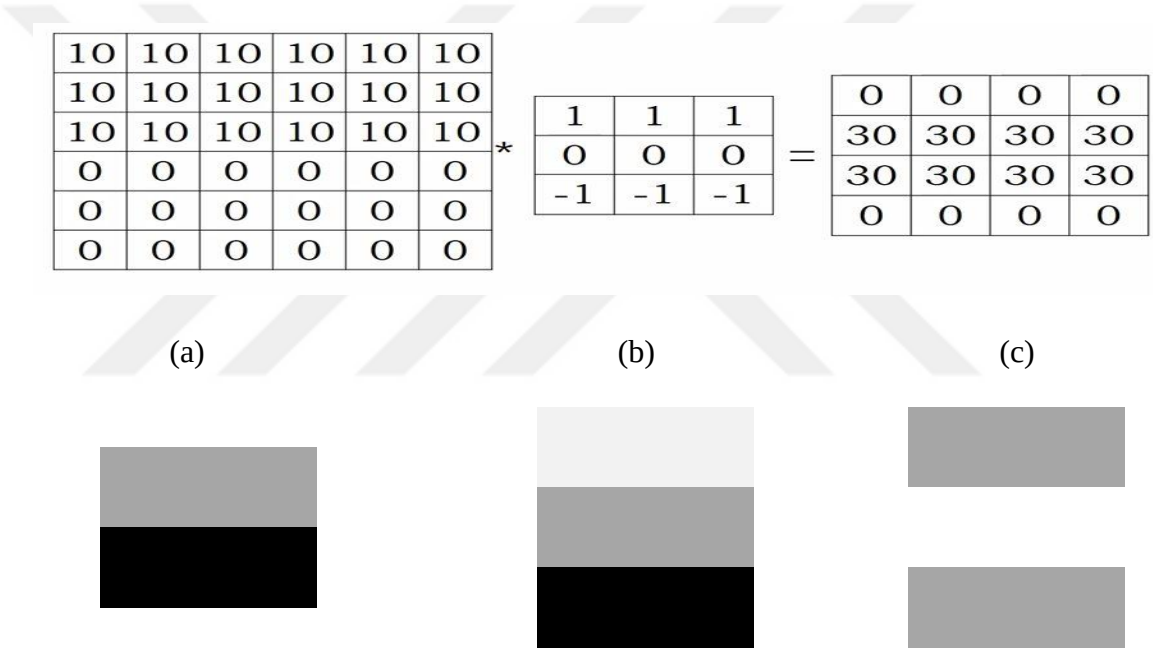
3.5. Kenar Bulma

Kenar bilgileri görüntüden elde edilen öznitelikler arasında en çok ihtiyaç duyulan bilgilerden biridir çalıştığımız görüntü üzerinde çok temel bilgileri bulmamızı sağlar. Giriş bilgisinin yüksek frekanslı bölgelerini simgelemektedir. Bu bilgileri elde etmek için dikey, yatay ya da farklı rotasyonlarda birtakım filtrelerden yararlanılır. Yapılan işleme evrişim denir. İşlem çıkışında yüksek frekanslı yeni görüntünün kenar bilgileri tespit edilmiş olur. Geleneksel yöntemlerde hazır olarak kullanılan (sobel, gabor, prewitt, ropert vs.) filtrelerle bu işlemleri yaparken, derin öğrenmede özelliklede evrişimli sinir ağlarında böyle bir şey yapmamıza gerek yoktur. Derin öğrenmede evrişim işlemi boyunca bu işleri ağımız kendisi çıkarır. Evrişim katmanında bulunan dikey ve yatay kenarları bulup bunları birleştirdiğimiz zaman nesnenin dış görüntüsünü (shape) yavaş yavaş elde etmeye başlarız. Kenar bulma işleminin genel yapısı şematik olarak Şekil 3.4 ’te verilmiştir.



Şekil 3.4. Evrişimle kenar bulma örneği

Kenar bulma işlemide evrişim işleminin benzeridir. Farklı olan tarafı bu iş için elimizde Şekil 3.5 ‘te görüldüğü gibi özel bir matrisin olmasıdır.



Şekil 3.5. Kenar bulma matris örneği

Şekil 3.5. teki bu matris görüldüğü gibi on (10) ve sıfır (0) değerlerinden oluşuyor. On ile sıfırı ayıran çizgi bize bir kenar bilgisi veriyor. Çünkü yüksek frenslı bölgenin olduğu bölgedir. Bu durumda buralarda hızlı geçişlerin olduğu anlamına gelir. Örneğin; Şekil 3.5 ‘te, (a) ile gösterilen şekle baktığımızda on ile ifade edilen kısımlar açık gri olan bölüm iken sıfır ile ifade edilen yerler siyah olarak gösterilmektedir. Bu iki bölümü ayıran kısım ise bize kenar bilgisini vermektedir.

Örneğimizdeki görüntümüzü 3x3’lük bir kenar bulma filtresi uyguladığımızı düşünelim. Filtremizde negatif sayılarla ifade edilen bölüm en koyu renkli olan bölümleri, sıfır olan yerler orta renli ve pozif olanlar ise açık renkleri temsil

etmektedir. Çünki, görüntüyü piksellerle ifade ettiğimiz zaman küçük değerler koyu renki ve büyük değerler açık rengi temsil ederler. Filtremizi giriş matrisimiz üzerinde işlem tabi tuttuğumuzda ve gerekli matematiksel işlemleri yaptığımızda sonuç olarak 4x4' lük bir çıkış matrisi buluruz. Neden 4x4'lük bir matris bulduk? Genel olarak böyle bir işlem sonucu bulacağımız çıkış matrisi,

Çıkış matrisi = (Giriş matrisinin boyutu – Filtre boyutu)+1 (3.1)

Formülüyle hesaplanır.

3.6. Piksel Ekleme

Görüntü işlemine uyguladığımız evrişim işleminden sonra oluşan boyut farkını; giriş matrisi ile çıkış matrisinin eşit boyutta olmasını istiyorsak giriş matrisine ekstra pikseller eklememiz gerek. Bu işleme piksel ekleme ya da dolgulamak denir. Piksel ekleme işlemi için literatürde sıklıklı kullanılan iki yöntem bulunmaktadır. Birinci yöntem, matrisimizin çevresine ihtiyacımız olduğu kadar çevre ekleyip içini sırfırlarla doldurmak. Giriş matrisimize ekleyeceğimiz çerçeve sayısını ise aşağıdaki formülle buluruz.

Giriş matrisi = (nxn), örnek olması açısından n=6 olsun

Filtremiz =(fxf) ile temsil edilsin ve örnek olarak f=3 olsun

Normalde giriş matrisimize bu filtreyi uyguladığımızda çıkış matrisimizin boyutu (n-f)+1 den 4x4 lük bir çıkış elde ederiz. Bu işlemden sonra görüldüğü gibi boyutumuz azalmış oldu. Biz giriş matrisimizin boyutu ile çıkış matrisimizin boyutu aynı olsun istiyoruz. Eğer piksel ekleme yapacaksak uygulanacak formül ise aşağıdaki gibidir.

Piksel ekleme var ise;

p = ekleme(paddink)

$p = \frac{f-1}{2}$

Çıkış = (n+2p-f+1)x(n+2p-f+1)=(6x6)

Eğer piksel ekleme yoksa ise çıkış =(n-f+1) x(n-f+1) =(4x4)

Bu işlemlerden sonra giriş matrisimize kaç çerçeve ekleneceğini bulmuş olduk. Eklediğimiz bu çerçeveleri neyle ve nasıl dolduracağız. Uygulamalarda en çok kullanılan iki yöntem bulunmaktadır. Birinci yöntem, örneğin giriş matrisimize iki satır ve iki sütun eklemişsek bunların içini sıfır (0) ile doldurmaktır. İkinci yöntem olarak da eklediğimiz ilk çerçeveye hemen yanındaki pikseller kopyalanır. İkinci çerçeveye ise ilk durumdaki matrisin dışardan ikinci satır ve sütunundaki pikseller kopyalanarak doldurma işlemi tamamlanır.

Eğer bu işlem sırasında ilk yöntemi (sıfır ile doldurma) tercih etmişsek, evrişim işlemi sırasında yapacağımız matematiksel işlemler sırasında değeri sürekli aşağıya çekecektir. Çünki evrişimde çarpma ve toplama işlemi yapıyoruz. Böylece elde edilen çıkış matrisinde birbirinden ayrık değerler oluşacak.

Eğer ikinci yöntemi kullanırsak (giriş matrisindeki değerlerin dışa doğru kopyalanması), eklenen piksellerle diğer pikseller arasında sayısal olarak çok fark olmayacağından yapılacak olan evrişim işlemini sonrasında elde edilecek çıkışlarda görüntüye dair daha yakın bilgiler elde edilmiş olacaktır. Bu işlemin dezavantajı ise, her adımda toplama ve çarpma yapılacağından işlem yükümüz çok olacaktır. Bu da daha yavaş çalışan bir evrişim işlemine neden olacaktır. Bununla birlikte elde edeceğimiz çıkış matrisi daha doğru bir matris olacaktır.

3.7. Adım Kaydırma

Evrişimli sinir ağlarının temeli olan evrişim işlemini yaparken dikkate aldığımız bir diğer önemli özellik ise adım kaydırmadır. Adım kaydırma, giriş matrisimiz üzerinde kullandığımız filtremizin kaç piksel aralıkla kaydırdığımız işlemin adıdır. Adım kaydırma işleminin genel yapısı şematik olarak Şekil 3.6 ’da verilmiştir. Bu durumu aşağıdaki gibi bir örnekle açıklamak gerekirse;

Şekil 3.6 ’da piksel işleme yapılmış (7x7) boyutunda bir matris görülmektedir.

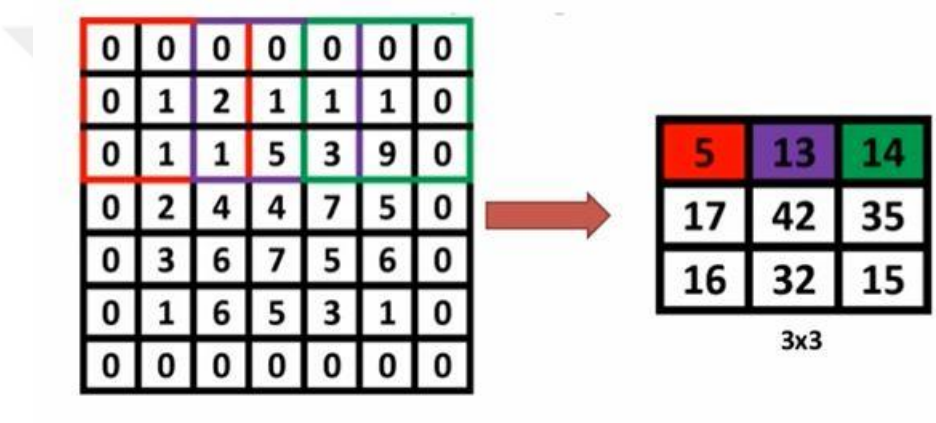
Giriş matrisi n=5x5

Uygulanan filtre f=3x3

Adım kaydırma s=2 (iki adımda bir)

Pikel ekleme p=1 (1x1 lik 0 (sıfır) lardan oluşmuş dış çerçeve)

Burada uyguladığımız filtremiz bir (1) lerden oluşmuş bir filtre olsun. Filtremizi sol üst köşeye yerletirip evrişim işlemi uyguluyoruz. Yapılan işlem sonucunda elde ettiğimiz değer Şekil 3.6 'da görüldüğü üzere beş(5) tir. Sonra iki adım sağa kaydırıp filtremi yeni değerlele evrişim işlemine sokuyorum. Elde ettiğim sonucum yine Şekil 3.6'da görüleceği üzere on üç(13) tür. Evrişim işlemini sağa ve aşağıya kaydırarak tüm matrisimize uygularız. Bu işlem sonucunda (3x3) boyutunda bir matris elde ederiz.



Şekil 3.6. Kaydirma işlemi uygulanmış matris örneği

Elde ettiğimiz (3X3) boyutundaki bu matrisi nasıl elde ettiğimizi hatırlayacak olursak;

$\left(\frac{(n+2p-f)}{s}+1\right) x \left(\frac{(n+2p-f)}{s}+1\right)$ formulü ile hesaplayacak olursak, (3.2)

$$\left(\frac{(5+2.1-3)}{2}+1\right) x \left(\frac{(5+2.1-3)}{2}+1\right) = (3)x(3)$$

Görüldüğü üzere adım kaydırma işlemi piksel eklemiş bile olsak çıkış matrisimizin boyutunu küçültmektedir. Eğer giriş matrisimiz ile çıkış matrisimizin boyut farkı çok

olmasın ya da aynı olsun istiyorsak, adım kaydırma aralığını ne kadar büyük tutarsak piksel ekleme sayımızda o kadar büyük olmalıdır.

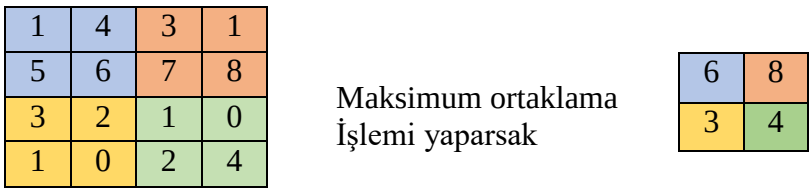
Burada örnek olarak kullandığımız matrisler iki boyutlu olduğundan gri seviye bir görüntüyü temsil etmektedir. Gerçek hayatta bizim gördüğümüz görüntüler genellikle renklidir. Buda üç kanallı (kırmızı, yeşil, mavi) matris yapılarından oluşur.

3.8. Ortaklama İşlemi

Literatürde ortaklama işlemi için havuzlamada olarak kullanılsa da, en yaygın kullanım şekli ortaklama işlemidir. Ortaklama işleminin genel yapısı şematik olarak Şekil 3.7 ’de verilmiştir. Bu işlemin kulllanılan üç yöntemi vardı. Bunlar; maksimum ortaklama, minumum ortaklama ve ortalama ortaklamadır. Bunların arasında en yaygın olarak kullanılan maksimum ortaklama yöntemidir. Maksimum ortaklama işlemi aşağıdaki gibi yapılmaktadır.

Filtre boyutu f=2

Adım kaydırma s=2

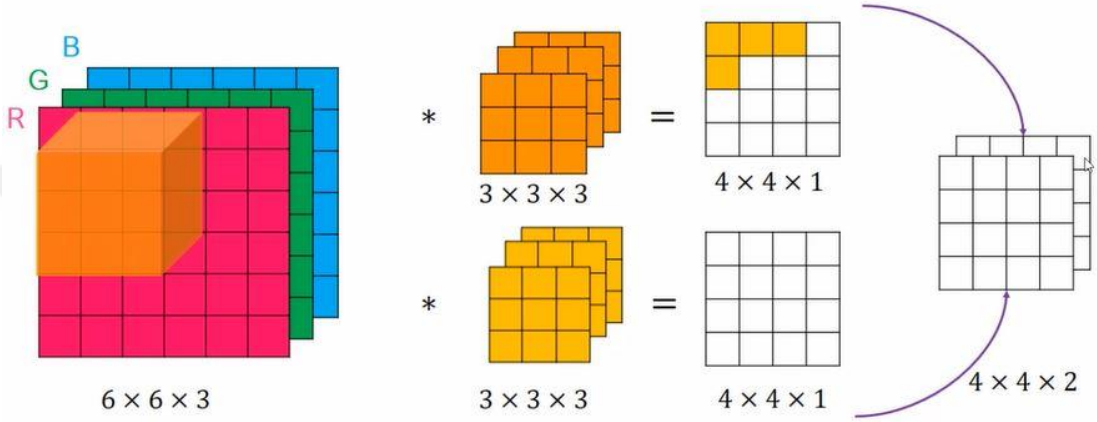


Şekil 3.7. Maksimum ortaklama işlemi

Giriş matrisimiz üzerine filtremizi yerleştirdikten sonra, filtemizin bulunduğu altında kalan alanda yer alan en büyük sayıyı alırız. Sonar bu işlemi adım kaydırma miktarı kadar sağa ve aşağıya giderek matrisimizin tamına uygularız. Görüldüğü üzere giriş matrisimizdeki her dört pikseli çıkış matrisimizde bir pikselle bir pikselle ifade etmiş oluyoruz. Bu aslında bir tür boyut azaltma işlemidir.

3.9. Çok Kanallı Evrişim İşlemi

Evrişim işleminin nasıl yapıldığı konusunda yukarıda değinildi. Bu işlemi yaparken adım kaydırma, piksel ekleme ve ortaklama işlemlerinden bahsedildi. Şu ana kadar bu işlemleri hep iki boyutlu gri seviye matrisler üzerinde gördük. Yani tek kanallı görüntüdür.



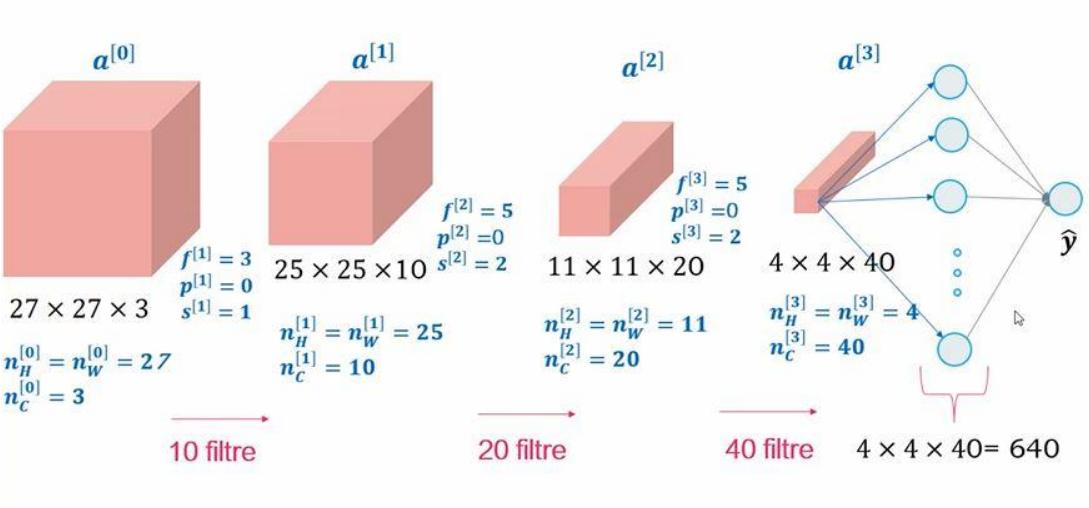
Şekil 3.8. Çok kanallı evrişim ağı

Çok kanallı görüntünün en basit hali kırmızı, yeşil ve maviden oluşan, günlük hayatta fotoğraf çekerken kullandığımız görüntü tipidir. Şekil 3.8. de görüldüğü gibi elimizdeki görüntü 6x6 boyuntunda, renkli ve üç kanallı bir görüntüyse 6x6x3 şeklilnde kanal sayısınıda ekleyerek ifade ederiz. Bu görüntüyü üç tane 6x6 boyutunda matristen oluşmuş bir yapı olarak düşünebiliriz. Bu görüntüye bir evrişim işlemi uygulayacaksak, filtremizde üç boyutlu olmak zorundadır. Şekil 3.8. de görüleceği üzere 6x6x3 boyutundaki matrisimize 2 adet 3x3x3 lük filte uyguladık. Bu arada uygulanan her bir filtre bir özniteliği temsil etmektedir. Filtre çıkışlarında elde ettiğim değerler 4x4x1 boyunda matrisdir. Burada 4 değeri, (giriş matrisinin boyutu – filtre matrisinin boyutu+1) formulü ile hesaplanarak bulundu. Bu işlem iki filtremiz olduğundan iki kez tekrar ediliyor. Sonuçta ise 4x4x2 boyutuda bir matris elde edilliyor. Eğer bir matris birden fazla kanaldan oluşuyorsa bu matrise literatürde tensor denilmektedir. Bu tensor yapısı bir görüntüyü simgelemiyor olabilir ama bir öznitelik çıkarma işlemi gerçekleştirmiş oluyor. Örneğin ilk filtremiz dikey kenarı,

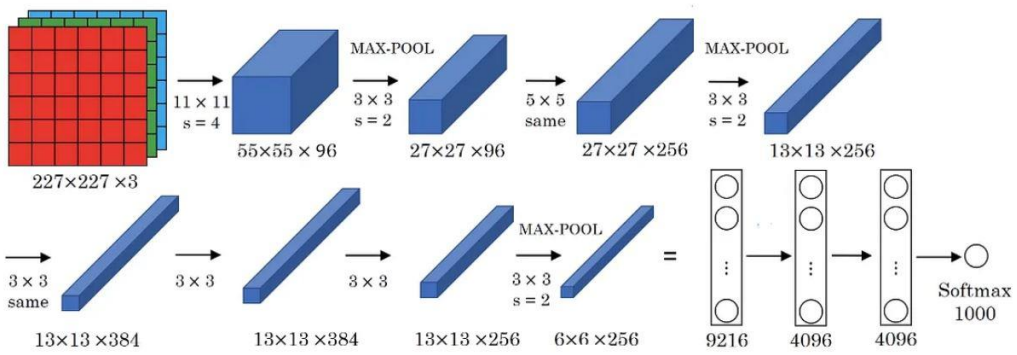
ikinci filtremiz yatay kenarı temsil ediyorsa, elde ettiğimiz tensor hem dikey hem de yatay kenarı ifade eder.

3.10. Evrişimli Sinir Ağı

Şekil 3.9 'da görültüğü gibi, 27x27x3 şeklinde ifade edilen bir görüntümüz olsun. Giriş katmanına $a^{[0]}$ ismini verdik. Her bir hesabımızın sonunda $a^{[1]}$, $a^{[2]}$ ve $a^{[3]}$ katmanlarındaki değerleri elde edeceğiz. Öncelikle bir filtre seçiyoruz. Örneğimizde ilk kullandığımız filtre için $f^{[1]} = 3$, piksek ekleme $p^{[1]}=0$ ve adım kaydırma değerimiz $s^{[1]}=1$ olarak verilmiştir. Bu filtreden on tane kullanılmıştır. Giriş görüntüsünün boyutlarına bakacak olursak, genişlik ve yükseklik derğerleri $n^{[0]}_H = n^{[0]}_W = 27$ ve kanal sayısıda $n^{[0]}_3 =3$ olarak verilmiştir. Bir sonraki adıma geçerken uygulamam gereken formülü $\left(\frac{(n+2p-f)}{s}+1\right)$ 'dir. Elde edilen yeni matris(tensor) 25x25x10 luk olacaktır. Burada eklenen on rakamı uygulamış olduğumuz on filtreyi temsil etmektedir. Böylece ilk katmandaki yapacağımız işlem bitmiştir. Sonraki katmanlarda da filtre, piksel ekleme ve adım kaydırma değerlerini değiştirerek aynı işlemi tekrar ederiz. Tüm bu işlemeleri yaparken bir aktivasyon fonkisyonu (relu, sigmoid vs.) ve bir bias değeri işleme tabi tutuyoruz. Son katmandan sonra tam bağlantı katmanı dediğimiz (full connect) katmanda vektör şeklinde bir yapı olur. Bu katmandan sonra sonuçlar bir çıkış nöronuna ulaştırılır. Burası ise elde ettiğimiz kestirim değerimiz olur. Örneğin bir nesne sınıflandırma işlemi yapıyorsak, sınıflandırma değerimiz olacak.



Şekil 3.9. Evrişimli sinir ağı örneği



Şekil 3.10. Evrişimli sinir ağı örneği2

3.11. Resnet-50 Modeli

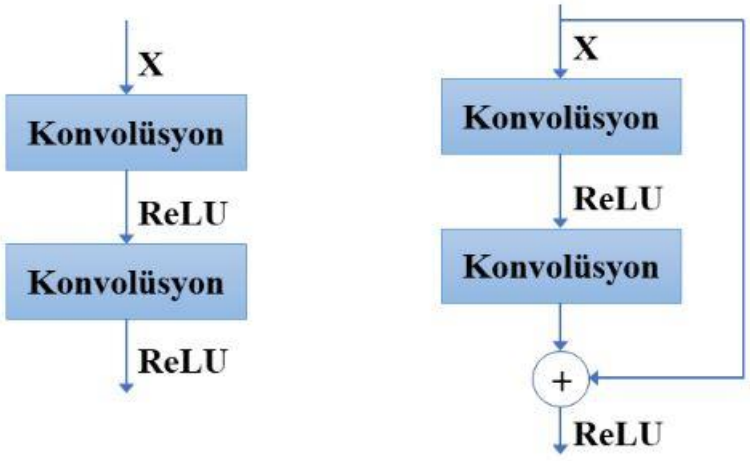
ResNet-50, "Residual Network" (ResNet) adı verilen bir derin öğrenme modeli ailesinin bir üyesidir. Bu aile, 2015 yılında Kaiming He ve arkadaşları tarafından Microsoft Research'te geliştirilmiştir. ResNet, bilgisayar görüşü (computer vision) görevlerinde büyük başarı elde etmiş ve ImageNet gibi büyük veri setlerindeki görüntü sınıflandırma görevlerinde özellikle etkili olmuştur. ResNet, diğer derin sinir ağlarına kıyasla daha derin ağların eğitilmesini kolaylaştırmak için geliştirilmiş bir mimariye sahiptir. Ana yenilik, "residual bloklar" olarak adlandırılan özel bir yapıdır. Bu bloklar, daha önceki katmanların çıktılarını (giriş) son katmana ekler. Bu, ağın daha derin hale getirilmesine olanak tanırken, aynı zamanda aşırı uçlarını (vanishing gradients) çözer ve eğitimi daha verimli hale getirir.

ResNet-50 modeli özellikle 50 katmanlı bir derin ağdır ve ortalama olarak 3.9 x 10^9 parametreye sahiptir. Model aşağıdaki ana bileşenleri içerir:

- ✓ **Giriş Katmanı**: Resimlerin RGB renk kanallarını (genellikle 224x224 piksel) kabul eder.
- ✓ **Beş İçerikli (Convolutional) Grup**: Bu gruplar, birbirini takip eden beş residual blok içerir. Her biri farklı filtre sayılarına ve evrişim çekirdek boyutlarına sahip olabilir.

- ✓ **Global Ortalama Havuzlama (Global Average Pooling)**: Son residual blok çıktılarını alır ve bunları global ortalama havuzlama katmanına ileterek her bir özellik haritasını tek bir değere dönüştürür.
- ✓ **Tam Bağlantı Katmanı (Fully Connected Layer)**: Global ortalama havuzlama sonucunu, sınıflandırma yapmak için kullanılır. Genellikle 1000 farklı sınıfı tanıyabilen bir sınıflandırma katmanına sahiptir.
- ✓ **Softmax Katmanı**: Sınıflandırma sonuçlarını olasılık dağılımlarına dönüştürür.

ResNet-50 modeli, çeşitli görsel görevlerde önceden eğitilmiş bir ağırlık modeli olarak kullanılabilir veya özelleştirilerek belirli görevler için eğitilebilir. Bu model, özellikle nesne tespiti, görüntü sınıflandırma ve transfer öğrenme görevlerinde popüler bir seçenektir. Şekil 3.11'de standart bir Evrişimli Sinir Ağı(ESA) ile ResNet mimarilerinde kullanılan kısa yol bağlantıları görülmektedir.



Şekil 3.11. Standart ESA (sol); ResNet mimarilerinde kullanılan kısayol bağlantıları (sağ)

4. KUANTUM MAKİNE ÖĞRENMESİ

Kuantum Makine Öğrenimi (QML), Kuantum Fiziği (QP) ve Makine Öğreniminin (ML) öğrenilmesine yönelik bütünleştirici bir yaklaşım olarak bilinmektedir. Bu bölümde, kuantum makine öğrenimi ile ilgili temel fikirlerin ve özelliklerin bir taslağı ortaya konmaktadır. Kuantum algoritmalarının farklı yönleri, kuantum takviyeli öğrenme, kuantum tavlamanın temel özellikleri, son olarak kuantum sinir ağlarının QML yönüne ışık tutacak şekilde ilerlemesi bu bölümde açıklanmıştır.

4.1. Kuantum Makine Öğrenmesi

Makine öğrenimi, son teknolojilerde gelişmekte olan bir disiplin haline gelmiştir. Hesaplamalı biyoloji, bilgisayarla görme, bilgisayar güvenliği ve diğer çeşitli alanlarda kullanılmaktadır. Veri analizi (DA), modern endüstride eşit derecede önemli bir parçadır. Makine öğrenimi ve DA istatistiksel yöntemler uygulayarak verileri analiz eder ve gözlemlenen verilerin analizi temelinde bilgisayarlara öğrenme yetenekleri sağlar. Öğrenme tarzı temelinde, Makine öğrenimi algoritmaları (MLA) temel olarak farklı gruplara, yani denetimli öğrenme (SL), denetimsiz öğrenme (UL) ve yarı denetimli öğrenme (semi-supervised learning-SSL) şeklinde ayrılır. ML tekniklerini kullanmanın en önemli eksikliklerinin, özellikle büyük miktarda veri içeren hesaplama süresi ve depolama olduğu bilinmektedir. Bunlara ek olarak, mevcut derin öğrenme algoritmaları kullanıldığında, eğitim süresi daha da uzun olabilmektedir. Sonraki nesilde araştırmacılar, yukarıda bahsedildiği gibi depolama ve hesaplama süresini azaltacak kadar akıllı olabilen kuantum bilgisayar yönteminin gücünü kullanarak uygun bir alternatif elde etmişlerdir.

Birçok makine öğrenimi problemi verileri matrislerle ifade ederek matris işlemlerini çözmek için doğrusal cebir kullanır. Kuantum hesaplama (QC), klasik ML görevlerini kayıtsız şartsız iyileştiren çeşitli doğrusal cebir hesaplamalarını daha hızlı hale getirebilir (Rebentrost ve diğ., 2014). Optimizasyon için sayısal yöntemler, söz konusu optimizasyon prosedürlerinin hesaplamalarını iyileştirmeyi amaçlayan çok beğenilen bir araştırma alanıdır. Klasik optimizasyon gibi, QC'nin bir dalı olan kuantum optimizasyonu da söz konusu teknikleri daha da geliştirmeye çalışır. Bu türden iki ünlü yöntem Kuantum Gradyan İnişi (QGD) (Kerenidis ve Prakash, 2017) ve Kuantum Yaklaşık Optimizasyon Algoritmasıdır (QAOA). Bu yöntemler

Kuantum Boltzman Makineleri (Amin ve diğ.,2018) gibi kuantum sinir ağlarında (QNN) verimli bir şekilde uygulanmaktadır. Son zamanlarda, çok yeni disiplinler arası araştırma alanı olan kuantum makine öğrenimi, makine öğrenimi teorisini kuantum hesaplamanın özellikleriyle birleştirmek amacıyla ortaya çıkmıştır. Kuantum makine öğrenimi (QML), farklı problemleri yüksek etkinlikle çözmek için kuantum süperpozisyonu ve kuantum dolanıklığı gibi kuantum özelliklerini uygulayarak makine öğrenimi algoritmalarını (MLA) kuantum atmosferinde (sistemlerinde) uygulamaya yöneliktir (Schuld ve diğ., 2015).

Temel olarak QML, ML'deki mevcut yaklaşımları iyileştirmek amacıyla genellikle verilerden öğrenen kuantum algoritmalarını tanıtmak amacıyla bilgi işlemenin kuantum araştırmasının bir alt disiplini olarak bilinmektedir. Dolayısıyla amaç, MLA'larının esnekliği ve öğrenme kabiliyetinin yanı sıra kuantum bilgisayarların etkisine sahip çeşitli MLA'larının kuantum uygulamalarını geliştirmektir. Sinir ağları (NN), grafik modeller, destek vektör makineleri (SVM) gibi çeşitli makine öğrenimi modelleri için çeşitli kuantum algoritmaları tanıtılmıştır. QML, kuantum perspektifinden öğrenme düşüncesi hakkında daha temel soruları araştırır. Bazı durumlarda QML, ML'yi kuantum bilgisine uygulamak için araştırmacılar tarafından kapsamlı bir şekilde tanımlanmaktadır. QML'de kullanılan örtük metodolojilerin yanı sıra, klasik ML algoritmalarının birkaç kuantum versiyonu bulunmaktadır. Temelde doğrusal sınıflandırma için uygulanan Kuantum Destek Vektör Makineleri (QSVM) bu türün popüler bir örneğidir. Buna ek olarak, boyut indirgeme için popüler bir yaklaşım olan Kuantum Temel Bileşen Analizi (QPCA) (Lloyd ve diğ., 2014), kümeleme ve yoğunluk tahmini için bir başka ünlü yaklaşım olan Kuantum Guassian Karışım Modelleridir (Rahman ve Geiger, 2016). Makine öğreniminin gelişmekte olan bir alt disiplini Kuantum Destek Vektör Makineleri (QSVM) olarak adlandırılmaktadır. Günümüzde kuantum bilgisayarlar, önemli miktarda depolama ve zaman gerektiren DL uygulamaları için kullanılmaktadır. Bu uygulamaların bazı popüler örnekleri Kuantum Boltzmann Makineleri, Kuantum Üretken Çekişmeli Ağlar (Lloyd ve Weedbrook, 2018), Kuantum Evrişimli Sinir Ağları (Cong ve diğ., 2018) ve Kuantum Varyasyonel Otomatik Kodlayıcılardır (Khoshaman ve diğ., 2018). Ek olarak, makine öğrenimi içinde daha aydınlanmış bir alan pekiştirmeli Öğrenme (RL) olarak bilinir. RL, çevreyi keşfederek zaman geçtikçe öğrenme olarak tanımlanabilir.

4.2. Kuantum Pekiştirmeli Öğrenme

Denetimli ve denetimsiz öğrenmenin yanı sıra, pekiştirmeli öğrenme de popüler bir öğrenme yöntemi kategorisidir. SL ve UL'nin aksine RL, girdi-çıktı çiftlerini değerlendirmek için ödül adı verilen skaler bir değer kullanır ve durumlardan eylemlere bir eşleme öğrenmek için çevre ile etkileşime girmek için deneme yanılma politikasını kullanır. 1980 yılından bu yana, RL giderek ML için önemli bir yaklaşım haline gelmiştir. Şimdiye kadar, yapay zekada (AI), özellikle robotikte yaygın olarak uygulanmıştır. Bunun nedeni, on-line adaptasyonda mükemmel bir performans göstermesi ve buna ek olarak, herhangi bir karmaşık doğrusal olmayan sistem için geçerli bir öğrenme yeteneğine sahip olmasıdır (Smart ve diğ., 2004).

Pratik uygulamalarla uğraşırken, keşif stratejisi, yavaş öğrenme hızı gibi bazı karmaşık problemler olabilir; özellikle karmaşık problemlerin ele alınması, durum-eylem uzayı devasa boyutlara ulaşırken ve öğrenilecek parametrelerin sayısı boyutun artmasıyla birlikte üstel olarak artarken gözlemlenebilir. Son yıllarda bu durumla mücadele etmek için çeşitli yöntemler ortaya konmuştur.

RL'yi optimize etmek için farklı öğrenme paradigmaları bir araya getirilmiştir. Smith (Smith, 2002), kendi kendini organize eden harita (SOM) ve kıyaslamalı Q-öğrenme temelinde modelsiz RL'de temsil etmek ve genelleştirmek için yeni bir model önermiştir. Ayrıca, büyük/sürekli durum-eylem uzaylarına sahip problemler için Watkins'in Q-öğrenmesi ile bulanık çıkarım sistemlerinin adaptasyonu da sunulmuştur (Er ve Deng, 2004).

Shor algoritması ve Grover algoritması olarak adlandırılan iki önemli kuantum algoritması tanıtılmıştır. Rigatos ve Tzafestas (2002), amacı bulanık çıkarımı hızlandırmak olan bulanık mantıksal kontrol algoritmasının (FCA) paralelleştirilmesinden faydalanmak için kuantum hesaplama teorisini uygulamıştır. Kuantum evrimsel algoritmalar (QIEA), mevcut evrimsel algoritmaların (EA) performansını artırmak için geliştirilmiştir (Sahin ve diğ., 2005). İlerleyen zamanlarda, Hogg ve Portnov (2000) aşırı kısıtlı doyurulabilirlik ve asimetrik gezgin satıcıya sahip kombinatoryal optimizasyon problemini çözmek için bir kuantum algoritması tanıtmıştır. Son zamanlarda, kuantum arama yöntemi dinamik programlamaya uygulanmıştır (Naguleswaran ve diğ., 2005). Hesaplamanın ruhunu dikkate alan (Dong ve diğ., 2005), kuantum hesaplamanın temel kavramı olan durum

süperpozisyon ilkesi ve paralellikten esinlenerek Kuantum Takviyeli Öğrenme (QRL) kavramını geliştirmişlerdir. QRL, öğrenmeyi hızlandırmak ve simüle edilmiş deneyler sırasında RL'nin sömürülmesi ve keşfedilmesi arasında bir denge sağlamak için geliştirilmiştir.

4.3. Kuantum Tavlama

İstatistiksel mekanikte, kuantum-mekanik dalgalanmalar alanında uygulanan kuantum tavlama (QA), benzetimli tavlamanın (SA) kuantum versiyonudur (Matsuda ve diğ., 2009). Kuantum Stokastik Optimizasyon algoritması olarak da bilinir. SA gibi, kuantum tavlama da zor optimizasyon problemini çözmek için başarıyla uygulanmaktadır. Kuantum hesaplamanın aktif alanında kuantum adyabatik evrim prensibine göre çalışır ve klasik ve kuantum teknolojisi arasında önemli bir hibridizasyondur. QA, bilgisayar bilimi (Choi, 2010), makine öğrenimi (Adachi ve Henderson, 2015), grafik teorisi (Vinci ve diğ., 2014), iletişim (Chancellor ve diğ., 2016), finans (Marzec, 2014), havacılık (Coxson ve diğ., 2014) ve diğer gerçek dünya problemleri gibi farklı alanlarda uygulanmaktadır.

Benzetimli tavlamada, istatistiksel-mekanik bir sistemin sıcaklığa bağlı rastgele yürüyüşü verilen bir optimizasyon probleminin maliyet fonksiyonu olarak kullanılır. Bu fonksiyon çözüm uzayının potansiyel enerji profilini belirler ve termal dalgalanmalar keşfin yerel bir minimumda takılıp kalmasını önler (de Falco ve Tamascelli, 2011). Zaman evrimi sırasında sistemin termal dengeye yakın kalması beklenir. Sıcaklık düşüş hızı yeterince yavaşsa bu gerçekleşebilir ve böylece sonunda en düşük enerjili durum olan sıfır sıcaklık denge durumuna yol açabilir. Genel uygulanabilirliği, makul performansı ve çoğu durumda nispeten kolay uygulanması nedeniyle SA, birçok gerçek hayat uygulamasında etkin bir şekilde kullanılmaktadır. Bir problem, sistemi termal dengeye yakın tutarak kesin çözümü elde etmek için sonsuz uzun bir süre gerektirdiğinde, SA ölçülebilir bir hesaplama süresi içinde yaklaşık çözümü elde etmek için uygulanır. Kuantum dalgalanmaları, QA'da yapay kuantum doğası dereceleri, komütatif olmayan operatörler eklenerek indüklenir. Kuantum dalgalanmalarının gücü, sıcaklığı yavaşça düşürerek temel duruma ulaşmak için kontrol edilir (Morita ve Nishimori, 2011).

Bir optimizasyon probleminin maliyet fonksiyonunun minimizasyonunu bulmak gibi, klasik bir Ising Hamiltonian H0'ın temel durumunu bulmak olarak düşünülebilir

(Lucas, 2014). Farklı türdeki pratik problemler, çok sayıda yerel minimuma sahip maliyet fonksiyonlarına sahiptir. Benzer şekilde, Ising Hamiltoniyenleri klasik spin camlarını anımsatmaktadır (Nishimori, 2001). Bu tür özellikler için, klasik algoritmalar için küresel minimumları bulmak çok zordur. Bu sorun, klasik Ising Hamiltonyeni H0'ı kuantum alanına yükseltme fikri ile aşılabilir. Kuantum mekaniğinin adyabatik teoremine dayanarak, klasik Ising modelinin temel durumu, sistemi bazı başlangıç Hamiltonyeni H1'in temel durumunda biçimlendirerek türetilebilir. Hem teorik hem de deneysel olarak geliştirilmesi kolaydır. H1, H0 ile değiş tokuş yapmayacak şekilde seçilir. Sistem parametreleri yeterince yavaş değiştikçe Hamiltonyen kademeli olarak H1'den H0'a değişir (Amin ve diğ., 2007).

4.4. Kuantum Sinir Ağları

Kuantum sinir ağları (QNN'ler), kuantum mekaniği ilkelerini içeren sinir ağı modellerinin enkarnasyonlarıdır. Hesaplama açısından bakıldığında, bir kuantum sinir ağı sınıfı yapay sinir ağı modellerini birleştirir ve daha sağlam ve verimli modeller geliştirmek için kuantum hesaplamanın özelliklerini içine yerleştirir. Bu çabaların arkasındaki temel felsefe, kuantum paralelliği, girişim ve dolaşıklık özelliklerine başvurarak klasik sinir ağlarının büyük verileri işlemedeki sınırlamalarını aşmaktır. Bununla birlikte, QNN modellerinin çoğu klasik ikili veya McCulloch-Pitts nöronlarını kuantum mekanik ilkeleriyle ortaya çıkan kübitlerle ("quron" olarak da adlandırılır) değiştirmeye çalışır (da Silva ve diğ., 2016).

1995 yılında Subhash Kak ve Ron Chrisley, nöral aktivasyon fonksiyonunun kuantum mekaniksel özdeğer denklemi ile benzerliğini ortaya koyarak bir kuantum nöral model fikrini ortaya atmışlardır. Ajit Narayanan ve Tammy Menneer, kuantum ölçümü uygulandığında istenen durumlara çöken çoklu-evren teorisini kullanarak bir kuantum sinir ağının fotonik bir uygulamasını tanıtmıştır (Narayanan ve Menneer, 2000). O zamandan beri, algılayıcının kuantum versiyonunu bulmak için çok çaba harcanmıştır. Ancak bu yöndeki ilerlemeler, karakteristik nöral doğrusal olmayan aktivasyon fonksiyonlarının, bir kuantum sistemindeki doğal doğrusal işlemler nedeniyle kuantum teorisinin matematiksel yapısını nadiren takip etmesi nedeniyle engellenmiştir. Uzun çabalardan sonra Schuld, Sinayskiy ve Petruccione aktivasyon fonksiyonunu uygulamak için kuantum faz tahmin algoritmasını (Schuld ve diğ., 2014) kullanmıştır. Bunun dışında, bulanık mantık tabanlı bir sinir ağını uygulamak

için kuantumdan ilham alan birçok model ortaya çıkmıştır. Elizabeth Behrman ve Jim Steck ayarlanabilir karşılıklı etkileşimlere sahip bir dizi kübitten oluşan yeni bir kuantum hesaplama düzeneği önermiştir. Modellerinde, etkileşim güçleri, klasik geri yayılım algoritmasını takiben istenen girdi-çıktı ilişkilerinden oluşan bir eğitim seti kullanılarak güncellenmekte ve böylece kuantum ağının bir algoritma öğrenmesi sağlanmaktadır. Kuantum ilişkisel bellek 1999 yılında Dan Ventura ve Tony Martinez tarafından tanıtılmıştır. Yazarlar, devre tabanlı bir kuantum bilgisayar için ilişkisel bir belleği taklit etmek üzere bir algoritma önermişlerdir. Bu algoritmada, bellek durumları kuantum durumlarının bir süperpozisyonu olarak öngörülmüştür. Daha sonra verilen bir girdiye en yakın bellek durumunu almak için bir kuantum arama algoritması kullanılır. Bu emülasyon, bellek durumlarının üstel bir depolama kapasitesini vaat etmektedir (Behrman ve diğ., 2008).

4.5. Kubit ve Fizik

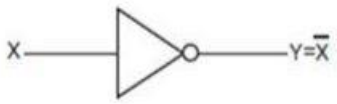
Klasik bir bilgisayarda bilgiyi en küçük yapı taşı olan ve bit diye ifade ettiğimiz (Binary Digit) yapılarda depolarız. Fiziksel cihazlarda bitler 0 ve 1 olmak üzere iki durumda tutulur. Bizim klasik bilgisayarda bitlerle ifade ettiğimiz yapılar kuantum bilgisayarlarda qubitlerle ifade edilmektedir. Qubitlerde (0) ve (1) olabileceği gibi bunların kombinasyonları olan süper pozisyonda da olabilir. Qubitlerin bu durumları |00>,|01>,|10>ve |11> ve şeklinde gösterilir. Qunatum bilgisayarda qubitlerin durumlarını anlamamız ve ölçümlememiz klasik bilgisayar bitleri gibi olamamasının yanı sıra klasik fizik kuralları ile de anlamamız mümkün değildir. Bu yüzden quantum bilgisayarda atom altı parçacıkları olan elektron spinlerinin yönünü ve değerini tespit edebilmemiz için quantum fiziğinden yararlanırız. Bu atom altı parçacıklarının yönü aşağı ve yukarı olmak üzere iki durumdan ibarettir. Bu durumlar bizim yaşadığımız dünyada anladığımız yukarı ve aşağı kavramları değildir. Eğer bir qubit içinde bulunduğu manyetik alanla aynı yönde ise yukarı, ters yönde ise aşağı yönlüdür. Bir elektron spinin aynı anda üç bileşeninden (x- ekseni, y-ekseni ve z-ekseni) manyetik alanı hangisine uygulamışsak onun durumunu ölçebiliriz. Çünkü manyetik alanı sadece bir yönde uygulayabiliriz.

4.6. Kuantum Kapıları

Bir kuantum kapısı veya kuantum mantık kapısı, az sayıda kübit üzerinde çalışan ilkel bir kuantum devresidir. Aynı zamanda bu kapılar birer matristir. Bu matrisler üzerinde değişik işlemler yaparak bit'in ya da kubit'in yeni durumlar almasını sağlarlar. Kuantum mantık kapıları, birçok klasik mantık kapısının aksine tersine çevrilebilir. Kuantum mantık kapıları üniter matrislerle temsil edilir (Nielsen ve Chuang, 2000).

4.6.1. X (not) kapısı

Geçidin girişi '0' olduğunda, çıkış '1' olur. Geçidin girişi '1' olduğunda, çıkış '0' olur. NOT kapısı yalnızca tek bir bit içerir, ancak diğer kapılar daha fazla bit içerir. Örneğin XOR (veya EXOR) kapısı 2 bitlik bir giriş alır ve 2 giriş bitinden tam olarak birinin 1 ve diğerinin '0' olması durumunda '1' olan bir değer çıkarır.

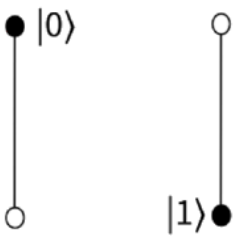


X	Y
0	1
1	0

Şekil 4.1. NOT kapısı ve doğruluk tablosu

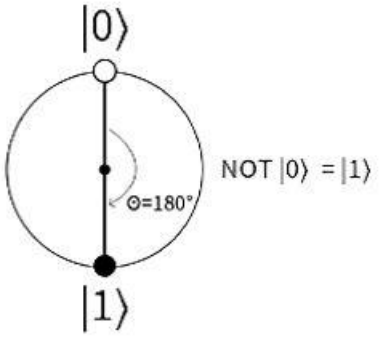
4.6.2. X (pauli x) kapısı

Klasik bilgisayarlar için NOT kapısının kuantum eşdeğeri Pauli-X kapısı olarak adlandırılır. Tek bir kübit üzerinde etki eder. Eğer kübit |0> ya da |1> durumlarındaysa, Pauli-X kapısı |0> değerini |1>'e çevirir ya da tam tersini yapar. İki saf durumu |0> ve |1> dikey eksen boyunca karşılıklı iki nokta olarak temsil edersek



Şekil 4.2. Pauli(X) kapısı gösteri

Pauli-X geçidinin uygulanması, kübitin durumunun 180 derece döndürülmesine eşdeğerdir. Bu doğası nedeniyle bazen bit-flip olarak adlandırılır.



Şekil 4.3. Pauli(X) kapısına uygulanma gösterimi

Klasik hesaplamada olduğu gibi, her kuantum kapısı bir sembolle temsil edilir; aşağıda genellikle Pauli kapısını nasıl temsil ettiğimiz gösterilmektedir.

Şekil 4.4. Pauli kapısı

Matematiksel olarak konuşmak gerekirse, bir kuantum kapısı uygulamak, matris biçimindeki doğrusal bir U operatörünü manipüle etmeye eşittir.

$$|0\rangle=\begin{bmatrix}1\\0\end{bmatrix},\ |1\rangle=\begin{bmatrix}0\\1\end{bmatrix} \tag{4.1}$$

Pauli-X geçidinin matris gösterimi, yani Pauli-X matrisi aşağıdaki 2x2 matrisidir:

$$X=\begin{bmatrix}0 & 1\\1 & 0\end{bmatrix} \tag{4.2}$$

|0> ve |1> saf durumlarına uygulandığında, |0>'ın |1>'e dönüştüğünü ve bunun tersinin de geçerli olduğu kolayca görülebilir:

$$\begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}\begin{bmatrix}1\\0\end{bmatrix}=\begin{bmatrix}0\\1\end{bmatrix} \tag{4.3}$$

$$\begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}\begin{bmatrix}0\\1\end{bmatrix}=\begin{bmatrix}1\\0\end{bmatrix} \tag{4.4}$$

4.6.3. Y (pauli y) kapısı

Pauli-Y kapısı tek bir kübit üzerinde etki eder. Bloch küresinin Y ekseni etrafında π radyan kadar bir dönüşe eşittir. |0>'ı |1⟩'e ve |1⟩'i, -i|0⟩'a eşler. Bloch küresi üzerinde i|1> yukarıdaki açıklama uyarınca |1> ile aynı nokta olacaktır (i|1> = $e^{i\gamma}$|1> γ=π/2 ile) ve -i|0> |0> ile aynı gösterime sahip olacaktır (-i|0> = $e^{-i\pi/2}$|0>). Y kapısının matris gösterimi aşağıdaki gibidir:

$$Y=\begin{bmatrix}0 & -i\\i & 0\end{bmatrix} \tag{4.5}$$

|ψ> = Karışık bir duruma α|0> + β|1> uygulandığında Pauli-Y geçidi Y|ψ> = Y(α|0> + β|1>) = -iβ|0> + iα|1> değerini verir.

$$Y=\begin{bmatrix}0 & -i\\i & 0\end{bmatrix}\begin{bmatrix}\alpha\\\beta\end{bmatrix}=\begin{bmatrix}-i\beta\\i\alpha\end{bmatrix} \tag{4.6}$$

4.6.4. Z (pauli z) kapısı

Pauli-Z kapısı tek bir kübit üzerinde de etkilidir. Bloch küresinin Z ekseni etrafında π radyan kadar bir dönüşe eşittir. |0> temel durumunu değiştirmeden bırakır ve |1>'i -|1>'e eşler. Z kapısının matris gösterimi aşağıdaki gibidir.

$$Z=\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix} \quad (4.7)$$

Karışık bir duruma uygulandığında $|\psi> = \alpha|0> + \beta|1>$ Pauli-Z kapısı $Z|\psi> = Z(\alpha|0> + \beta|1>) = \alpha|0> - \beta|1>$ sonucunu verir.

$$Z=\begin{bmatrix}1 & 0\\ 0 & -i\end{bmatrix}\begin{bmatrix}\alpha\\ \beta\end{bmatrix}=\begin{bmatrix}\alpha\\ -\beta\end{bmatrix} \quad (4.8)$$

4.6.5. H (hadamard) kapısı

Eğer kübitimiz $\theta = \pi/2$ ve $\varphi=0$ (ya da kartezyen koordinatlarda x=1, y=0, z=0) açılı bir konumda bulunuyorsa neler olduğunu görelim. Yukarıdaki denklemdeki değerleri değiştirerek, kübit durumu şu şekilde genişler:

$$|state> = \cos\left(\frac{\pi}{4}\right)\left|0> + e^{i0}\sin\left(\frac{\pi}{4}\right)\right|1> \quad (4.9)$$

$$|state> = \frac{1}{\sqrt{2}}|0> + \frac{1}{\sqrt{2}}|1>$$

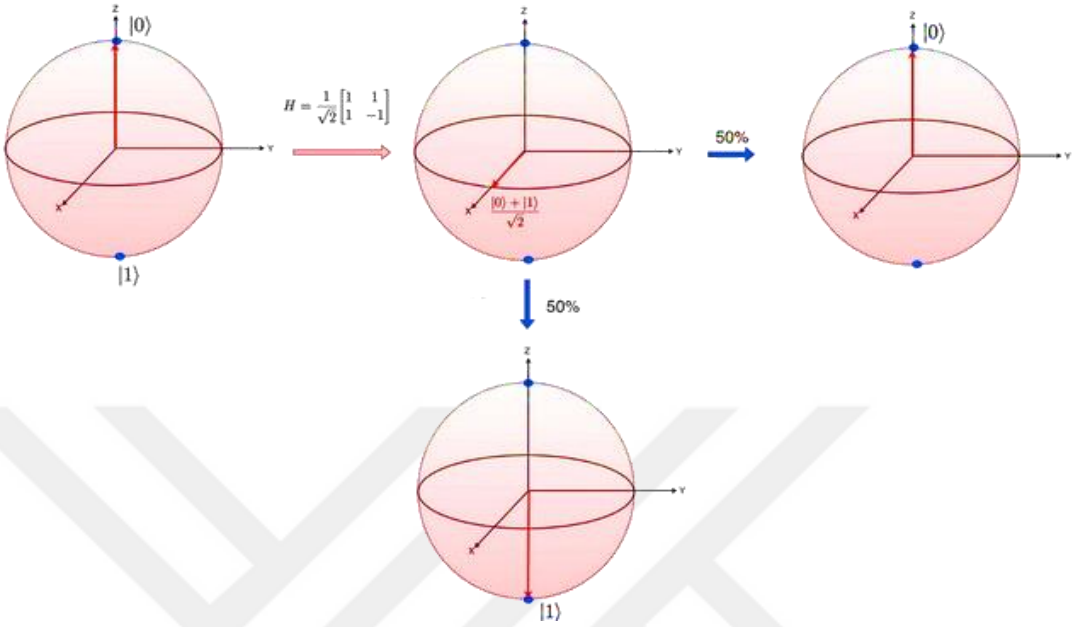
Hesaplama temeli durumlarının eşit ağırlığına sahip bir süperpozisyon durumu $1/\sqrt{2}(|0> + |1>)$ elde ederiz. Hadamard kapısını $|0>$ durumundaki bir kubite uygulamak, kubiti 0 ölçme olasılığının 1 ölçme olasılığına eşit olduğu (ve $1/\sqrt{2}^2=1/2$'ye eşit olduğu) bir $|0>$ ve $|1>$ süperpozisyon durumuna getirir. Benzer şekilde, Hadamard kapısını $|1>$ aşağı durumuna uygulamak kubiti $1/\sqrt{2}(|0> - |1>)$ süperpozisyon durumuna getirir.

$$H=\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} \quad (4.10)$$

Bu geçit $|0>$ ve $|1>$ baz vektörlerine uygulandığında,

$$\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\begin{bmatrix}1\\ 0\end{bmatrix}=\frac{1}{\sqrt{2}}\begin{bmatrix}1\\ 1\end{bmatrix} \quad (4.11)$$

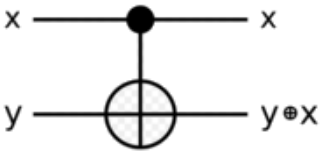
$$\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\begin{bmatrix}0\\ 1\end{bmatrix}=\frac{1}{\sqrt{2}}\begin{bmatrix}1\\ -1\end{bmatrix} \quad (4.11)$$

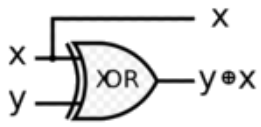


Şekil 4.5. Hadamard kapısı manipülasyonunun bloch küresinde gösterimi

4.6.6. Cnot kapısı

Kuantum CNOT kapısının iki girişi ve dolayısıyla iki çıkışı vardır. Hedef giriş yalnızca kontrol girişi 1'e ayarlanmışsa olumsuzlanır. Kontrol girişi 0 ise geçidin hiçbir etkisi olmaz. Kontrol kübiti geçit tarafından değiştirilmez. Aşağıda hem klasik hem de kuantum diyagramlarının bir anlık görüntüsü yer almaktadır. Birbirlerini yansıttıklarını kolayca doğrulayabiliriz: kuantum hedef çıkış sütunu klasik XOR geçidinin y+x sütunuyla eşleşir.



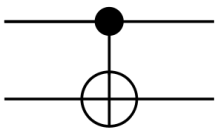


giriş		çıkış	
x	y	x	y+x
|0⟩	|0⟩	|0⟩	|0⟩
|0⟩	|1⟩	|0⟩	|1⟩
|1⟩	|0⟩	|1⟩	|1⟩
|1⟩	|1⟩	|1⟩	|0⟩

giriş		çıkış	
x	y	x	y+x
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

Şekil 4.6. CNOT Kapısı

Bildiğimiz gibi, her geçit/operatör bir matris olarak ifade edilebilir. C-NOT kapısı girdi olarak iki kübit ve çıktı olarak iki kübit aldığı için 4x4'lük bir matris olacaktır. Bir doğruluk tablosunu matrise dönüştürmek için kullanışlı bir teknik vardır. 0. satır 0. sütundan başlayarak, sütunları ve satırları örneğin 00'dan 11'e kadar ikili olarak ardışık bir şekilde etiketlersiniz. Daha sonra, girdi çıktı ile eşleşiyorsa bir hücreye 1 yerleştirirsiniz; aksi takdirde 0'dır. Örneğin, bizim durumumuzda, |11>, |10> ile eşleştiği için matrisin 4. satır, 3. sütun değerini 1 olarak ayarlanır. Böylelikle kapı için bir matris elde edilir.



$$CNOT=\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}$$

Şekil 4.7. CNOT kapısı'nın şekil ve matris gösterimi

CNOT matrisini temel durum vektörüyle çarparak CNOT geçidini örnek olarak |00> durumuna uygulamaya çalışalım.

$$\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}\begin{bmatrix}1\\0\\0\\0\end{bmatrix}=\begin{bmatrix}1\\0\\0\\0\end{bmatrix}=|00> \tag{4.12}$$

4.6.7. Z (z gate) kapısı

Z-kapısı, sadece bir kübit üzerinde etkili olan üniter bir kapıdır. Özellikle 1'i -1'e eşler ve 0'ı değiştirmeden bırakır. Bu manipulasyonu, kübiti Z ekseni etrafında π radyan (180 derece) döndürerek yapar. Bunu yaparak kübitin fazını değiştirmiş olur. Z-kapıları işlemi aşağıdaki matris ile tanımlanır:

$$Z=\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix} \tag{4.13}$$

Pauli Z kapısının kübit üzerinde nasıl çalıştığını, kübit durumunun sütun vektörünü Pauli Z matrisiyle çarparak görebiliriz. Örneğin, kübit |0⟩ olarak başlatılırsa:

$$\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}\begin{bmatrix}1\\ 0\end{bmatrix}=\begin{bmatrix}1\\ 0\end{bmatrix} \tag{4.14}$$

Bu da |0>'ı değişmeden bırakır. Şimdi kübiti |1> olarak başlatalım ve Z geçidinin kübit durumunu nasıl dönüştürdüğünü görelim:

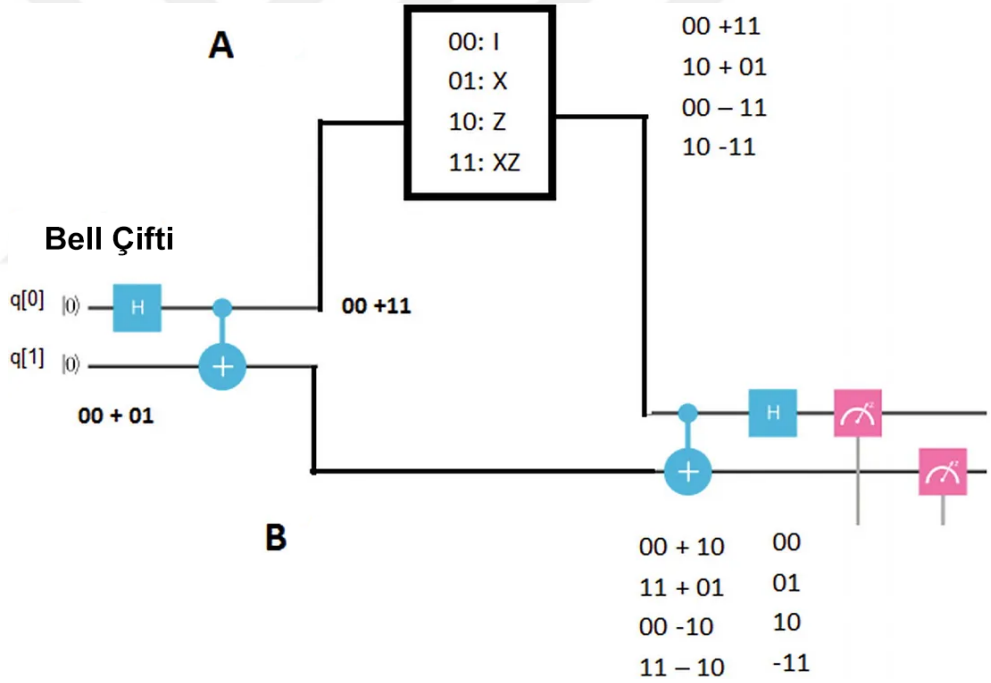
$$\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}\begin{bmatrix}0\\ 1\end{bmatrix}=\begin{bmatrix}0\\ -1\end{bmatrix} \tag{4.15}$$

Kübitlerin durumunun |1⟩'den -|1⟩'e dönüştüğü görülmektedir. Bu durumda sadece faz değişmiştir. Faz değişiminden sonra ölçüm yapıldığında kubit'in |1⟩'e çöktüğü gözlenmektedir.

4.7. Süper Yoğun Kodlama

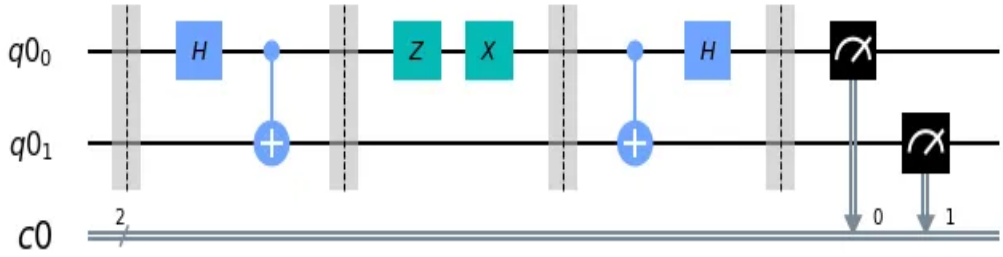
Bu kodlama, dolanık durumların ilginç bir uygulamasıdır. Bu durumu kısaca açıklamak gerekirse tek bir kubit paylaşımı ile iki bitlik klasik bilgi paylaşımını mümkün kılan bir protokoldür. Şekil 4.8' de bu durum gösterilmiştir. Burada A ve B kişileri birbirine çok uzak noktada bulunan iki arkadaş olarak temsil edilmektedir. A, B'ye iki bitlik bir bilgi göndermek istemektedir. Ancak, A'nın yalnız bir kubit'lik bilgi göndermesi mümkündür. Burada soru şu: "A, tek bir kubit kullanarak iki bitlik bir bilgiyi B ile paylaşabilir mi?". Evet, bunu yapabilir. Bunun bir tek şartı var. Daha

önceden bir Dolanık durumu kendi aralarında paylaşmış olmaları gerekmektedir. Sistemin çalışabilmesi için girişinde iki bitlik bir dolanık durum (Bell Çifti) hazırlanmalı ve önce Hadamard, sonra CNOT kapısından geçen bu bitlerden bir tanesi A diğeri B tarafından alınır. Bu gönderim kuantum kanal vasıtasıyla mümkün olur. Hem A hem de B kendilerine ait kubit'in |0>'mı yoksa |1>'mi olduğunu bilmemektedir. Eğer A kendi aldığı biti olduğu gibi B'ye göndermek istiyorsa üzerinde hiçbir işlem yapmaz. Eğer herhangi bir bilgi gönderecekse (00, 01,10,11) üzerinde işlem yapar ve B'ye gönderir. B gelen kubiti alınca yeni duruma geçmiş olur. B bu sahip olduğu durumu önce CNOT sonra Hadamard kapısından geçirir. Çıkan sonucu görmek için ölçüm yapar. Böylece B de tek bir kubit almasına rağmen, A'nın iki bitlik hangi bilgi dizisini gönderdiğini yüzde yüz olarak öğrenir. Kuantum bilgisayar ortamında gerçekleşen bu olaya Süper Yoğun Kodlama denir.



Şekil 4.8. Süper yoğun kodlama protokol diyagramı

Yukarda verilmiş olan diyagram gösterimi bir kuantum bilgisayarda simüle ettiğimiz zaman aşağıdaki gibi bir sonuç ortaya çıkar.



Şekil 4.9. Süper yoğun kodlama devresi

4.8. Dolanıklık

Birden fazla alt sistemden oluşmuş sistem topluluğuna bileşik sistem denir. Bileşik bir sistemin durum uzayı, sistemi oluşturan fiziksel sistemlerin durum uzaylarının tensor çarpımı şeklinde ifade edilir. Eğer elimizde |A> ve |B> gibi iki sistem varsa ve bu sistemler arasında da bir korelasyon bulunuyorsa bunlar dolanık sistemlerdir. Yani birisine ait bir özelliği ölçtüğümüz zaman diğer sisteme ait başka bir bilgiye ulaşıyorsak bu iki sistem dolanık sistemlerdir. Dolanıklık tamamen kuantum mekaniksel bir olay olup, biz sadece klasik olarak anoloji yapabiliriz. Örneğin, içerisine birer mıknatıs gömdüğümüz iki zar düşümelim. Bu iki zar üzerindeki manyetik alan öyle oluşur ki, birinin üzeride gelen sayı diğerinden etkileniyorsa bu iki zar dolanıktır. Yani birinci zarı attığınızda üst yüzde gelen sayı 6 ise, ikinci zara bakmadan onun üst yüzeyindeki sayının kesinlikle 4 olduğunu söyleyebiliyorsak bu iki zar dolanıktır. Bu durumu iki kübit üzerinde düşünecek olursak; eğer kübitlerden birisini ölçtüğümüzde sonucu |1> bulmuşsak diğer kübitte |1>'dir. |0> bulmuşsak diğer kübitte |0>'dır.

4.9. Kuantum Işınlanma (Teleportasyon)

Kuantum Teleportasyon, bir durum vektörünün bir yerden başka bir yere gönderme tekniğidir. Bu iki nokta arasındaki mesafa ışık yılı kadar uzakta olabilir. Burada önemli olan nokta, Kuantum durumunu iki nokta arasında birinden diğerine gönderirken aralarında bir kuantum kanal olmasına gerek yoktur. Kuantum ışınlama olayının genel olarak şematik yapısı Şekil 4.10 'da verilmiştir. Burada gerçekleşen olay şöyle; Süper Yoğun Kodlamada olduğu gibi A ve B'nin uzun zaman önce birlikteyken, aralarında dolanık durumlardan (Bell Durumlarından) bir tanesini

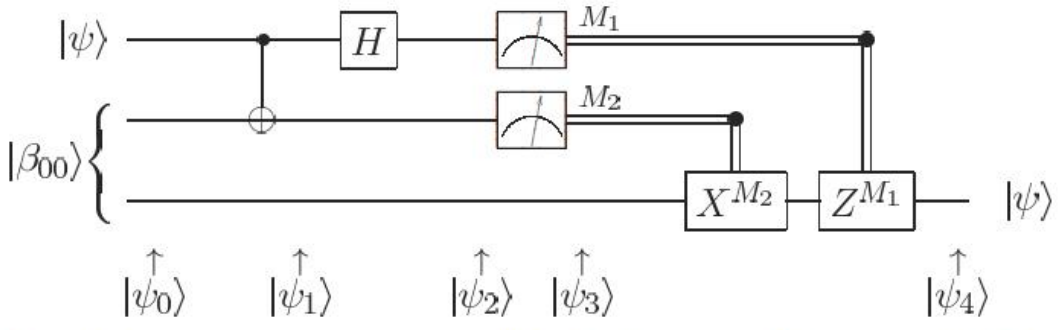
paylaşıp bir kubit'ini A, diğer kubit'ini B almış ve sonra birbirlerinden uzaklaşmış olsun. Bir süre sonra A, B'ye |Ψ> kuantum mekaniksel durum vektörü göndermek istesin. A bunu aralarında bir kanal ya da kuantum kablo yokken başarabilir mi? Bununla birlikte, A, B ile herhangi bir kübit paylaşımında bulunamayıp, klasik bit (klasik kanallarla) paylaşabiliyor. Bunu biraz düşündüğümüzde imkansız olduğu kanaatine varırız. Doğru olan da budur. Çünkü A'nın elindeki kendi durum vektörünü B'ye nasıl gönderecek? A'nın elindeki bir kubitlik durum vektörü, |Ψ>=α|0>+β|1> şeklindedir. α ve β bütün durumları alabilen kompleks sayılardır ve bu vektörü B'ye göndermek istemektedir. Aşağıdaki, Şekil 4.10'da bir CNOT kapısı, bir Hadamard kapısı ve iki adet ölçüm operatörü bulunmaktadır. Tek çizgi ile gösterilen kısım kuantum kanalını gösterilirken, ölçüm operatöründen sonraki çift çizgiler ise klasik haberleşme kanallarını göstermektedir. Ölçüm yapıldıktan sonra durum vektörü yok olur geriye (0) ya da (1) gibi reel bir sayı kalır.

A ve B'nin, Bell durumlarından birisini kendi aralarında paylaştığını biliyoruz.

$$|\Psi>=\frac{1}{\sqrt{2}}(|00>+|11>) \tag{4.16}$$

Bu formüldeki birinci kubit A'nın ikinci kubit B'nin olsun. Her ikisi de kendi kübitine sahip olup, birbirlerinden çok uzakta bulunmaktadırdlar ve dolanık durumdadırlar.

Aşağıdaki şekilde |Ψ> durumu, A'nın B'ye göndermek istediği durum olup, $|\beta_{00}>$ ise dolanık Bell durumlarından bir tanesidir. Öncelikli olarak A, göndermek istediği durumu kendi payına düşen kübit ile etkileşime sokar. Böylelikle A iki kübite B ise tek kübite sahip olur. A kendi iki kübitli durumunu önce CNOT kapısından sonra Hadamard kapısından geçirdikten sonra ölçüm (M_1) yapar.



Şekil 4.10. Kuantum ışınlanma

Şekilde görüleceği üzere ilk durumdaki durumu vektörü $|\Psi_0\rangle$, CNOT kapısından geçtikten sonra $|\Psi_1\rangle$, Hadamard kapısından geçtikten sonra $|\Psi_2\rangle$ ve ölçümler yapıldıktan sonraki durum ise $|\Psi_3\rangle$ tür. Öncelikle $|\Psi_1\rangle$ durum vektörüne bakacak olursak,

$$|\Psi_1\rangle=|\Psi\rangle|\beta_{00}\rangle \quad (4.17)$$

$$|\Psi_1\rangle= (\alpha|0\rangle+\beta|1\rangle)\left(\frac{|00\rangle+|11\rangle}{\sqrt{2}}\right)$$

$$|\Psi_1\rangle= \frac{1}{\sqrt{2}}[\alpha|000\rangle+ \alpha|011\rangle+ \beta|100\rangle+ \beta|111\rangle]$$

Bu $|\Psi_1\rangle$ durum vektörünün ilk iki kubiti A'ya ait sonuncu kubit ise B'ye aittir. Bundan sonra $|\Psi_1\rangle$ durum vektörü CNOT kapısından geçiyor ve $|\Psi_2\rangle$ durum vektörüne sahip olur. CNOT kapısından geçişi sırasında ise CNOT kapısı ilk iki kubiti manipüle edecek üçüncü kübite etki etmeyecektir. CNOT kapısında ilk kubit kontrol (control) kübit, ikinci kübit ise hedef (target) kübittir. Eğer kontrol kübit ($|0\rangle$) ise, hedef kübit herhangi bir değişikliğe uğramaz. Kontol kübit ($|1\rangle$) ise, hedef kübit ($|0\rangle$) ise ($|1\rangle$), ($|1\rangle$) ise ($|0\rangle$) olur. $|\Psi_2\rangle$ durum kapısına bakacak olursak,

$$|\Psi_2\rangle=\frac{1}{\sqrt{2}}[\alpha|00\rangle|0\rangle+ \alpha|01\rangle|1\rangle+ \beta|11\rangle|0\rangle+ \beta|10\rangle|1\rangle] \quad (4.18)$$

İlk iki kübit CNOT kapısından geçtikten sonra yukardaki $|\Psi_2\rangle$ durumu elde edilir. Bundan sonra birinci kübit Hadamard kapısından geçecek ve $|\Psi_3\rangle$ durumu elde edilecek. Hadamard kapısı ilk kübite etki ettiğinde eğer bu kubit ($|0\rangle$) ise, $H|0\rangle=\frac{1}{\sqrt{2}}|0\rangle+|1\rangle$ eğer ($|1\rangle$) ise $H|0\rangle=\frac{1}{\sqrt{2}}|0\rangle-|1\rangle$ ile manipüle edip süperpozisyona sokar. Bu işlemden sonra oluşacak $|\Psi_3\rangle$ durum vektörü aşağıdaki gibi oluşur.

$$|\Psi_3\rangle=\frac{1}{\sqrt{2}}\left[\alpha\left(\frac{|0\rangle+|1\rangle}{\sqrt{2}}\right)|00\rangle+ \alpha\left(\frac{|0\rangle+|1\rangle}{\sqrt{2}}\right)|11\rangle + \beta\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right)|10\rangle+ \beta\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right)|01\rangle\right] \quad (4.19)$$

$$|\Psi_3\rangle=\frac{1}{2}[\alpha|000\rangle+ \alpha|100\rangle+ \alpha|011\rangle+ \alpha|111\rangle+ \beta|010\rangle- \beta|110\rangle+ \beta|001\rangle- \beta|101\rangle]$$

$$|\Psi_3\rangle=\frac{1}{2}[\ |00\rangle(\ \alpha|0\rangle+ \beta|1\rangle)+ |01\rangle(\ \alpha|1\rangle+ \beta|0\rangle) + |10\rangle(\ \alpha|0\rangle- \beta|1\rangle)+ |11\rangle(\ \alpha|1\rangle- \beta|0\rangle$$

$|\Psi_3\rangle$ durumdan sonra A kendine ait iki kübiti (M_1) ölçümler. A'nın iki kübiti olduğundan muhtemel dört durumundan ($|00\rangle$,$|01\rangle$,$|10\rangle$,$|11\rangle$) birini gözler. Bu dört

durumdan her birisinin gözlemlenme ihtimali $\frac{1}{4}$' tür. Bunu nereden biliyoruz. A'nın önünden bulunan katsayılardan biliyoruz. A'nın kübitlerinin önündeki katsayı $\frac{1}{2}$ 'dir ve herhangi bir durum gelme ihtimali bu katsayının karesine eşittir. A, ölçümü yaptıktan sonra bu bilgiyi B'ye gönderir. Bunu yapabilmesi için B ile iletişime geçmesi ve kübit değerlerini yani iki bitlik bilgiyi (ölçümden sonra artık kübit değil bittir) gönderir. B kendisine A tarafından gönerilen iki bitlik bilgiye bağlı olarak, başlangıçta kendinde olan kübiti (A ile birer kübitini paylaştıkları dolanık durumdan) şekildeki (X), (Z) ya da hem (X) hem de (Z) kapılarından geçirerek A'nın kendisine göndermek istediği durum vektörüne sahip olur. İşte bu duruma (olaya) kuantum ışınlama (Teleportasyon) denir.

5. TEMEL KUANTUM ALGORİTMALARI

Kuantum makine öğrenmesi algoritmaları aşağıda ele alınmaktadır.

5.1. Grovers Algoritması

Grover'ın Arama Algoritması (GAA) çok popüler bir kuantum arama algoritmasıdır. Bu algoritma belirli bir koşulu sağlayan bir grup elemanı bulur. Bu görevi yerine getirmek için, genellikle oracle olarak bilinen, elemanları tanımlama yeteneğine sahip bir kara kutu, bulmak için gerekli kriterleri karşılamalıdır. Grubun N elemanı olduğunu varsayalım, kâhin yukarıda belirtilen kriterleri karşılayan tüm elemanları elde etmek için klasik hesaplamalar için O(M) kez çağrılır. Kuantum mekaniğini kullanan bu algoritma, kâhine O√M p çağrı yaparak aynı sonuca ulaşabilmektedir.

Algoritma, kuantum hesaplamanın paralel işleme adı verilen özel bir özelliğinden yararlanarak kâhine aynı anda birden fazla çağrı yapma yeteneğine sahiptir. Bir arama listesinin N sayıda elemana sahip olduğunu varsayalım. GAA bunları temsil etmek için N boyutlu bir Hilbert uzayı kullanır, bu da $n = \log_2 M$ kübit ile elde edilebilir. İndeksi y olan her e ϵ N, kübitlerin durum uzayında j‘i olarak adlandırılan ortonormal bir vektör ile gösterilir. Amaç verilen arama kriterlerini karşılayan belirli bir elemanın z indeksini belirlemektir. Başlangıçta, aşağıdaki özelliklere sahip üniter bir işlemin kullanıldığı öngörü kullanılmaktadır

$$y=z,\ U_z|y\rangle=-|y\rangle \tag{5.1}$$

$$y\neq z,\ U_z|y\rangle=|y\rangle \tag{5.2}$$

Eşitlik (5.1) ve (5.2) aşağıdaki gibi de ifade edilebilir:

$$U_z = I-2|z\rangle\langle z| \tag{5.3}$$

Burada I, birim operatörünü göstermektedir. GAA eşzamanlı olarak son oracle operatörünü ve Grover difüzyon operatörlerini aşağıdaki şekilde tanımlandığı gibi kullanır.

$$U_p = 2|p><p|-I \quad (4.4)$$

Başlangıçta, kübitlerin durumu |p> durumuna başlatılır. Daha sonra, U_z ve U_p iteratif olarak r(N) sayıda art arda uygulanır. Daha sonra sistem değerlendirilir ve bu da söz konusu (z) indeksini sonuçlandırabilecek özdeğeri (λ_z) sağlar.

5.2. Bernstein - Vazirani Algoritması

Bernstein-Vazirani Algoritması, Ethan Bernstein ve Umesh Vazirani tarafından 1992 yılında geliştirilen bir kuantum algoritmasıdır. Esasen, bir fonksiyonun içerdiği ikili bir {s} dizisini, yani sıfır ve birlerden oluşan bir karakter dizisini (örneğin s = 0010110101001) bilmeyi sağlar. Daha doğrusu, böyle bir fonksiyonun ***f(x)=sx mod (2)*** biçimini aldığı bilinmektedir; burada x başka bir dizedir ve çarpma ikili çarpım olarak anlaşılır. Bu algoritma Deutsch-Jozsa algoritmasına benzer bir şekilde çalışır, ancak işlev sınıfları arasında ayrım yapmaya çalışmak yerine verilen işlevi karakterize eden dizeyi arar. Örnek olarak, ikili kodla yazılmış gizli bir sayıyı bulmak için bir oyun oynadığınızı varsayalım. Algoritmanın klasik versiyonu ile çözümü elde etmenin tek yolu gizli sayıyı bit bit kontrol etmek olacaktır ki bu da en az N çalıştırma gerektirir, burada N s'nin bit sayısıdır (bu hesaplama karmaşıklığı teorisinde O(N) olarak gösterilir). Bernstein-Vazirani algoritması durumunda, bu sayı s dizesinde kodlanmışsa, tam sayıyı bulmak için algoritmanın tek bir çalıştırılması yeterli olacaktır. Bu algoritmanın önemi, aranan dizenin tek bir çalıştırmadan sonra bulunabildiği klasik muadiline göre üstünlüğünde yatmaktadır (Bernstein ve Vazirani, 1993).

5.3. Deutsch-Jozsa Algoritması

Kuantum hesaplamada Deutsch-Jozsa algoritması, 1992 yılında David Deutsch ve Richard Jozsa tarafından önerilen bir kuantum algoritmasıdır. Bir kuantum bilgisayarda çalışmak üzere tasarlanan ilk algoritmalardan biridir ve kuantum süperpozisyon durumlarının doğal paralelliğinden yararlanarak klasik algoritmalardan daha verimli olma potansiyeline sahiptir. Deutsch-Jozsa probleminde, n adet x_1, x_2,..., x_n giriş biti alan ve f(x_1, x_2,..., x_n)= 0 veya 1 ikili değerini döndüren bir f(x_1, x_2,..., x_n) fonksiyonu (bir kehanet veya kara kutu olarak düşünülebilir) vardır. Amaç, fonksiyonun sabit (tüm girişlerde 0 veya tüm girişlerde

1) veya dengeli (girişlerin yarısı için 1 ve diğer yarısı için 0 döndürür) olup olmadığını belirlemektir. Problem, kara kutuya girdiler uygulayarak ve çıktısını gözlemleyerek fonksiyonun neye benzediğini (sabit veya dengeli) belirlemektir. Örnek olarak, ***f(x)=x%2*** fonksiyonunu, yani girdiyi ikiye böldüğünüzde kalanı döndüren fonksiyonu düşünün. Bu fonksiyon, argüman tek ise 1, çift ise 0 döndürür, bu nedenle dengeli bir fonksiyondur. Algoritmanın işlevi, aynı sonuca mümkün olduğunca az yinelemeyle ulaşmak olacaktır; klasik durumda bu, iki farklı sonuca ulaşılana kadar işlevin tekrar tekrar değerlendirilmesini gerektirecektir ve bu nedenle yineleme sayısı girdi değişkenlerinin seçilme sırasına bağlı olacaktır (Johansson ve Larsson, 2017).

5.4. Shor's Algoritması

Kuantum hesaplamada Shor'un algoritması, bir N sayısını $O((\log N)^3)$ zamanda ve O(logN) uzayda çarpanlarına ayırmak için kullanılan ve Peter Shor'un adını taşıyan bir kuantum algoritmasıdır. Shor'un algoritması, bir sayının çarpanlarını verimli bir şekilde bulmak için kullanılan bir prosedür dür. Bu algoritmanın uygulanması klasik olarak veya kuantum devreleri kullanılarak gerçekleştirilebilir (henüz pratikte uygulanmamıştır). İkinci uygulama (elbette) belirli bir sayının asal çarpanlarını bulurken çok gerekli bir parametre olan sıralamayı bulmak istediğinizde en uygun olanıdır. Shor'un algoritmasının pratik bir kuantum bilgisayarında uygulanması halinde RSA gibi pek çok açık anahtarlı kriptografi geçersiz hale gelecektir. RSA ile şifrelenmiş bir mesajın şifresi, iki asal sayının çarpımı olan N açık anahtarının çarpanlarına ayrılmasıyla çözülebilir. Bilinen klasik algoritmalar bunu herhangi bir k için $O((\log N)^k)$ zamanda yapamazlar, bu nedenle N arttıkça hızla pratik olmaktan çıkarlar. Buna karşın, Shor'un algoritması RSA'yı polinom zamanda kırabilir. Tüm kuantum hesaplama algoritmaları gibi Shor'un algoritması da olasılıksaldır. Yüksek olasılıkla doğru cevabı verir ve algoritma tekrarlanarak başarısızlık olasılığı azaltılabilir. Shor'un algoritması 2001 yılında IBM'deki bir grup tarafından pratikte uygulanmış ve 7 kübitlik bir kuantum bilgisayar kullanılarak 15, 3 ve 5 faktörlerine ayrıştırılmıştır (Nielsen ve Chuang, 2000).

Shor'un algoritmasının çözmeye çalıştığı problem, bir N tamsayısı verildiğinde, 1 ile N arasında N'yi bölen başka bir p tamsayısı bulmaya çalışmaktır. Shor'un algoritması iki bölümden oluşmaktadır:

- Faktörlere ayrıştırma probleminin, klasik bir bilgisayarda yapılabilen sırayı bulma problemine indirgenmesi.
- Periyot bulma problemini çözmek için bir kuantum algoritması.

Shor'un periyodunu bulma algoritması, bir kuantum bilgisayarının aynı anda birçok durumda bulunabilme yeteneğine kökten bağlıdır. Fizikçiler bu davranışı kuantum süperpozisyonu olarak adlandırır. Bir f fonksiyonunun periyodunu hesaplamak için, fonksiyonu tüm noktalarda aynı anda değerlendiririz. Ancak kuantum fiziği tüm bu bilgilere doğrudan erişmemize izin vermiyor. Bir kuantum ölçümü tüm olası değerlerden yalnızca birini verecek, diğerlerini yok edecektir. Bu nedenle süperpozisyonu, yüksek olasılıkla doğru cevabı veren başka bir duruma dikkatlice dönüştürmemiz gerekir. Bu, kuantum Fourier dönüşümü kullanılarak gerçekleştirilir. Bu nedenle Shor'un üç "uygulama sorununu" çözmesi gerekmiştir. Tümünün "hızlı" uygulanması gerekmiş, bu da logN'de polinom olan sayıda kuantum kapısı ile çalışmak anlamına gelmiştir.

- **Durumların süperpozisyonunu oluşturma:** Bu, giriş kaydındaki tüm kübitlere Hadamard kapıları uygulanarak yapılabilir. Başka bir yaklaşım da kuantum Fourier dönüşümünü kullanmaktır.
- **f fonksiyonunu bir kuantum dönüşümü olarak uygulama:** Bunu başarmak için Shor, modüler üs alma dönüşümü için karelerle üs alma yöntemini kullanmıştır.
- **Bir kuantum Fourier dönüşümü gerçekleştirme:** Shor, NOT kontrollü kapılar ve tek rotasyonlu kübit kapıları kullanarak kuantum Fourier dönüşümü için tam olarak $((\log N)^2)$ kapı kullanan bir devre tasarlamıştır.

Tüm bu dönüşümlerden sonra bir ölçüm r periyoduna yaklaşık bir değer verecektir. Basitlik açısından, yr/N bir tamsayı olacak şekilde bir y olduğunu varsayalım. O halde y'yi ölçme olasılığı 1'dir. Bunu görmek için şu denklem kullanılabilir:

$$e^{\frac{2\pi i b y r}{N}} = 1 \tag{4.5}$$

Bu nedenle y ölçümünün olasılığını veren toplam N/r olacaktır çünkü b yaklaşık N/r değer alır ve bu nedenle olasılık 1/r'dir. Öyle r, y vardır ki yr/N bir tam sayıdır, dolayısıyla olasılıkların toplamı 1'dir. Shor'un algoritmasını açıklamanın bir başka yolu da bunun tam olarak şekil değiştirmiş bir kuantum faz kestirimi algoritması olduğunu belirtmektir (Shor, 1999).

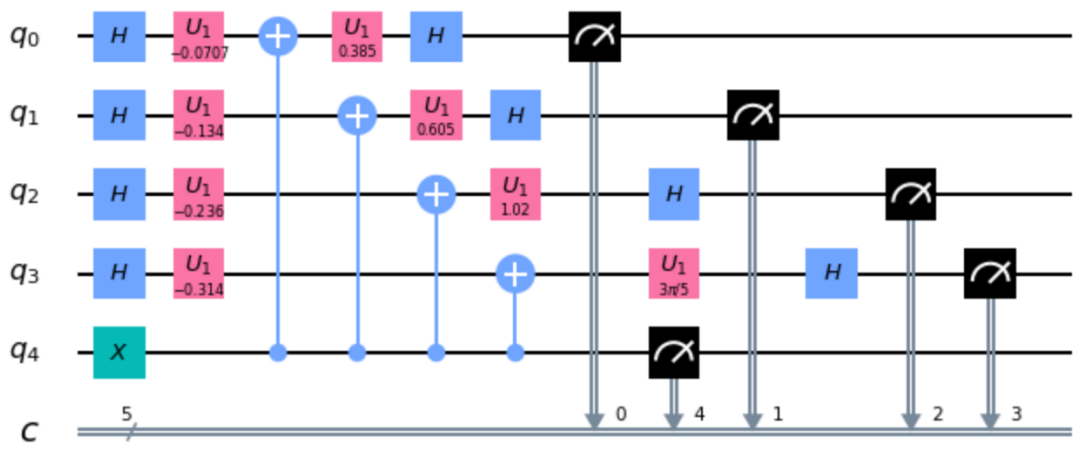
6. QİSKİT NEDİR?

Qiskit, IBM tarafından devre ve algoritma düzeyinde kuantum bilgisayarlarla çalışmak için oluşturulan bir yazılım geliştirme kitidir. Qiskit kullanılarak tasarlanmış bir kuantum devre örneği Şekil 6.1'de verilmiştir. Qiskit kuantum programları oluşturmak, manipüle etmek ve bunları IBM Quantum Experience'daki prototip kuantum cihazlarında veya yerel bir bilgisayardaki simülatörlerde çalıştırmak için araçlar sağlar. Evrensel kuantum hesaplama için devre modelini takip eder ve bu modeli takip eden herhangi bir kuantum donanımı (şu anda süper iletken kübitleri ve tuzaklanmış iyonları desteklemektedir) için kullanılabilir. Qiskit, IBM Research tarafından buluttaki kuantum bilişim hizmeti IBM Quantum Experience için yazılım geliştirilmesini sağlamak amacıyla kurulmuştur. Genellikle akademik kurumlardan olmak üzere dışarıdan meraklılar da katkıda bulunmaktadır (Hemsoth, 2018).

Qiskit'in ana sürümü Python programlama dilini kullanır. Swift ve JavaScript sürümleri başlangıçta araştırılmış, ancak bu sürümlerin geliştirilmesi durdurulmuştur. Bunun yerine, temel özelliklerin minimal bir yeniden uygulaması, alternatif platformlara taşınması kolay olacak şekilde yapılmış MicroQiskit olarak mevcuttur. Kuantum hesaplamanın kullanımına ilişkin örnekler içeren bir dizi Jupyter not defteri sunulmaktadır. Bu örnekler arasında Qiskit kullanılarak yapılan bilimsel çalışmaların kaynak kodlarının yanı sıra insanların kuantum programlamanın temellerini öğrenmelerine yardımcı olacak bir dizi alıştırma da yer almaktadır. Qiskit tabanlı açık kaynaklı bir ders kitabı, kuantum hesaplama veya kuantum algoritmaları üzerine bir derse ek olarak üniversite düzeyinde mevcuttur (Wille ve diğ., 2019).

Qiskit açık kaynak kodlu bir online simülatörü olarak tanımlanmaktadır. Burada yerel bir makine içerisinda ya da çevrim içi olarak çalışan çok yönlü bir sistem görülmektedir. Bu simülatörlerle bahsedilmiş olan iki tane beş kubit sistem içerisinde testlerin yapılması sağlanmaktadır. Qiskit'te bir kod editörü (QASM) ile grafiksel bir kullanıcı arayüzün bulunduğu belirlenmiştir.

Qiskit'te sürükle bırak yöntemiyle kuantum devrenin şeması oluşturulabildiği belirlenmiştir. Burada OPENQASM kullanılarak doğrudan kod yazma işlemi gerçekleştirilmektedir.



Şekil 6.1. Qiskit SDK kullanılarak tasarlanmış devre örneği

7. KUANTUM MAKİNE ÖĞRENMESİ UYGULAMASI

7.1. Metodoloji

Bu tez çalışması kapsamında kuantum makine öğrenimi uygulaması gerçekleştirilmiştir. Kuantumsal Sinir Ağı kullanılarak Covid'19 Veri Setlerinin Analizi, Tahmini ve Değerlendirilmesi yapılmıştır. Yapılmış olan çalışmanın uygulama kısmı ilerleyen bölümlerde ayrıntılı olarak verilmiştir.

7.2. Veri seti ve ön işleme

Uygulamamızda sınıflandırmada kullandığımız veri seti https://www.kaggle.com/datasets/pranavraikokte/covid19-image-dataset adresinden elde edilmiştir. Veri setimiz göğüs röntgeni görüntülerinden oluşmaktadır. Test ve eğitim dizinlerinde yapılandırılmış Covid'19, Viral Pnömoni ve Normal Göğüs Röntgenlerini içermektedr. Bu verilerin dağılımına baktığımızda ise; test verisi için 26 adet Covid'19 verisi, 20 adet Normal ve 20 adet de Viral Pnömoni görüntüsü olmak üzere 66 adet görüntü bulunmaktadır. Eğitim veri setinin dağılımı ise, 111 adet Covid'19, 70 adet Normal ve 70 adet de Viral Pnömani olmak üzere toplamda 251 görüntüden oluşmaktadır. Veri kümesi, her biri 3 sınıfın görüntülerinden oluşan 3 klasör içeren eğitim ve test dizinlerine bölünmüştür.

Kullanılan veri seti modeli için 251 eğitim ve 66 test görüntüsü içermektedir. Veri setine ait görüntüler gerçek hayattaki göğüs röntgenidir ve önceden değiştirilmemiştir. Yani hepsinin farklı boyutları vardır. Bundan dolayı görsellerin ölçüsü belirli bir boyuta indirilmiştir. Bir görüntüyü yeniden boyutlandırmak, yalnızca genişlik, yalnızca yükseklik veya her ikisini birden değiştirmek olsun, boyutlarının değiştirilmesi anlamına gelir. Başlangıçta veri seti, her sınıfın tüm görüntülerinin farklı klasörlere yerleştirildiği bir klasör formatında kullanılmaktadır. Bu görselleri openCV kütüphanesini kullanarak 28x28'e dönüştürmek için bir Python betiği kullanılmıştır. Son olarak csv formatında kaydedilmiştir.

Bir görüntüyü yeniden boyutlandırmak için OpenCV, cv2.resize() fonksiyonuna sahiptir. Bu çalışmada ilgili fonksiyon kullanılarak boyut küçültme işlemi gerçekleştirilmiştir.

Desenlerin çoğunun bozulmadan kalması için görüntü boyutunu 256x256 olarak sabitlemek daha uygun olacaktır. Ancak hesaplama kaynaklarındaki bazı sınırlamalar nedeniyle bu çalışmada boyut 28x28 olarak tutulmuştur. Veri kümesindeki gerçek röntgen görüntüleri birçok bilgiyi barındıracak kadar büyüktür. Ancak hesaplama kaynaklarının eksikliği nedeniyle, openCV kütüphanesi kullanılarak boyut 28x28'e düşürülmüştür.

7.3. Performans Değerlendirmesi

Performans değerlendirmesi için doğruluk (accuracy), f1 skoru, kesinlik (precision) ve duyarlılık (recall) değerleri hesaplatılmıştır. Hata matrisleri elde edilmiştir. Bu hesaplamalar aşağıda yer alan formüllere göre yapılmıştır.

- **Doğruluk (accuracy) :** Modelde doğru tahmin edilen alanların toplam veri kümesine oranını veren metriktir.

$$Doğruluk = \frac{\mathrm{TP+TN}}{\mathrm{TP+FP+TN+FN}} \tag{7.1}$$

- **Kesinlik (precision) :** Pozitif olarak tahmin edilen değerlerin gerçekte kaçının pozitif olduğunun gözlemlendiği sonucunu veren bir metriktir.

$$\text{Kesinlik} = \frac{TP}{\mathrm{TP+FP}} \tag{7.2}$$

- **Duyarlılık (Recall):** Uygulamamızda bulunan sınıflar içinde pozitiv olarak tahmin etmemiz gereken işlemlerin ne kadarını pozitiv olarak gözlemlediğimizi gösteren bir metriktir.

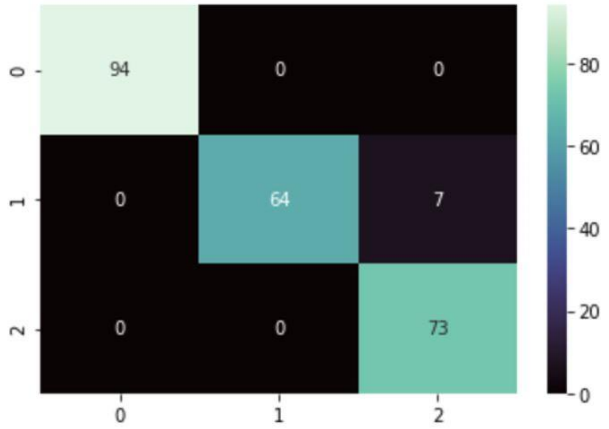
$$\text{Duyarlılık} = \frac{\mathrm{TP}}{\mathrm{TP+FN}} \tag{7.3}$$

- **F1 skoru:** Kesinlik (Precision) ve Duyarlılık (Recall) değerlerinin harmonik ortalamasını veren bir metriktir.

$$\mathrm{F} - Skor = 2 * \frac{\mathrm{precision*recall}}{\mathrm{precision+recall}} \tag{7.4}$$

7.4. Karmaşıklık Matrisi

Bu tez çalışmasında kuantum sınıflandırıcılar kullanılmadan önce transfer öğrenimi yoluyla Imagenet üzerinde ResNet-50 ağırlıklarıyla evrişimli bir sinir ağı eğitilmiştir. Bu sayede öznitelikler belirlenmiştir. Karmaşıklık matrisi Şekil 7.1'de verilmiştir.



Şekil 7.1. Karmaşıklık matrisi

Duyarlılık, kesinlik, doğruluk ve F1 skoru değerleri Tablo 7.1'de gösterilmiştir.

	Kesinlik	Duyarlılık	f1-skoru	Destek*
0	1	1	1	94
1	1	0.90	0.95	71
2	0.91	1.00	0.95	73
Doğruluk	0.97	0.97	0.97	238

*: Destek, o sınıfta yer alan gerçek yanıtın örneklerinin sayısıdır.

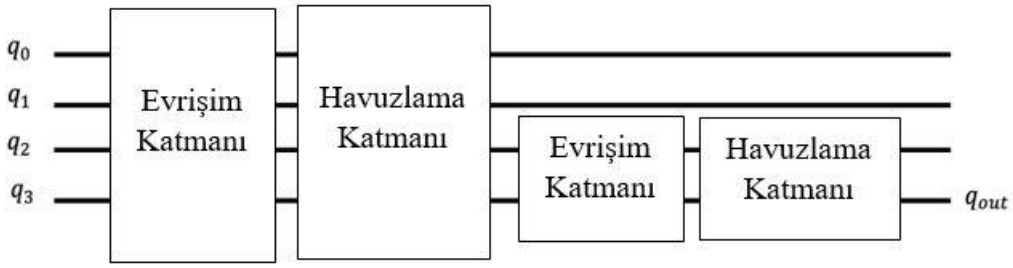
Tablo 7.1'de en sol sütunda yer alan 0,1 ve 2 değerleri sınıflandırmada kullanılan sınıfları ifade etmektedir. Bunlar sırasıyla Covid'19, Normal, Viral Pnömonidir.

Doğrulama setinde elde edilen accuracy puanı %96,97'dir. Ayrıca karmaşıklık matrisine baktığımızda temel tanı unsurlarının çok yüksek, diğerlerinin ise sıfır olduğunu görüyoruz. Anlaşılacağı üzere test setindeki tüm veri noktaları doğru şekilde sınıflandırılmaktadır. Performans değerlendirmesinde kullanılan bu metriklerin tanımları ise bölüm 7.3'te verilmiştir.

7.5. Kuantum Devresi

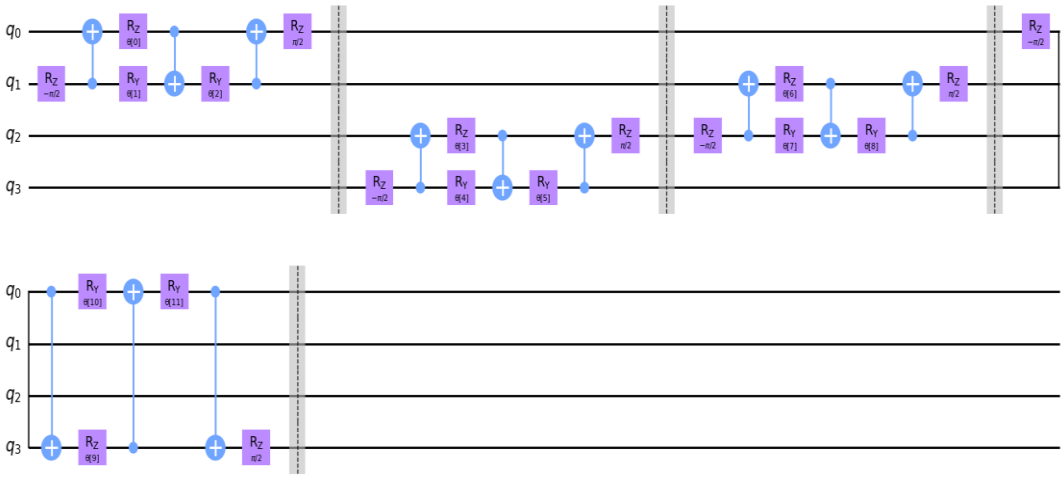
QCNN'de her katman parametrelendirilmiş devreler içerir; bu, her katmanın parametrelerini ayarlayarak çıktı sonucumuzu değiştirdiğimiz anlamına gelir.

QCNN'yi eğitirken, kayıp fonksiyonunu azaltmak için ayarlananlar bu parametrelerdir. Dört kübitlik QCNN örneği aşağıda Şekil 7.2'de gösterilmiştir.

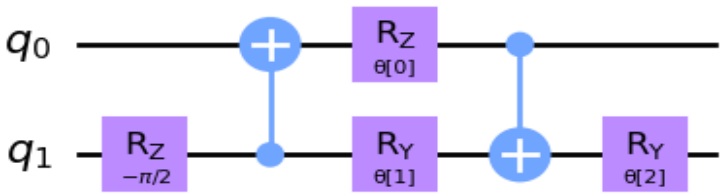


Şekil 7.2. Dört kübitlik QCNN örneği

Bu tez çalışması kapsamında kullanılan kuantum devresinin evrişim katmanı Şekil 7.3'te gösterilmiştir. Havuzlama katmanı ise Şekil 7.4'te gösterilmiştir.

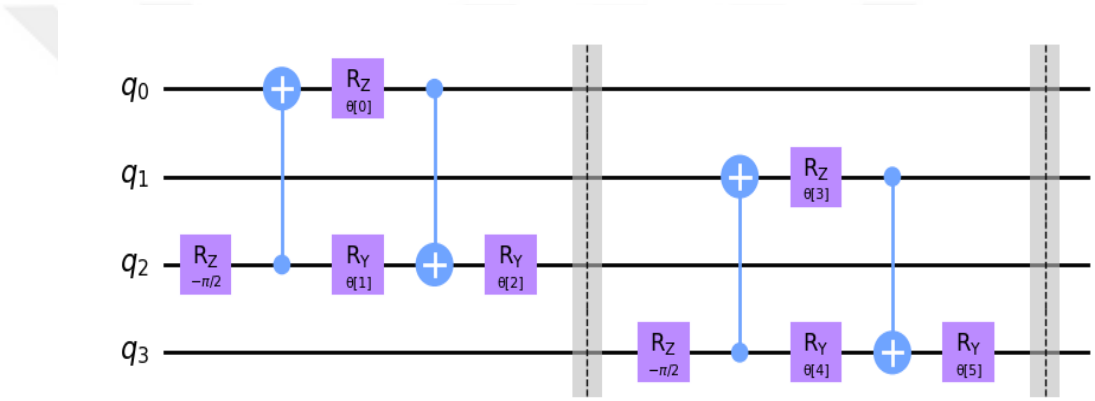


Şekil 7.3. Kuantum devresinde evrişim katmanı



Şekil 7.4. Kuantum devresinin havuzlama katmanı

Bu yaklaşımda, her havuzlama katmanındaki kübitleri göz ardı edilmiştir. N kübit Kuantum Devre boyutlarını N/2'ye dönüştüren bir QCNN Havuzlama Katmanı oluşturulmuştur. Dört kübit devrenin boyutsallığı son iki kubite, yani bu özel örnekteki son iki kübite indirgenmiştir. Bu kübitler daha sonra bir sonraki katmanda kullanılırken, ilk ikisi QCNN'nin geri kalanında ihmal edilir. Bu iki kübitlik üniter devreyi uyguladıktan sonra sonraki katmanlarda ilk kübiti (q0) ihmal edilmiş ve sadece ikinci kübit (q1) kullanılmıştır. N kübit için havuzlama katmanımızı oluşturmak amacıyla bu iki kübit havuzlama katmanı farklı kübit çiftlerine uygulanmıştır. Bu tez çalışmasında kuantum sınıflandırmada kullanılan devre Şekil 7.5'te gösterilmiştir.



Şekil 7.5. Kuantum devresi

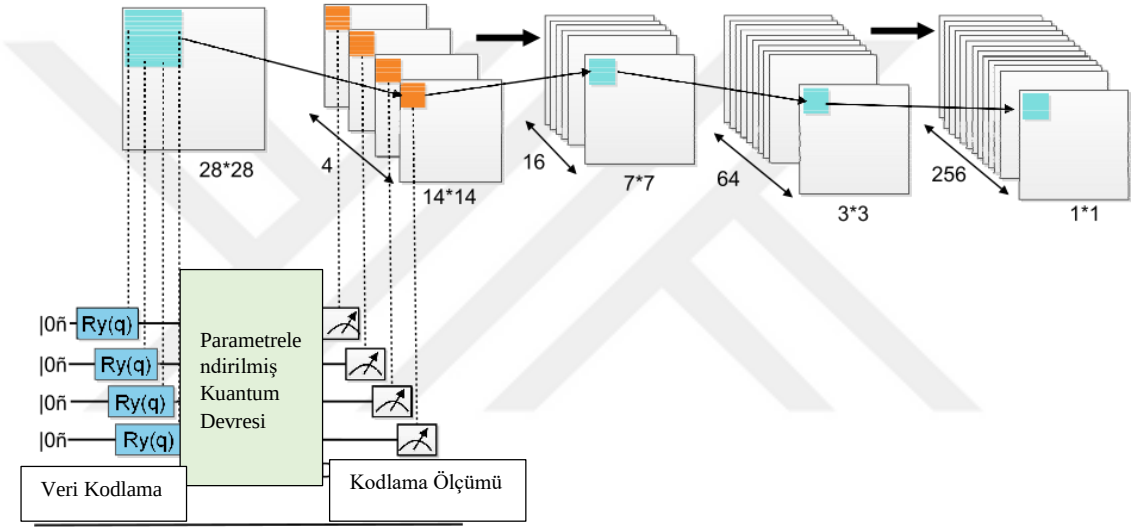
7.6. Evrişim Katmanının Uygulanması

Tek bir evrişim filtresinin, veri kümesindeki görüntülerin uzamsal olarak yerel alt bölümlerini girdi olarak alan rastgele bir kuantum devresi q kullandığı düşünülmüştür. Her giriş (ui), n*n boyutunda 2 boyutlu bir matristir; burada n >1'dir. 4 kübitlik bir sistemi simüle eden bir PennyLane default.qubit cihazı başlatılmıştır. Şekil 7.6 'da verilen Kuantum devresi aşağıdakilerden oluşmaktadır:

- Yerel Ry rotasyonlarının gömülü katmanı;
- n katmanlı, rastgele parametrelendirilmiş bir kuantum devresi;
- Hesaplamalı temelde 4 beklenti değerini tahmin eden son bir ölçüm.

Görüntü 2*2 piksellik karelere bölünür ve her kare kuantum devresi tarafından işlenir ve son olarak 4 beklenti değeri, tek bir çıkış pikselinin 4 farklı kanalına eşlenir.

Evrişimli Sinir Ağının tek katmanlı yaklaşımı bu modelde tam 4 katman olacak şekilde çoklu katmanlara genişletilmiştir. Başlangıçta her görüntü (28x28x1) boyutuna sahiptir ve bu boyut ilk Evrişim katmanıyla beslenerek (14x14x4) biçimine dönüştürülür. 2. Katman (7x7x16), 3. Katman (3x3x64) ve son olarak 4. ve son katman her birini (1x1x256) boyutlu bir veri matrisine dönüştürür. Kuantumsal Katman kapılarının parametreleri eşit şekilde rastgele olmasına rağmen, bu parametreleri eğitme yaklaşımı da göz önünde bulundurulmuştur ve bu uygulama çalışmasının genişletilmiş versiyonunda daha sonra sonuçta herhangi bir gelişme olup olmadığı değerlendirilmiştir.



Şekil 7.6: Evrişimli sinir ağının uygulanması

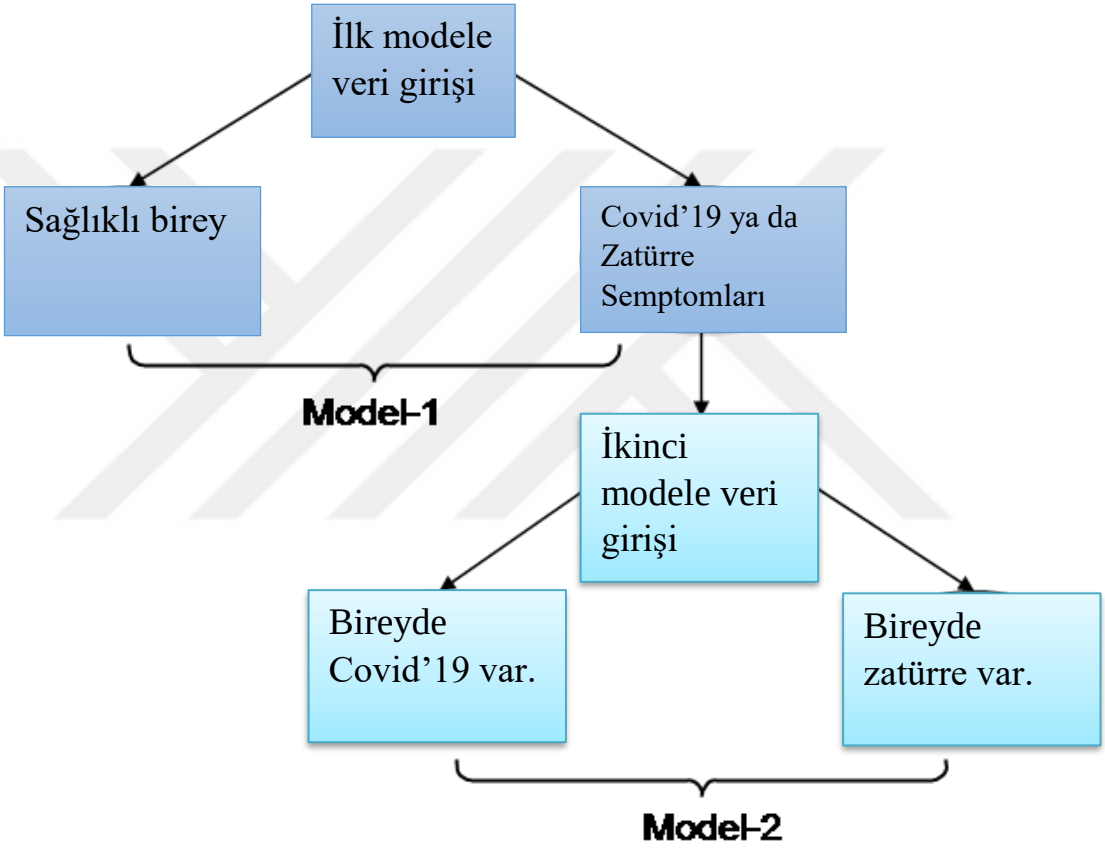
7.7. Sınıflandırıcı Modeli

Evrişim katmanlardan sonra sırada sınıflandırıcı modeli var. Sınıflandırıcı modeli, her biri ikili sınıflandırıcı olan iki alt sınıflandırıcıdan oluşmaktadır. Bu ikisi, 'Model-1' ve 'Model-2' olarak adlandırılmış olup, Şekil 7.7'de gösterilmiştir.

Model-1, 'Normal Kişi' ve 'Covid'19/Viral Pnömoni' olmak üzere iki sınıf arasında sınıflandırma yapar. Model-2, 'Covid'19' ve 'Viral Pnömoni' olmak üzere iki sınıf arasında sınıflandırma yapar.

Burada belirtilmesi gereken önemli bir nokta, başlangıçta çok sınıflı sınıflandırma için tek model kullanmaya çalışılmış olunması ve sonucun oldukça düşük çıkmasıdır. Dolayısıyla iki modelli yaklaşım son yaklaşım olarak değerlendirilmiştir.

Ayrıca bir diğer husus ise Model-1'in doğruluğu her zaman model-2'den daha yüksek oluşudur. Çünkü 'Normal Kişi' ile 'Covid’19/Viral Pnömoni'yi ayırt etmek 'Covid’19' ile 'Viral Pnömoni'yi ayırt etmekten daha kolaydır.



Şekil 7.7. Sınıflandırıcı modeli

Bunun için öznitelik boyutu küçültme tekniği ve özellik haritası devre tasarımı üç farklı yaklaşımla kullanılmıştır. Bu üç farklı kuantum sınıflandırıcısı ise şunlardır:

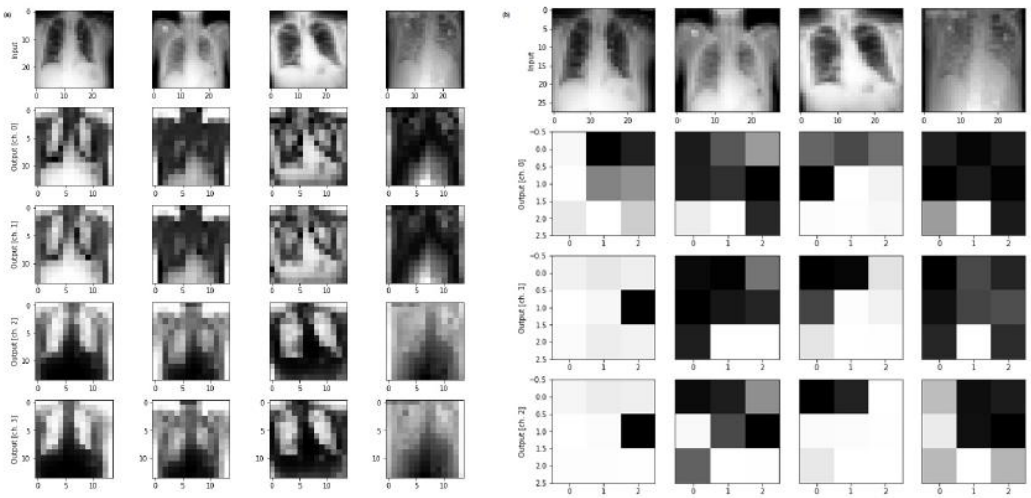
- Kuantum Sınıflandırıcısı 1'de, Temel Veri Analizi ile çıkarılan 256 öznitelik boyutlu girdi verisinden 11 öznitelik kullanılmıştır. Burada yaklaşık %70 doğruluk elde edilmiştir.

- Kuantum Sınıflandırıcısı 2’de TruncatedSVD yöntemini kullanarak her görüntünün 256 özniteliği 4'e indirilmiştir. Yaklaşık %72 doğruluk (accuracy) elde edilmiştir.
- Kuantum Sınıflandırıcısı 3’de verileri yalnızca 2 özniteliğe indirgenmiştir. Beklenmedik bir şekilde bu bize daha önce yaklaşılanların en yükseği olan %76 doğruluğu vermiştir.

7.8. Tahminleme

Tahmin yaparken öncelikle Model-1'e girdi verilmiştir. Normal kişi olarak tahmin yapıyorsa bu, girişe atanan son tahmindir. Değilse, aynı girdi Model-2'ye verilir ve sonunda göğüs röntgeninin hastalarda Covid’19 veya Viral Pnömoni olup olmadığını ortaya çıkaracağını tahmin edilir.

Her giriş görüntüsünün altında kuantum evrişimi tarafından oluşturulan 4 çıkış kanalı gri tonlamayla görselleştirilir. Kuantum çekirdeği ve çözünürlüğün aşağı örneklemesi (down sampling) tarafından bazı yerel bozulmaların ortaya çıktığı açıkça gözlemlenebilir. Bununla birlikte, bir evrişim katmanından beklendiği gibi görüntünün global şekli korunur. Bu durum şekil 7.8’de gösterilmiştir.

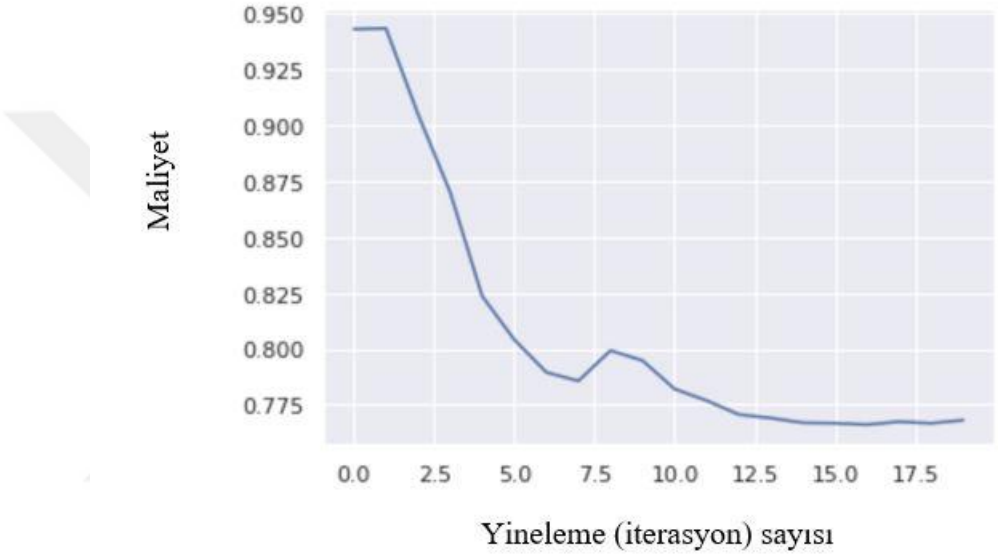


Şekil 7.8. (a) Sıkıştırılmış görüntüler 14*14. Şekil 7.8. (b) Covid’19 veri kümesindeki kuantumsal sinir ağı

7.9. Bulgular

7.9.1. Kuantum sınıflandırıcısı -1 için elde edilen bulgular

Model 1 ve model 2 için maliyet ve eğitim doğruluk grafikleri Şekil 7.9, 7.10, 7.11 ve 7.12'de gösterilmiştir. Kuantum Sınıflandırıcısı 1'de, Temel Veri Analizi ile çıkarılan 256 öznitelik boyutlu girdi verisinden 11 öznitelik kullanılmıştır. Burada yaklaşık %70 doğruluk elde edilmiştir.

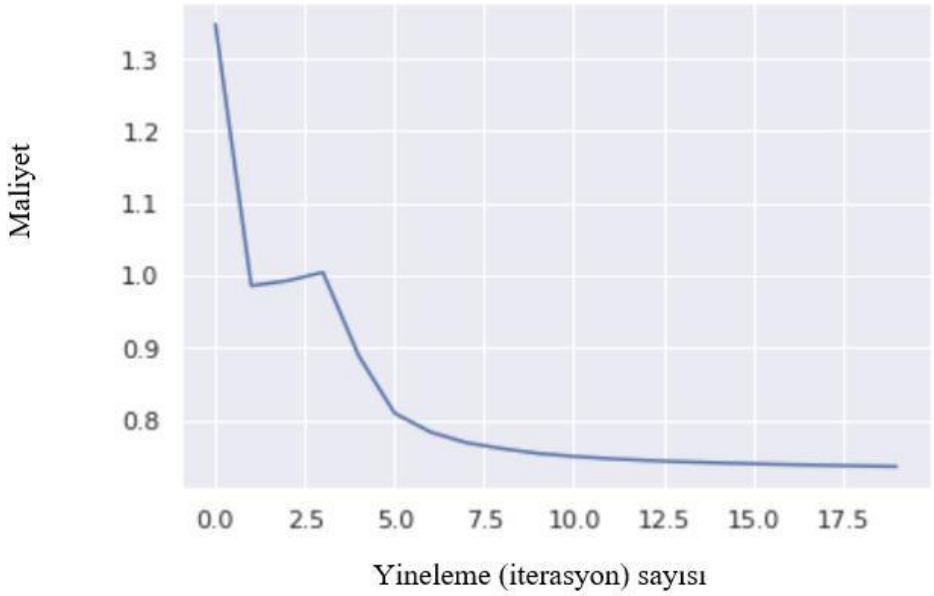


Şekil 7.9. Kuantum sınıflandırıcı-1, model - 1 için maliyet grafiği

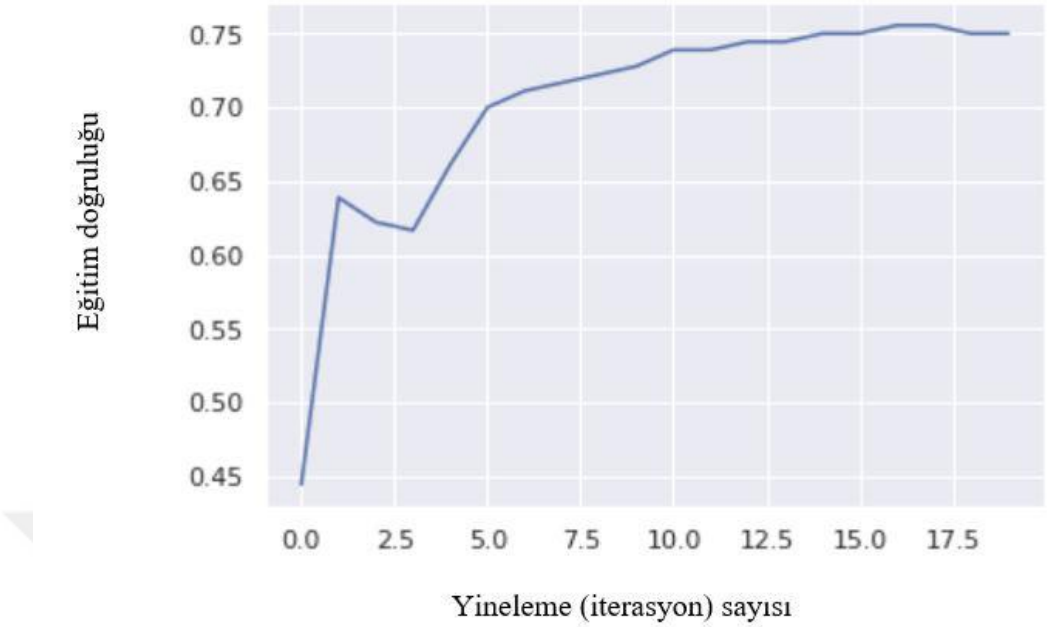
Maliyet fonksiyonları genel olarak modellerin performansını ölçmek için kullanılır. Yapılan çalışmada, yineleme sayısı belirli bir işlemin tekrarlanma sayısını ifade etmektedir. Yineleme sayısı, genel olarak optimal çözüme yakınsama için gerekli olan sayıyı temsil eder. Bu bağlamda Şekil 7.9'daki grafiğe göre iterasyon sayısı arttıkça model performansının arttığı gözlemlenmektedir.

Şekil 7.10. Kuantum sınıflandırıcı-1, model -1 için eğitim doğruluk grafiği

Eğitim doğruluğu, bir makine öğrenimi modelinin eğitim sırasında ne kadar iyi performans gösterdiğinin ölçüsüdür. Bu çalışmada Şekil 7.10'daki grafiğe göre iterasyon sayısı arttıkça model performansının arttığı gözlemlenmektedir.



Şekil 7.11. Kuantum sınıflandırıcı -1, model - 2 için maliyet grafiği



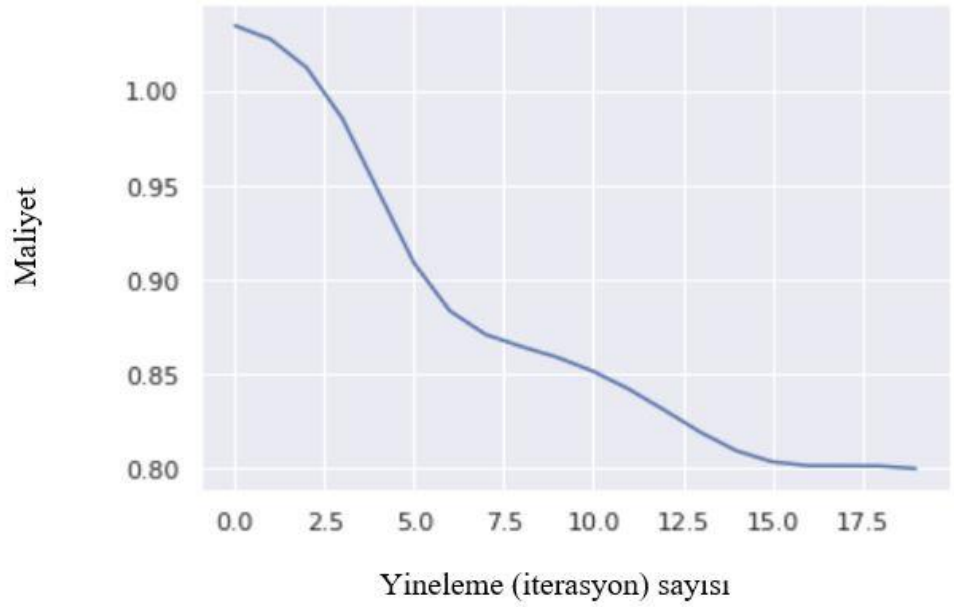
Şekil 7.12. Kuantum sınıflandırıcı-1, model - 2 için eğitim doğruluk grafiği

Yapılan çalışmada, yineleme sayısı belirli bir işlemi veya işlemin tekrarlanma sayısını ifade etmektedir. Bura da amaç, optimal çözüme yakınsama için gereken yineleme sayısını göstermektir. Bundan dolayı, Şekil 7.11'deki grafiğe göre iterasyon sayısı arttıkça model performansının arttığı gözlemlenmektedir.

Eğitim doğruluğu, bir makine öğrenimi modelinin eğitim sırasında ne kadar iyi performans gösterdiğinin ölçüsüdür. Bu bağlamda yapılan çalışmada, Şekil 7.12'deki grafiğe göre iterasyon sayısı arttıkça model performansının arttığı gözlemlenmektedir. Önceden de bahsedildiği üzere bir diğer husus ise Model-1'in doğruluğu her zaman model-2'den daha yüksek oluşudur. Bunun sebebi ise 'Normal Kişi' ile 'Covid'19/Viral Pnömoni'yi ayırt etmek 'Covid'19' ile 'Viral Pnömoni'yi ayırt etmekten daha kolay olmasıdır.

7.9.2. Kuantum sınıflandırıcısı - 2 için elde edilen bulgular

Yapılan uygulamada Kuantum Sınıflandırma için kullandığımız model 2 için maliyet ve eğitim doğruluk grafikleri Şekil 7.13, 7.14, 7.15 ve 7.16'de gösterilmiştir.

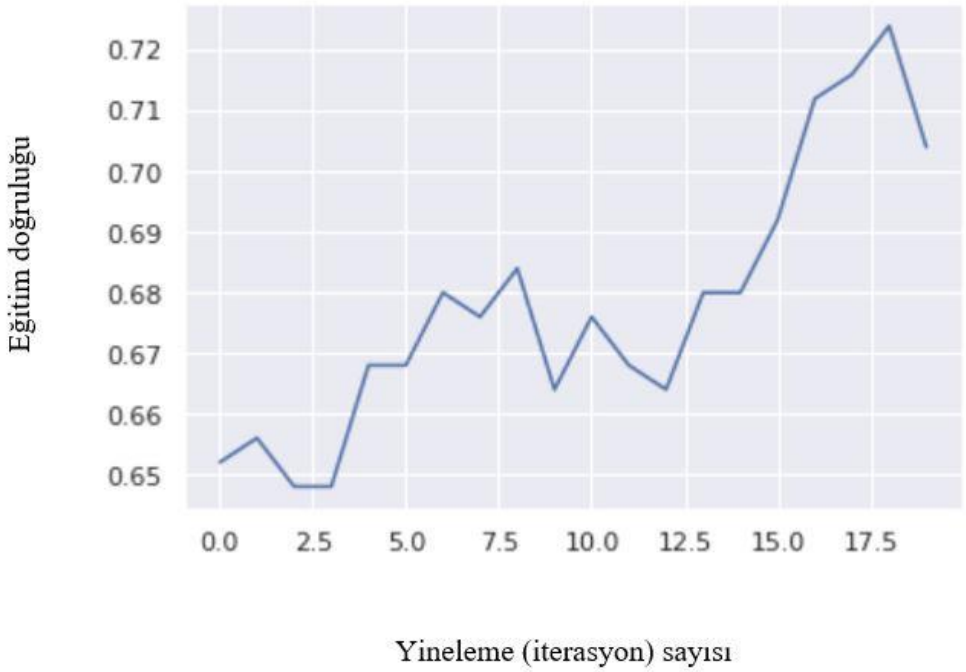


Şekil 7.13. Kuantum sınıflandırıcı- 2, model-1 için maliyet grafiği

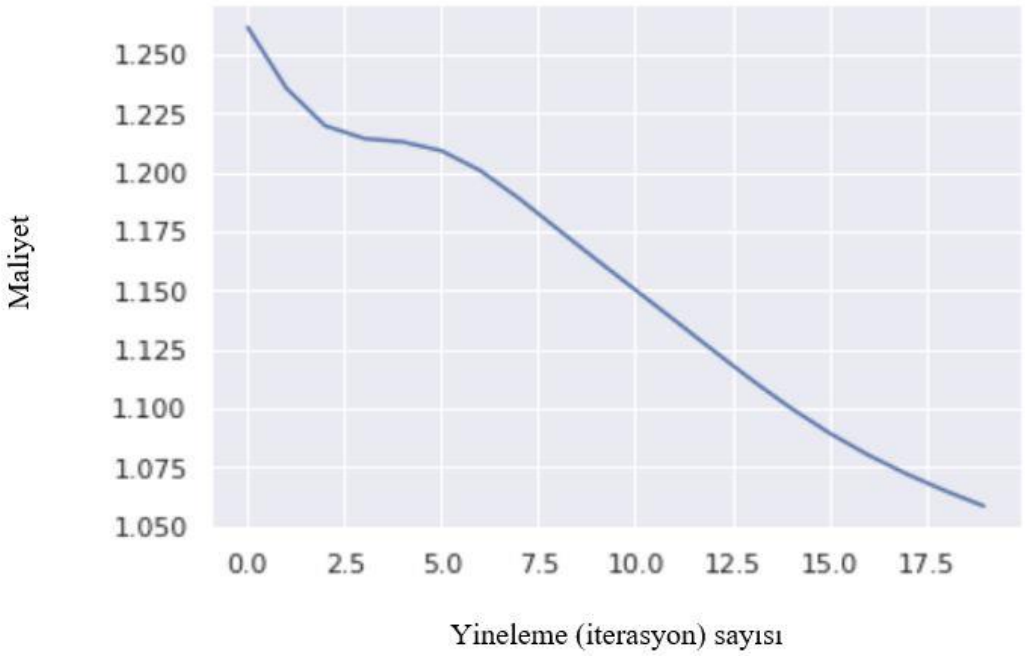
Uygulamada yapılan ikinci modellemede, maliyet fonksiyonları yineleme sayısı optimal çözüme yakınsadıkça Şekil 7.13'deki grafikte görüldüğü üzere model performansının arttığı gözlemlenmektedir.

Bu modelin eğitimi sırasında ne kadar iyi performans gösterdiği Şekil 7.14'deki grafikte görmekteyiz. Buna göre iterasyon sayısı arttıkça model performansının arttığı gözlemlenmektedir.

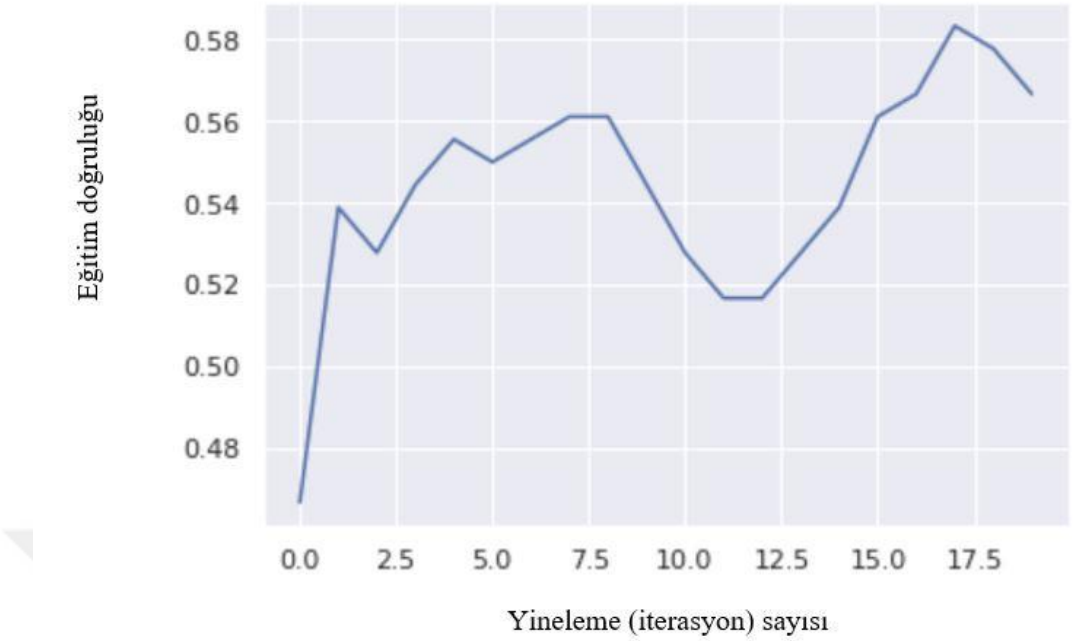
Model-2 için incelediğimiz bir diğer foksiyon ise maliyet fonksiyonudur. Modelimizin performansını göstermek için Şekil 7.15'deki grafik kullanılmaktadır. Bu bağlamda Şekil 7.15'deki grafiğe göre iterasyon sayısı arttıkça model performansının arttığı açıkça gözlemlenmektedir.



Şekil 7.14. Kuantum sınıflandırıcı-2, model-1 için eğitim doğruluk grafiği



Şekil 7.15. Kuantum sınıflandırıcı-2, model-2 için maliyet grafiği



Şekil 7.16. Kuantum sınıflandırıcı-2, model-2 için eğitim doğruluk grafiği

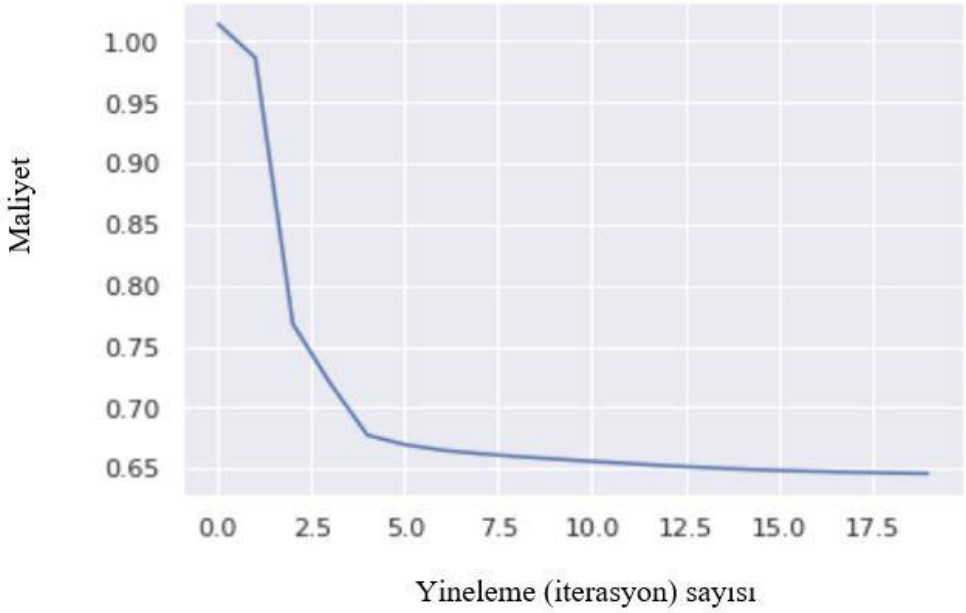
Yapılan eğitimin doğruluğu, modelinmizin eğitim sırasında ne kadar iyi performans gösterdiğinin ölçüsüdür. Bu bağlamda Şekil 7.16'daki grafiğe göre iterasyon sayısı arttıkça model performansının arttığı gözlemlenmektedir.

7.9.3. Kuantum sınıflandırıcısı -3 için elde edilen bulgular

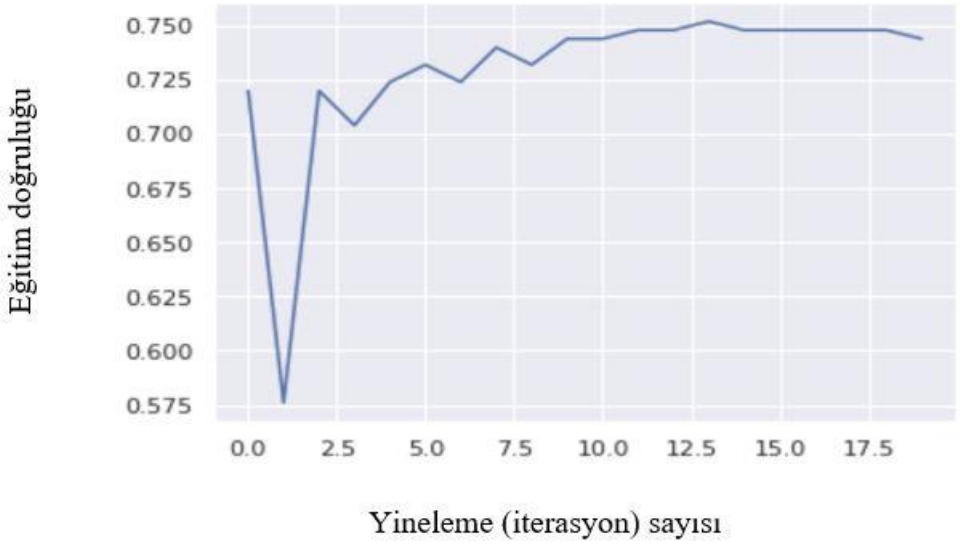
Model- 1 ve model- 2 için maliyet ve eğitim doğruluk grafikleri Şekil 7.17, 7.18, 7.19 ve 7.20'de gösterilmiştir.

Bu grafiklerde önceki modellerde olduğu gibi maliyet fonksiyonlarının performansını görmek için kullanıldı. Burada Şekil 7.17'deki grafiğe göre iterasyon sayısı arttıkça model performansının arttığı gözlemlenmektedir.

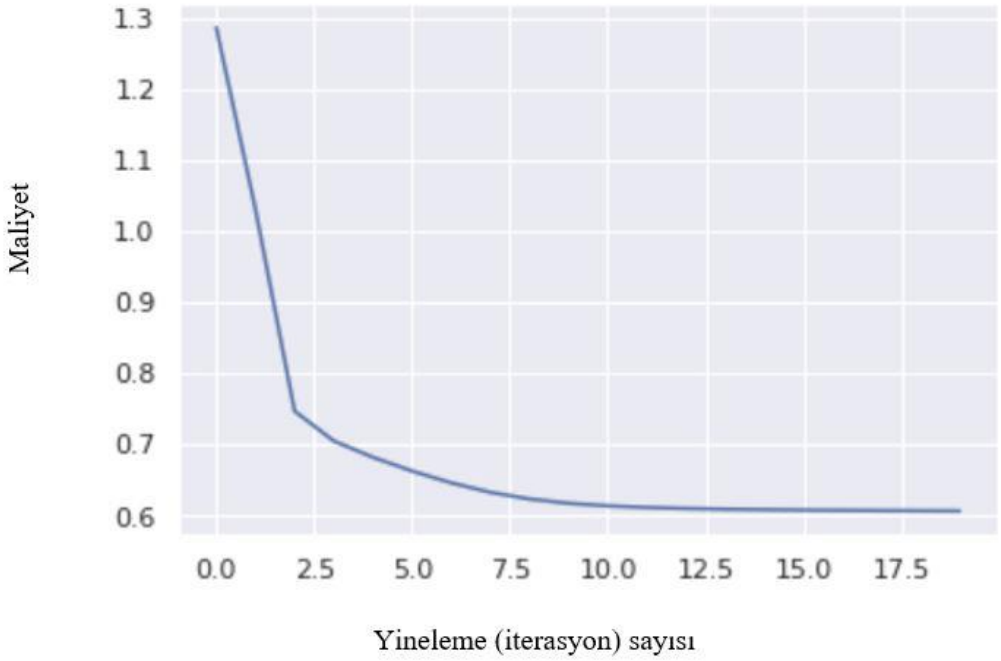
Oluşturulan sınıflandırıcı için modelin eğitim doğruluğu, performans değerleri açısından bir ölçüdür. Şekil 7.18'deki grafiğe göre iterasyon sayısı arttıkça model performansının arttığı gözlemlenmektedir.



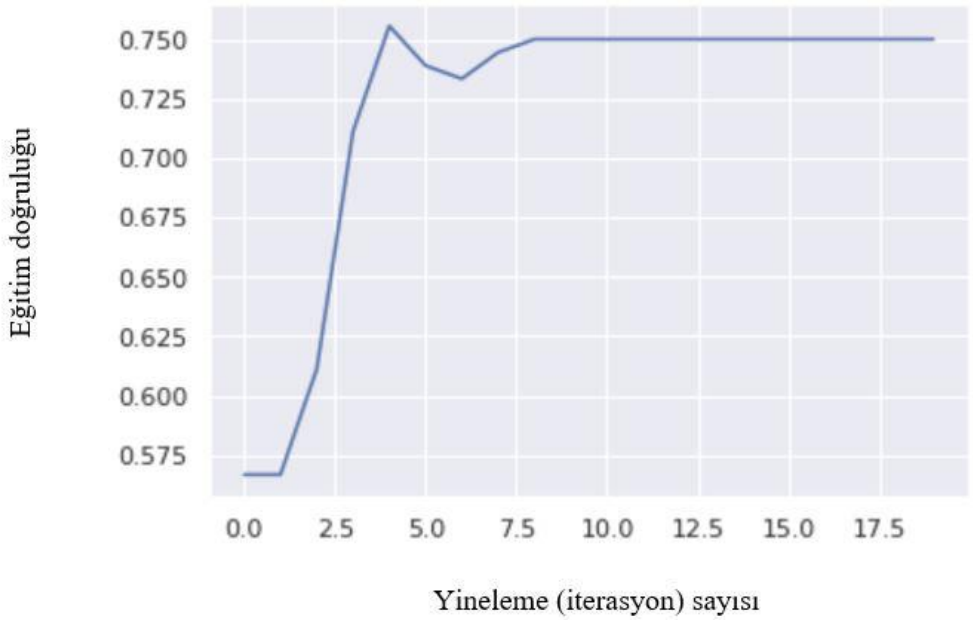
Şekil 7.17. Kuantum sınıflandırıcı-3, model-1 için maliyet grafiği



Şekil 7.18. Kuantum sınıflandırıcı-3, model-1 için eğitim doğruluk grafiği



Şekil 7.19. Kuantum sınıflandırıcı-3, model-2 için maliyet grafiği



Şekil 7.20. Kuantum sınıflandırıcı-3, model-2 için eğitim doğruluk grafiği

Bu grafiklerde bir önceki bölümlerde olduğu gibi maliyet fonksiyonları modellerin ne kadar performanslı çalıştığını görmek için kullanılır. Şekil 7.19’daki grafiğe göre iterasyon sayısı arttıkça model performansının arttığı gözlemlenmektedir.

Şekil 7.20’deki grafiğe göre iterasyon sayısı arttıkça model performansının arttığı gözlemlenmektedir.

Önceden de bahsedildiği üzere bir diğer husus ise Model-1'in doğruluğunun (accuracy) her zaman model-2'den daha yüksek oluşudur. Bunun sebebi ise 'Normal Kişi' ile 'Covid’19/Viral Pnömoni'yi ayırt etmek 'Covid’19' ile 'Viral Pnömoni'yi ayırt etmekten daha kolay olmasıdır.

8. SONUÇLAR VE ÖNERİLER

Kuantum evrişimli sinir ağları (QCNN'ler), kuantum hesaplamanın potansiyel olarak güçlü bazı yönlerinden yararlanarak CNN'lerin yeteneklerini genişletir. Bir dizi rastgele kuantum devresi kullanarak verileri yerel olarak dönüştürerek giriş verileri üzerinde çalışır.

Klasik evrişimli sinir ağlarının verimliliğinden yola çıkarak, Evrişimli sinir ağını (QNN'ler) kullanarak veriler analiz edilmiş, tahminler yapılmış ve sonuçlar değerlendirilmiştir. Kuantum halinde kodlanmış covid’19 veri setinin ikili sınıflandırması gerçekleştirilmiştir. Ayrıca Pennylane'in "varsayılan qubit" cihazındaki farklı parametreleri de dikkate alarak performansı araştırılmıştır.

Bu bağlamda, kuantum devrelerinin, polinom boyutlu klasik hesaplama kaynakları kullanılarak gerçekleştirilmesi mümkün olmayan karmaşık fonksiyonel ilişkileri modelleyebildiği gösterilmiştir. Kuantum devresinin sağlamış olduğu fayda klasik olarak anlaşılması zor olan oldukça karmaşık çekirdekler üretebilmesidir.

Veri kümesindeki gerçek röntgen görüntüleri birçok bilgiyi barındıracak kadar büyüktür. Ancak hesaplama kaynaklarının eksikliği nedeniyle, openCV kütüphanesi kullanılarak boyut 28x28'e düşürülmüştür, bu da birçok önemli bilgiyi bastırmış olabilir. Daha sonra, daha fazla hesaplama kaynağının bulunmasıyla birlikte, modelin doğruluğunu artırabilecek 256x256 boyutlu görüntüyü kullanmak mümkün olabilir.

Evrişim uygulanıp veriler düzleştirildikten sonra her görüntünün 256 özelliği elde edildi ve 11 özellik, kübit eksikliğinden dolayı özellik seçme yöntemiyle kullanıldı. Bu çalışma, kuantum sistemi hakkında fikir edinmek için daha fazla sayıda mevcut kübit ve kuantum devresinin gerçek zamanlı simülasyonu ile gerçek zamanlı bir kuantum bilgisayarında uygulanabilir. Dahası, daha fazla kübitin varlığıyla rastgele oluşturulmuş görüntü verileri üzerinde dört evrişim katmanın eğitimi test edilebilir. Sonuç olarak, geleceğe dönük çalışmalar ve öneriler aşağıda verilmiştir:

- Veri kümesindeki gerçek röntgen görüntüleri birçok bilgiyi barındıracak kadar büyüktür. Ancak önceden de açıklandığı üzere hesaplama kaynaklarının eksikliği nedeniyle, openCV kütüphanesini kullanarak boyutu

28x28'e düşürülmüştür, bu da birçok önemli bilgiyi bastırmış olabilir. Daha sonra modelin doğruluğunu artıracak 256x256 boyutlu görüntü ile denemeler yapılabilir.

- Şu anda evrişimi uyguladıktan ve verileri düzleştirdikten sonra her görüntünün 256 özelliğine sahip olmamıza rağmen, çeşitli öznitelik boyutu küçültme yöntemleriyle sırasıyla Kuantum Sınıflandırıcı-1, Kuantum Sınıflandırıcı-2 ve Kuantum Sınıflandırıcı-3'te yalnızca 11 öznitelik, 4 öznitelik ve 2 öznitelik kullanılmıştır. Daha yüksek boyutlu verilerle doğruluğun artıp artmadığını veya kuantum bilgisayarların yalnızca daha az sayıda özellik ile daha iyi çalışıp çalışmadığını denemek için daha fazla öznitelik denenebilir.
- Görüntü verilerine uygulanan dört evrişim katmanın tümü, daha fazla eğitilmeyen, tek biçimli oluşturulmuş rastgele parametreler kullanmaktadır. Daha sonra, değiştirilmiş veri kümesinin gerçek görüntüler hakkında çok daha fazla veri içerebilmesi için bu evrişim katmanların da eğitilmesi söz konusu olabilir.
- Son olarak, daha doğru simülatörler üzerinde eğitilen modellerin eğitim sonucunu ve nihai doğruluğu elde edilmesi ve daha gerçekçi deneysel veriler elde etmek için bunları yavaş yavaş gerçek kuantum bilgisayarlarında çalıştırmayı denemek doğruluk (accuracy) değerlerinin artmasını şüphesiz ki sağlayacaktır.

9. KAYNAKLAR

Abraham, A. (2005). *Artificial neural networks. Handbook of measuring system design*.

Acar, E., Yılmaz, İ. (2021). *COVID-19 detection on IBM quantum computer with classical quantum transfer learning*, Turkish J. Electr. Eng. Comput. Sci, 29, 46–61, DOI: 10.3906/elk-2006-94

Adachi S. H. ve Henderson, M. P. (2015). *Application of Quantum Annealing to Training of Deep Neural Networks*. https://arxiv.org/abs/1510.06356.

Aksoy, B. , Bayrakçı, H. C. , Bayrakçı, E. ve Uğuz, S. (2017). Büyük Verinin Kurumlarda Kullanımı. *Süleyman Demirel Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi*, Kayfor 15 Özel Sayısı, 1915-1920.

Aktan, E. (2018). Büyük veri: Uygulama alanları, analitiği ve güvenlik boyutu. *Bilgi Yönetimi*, 1(1), 1-22.

Alabdullah, B., Beloff, N.ve White, M. (2018). *Rise of Big Data - Issues and Challenges*. In 2018 21st Saudi Computer Society National Computer Conference (NCC) (pp. 1–6). Riyadh, Saudi Arabia: IEEE. https://doi.org/10.1109/NCG.2018.8593166.

Albayrak, A.S. ve Yılmaz, K.S. (2009). Veri madenciligi: karar agacı algoritmaları ve _MKB verileri üzerine bir uygulama", *Süleyman Demirel Üniversitesi İktisadi ve 0dari Bilimler Fakültesi Dergisi*,14(1), 31-52.

Alcan, U. (2022). *Büyük Veri Araçları İle Covid-19 Pandemisinin İstanbul Trafiğine Etkisinin Analizi.* (Yüksek Lisans Tezi). İstanbul Üniversitesi-Cerrahpaşa Lisansüstü Eğitim Enstitüsü Elektrik-Elektronik Mühendisliği Anabilim Dalı.

Alimi, O. A., Ouahada, K. ve Abu-Mahfouz, A. M. (2020). *A review of machine learning approaches to power system security and stability*. *IEEE Access*, 8, 113512-113531.

AlMoammar, A., AlHenaki, L. ve Kurdi, H. (2018). *Selecting accurate classifier models for a MERS-CoV dataset*, Intell. Syst. Appl. Proc. 2018 Intell. Syst. Conf., 1, 868, 1070–1084, doi: 10.1007/978-3-030-01054-6_74

Amasyalı, M.F. (2008). *Yeni makine ögrenmesi metotları ve ilaç tasarımına uygulamaları.* (Doktora Tezi). Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü.

Amasyalı, M.F. (2010). Yıldız Teknik Üniversitesi Makine Ögrenmesine Giris (*M.Fatih Amasyalı Ders Notları*)" http://www.ce.yildiz.edu.tr/mygetfile.php?id=868 (2010).

Amin, M. H. (2009). *Physical Review Letters*. 102, 220401, ISSN 0031-9007.

Amin, M. H., Andriyash, E., Rolfe, J., Kulchytskyy, B. ve Melko, R. (2018). Quantum boltzmann machine. *Physical review letters*, 8, 021050.

Amin, M. H., Andriyash, E., Rolfe, J., Kulchytskyy, B. ve Melko, R. (2016). *Quantum Boltzmann Machine*. https://arxiv.org/abs/1601.02036.

Amisha, P. Malik, M. Pathania ve Rathaur, V. K. (2019). *Overview of artificial intelligence in medicine*, J. Fam. Med. Prim. Care, 8 (7), 2328–2331, doi: 10.4103/jfmpc.jfmpc_440_19

Andhavarapu, A. (2017). *Learning Elasticsearch*. Packt Publishing Ltd.

Apache Software Foundation (2017). *A Quick Overview*. https://solr.apache.org/guide/7_2/a-quick-overview.html

Bacon, D., van Dam, W. (2010). *Recent progress in quantum algorithms*. Commun. *ACM* 53(2), 84–93.

Bajer, M. (2017). *Building an IoT data hub with elasticsearch, Logstash and Kibana*. In 2017 5th international conference on future internet of things and cloud workshops (FiCloudW), 63-68. *Building an IoT data hub with elasticsearch, Logstash and Kibana*. In 2017 5th international conference on future internet of things and cloud workshops (FiCloudW), 63-68. IEEE.

Barbierato, E., Gribaudo, M. ve Iacono, M. (2013, December). *Modeling apache hive based applications in big data architectures*. In Proceedings of the 7th International Conference on Performance Evaluation Methodologies and Tools (pp. 30-38).

Behrman, E. C., Steck, J. E., Kumar, P. ve Walsh, K. A. (2008). Quantum Algorithm design using dynamic learning. *Quantum Information and Computation*. 8(1ve2), 12–29.

Bennett, C., Bernstein, E., Brassard, G., Vazirani, U. (1997). Strengths and weaknesses of quantum computing. *SIAM J. Comput*. 26(5), 1510–1523.

Bercovich, E., Javitt, M. C. (2018). Medical imaging: from roentgen to the digital revolution, and beyond, *Rambam Maimonides Med. J*., 9(4), e0034, doi: 10.5041/RMMJ.10355

Bernstein, E. ve Vazirani, U. (1993, June). *Quantum complexity theory. In Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*, 11-20.

Bhattacharyya, S., Pan, I., Mani, A., De, S., Behrman, E. ve Chakraborti, S. (Eds.). (2020). *Quantum Machine Learning* (Vol. 6). Walter de Gruyter GmbH ve Co KG.

Biamonte, J., Wittek, P., Pancotti, N., Rebentrost, P., Wiebe, N. ve Lloyd, S. (2017). *Quantum machine learning*. Nature, 549(7671), 195-202.

Birjali, M., Beni-Hssane, A.ve Erritali, M. (2017). *Analyzing social media through big data using infosphere biginsights and apache flume*. Procedia computer science, 113, 280-285.

Bonner, R., Freivalds, R. (2002). A survey of quantum learning. In: Bonner, R., Freivalds, R. (Eds.), Proceedings of QCL-02, 3rd *International Workshop on Quantum Computation and Learning. M¨alardalen University Press, V¨asterås and Eskilstuna.*

Borthakur, D. (2007). *The hadoop distributed file system: Architecture and design.* Hadoop Project Website, 11(2007), 21.

Brown, M. (2015). *Learning Apache Cassandra*. Packt Publishing Ltd.

Camacho-Rodríguez, J., Chauhan, A., Gates, A., Koifman, E., O'Malley, O., Garg, V., ... & Hagleitner, G. (2019, June*). Apache hive: From mapreduce to enterprise-grade big data warehousing*. In Proceedings of the 2019 International Conference on Management of Data, 773-1786.

Cappart, Q., Goutierre, E., Bergman, D. ve Rousseau, L. M. (2019). *Improving optimization bounds using machine learning: Decision diagrams meet deep reinforcement learning*. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 33, No. 01, 1443-1451).

Carbone, P., Katsifodimos, A., Ewen, S., Markl, V., Haridi, S.ve Tzoumas, K. (2015). *Apache flink: Stream and batch processing in a single engine*. Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, 36(4).

Carleo, G., Nomura, Y. ve Imada, M. (2018). *Constructing exact representations of quantum many-body systems with deep neural networks*. Nature communications, 9(1), 1-11.

Carr, J. (2014). An introduction to genetic algorithms. *Senior Project*, 1(40), 7.

Cassandra, A. (2014). Apache cassandra. Website. Available online at http://planetcassandra. org/what-is-apache-cassandra, 13.

Chancellor, N., Szoke, S., Vinci, W., Aeppli, G. ve Warburton, P. A. (2016). *Maximum-Entropy Inference with a Programmable Annealer*. Scientific Reports, 6(22318).

Chebotko, A., Kashlev, A.ve Lu, S. (2015). *A big data modeling methodology for Apache Cassandra*. In 2015 IEEE International Congress on Big Data,238-245). IEEE.

Chen, C. L., Li, H. X. ve Dong, D. Y. (2008). Hybrid control for Robot Navigation: A hierarchical Q-learning algorithm. *IEEE Robotics ve Automation Magazine*. 2008, 15(2),37–47.

Choi, V. (2010). *Adiabatic Quantum Algorithms for the NP-Complete Maximum-Weight Independent Set. Exact Cover and 3SAT Problems*. http://arxiv.org/abs/1004.2226.

Cong, I., Choi, S. ve Lukin, M. D. (2018). *Quantum convolutional neural networks*. 2018. arXiv:1810.03787v1 (quant-ph).

Coxson, G. E., Hill, C. R. ve Russo, J. C. (2014). *Adiabatic quantum computing for finding low-peaksidelobe codes*. High Performance Extreme Computing conference, IEEE 7. 3910–3916. doi:10.1039/b509983h.

Cunningham, P., Cord, M. ve Delany, S. J. (2008). *Supervised learning. In Machine learning techniques for multimedia*, 21-49. Springer, Berlin, Heidelberg.

da Silva, A. J., Ludermir, T. B. ve de Oliveira, W. D. (2016*). Quantum perceptron over a field and neural network architecture selection in a quantum computer*. Neural Networks, 76, 55–64.

Dalyan, T. (2006). *Makine ögrenmesinde 1R algoritması ve ikinci kuralın (2R) oluşturulması.* [Yüksek Lisans Tezi]. Kocaeli Üniversitesi Fen Bilimleri Enstitüsü, Kocaeli, 14,17.

De Falco, D. ve Tamascelli, D. (2011). *An introduction to quantum annealing RAIRO* – Theoretical Informatics and Applications., 45, 99–116.

Desai, P. V. (2018). *A Survey on Big Data Applications and Challenges*. In 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT 2018) (pp. 737–740). Coimbatore, India: IEEE. https://doi.org/10.1109/ICICCT.2018.8472999.

Dewi, P. S. ve Kurniawan, E. (2022). A E-Repaır Desıgn And E-Crm Utılızatıon On Mustıka Com Based On The Websıte. *Journal Of Dynamıcs (International Journal of Dynamics in Engineering and Sciences)*, 7(1), 110-113.

Dittrich, J.ve Quiané-Ruiz, J. A. (2012*). Efficient big data processing in Hadoop MapReduce. Proceedings of the VLDB Endowment*, 5(12), 2014-2015.

Doğan, K.ve Arslantekin, S. (2016). Büyük Veri: Önemi, Yapısı Ve Günümüzdeki Durum. *Ankara Üniversitesi Dil ve Tarih-Coğrafya Fakültesi Dergisi*, 56(1), 15-36.

Dong, D. Y., Chen, C. L. ve Chen, Z. H. (2005). *Quantum reinforcement learning*. Proceedings of the 1st International Conference on Natural Computing, 3611, 686–689.

Du, D. (2015). *Apache Hive Essentials*. Packt Publishing Ltd.

Dunjko, V. ve Briegel, H. J. (2018). *Machine learning ve artificial intelligence in the quantum domain: a review of recent progress*. Reports on Progress in Physics, 81(7), 074001.

Ekmekçi, U. (2021). *Hadoop ecosystem*, https://techbros.com.tr/bigdata/hadoop ecosystem/.

Er, M. J ve Deng, C. (2004). *Online tuning of fuzzy inference systems using dynamic fuzzy Q-learning*. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics). 2004, 34(3), 1478–1489.

Espinosa, C. V., Martin-Martin, E., Riesco, A. ve Rodríguez-Hortalá, J. (2019). FlinkCheck: *property-based testing for apache flink*. IEEE Access, 7, 150369-150382.

Ester, M. ve Sander, J. (2013). *Knowledge discovery in databases: Techniken und Anwendungen*. Springer-Verlag.

Farhi, E., Goldstone, J., Gutmann, S., Sipser, M. (2000). *Quantum computation by adiabatic evolution*. arXiv:quant-ph/0001106.

Farhi, E., Goldstone, J. ve Gutmann, S. A (2014). *quantum approximate optimization algorithm.* arXiv:1411.4028 (quant-ph).

Friedman, E.ve Tzoumas, K. (2016). *Introduction to Apache Flink: stream processing for real time and beyond*. " O'Reilly Media, Inc.".

Fuad, A., Erwin, A.ve Ipung, H. P. (2014). *Processing performance on apache pig, apache hive and MySQL cluster*. *In Proceedings of International Conference on Information, Communication Technology and System* (ICTS) 2014, 297-302. IEEE.

Fuchs, C. (2002). *Quantum mechanics as quantum information (and only a little more)*. arXiv:quant-ph/0205039.

Gheorghe, R., Hinman, M. L.ve Russo, R. (2015). *Elasticsearch in action. Shelter Island*, NY: Manning.

Ghosh, S., Opala, A., Matuszewski, M., Paterek, T. ve Liew, T. C. (2019). *Quantum reservoir processing*. npj Quantum Information, 5(1), 1-6.

Giudici, P. (2005). *Applied data mining: statistical methods for business and industry*. John Wiley ve Sons.

Grishikashvili, K., Dibb, S.ve Meadows, M. (2014). *Investigation into Big Data Impact on Digital Marketing*. Online Journal of Communication and Media Technologies, (Special Issue), 26–37.

Hadoop, Y. A. (2013). *Yet another resource negotiator. In Proceedings of the 4th Annual Symposium on Cloud Computing* (SOCC'13) (Vol. 5, p. 16). ACM.

Halevi, G. (2012). The 30th issue of Research Trends : *Special Issue on Big Data.* http://www.researchtrends.com/wpcontent/uploads/2012/09/Research_Trends_ Issue30.pdf

Haloi, S. (2015). *Apache zookeeper essentials*. Packt Publishing Ltd.

Harrow, A. W., Hassidim, A. ve Lloyd, S. (2009). *Quantum algorithm for solving linear systems of equations*. Physical review letters, 15(103), 150502.

Hemdan, E., Shouman, M, A. ve Karar, M. (2021). Covidx-net: *A framework of deep learning classifiers to diagnose covid-19 in x-ray images* https://arxiv.org/abs/2003.11055

Hemsoth, N. (2018). *QISKit Developments Key to IBM Quantum Engagement*.

Hoffman, S. (2013). *Apache Flume: distributed log collection for Hadoop*. Packt Publishing Ltd.

Hogg, T. ve Portnov, D. (2000). *Quantum optimization*. Information Sciences, 128(3),81–197.

Holzinger, A. (2015). *Data mining with decision trees: theory and applications*. Online Information Review.

Hu, F., Wang, B. N., Wang, N. ve Wang, C. (2019). *Quantum machine learning with D-wave quantum computer*. Quantum Engineering, 1(2), e12.

Huembeli, P., Dauphin, A. ve Wittek, P. (2018). *Identifying quantum phase transitions with adversarial neural networks*. Physical Review *B*, 97(13), 134109.

Huggins, W., Patil, P., Mitchell, B., Whaley, K. B. ve Stoudenmire, E. M. (2019). *Towards quantum machine learning with tensor networks*. Quantum Science and technology, 4(2), 024001.

Hurwitz, J., Nugent, A., Halper, F.ve Kaufman, M. (2013). *Big Data for Dummies* (1st ed). Hoboken, New Jersey: John Wiley & Sons, Inc.

Iqbal, M. H.ve Soomro, T. R. (2015). Big data analysis: Apache storm perspective. *International journal of computer trends and technology*, 19(1), 9-14.

J. Zurada, S. Lonial (2005). *Comparison of the performance of several data mining methods for bad debt recovery in the healthcare industry*, Journal of Applied Business Research 21:2, 37–54.

J. Zurada, W. Karwowski, W. Marras (2004*). Classification of jobs with risk of low back disorders by applying data mining techniques*, Occupational Ergonomics 4 (4), 291–305.

J.M. Zurada, A.S. Levitan, J. Guan (2006). *Non-conventional approaches to property value assessment*, Journal of Applied Business Research 22:3, 1–14.

Jansen, S., Ruska, M. B. ve Seiler, R. (2007). *Bounds for the adiabatic approximation with applications to quantum computation*. Journal of Mathematical Physics, 48, 102111, ISSN 00222488.

Jasim Hadi, H., Hameed Shnain, A., Hadishaheed, S.ve Haji Ahmad, A. (2015). *Big Data and Five V'S Characteristics*. International Journal of Advances in Electronics and Computer Science, 2(1), 2393–2835.

Javatpoint (2021). *What is Sqoop*. https://www.javatpoint.com/what-is-sqoop.

Johansson, N. ve Larsson, JÅ. (2017). *Efficient classical simulation of the Deutsch–Jozsa and Simon's algorithms*. Quantum Inf Process, 16, 233.

Houssein, E. H., Abohashima, Z., Elhoseny M. ve Mohamed, W.M. (2021). *Hybrid quantum convolutional neural netarxworks model for COVID-19 prediction using chest X-Ray images* [Online]. https://arxiv.org/abs/2102.06535v1

Katsifodimos, A., Ewen, S., Markl, V., Haridi, S.ve Tzoumas, K. (2015). Apache flink: Stream and batch processing in a single engine. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 36(4).

Kaur, K. (2016). Big Data Barriers and Opportunities. *International Journal of Advanced Research in Computer Science,* 7(6 (Special Issue)), 263–266.

Kaya, M. ve Alhajj, R. (2005). *A novel approach to multiagent reinforcement learning: Utilizing OLAP mining in the learning process. IEEE Transactions on Systems, Man, and Cybernetics, Part C* (Applications and Reviews), 35(4),582–590.

Kerenidis, I. ve Prakash, A. (2017). *Quantum gradient descent for linear systems and least squares.* arXiv:1704.04992 (quant-ph).

Khoshaman, A., Vinci, W., Denis, B., Andriyash, E. ve Amin, M. H. (2018). *Quantum variational autoencoder*. arXiv:1802.05779v1 (quant-ph).

Knott, P. A. (2016). A search algorithm for quantum state engineering and metrology. *New Journal of Physics*, 18(7), 073033.

Kondo,T. ve Ito, K. (2004). *A reinforcement learning with evolutionary state recruitment strategy for autonomous mobile robots control. Robotics and Autonomous Systems*, 46(2),111–124.

Köseoğlu, Ö.ve Demirci, Y. (2017). Türkiye'de Büyük Veri ve Veri Madenciliğine İlişkin Politika ve Stratejiler: Ulusal Politika Belgelerinin İçerik Analizi. *Süleyman Demirel Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi*, 22(Kayfor 15 Özel Sayısı), 2223-2239.

Krenn, M., Malik, M., Fickler, R., Lapkiewicz, R. ve Zeilinger, A. (2016). *Automated search for new quantum experiments*. Physical review letters, 116(9), 090405.

Kuc, R.ve Rogozinski, M. (2013). Elasticsearch server. Packt Publishing Ltd.

Langley, P. (2011). The changing science of machine learning. *Machine learning*, 82(3), 275-279.

Lavor, C., Manssur, L. R. U. ve Portugal, R. (2003). *Grover's Algorithm: Quantum Database Search.* arXiv:quant-ph/0301079.

Lloyd, S., Mohseni, M., Rebentrost, P. (2013). *Quantum algorithms for supervised and unsupervised machine learning*. arXiv:1307.0411.

Lloyd, S., Mohseni, M. ve Rebentrost, P. (2014). Quantum principal component analysis. *Nature Physics*, 10(9),631–633.

Lloyd, S. ve Weedbrook, C. (2018). Quantum generative adversarial learning. *Physical review letters*, 040502.

Lohr, S. (2013). *The Origins of 'Big Data' An Etymological Detective Story*. Retrieved 11 December 2020, from https://bits.blogs.nytimes.com/2013/02/01/the-origins-of-big-data-an-etymological-detective-story/

Lucas, A. (2014). Ising formulations of many NP problems. *Front Physics*, 12(5).

Lyubimov, D.ve Palumbo, A. (2016). Apache mahout: beyond MapReduce. North Charleston, SC: *CreateSpace Independent Publishing*.

Mahesh, B. (2020). Machine learning algorithms-a review. *International Journal of Science and Research (IJSR),* 9, 381-386.

Mahrt, M.ve Scharkow, M. (2013). The Value of Big Data in Digital Media Research. *Journal of Broadcasting & Electronic Media*, 57(1), 20–33.

Maimon, O. Z. ve Rokach, L. (2014). Data mining with decision trees: theory and applications (Vol. 81). *World scientific*.

Manyika, J., Chui, M., Brown, B., Bughin, G., Dobbs, R., Roxburgh, C.ve Byers, A. H. (2011). *Big Data : The Next Frontier for Innovation, Competition, and Productivity.*

Manyika, J., Chui, M., Brown, B., Bughin, G., Dobbs, R., Roxburgh, C.ve Byers, A. H. (2011). Big Data : *The Next Frontier for Innovation, Competition, and Productivity.*

Marzec, M. (2014). Portfolio Optimization: *Applications in Quantum Computing. Social Science Research Network Technical Report*.

Matsuda, Y., Nishimori, H. ve Katzgraber, H. G. (2009). *Ground-state statistics from annealing algorithms: Quantum vs classical approaches*. arXiv:0808.0365.

McGeoch, C.C., Wang, C. (2013). *Experimental evaluation of an adiabatic quantum system for combinatorial optimization*. In: Proceedings of CF-13, ACM International Conference on Computing Frontiers, 23:1-23:11.

Mei, X., H. Lee, K. Diao, M. Huang, B. Lin, C. Liu, Z. Xie, Y. Ma, P. Robson, M. Chung, A. Bernheim, V. Mani, C. Calcagno, K. Li, S. Li, H. Shan, J. Lv, T. Zhao, J. Xia, Q. Long, S. Steinberger, A. Jacobi, T. Deyer, M. Luksza, F. Liu, B. P. Little, Z. A. Fayad and Y. Yang, (2020). *Artificial intelligence–enabled rapid diagnosis of patients with COVID-19*, Nat. Med., doi: 10.1101/2020.04.12.20062661.

Mikut, R. (2008). Data Mining in der Medizin und Medizintechnik (Vol. 22). *KIT Scientific Publishing.*

Mishra, V. (2014). Beginning Apache Cassandra Development. Apress.

Mohıeldın, T. A. G. (2022). *Büyük Veri Analitiğinin Dijital Pazarlama Stratejileri Üzerindeki Rolü.* (Yüksek Lisans Tezi). İstanbul Ticaret Üniversitesi Dış Ticaret Enstitüsü Küresel Pazarlama Ve Marka Yönetimi Ana Bilim Dalı.

Morita, S. ve Nishimori, H. (2011). *Mathematical Foundation of Quantum Annealing*. https://arxiv.org/abs/0806.1859v1.

Mukherjee, S.ve Shaw, R. (2016). Big Data- Concepts, Applications, Challenges and Future Scope. *İnternational Journal of Advanced Research in Computer and Communication Engineering*, 5(2), 66–74.

Mybb (2010). *PHP nedir?* http://www.mybbturkiye.com/Thread-PHP-Nedir-ve-Neden-PHP.

Naguleswaran, S., White, L. ve Fuss, I. (2005). *Quantum search in stochastic planning*. Proceedings of the Sixteenth International Conference on Automated Planning and Scheduling, 34–45.

Narayanan, A. ve Menneer, T. (2000). Quantum artificial neural network architectures and components. *Information Sciences*, 128, 231–255.

Nasteski, V. (2017). *An overview of the supervised machine learning methods*. Horizons. b, 4, 51-62.

Neeraj, N. (2015). *Mastering Apache Cassandra*. Packt Publishing Ltd.

Neugebauer, L. M. ve Zanko, I. (2021). *New Digital World, New Digital Terms*. In Lead Community Fundraising, 39-43. Springer, Cham.

Nielsen, M. A. ve Chuang, I. L. (2000). *Quantum information and quantum computation*. Cambridge: Cambridge University Press, 2(8), 23.

Nielsen, M. A., & Chuang, I. (2002). *Quantum computation and quantum information.*

Nishimori, H. (2001). *Statistical Physics of Spin Glasses and Information Processing: An Introduction*. Oxford University Press.

Olson, D.L., Delen, D. (2008). *Memory-Based Reasoning Methods. In: Advanced Data Mining Techniques*. Springer, Berlin, Heidelberg.

Omran, B. A. (2016). *Application of Data Mining and Big Data Analytics in the Construction Industry*. The Ohio State University.

Otsubo, Y., Inoue, J. I., Nagata, K. ve Okada, M. (2014). *Code-division multiple-access multiuser demodulator by using quantum fluctuations*. Physical Review E. 90(012126).

Otsubo, Y., Inoue, J. I., Nagata, K. ve Okada, M. (2012). Effect of quantum fluctuation in errorcorrecting codes. *Physical Review E*, 86(051138).

Özçakır, C.Ö. ve Çamurcu, A.Y. (2007). Birliktelik kuralı yöntemi için bir veri madenciligi yazılımı tasarımı ve uygulaması. *İstanbul Ticaret Üniversitesi Fen Bilimleri Dergisi*, 6(12), 21-37.

Özdamar, K. (2004). *Paket Programlar İle _statistiksel Veri Analizi -2 (Çok Degiskenli Analizler)*. Eskisehir: Kaan Kitapevi.

Özkan, Y. (2008). *Veri Madenciligi Yöntemleri*. İstanbul: Papatya Yayıncılık Egitim.

Pandya, A., Kostakos, P., Mehmood, H., Cortes, M., Gilman, E., Oussalah, M.ve Pirttikangas, S. (2019). *Privacy preserving sentiment analysis on multiple edge data streams with Apache NiFi*. In 2019 European Intelligence and Security Informatics Conference (EISIC) (pp. 130-133). IEEE.

Panella, M. ve Martinelli, G. (2011). Neural networks with quantum architecture and quantum learning. *International Journal of Circuit Theory and Applications*, 39, 61–77.

Pei, J. (2012). *Jiawei Han, Micheline* Kamber. Data mining: concepts and techniques.

Pierson, L. (2021). *Data science for dummies*. John Wiley ve Sons.

Plotnikova, V., Dumas, M. ve Milani, F. (2020). Adaptations of data mining methodologies: a systematic literature review. *PeerJ Computer Science*, 6, e267.

Podgorelec, V., Kokol, P., Stiglic, B. ve Rozman, I. (2002). Decision trees: an overview and their use in medicine. *Journal of medical systems*, 26(5), 445-463.

Pokorný, J. (2017). *Big Data Storage and Management: Challenges and Opportunities*. Hřebíček J., Denzer R., Schimak G., Pitner T. (Eds) Environmental Software Systems. Computer Science for Environmental

Protection. ISESS 2017. *IFIP Advances in Information and Communication Technology*, 507, 28–38.

Rahman M. ve Geiger, D. (2016). *Quantum clustering and gaussian mixtures*. arXiv:1612.09199v1 (stat.ML).

Rawat, B. ve Purnama, S. (2021*). MySQL Database Management System (DBMS) On FTP Site LAPAN Bandung*. International Journal of Cyber and IT Service Management, 1(2), 173-179.

Rebentrost, P., Mohseni, M. ve Lloyd, S. (2014). Quantum support vector machine for big data classification. *Physical review letters*, 113(13), 130503.

Reese, G. (2003). *MySQL Pocket Reference*. O'Reilly Publishing.

Reinsel, D., Gantz, J.ve Rydning, J. (2018*). The Digitization of the World - From Edge to Core. International Data Corporation (IDC)*. Framingham, USA: https://www.seagate.com/files/www-content/our-story/trends/files/idc-seagate-dataage-whitepaper.pdf.

Reitmaier, T. (2015). *Aktives Lernen für Klassifikationsprobleme unter der Nutzung von Strukturinformationen* (Vol. 5). kassel university press GmbH.

Rigatos, G. G. ve Tzafestas, S. G. (2002). *Parallelization of a fuzzy control algorithm using quantum computation*. IEEE Trans. Fuzzy Syst, 10(4),451–460.

Rønnow, T.F.,Wang, Z., Job, J., Boixo, S., Isakov, S.V.,Wecker, D.,Martinis, J.M., Lidar, D.A., Troyer, M. (2014). *Defining and detecting quantum speedup*. arXiv:1401.2910.

Sahin, M., Atav, U. ve Tomak, M. (2005). Quantum genetic algorithm method in self-consistent electronic structure calculations of a quantum dot with many electrons. *International Journal of Modern Physics C*, 16(9),1379–1393.

Salloum, S., Dautov, R., Chen, X., Peng, P. X.ve Huang, J. Z. (2016). Big data analytics on Apache Spark. *International Journal of Data Science and Analytics*, 1(3), 145-164.

Santoro, G. E., Martonak, R., Tosatti, E. ve Car, R. (2002). *Theory of quantum annealing of an Ising spin glass*. Science, 295(5564), 2427–2430.

Sassi, I., Anter, S. ve Bekkhoucha, A. (2020). A new improved baum-welch algorithm for unsupervised learning for continuous-time hmm using spark. *International Journal of Intelligent Engineering and Systems*, 13(1), 214-226.

Schelter, S.ve Owen, S. (2012*). Collaborative filtering with apache mahout*. Proc. of ACM RecSys challenge.

Schuld, M., Sinayskiy, I. ve Petruccione, F. (2015). *An introduction to quantum machine learning*. Contemporary Physics, 56(2), 172-185.

Schuld, M., Sinayskiy, I. ve Petruccione, F. (2015). *An introduction to quantum machine learning*. Contemporary Physics, 56(2), 172–185.

Schuld, M., Sinayskiy, I. ve Petruccione, F. (2014). *Simulating a perceptron on a quantum computer*.. ArXiv:1412.3635.

Schuld, M. ve Killoran, N. (2019). *Quantum machine learning in feature Hilbert spaces*. Physical review letters, 122(4), 040504.

Schuld, M. ve Petruccione, F. (2018). *Supervised learning with quantum computers* (Vol. 17). Berlin: Springer.

Servedio, R.A., Gortler, S.J. (2004). *Equivalences and separations between quantum and classical learnability*. *SIAM J. Comput*. 33(5), 1067–1092.

Shanmuganathan, S. (2016). *Artificial neural network modelling: An introduction. In Artificial neural network modelling* (pp. 1-14). Springer, Cham.

Shor, P. W. (1999). *Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer*. *SIAM review*, 41(2), 303-332.

Small, M. (2019). *Big Data Analytics – Security and Compliance Challenges in 2019.*

Smart, W. D. ve Kaelbling, L. P. (2002). *Effective reinforcement learning for mobile robots. Proceedings* – IEEE International Conference on Robotics and Automation, 3404–3410.

Smith, A. J. (2002). Applications of the self-organising map to reinforcement learning. *Neural Netw*, 15(8/9), 1107–1124.

Sohail, A. (2021). *Genetic algorithms in the fields of artificial intelligence and data sciences*. Annals of Data Science, 1-12.

Steinmacher, I., Wiese, I. S., Chaves, A. P. ve Gerosa, M. A. (2012). *Newcomers withdrawal in open source software projects: Analysis of Hadoop Common project*. In 2012 Brazilian Symposium on Collaborative Systems, 65-74. IEEE.

Stephens, M. (2008). RSS. *Library Technology Reports*, 42(4), 36-44.

Tailor, A. G. (2003). *SQL For Dummies*. 5th Edition. Wiley Publishing, Inc.

The Apache Software Foundation (2022). *What is Apache Flink? — Architecture*. https://flink.apache.org/flink-architecture.html.

Touesnard, B. (2001). *Active server pages 3.0 versus hypertext preprocessor 4.0*. The University of New Brunswick. Faculty of computer science.

Trent, S., Tatsubori, M., Suzumura, T., Tozawa, A. ve Onodera, T. (2008). *Performance comparison of PHP and JSP as server-side scripting languages*.

In ACM/IFIP/USENIX *International Conference on Distributed Systems Platforms and Open Distributed Processing* (pp. 164-182). Springer, Berlin, Heidelberg.

Twin, A. (2022). What Is Data Mining? https://www.investopedia.com/terms/d/datamining.asp.

Tykheev, D. (2018). *Big Data in Marketing. Saimaa University of Applied Sciences*.

Tzafestas, S. G.ve Rigatos, G. G. (2002). *Fuzzy reinforcement learning control for compliance tasks of robotic manipulators*. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)., 32(1),107–113.

Uzun, Y., *Tıbbi veriler üzerinde makine ögrenme algoritmalar ve bulanık mantık ile kurallar ögrenme,* (Yüksek Lisans Tezi), Selçuk Üniversitesi Fen Bilimleri Enstitüsü, Konya, 1-3,12 (2005).

Ünsal, Ö. (2011). *Mesleki Alan Seçimlerinin Makine Öğrenmesi Algoritması Kullanılarak Belirlenmesi*. (Yüksek Lisans Tezi). Gazi Üniversitesi Bilişim Enstitüsü.

Vengerov, D., Bambos, N.ve Berenji, H. *A fuzzy reinforcement learning approach to power control in wireless transmitters IEEE Transactions on Systems*, Man, and Cybernetics, Part B (Cybernetics). 2005, 35(4),768–778.

Verhoef, P. C., Kooge, E.ve Walk, N. (2016). *Creating Value with Big Data Analytics: Making Smarter Marketing Decisions* (First). New York, NY, USA: Routledge.

Vinci, W., Markström, K., Boixo, S., Roy, A., Spedalieri, F. M., Warburton, P. A. ve Severini, S. Hearing the Shape of the Ising Model with a Programmable Superconducting-Flux Annealer. *Scientific Reports*. 2014, 4(5703).

Vohra, D. (2016). *Apache HBase Primer* (Vol. 151). Apress.

Vora, M. N. (2011). *Hadoop-HBase for large-scale data*. In Proceedings of 2011 International Conference on Computer Science and Network Technology (Vol. 1, 601-605. IEEE.

Walunj, S. G. ve Sadafale, K. (2013). *An online recommendation system for e-commerce based on apache mahout framework*. In Proceedings of the 2013 annual conference on Computers and people research, 153-158.

Whiteson, S. ve Stone, P. (2006). Evolutionary function approximation for reinforcement learning. *Journal of Machine Learning Research*, 7, 877–917.

Wille, R., Van Meter, R. ve Naveh, Y. (2019). *IBM's Qiskit tool chain: Working with and developing for real quantum computers*. In 2019 Design, Automation ve Test in Europe Conference ve Exhibition (DATE), 1234-1240. IEEE.

Williams, H. E. ve Lane, D. (2004). *Web Database Applications with PHP and MySQL: Building Effective Database-Driven Web Sites*. O'Reilly Media, Inc..

Wittek, P. (2014). *Quantum machine learning: what quantum computing means to data mining*. Academic Press.

Xin, T., Wang, B. X., Li, K. R., Kong, X. Y., Wei, S. J., Wang, T., ... ve Long, G. L. (2018). Nuclear magnetic resonance for quantum computing: Techniques and recent achievements. *Chinese Physics B*, 27(2), 020308.

Xu, X., Jiang, X., Ma, C., Du, P., Li, X., Lv, S., Yu, L., Ni, Q., Chen, Y., Su, J., Lang, G., Li, Y., Zhao, H., Liu, J., Xu, K., Ruan, L, Sheng, J., Qiu, Y., Wu, W., Liang, T. ve Li, L. (2020). A deep learning system to screen novel coronavirus disease 2019 pneumonia, *Engineering*, 6(10), 1122–1129, doi: 10.1016/j.eng.2020.04.010

Ylijoki, O.ve Porras, J. (2016). Perspectives to Definition of Big Data: A Mapping Study and Discussion. *Journal of Innovation Management* - JIM, 4(1), 69–91.

Zhang, Y. ve Ni, Q. (2020). Recent advances in quantum machine learning. *Quantum Engineering*, 2(1), e34.

Zhou, D., He, J. ve Gu, Q. (2021). *Provably efficient reinforcement learning for discounted mdps with feature mapping*. In International Conference on Machine Learning (pp. 12793-12802). PMLR.

Pepper, A., Tischler, N., & Pryde, G. J. (2019). *Experimental realization of a quantum autoencoder: The compression of qutrits via machine learning*. Physical Review Letters, 122, Article 060501.

Khoshaman, A., Vinci, W., Denis, B., Andriyash, E., Sadeghi, H., & Amin, M. H. (2018). *Quantum variational autoencoder. Quantum Science and Technology*, 4, Article 014001.

Romero, J., Olson, J. P., & Aspuru-Guzik, A. (2017). *Quantum autoencoders for efficient compression of quantum data. Quantum Science and Technology*, 2, Article 045001.

Alvarez-Rodriguez, U., Sanz, M., Lamata, L., & Solano, E. (2018). *Quantum artificial life in an ibm quantum computer*. Scientific Reports, 8, 1–9.

Nawaz, S. J., Sharma, S. K., Wyne, S., Patwary, M. N., & Asaduzzaman, M. (2019). *Quantum machine learning for 6 g communication networks: State-of-the-art and vision for the future*. IEEE Access, 7, 46317–46350.

Lamata, L. (2020). *Quantum machine learning and quantum biomimetics: A perspective. Machine Learning: Science and Technology*, 1, Article 033002.

Sheng, Y.-B., & Zhou, L. (2017). Distributed secure quantum machine learning. *Science Bulletin*, 62, 1025–1029.

Wallnöfer, J., Melnikov, A. A., Dür, W., & Briegel, H. J. (2020). *Machine learning for long-distance quantum communication. PRX Quantum*, 1, Article 010301.

Von Lilienfeld, O. A. (2018). *Quantum machine learning in chemical compound space*. Angewandte Chemie International Edition, 57, 4164–4169.

Rieffel, E. G., Venturelli, D., O'Gorman, B., Do, M. B., Prystay, E. M., & Smelyanskiy, V. N. (2015). *A case study in programming a quantum annealer for hard operational planning problems. Quantum Information Processing*, 14, 1–36.

Amin, M. H., Andriyash, E., Rolfe, J., Kulchytskyy, B., & Melko, R. (2018). *Quantum boltzmann machine*. Physical Review X, 8, Article 021050.

Von Lilienfeld, O. A. (2018). *Quantum machine learning in chemical compound space*. Angewandte Chemie International Edition, 57, 4164–4169.

McArdle, S., Endo, S., Aspuru-Guzik, A., Benjamin, S. C., & Yuan, X. (2020). *Quantum computational chemistry. Reviews of Modern Physics, 92*, Article 015003.

Beer, K., Bondarenko, D., Farrelly, T., Osborne, T. J., Salzmann, R., Scheiermann, D., et al. (2020). Training deep quantum neural networks. *Nature Communications*, 11, 1–6.

Dunjko, V., & Briegel, H. J. (2018). *Machine learning & artificial intelligence in the quantum domain: a review of recent progress. Reports on Progress in Physics*, 81, Article 074001

Levine, Y., Sharir, O., Cohen, N., & Shashua, A. (2019). *Quantum entanglement in deep learning architectures*. Physical Review Letters, 122, Article 065301.

Killoran, N., Bromley, T. R., Arrazola, J. M., Schuld, M., Quesada, N., & Lloyd, S. (2019). *Continuous-variable quantum neural networks. Physical Review Research*, 1, Article 033063.

Mari, A., Bromley, T. R., Izaac, J., Schuld, M., & Killoran, N. (2020). *Transfer learning in hybrid classical-quantum neural networks*. Quantum, 4, 340.

Gao, Z., Ma, C., Song, D., & Liu, Y. (2017). *Deep quantum inspired neural network with application to aircraft fuel system fault diagnosis*. Neurocomputing, 238, 13–23.

Pomarico, D., Fanizzi, A., Amoroso, N., Bellotti, R., Biafora, A., Bove, S., et al. (2021). *A proposal of quantum-inspired machine learning for medical purposes: An application case*. Mathematics, 9, 410.

Huang, H.-Y., Broughton, M., Mohseni, M., Babbush, R., Boixo, S., Neven, H., et al. (2021). *Power of data in quantum machine learning*. Nature Communications, 12, 1–9.

Schuld, M., Sinayskiy, I., & Petruccione, F. (2016). *Prediction by linear regression on a quantum computer*. Physical Review A, 94, Article 022342.

Rebentrost, P., Mohseni, M., & Lloyd, S. (2014). *Quantum support vector machine for big data classification*. Physical Review Letters, 113, Article 130503.

Ezhov, A. A., & Ventura, D. (2000). *Quantum neural networks. In Future directions for intelligent systems and information sciences*, 213–235. Springer

da Silva, A. J., Ludermir, T. B., & de Oliveira, W. R. (2016). *Quantum perceptron over a field and neural network architecture selection in a quantum computer*. Neural Networks, 76, 55–64.

Schuld, M., Sinayskiy, I., & Petruccione, F. (2015b). *Simulating a perceptron on a quantum computer*. Physics Letters. A, 379, 660–663.

Zhou, R. (2010). *Quantum competitive neural network. International Journal of Theoeretical Physics*, 49, 110–119.

Zhong, Y., & Yuan, C. (2012). *Quantum competition network model based on quantum entanglement*. Journal of Computers, 7, 2312–2317.

Zidan, M., Abdel-Aty, A.-H., El-shafei, M., Feraig, M., Al-Sbou, Y., Eleuch, H., et al. (2019). *Quantum classification algorithm based on competitive learning neural network and entanglement measure*. Applied Sciences, 9, 1277

Abbas, A., Sutter, D., Zoufal, C., Lucchi, A., Figalli, A., & Woerner, S. (2021). *The power of quantum neural networks*. Nature Computational Science, 1, 403–409.

Chen, S. Y.-C., & Yoo, S. (2021). *Federated quantum machine learning*. Entropy, 23, 460.

Dang, Y., Jiang, N., Hu, H., Ji, Z., & Zhang, W. (2018). *Image classification based on quantum k-nearest-neighbor algorithm*. Quantum Information Processing, 17, 239.

Adhikary, S., Dangwal, S., & Bhowmik, D. (2020). *Supervised learning with a quantum classifier using multi-level systems.* Quantum Information Processing, 19, 89.

Mitarai, K., Negoro, M., Kitagawa, M., & Fujii, K. (2018). *Quantum circuit learning*. Physical Review A, 98, Article 032309.

Henderson, M., Shakya, S., Pradhan, S., & Cook, T. (2020). *Quanvolutional neural networks: powering image recognition with quantum circuits*. Quantum Machine Intelligence, 2, 1–9.

Bausch, J. (2020). *Recurrent quantum neural networks. In Advances in neural information processing systems*, 33.

Benedetti, M., Garcia-Pintos, D., Perdomo, O., Leyton-Ortega, V., Nam, Y., & PerdomoOrtiz, A. (2019). *A generative modeling approach for benchmarking and training shallow quantum circuits*. Npj Quantum Information, 5, 1–9.

Zhao, Z., Pozas-Kerstjens, A., Rebentrost, P., & Wittek, P. (2019). *Bayesian deep learning on a quantum computer*. Quantum Machine Intelligence, 1, 41–51.

Chrisley, R. (1995). *Quantum learning*. In New directions in cognitive science: Proceedings of the international symposium, Vol. 4. Saariselka: Citeseer.

Aïmeur, E., Brassard, G., & Gambs, S. (2006). *Machine learning in a quantum world*. In Conference of the Canadian society for computational studies of intelligence (pp. 431–442). Springer.

Lloyd, S., Mohseni, M., & Rebentrost, P. (2013*). Quantum algorithms for supervised and unsupervised machine learning*. arXiv preprint arXiv:1307.0411.

Dunjko, V., Taylor, J. M., & Briegel, H. J. (2016*). Quantum-enhanced machine learning*. Physical Review Letters, 117, Article 130501.

Schuld, M. (2018). *Supervised learning with quantum computers*. Springer.

ÖZGEÇMİŞ

İlk, orta ve lise öğrenimimi Kayseri'de tamamladım. Üniversite hayatımda ise ilk olarak Erciyes Üniversitesi Bilgisayar Programcılığı ardından AÖF İşletme Fakültesinden mezun oldum. 2006 yılında girmiş olduğum KPSS sınavı ile kamuda çalışmaya başladım. 2014 yılında DGS sınavı ile Yalova Üniversitesi Bilgisayar Mühendisliği Bölümüne kayıt yaptırdım. Mühendislik eğitimimi tamamladıktandan sonra, Yalova Üniversitesi Bilgisayar Mühendisliği bölümünde Yüksek Lisans eğitimime başladım ve 2023 yılında mezun oldum. Halen Bursa Uludağ Üniversitesi Hukuk Fakültesinde çalışmaktayım.

TEZDEN TÜRETİLEN YAYIN VE ESERLER

Şahin S., Harman G. (2023, Temmuz). Quantum Machine Learning Application On Covid'19. *Uluslararası İzmir Fen, Mühendislik ve Uygulamalı Bilimler Kongresi*, İzmir.