

Çanakkale Onsekiz Mart Üniversitesi, Mühendislik Fakültesi,
Bilgisayar Mühendisliği Akademik Dönem 2022-2023
Ders: BLM-4014 Yapay Sinir Ağları/Bahar Dönemi
ARA SINAV SORU VE CEVAP KAĞIDI
Dersi Veren Öğretim Elemanı: Dr. Öğretim Üyesi Ulya Bayram

Öğrenci Adı Soyadı: ÖZNUR KAN
Öğrenci No: 180401046

14 Nisan 2023

Açıklamalar:

- Vizeyi çözüp, üzerinde aynı sorular, sizin cevaplar ve sonuçlar olan versiyonunu bu formatta PDF olarak, Teams üzerinden açtığım assignment kısmına yüklemeniz gerekiyor. Bu bahsi geçen PDF'i oluşturmak için LaTeX kullandıysanız, tex dosyasının da yer aldığı Github linkini de ödevin en başına (aşağı url olarak) eklerseniz bonus 5 Puan! (Tavsiye: Overleaf)
- Çözümlerde ya da çözümlerin kontrolünü yapmada internetten faydalanmak, ChatGPT gibi servisleri kullanmak serbest. Fakat, herkesin çözümü kendi emeğinden oluşmak zorunda. Çözümlerinizi, cevaplarınızı aşağıda belirttiğim tarih ve saate kadar kimseyle paylaşmayınız.
- Kopyayı önlemek için Github repository'lerinizin hiçbirini **14 Nisan 2023, saat 15:00'a kadar halka açık (public) yapmayınız!** (Assignment son yükleme saati 13:00 ama internet bağlantısı sorunları olabilir diye en fazla ekstra 2 saat daha vaktiniz var. **Fakat 13:00 - 15:00 arası yüklemelerden -5 puan!**)
- Ek puan almak için sağlayacağımız tüm Github repository'lerini **en geç 15 Nisan 2023 15:00'da halka açık (public) yapmış olun linklerden puan alabilmek için!**
- **14 Nisan 2023, saat 15:00'dan sonra gönderilen vizeler değerlendirilmeye alınmayacak, vize notu olarak 0 (sıfır) verilecektir!** Son anda internet bağlantısı gibi sebeplerden sıfır almayı önlemek için assignment kısmından ara ara çözümlerinizi yükleyebilirsiniz yedekleme için. Verilen son tarih/saatte (14 Nisan 2023, saat 15:00) sistemdeki en son yüklü PDF geçerli olacak.
- Çözümlerin ve kodların size ait ve özgün olup olmadığını kontrol eden bir algoritma kullanılacaktır. Kopya çektiği belirlenen vizeler otomatikman 0 (sıfır) alacaktır. Bu nedenle çözümlerinizi ve kodlarınızı yukarıda sağladığım gün ve saatlere kadar kimseyle paylaşmayınız.
- Bu vizeden alınabilecek en yüksek not 100'dür. Toplam aldığımız puan 100'ü geçerse, aldığımız not 100'e sabitlenecektir.
- LaTeX kullanarak PDF oluşturanlar öz geçmişlerine LaTeX bildiklerini de eklemeyi unutmasınlar :)
- Bu vizedeki soruların çözümleri ve tex dosyası için istediğiniz kadar sayıda Github repository'si oluşturabilirsiniz. Sadece yukarıda belirttiğim tarihlerde (14 Nisan 2023 saat 15:00 ile 15 Nisan 2023 saat 15:00 arasında) public yapmayı/halka açmayı ve Github profilinizi de öz geçmişinize eklemeyi unutmayın :)
- Bilgisayarınıza hiçbir program kurmadan, Overleaf, Google Colab, Kaggle gibi ortamlar üzerinden bu vizeyi tamamlamanız mümkün. İyi çalışmalar!!

(Ekstra 5 puan) https://github.com/oznurkan/Project_artificial_neural_network_tex/upload/main

1 (Toplam 10 Puan) Hopfield Networks:

1.1 (2 Puan) ChatGPT'ye size Hopfield Network'ler ile ilgili detaylı bilgi vermesini isteyin. Sorduğunuz soruyu, ChatGPT'nin çıkarttığı cevabı aşağı koyun. Bu cevapta anlamadığınız terimleri de ChatGPT'ye sorup, sorduğunuz soruları ve ChatGPT'nin verdiği cevapları da sırayla aşağı koyun.

SORU (CHATGPT) = Hopfield Network'ler ile ilgili detaylı bilgi verebilir misin?

CEVAP = Hopfield ağları, sinir ağı modellerinden biridir ve en eski ve popüler olanlarından biridir. Hopfield ağları, birbiriyle etkileşim halinde olan bir dizi nöron (veya hücre) koleksiyonudur. Her nöron, girdi olarak diğer nöronların çıkışı alır ve çıkışı da diğer nöronların girdisine eklenir. Böylece, her nöron, diğer nöronların aktivasyonu ile belirlenir.

Hopfield ağları, özellikle bellek veya tanıma sorunları için kullanılır. Ağ, bir bellek çiftini depolayabilir ve ardından bir girdi verildiğinde, depolanmış belleklerden birini hatırlamaya çalışır. Hopfield ağları, asenkron dinamikleri ile çalışır, yani tüm nöronlar eşzamanlı olarak değil, sırayla güncellenir.

Bir Hopfield ağı, genellikle iki durumlu (on-off) nöronlar kullanır. Her nöron, iki durumlu bir cihaz gibi davranır ve bu durumlar, genellikle 1 ve -1 olarak tanımlanan birçok farklı şekilde belirlenebilir. Ayrıca ağ, bir enerji fonksiyonu kullanır, böylece ağın durumunu bir enerji seviyesiyle ifade edebilirsiniz.

Hopfield ağları, birçok farklı uygulamada kullanılabilir. Örneğin, bunlar veri sıkıştırma, modelleme ve tanıma, ham verilerin temizlenmesi ve hatırlama sorunları gibi alanlarda kullanılabilir.

Hopfield ağlarının bir dezavantajı, büyük ölçekli problemlerde çalışırken verimliliklerinin azalmasıdır. Bu nedenle, daha büyük ölçekli problemlerde, özellikle daha karmaşık sinir ağı modelleri kullanmak daha verimlidir.

1.2 (8 Puan) ChatGPT'nin açıklamalarından ne anladığınızı buraya kendi kelimelerinizle özetleyin. Örneğin "Hopfield Network nedir? Ne işe yarar? Neden bilmemiz gerekir bunu? Gerçek hayatta kullanılıyor mu?" gibi. Anlamadığınız kısımlar varsa ve ChatGPT'nin açıklamaları yeterli gelmezse internet dahil farklı kaynaklardan araştırıp, bilgilerin doğruluğunu sorgulamakta serbestsiniz. Konuyu doğru anlayıp anlamamanız puana etki edecektir. Kullandığınız kaynakları da belirtin!

SORU (CHATGPT) = Hopfield Network nedir?

CEVAP (CHATGPT and me) = Hopfield ağı, bir tür sinir ağı modelidir. Birbirine bağlı bir dizi nöron koleksiyonundan oluşur ve bellek veya tanıma sorunları için kullanılır. Ağ, bir bellek çiftini depolayabilir ve ardından bir girdi verildiğinde, depolanmış belleklerden birini hatırlamaya çalışır. Hopfield ağları, asenkron dinamikleri ile çalışır. Hopfield ağları biyolojik sinir ağlarının matematiksel modellemesi için de kullanılabilir.

SORU (CHATGPT) = Ne işe yarar?

CEVAP = Hopfield ağları, bellek veya tanıma sorunlarının çözülmesinde kullanılır. Ağ, bir bellek çiftini depolayabilir ve ardından bir girdi verildiğinde, depolanmış belleklerden birini hatırlamaya çalışır. Özellikle veri sıkıştırma, modelleme ve tanıma, ham verilerin temizlenmesi ve hatırlama sorunları gibi alanlarda kullanılır. Hopfield ağları, ayrıca biyolojik sinir ağlarının matematiksel modellemesi için de kullanılabilir. Bunun yanı sıra, sınırlı bellekli ve düşük güç tüketimli çip tasarımlarında da sık kullanılır.

SORU (CHATGPT) = Neden bilmemiz gerekir bunu?

CEVAP = Hopfield ağlarının ne olduğunu ve nasıl çalıştığını bilmek, yapay zeka ve sinir ağı modelleri hakkında bilgi sahibi olmak isteyen kişiler için önemlidir. Bu ağlar, bellek veya tanıma sorunlarının çözümü için kullanılan güçlü bir araçtır ve birçok alanda kullanım potansiyeli vardır. Ayrıca, Hopfield ağları biyolojik sinir ağlarının matematiksel modellemesi için kullanılabileceğinden, nörobilim

araştırmalarında da kullanılmaktadır. Dolayısıyla, Hopfield ağlarını bilmek, yapay zeka, sinir ağı modelleri, bellek veya tanıma sorunları, cihaz tasarımı ve nörobilim gibi konularda ilgi duyanlar için faydalı olabilir.

SORU (CHATGPT) = Gerçek hayatta kullanılıyor mu?

CEVAP = Evet, Hopfield ağları gerçek hayatta birçok alanda kullanılmaktadır. Örneğin, Hopfield ağları, veri sıkıştırma, modelleme ve tanıma, ham verilerin temizlenmesi ve hatırlama sorunları gibi alanlarda kullanılır. Hopfield ağları ayrıca, sınırlı bellekli ve düşük güç tüketimli çip tasarımlarında da sık kullanılır. Bunun yanı sıra, Hopfield ağları biyolojik sinir ağlarının matematiksel modellenmesi için kullanılabileceğinden, nörobilim araştırmalarında da kullanılmaktadır. Dolayısıyla, Hopfield ağları, gerçek hayatta yaygın bir şekilde kullanılan bir yapay sinir ağı modelidir.

2 (Toplam 10 Puan) Boltzman Machine:

2.1 (2 Puan) ChatGPT'ye size Boltzman Machine ilgili detaylı bilgi vermesini isteyin. Sorduğunuz soruyu, ChatGPT'nin çıkarttığı cevabı aşağı koyun. Bu cevapta anlamadığınız terimleri de ChatGPT'ye sorup, sorduğunuz soruları ve ChatGPT'nin verdiği cevapları da sırayla aşağı koyun.

SORU (CHATGPT) = Boltzman Machine ilgili detaylı bilgi verebilir misin?

CEVAP = Boltzmann Machine (BM), yapay sinir ağı modellerinden biridir ve özellikle büyük ve karmaşık veri kümelerini analiz etmek için kullanılır. Adını istatistik mekaniğin önemli isimlerinden olan Ludwig Boltzmann'dan almıştır. Boltzmann Machine, özellikle desen tanıma, doğal dil işleme, nesne tanıma, sınıflandırma ve öneri sistemleri gibi alanlarda kullanılır.

BM, iki tip nöronun birbiriyle etkileşimini içerir: Görünür ve Gizli nöronlar. Görünür nöronlar, veri kümesinde doğrudan gözlemlenebilen nöronlardır. Gizli nöronlar ise, sadece diğer gizli nöronlar ve görünür nöronlarla etkileşimde bulunarak veri üzerindeki gizli yapıları belirleyebilirler.

BM, verilerin enerji seviyelerinin azaltıldığı bir süreç olan eğitim sürecinde öğrenir. BM, bir matematiksel enerji fonksiyonu kullanarak çalışır. Bu fonksiyon, bir veri örneği için bir enerji değeri hesaplar ve bu enerji değeri, verinin ne kadar "iyi" veya "kötü" olduğunu temsil eder.

BM, eğitim sırasında bir öğrenme algoritması kullanarak veri kümesini işler ve ağırlıklarını, enerji fonksiyonunu minimize edecek şekilde ayarlar. Bir kez eğitildikten sonra, BM, belirli bir girdiye yanıt vermek üzere kullanılabilir.

BM, özellikle büyük ve karmaşık veri kümeleri üzerinde çalışırken, veri içindeki desenleri keşfetmek için kullanılır. Örneğin, bir BM, bir görüntüdeki nesneleri tanımak, bir doğal dil belgesindeki kelimeleri tanımak veya bir müşterinin alışveriş alışkanlıklarını öğrenmek için kullanılabilir.

BM, derin öğrenme alanında önemli bir konudur ve birçok farklı türü vardır. Bunlar arasında Sınır İtici Boltzmann Makinesi (RBM), Tersinir Boltzmann Makinesi (DBN) ve Etkileşimli Boltzmann Makinesi (IBM) bulunur.

Sonuç olarak, Boltzmann Machine, yapay zeka ve makine öğrenimi gibi teknolojilerde önemli bir rol oynar ve büyük veri kümelerinin analizi için çok faydalıdır. Bu teknolojinin anlaşılabilir şekilde açıklanması, insanların bu alanda çalışan araştırmacıların geliştirdiği yenilikleri anlamasına yardımcı olur.

2.2 (8 Puan) ChatGPT'nin açıklamalarından ne anladığınızı buraya kendi kelimelerinizle özetleyin. Örneğin "Boltzman Machine nedir? Ne işe yarar? Neden bilmemiz gerekir bunu? Gerçek hayatta kullanılıyor mu?" gibi. Anlamadığınız kısımlar varsa ve ChatGPT'nin açıklamaları yeterli gelmezse internet dahil farklı kaynaklardan araştırıp, bilgilerin doğruluğunu sorgulamakta serbestsiniz. Konuyu doğru anlayıp anlamamanız puana etki edecektir. Kullandığınız kaynakları da belirtin!

SORU (CHATGPT) = Boltzman Machine nedir?

CEVAP = Boltzmann Machine (BM) iki tip nöronun birbiriyle etkileşimini içeren yapay sinir ağı modellerinden biridir ve özellikle büyük ve karmaşık veri kümelerini analiz etmek için kullanılır. Bu iki tip Görünür ve Gizli nöronlardır. Görünür nöronlar, veri kümesinde doğrudan gözlemlenebilen nöronlardır. Gizli nöronlar ise, sadece diğer gizli nöronlar ve görünür nöronlarla etkileşimde bulunarak veri üzerindeki gizli yapıları belirleyebilirler.

BM, bir matematiksel enerji fonksiyonu kullanarak çalışır. Bu fonksiyon, bir veri örneği için bir enerji değeri hesaplar ve bu enerji değeri, verinin ne kadar "iyi" veya "kötü" olduğunu temsil eder.

BM, eğitim sırasında bir öğrenme algoritması kullanarak veri kümesini işler ve ağırlıklarını, enerji fonksiyonunu minimize edecek şekilde ayarlar. Bir kez eğitildikten sonra, BM, belirli bir girdiye yanıt vermek üzere kullanılabilir.

BM, özellikle büyük ve karmaşık veri kümeleri üzerinde çalışırken, veri içindeki desenleri keşfetmek için kullanılır. Örneğin, bir BM, bir görüntüdeki nesneleri tanımak, bir doğal dil belgesindeki kelimeleri tanımak veya bir müşterinin alışveriş alışkanlıklarını öğrenmek için kullanılabilir.

SORU (CHATGPT) = Ne işe yarar?

CEVAP = Boltzmann Machine (BM), büyük ve karmaşık veri kümelerini analiz etmek için kullanılır. BM, veri içindeki desenleri keşfetmek için kullanılır. Bir görüntüdeki nesneleri tanımak, bir doğal dil belgesindeki kelimeleri tanımak veya bir müşterinin alışveriş alışkanlıklarını öğrenmek için kullanılabilir.

BM, özellikle derin öğrenme alanında önemli bir konudur. BM'nin farklı türleri, örneğin Sınır İtici Boltzmann Makinesi (RBM), Tersinir Boltzmann Makinesi (DBN) ve Etkileşimli Boltzmann Makinesi (IBM) gibi, farklı problemler için özelleştirilebilir ve uyarlanabilir.

BM, özellikle veri analizi, doğal dil işleme, görüntü işleme ve öneri sistemleri gibi alanlarda kullanılır. BM, bu alanlarda birçok uygulama bulabilir, örneğin bir müşterinin alışveriş alışkanlıklarına dayalı olarak öneriler sunmak, bir görüntüdeki nesneleri tanımak veya bir metin belgesindeki anahtar kelimeleri belirlemek gibi.

BM, özellikle büyük veri kümeleri için kullanışlıdır ve geleneksel makine öğrenme teknikleri bu verileri işlemekte yetersiz kalabilir. BM, veri içindeki yapıları keşfetmek için kullanılabildiği için, yeni fikirlerin ve ilişkilerin keşfi için de kullanılabilir.

SORU (CHATGPT) = Neden bilmemiz gerekir bunu?

CEVAP = Boltzmann Machine (BM), derin öğrenme alanında önemli bir konudur ve büyük veri kümelerini analiz etmek için kullanılır. Günümüzde, birçok farklı endüstri, özellikle veri analizi, doğal dil işleme, görüntü işleme ve öneri sistemleri gibi alanlarda BM teknolojilerini kullanmaktadır.

Bu nedenle, BM teknolojileri hakkında bilgi sahibi olmak, günümüzdeki birçok endüstrinin veri analizi süreçlerinde faydalıdır. BM, büyük veri kümeleri üzerinde çalışırken, veri içindeki desenleri keşfetmek için kullanılır ve bu da farklı problemleri çözmek için kullanılabilir.

Ayrıca, BM teknolojilerinin işleyişinin anlaşılması, derin öğrenme alanında çalışan araştırmacılar ve geliştiriciler için de önemlidir. BM, özellikle büyük ve karmaşık veri kümeleri üzerinde çalışırken, diğer makine öğrenme tekniklerinden farklı bir yaklaşım sunar ve bu nedenle, öğrenme süreci ve veri analizi süreci için farklı bir bakış açısı sağlar.

Sonuç olarak, BM hakkında bilgi sahibi olmak, günümüzdeki veri analizi süreçlerinde daha verimli ve etkili bir şekilde çalışmak için faydalıdır ve derin öğrenme alanında çalışanlar için de önemlidir.

SORU (CHATGPT) = Gerçek hayatta kullanılıyor mu?

CEVAP = Evet, Boltzmann Machine (BM) gerçek hayatta birçok alanda kullanılıyor. Özellikle, büyük veri kümelerinin analizi için kullanılan BM teknolojileri, birçok endüstride yaygın olarak kullanılıyor.

BM, özellikle veri analizi, doğal dil işleme, görüntü işleme ve öneri sistemleri gibi alanlarda kullanılmaktadır. Örneğin, bir online mağaza, müşteri davranışlarını analiz etmek ve müşterilere öneriler sunmak için BM teknolojilerini kullanabilir. Benzer şekilde, bir arama motoru, arama sonuçlarını belirlemek için BM kullanabilir ve bir görüntü tanıma sistemleri, görüntülerdeki nesneleri tanımak için BM teknolojilerinden faydalanabilir.

BM ayrıca, derin öğrenme alanında önemli bir konudur ve birçok farklı uygulama ve teknolojiyi de içeren geniş bir araştırma alanıdır. Örneğin, Sınırlı Boltzmann Makineleri (RBM), Tersinir Boltzmann Makineleri (DBN) ve Etkileşimli Boltzmann Makineleri (IBM) gibi farklı BM türleri, farklı problemler için özelleştirilebilir ve uyarlanabilir.

Sonuç olarak, BM teknolojileri, gerçek hayatta birçok farklı alanda kullanılmaktadır ve büyük veri kümelerinin analizi için son derece önemlidir. Ayrıca, derin öğrenme alanında araştırma yapmak için de önemli bir konudur.

3 (Toplam 10 Puan) Markov Assumption ve Markov Chain:

3.1 (2 Puan) ChatGPT'ye size Markov assumption ve Markov Chain ilgili detaylı bilgi vermesini isteyin. Sorduğunuz soruyu, ChatGPT'nin çıkarttığı cevabı aşağı koyun. Bu cevapta anlamadığınız terimleri de ChatGPT'ye sorup, sorduğunuz soruları ve ChatGPT'nin verdiği cevapları da sırayla aşağı koyun.

SORU (CHATGPT) = Markov assumption ve Markov Chain ilgili detaylı bilgi verebilir misin?

CEVAP = Markov süreci ya da Markov Zinciri, bir sistemin geçmiş durumları yerine sadece şu anki durumuna bağlı olduğu varsayımıyla modellenir. Bu varsayımı "Markov varsayımı" denir. Bu varsayım, bir sonraki durumun yalnızca şu anki duruma bağlı olduğunu varsayar ve önceki durumların ve geçmiş durumların bilinmesine gerek yoktur.

Markov zinciri, bir dizi adımdan oluşan bir süreçtir. Her adımda, sistem belli bir durumda bulunur ve bir sonraki durum, geçmiş durumlardan bağımsız olarak, olasılıksal bir şekilde belirlenir. Bu olasılıklar, bir geçiş matrisi olarak adlandırılır ve her satır, bir durumu temsil eder ve her sütun, bir sonraki durumu temsil eder.

Markov zincirleri birçok alanda kullanılır, özellikle de doğal dil işleme, finans, biyoloji ve fizikte. Örneğin, dil modelleri, her kelimenin önceki kelimeye bağlı olduğu bir Markov zinciri olarak modellenir.

Markov zincirleri, özellikle kuyruklu Markov zincirleri, geleceği tahmin etmek, sınıflandırma yapmak, dönemik sistemleri modellerlemek ve veri sıkıştırma kullanılır.

3.2 (8 Puan) ChatGPT'nin açıklamalarından ne anladığınızı buraya kendi kelimelerinizle özetleyin. Örneğin "Markov assumption ve Markov Chain nedir? Ne işe yarar? Neden bilmemiz gerekir bunu? Gerçek hayatta kullanılıyor mu?" gibi. Anlamadığınız kısımlar varsa ve ChatGPT'nin açıklamaları yeterli gelmezse internet dahil farklı kaynaklardan araştırıp, bilgilerin doğruluğunu sorgulamakta serbestsiniz. Konuyu doğru anlayıp anlamamanız puana etki edecektir. Kullandığınız kaynakları da belirtin!

SORU (CHATGPT) = Markov assumption ve Markov Chain nedir?

CEVAP = Markov varsayımı, bir sistemin gelecekteki durumu, sadece o anki durumuna bağlı olduğunu varsayar ve önceki durumların bilinmesine gerek yoktur. Markov zinciri ise, bu varsayımı dayanan ve bir dizi adımdan oluşan bir süreçtir. Her adımda, sistem belirli bir durumda bulunur ve bir sonraki durum, geçmiş durumlardan bağımsız olarak, olasılıksal bir şekilde belirlenir.

Markov zincirleri, birçok alanda kullanılır. Özellikle, doğal dil işleme, finans, biyoloji ve fizikte kullanılır. Dil modelleri, örneğin, her kelimenin önceki kelimeye bağlı olduğu bir Markov zinciri olarak

modellenebilir. Bu zincirler, geleceği tahmin etmek, sınıflandırma yapmak, dönemik sistemleri modellerlemek ve veri sıkıştırma kullanılır.

SORU (CHATGPT) = Ne işe yarar?

CEVAP = Markov zincirleri, sistemin geçmiş durumları yerine sadece şu anki durumuna bağlı olduğu varsayımıyla modellenir. Bu varsayım, bir sonraki durumun yalnızca şu anki duruma bağlı olduğunu varsayar ve önceki durumların ve geçmiş durumların bilinmesine gerek yoktur. Bu nedenle, Markov zincirleri geleceği tahmin etmek, sınıflandırma yapmak, dönemik sistemleri modellerlemek ve veri sıkıştırma kullanılır.

Örneğin, doğal dil işlemede, bir dil modeli, her kelimenin önceki kelimeye bağlı olduğu bir Markov zinciri olarak modellenebilir. Bu model, bir kelimenin ardından gelen kelimenin olasılığını tahmin etmek için kullanılabilir. Finansal piyasalarda, hisse senedi fiyatları veya döviz kurları için Markov zincirleri kullanılarak gelecekteki fiyatların tahmini yapılabilir. Biyolojide, Markov zincirleri, proteinlerin yapılarının modellenmesi ve moleküler dinamik simülasyonları gibi alanlarda kullanılır.

Markov zincirleri, ayrıca rastgele yürüyüşler ve istatistiksel mekanik gibi fiziksel sistemlerin modellenmesinde de kullanılır.

SORU (CHATGPT) = Neden bilmemiz gerekir bunu?

CEVAP = Markov zincirleri ve Markov varsayımı, birçok alanda kullanılan önemli matematiksel kavramlardır. Bu nedenle, bu konuda bilgi sahibi olmak, birçok alanda çalışan ve bu süreçleri kullanmayı düşünen insanlar için önemlidir. Örneğin, finansal piyasalarda çalışanlar, hisse senedi fiyatlarının tahmini için Markov zincirlerini kullanabilirler. Biyoloji alanında çalışanlar, proteinlerin yapılarının modellenmesi için Markov zincirlerini kullanabilirler. Doğal dil işlemede çalışanlar, dil modelleri oluşturmak için Markov zincirlerini kullanabilirler.

Markov zincirleri ayrıca makine öğrenmesi ve yapay zeka alanlarında da sıkça kullanılır. Bu nedenle, bu konuda bilgi sahibi olmak, gelecekte bu alanda çalışmak isteyenler için de önemlidir.

SORU (CHATGPT) = Gerçek hayatta kullanılıyor mu?

CEVAP = Evet, Markov zincirleri gerçek hayatta birçok alanda kullanılır. Örneğin:

Doğal Dil İşleme: Doğal dil işlemede, dil modelleri oluşturmak için Markov zincirleri kullanılır. Bu modeller, bir kelimenin ardından gelen kelimenin olasılığını tahmin etmek için kullanılır. Örneğin, bir kelime işlemcinin otomatik tamamlama özelliğinde kullanılabilir.

Finans: Finansal piyasalarda, hisse senedi fiyatları veya döviz kurları için Markov zincirleri kullanılarak gelecekteki fiyatların tahmini yapılabilir. Bu tahminler, yatırımcıların piyasayı anlamalarına ve karar vermelerine yardımcı olabilir.

Biyoloji: Biyolojide, proteinlerin yapılarının modellenmesi için Markov zincirleri kullanılır. Bu modeller, proteinlerin yapısını tahmin etmek için kullanılabilir.

Fizik: Markov zincirleri, rastgele yürüyüşler ve istatistiksel mekanik gibi fiziksel sistemlerin modellenmesinde de kullanılır. Bu modeller, moleküler dinamik simülasyonları gibi alanlarda kullanılabilir.

Makine Öğrenmesi: Markov zincirleri, makine öğrenmesi ve yapay zeka alanlarında da sıkça kullanılır. Örneğin, konuşma tanıma ve metin sınıflandırma gibi uygulamalarda kullanılabilir.

Bu örnekler gösteriyor ki, Markov zincirleri gerçek hayatta birçok alanda kullanılan önemli bir matematiksel kavramdır.

4 (Toplam 20 Puan) Feed Forward:

- Forward propagation için, input olarak şu X matrisini verin (tensöre çevirmeyi unutmayın):

$$X = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \text{ Satırlar veriler (sample'lar), kolonlar öznitelikler (feature'lar).}$$

- Bir adet hidden layer olsun ve içinde tanh aktivasyon fonksiyonu olsun
- Hidden layer'da 50 nöron olsun
- Bir adet output layer olsun, tek nöronu olsun ve içinde sigmoid aktivasyon fonksiyonu olsun

Tanh fonksiyonu:

$$f(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)}$$

Sigmoid fonksiyonu:

$$f(x) = \frac{1}{1 + \exp(-x)}$$

Pytorch kütüphanesi ile, ama kütüphanenin hazır aktivasyon fonksiyonlarını kullanmadan, formülünü verdiğim iki aktivasyon fonksiyonunun kodunu ikinci haftada yaptığımız gibi kendiniz yazarak bu yapay sinir ağını oluşturun ve aşağıdaki üç soruya cevap verin.

4.1 (10 Puan) Yukarıdaki yapay sinir ağını çalıştırmadan önce pytorch için Seed değerini 1 olarak set edin, kodu aşağıdaki kod bloğuna ve altına da sonucu yapıştırın:

```
import torch
import torch.nn as nn

torch.manual_seed(1)

class MyModel(nn.Module):

    def __init__(self):
        super(MyModel, self).__init__()
        self.hidden_layer = nn.Linear(3, 50)
        self.output_layer = nn.Linear(50, 1)

    def tanh(self, x):
        return (torch.exp(x) - torch.exp(-x)) / (torch.exp(x) + torch.exp(-x))

    def sigmoid(self, x):
        return 1 / (1 + torch.exp(-x))

    def forward(self, x):
        out = self.hidden_layer(x)
        out = self.tanh(out)
        out = self.output_layer(out)
        out = self.sigmoid(out)
        return out

X = torch.tensor([[1, 2, 3], [4, 5, 6]], dtype=torch.float32)
model = MyModel()
output = model(X)
print(output.data)
```

tensor([[0.4892], [0.5566]])

4.2 (5 Puan) Yukarıdaki yapay sinir ağını çalıştırmadan önce Seed değerini öğrenci numaranız olarak değiştirip, kodu aşağıdaki kod bloğuna ve altına da sonucu yapıştırın:

```
import torch
import torch.nn as nn

torch.manual_seed(180401046)

class MyModel(nn.Module):
```



```

def __init__(self):
    super(MyModel, self).__init__()
    self.hidden_layer = nn.Linear(3, 50)
    self.output_layer = nn.Linear(50, 1)

def tanh(self, x):
    return (torch.exp(x) - torch.exp(-x)) / (torch.exp(x) + torch.exp(-x))

def sigmoid(self, x):
    return 1 / (1 + torch.exp(-x))

def forward(self, x):
    out = self.hidden_layer(x)
    out = self.tanh(out)
    out = self.output_layer(out)
    out = self.sigmoid(out)
    return out

X = torch.tensor([[1, 2, 3], [4, 5, 6]], dtype=torch.float32)
model = MyModel()
output = model(X)
print(output.data)

```

tensor([[0.3094], [0.2490]])

4.3 (5 Puan) Kodlarınızın ve sonuçlarınızın olduğu jupyter notebook'un Github repository'sindeki linkini aşağıdaki url kısmının içine yapıştırın. İlk sayfada belirttiğim gün ve saate kadar halka açık (public) olmasın:

https://github.com/oznurkan/Project_Artificial_neural_networks_1

5 (Toplam 40 Puan) Multilayer Perceptron (MLP):

Bu bölümdeki sorularda benim vize ile beraber paylaştığım Prensesi İyileştir (Cure The Princess) Veri Seti parçaları kullanılacak. Hikaye şöyle (soruyu çözmek için hikaye kısmını okumak zorunda değilsiniz):

“Bir zamanlar, çok uzaklarda bir ülkede, ağır bir hastalığa yakalanmış bir prenses yaşamış. Ülkenin kralı ve kraliçesi onu iyileştirmek için ellerinden gelen her şeyi yapmışlar, ancak denedikleri hiçbir çare işe yaramamış.

Yerel bir grup köylü, herhangi bir hastalığı iyileştirmek için gücü olduğu söylenen bir dizi sihirli malzemeden bahsederek kral ve kraliçeye yaklaşmış. Ancak, köylüler kral ile kraliçeyi, bu malzemelerin etkilerinin patlayıcı olabileceği ve son zamanlarda yaşanan kuraklıklar nedeniyle bu malzemelerden sadece birkaçının herhangi bir zamanda bulunabileceği konusunda uyarılmışlar. Ayrıca, sadece deneyimli bir simyacı bu özelliklere sahip patlayıcı ve az bulunan malzemelerin belirli bir kombinasyonunun prensesi iyileştireceğini belirleyebilecekmiş.

Kral ve kraliçe kızlarını kurtarmak için umutsuzlar, bu yüzden ülkedeki en iyi simyacıyı bulmak için yola çıkmışlar. Dağları tepeleri aşmışlar ve nihayet “Yapay Sinir Ağları Uzmanı” olarak bilinen yeni bir sihirli sanatın ustası olarak ün yapmış bir simyacı bulmuşlar.

Simyacı önce köylülerin iddialarını ve her bir malzemenin alınan miktarlarını, ayrıca iyileşmeye yol açıp açmadığını incelemiş. Simyacı biliyormuş ki bu prensesi iyileştirmek için tek bir şans varmış ve bunu doğru yapmak zorundaymış. (Original source: <https://www.kaggle.com/datasets/unmoved/cure-the-princess>)

(Buradan itibaren ChatGPT ve Dr. Ulya Bayram’a ait hikayenin devamı)

Simyacı, büyülu bileşenlerin farklı kombinasyonlarını analiz etmek ve denemek için günler harcamış. Sonunda birkaç denemenin ardından prensesi iyileştirecek çeşitli karışım kombinasyonları bulmuş ve bunları bir veri setinde toplamış. Daha sonra bu veri setini eğitim, validasyon ve test setleri olarak üç parçaya ayırmış ve bunun üzerinde bir yapay sinir ağı eğiterek kendi yöntemi ile prensesi iyileştirme ihtimalini hesaplamış ve ikna olunca kral ve kraliçeye haber vermiş. Heyecanlı ve umutlu olan kral ve kraliçe, simyacının prensese hazırladığı ilacı vermesine izin vermiş ve ilaç işe yararmış ve prenses hastalığından kurtulmuş.

Kral ve kraliçe, kızlarının hayatını kurtardığı için simyacıya krallıkta kalması ve çalışmalarına devam etmesi için büyük bir araştırma bütçesi ve çok sayıda GPU'su olan bir server vermiş. İyileşen prenses de kendisini iyileştiren yöntemleri öğrenmeye merak salıp, krallıktaki üniversitenin bilgisayar mühendisliği bölümüne girmiş ve mezun olur olmaz da simyacının yanında, onun araştırma grubunda çalışmaya başlamış. Uzun yıllar birlikte krallıktaki insanlara, hayvanlara ve doğaya faydalı olacak yazılımlar geliştirmişler, ve simyacı emekli olduğunda prenses hem araştırma grubunun hem de krallığın lideri olarak hayatına devam etmiş.

Prens, kendisini iyileştiren veri setini de, gelecekte onların izinden gidecek bilgisayar mühendisi prensler ve prensesler başkalarına faydalı olabilecek yapay sinir ağları oluşturmayı öğrensinler diye halka açmış ve sınavlarda kullanılmasını salık vermiş.”

İki hidden layer'lı bir Multilayer Perceptron (MLP) oluşturun beşinci ve altıncı haftalarda yaptığımız gibi. Hazır aktivasyon fonksiyonlarını kullanmak serbest. İlk hidden layer'da 100, ikinci hidden layer'da 50 nöron olsun. Hidden layer'larda ReLU, output layer'da sigmoid aktivasyonu olsun.

Output layer'da kaç nöron olacağını veri setinden bakıp bulacaksınız. Elbette bu veriye uygun Cross Entropy loss yöntemini uygulayacaksınız. Optimizasyon için Stochastic Gradient Descent yeterli. Epoch sayınızı ve learning rate'i validasyon seti üzerinde denemeler yaparak (loss'lara overfit var mı diye bakarak) kendiniz belirleyeceksiniz. Batch size'ı 16 seçebilirsiniz.

5.1 (10 Puan) Bu MLP'nin pytorch ile yazılmış class'ının kodunu aşağı kod bloğuna yapıştırın:

```
class MLP(nn.Module):
    def __init__(self, input_dim, output_neuron):
        super(MLP, self).__init__()
        self.hidden_layer_1 = nn.Linear(input_dim, 100)
        self.hidden_layer_2 = nn.Linear(100, 50)
        self.output_layer = nn.Linear(50, output_neuron)
        self.relu = nn.ReLU()
        self.sigmoid = nn.Sigmoid()

    def forward(self, x):
        x = x.view(x.size(0), -1)
        hidden_1 = self.relu(self.hidden_layer_1(x))
        hidden_2 = self.relu(self.hidden_layer_2(hidden_1))
        output = self.sigmoid(self.output_layer(hidden_2))
        return output
```

5.2 (10 Puan) SEED=öğrenci numaranız set ettikten sonra altıncı haftada yazdığımız gibi training batch'lerinden eğitim loss'ları, validation batch'lerinden validasyon loss değerlerini hesaplayan kodu aşağıdaki kod bloğuna yapıştırın ve çıkan figürü de alta ekleyin.

```
def train(model, optimizer, criterion, train_loader, validation_loader, num_epochs,
          train_loss, validation_loss):
```

```

best_loss = float('inf')
for epoch in range(num_epochs):
    running_loss = 0.0
    running_corrects = 0
    total_train = 0

    model.train()
    for i, (inputs, labels) in enumerate(train_loader):
        inputs = inputs.to(device).float()
        labels = labels.to(device)

        optimizer.zero_grad()
        outputs = model(inputs)
        loss = criterion(outputs, labels)
        loss.backward()
        optimizer.step()

        train_loss += loss.item()

        _, preds = torch.max(outputs, 1)
        running_corrects += torch.sum(preds == labels.data)
        total_train += labels.size(0)

    train_loss = train_loss / len(train_loader)
    train_acc = 100. * running_corrects / total_train
    train_losses.append(train_loss)
    train_accs.append(train_acc)

    #validation loss ve accuracy hesapla
    validation_loss, val_acc = evaluate(model, criterion, validation_loader,
                                       validation_loss, train_loss)

    validation_losses.append(validation_loss)
    val_accs.append(val_acc)

    print(f"Epoch {epoch+1}/{num_epochs} - Train Loss: {train_loss:.4f} - Train
          Acc: {train_acc:.2f} - Val Loss: {
          validation_loss:.4f} - Val Acc: {
          val_acc:.2f}")

```

```

def evaluate(model, criterion, data_loader, validation_loss, train_loss):
    running_loss = 0.0
    running_corrects = 0
    total_val = 0
    model.eval()

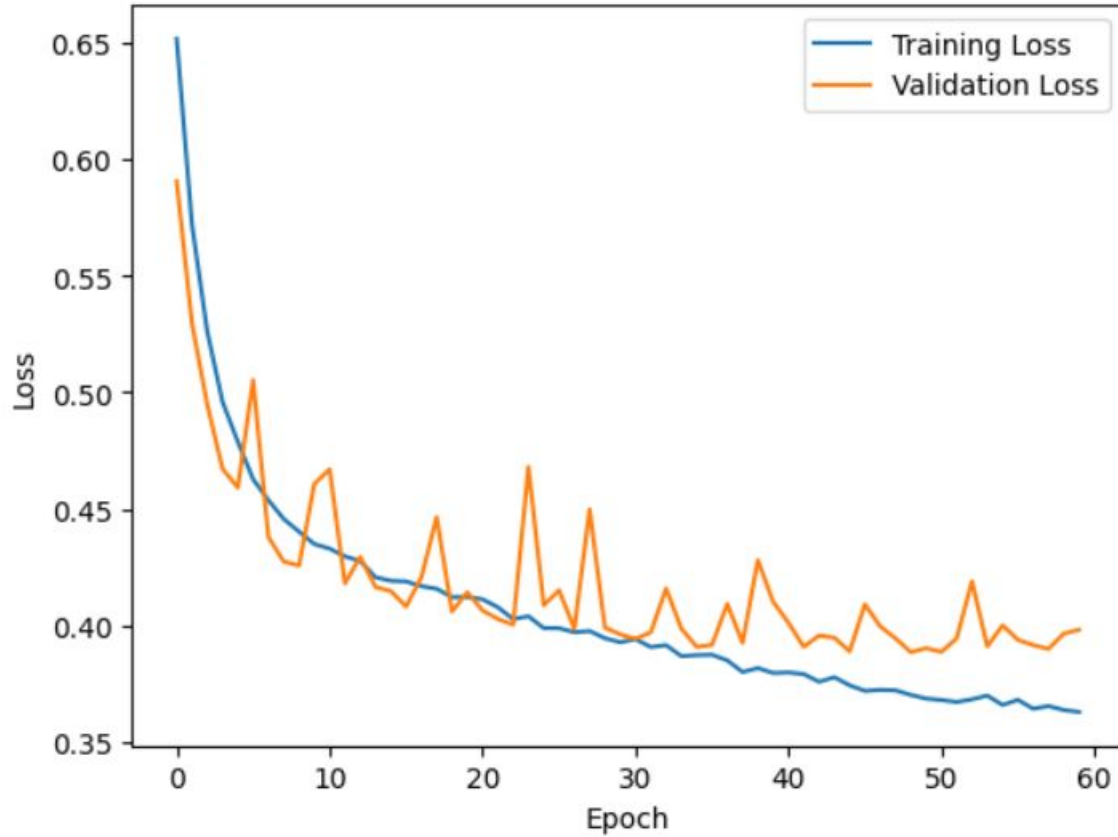
    with torch.no_grad():
        for inputs, labels in validation_loader:
            inputs = inputs.to(device).float()
            labels = labels.to(device)

            outputs = model(inputs)
            loss = criterion(outputs.squeeze(), labels.long())

            _, preds = torch.max(outputs, 1)
            validation_loss += loss.item()
            total_val += labels.size(0)
            running_corrects += torch.sum(preds == labels.data)

```

```
validation_loss = validation_loss / len(validation_loader)
val_acc = 100. * running_corrects / total_val
return validation_loss, val_acc
```



Şekil 1: Buraya açıklama yazın

5.3 (10 Puan)

SEED=öğrenci numaranız set ettikten sonra altıncı haftada ödev olarak verdiğim gibi earlystopping'deki en iyi modeli kullanarak, Prensesi İyileştir test setinden accuracy, F1, precision ve recall değerlerini hesaplayan kodu yazın ve sonucu da aşağı yapıştırın. %80'den fazla başarı bekliyorum test setinden. Daha düşükse başarı oranınız, nerede hata yaptığınızı bulmaya çalışın. %90'dan fazla başarı almak mümkün (ben denedim).

```
checkpoint = torch.load('/content/drive/MyDrive/dataset/checkpoint.pt')
model_2 = MLP(input_dim, output_neuron)
model_2.load_state_dict(checkpoint)
model_2.to(device)
```

```
validation_score = validation_loss
if best_validation_loss is None:
    patience_counter = 0
    best_validation_loss = validation_score
    torch.save(model.state_dict(), "checkpoint.pt")
elif best_validation_loss < validation_score:
    patience_counter += 1
    print("Earlystopping Patience Counter", patience_counter)
    if patience_counter == patience:
        break
else:
    best_validation_loss = validation_score
```

```
torch.save(model.state_dict(), "checkpoint.pt")
patience_counter = 0
```

```
evaluate_metrics(model_2, criterion, test_loader)
```

Test Accuracy: 0.9236, Prec Accuracy: 0.9237, F1 score: 0.9236, Recall score: 0.9236

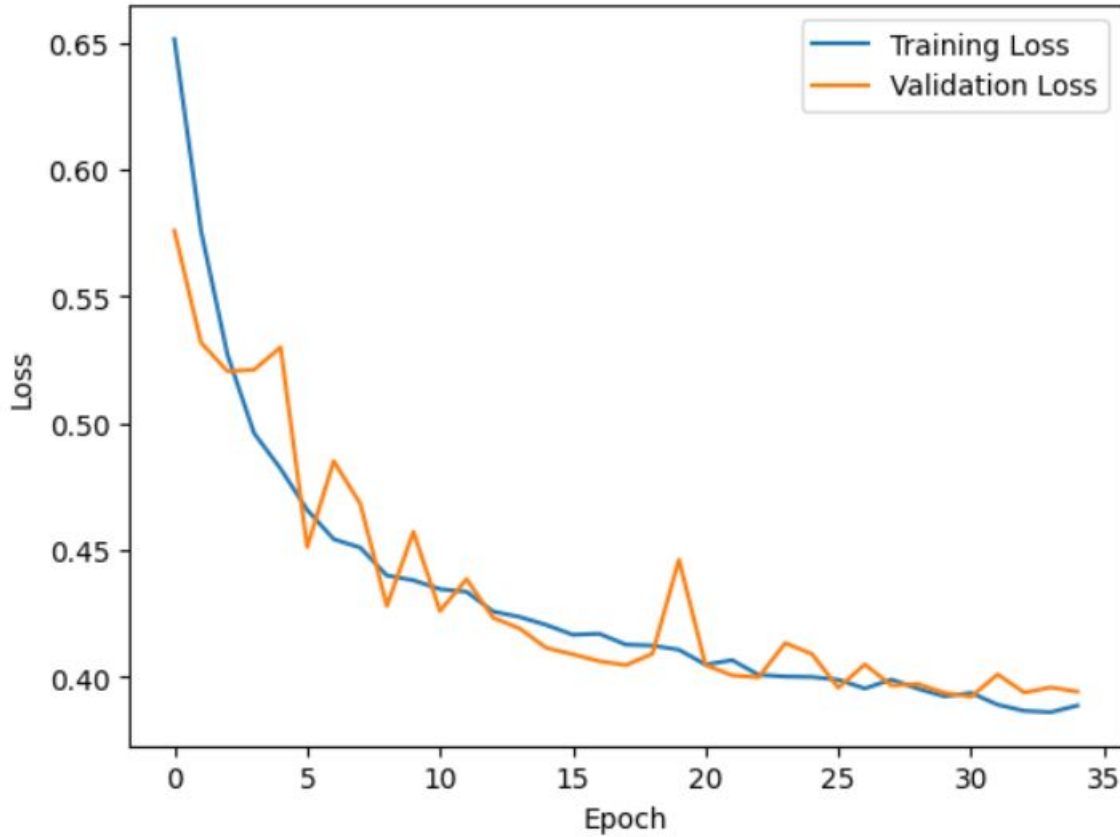
5.4 (5 Puan) Tüm kodların CPU’da çalışması ne kadar sürüyor hesaplayın. Sonra to device yöntemini kullanarak modeli ve verileri GPU’ya atıp kodu bir de böyle çalıştırın ve ne kadar sürdüğünü hesaplayın. Süreleri aşağıdaki tabloya koyun. GPU için Google Colab ya da Kaggle’ı kullanabilirsiniz, iki ortam da her hafta saatlerce GPU hakkı veriyor.

Tablo 1: PyTorch kütüphanesinin device sınıfı kullanılarak çalışma zamanı ölçülmüştür.

Ortam	Süre (saniye)
CPU	0:00:08.260194
GPU	0:00:17.042746

5.5 (3 Puan) Modelin eğitim setine overfit etmesi için elinizden geldiği kadar kodu gereken şekilde değiştirin, validasyon loss’unun açıkça yükselmeye başladığı, training ve validation loss’ları içeren figürü aşağı koyun ve overfit için yaptığınız değişiklikleri aşağı yazın. Overfit, tam bir çanak gibi olmalı ve yükselmeli. Ona göre parametrelerle oynayın.

Cevap= Overfit için epoch değerlerinde ve earlystopping de en iyi sonucu oluşturmak için patience üzerinde de oynamalar yapılmıştır. Gerekli denemeler sonucu en iyi sonuç oluşturulmaya çalışılmıştır. Regülerizasyon çalışmaları yapılmaya çalışılmıştır. Ortaya aşağıdaki görsel çıkmıştır.



Şekil 2: Overfit çalışmaları sonucu

5.6 (2 Puan) Beşinci soruya ait tüm kodların ve cevapların olduğu jupyter notebook'un Github linkini aşağıdaki url'e koyun.

https://github.com/oznurkan/Project_artificial_neural_network_2

6 (Toplam 10 Puan)

Bir önceki sorudaki Prensisi İyileştir problemindeki yapay sinir ağına seçtiğiniz herhangi iki farklı regülarizasyon yöntemi ekleyin ve aşağıdaki soruları cevaplayın.

6.1 (2 puan) Kodlarda regülarizasyon eklediğiniz kısımları aşağı koyun:

```
kod_buraya = None
if kod_buraya:
    devam_ise_buraya = 0

print(devam_ise_buraya)
```

6.2 (2 puan) Test setinden yeni accuracy, F1, precision ve recall değerlerini hesaplayıp aşağı koyun:

Sonuçlar buraya.

6.3 (5 puan) Regülarizasyon yöntemi seçimlerinizin sebeplerini ve sonuçlara etkisini yorumlayın:

Yorumlar buraya.

6.4 (1 puan) Sonucun github linkini aşağıya koyun:

www.benimgithublinkim2.com