Bir model tasarlarken; [1]

- Dropout değeri
- Katman sayısı
- Nöron sayısı

değişkenleri veriseti ve probleme göre belirlenir. Bunlara hiperparametre denir.

Learning Rate ve Momentum:
Deep learning parametrelerin güncellenmesi backpropagation ile yapılmaktadır. Backpropagation içinde bu işi bu güncelleme işine “chain rule” denir. Weigth güncellenirken kullanılan learning rate parametresi sabit ya da adım adım artan bir değer olarak belirlenebilir.
Öğrenme hızını belirlemek için en uygun çözüm; öğrenme hızını yüksek tutup, giderek azaltmaktır.
Genellikle varsayılan değer olarak 0.01 kullanılır, belli bir epoch’tan sonra 0.001’e düşürülür.
Momentum beta katsayısı ise 0.9 olarak alınır. Uygun parametre aralığı 0.8-0.99’dur.

Optimizer:
Ağırlık katsayılarının güncellenmesi için kullanılır.

Optimizasyon Algoritmasının Seçimi:
Doğrusal olmayan problemlerin çözümünde optimum değerler bulmak için optimizasyon yöntemleri kullanılır. Bunlar;

- Stochastic Gradient Descent (SGD)
- RMSprop
- AdaGrad
- AdaDelta
- Adam
- Adamax

Bu algoritmalar içinde başarım ve hız farklılıkları vardır.
İki tür gradient hesaplama yöntemi vardır. Sayısal ve analitik.
Sayısal gradient yaklaşık sonuç verir, yavaş çalışır fakat uygulanması kolaydır.
Analitik gradient hızlı ve tam sonuç verir fakat hataya eğilimlidir. Analitik gradientle hesaplama yapılıp sonuçlar sayısal gradient ile kontrol edilir.

Varsayılan optimizasyon algoritması olarak stochastic gradient descent (SGD) kullanılır. Diğer yöntemlere göre daha yavaş çalışır. Resim tanıma problemlerinde başarısızdır.
Adaptive algoritmalar öğrenme hızını (learning rate) kendisi öğrenir ve dinamiktir.

AdaGrad seyrek parametreler için büyük güncellemeler yaparken sık parametreler için daha küçük güncellemeler yapar. Bu nedenle NLP ve resim tanıma gibi seyrek

veriler için uygundur. Her parametrenin kendi öğrenme hızı vardır. Algoritmanın özelliklerine bağlı olarak öğrenme hızı giderek azalır ve bir noktada sistem öğrenmeyi bırakır.

Epoch Sayısı:
Model eğitilirken tüm veri aynı anda eğitime katılmaz. Belli sayıda parçalar halinde eğitilir. İlk parça eğitilir, modelin başarımı test edilir, başarıma göre backpropagation ile ağırlıklar güncellenir ve yeni eğitim kümesine geçilir. Bu işlem her bir eğitim adımında tekrarlanarak model için en uygun ağırlık değerleri hesaplanmaya çalışılır. Bu eğitim adımlarının her birine epoch denir.
Epoch sayısı arttıkça başarım oranı yükselir. Başarım belli bir noktadan sonra çok küçük birimlerle artış gösterir ve bu durumda eğitim sonlandırılabilir.

Verbose: [3]
Eğitim sırasında her bir epoch'dan sonra elde edilen sonuçların gösterilmesini sağlar. Sıfır olarak girilirse gösterilmez. Sonuçları incelemek için bir değeri girilmelidir.

Weight Başlangıç Değerlerinin Belirlenmesi:
Weight değerleri model başlatılırken hepsi 0 olacak şekilde 0.5-0.9 arasında uniform dağılıma sahip olacak şekilde tanımlanabilir. Ağırlıkların nasıl belirlendiği modelin öğrenme hızını ve başarımını etkiler.

Sequential Modeli: Sıralı katmanlarda oluşturulan bir Keras modelidir.

Aktivasyon Fonksiyonu:
Modele non-linearite katar. Gizli katmanlarda lineer fonksiyon matris çarpımı yapılır ve nöronların ağırlığı hesaplanır. Çıktı non-linear bir değere dönüştürülür. Çok katmanlı yapay sinir ağlarında doğrusal olmayan dönüşüm işlemleri için kullanılır. Gizli katmanlarda geri türev alınabilmesi için (gradient descent) gizli katmanların çıktısı bazı aktivasyon fonksiyonları ile normalize edilmektedir.

Aktivasyon fonksiyonu nedir? Neden gereklidir?
-relu gibi bir aktivasyon fonksiyonu (doğrusallığı bozan) olmaksızın Dense katmanı iki doğrusal işlem (bir iç çarpım ve bir toplama) barındırır.
çıktı = dot(W, input) + b
çıktı = W : ağırlık değeri x input + bias
Bu denklem yalnızca doğrusal dönüşümlerin öğrenmesini sağlar. Yeni katmanlar eklemek ve boyutu artırmak bunun için çözüm değildir.
*Zengin hipotez uzayı elde edebilmek için doğrusallığı bozan aktivasyon fonksiyonlarına ihtiyaç vardır. relu bunlar içinde en popüleridir.

Aktivasyon fonksiyonlarında en çok tercih edilen Relu'dur. Relu'da sigmoid'e göre parametreler daha hızlı bir şekilde öğrenilmektedir. PRelu, Relu'nun kaçırdığı negatif değerleri yakalamaktadır.

Relu fonksiyonu:
İleri beslemeli (feedforward) ağlarda genelde Relu fonksiyonları kullanılmaktadır.
Ağınız çok derinse ve işlem yükü önemli bir problemse ReLU tercih edilebilir.
ReLU'daki gradyanların ölmesi sorununa çözüm olarak Leaky ReLU kullanmaya karar verebilirsiniz.
Ağın kolay ve hızlı yakınsaması ilk kriter olabilir.
ReLU hız bakımından avantajlı olacaktır. Gradyanların ölmesini göze almanız gerek.
Genellikle çıkış değil ara katmanlarda kullanılır.
Gradyanların ölmesi problemine ilk çözüm Leaky ReLU olabilir.
Derin öğrenme modelleri için ReLU ile denemelere başlamanız tavsiye edilebilir.
Çıkış katmanlarında genellikle Softmax kullanılır.

Loss fonksiyonu: [2]
Modelin hata oranını ve başarımını ölçer. Ağ modelinin en son katmanında bulunur.
Hata hesaplama problemini bir optimizasyon problemine dönüştürür.

Loss fonksiyonu amacı:
Negatif en çok olabilirlik değerini minimize etmektir.
En az fark, en çok benzerlik gösteren sınıfı bulmaktır.

En Sık Kullanılan Loss&Aktivasyon Fonksiyonları:
Sigmoid: Sınıflandırma sonucu binary output ise loss hesaplamak için kullanılır(iki sınıflı problemlerin çözümü için). Çok sınıflı problemler için Sigmoid uygun değildir.

Step: Bir eşik değeri alarak ikili bir sınıflandırma çıktısı (0 veya 1) üretir.

Softmax fonksiyonu:
Yapay sinir ağı tarafından üretilen skor değerlerini kullanarak olasılık temelli loss değeri üretir. Softmax sonucunda test girdisinin her bir sınıfa ait benzerliği için olasılık değeri üretilir.
Yapay sinir ağının çıktı olarak verdiği değerler normalize edilmemiş değerlerdir.
Softmax bu değerleri normalize ederek olasılık değerine dönüştürür. "En çok olabilirlik" istatistik fonksiyonunu kullanır.
Normalizasyon işlemiyle olasılıksal bir hale getirdiği için sınıflar arası farkı daha kolay yorumlanabilir hale getirir.

Softmax fonksiyonunun amacı:
Eğitim sonucunda elde edilen weight ve bias parametrelerinin problem çözümü için ne kadar uygun olduğunu hesaplar.
Test verisinin doğru sınıfı için log(en çok olabilirlik) değerini maksimize etmektir.

Dropout (Seyreltme) Değeri:
Tam bağlı (fully connected) katmanlarda belli eşik değerin altındaki düğümlerin seyreltilmesi başarımı artırmaktadır. (Zayıf bilgilerin unutulması öğrenimi arttırır.)
Dropout değeri genellikle 0.5 alınır. Problem ve verisetine göre değişiklik gösterebilir.
Dropout eşik değeri olarak kullanıldığında [0,1] aralığında bir değer alır.
Farklı katmanların farklı dropout değeri olabilir.

Katman Sayısı ve Gizli Katmanlardaki Nöronlar:
Derin öğrenmeyi diğer yapay sinir ağlarından ayıran ve karmaşık problemlerde iyi sonuçlar vermesini sağlayan katman sayısıdır. Katman sayısı arttıkça modelin daha iyi öğrendiği gözlenmiştir.
Katman sayısını belirlemek için net bir yöntem bulunmamaktadır. Andrew Ng bunun için başta tek veya iki katmanlı bir ağ ile başlayarak uygun hiperparametreler bulunması üzerine yoğunlaşılabilir, sonrasında katman sayısı arttırılabilir. Katman sayısı arttıkça modelin yapısını değiştirmek gerekmektedir. Katman sayısı arttıkça backpropagation etkisi ilk katmanlara daha az ulaşmaya başlayacaktır. Böylelikle katmanın artması çok fazla etki etmeyecektir. Bu noktadan sonra diğer hiperparametreler ile modelin başarımı artırılmaya çalışılmalıdır.

Nöron sayısı hafızada tutulan bilgi sayısını gösterir. Nöron sayısının fazla olması bellek ihtiyacı ve hesaplama zamanını arttırır. Nöron sayısının az olması ise yetersiz uyuma (underfitting) sebep olur.
İlk katmanlarda fazla nöron kullanılıp ilerleyen katmanlarda azaltılarak devam etmesi “regularization” (başarım iyileştirme) etkisi yaratır. Bunun için Dropout, sparse-dense-sparse yöntemleri kullanılır.

Evrişimli Sinir Ağı (CNN) Kernel Boyutu:
Evrişimli sinir ağlarında her bir katmanda matris üzerinde işlem yapan kerneller kullanılır. Kernel boyutu ile ne kadar genişlikte verinin birbirini etkileyeceğine karar verilir. Genellikle 3x3, 5x5, 7x7 kerneller kullanılır. Büyük boyutlu kernel kullanımı bilgi kaybına neden olabileceğinden genelde 3x3 küçük boyutlu kernel kullanılır.

Pooling Yöntemi: max, min, mean:
Evrişimsel sinir ağlarında kullanılan kernellerin çıktısı üzerinde “pooling” işlemi yapılmaktadır. Pooling işlemi kernelde bulunan veriler için filtreleme tekniğidir. Kernel içindeki matriste bulunanların en büyüğünü alma “max pooling”, en küçüğünü alma “min pooling” ve ortalamasını alma “mean pooling”dir. En sık tercih edileni “max pooling”dir.

Categorical Cross Entropy:
İki olasılık dağılımı arasındaki mesafeyi ölçer. Birinci olasılık dağılımı ağın çıktısı diğeri ise etiketlerin doğru dağılımıdır. Bu mesafeyi en aza indirerek ağı mümkün olduğunca doğru etiketleri öğrenmesi için eğitilir.

Verilerin Hazırlanması:
- Veriler öncelikle tensöre dönüştürülür.
- Tensörlerin aldığı değer küçük bir aralığa ölçeklendirilir. [-1,1] veya [0,1]
- Nitelikler farklı aralıklarda olduğunda (heterojen) veriler normalize edilir.

Girdi ve çıktılar için tensörler hazır olduğunda ağ eğitime başlayabilir.

Problem Tipi	Son Katman Aktivasyonu	Kayıp Fonksiyonu
İkili sınıflandırma	sigmoid	binary_crossentropy
Tek etiketli çoklu sınıflandırma	softmax	categorical_crossentropy
Çok etiketli çoklu sınıflandırma	sigmoid	binary_crossentropy
Herhangi bir değere bağlanım	-	mse
0-1 aralığında bağlanım	sigmoid	mse veya binary_crossentropy

Düzenlileştirme ve Hiperparametreleri Ayarlama:
- Dropout eklenir
- Farklı mimariler dene: katman eklenir veya çıkartılır
- L1/L2 düzenlileştirme eklenir
- En iyi yapılandırmayı bulana kadar (katmandaki gizli birim sayısı, öğrenme oranı gibi) farklı parametreler denenir
- İsteğe bağlı olarak öznitelik çıkarımı ile yeni nitelikler ekleme veya gereksiz olanlarını çıkartma.

Değerlendirme Protokolü Seçimi:

- Holdout: Bol miktarda veri olduğunda, (genellikle bu kullanılır)
- K-fold çapraz doğrulama: Çok az veri olduğunda,
- Döngüsel K-fold: Az veri ile yüksek hassasiyetle modeli değerlendirmek için.

Ağ Modeli 1: [s.87]

- Veriler N sınıfa sınıflandırmak isteniyorsa, ağda N boyutlu Dense katman olmalıdır.
- Tek etiketli çoklu sınıflandırma problemlerinde ağ softmax aktivasyon fonksiyonu ile sonlanmalı ki N çıktı sınıfına bir olasılık dağılımı hesaplanabilsin.
- Categorical crossentropy kayıp fonksiyonu olması gereken olasılık dağılımı ile ağın çıktısı olan olasılık dağılımı arasındaki mesafeyi en aza indirir.
- Çok sınıflı sınıflandırmada etiketleri kullanmanın iki yolu vardır:
    - Kategorik (bir-elemanı-bir) kodlama ve kayıp fonksiyonu olarak categorical crossentropy kullanmak.
    - Sparse categorical crossentropy kayıp fonksiyonunu kullanmak ve etiketleri tam sayı olarak kodlamak.
- Eğer veriler çok sayıda kategoriye sınıflandırmak isteniyorsa, ağ ara katmanları çok küçük seçilerek bilgi darboğazı oluşturulmamalıdır.

Ağ Modeli 2: [s.95]

- Bağlanım sınıflandırma için ortalama karesel hata (MSE) en sık kullanılan kayıp fonksiyonudur.
- Bağlanım için kullanılan metrikler sınıflandırmada kullanılanlardan farklıdır. Başarım kavramı bağlanıma uymaz. Bağlanım için ortalama mutlak hata (MAE) sıkça kullanılan bir metriktir.
- Verilerin nitelikleri farklı aralıklarda ise ayrı ayrı önişlemden geçirilerek ölçeklenmelidir.
- Çok az veri varken K-fold çapraz doğrulama kullanmak modeli değerlendirmek için güvenilir bir yoldur.
- Çok az veri varken küçük modeller kullanmak aşırı uydurmanın önüne geçer.

Çıkarımlar 1:

- Sinir ağlarını ham veriyle beslemeden önce ham veriyi önişlemden geçirmek gerekmektedir.
- Verilerinizde farklı aralıklarda veriler bulunduğunda bu nitelikler önişlemede ölçeklendirilmelidir.
- Eğitim işleminde sinir ağları bir noktadan sonra aşırı uydurmaya meyillidir ve başarım düşer.
- Çok az veri varken bir veya iki katmanlı küçük ağlar kullanarak aşırı uydurmanın önüne geçilebilir.

- Eğer veriler çok fazla kategoriye ayrılmışsa ve ara katmanlar küçük kullanılmışsa bilgi darboğazı olabilir.
- Bağlanım, sınıflandırmadan farklı kayıp fonksiyonu ve metrikler kullanılır.

Çıkarımlar 2:
Sinir ağlarında aşırı uydurmayı önlemek için:
- Veri sayısını artırmak
- Ağın kapasitesini düşürmek
- Ağırlık düzenlileştirmek
- Dropout uygulamak

Çıkarımlar 3: [Evrişimli sinir ağları s.140]
2000 eğitim 1000 doğrulama ve 1000 test (kedi ve köpek) resmi için;
Her set kendi içinde sınıfların eşit sayıda örneği barındırması nedeniyle dengeli ikili sınıflandırma problemi oluşturur. Bu da sınıflandırma başarımının geçerli bir başarı ölçütüdür.

[1] :
https://medium.com/deep-learning-turkiye/derin-ogrenme-uygulamalarinda-en-sik-kullanilan-hiper-parametreler-ece8e9125c4

[2]:
https://www.linkedin.com/pulse/derin-%25C3%25B6%25C4%259Frenme-uygulamalar%25C4%25B1nda-temel-kavramlar-skor-ve-%25C3%25A7arkac%25C4%25B1/?trackingId=8b%2FOY0FJfC5B1UmOmXrUZg%3D%3D

[3]:
https://medium.com/deep-learning-turkiye/keras-ile-derin-%C3%B6%C4%9Frenme-1-30e5856c6cb9

[4]:
https://dergipark.org.tr/tr/download/article-file/821607

[5]: optimizer, RMSprop
https://towardsdatascience.com/a-look-at-gradient-descent-and-rmsprop-optimizers-f77d483ef08b