

# Derin Öğrenme El Kitabı Super VIP

Afshine AMIDI ve Shervine AMIDI

April 30, 2019

## Contents

### 1 Evrişimli Sinir Ağları

1.1	Genel bakış	1
1.2	Katman tipleri	1
1.3	Hiperparametrelerin filtrelenmesi	2
1.4	Hiperparametreleri ayarlama	2
1.5	Yayın olarak kullanılan aktivasyon fonksiyonları	3
1.6	Nesne algılama	3
1.6.1	Yüz doğrulama ve tanıma	5
1.6.2	Sinirsiz stil transferi (aktarımı)	5
1.6.3	Hesaplama ipuçları kullanan mimariler	5

### 2 Tekrarlayan Yapay Sinir Ağları

2.1	Genel Bakış	7
2.2	Uzun vadeli bağımlılıkların ele alınması	7
2.3	Kelime temsilini öğrenme	8
2.4	Kelimelerin karşılaştırılması	9
2.5	Dil modeli	10
2.6	Makine çevirisisi	10
2.7	Dikkat	11

### 3 Derin Öğrenme püf noktaları ve ipuçları

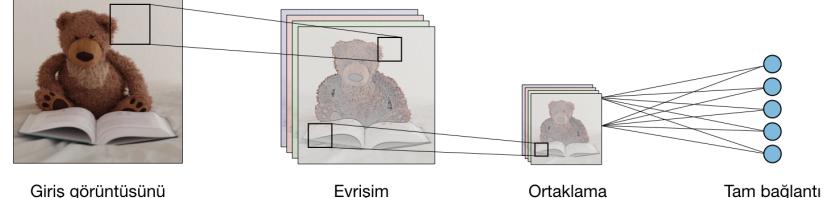
3.1	Veri işleme	11
3.2	Bir sinir ağının eğitilmesi	12
3.3	Parametre ayarlama	12
3.4	Düzenleştirme	13
3.5	İyi uygulamalar	13

## 1 Evrişimli Sinir Ağları

*Ayyüce Kızrak ve Yavuz Kömeçoğlu tarafından çevrilmiştir*

### 1.1 Genel bakış

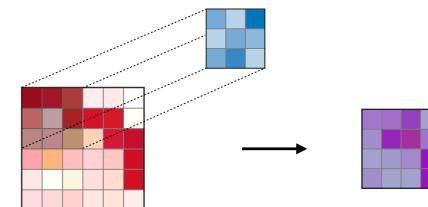
◻ **Geleneksel bir CNN (Evrişimli Sinir Ağı) mimarisi** – CNN’ler olarak da bilinen evrişimli sinir ağları, genellikle aşağıdaki katmanlardan oluşan belirli bir tür sinir ağıdır:



6 Evrişim katmanı ve ortaklama katmanı, sonraki bölümlerde açıklanan hiperparametreler ile ince ayar (fine-tuned) yapılabilir.

### 1.2 Katman tipleri

◻ **Evrişim katmanı (CONV)** – Evrişim katmanı (CONV) evrişim işlemlerini gerçekleştiren filtreleri,  $I$  girişini boyutlarına göre tararken kullanır. Hiperparametreleri  $F$  filtré boyutunu ve  $S$  adımını içerir. Elde edilen çıktı  $O$ , öznitelik haritası veya aktivasyon haritası olarak adlandırılır.

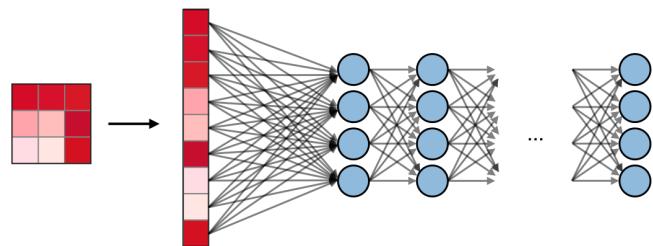


*Not: evrişim adımı, 1B ve 3B durumlarda da genelleştirilebilir ( $B$ : boyut).*

◻ **Ortaklama (POOL)** – Ortaklama katmanı (POOL), tipik olarak bir miktar uzamsal değişkenlik gösteren bir evrişim katmanından sonra uygulanan bir örnekleme işlemidir. Özellikle, maksimum ve ortalama ortaklama, sırasıyla maksimum ve ortalama değerin alındığı özel ortaklama türleridir.

	Maksimum ortaklama	Ortalama ortaklama
Amaç	Her ortaklama işlemi, geçerli matrisin maksimum değerini seçer	Her ortaklama işlemi, geçerli matrisin değerlerinin ortalaması alır
Görsel Açıklama		
Açıklama	- Algılanan özellikleri korur - En çok kullanılan	- Boyut azaltarak örneklenmişlik öznitelik haritası - LeNet'te kullanılmış

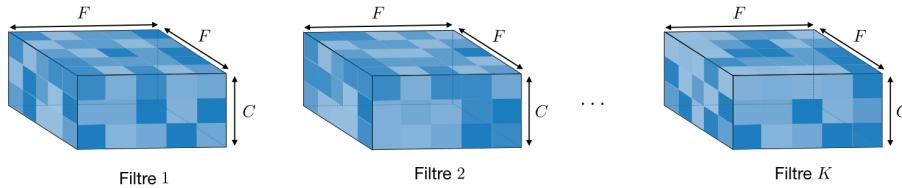
□ **Tam Bağlantı (FC)** – Tam bağlı katman (FC), her girişin tüm nöronlara bağlı olduğu bir giriş üzerinde çalışır. Eğer varsa, FC katmanları genellikle CNN mimarisinin sonuna doğru bulunur ve sınıf skorları gibi hedefleri optimize etmek için kullanılabilir.



### 1.3 Hiperparametrelerin filtrelenmesi

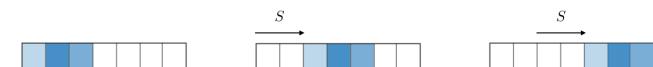
Evrişim katmanı, hiperparametrelerininardındaki anlamı bilmenin önemli olduğu filtreler içerir.

□ **Bir filtrenin boyutları** –  $C$  kanalları içeren bir girişe uygulanan  $F \times F$  boyutunda bir filtr,  $I \times I \times C$  boyutundaki bir girişte evrişim gerçekleştiren ve aynı zamanda bir çıkış öznitelik haritası üretken  $F$  aktivitesi (aktivasyon olarak da adlandırılır  $O$ )  $O \times O \times K$  boyutunda harita.



Not:  $F \times F$  boyutunda  $K$  filtrelereinin uygulanması,  $O \times O \times K$  boyutunda bir çıktı öznitelik haritasının olmasını sağlar.

□ **Adım aralığı** – Evrişimli veya bir ortaklama işlemi için,  $S$  adım (adım aralığı), her işleminden sonra pencerenin hareket ettiği piksel sayısını belirtir.



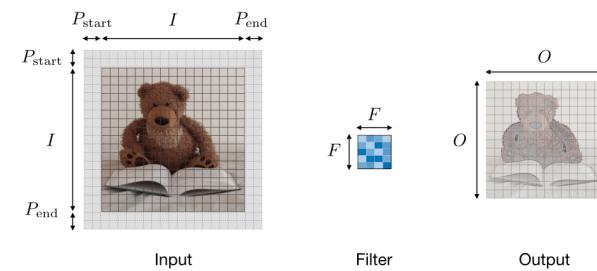
□ **Sıfır ekleme/doldurma** – Sıfır ekleme/doldurma, girişin sınırlarının her bir tarafına  $P$  sıfır ekleme işlemini belirtir. Bu değer manuel olarak belirlenebilir veya aşağıda detaylandırılan üç moddan biri ile otomatik olarak ayarlanabilir:

	Geçerli	Aynı	Tüm
Değer	$P = 0$	$P_{\text{start}} = \left\lceil \frac{S \lceil \frac{I}{S} \rceil - I + F - S}{2} \right\rceil$ $P_{\text{end}} = \left\lceil \frac{S \lceil \frac{I}{S} \rceil - I + F - S}{2} \right\rceil$	$P_{\text{start}} \in [0, F - 1]$ $P_{\text{end}} = F - 1$
Görsel Açıklama			
Amaç	- Ekleme/doldurma yok - Boyutlar uyumuyorsa son evrişimi düşürür	- Öznitelik harita büyüğünü sahip ekleme/doldurma $\lceil \frac{I}{S} \rceil$ - Çıktı boyutu matematiksel olarak uygundur - 'Yarım' ekleme olarak da bilinir	- Son konvolüsyonların giriş sınırlarına uygulandığı maksimum ekleme - Filtre girişi uçtan uca 'görür'

### 1.4 Hiperparametreleri ayarlama

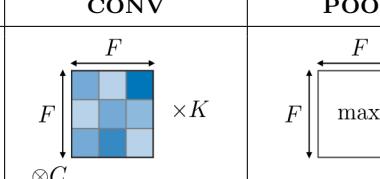
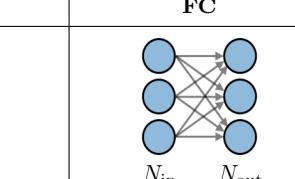
□ **Evrişim katmanında parametre uyumu** – Girdinin hacim büyülüği  $I$  uzunluğu,  $F$  filtresinin uzunluğu,  $P$  sıfır ekleme miktarı,  $S$  adım aralığı, daha sonra bu boyut boyunca öznitelik haritasının  $O$  çıkış büyülüği belirtilir:

$$O = \frac{I - F + P_{\text{start}} + P_{\text{end}}}{S} + 1$$



*Not: çoğunlukla,  $P_{start} = P_{end} \triangleq P$ , bu durumda  $P_{start} + P_{end}$ 'i yukarıdaki formülde  $2P$  ile değiştirebiliriz.*

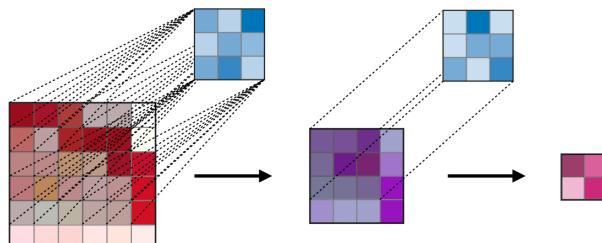
□ **Modelin karmaşıklığını anlama** – Bir modelin karmaşıklığını değerlendirmek için mimarisinin sahip olduğu parametrelerin sayısını belirlemek genellikle yararlıdır. Bir evrişimsi sınırlarının belirli bir katmanında, aşağıdaki şekilde yapılır:

	CONV	POOL	FC
Görsel Açıklama			
Giriş boyutu	$I \times I \times C$	$I \times I \times C$	$N_{in}$
Çıkış boyutu	$O \times O \times K$	$O \times O \times C$	$N_{out}$
Parametre sayısı	$(F \times F \times C + 1) \cdot K$	0	$(N_{in} + 1) \times N_{out}$
Not	<ul style="list-style-type: none"> <li>- Filtre başına bir bias(önyargı) parametresi</li> <li>- Çoğu durumda, <math>S &lt; F</math></li> <li>- <math>K</math> için ortak bir seçenek <math>2C</math>'dir</li> </ul>	<ul style="list-style-type: none"> <li>- Ortaklama işlemi kanal bazında yapılır</li> <li>- Çoğu durumda, <math>S = F</math></li> </ul>	<ul style="list-style-type: none"> <li>- Giriş bağlantılılmış</li> <li>- Nöron başına bir bias parametresi</li> <li>- Tam bağlantı (FC) nöronlarının sayısı yapısal kısıtlamalardan arındırılmış</li> </ul>

□ **Evrişim sonucu oluşan haritanın boyutu** –  $k$  katmanında filtre çıkışı,  $k$ -inci aktivasyon haritasının her bir pikselinin 'görebileceği' girişin  $R_k \times R_k$  olarak belirtilen alanını ifade eder.  $F_j$ ,  $j$  ve  $S_i$  katmanlarının filtre boyutu,  $i$  katmanın adım aralığı ve  $S_0 = 1$  (ilk adım aralığının 1 seçilmesi durumu) kuralıyla,  $k$  katmanındaki işlem sonucunda elde edilen aktivasyon haritasının boyutları bu formülle hesaplanabilir:

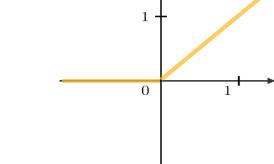
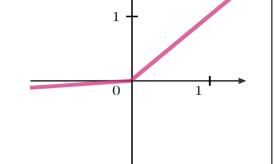
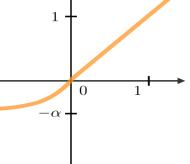
$$R_k = 1 + \sum_{j=1}^k (F_j - 1) \prod_{i=0}^{j-1} S_i$$

Aşağıdaki örnekte,  $F_1 = F_2 = 3$  ve  $S_1 = S_2 = 1$  için  $R_2 = 1 + 2 \cdot 1 + 2 \cdot 1 = 5$  sonucu elde edilir.



## 1.5 Yaygın olarak kullanılan aktivasyon fonksiyonları

□ **Düzeltilmiş Doğrusal Birim** – Düzeltilmiş doğrusal birim katmanı (ReLU), ( $g$ )'nın tüm elemanlarında kullanılan bir aktivasyon fonksiyonudur. Doğrusal olmamaları ile ağıın öğrenmesi amaçlanmaktadır. Çeşitleri aşağıdaki tabloda özetlenmiştir:

ReLU	Sızıntı ReLU	ELU
$g(z) = \max(0, z)$	$g(z) = \max(\epsilon z, z)$ ile $\epsilon \ll 1$	$g(z) = \max(\alpha(e^z - 1), z)$ ile $\alpha \ll 1$
		
Doğrusal olmama karmaşıklığı biyolojik olarak yorumlanabilir	Negatif değerler için ölen ReLU sorununu giderir	Her yerde türevlenebilir

□ **Softmax** – Softmax adımı,  $x \in \mathbb{R}^n$  skorlarının bir vektörünü girdi olarak alan ve mimarinin sonunda softmax fonksiyonundan  $p \in \mathbb{R}^n$  çıkış olasılık vektörünü oluşturan genelleştirilmiş bir lojistik fonksiyon olarak görülebilir. Aşağıdaki gibi tanımlanır:

$$p = \begin{pmatrix} p_1 \\ \vdots \\ p_n \end{pmatrix} \quad \text{buna karşılık} \quad p_i = \frac{e^{x_i}}{\sum_{j=1}^n e^{x_j}}$$

## 1.6 Nesne algılama

□ **Model tipleri** – Burada, nesne tanım algoritmasının doğası gereği 3 farklı kestirim türü vardır. Aşağıdaki tabloda açıklanmıştır:

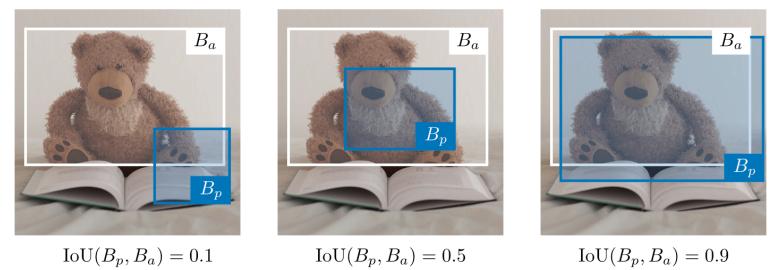
Görüntü sınıflandırma	Sınıflandırma ve lokalizasyon (konumlama)	Algılama
Oyuncak aya	Oyuncak aya	Oyuncak aya Kitap
- Bir görüntüyü sınıflandırır  - Nesnenin olasılığını tahmin eder	- Görüntüdeki bir nesneyi algılar/tanır  - Nesnenin olasılığını ve bulunduğu yeri tahmin eder	- Bir görüntüdeki birden fazla nesneyi algılar  - Nesnelerin olasılıklarını ve nerede oldukları tahmin eder
Geleneksel CNN	Basitleştirilmiş YOLO, R-CNN	YOLO, R-CNN

◻ **Algılama** – Nesne algılama bağlamında, nesneyi konumlandırmak veya görüntüdeki daha karmaşık bir şekli tespit etmek isteyip istemediğimize bağlı olarak farklı yöntemler kullanılır. İki ana tablo aşağıdaki tabloda özetlenmiştir:

Sınırlayıcı kutu ile tespit	Karakteristik nokta algılama
Görüntüde nesnenin bulunduğu yeri algılar	- Bir nesnenin şeklini veya özelliklerini algılar (örneğin gözler) - Daha ayrıntılı
Kutu merkezi $(b_x, b_y)$ , yükseklik $b_h$ ve genişlik $b_w$	Referans noktalar $(l_{1x}, l_{1y}), \dots, (l_{nx}, l_{ny})$

◻ **Kesiştilmiş Bölgeler** – Kesiştilmiş Bölgeler, IoU (*<i>Intersection over Union</i>*) olarak da bilinir. Birleştirilmiş sınırlama kutusu, tahmin edilen sınırlama kutusu ( $B_p$ ) ile gerçek sınırlama kutusu  $B_a$  üzerinde ne kadar doğru konumlandırıldığı ölçülen bir fonksiyondur. Olarak tanımlanır:

$$\text{IoU}(B_p, B_a) = \frac{B_p \cap B_a}{B_p \cup B_a}$$

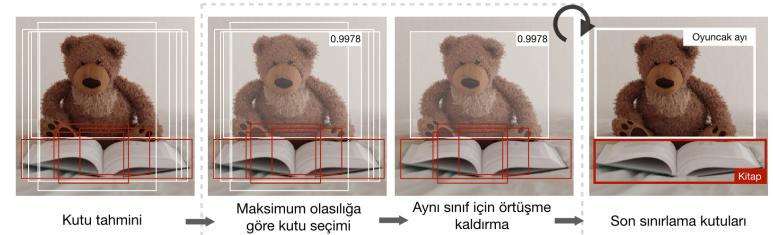


*Not: Her zaman  $\text{IoU} \in [0,1]$  ile başlarız. Kural olarak, Öngörülen bir sınırlama kutusu  $B_p$ ,  $\text{IoU}(B_p, B_a) \geq 0.5$  olması durumunda makul derecede iyi olarak kabul edilir.*

◻ **Öneri kutular** – Öneri (Anchor) kutular, örtüsen sınırlayıcı kutuları öngörmek için kullanılan bir tekniktir. Uygulamada, aynı anda birden fazla kutuya tahmin etmesine izin verilir, burada her kutu tahmini belirli bir geometrik öznitelik setine sahip olmakla sınırlıdır. Örneğin, ilk tahmin potansiyel olarak verilen bir formun dikdörtgen bir kutusudur, ikincisi ise farklı bir geometrik formun başka bir dikdörtgen kutusudur.

◻ **Maksimum olmayan bastırma** – Maksimum olmayan bastırma tekniği, nesne için yineleme ve örtüsen önerileri içinde en uygun temsilleri seçerek örtüşmesi düşük olan kutuları kaldırmayı amaçlar. Olasılık tahmini 0.6'dan daha düşük olan tüm kutuları çıkardıktan sonra, kalan kutular ile aşağıdaki adımlar tekrarlanır: Verilen bir sınıf için,

- Adım 1: En büyük tahmin olasılığı olan kutuyu seçin.
- Adım 2: Önceki kutuya  $\text{IoU} \geq 0.5$  olan herhangi bir kutuyu çıkarın.



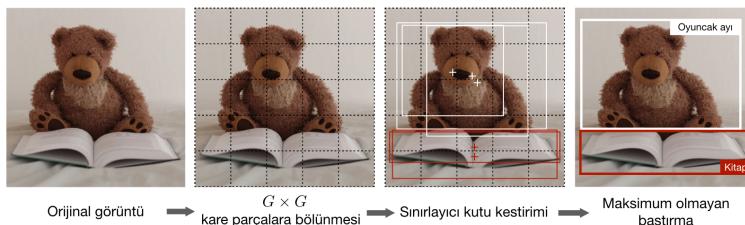
◻ **YOLO** – You Only Look Once (YOLO), aşağıdaki adımları uygulayan bir nesne algılama algoritmasıdır:

- Adım 1: Giriş görüntüsünü  $G \times G$  kare parçalara (hücrelere) bölün.
- Adım 2: Her bir hücre için, aşağıdaki formdan  $y$ 'yi öngören bir CNN çalıştırın:

$$y = \left[ \underbrace{p_c, b_x, b_y, b_h, b_w, c_1, c_2, \dots, c_p, \dots}_{k \text{ kez tekrarlayın}} \right]^T \in \mathbb{R}^{G \times G \times k \times (5+p)}$$

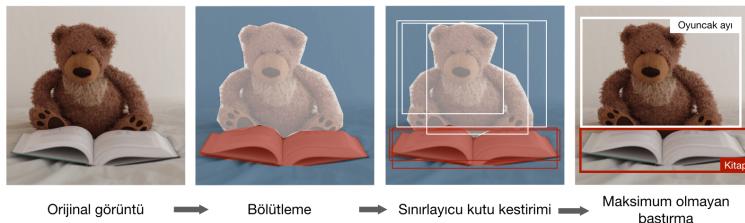
$p_c$ 'nin bir nesneyi algılama olasılığı olduğu durumlarda,  $b_x, b_y, b_h, b_w$  tespit edilen olası sınırlayıcı kutusunun özellikleridir,  $c_1, \dots, c_p$ ,  $p$  sınıflarının tespit edilen one-hot temsildir ve  $k$  öneri (*<i>anchor</i>*) kutularının sayısıdır.

- Adım 3: Potansiyel yinelik çakışan sınırlayıcı kutuları kaldırmak için maksimum olmayan bastırma algoritmasını çalıştır.



*Not:  $p_c = 0$  olduğunda, ağaç herhangi bir nesne algılamamaktadır. Bu durumda, ilgili  $b_x, \dots, c_p$  tahminleri dikkate alınmamalıdır.*

□ **R-CNN** – Evrişimli Sinir Ağları ile Bölge Bulma (R-CNN), potansiyel olarak sınırlayıcı kutuları bulmak için görüntüyü böülüten (segmente eden) ve daha sonra sınırlayıcı kutularda en olası nesneleri bulmak için algılama algoritmasını çalıştan bir nesne algılama algoritmasıdır.



*Not: Orijinal algoritma hesaplamalı olarak maliyetli ve yavaş olmasına rağmen, yeni mimariler algoritmanın Hızlı R-CNN ve Daha Hızlı R-CNN gibi daha hızlı çalışmasını sağlamıştır.*

### 1.6.1 Yüz doğrulama ve tanıma

□ **Model tipleri** – İki temel model aşağıdaki tabloda özetlenmiştir:

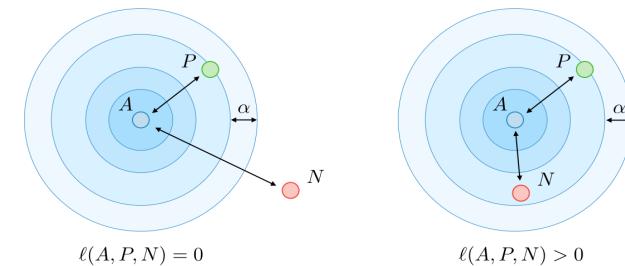
Yüz doğrulama	Yüz tanıma
- Bu doğru kişi mi? - Bire bir arama	- Veritabanındaki $K$ kişilerden biri mi? - Bire-çok arama
Sorgu  Kaynak 	Sorgu  Veri tabanı 

□ **Tek Atış (One-Shot) Öğrenme** – Tek Atış Öğrenme, verilen iki görüntünün ne kadar farklı olduğunu belirleyen benzerlik fonksiyonunu öğrenmek için sınırlı bir eğitim seti kullanan bir yüz doğrulama algoritmasıdır. İki resme uygulanan benzerlik fonksiyonu sıkılıkla kaydedilir (*d*(resim 1, resim 2)).

□ **Siyam (Siamese) Ağı** – Siyam Ağı, iki görüntünün ne kadar farklı olduğunu ölçmek için görüntülerin nasıl kodlanacağını öğrenmeyi amaçlar. Belirli bir giriş görüntüsü  $x^{(i)}$  için kodlanmış çıkış genellikle  $f(x^{(i)})$  olarak alınır.

□ **Üçlü kayıp** – Üçlü kayıp  $\ell$ ,  $A$  (öneri),  $P$  (pozitif) ve  $N$  (negatif) görüntülerinin üçlüsünün gömülü gösterimde hesaplanan bir kayıp fonksiyonudur. Öneri ve pozitif örnek aynı sınıfa aitken, negatif örnek bir diğerine aittir.  $\alpha \in \mathbb{R}^+$  marjin parametresini çağırarak, bu kayıp aşağıdaki gibi tanımlanır:

$$\ell(A, P, N) = \max(d(A, P) - d(A, N) + \alpha, 0)$$



### 1.6.2 Sinirsel stil transferi (aktarımı)

□ **Motivasyon** – Sinirsel stil transferinin amacı, verilen bir  $C$  içeriğine ve verilen bir  $S$  stiline dayanan bir  $G$  görüntüsü oluşturmaktr.



□ **Aktivasyon** – Belirli bir  $l$  katmanında, aktivasyon  $a^{[l]}$  olarak gösterilir ve  $n_H \times n_w \times n_c$  boyutlarındadır.

□ **İçerik maliyeti fonksiyonu** – İçerik maliyeti fonksiyonu  $J_{\text{content}}(C, G)$ ,  $G$  oluşturulan görüntüsünün,  $C$  orijinal içerik görüntüsünden ne kadar farklı olduğunu belirlemek için kullanılır. Aşağıdaki gibi tanımlanır:

$$J_{\text{content}}(C, G) = \frac{1}{2} \|a^{[l](C)} - a^{[l](G)}\|^2$$

□ **Stil matrisi** – Stil matrisi  $G^{[l]}$ , belirli bir  $l$  katmanının her birinin  $G_{kk'}^{[l]}$  elemanlarının  $k$  ve  $k'$  kanallarının ne kadar ilişkili olduğunu belirlediği bir Gram matristir.  $a^{[l]}$  aktivasyonlarına göre aşağıdaki gibi tanımlanır:

$$G_{kk'}^{[l]} = \sum_{i=1}^{n_H^{[l]}} \sum_{j=1}^{n_w^{[l]}} a_{ijk}^{[l]} a_{ijk'}^{[l]}$$

*Not: Stil görüntüsü ve oluşturulan görüntü için stil matrisi, sırasıyla  $G^{[l](S)}$  ve  $G^{[l](G)}$  olarak belirtimmiştir.*

◻ **Stil maliyeti fonksiyonu** – Stil maliyeti fonksiyonu  $J_{\text{style}}(S, G)$ , oluşturulan  $G$  görüntüsünün  $S$  stilinden ne kadar farklı olduğunu belirlemek için kullanılır. Aşağıdaki gibi tanımlanır:

$$J_{\text{style}}^{[l]}(S, G) = \frac{1}{(2n_H n_w n_c)^2} \|G^{[l](S)} - G^{[l](G)}\|_F^2 = \frac{1}{(2n_H n_w n_c)^2} \sum_{k,k'=1}^{n_c} \left( G_{kk'}^{[l](S)} - G_{kk'}^{[l](G)} \right)^2$$

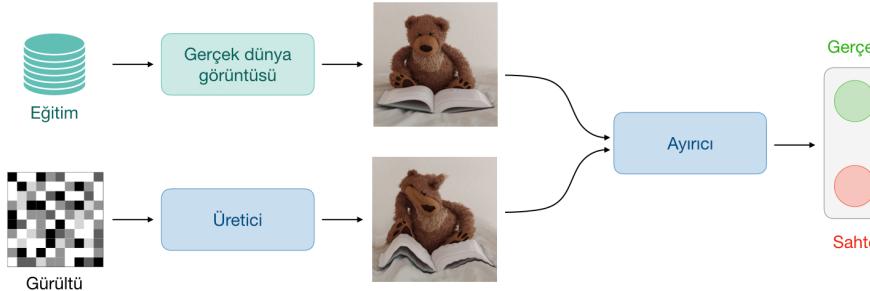
◻ **Genel maliyet fonksiyonu** – Genel maliyet fonksiyonu,  $\alpha, \beta$  parametreleriyle ağırlıklandırılan içerik ve stil maliyet fonksiyonlarının bir kombinasyonu olarak tanımlanır:

$$J(G) = \alpha J_{\text{content}}(C, G) + \beta J_{\text{style}}(S, G)$$

*Not: yüksek bir  $\alpha$  değeri modelin içeriğe daha fazla önem vermesini sağlarken, yüksek bir  $\beta$  değeri de stile önem verir.*

### 1.6.3 Hesaplama ipuçları kullanan mimariler

◻ **Çekişmeli Üretici Ağlar** – GAN olarak da bilinen çekişmeli üretici ağlar, modelin üretici denilen ve gerçek imajı ayırt etmeye amaçlayan ayrıciya beslenecek en doğru çıktıının oluşturulmasını amaçladığı üretici ve ayırt edici bir modelden oluşur.



*Not: GAN'ın kullanım alanları, yazdan görüntüye, müzik üretimi ve sentezi.*

◻ **ResNet** – Artık Ağ mimarisi (ResNet olarak da bilinir), eğitim hatasını azaltmak için çok sayıda katman içeren artık bloklar kullanır. Artık blok aşağıdaki karakterizasyon denklemine sahiptir:

$$a^{[l+2]} = g(a^{[l]} + z^{[l+2]})$$

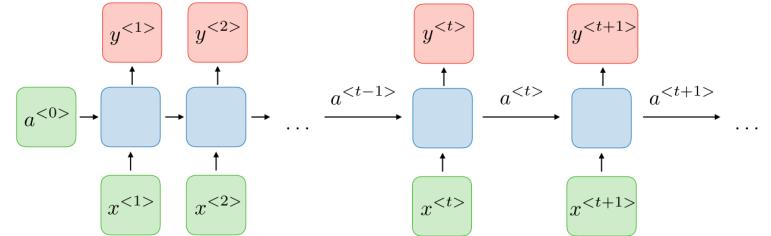
◻ **Inception Ağ** – Bu mimari inception modüllerini kullanır ve özelliklerini çeşitlendirme yoluyla performansını artırmak için farklı evrişim kombinasyonları denemeyi amaçlamaktadır. Özellikle, hesaplama yükünü sınırlamak için  $1 \times 1$  evrişim hilesini kullanır.

## 2 Tekrarlayan Yapay Sinir Ağları

Başak Buluz ve Yavuz Kömeçoğlu tarafından çevrilmiştir

### 2.1 Genel Bakış

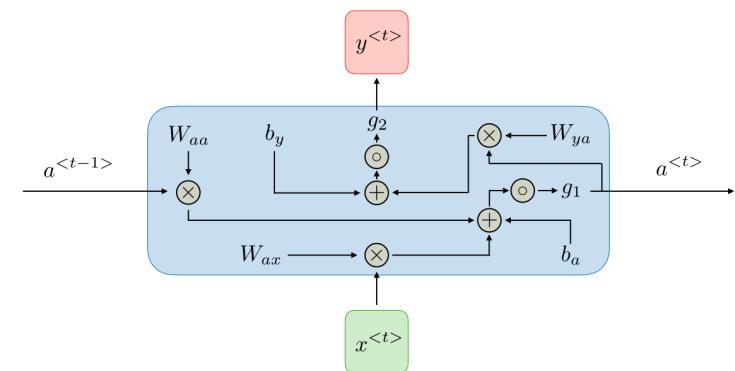
◻ **Geleneksel bir RNN mimarisi** – RNN'ler olarak da bilinen tekrarlayan sinir ağları, gizli durumları sahipken önceki çıktıların girdi olarak kullanılmasına izin veren bir sinir ağları sınıfıdır. Tipik olarak aşağıdaki gibidirler:



Her bir  $t$  zamanda,  $a^{<t>}$  aktivasyonu ve  $y^{<t>}$  çıktısı aşağıdaki gibi ifade edilir:

$$a^{<t>} = g_1(W_{aa}a^{<t-1>} + W_{ax}x^{<t>} + b_a) \quad \text{and} \quad y^{<t>} = g_2(W_{ya}a^{<t>} + b_y)$$

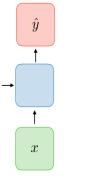
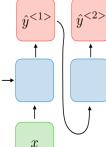
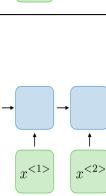
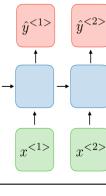
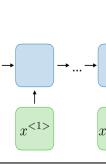
burada  $W_{ax}, W_{aa}, W_{ya}, b_a, b_y$  geçici olarak paylaşılacak katsayılardır ve  $g_1, g_2$  aktivasyon fonksiyonlarındır.



Tipik bir RNN mimarisinin优点ları ve eksileri aşağıdaki tabloda özetlenmiştir:

Avantajlar	Dezavantajları
<ul style="list-style-type: none"> <li>- Herhangi bir uzunluktaki girdilerin işlenmesi imkanı</li> <li>- Girdi büyüklüğüyle artmayan model boyutu</li> <li>- Geçmiş bilgileri dikkate alarak hesaplama</li> <li>- Zaman içinde paylaşılan ağırlıklar</li> </ul>	<ul style="list-style-type: none"> <li>- Yavaş hesaplama</li> <li>- Uzun zaman önceki bilgiye erişme zorluğu</li> <li>- Mevcut durum için gelecekteki herhangi bir girdinin düşünülememesi</li> </ul>

**RNN'lerin Uygulamaları** – RNN modelleri çoğunlukla doğal dil işleme ve konuşma tanıma alanlarında kullanılır. Farklı uygulamalar aşağıdaki tabloda özetlenmiştir:

RNN Türü	Örneklemme	Örnek
Bire bir $T_x = T_y = 1$		Geleneksel sinir ağı
Bire çok $T_x = 1, T_y > 1$		Müzik üretimi
Çoka bir $T_x > 1, T_y = 1$		Duygu sınıflandırma
Çoka çok $T_x = T_y$		İsim varlık tanıma
Çoka çok $T_x \neq T_y$		Makine çevirisisi

**Kayıp fonksiyonu** – Tekrarlayan bir sinir ağı olması durumunda, tüm zaman dilimlerindeki  $\mathcal{L}$  kayıp fonksiyonu, her zaman dilimindeki kayibi temel alınarak aşağıdaki gibi tanımlanır:

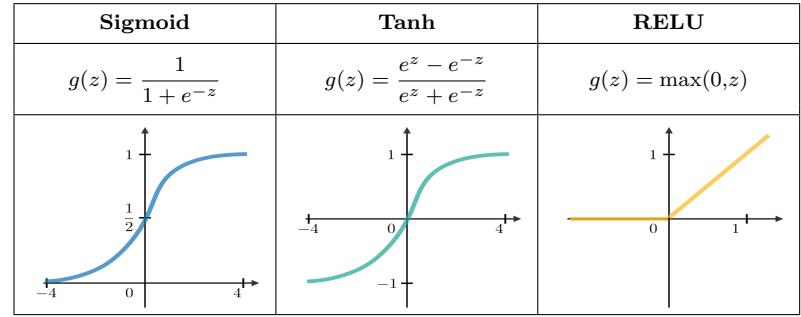
$$\mathcal{L}(\hat{y}, y) = \sum_{t=1}^{T_y} \mathcal{L}(\hat{y}^{<t>}, y^{<t>})$$

**Zamanla geri yayılma** – Geriye yayılma zamanın her noktasında yapılır.  $T$  zaman diliminde, ağırlık matrisi  $W$ 'ye göre  $\mathcal{L}$  kaybinin türevi aşağıdaki gibi ifade edilir:

$$\frac{\partial \mathcal{L}^{(T)}}{\partial W} = \sum_{t=1}^T \frac{\partial \mathcal{L}^{(T)}}{\partial W} \Big|_{(t)}$$

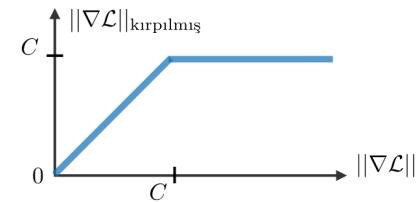
## 2.2 Uzun vadeli bağımlılıkların ele alınması

**Yaygın olarak kullanılan aktivasyon fonksiyonları** – RNN modüllerinde kullanılan en yaygın aktivasyon fonksiyonları aşağıda açıklanmıştır:



**Kaybolan/patlayan gradyan** – Kaybolan ve patlayan gradyan fenomenlerine RNN'ler bağlamında sıklıkla rastlanır. Bunların olmasının nedeni, katman sayısına göre katlanarak azalan/artan olabilen çarpımsal gradyan nedeniyle uzun vadeli bağımlılıkları yakalamanın zor olmasıdır.

**Gradyan kırpmacı** – Geri yayılma işlemi sırasında bazen karşılaşılan patlayan gradyan sorunuyla başa çıkmak için kullanılan bir tekniktir. Gradyan için maksimum değeri sınırlayarak, bu durum pratikte kontrol edilir.



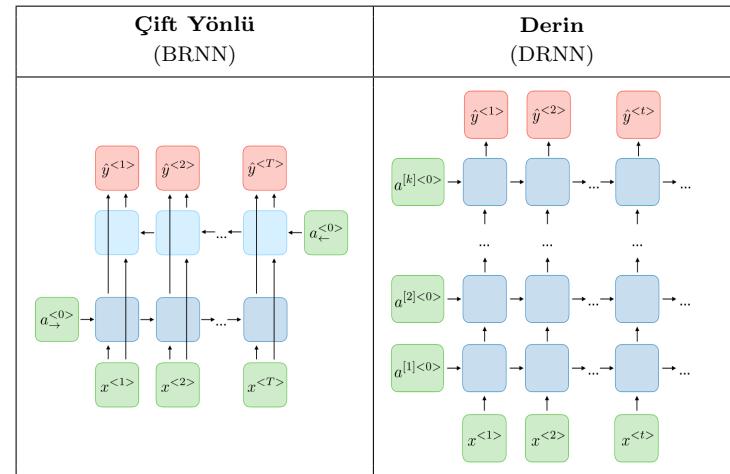
**Giriş kapıları çeşitleri** – Kaybolan gradyan problemini çözmek için bazı RNN türlerinde belirli kapılar kullanılır ve genellikle iyi tanımlanmış bir amaca sahiptir. Genellikle  $\Gamma$  olarak ifade edilir ve şuna eşittir:

$$\Gamma = \sigma(Wx^{<t>} + Ua^{<t-1>} + b)$$

burada  $W, U, b$  kapıya özgü katsayılardır ve  $\sigma$  ise sigmoid fonksiyondur. Temel olanlar aşağıdaki tabloda özetlenmiştir:

Kapının tipi	Rol	Kullanılan
Güncelleme kapısı $\Gamma_u$	Şimdi ne kadar geçmiş olması gereklidir?	GRU, LSTM
Uygunluk kapısı $\Gamma_r$	Önceki bilgiyi bırakır mı?	GRU, LSTM
Unutma kapısı $\Gamma_f$	Bir hücreyi sil ya da silme?	LSTM
Cıkış kapısı $\Gamma_o$	Bir hücreyi ortaya çıkarmak için ne kadar?	LSTM

◻ **GRU/LSTM** – Geçitli Tekrarlayan Birim (Gated Recurrent Unit-GRU) ve Uzun Kısa Süreli Bellek Birimleri (Long Short-Term Memory-LSTM), geleneksel RNN'lerin karşılaştığı kaybolan gradyan problemini ele alır, LSTM ise GRU'nun genelleştirilmiş halidir. Her bir mimarinin karakterizasyon denklemlerini özetleyen tablo aşağıdadır:

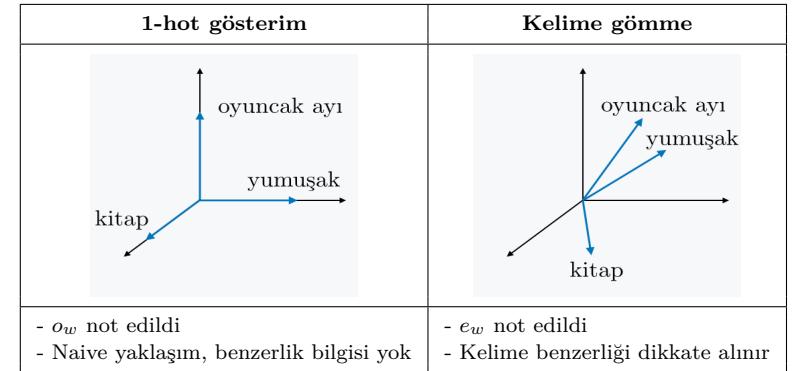


### 2.3 Kelime temsilini öğrenme

Bu bölümde  $V$  kelimeleri,  $|V|$  ise kelimelerin boyutlarını ifade eder.

◻ **Temsil etme teknikleri** – Kelimeleri temsil etmenin iki temel yolu aşağıdaki tabloda özetlenmiştir:

	Geçitli Tekrarlayan Birim (GRU)	Uzun Kısa Süreli Bellek (LSTM)
$\tilde{c}^{<t>}$	$\tanh(W_c[\Gamma_r * a^{<t-1>}, x^{<t>}] + b_c)$	$\tanh(W_c[\Gamma_r * a^{<t-1>}, x^{<t>}] + b_c)$
$c^{<t>}$	$\Gamma_u * \tilde{c}^{<t>} + (1 - \Gamma_u) * c^{<t-1>}$	$\Gamma_u * \tilde{c}^{<t>} + \Gamma_f * c^{<t-1>}$
$a^{<t>}$	$c^{<t>}$	$\Gamma_o * c^{<t>}$
Bağımlılıklar	$\begin{array}{c} c^{<t-1>} \\ \downarrow \\ \text{GRU} \\ \downarrow \\ \tilde{c}^{<t>} \\ \downarrow \\ \Gamma_u \quad \Gamma_r \\ \downarrow \quad \downarrow \\ a^{<t-1>} \quad x^{<t>} \end{array}$	$\begin{array}{c} c^{<t-1>} \\ \downarrow \\ \text{LSTM} \\ \downarrow \\ \Gamma_f \quad \Gamma_u \quad \Gamma_r \\ \downarrow \quad \downarrow \quad \downarrow \\ a^{<t-1>} \quad x^{<t>} \end{array}$



◻ **Gömme matrisi** – Belirli bir  $w$  kelimesi için  $E$  gömme matrisi, 1-hot temsilini  $e_w$  gömmesi sayesinde aşağıdaki gibi eşleştirilen bir matristir:

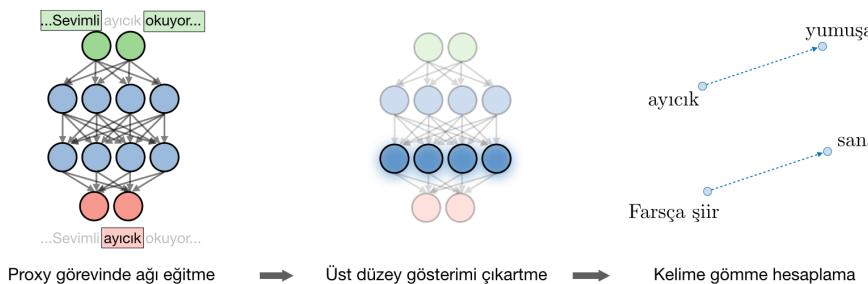
$$e_w = E o_w$$

*Not: Gömme matrisinin öğrenilmesi hedef/içerik olabilirlik modelleri kullanılarak yapılabilir.*

◻ **Word2vec** – Word2vec, belirli bir kelimenin diğer kelimelerle çevrili olma olasılığını tahmin ederek kelime gömmelerini öğrenmeye amaçlayan bir çerçevedir. Popüler modeller arasında skip-gram, negatif örneklemme ve CBOW bulunur.

Not:  $*$  işaretti iki vektör arasındaki birimsel çarpımı belirtir.

◻ **RNN varyantları** – Aşağıdaki tablo, diğer yaygın kullanılan RNN mimarilerini özetlemektedir:



**Skip-gram** – Skip-gram word2vec modeli verilen herhangi bir  $t$  hedef kelimesinin  $c$  gibi bir bağlam kelimesi ile gerçekleşme olasılığını değerlendirerek kelime gömmelerini öğrenen denetimli bir öğrenme görevidir.

$$P(t|c) = \frac{\exp(\theta_t^T e_c)}{\sum_{j=1}^{|V|} \exp(\theta_j^T e_c)}$$

Not: Softmax bölümünün paydasındaki tüm kelime dağarcığını toplamak, bu modeli hesaplama açısından maliyetli kilar. CBOW, verilen bir kelimeyi tahmin etmek için çevreleyen kelimeleri kullanan başka bir word2vec modelidir.

**Negatif örnekleme** – Belirli bir bağlamın ve belirli bir hedef kelimenin eşzamanlı olarak ortaya çıkışının muhtemel olup olmadığınnı değerlendirmesini, modellerin  $k$  negatif örnek kümeleri ve 1 pozitif örnek kümelerde eğitilmesini hedefleyen, lojistik regresyon kullanılarak ikili sınıflandırma kümesidir. Bağlam sözcüğü  $c$  ve hedef sözcüğü  $t$  göz önüne alındığında, tahmin şöyle ifade edilir:

$$P(y = 1|c,t) = \sigma(\theta_t^T e_c)$$

Not: Bu yöntem, skip-gram modelinden daha az hesaplama yapar.

**GloVe** – Kelime gösterimi için Global vektörler tanımının kısaltılmış hali olan GloVe, eşzamanlı bir  $X$  matrisi kullanan ki burada her bir  $X_{i,j}$ , bir hedefin bir  $j$  bağlamında gerçekleştiği sayısını belirten bir kelime gömme teknigidir. Maliyet fonksiyonu  $J$  aşağıdaki gibidir:

$$J(\theta) = \frac{1}{2} \sum_{i,j=1}^{|V|} f(X_{i,j})(\theta_i^T e_j + b_i + b'_j - \log(X_{i,j}))^2$$

$f, X_{i,j} = 0 \implies f(X_{i,j}) = 0$  olacak şekilde bir ağırlıklandırma fonksiyonudur.

Bu modelde  $e$  ve  $\theta$ 'nın oynadığı simetri göz önüne alındığında,  $e_w^{(\text{final})}$ ,nin kelime gömmesi söyle ifade edilir:

$$e_w^{(\text{final})} = \frac{e_w + \theta_w}{2}$$

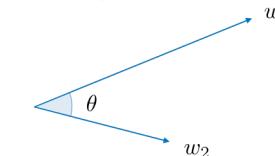
Not: Öğrenilen kelime gömme bileşenlerinin ayrı ayrı bileşenleri tam olarak yorumlanamaz.

## 2.4 Kelimelerin karşılaştırılması

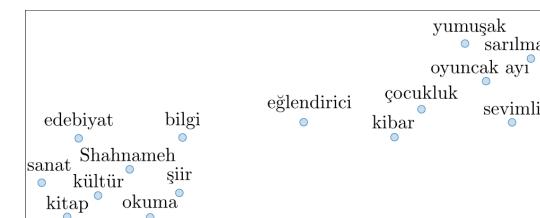
**Kosinüs benzerliği** –  $w_1$  ve  $w_2$  kelimeleri arasındaki kosinüs benzerliği şu şekilde ifade edilir:

$$\text{similarity} = \frac{w_1 \cdot w_2}{\|w_1\| \|w_2\|} = \cos(\theta)$$

Not:  $\theta$ ,  $w_1$  ve  $w_2$  kelimeleri arasındaki açıdır.



**t-SNE** – t-SNE ( $t$ -dağıtımlı Stokastik Komşu Gömme), yüksek boyutlu gömmeleri daha düşük boyutlu bir alana indirmeyi amaçlayan bir tekniktir. Uygulamada, kelime uzaylarını 2B alanda görselleştirmek için yaygın olarak kullanılır.



## 2.5 Dil modeli

**Genel bakış** – Bir dil modeli  $P(y)$  cümlesinin olasılığını tahmin etmeyi amaçlar.

**n-gram modeli** – Bu model, eğitim verilerindeki görünüm sayısını sayarak bir ifadenin bir korpusa ortaya çıkma olasılığını ölçmeye amaçlayan naif bir yaklaşımdır.

**Karışıklık** – Dil modelleri yaygın olarak, PP olarak da bilinen karışıklık metriği kullanılarak değerlendirilir ve bunlar  $T$  kelimelerinin sayısıyla normalize edilmiş veri setinin ters olasılığı olarak yorumlanabilir. Karışıklık, daha düşük, daha iyi ve şöyle tanımlanır:

$$\text{PP} = \prod_{t=1}^T \left( \frac{1}{\sum_{j=1}^{|V|} y_j^{(t)} \cdot \hat{y}_j^{(t)}} \right)^{\frac{1}{T}}$$

Not: PP, t-SNE'de yaygın olarak kullanılır.

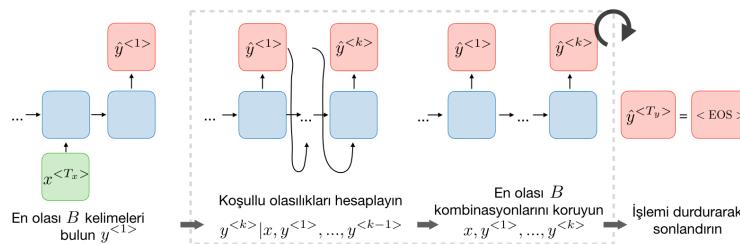
## 2.6 Makine çevirisi

**Genel bakış** – Bir makine çeviri modeli, daha önce yerleştirilmiş bir kodlayıcı ağına sahip olması dışında, bir dil modeline benzer. Bu nedenle, bazen koşullu dil modeli olarak da adlandırılır. Amaç şu şekilde bir cümle bulmaktır:

$$y = \arg \max_{y^{<1>} \dots, y^{<T_y>}} P(y^{<1>} \dots, y^{<T_y>} | x)$$

**Işın arama** – Makine çevirisinde ve konuşma tanımada kullanılan ve  $x$  girişi verilen en olası cümleyi bulmak için kullanılan sezgisel bir arama algoritmasıdır.

- Adım 1: En olası  $B$  kelimeleri bulun  $y^{<1>}$
- Adım 2: Koşullu olasılıkları hesaplayın  $y^{<k>}|x, y^{<1>}, \dots, y^{<k-1>}$
- Adım 3: En olası  $B$  kombinasyonlarını koruyun  $x, y^{<1>}, \dots, y^{<k>}$



Not: Eğer işin genişliği 1 olarak ayarlanmışsa, bu naif (naive) bir ağızozlu (greedy) aramaya eşdeğerdir.

**Işın genişliği** – İşin genişliği  $B$ , işin araması için bir parametredir. Daha yüksek  $B$  değerleri daha iyi sonuç elde edilmesini sağlar fakat daha düşük performans ve daha yüksek hafıza ile. Küçük  $B$  değerleri daha kötü sonuçlara neden olur, ancak hesaplama açısından daha az yoğundur.  $B$  için standart bir değer 10 civarındadır.

**Uzunluk normalizasyonu** – Sayısal stabiliteti artırmak için, işin arama genellikle, aşağıdaki gibi tanımlanan normalize edilmiş log-olabilirlik amacıyla adlandırılan normalize edilmiş hedefe uygulanır:

$$\text{Objective} = \frac{1}{T_y^\alpha} \sum_{t=1}^{T_y} \log [p(y^{<t>}|x, y^{<1>}, \dots, y^{<t-1>})]$$

Not:  $\alpha$  parametresi yumuşatıcı olarak görülebilir ve değeri genellikle 0,5 ile 1 arasındadır.

**Hata analizi** – Kötü bir çeviri elde edildiğinde, aşağıdaki hata analizini yaparak neden iyi bir çeviri almadığımızı araştırabiliriz:

Durum	$P(y^* x) > P(\hat{y} x)$	$P(y^* x) \leq P(\hat{y} x)$
Ana neden	Işin arama hatası	RNN hatası
Cözümler	Işin genişliğini artırma	- Farklı mimariyi deneme - Düzenleştirme - Daha fazla bilgi edinme

**Bleu puanı** – İki dilli değerlendirme alt ölçüği (bleu) puanı, makine çevirisinin ne kadar iyi olduğunu,  $n$ -gram hassasiyetine dayalı bir benzerlik puanı hesaplayarak belirler. Aşağıdaki gibi tanımlanır:

$$\text{bleu score} = \exp \left( \frac{1}{n} \sum_{k=1}^n p_k \right)$$

$p_n$ ,  $n$ -gramdaki bleu skorunun sadece aşağıdaki şekilde tanımlandığı durumlarda:

$$p_n = \frac{\sum_{\substack{\text{n-gram } \in \hat{y} \\ \text{n-gram } \in y}} \text{count}_{\text{clip}}(\text{n-gram})}{\sum_{\substack{\text{n-gram } \in \hat{y} \\ \text{n-gram } \in y}} \text{count}(\text{n-gram})}$$

Not: Yapay olarak şışirilmiş bir bleu skorunu önlemek için kısa öngörülen çevrilere küçük bir ceza verilebilir.

## 2.7 Dikkat

**Dikkat modeli** – Bu model, bir RNN'de girişin önemli olduğu düşünülen belirli kısımlarına dikkat etmesine olanak sağlar, sonuçta ortaya çıkan modelin pratikteki performansını artırır.  $\alpha^{<t,t'>}$  ile ifade edilen dikkat miktarı,  $a^{<t'>}$  aktivasyonu ve  $t$  zamanındaki  $c^{<t>}$  bağlamını  $y^{<t>}$  çıktıları olarak verir.

$$c^{<t>} = \sum_{t'} \alpha^{<t,t'>} a^{<t'>} \quad \text{with} \quad \sum_{t'} \alpha^{<t,t'>} = 1$$

Not: Dikkat skorları, görüntü alt yazılama ve makine çevirisinde yaygın kullanılır.



Sevimli bir oyuncak ayı Fars edebiyatı okuyor



Sevimli bir oyuncak ayı Fars edebiyatı okuyor

**Dikkat ağırlığı** –  $y^{<t>}$  çıktısının  $a^{<t'>}$  aktivasyonuna vermesi gereken dikkat miktarı, aşağıdaki gibi hesaplanan  $\alpha^{<t,t'>}$  ile verilir:

$$\alpha^{<t,t'>} = \frac{\exp(e^{<t,t'>})}{\sum_{t''=1}^{T_x} \exp(e^{<t,t''>})}$$

Not: hesaplama karmaşıklığı  $T_x$  'e göre ikinci derecedendir.

### 3 Derin Öğrenme püf noktaları ve ipuçları

Ayyuce Kızrak ve Yavuz Kömeçoğlu tarafından çevrilmiştir

#### 3.1 Veri işleme

**Veri artırma** – Derin öğrenme modelleri genellikle uygun şekilde eğitilmek için çok fazla veriye ihtiyaç duyar. Veri artırma tekniklerini kullanarak mevcut verilerden daha fazla veri üretmek genellikle yararlıdır. Temel işlemler aşağıdaki tabloda özetlenmiştir. Daha doğrusu, aşağıdaki girdi görüntüsüne bakıldığından, uygulayabileceğimiz teknikler şunlardır:

Orijinal	Çevirme	Rotasyon	Rastgele kırpmacı
			
- Herhangi bir değişiklik yapılmamış görüntü	- Görüntünün anlamının korunduğu bir eksene göre çevrilişmiş görüntü	- Hafif açılı döndürme - Yanlış yatay kalibrasyonu simule eder	- Görüntünün bir bölümüne rastgele odaklanma - Arka arkaya birkaç rasgele kesme yapılabilir

Renk değişimi	Gürültü ekleme	Bilgi kaybı	Kontrast değişimi
			
- RGB'nin nüansları biraz değiştirilmesi - Işığa maruz kahrken oluşabilecek gürültü	- Gürültü ekleme - Girdilerin kalite değişkenliğine daha fazla toleranslı olması	- Yok sayılan görüntüler - Görüntünün parçalardaki olası kayıplarını kopyalanması	- Gün içindeki ışık ve renk değişimini kontrolü

**Küme normalleştirme** – Bu,  $\{x_i\}$  kümesini normallestiren,  $\gamma, \beta$  hiperparametresinin bir adımıdır.  $\mu_B$  ve  $\sigma_B^2$ 'ye dikkat ederek, kümeyi düzeltmek istediklerimizin ortalaması ve varyansı şu şekilde yapılr:

$$x_i \leftarrow \gamma \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} + \beta$$

Genellikle tam-tüm bağlı/evrişimli bir katmandan sonra ve doğrusal olmayan bir katmandan önce yapılır. Daha yüksek öğrenme oranlarına izin vermeyi ve başlangıç durumuna güclü bir şekilde bağımlılığı azaltmayı amaçlar.

#### 3.2 Bir sinir ağının eğitilmesi

**Dönem (Epok/EPOCH)** – Bir modelin eğitimi kapsamında, modelin ağırlıklarını güncellemek için tüm eğitim setini kullandığı bir yinelemeye ifade etmek için kullanılan bir terimdir.

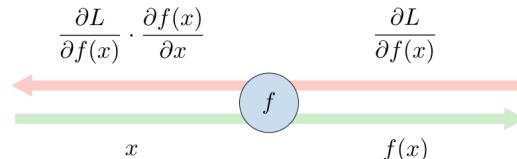
**Mini-küme gradyan (bayır) iniş** – Eğitim aşamasında, ağırlıkların güncellenmesi genellikle hesaplama karmaşıklıkları nedeniyle bir kerede ayarlanan tüm eğitime veya gürültü sorunları nedeniyle bir veri noktasına dayanmaz. Bunun yerine, güncelleme adımı bir toplu işdeki veri noktalarının sayısının ayarlayabileceğimiz bir hiperparametre olduğu mini kümelerle yapılır. Veriler mini-kümeler halinde alınır.

**Yitim fonksiyonu** – Belirli bir modelin nasıl bir performans gösterdiğini ölçmek için,  $L$  yitim (ayıp) fonksiyonu genellikle  $y$  gerçek çıktıların,  $z$  model çıktıları tarafından ne kadar doğru tahmin edildiğini değerlendirmek için kullanılır.

**Çapraz-entropi kaybı** – Yapay sinir ağlarında ikili sınıflandırma bağlamında, çapraz entropi kaybı  $L(z,y)$  yaygın olarak kullanılır ve şöyle tanımlanır:

$$L(z,y) = - \left[ y \log(z) + (1-y) \log(1-z) \right]$$

**Geriye yayılma** – Geri yayılma, asıl çıktıyi ve istenen çıktıyı dikkate alarak sinir ağındaki ağırlıkları güncellemek için kullanılan bir yöntemdir. Her bir ağırlığa göre türev, zincir kuralı kullanılarak hesaplanır.

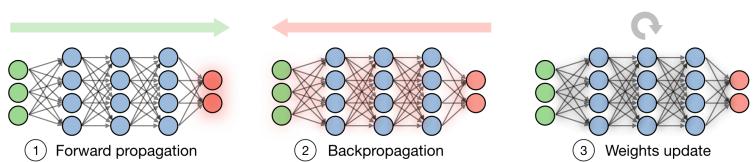


Bu yöntemi kullanarak, her ağırlık kurala göre güncellenir:

$$w \leftarrow w - \alpha \frac{\partial L(z,y)}{\partial w}$$

**Ağırlıkların güncellenmesi** – Bir sinir ağında, ağırlıklar aşağıdaki gibi güncellenir:

- Adım 1: Bir küme eğitim verisi alın ve kaybı hesaplamak için ileriye doğru ilerleyin.
- Adım 2: Her ağırlığa göre kaybm derecesini elde etmek için kaybı tekrar geriye doğru yayın.
- Adım 3: Ağın ağırlıklarını güncellemek için gradyanları kullanın.



### 3.3 Parametre ayarlama

**Xavier başlangıcı (ilkendirmeye)** – Ağırlıkları tamamen rastgele bir şekilde başlatmak yerine, Xavier başlangıcı, mimariye özgü özelliklerini dikkate alan ilk ağırlıkların alınmasını sağlar.

**Transfer öğrenme** – Bir derin öğrenme modelini eğitmek çok fazla veri ve daha da önemlisi çok zaman gerektirir. Kullanım durumumuza yönelik eğitim yapmak ve güçlendirmek için günler/haftalar süren dev veri setleri üzerinde önceden eğitilmiş ağırlıklardan yararlanmak genellikle yararlıdır. Elimizdeki ne kadar veri olduğuna bağlı olarak, aşağıdakilerden yararlanmanın farklı yolları:

Yöntem	Açıklama	w	b
Momentum	<ul style="list-style-type: none"> <li>- Osilasyonların azaltılması/yumuşatılması</li> <li>- SGD (Stokastik Gradyan/Bayır İnis) iyileştirmesi</li> <li>- Ayarlanacak 2 parametre</li> </ul>	$w - \alpha v_{dw}$	$b - \alpha v_{db}$
RMSprop	<ul style="list-style-type: none"> <li>- Ortalama Karekök yayılımı</li> <li>- Osilasyonları kontrol ederek öğrenme algoritmasını hızlandırır</li> </ul>	$w - \alpha \frac{dw}{\sqrt{s_{dw}}}$	$b \leftarrow b - \alpha \frac{db}{\sqrt{s_{db}}}$
Adam	<ul style="list-style-type: none"> <li>- Uyarlamalı Moment tahmini/kestirimi</li> <li>- En popüler yöntem</li> <li>- Ayarlanacak 4 parametre</li> </ul>	$w - \alpha \frac{v_{dw}}{\sqrt{s_{dw}} + \epsilon}$	$b \leftarrow b - \alpha \frac{v_{db}}{\sqrt{s_{db}} + \epsilon}$

Eğitim boyutu	Görselleştirme	Açıklama
Küçük		Tüm katmanlar dondurulur, softmax'taki ağırlıkları eğitilir
Orta		Çoğu katmanlar dondurulur, son katmanlar ve softmax katmanı ağırlıklar ile eğitilir
Büyük		Önceki eğitimlerde elde edilen ağırlıkları kullanarak katmanlar ve softmax için kullanır

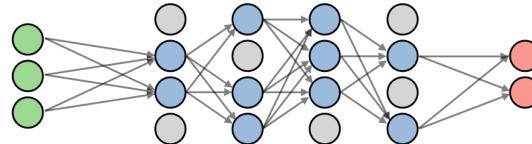
**Öğrenme oranı (adımı)** – Genellikle  $\alpha$  veya bazen  $\eta$  olarak belirtilen öğrenme oranı, ağırlıkların hangi hızda güncellendiğini belirler. Sabitlenebilir veya uyarlanabilir şekilde değiştirilebilir. Mevcut en popüler yöntemin adı Adam'dır ve öğrenme hızını ayarlayan bir yöntemdir.

**Uyarlanabilir öğrenme oranları** – Bir modelin eğitilmesi sırasında öğrenme oranının değişmesine izin vermek eğitim süresini kısaltabilir ve sayısal optimum çözümü iyileştirebilir. Adam optimizasyon yöntemi en çok kullanılan teknik olmasına rağmen, diğer yöntemler de faydalı olabilir. Bunlar aşağıdaki tabloda özetlenmiştir:

*Not: diğer yöntemler içinde Adadelta, Adagrad ve SGD.*

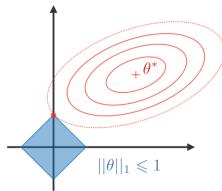
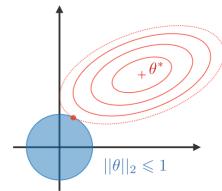
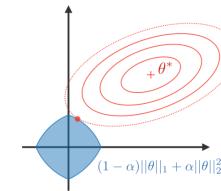
### 3.4 Düzenleştirme

**Seyreltme** – Seyreltme, sinir ağlarında,  $p > 0$  olasılıklı nöronları silerek eğitim verilerinin fazla kullanılmaması için kullanılan bir tekniktir. Modeli, belirli özellik kümelerine çok fazla güvenmekten kaçınmaya zorlar.



*Not: Coğuluklu derin öğrenme kütüphanesi, 'keep' ('tutma') parametresi  $1 - p$  aracılığıyla seyreltmeyi parametrize eder.*

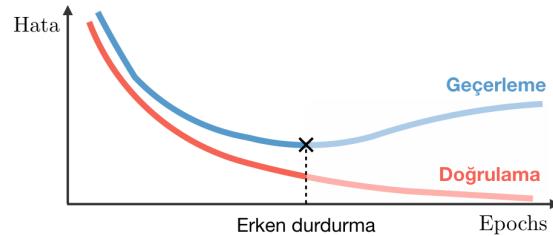
**Ağırlık düzenleme** – Ağırlıkların çok büyük olmadığından ve modelin eğitim setine uygun olmadığından emin olmak için, genellikle model ağırlıklarında düzenleme teknikleri uygulanır. Temel olanlar aşağıdaki tabloda özetlenmiştir:

LASSO	Ridge	Elastic Net
- Katsayıları 0'a düşürür - Değişken seçimi için iyi	Katsayıları daha küçük yapar	Değişken seçimi ile küçük katsayılar arasında ödünlendirme sağlar
		
$\dots + \lambda \ \theta\ _1$ $\lambda \in \mathbb{R}$	$\dots + \lambda \ \theta\ _2^2$ $\lambda \in \mathbb{R}$	$\dots + \lambda \left[ (1 - \alpha) \ \theta\ _1 + \alpha \ \theta\ _2^2 \right]$ $\lambda \in \mathbb{R}, \alpha \in [0,1]$

	Sayısal gradyan	Analitik gradyan
Formül	$\frac{df}{dx}(x) \approx \frac{f(x+h) - f(x-h)}{2h}$	$\frac{df}{dx}(x) = f'(x)$
Açıklamalar	<ul style="list-style-type: none"> <li>- Maliyetli; Kayıp, boyut başına iki kere hesaplanmalı</li> <li>- Analitik uygulamanın doğruluğunu anlamak için kullanılır</li> <li>- Ne çok küçük (sayısal dengesizlik) ne de çok büyük (zayıf gradyan yaklaşımı) seçimi yapılmalı, bunun için ödünlendirme gerekir</li> </ul>	<ul style="list-style-type: none"> <li>- 'Kesin' sonuç</li> <li>- Doğrudan hesaplama</li> <li>- Son uygulamada kullanılır</li> </ul>

★ ★ ★

□ **Erken durdurma** – Bu düzenlemeye teknigi, onaylama kaybı bir stabilliğe ulaştığında veya artmaya başladığında eğitim sürecini durdurur.



### 3.5 İyi uygulamalar

□ **Küçük kümelerin ezberlenmesi** – Bir modelde hata ayıklama yaparken, modelin mimarisinde büyük bir sorun olup olmadığını görmek için hızlı testler yapmak genellikle yararlıdır. Özellikle, modelin uygun şekilde eğitilebildiğinden emin olmak için, ezberleyecek mi diye görmek için ağ içinde bir mini kümeye eğitilir. Olmazsa, modelin normal boyutta bir eğitim setini bırakmadan, küçük bir kümeyi bile ezberleyecek kadar çok karmaşık ya da yeterince karmaşık olmadığı anlamına gelir.

□ **Gradyanların kontrolü** – Gradyan kontrolü, bir sınır ağının geriye doğru geçişinin uygulanması sırasında kullanılan bir yöntemdir. Analitik gradyanların değerini verilen noktalardaki sayısal gradyanlarla karşılaştırır ve doğruluk için bir kontrol rolü oynar.