Tekrarlayan Yapay Sinir Ağları El Kitabı VIP

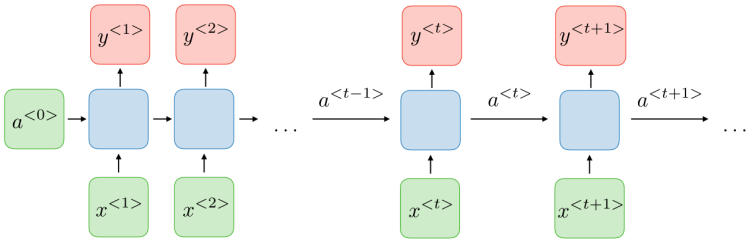
Afshine Amidi ve Shervine Amidi

April 30, 2019

Başak Buluz ve Yavuz Kömeçoğlu tarafından çevrilmiştir

Genel Bakış

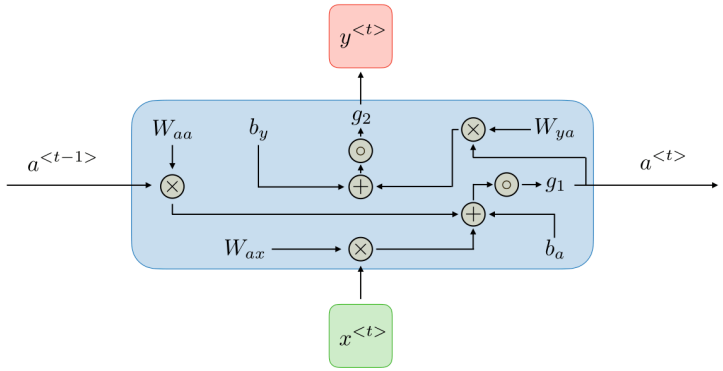
❒ **Geleneksel bir RNN mimarisi** – RNN'ler olarak da bilinen tekrarlayan sinir ağları, gizli durumlara sahipken önceki çıktıların girdi olarak kullanılmasına izin veren bir sinir ağları sınıfıdır. Tipik olarak aşağıdaki gibidirler:



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

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

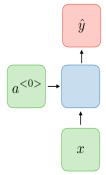
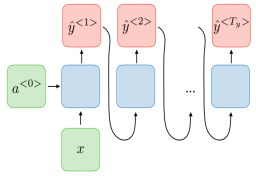
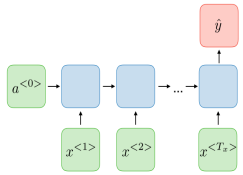
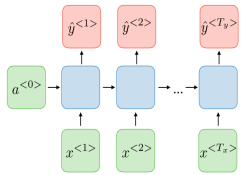
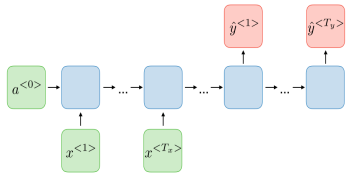
burada $W_{ax}, W_{aa}, W_{ya}, b_a, b_y$ geçici olarak paylaşılan katsayılardır ve g_1, g_2 aktivasyon fonksiyonlarıdır.



Tipik bir RNN mimarisinin artıları ve eksileri aşağıdaki tabloda özetlenmiştir:

Avantajlar	Dezavantajları
- Herhangi bir uzunluktaki girdilerin işlenmesi imkanı - Girdi büyüklüğüyle artmayan model boyutu - Geçmiş bilgileri dikkate alarak hesaplama - Zaman içinde paylaşılan ağırlıklar	- Yavaş hesaplama - Uzun zaman önceki bilgiye erişme zorluğu - Mevcut durum için gelecekteki herhangi bir girdinin düşünülememesi

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

RNN Türü	Örnekleme	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 çevirisi

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

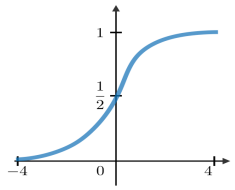
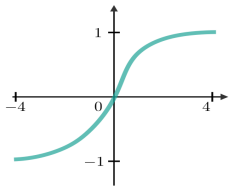
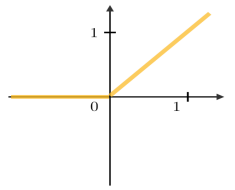
$$\boxed{\mathcal{L}(\widehat{y},y)=\sum_{t=1}^{T_y}\mathcal{L}(\widehat{y}^{<t>},y^{<t>})}$$

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

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

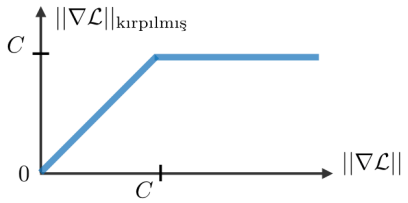
Uzun vadeli bağımlılıkların ele alınması

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

Sigmoid	Tanh	RELU
$g(z)=\frac{1}{1+e^{-z}}$	$g(z)=\frac{e^z-e^{-z}}{e^z+e^{-z}}$	$g(z)=\max(0,z)$
		

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



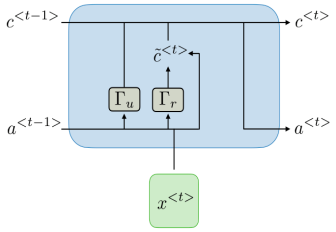
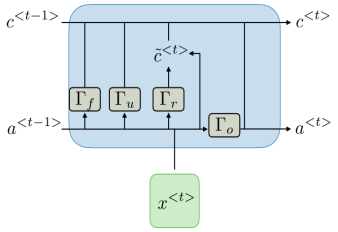
❒ **Giriş kapıları çeşitleri** – Kaybolan gradyan problemini çözmek için bazı RNN türlerinde belirli kapılar kullanılır ve genellikle iyi tanımlanmış bir amaca sahiptir. Genellikle Γ olarak ifade edilir ve şuna eşittir:

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

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

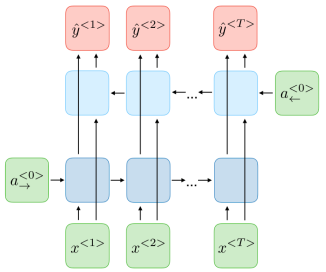
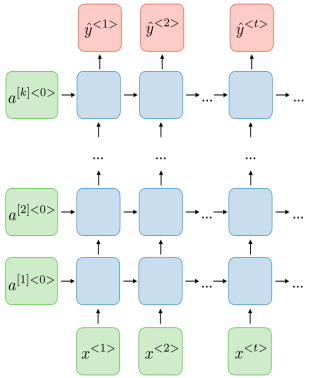
Kapının tipi	Rol	Kullanılan
Güncelleme kapısı Γ_u	Şimdi ne kadar geçmiş olması gerekir?	GRU, LSTM
Uygunluk kapısı Γ_r	Önceki bilgiyi bırak?	GRU, LSTM
Unutma kapısı Γ_f	Bir hücreyi sil ya da silme?	LSTM
Çıkış kapısı Γ_o	Bir hücreyi ortaya çıkarmak için ne kadar?	LSTM

❒ **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:

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

Not: $\star$ işareti iki vektör arasındaki birimsel çarpımı belirtir.

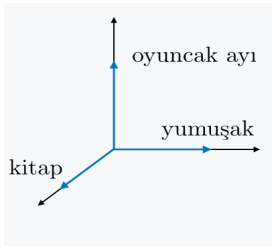
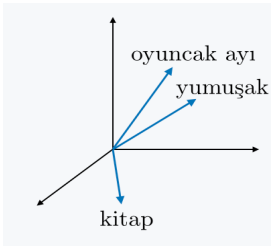
❒ **RNN varyantları** – Aşağıdaki tablo, diğer yaygın kullanılan RNN mimarilerini özetlemektedir:

Çift Yönlü (BRNN)	Derin (DRNN)
	

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:

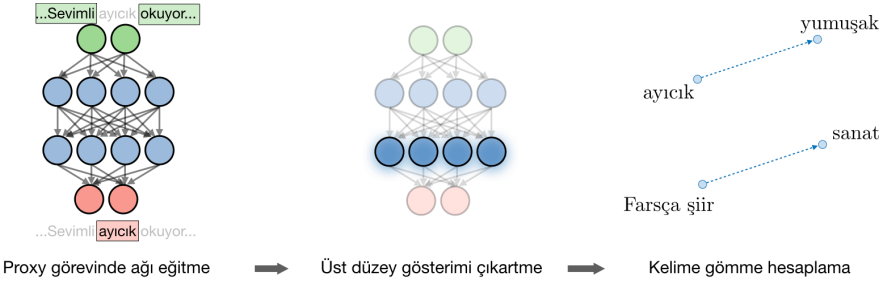
1-hot gösterim	Kelime gömme
	
- o_w not edildi - Naive yaklaşım, benzerlik bilgisi yok	- e_w not edildi - Kelime benzerliği dikkate alınır

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

$$\boxed{e_w = Eo_w}$$

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

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



❒ **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.

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

Not: Softmax bölümünün paydasındaki tüm kelime dağarcığını toplamak, bu modeli hesaplama açısından maliyetli kılar. 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 çıkmasının muhtemel olup olmadığının değerlendirilmesini, modellerin k negatif örnek kümeleri ve 1 pozitif örnek kümesinde eğitilmesini hedefleyen, lojistik regresyon kullanan bir ikili sınıflandırma kümesidir. Bağlam sözcüğü c ve hedef sözcüğü t göz önüne alındığında, tahmin şöyle ifade edilir:

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

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

❒ **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 tekniğidir. Maliyet fonksiyonu J aşağıdaki gibidir:

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

f, $X_{i,j} = 0 \Longrightarrow f(X_{i,j}) = 0$ olacak şekilde bir ağırlıklandırma fonksiyonudur.
Bu modelde e ve θ'nin oynadığı simetri göz önüne alındığında, $e_w^{(\text{final})}$'nin kelime gömmesi şöyle ifade edilir:

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

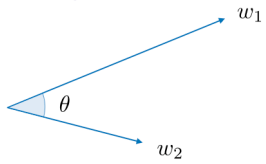
Not: Öğrenilen kelime gömme bileşenlerinin ayrı ayrı bileşenleri tam olarak yorumlanamaz.

Kelimelerin karşılaştırılması

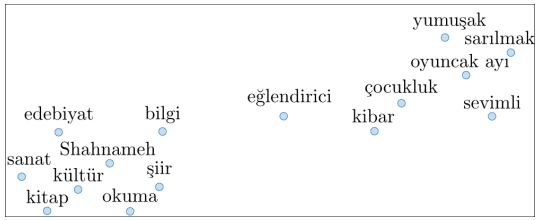
❒ **Kosinüs benzerliği** – w_1 ve w_2 kelimeleri arasındaki kosinüs benzerliği şu şekilde ifade edilir:

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

Not: θ, 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.



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 korpusta ortaya çıkma olasılığını ölçmeyi 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.

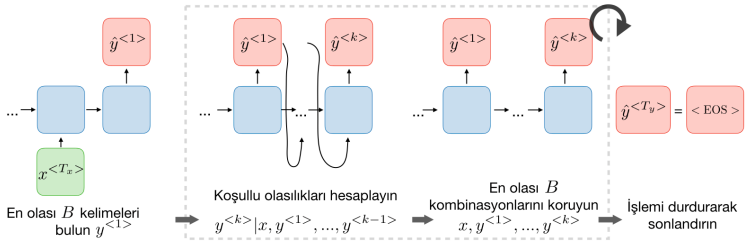
Makine çevirisi

❒ **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:

$$y = \underset{y^{<1>},\ldots,y^{<T_y>}}{\arg\max}\ P(y^{<1>},\ldots,y^{<T_y>}|x)$$

❒ **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 $y^{<k>}|x,y^{<1>},\ldots,y^{<k-1>}$
- Adım 3: En olası B kombinasyonlarını koruyun $x,y^{<1>},\ldots,y^{<k>}$



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

❒ **Işın genişliği** – Işın genişliği B, ışın 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 stabiliteyi arttırmak için, ışın arama genellikle, aşağıdaki gibi tanımlanan normalize edilmiş log-olabilirlik amacı olarak adlandırılan normalize edilmiş hedefe uygulanır:

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

Not: α 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:

	$P(y^*\|x) > P(\hat{y}\|x)$	$P(y^*\|x) \leqslant P(\hat{y}\|x)$
Durum		
Ana neden	Işın arama hatası	RNN hatası
Çözümler	Işın genişliğini artırma	- Farklı mimariyi deneme - Düzenlileştirme - Daha fazla bilgi edinme

❒ **Bleu puanı** – İki dilli değerlendirme alt ölçeği (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:

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

Not: Yapay olarak şişirilmiş bir bleu skorunu önlemek için kısa öngörülen çevirilere küçük bir ceza verilebilir.

Dikkat

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

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

Not: Dikkat skorları, görüntü altyazılama ve makine çevirisinde yaygın olarak 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:

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

Not: hesaplama karmaşıklığı T_x'e göre ikinci derecedendir.

⋆ ⋆ ⋆