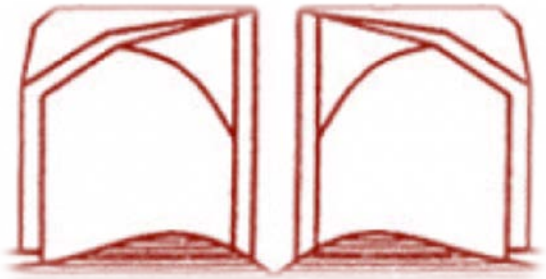


شبکه‌های عصبی مصنوعی

شبکه‌های بازگشتی

هادی ویسی

h.veisi@ut.ac.ir

دانشگاه تهران – دانشکده علوم و فنون نوین

فهرست

معرفی . . .

عقیده کاوی (دسته‌بندی نظرات کاربران)

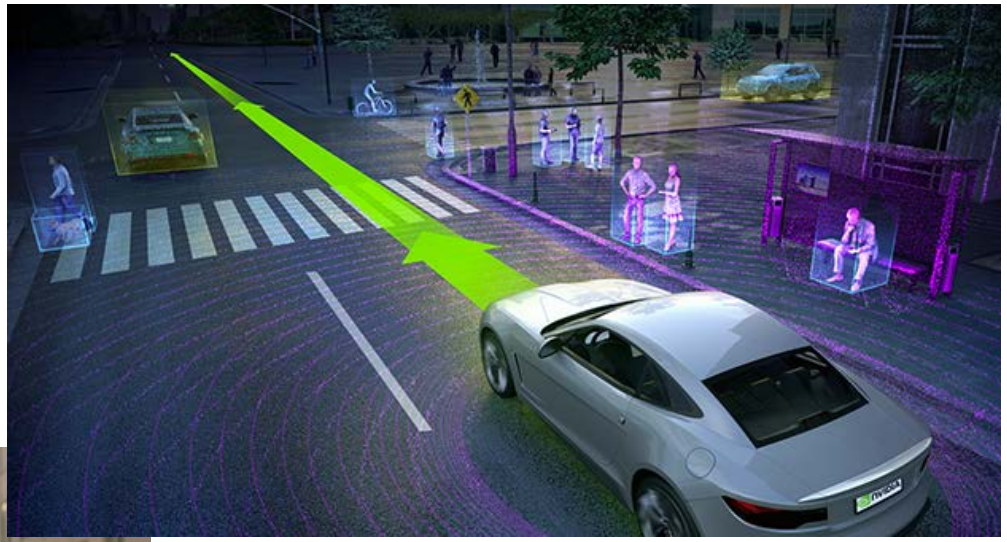
موافق/مخالف

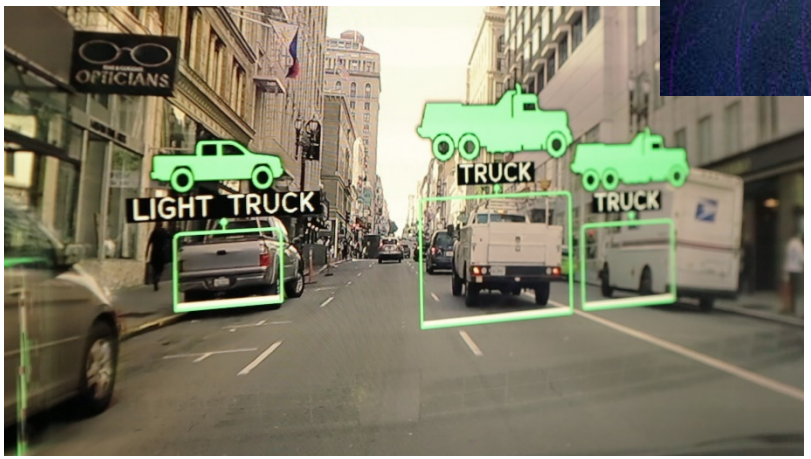
- **امید دلبر** (1393/1/24) : 1
  فوق العادس
- **محمود ابراهیمی جاویدی** (1393/1/22) : 0
  من این لپ تاپ 5 ماه دارم و از همه نظر عالی مثل: کیفیت صفحه نمایش یا کیفیت لولاش برای لمسی بودنش و...
- **مرتضی اکبری** (1393/1/17) : 1
  یکی از مشکلاش ازنظر من کارت گرافیکشه.
  منظورم باس کارت گرافیکه!
- **هادی کریم** (1392/12/22) : 0
  سلام..اصلا تو خریدنش شک نکنید.من خریدم خیلی عالیه.خیلی بیشتر از عکسش خوشگله.واقعا زیباست.اصلا لولایی دیده نمیشه واستحکامش خیلی بیشتره.به قیمتش می ارزه.صدای فنش هم اصلا اذیت نمیکنه.جون میده برای بازی
- **ابوذر هومن** (1392/12/17) : 5
  با سلام
  بعد از 4ماه کار مداوم
  از هر نظر عالی
  قدرت خوبی داره.بسیار سریع
  تاچ خیلی خوبی داره.کیفیت واقعا عالی.صفحه عجیبی داره که اصلا خسته نمیشین موقع کار باهاش
  وزنش هم که عالیه.یه همراه همیشگی میشه براتون
  مرسی
- **هادی کریم** (1392/12/1) : 0
  سلام..اصلا تو خریدنش شک نکنید.من خریدم خیلی عالیه.خیلی بیشتر از عکسش خوشگله.واقعا زیباست.اصلا لولایی دیده نمیشه واستحکامش خیلی بیشتره.به قیمتش می ارزه.صدای فنش هم اصلا اذیت نمیکنه.جون میده برای بازی

نحوه دسته بندی با شبکه عصبی؟

معرفی . . .

- خودروهای خودران!

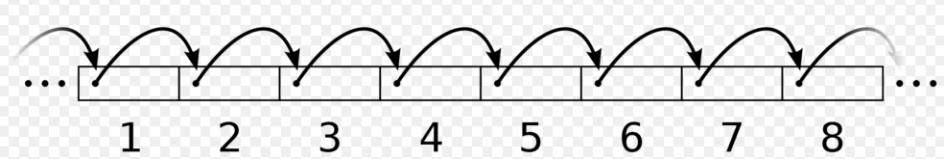




معرفی . . .

- داده متوالی (Sequential Data): داده‌هایی که مقدار فعلی آنها به مقادیر قبلی وابسته است

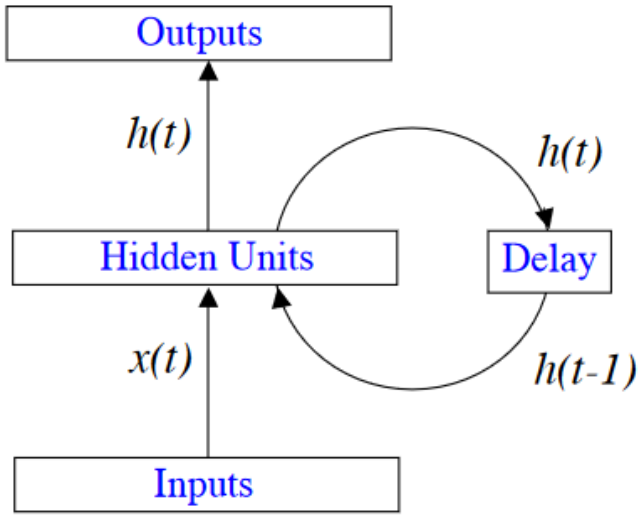
  - فریم‌های (نمونه‌های) سیگنال گفتار
  - فریم‌های (تصاویر) متوالی ویدئو
  - وضعیت آب و هوا
  - قیمت سهام یک شرکت/صنعت
  - دنباله‌های تولید شده توسط گرامرها
    - کلمات داخل یک متن
  - ...



معرفی . . .

- **شبکه‌های عصبی بازگشتی (RNN: Recurrent Neural Networks)**
  - شبکه‌هایی که در ساختار آنها یال‌های بازگشت کننده وجود دارد
  - بر خلاف شبکه‌های عصبی رو به جلو، یال‌ها می‌توانند تشکیل دور بدهند.

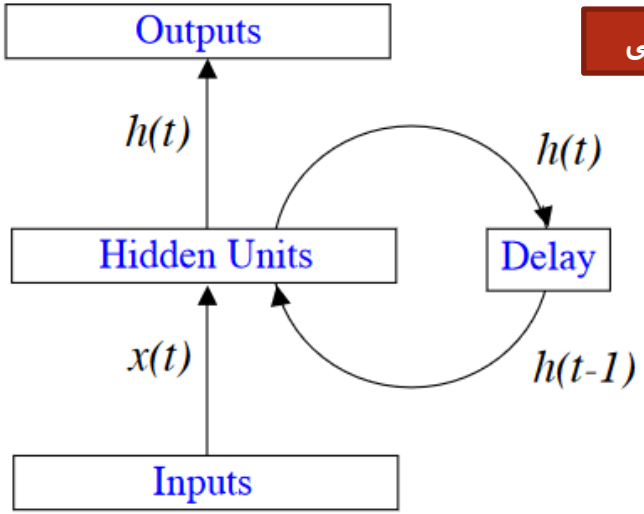
  - به دلیل داشتن یال بازگشتی در ساختار خود، قدرت **حافظه**‌ای دارند.
    - مناسب برای پردازش داده‌های متوالی (Sequential Data)



- ساختار: شبیه MLP با بازگشت از نرون های مخفی

معرفی . . .

- **شبکه عصبی بازگشتی به عنوان یک سیستم پویا (Dynamic System)**
  - **تعریف کامل سیستم با حالت (State) سیستم: مجموعه‌ای مقادیر حاوی همه اطلاعات**
    - مقادیر فعال‌سازی‌های لایه مخفی: $h(t)$
    - مرتبه سیستم پویا = ابعاد فضای حالت
      - در اینجا = تعداد نرون‌های لایه مخفی



فعال سازی لایه مخفی

وزن لایه ورودی به مخفی

وزن لایه مخفی به مخفی

$$h(t) = f_H\left(W_{IH}x(t) + W_{HH}h(t-1)\right)$$

$$y(t) = f_O\left(W_{HO}h(t)\right)$$

فعال سازی لایه خروجی

وزن لایه مخفی به خروجی

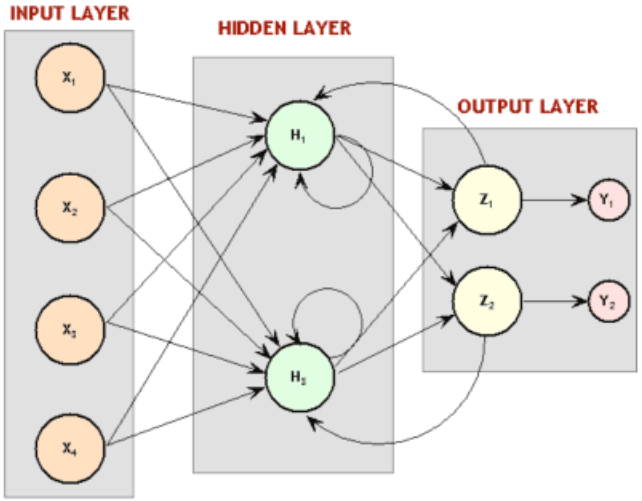
معرفی . . .

- **قضیه**
  - **شبکه عصبی بازگشتی یک تقریب‌زننده جهانی (Universal Approximation) است**

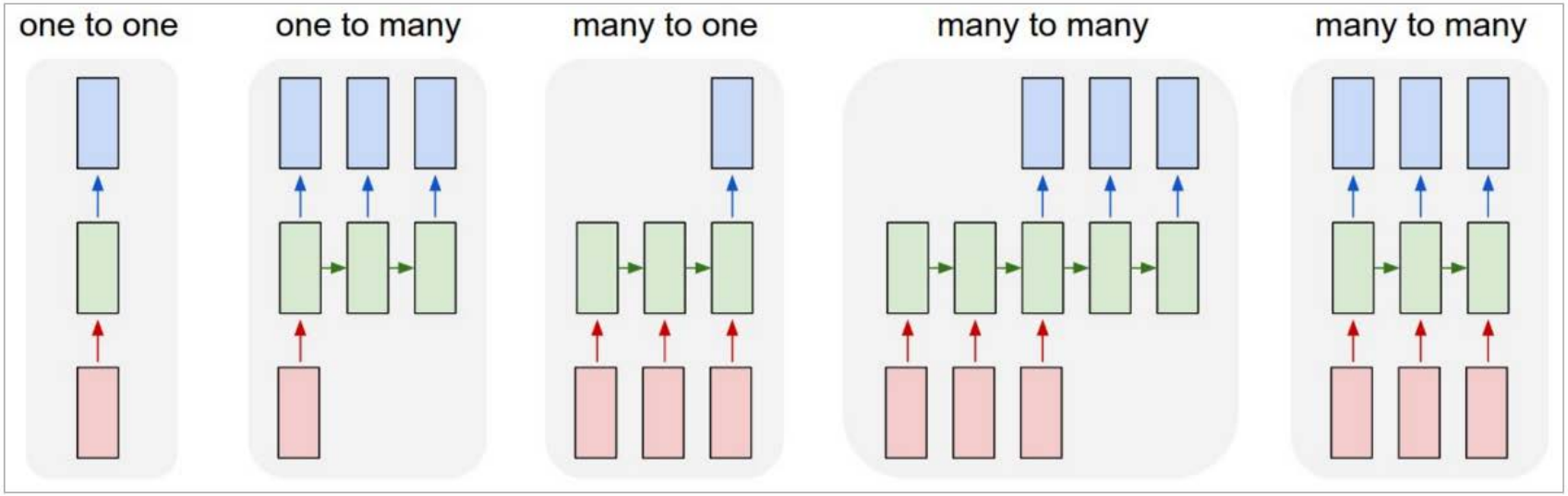
یک سیستم پویای غیرخطی را می توان با هر تقریبی توسط یک شبکه عصبی بازگشتی که دارای تعداد کافی واحد مخفی سیگموییدی دارد، تقریب زد

کاربردهای شبکه عصبی بازگشتی

- مدل‌سازی و پیش‌بینی داده‌های متوالی (Sequential Data) و سری زمانی (Time Series)
  - سیگنال گفتار: تشخیص گفتار
  - دست‌نوشته: تبدیل دست نوشته به متن الکترونیکی
  - امضا: تایید/تشخیص هویت
  - دنباله نویسه‌ها یا واژه‌ها در یک متن: تشخیص موضوع متن
  - تصاویر پشت سر هم (ویدئو): دنبال کردن اشیا
  - پیش‌بینی نرخ ارز/شاخص سهام
  - پیش‌بینی بارش باران در روزهای متوالی سال
  - پیش‌بینی مقدار مصرف آب در روزهای متوالی
  - ...



شبکه عصبی بازگشتی: انواع استفاده



ورودی ثابت و خروجی ثابت
مثال: دسته‌بندی تصویر

ورودی ثابت و خروجی متغیر
مثال: عنوان گذاری تصویر

ورودی متغیر و خروجی ثابت
مثال: تحلیل احساس در متن

ورودی متغیر و خروجی متغیر
مثال: ترجمه ماشینی متن

ورودی متغیر و خروجی متغیر (همزمان)
مثال: تعیین نقش دستوری کلمات در متن

تاریخچه...

- **دهه ۷۰- اولین شبکه عصبی بازگشتی**
  - **۱۹۷۷ -شبکه عصبی کاملا بازگشتی**
    - ارائه شده توسط اندرسون و کوهونن
    - هر نرون به تمامی نرون‌های دیگر متصل است
- **دهه ۸۰**
  - **۱۹۸۲- شبکه عصبی هاپفیلد**
    - ارائه توسط هاپفیلد
    - قانون یادگیری: هب
    - کاربرد: برای تشخیص «شناخته بودن» یا «ناشناس» بودن بردار ورودی استفاده می‌شود
  - **۱۹۸۸- شبکه عصبی حافظه انجمنی دوسویه**
    - نام لاتین: Bidirectional Associative Memory (BAM)
    - توسط Kosko ارائه شد
    - یک شبکه دیگر انجمنی بازگشتی
    - این شبکه شامل دولایه نرون است که کاملا به یکدیگر متصل شده اند

تاریخچه...

- دهه ۹۰
  - ۱۹۹۰- شبکه عصبی المان
    - به این شبکه Simple Recurrent Network (SRN) نیز گفته می‌شود
    - معرفی شده توسط جفری المان
    - شامل ٤ لایه ورودی، پنهان، بافت و خروجی است.
    - دارای اتصالات بازگشتی از لایه پنهان به لایه بافت
  - ۱۹۹۶- شبکه عصبی جردن
    - ارائه شده توسط مایکل جردن
    - شباهت بسیار زیادی به شبکه المان دارد
    - دارای اتصالات بازگشتی از لایه خروجی به لایه بافت
  - ۱۹۹۷- شبکه عصبی بازگشتی دوطرفه
    - نام لاتین شبکه Bidirectional Recurrent Neural Network است
    - ارائه توسط توسط شوستر و پالیوال
    - شامل دو لایه پنهان مجزا
    - مزیت: مقدار هدف در هر گام زمانی به کل دنباله ورودی وابسته است

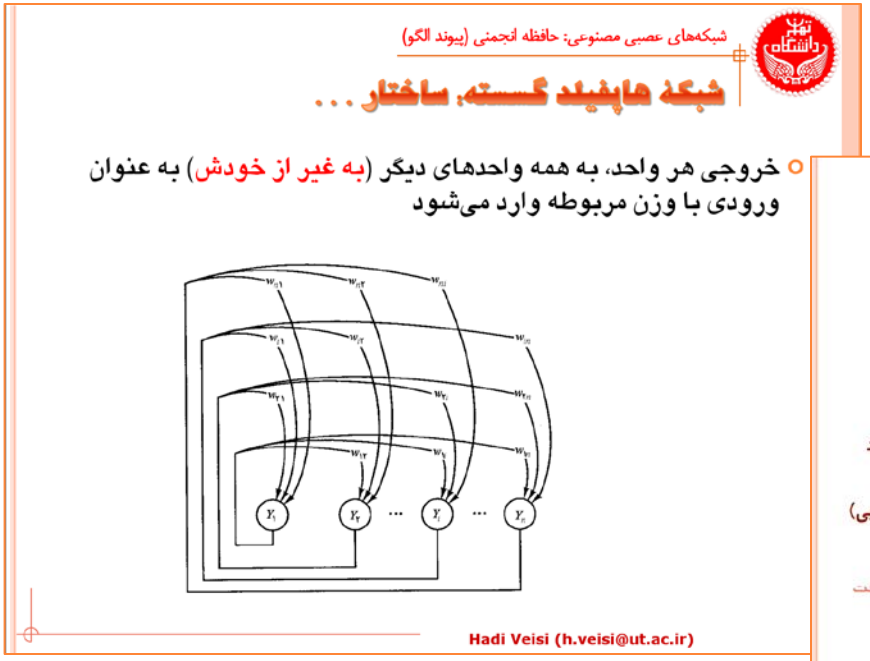
تاریخچه...

- **۱۹۹۷- شبکه عصبی حافظه کوتاه مدت ماندگار**
  - نام لاتین: Long Short Term Memory (LSTM)
  - توسط هاکریتر و اشمیدبر معرفی شد
  - نرون‌های لایه پنهان با بلوک‌های حافظه جایگزین شدند
    - حل شدن مشکل فراموشی دنباله‌های طولانی
- **۲۰۰۰ به بعد**
  - **۲۰۰۱- توسعه LSTM**
    - توسط فلیکس گرز توسعه داده شد
      - اضافه کردن دروازه فراموشی به بلوک حافظه

شبکه هاپفیلد

- در لینک زیر دریافت کنید

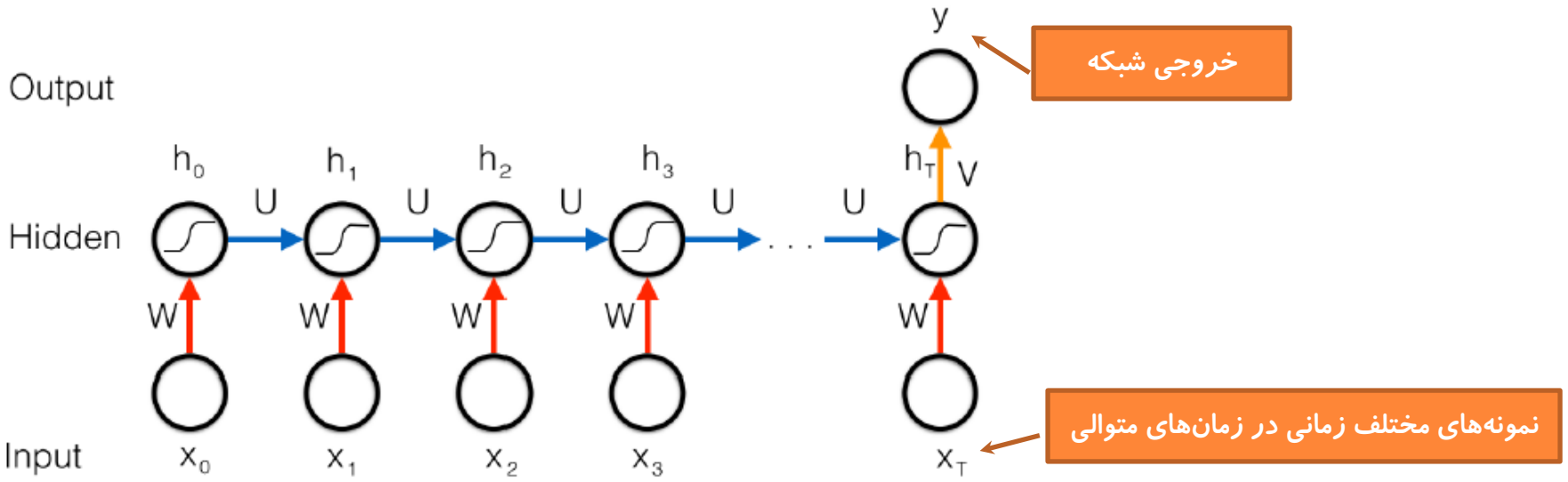
- http://dsp.ut.ac.ir/en/wp-content/uploads/2016/04/ANN-Lecture5-Associative.pdf



آموزش شبکه ...

عملکرد شبکه

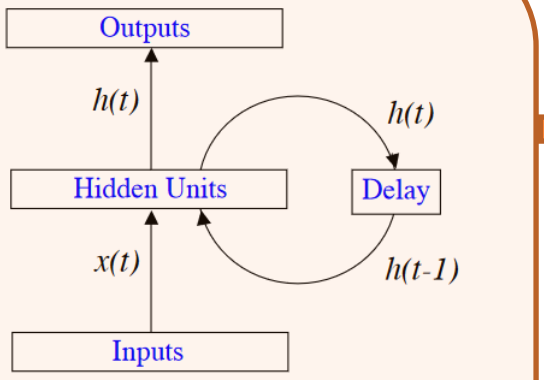
- وابستگی خروجی شبکه به خروجی‌های لایه مخفی به ازای همه ورودی‌ها



$$f(x) = Vh_T$$
$$h_t = \sigma(Uh_{t-1} + Wx_t), \text{ for } t = T, .., 1$$
$$\dots$$
$$h_0 = \sigma(Wx_0)$$

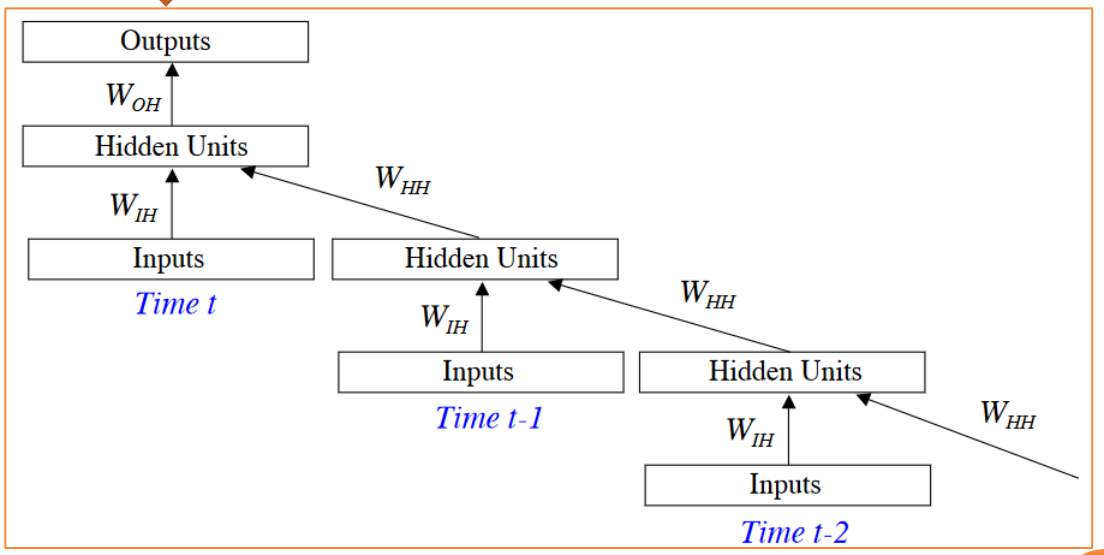
آموزش شبکه ...

- استفاده از الگوریتم پس‌انتشار . . .
  - محاسبه خطای خروجی و استفاده از گرادیان برای تخمین وزن‌ها
  - تبدیل شبکه بازگشتی به شبکه جلوسو
  - باز کردن در زمان (Unfolding over Time)



$$h(t) = f_H\left(W_{IH}x(t) + W_{HH}h(t-1)\right)$$

$$y(t) = f_O\left(W_{HO}h(t)\right)$$



آموزش شبکه ...

- استفاده از الگوریتم پس‌انتشار . . .
  - Backpropagation Through Time (BPTT)
  - محاسبه خطای شبکه برای همه نمونه‌های بین دو زمان شروع t_0 و پایان t_1

$$E_{total}(t_0,t_1)=\sum_{t=t_0}^{t_1} E_{sse/ce}(t)$$

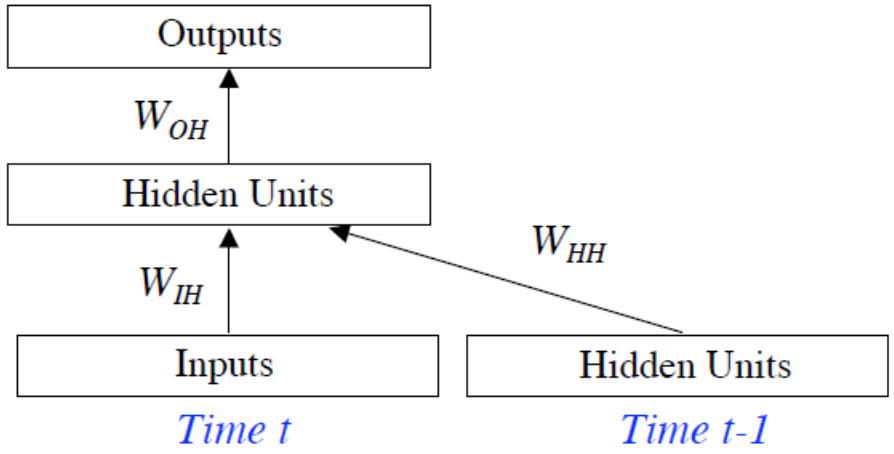
  - گرادیان وزن‌های شبکه برای بدست آوردن مقدار تغییرات وزن

$$\Delta w_{ij}=-\eta\frac{\partial E_{total}(t_0,t_1)}{\partial w_{ij}}=-\eta\sum_{t=t_0}^{t_1}\frac{\partial E_{sse/ce}(t)}{\partial w_{ij}}$$

$$w_{ij}\in\{W_{IH},\ W_{HH}\}$$

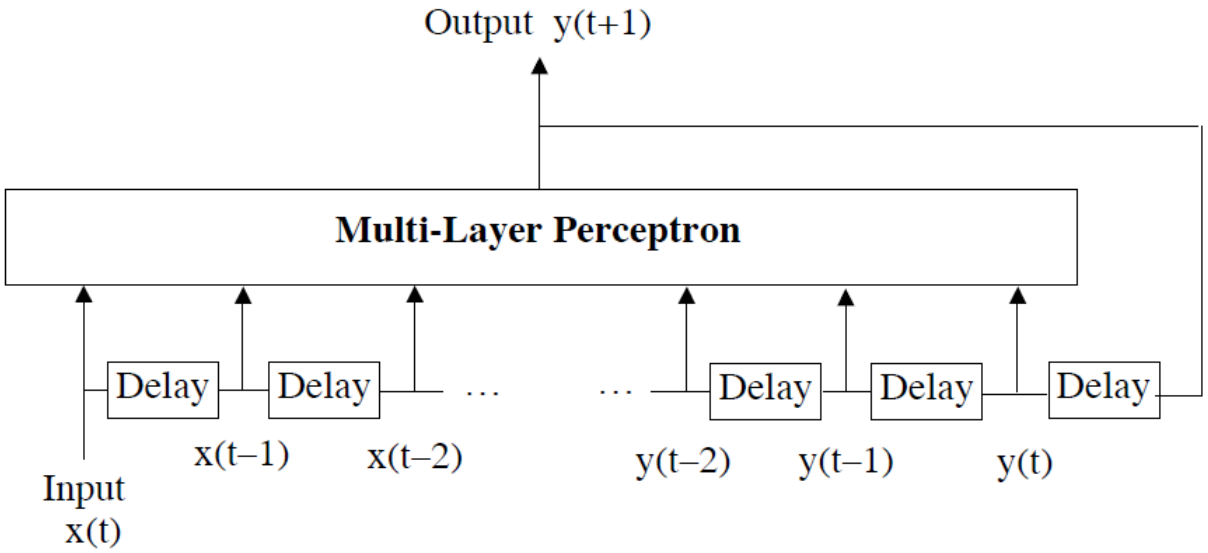
آموزش شبکه ...

- **استفاده از الگوریتم پس‌انتشار**
  - استفاده از این روش نیازمند نگهداری حالت‌های قبلی شبکه (خروجی لایه مخفی) و کلیه ورودی‌های قبلی
  - در عمل نگهداری همه اطلاعات قبلی مشکل است و تنها از تعداد محدودی از آنها (مثلا ۳۰ مقدار قبلی) استفاده می‌شود = truncation
  - حالت ساده = نگهداری فقط یک مرحله قبل = شبکه المان (Elman Network)



آموزش شبکه ...

- مدل NARX: Non-linear Auto-Regressive with eXogeneous inputs
  - یک شبکه MLP با یک ورودی و یک خروجی
  - ورودی‌ها و خروجی‌ها در زمان‌های مختلف تاخیر داده می‌شوند

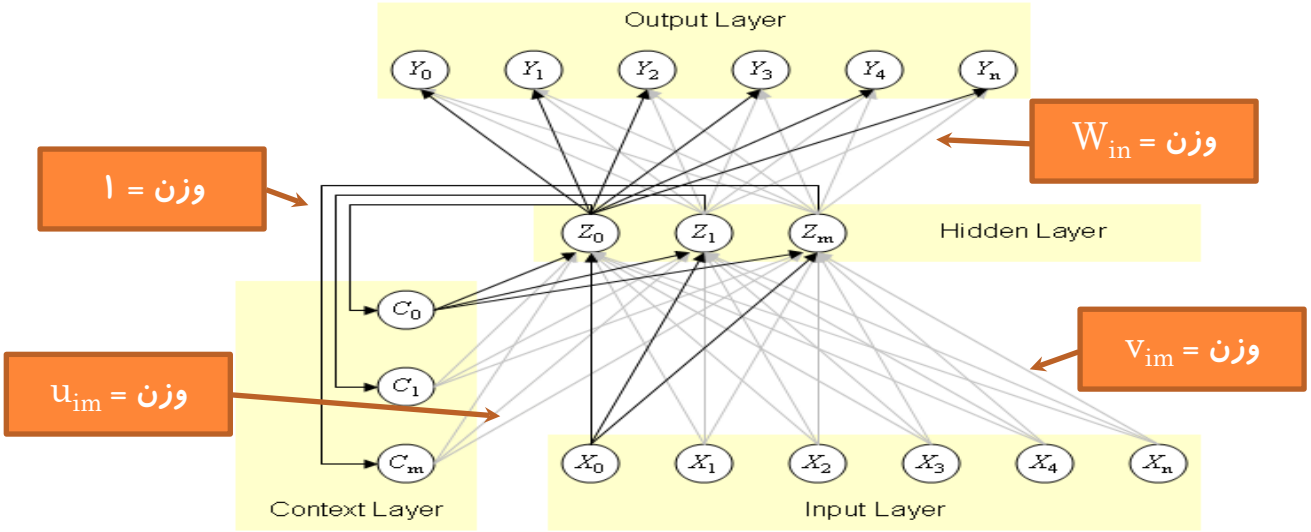


- این ساختار برای پیش‌بینی سری‌های زمانی مناسب است

شبکه عصبی المان ...

ساختار

- دارای چهار لایه ورودی، مخفی، بافت و خروجی
- لایه بافت
  - نرون‌های لایه بافت یک کپی از فعال‌سازهای نرون‌های لایه پنهان را دریافت می‌کنند
  - اتصالات بازگشتی لایه بافت به لایه پنهان، یک حافظه کوتاه‌مدت را برای شبکه ایجاد می‌کند
  - تعداد نرون‌های لایه بافت با تعداد نرون‌های لایه پنهان برابر است
  - وزن یال‌هایی که لایه پنهان را به لایه بافت متصل می‌کنند برابر مقدار ثابت یک می‌باشد



شبکه عصبی المان: آموزش ...

- به ابتدا و انتهای هر دنباله، نماد آغازین و نماد پایانی اضافه کنید
- الگوریتم آموزش همان پس انتشار استاندارد است

- مرحله ۰: برای هر دنباله آموزش $X = (x_1,x_i,...,x_k)$ که x_1 و x_k به‌ترتیب نمادهای مربوط به شروع و خاتمه دنباله می‌باشد، مراحل ۱ تا ۱۴ را انجام دهید
- مرحله ۱: مقدار فعال‌ساز تمامی نرون‌های لایه بافت (C_i) را برابر ۰.۵ قرار دهید.
- مرحله ۲: مراحل ۳ تا ۱۴ را انجام دهید تا به انتهای دنباله برسید
- مرحله ۳: بردار x_i را به شبکه ارائه دهید.
- مرحله ۴: بردار x_{i+1} در دنباله را به عنوان پاسخ هدف به واحدهای خروجی ارائه دهید.
- مرحله ۵: مقدار فعال‌ساز نرون‌های لایه مخفی را محاسبه کنید

$$z\_in_q = \sum_{i=1}^{n} x_i\, v_{iq} + \sum_{i=1}^{m} C_i\, u_{iq} \quad , \quad Z_q = f(z\_in_q)$$

شبکه عصبی المان: آموزش ...

- مرحله ۶: مقدار فعال‌ساز نرون‌های لایه خروجی را محاسبه کنید

$$y\_in_j = \sum_{i=1}^{m} Z_i\, w_{ij} \quad , \quad y_j = f(y\_in_j)$$

- مرحله ۷: دلتای نرون‌های لایه خروجی را محاسبه کنید

$$\delta_j = (t_j - y_j)\, f'(y\_in_j)$$

- مرحله ۸: تغییرات وزن نرون‌های لایه پنهان به لایه خروجی را محاسبه کنید

$$\Delta W_{ij} = \alpha\, \delta_j\, Z_i$$

- مرحله ۹: خطای ورودی به نرون‌های لایه مخفی را محاسبه کنید (پس انتشار خطا)

$$\delta\_in_q = \sum_{i=1}^{n} \delta_i\, w_{qi}$$

- مرحله ۱۰: دلتای نرون‌های لایه پنهان را محاسبه کنید

$$\delta_q = \delta\_in_q\, f'(z\_in_q)$$

- مرحله ۱۱: تغییرات وزن نرون‌های لایه ورودی به لایه پنهان را محاسبه کنید

$$\Delta V_{iq} = \alpha\, x_i \delta_q$$

Hadi Veisi (h.veisi@ut.ac.ir)

شبکه عصبی المان: آموزش

- **مرحله ۱۲: تغییرات وزن نرون‌های لایه بافت به لایه پنهان را محاسبه کنید**

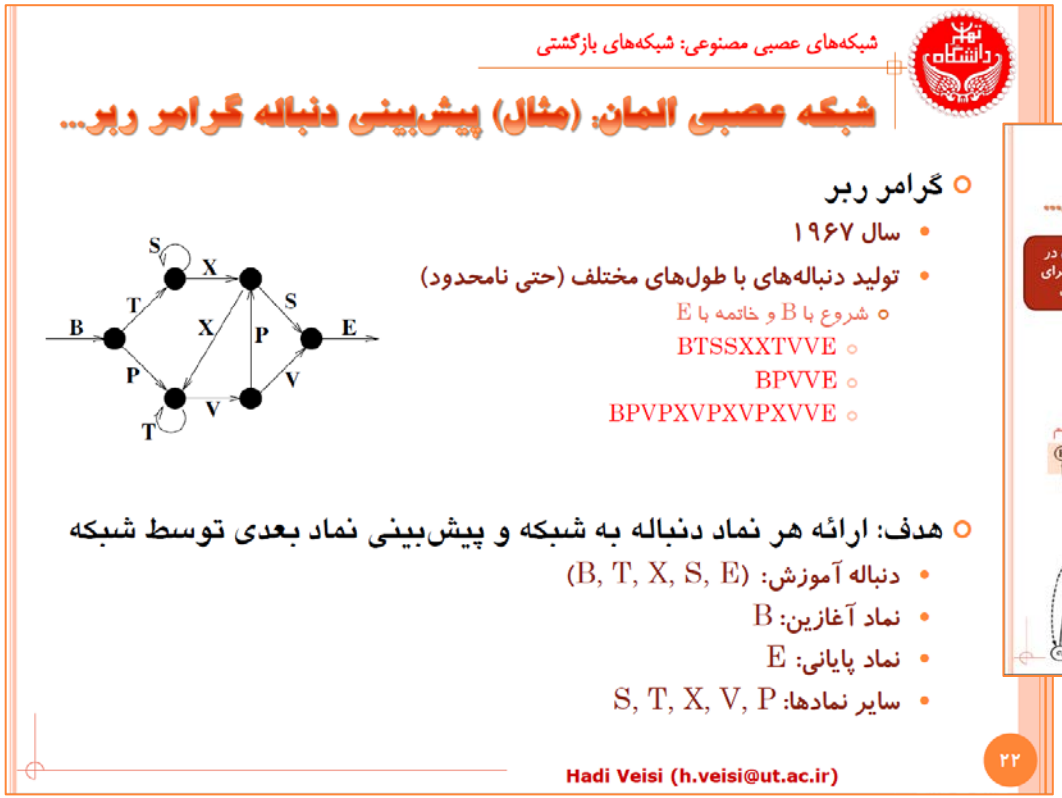
$$\Delta U_{iq} = \alpha\, C_i\, \delta_q$$

- **مرحله ۱۳: کلیه وزن‌ها را بروز رسانی کنید**
- **مرحله ۱۴: شرط توقف را بررسی کنید:**
  - اگر هدف نماد پایانی باشد، الگوریتم را خاتمه دهید، در غیر این صورت فعال‌ساز نرون‌های لایه مخفی را در نرون‌های لایه بافت کپی کنید.

شبکه عصبی المان: (مثال)

- در لینک زیر ببینید

- http://dsp.ut.ac.ir/en/wp-content/uploads/2017/06/ANN-Lecture6-RNN.pdf



شبکه عصبی المان: الگوریتم کاربرد

- الگوریتم کاربرد (آزمون)
- مرحله ۰: برای هر دنباله آزمون $X = (x_1, x_i, \ldots, x_k)$ مراحل ۱ تا ۱۴ را انجام دهید
- مرحله ۱: مقدار فعال‌ساز تمامی نرون‌های لایه بافت را برابر ۰.۵ قرار دهید.
- مرحله ۲: برای تمامی بردارهای x_i مراحل ۳ تا ۶ را انجام دهید
- مرحله ۳: بردار x_i را به شبکه ارائه دهید.
- مرحله ۴: مقدار فعال‌ساز نرون‌های لایه مخفی را محاسبه کنید

$$z\_in_m = \sum_{i=1}^{n} x_i\, v_{i\,m} + \sum_{i=1}^{m} C_i\, u_{i\,m} \quad , \quad Z_m = f(z\_in_m)$$

- مرحله ۵: مقدار فعال‌ساز نرون‌های لایه خروجی را محاسبه کنید

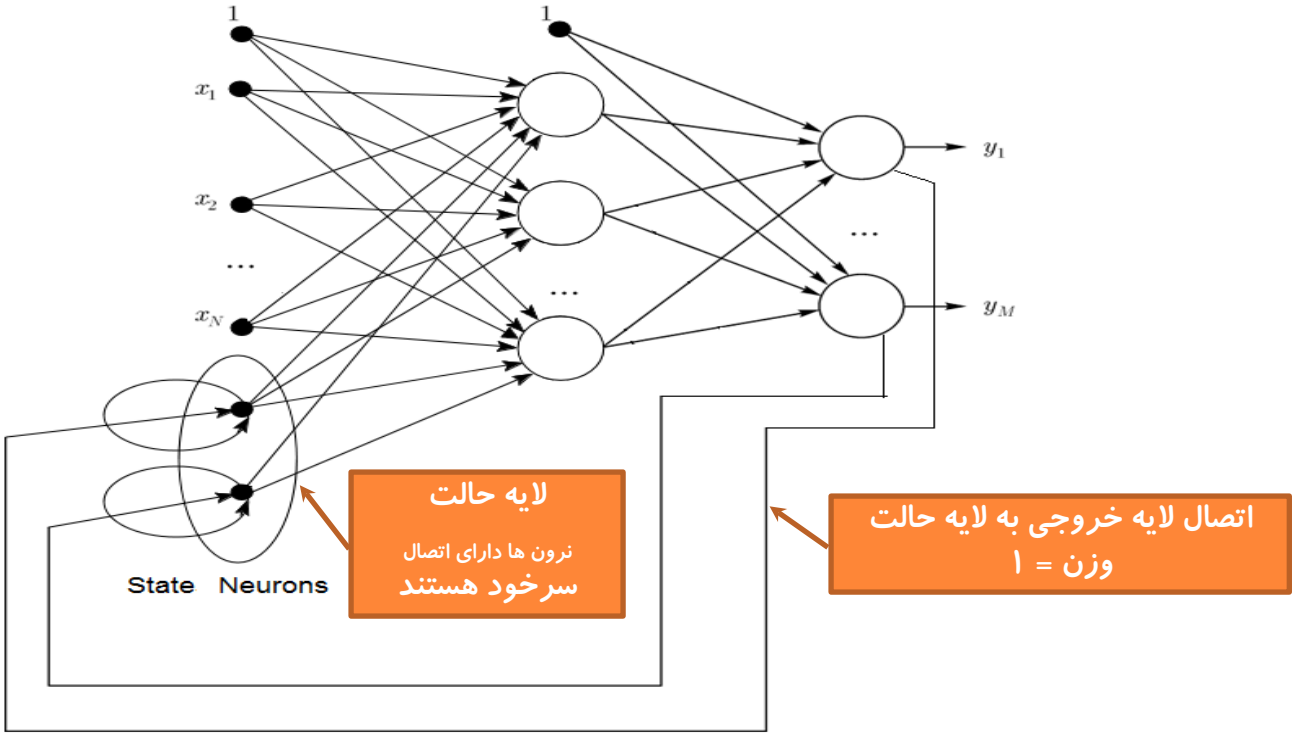
$$y\_in_n = \sum_{i=1}^{m} Z_i\, w_{i\,n} \quad , \quad Y_n = f(y\_in_n)$$

- مرحله ۶: نماد با بیشترین مقدار فعال‌ساز را به عنوان خروجی شبکه اعلام کنید

شبکه عصبی جردن ...

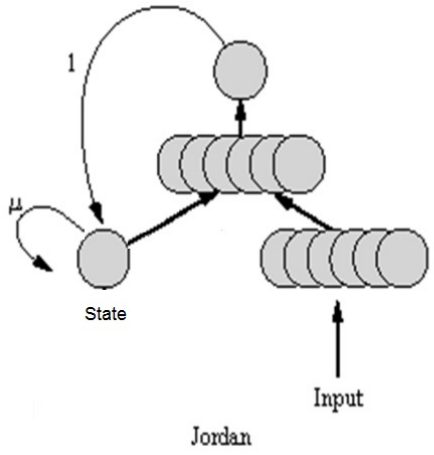
معرفی و ساختار

- ارائه شده در سال ۱۹۹۶ توسط مایکل جردن
- دارای شباهت بسیار زیاد به شبکه عصبی المان
- شبکه دارای اتصالات بازگشتی از لایه خروجی به لایه حالت و همچنین از لایه حالت به خودش می‌باشد

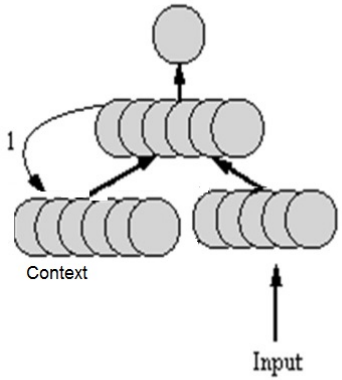


شبکه عصبی جردن: تفاوت با شبکه المان

- در شبکه جردن
  - اتصالات بازگشتی به جای لایه پنهان از لایه خروجی شروع می‌شود (با وزن ثابت یک)
  - در این شبکه لایه بافت، لایه حالت (State Layer) نامیده می‌شود
  - لایه حالت شامل اتصالات بازگشتی از خودش به خودش با وزن ثابت می‌باشد
  - تعداد نرون‌های لایه حالت با تعداد نرون‌های لایه خروجی برابر است
    - نرون‌های لایه حالت یک کپی از فعال‌سازهای نرون‌های لایه خروجی را دریافت می‌کنند.



Jordan

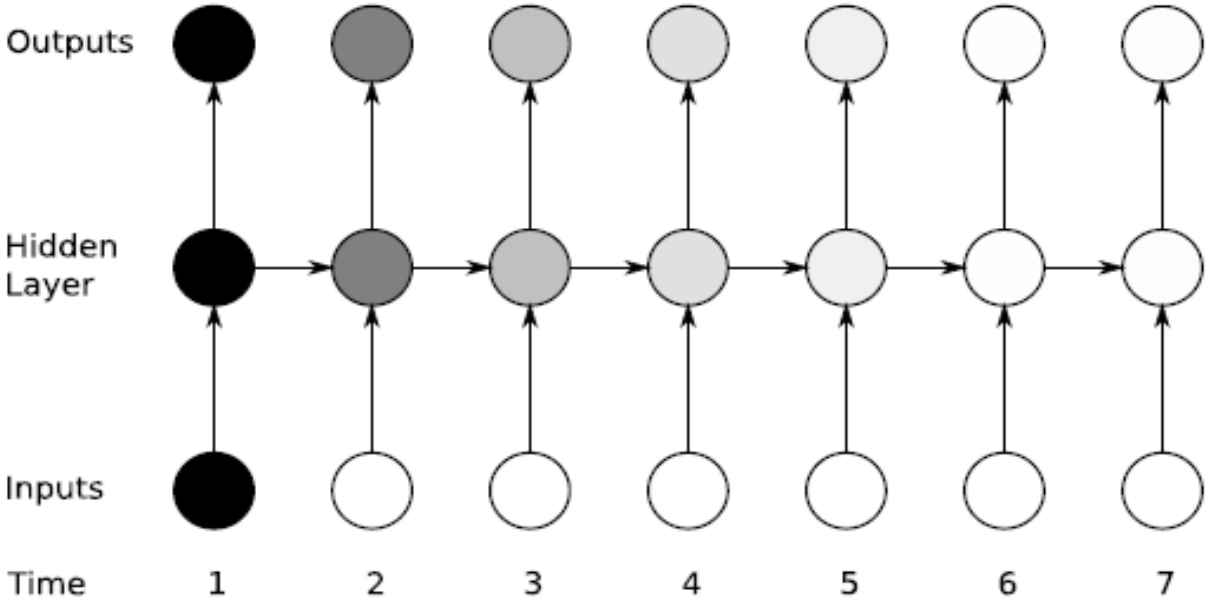


Elman

مشکل فراموشی در شبکه‌های عصبی بازگشتی . . .

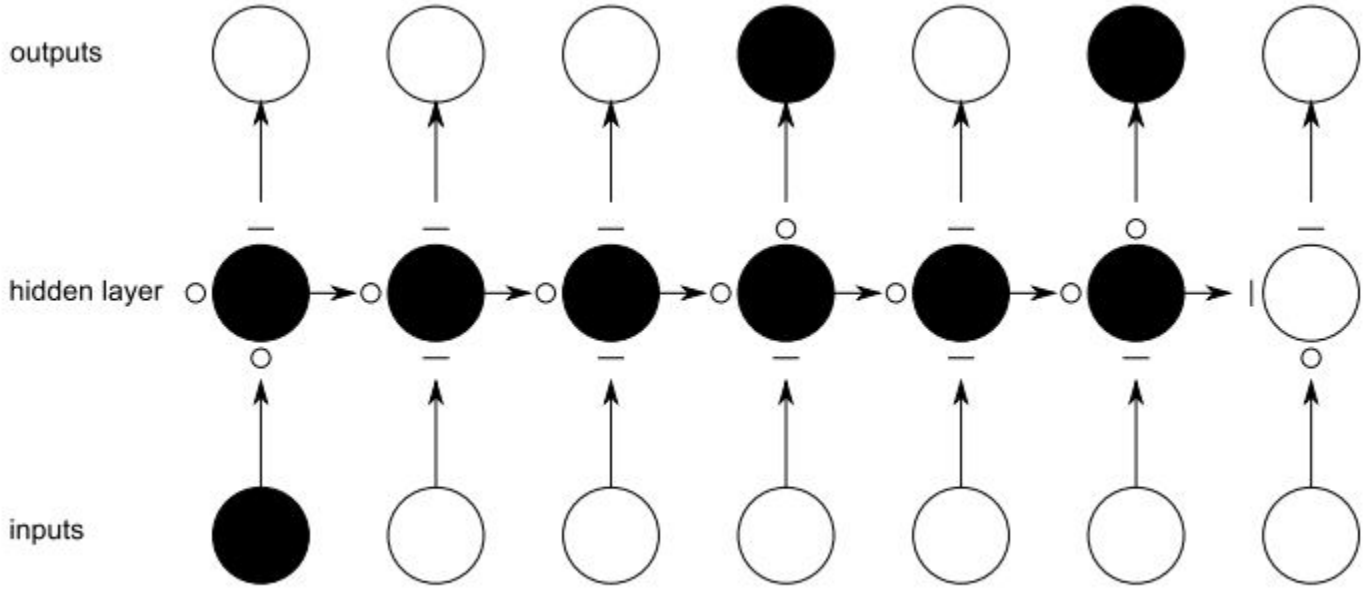
مشکل

- با طولانی شدن دنباله ورودی، شبکه عصبی بازگشتی به‌مرور داده‌های اولیه را فراموش می‌کند که به آن مشکل فراموشی گفته می‌شود
  - سایه‌های پررنگ‌تر به معنای تاثیر بیشتر بر لایه پنهان و خروجی می‌باشد



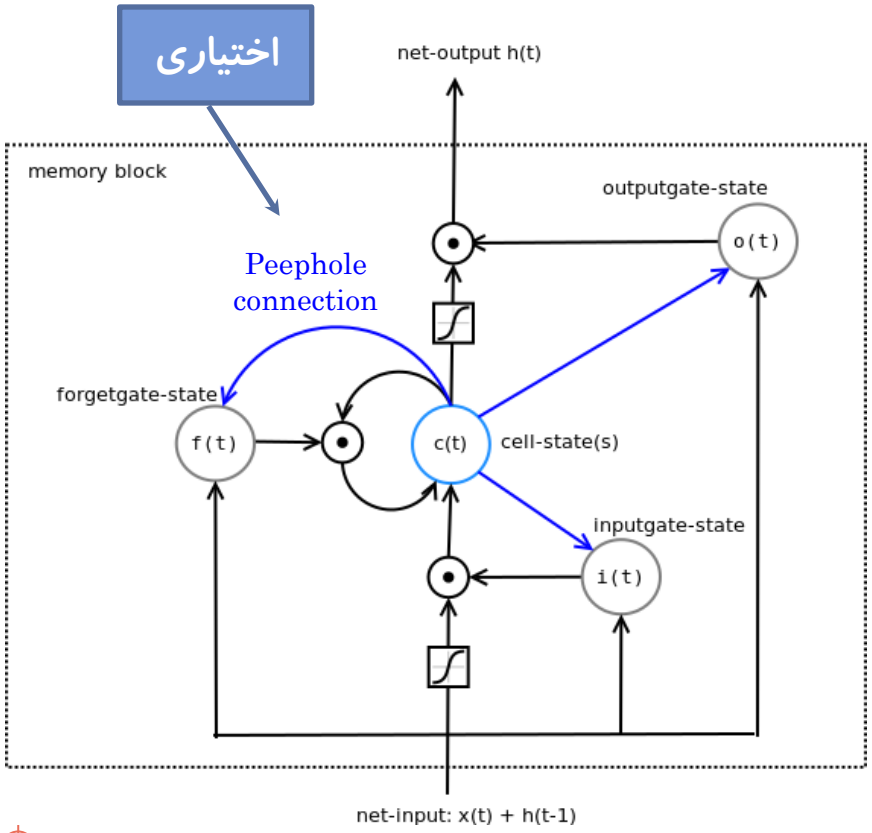
مشکل فراموشی در شبکه‌های عصبی بازگشتی

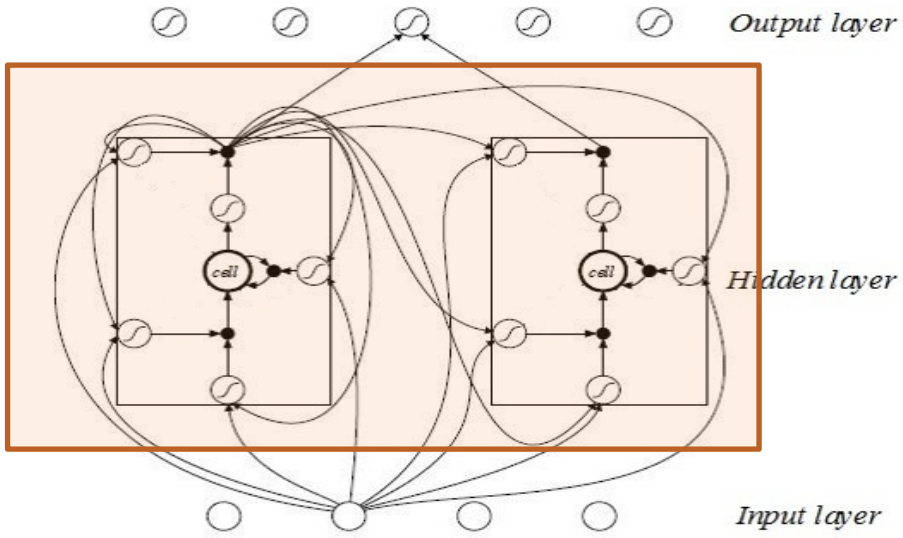
- حل مشکل فراموش
  - کنترل ورود و خروج داده در واحدهای لایه میانی شبکه



شبکه عصبی حافظه کوتاه مدت ماندگار (LSTM) ...

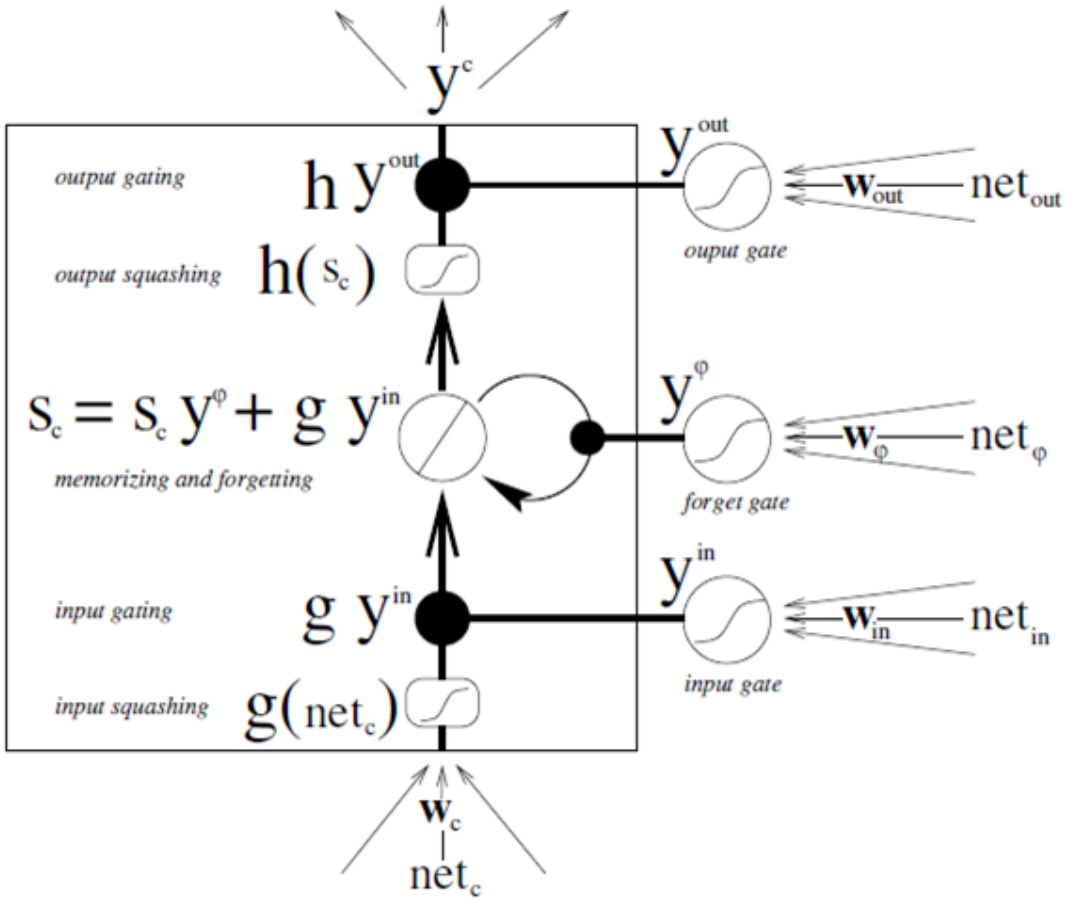
- شبکه Long Short Term Memory (LSTM)
  - نرون‌های لایه پنهان با بلوک‌های حافظه جایگزین شدند
    - حل شدن مشکل فراموشی دنباله‌های طولانی





شبکه عصبی LSTM: ساختار بلوک حافظه ...

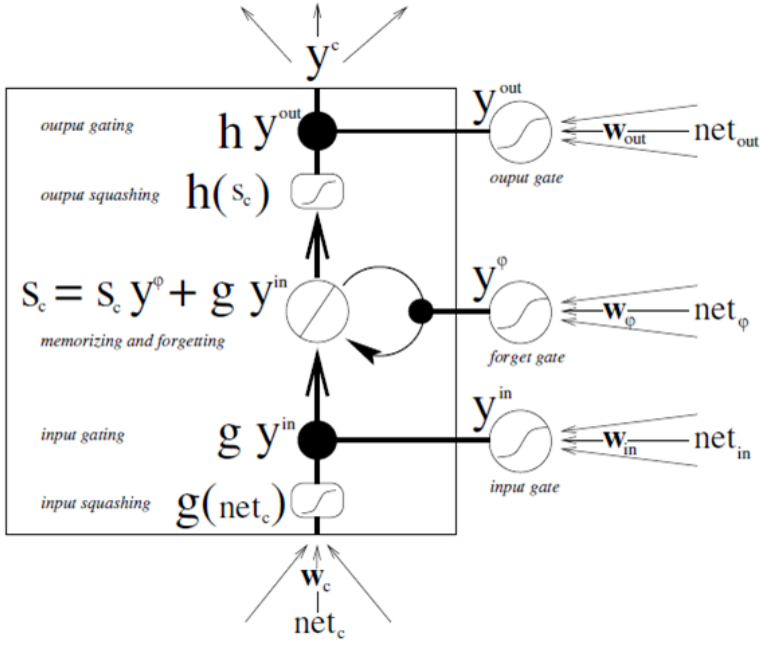
- هر بلوک حافظه، شامل
  - سلول
    - برای ذخیره اطلاعات در بلوک
  - دروازه ورودی
  - دروازه فراموشی
  - دروازه خروجی



- دروازه‌ها وظیفه کنترل ورود و خروج داده‌ها و پاک کردن حافظه بلوک را برعهده دارند

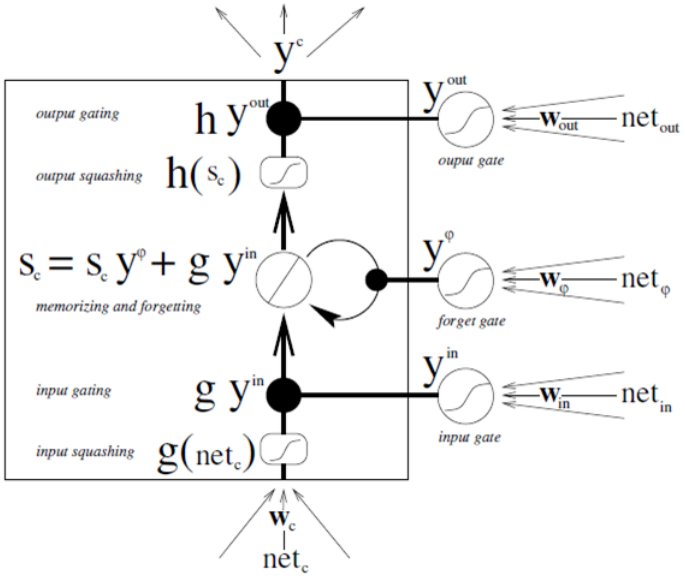
شبکه عصبی LSTM: ساختار بلوک حافظه . . .

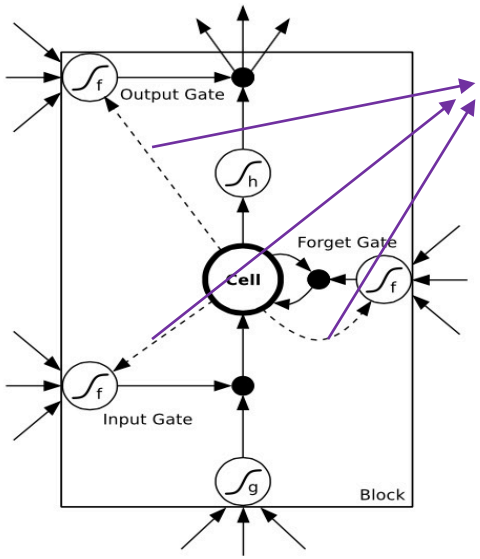
- وظیفه دروازه‌ها: کنترل ورود و خروج داده‌ها و پاک کردن حافظه‌ی بلوک
  - فعال‌ساز دروازه‌ها مقداری بین صفر و یک می‌گیرند.
  - در صورتی که دروازه کاملا باز باشد فعال‌ساز آن برابر یک و در صورتی‌که کاملا بسته باشد فعال‌ساز آن برابر صفر است
- هر سلول حافظه در مرکز خود یک واحد به نام CEC دارد که به فعال‌سازی آن **حالت سلول**، S_c می‌گویند



شبکه عصبی LSTM: ساختار بلوک حافظه

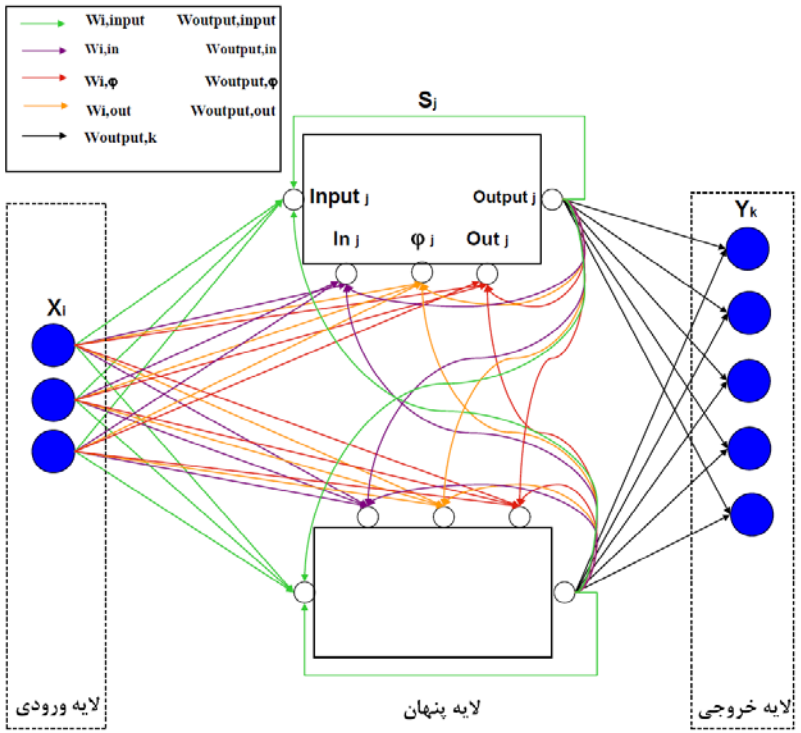
- اتصالات روزنه (peephole)
  - این اتصالات دلخواه هستند (optional) و حالت سلول، S_c، را به دروازه‌ها متصل می‌سازند
  - در این اسلایدها از این اتصالات صرفنظر شده است





شبکه عصبی LSTM: اتصالات (وزن ها)

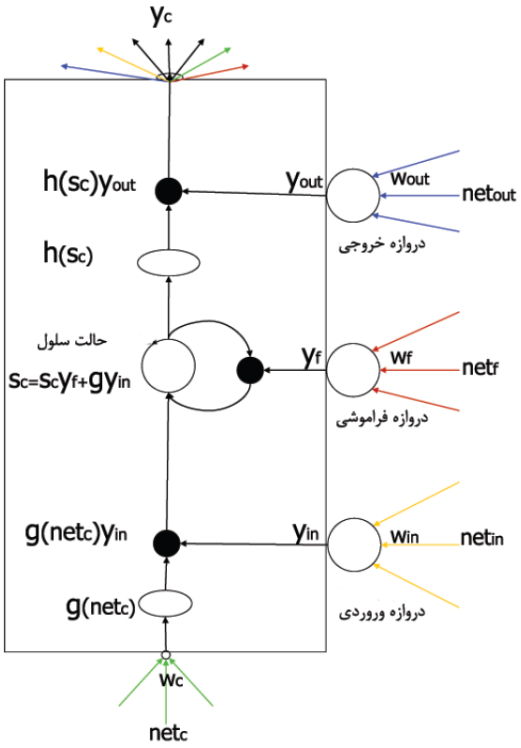
- از ورودی
  - وزن بین لایه ورودی و بلوک حافظه
  - وزن بین لایه ورودی و دروازه ورودی
  - وزن بین لایه ورودی و دروازه فراموشی
  - وزن بین لایه ورودی و دروازه خروجی
- از خروجی بلوک (سلول حافظه)
  - وزن بین خروجی بلوک‌ها و ورودی بلوک‌ها
  - وزن بین خروجی بلوک‌ها و دروازه ورودی
  - وزن بین خروجی بلوک‌ها و دروازه فراموشی
  - وزن بین خروجی بلوک‌ها و دروازه خروجی
  - وزن بین خروجی‌های بلوک‌ها و لایه خروجی



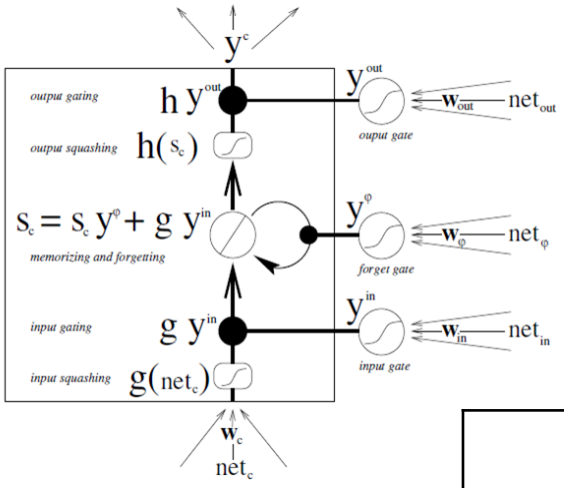
شبکه عصبی LSTM: آموزش ...

نمادها

مقدار خالص ورودی به سلول j	$net_{c_j}(t)$
مقدار خالص ورودی به دروازه ورودی سلول j	$net_{in_j}(t)$
مقدار خالص ورودی به دروازه فراموشی سلول j	$net_{\varphi_j}(t)$
مقدار خالص ورودی به دروازه خروجی سلول j	$net_{out_j}(t)$
حالت سلول j	$s_{c_j}(t)$
فعال‌ساز دروازه ورودی سلول j	$y^{in_j}(t)$
فعال‌ساز دروازه خروجی سلول j	$y^{out_j}(t)$
فعال‌ساز دروازه فراموشی سلول j	$y^{\varphi_j}(t)$
فعال‌ساز خروجی سلول j	$y^{c_j}(t)$
مقدار خالص ورودی به نرون خروجی شماره k	$net_k(t)$
فعال‌ساز نرون خروجی شماره k	$y^k(t)$
یال اتصالی از واحد m به واحد l (اندیس کلی وزن‌ها)	$w_{l\,m}$
خطای دروازه خروجی سلولj	$\delta_{out_j}(t)$
خطای نرون خروجی شماره k	$\delta_k(t)$
خطای حالت سلول j	$e_{s_{c_j}}(t)$
نرخ یادگیری	$0 \le \alpha \le 1$

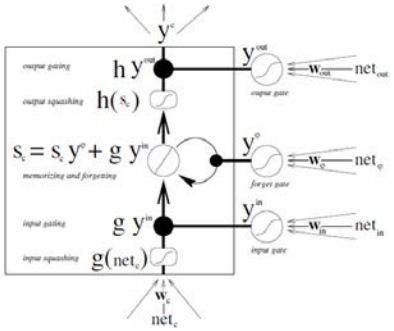


شبکه عصبی LSTM: آموزش ...



نمادها

فعال‌ساز $f(x)$	$f(x) = \frac{1}{1+e^{-x}}$
فعال‌ساز $h(x)$	$h(x) = \frac{2}{1+e^{-x}} - 1$
فعال‌ساز $g(x)$	$g(x) = \frac{4}{1+e^{-x}} - 2$
$y^m(t-1)$	به فعال‌ساز واحد m در گام زمانی قبل اشاره می‌کند. در صورتی که واحد m معادل نرون i از لایه ورودی باشد این مقدار معادل ورودی iاُم شبکه (یعنی x_i) در مرحله زمانی فعلی خواهد بود.



شبکه عصبی LSTM: آموزش ...

- برای هر دنباله آموزش $X^T = (x_1, x_2, \ldots, x_{T-1}, x_T)$ مراحل ۱ تا ۱۵ را انجام دهید.
- مرحله ۰: انتخاب وزن‌های اولیه به‌صورت تصادفی
- مرحله ۱: مقدار اولیه حالت سلول‌ها، S_c، را برابر صفر قرار دهید
- مرحله ۲: مقدار خالص همه ورودی های ممکن به دروازه‌های ورودی و فعال‌سازهای آن‌ها را محاسبه کنید

$$net_{in_j}(t) = \sum_m w_{in_j m}\ y^m(t-1) \quad , y^{in_j}(t) = f(net_{in_j}(t))$$

فعال سازی بین صفر و یک

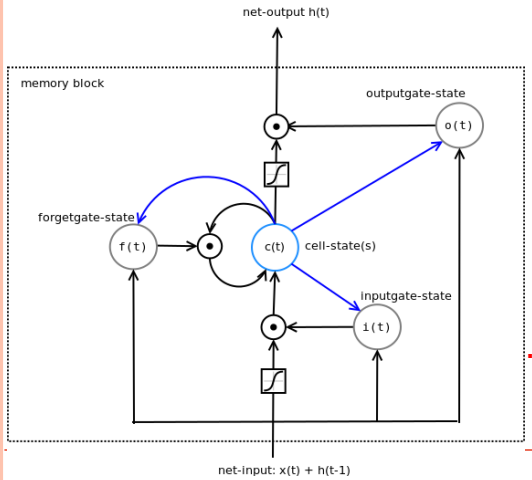
- net_{in_j} معادل جمع وزندار کلیه ورودی‌های ممکن به دروازه ورودی j اُمین بلوک حافظه می‌باشد که این ورودی‌ها عبارتنداز:

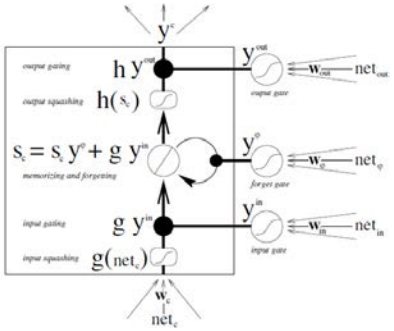
1. نرون‌های لایه ورودی

که در این حالت $y^m(t-1)$ معادل بردار X خواهد بود.

1. خروجی مرحله قبل کلیه بلوک‌های شبکه از جمله خود بلوک j

که در این حالت $y^m(t-1)$ معادل بردار $y^c(t-1)$ خواهد بود.





شبکه عصبی LSTM: آموزش ...

- **مرحله ۳: مقدار خالص همه ورودی های ممکن به دروازه‌های فراموشی و فعال‌سازهای آن‌ها را محاسبه کنید**

$$net_{\varphi_j}(t) = \sum_m w_{\varphi_j m}\, y^m(t-1) \quad , y^{\varphi_j}(t) = f(net_{\varphi_j}(t))$$

فعال سازی بین صفر و یک

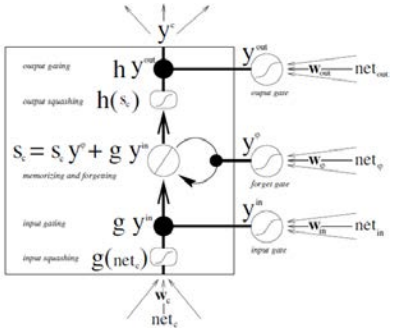
  - net_{φ_j} معادل جمع وزندار کلیه ورودی‌های ممکن به دروازه فراموشی j اُمین بلوک حافظه می‌باشد که این ورودی‌ها شامل:

1. نرون‌های لایه ورودی: که در این حالت $y^m(t-1)$ معادل بردار X خواهد بود.
2. خروجی مرحله قبل کلیه بلوک های شبکه از جمله خود بلوک j: که در این حالت $y^m(t-1)$ معادل بردار $y^c(t-1)$ خواهد بود.

- **مرحله ۴ : مقدار خالص ورودی به سلول‌ها و همچنین حالت سلول‌ها را محاسبه کنید**

$$net_{c_j}(t) = \sum_m w_{c_j m}\, y^m(t-1)$$

$$s_{c_j}(t) = s_{c_j}(t-1) y^{\varphi_j}(t) + y^{in_j}(t)\, g\left(net_{c_j}(t)\right) \;;\; for\ t > 0$$



شبکه عصبی LSTM: آموزش ...

- **مرحله ۴ : مقدار خالص همه ورودی های ممکن به سلول‌ها و همچنین حالت سلول‌ها را محاسبه کنید**

$$net_{c_j}(t) = \sum_m w_{c_j m}\, y^m(t-1)$$

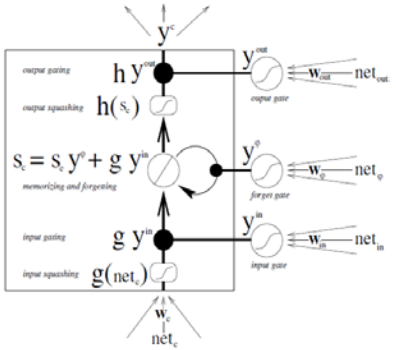
- net_{c_j} معادل جمع وزندار کلیه ورودی‌های ممکن به j اُمین بلوک حافظه می‌باشد که این ورودی‌ها عبارتنداز:

1. نرون‌های لایه ورودی: که در این حالت $y^m(t-1)$ معادل بردار X خواهد بود.
2. خروجی مرحله قبل کلیه بلوک های شبکه از جمله خود بلوک j: که در این حالت $y^m(t-1)$ معادل بردار $y^C(t-1)$ خواهد بود.

$$s_{c_j}(t) = s_{c_j}(t-1)y^{\varphi_j}(t) + y^{in_j}(t)\, g\left(net_{c_j}(t)\right)\ ;\ for\ t > 0$$

فعال سازی بین ۲+ و ۲-

شبکه عصبی LSTM: آموزش ...



- **مرحله ۵: مقدار خالص همه ورودی های ممکن به دروازه‌های خروجی و فعال‌سازهای آن‌ها را محاسبه کنید**

$$net_{out_j}(t) = \sum_m w_{out_j\,m}\, y^m(t-1) \quad , \quad y^{out_j}(t) = f\left(net_{out_j}(t)\right)$$

فعال سازی بین صفر و یک

  - net_{out_j} معادل جمع وزندار کلیه ورودی‌های ممکن به دروازه خروجی j اُمین بلوک حافظه می‌باشد که این ورودی‌ها شامل:

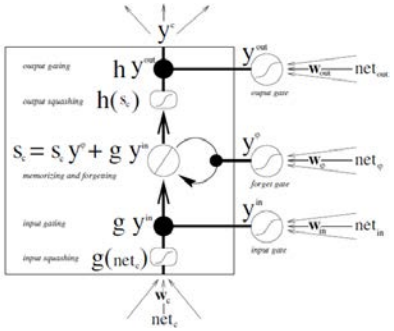
1. نرون‌های لایه ورودی: که در این حالت $y^m(t-1)$ معادل بردار X خواهد بود.
2. خروجی مرحله قبل کلیه بلوک های شبکه از جمله خود بلوک j: که در این حالت $y^m(t-1)$ معادل بردار $y^c(t-1)$ خواهد بود.

است

- **مرحله ۶: خروجی سلول‌ها را محاسبه کنید**

$$y^{c_j}(t) = y^{out_j}(t)\, h(s_{c_j}(t))$$

فعال سازی بین ۱+ و ۱−



شبکه عصبی LSTM: آموزش ...

- مرحله ۷: مقدار خالص ورودی به نرون‌های لایه خروجی و فعال‌سازهای آن‌ها را محاسبه کنید

$$net_k(t) = \sum_m w_{km} y^m(t) \quad , y^k(t) = f\left(net_k(t)\right)$$

- مرحله ۸: خطای نرون‌های لایه خروجی را محاسبه کنید

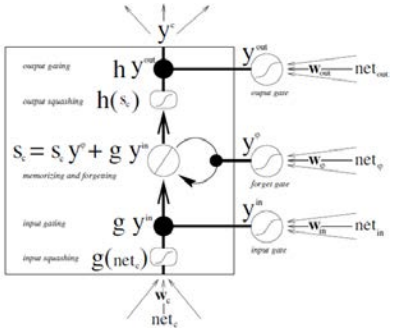
$$\delta_k(t) = f_k'\left(net_k(t)\right)\, e_k(t) \quad , \quad e_k(t) := t^k(t) - y^k(t)$$

- مرحله ۹: خطای دروازه‌های خروجی را محاسبه کنید

$$\delta_{out_j}(t) = f'_{out_j}(net_{out_j} t))\, h(s_{c_j}(t)) \left(\sum_K w_{k\, c_j}\, \delta_k(t) \right)$$

وزن یال اتصالی از خروجی بلوک j به نرون k از لایه خروجی

جمع وزن‌دار خطاهای رسیده از لایه خروجی به بلوک j اُم حافظه



شبکه عصبی LSTM: آموزش ...

- مرحله ۱۰: خطای حالت سلول‌ها را به کمک رابطه زیر محاسبه کنید

$$e_{s_{c_j}}(t) = y^{out_j}(t)\, h'(s_{c_j}(t)) \left(\sum_k w_{k\ c_j}\, \delta_k(t) \right)$$

- مرحله ۱۱: برای محاسبه تغییرات وزنی یال‌های متصل به ورودی سلول‌ها، دروازه‌های ورودی و دروازه‌های فراموشی رُندهای زیر را محاسبه کنید

مقدار اولیه کلیه رُندها را برابر صفر قرار می‌دهیم

$$\frac{\partial s_{c_j}(t=0)}{\partial w_{lm}} = 0 \quad ; for\ l \in \{\varphi, in, c_j\}$$

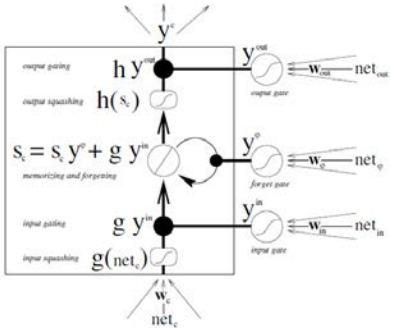
$$\frac{\partial s_{c_j}(t)}{\partial w_{c_j\,m}} = \frac{\partial s_{c_j}(t-1)}{\partial w_{c_j\,m}}\, y^{\varphi_j}(t) + g'\left(net_{c_j}(t)\right) y^{in_j}(t)\ y^m(t-1)$$

$$\frac{\partial s_{c_j}(t)}{\partial w_{in_j\,m}} = \frac{\partial s_{c_j}(t-1)}{\partial w_{in_j\,m}} y^{\varphi_j}(t) + g\left(net_{c_j}(t)\right) f'_{in_j}\left(net_{in_j}(t)\right) y^m(t-1)$$

$$\frac{\partial s_{c_j}(t)}{\partial w_{\varphi_j\,m}} = \frac{\partial s_{c_j}(t-1)}{\partial w_{\varphi_j\,m}} y^{\varphi_j}(t) + s_{c_j}(t-1)\, f'_{\varphi_j}\left(net_{\varphi_j}(t)\right) y^m(t-1)$$

- مرحله ۱۲: تغییرات وزنی یال‌های متصل به لایه خروجی (خروجی سلول به نرون‌های خروجی) را محاسبه کنید

$$\Delta w_{k\,c_j}(t) = \alpha\, \delta_k(t)\, y^{c_j}(t)$$



شبکه عصبی LSTM: آموزش ...

- مرحله ۱۳: تغییرات وزنی یال‌های متصل (وارده) به دروازه‌های ورودی و دروازه‌های فراموشی را محاسبه کنید

$$\Delta w_{l\,m}(t) = \alpha\, e_{s_{c_j}}(t)\, \frac{\partial s_{c_j}(t)}{\partial w_{l\,m}} \quad for\ l \in \{\varphi, in\}$$

- مرحله ۱۴: تغییرات وزنی یال‌های متصل به سلول‌ها (لایه ورودی به سلول) را محاسبه کنید

$$\Delta w_{c_j\ m}(t) = \alpha\, e_{s_{c_j}}(t)\, \frac{\partial s_{c_j}(t)}{\partial w_{c_j\,m}}$$

- مرحله ۱۵: تغییرات وزنی یال‌های متصل (وارده) به دروازه‌های خروجی را محاسبه کنید

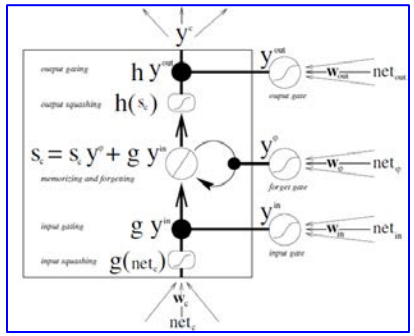
$$\Delta w_{out_j\,m}(t) = \alpha\, \delta_{out_j}(t)\, y^m(t-1)$$

شبکه عصبی LSTM: آموزش

نکات کاربردی مهم

- در کلیه روابط، $y^m(t-1)$ به کلیه منابع ورودی به سلول در گام زمانی قبل اشاره می‌کند
  - اگر $y^m(t-1)$ به نرون‌های لایه ورودی اشاره کند آن را برابر مقدار نرون‌های لایه ورودی در مرحله زمانی فعلی قرار می‌دهیم.
- در انتهای هر دنباله آموزش، مقدار حالت و خروجی تمامی سلول‌ها و همچنین تمامی رُندها را برابر صفر قرار می‌دهیم
- به‌روز رسانی وزن‌ها در انتهای هر دنباله آموزشی صورت می‌گیرد.
  - مجموع تغییرات وزن‌ها به ازای همه دنباله ها به وزن‌های قبلی اضافه می شود
- بهتر است وزن‌های اولیه به‌صورت تصادفی و در بازه ۰.۱- تا ۰.۱ انتخاب شود.
- مقدار نرخ یادگیری را کوچک بگیرید، مثلا ۰.۰۱ یا ۰.۰۰۱

Hadi Veisi (h.veisi@ut.ac.ir)



شبکه عصبی LSTM: کاربرد ...

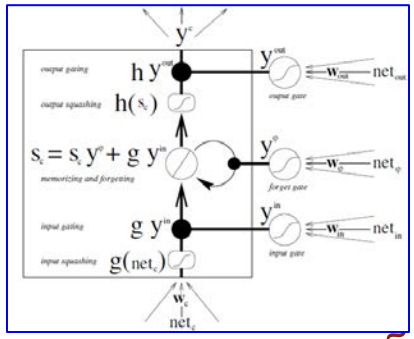
- مرحله ۰: برای هر دنباله آزمون $X^T = (x_1, x_2, \dots, x_{T-1}, x_T)$ مراحل ۱ تا ۷ را انجام دهید
- مرحله ۱: مقدار اولیه حالت تمامی سلول‌ها را برابر صفر قرار دهید
- مرحله ۲: برای تمامی بردارهای x_i مراحل ۳ تا ۸ را انجام دهید
- مرحله ۳: مقدار خالص ورودی به دروازه‌های ورودی و فعال‌سازهای آن‌ها را محاسبه کنید

$$net_{in_j}(t) = \sum_m w_{in_j m}\ y^m(t-1) \quad , y^{in_j}(t) = f(net_{in_j}(t))$$

- net_{in_j} معادل جمع وزندار کلیه ورودی‌های ممکن به دروازه ورودی j اُمین بلوک حافظه می‌باشد که این ورودی‌ها شامل:

1. نرون‌های لایه ورودی: که در این حالت $y^m(t-1)$ معادل بردار X خواهد بود.
2. خروجی مرحله قبل کلیه بلوک های شبکه از جمله خود بلوک j: که در این حالت $y^m(t-1)$ معادل بردار $y^c(t-1)$ خواهد بود.

است



شبکه عصبی LSTM: کاربرد ...

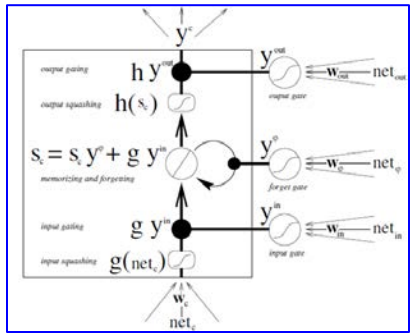
- **مرحله ۴: مقدار خالص ورودی به دروازه‌های فراموشی و فعال‌سازهای آن‌ها را محاسبه کنید**

$$net_{\varphi_j}(t) = \sum_m w_{\varphi_j m}\, y^m(t-1) \quad , y^{\varphi_j}(t) = f(net_{\varphi_j}(t))$$

  - net_{φ_j} معادل جمع وزندار کلیه ورودی‌های ممکن به دروازه فراموشی j اُمین بلوک حافظه می‌باشد که این ورودی‌ها شامل:

1. نرون‌های لایه ورودی: که در این حالت $y^m(t-1)$ معادل بردار X خواهد بود.
2. خروجی مرحله قبل کلیه بلوک های شبکه از جمله خود بلوک j: که در این حالت $y^m(t-1)$ معادل بردار $y^c(t-1)$ خواهد بود.

است



شبکه عصبی LSTM: کاربرد ...

- مرحله ۵ : مقدار خالص ورودی به سلول‌ها و همچنین حالت سلول‌ها را محاسبه کنید

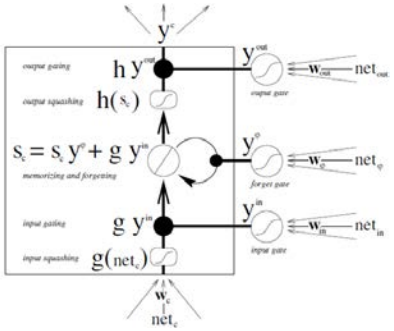
$$net_{c_j}(t) = \sum_m w_{c_j m}\, y^m(t-1)$$

- net_{c_j} معادل جمع وزندار کلیه ورودی‌های ممکن به j اُمین بلوک حافظه می‌باشد که این ورودی‌ها شامل:

1. نرون‌های لایه ورودی: که در این حالت $y^m(t-1)$ معادل بردار X خواهد بود.
2. خروجی مرحله قبل کلیه بلوک های شبکه از جمله خود بلوک j: که در این حالت $y^m(t-1)$ معادل بردار $y^c(t-1)$ خواهد بود.

است

$$s_{c_j}(t) = s_{c_j}(t-1)y^{\varphi_j}(t) + y^{in_j}(t)\, g\left(net_{c_j}(t)\right)\ ;\ for\ t > 0$$



شبکه عصبی LSTM: کاربرد ...

- **مرحله ۶: مقدار خالص ورودی به دروازه‌های خروجی و فعال‌سازهای آن‌ها را محاسبه کنید**

$$net_{out_j}(t) = \sum_m w_{out_j m}\, y^m(t-1) \quad , \quad y^{out_j}(t) = f(net_{out_j}(t))$$

- net_{out_j} معادل جمع وزندار کلیه ورودی‌های ممکن به دروازه خروجی j اُمین بلوک حافظه می‌باشد که این ورودی‌ها شامل:

1. نرون‌های لایه ورودی: که در این حالت $y^m(t-1)$ معادل بردار X خواهد بود.
2. خروجی مرحله قبل کلیه بلوک های شبکه از جمله خود بلوک j: که در این حالت $y^m(t-1)$ معادل بردار $y^c(t-1)$ خواهد بود.

است

- **مرحله ۷: خروجی سلول‌ها را محاسبه کنید**

$$y^{c_j}(t) = y^{out_j}(t)\, h(s_{c_j}(t))$$

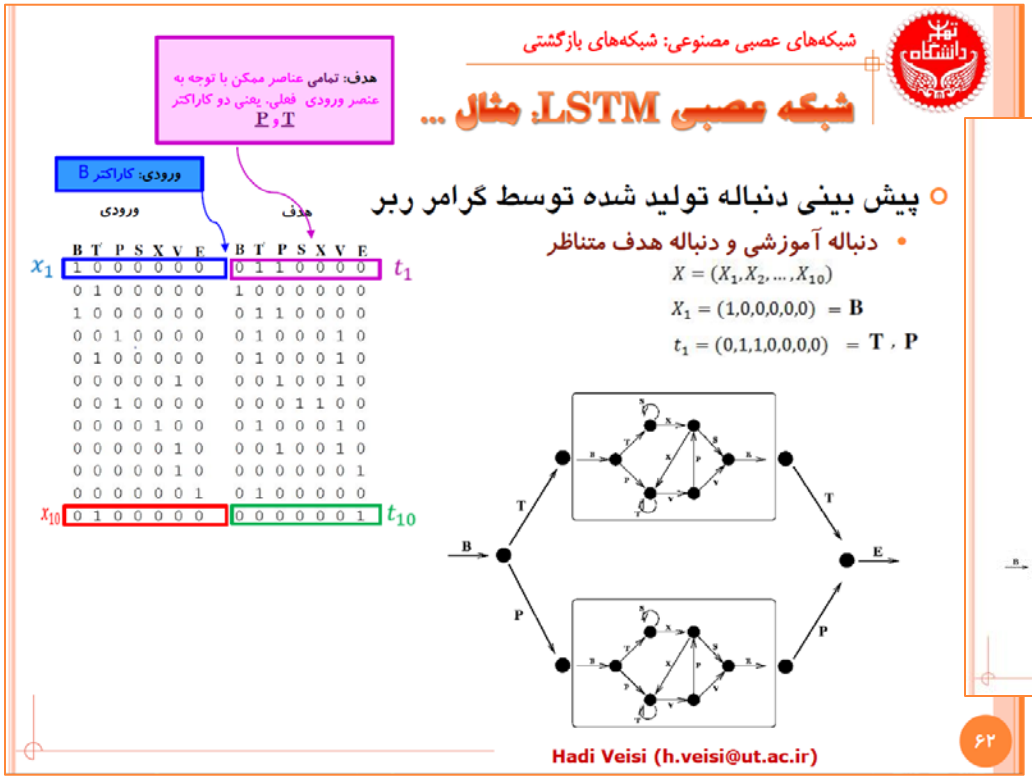
- **مرحله ۸: مقدار خالص ورودی به نرون‌های لایه خروجی و فعال‌سازهای آن‌ها را محاسبه کنید**

$$net_k(t) = \sum_m w_{km} y^m(t) \quad , y^k(t) = f\big(net_k(t)\big)$$

شبکه عصبی LSTM: مثال

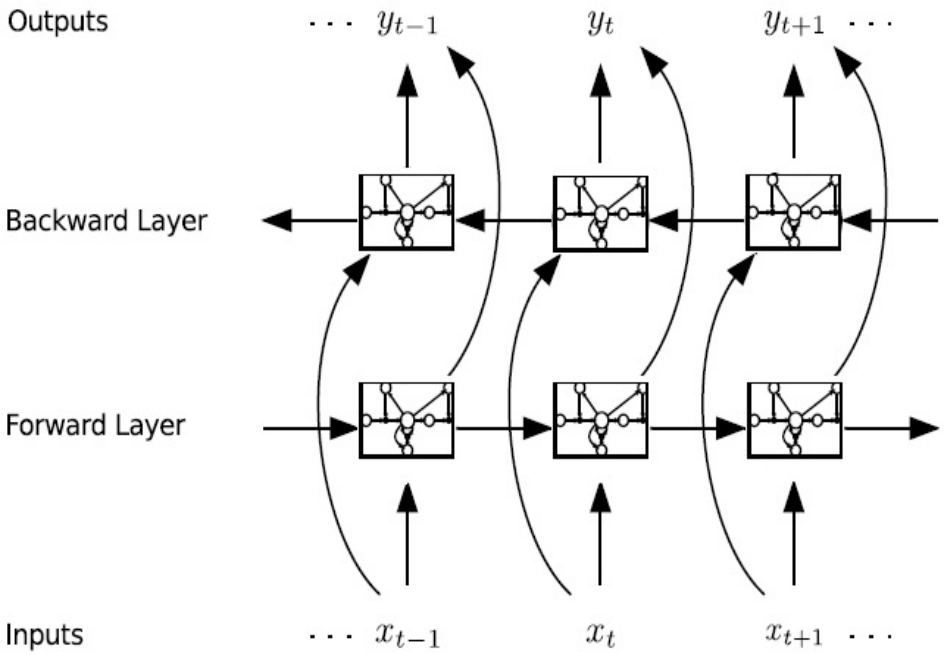
- در لینک زیر ببینید

- http://dsp.ut.ac.ir/en/wp-content/uploads/2017/06/ANN-Lecture6-RNN.pdf



شبکه‌های عصبی LSTM دوطرفه (Bidirectional) ...

- ساختار
  - شامل دو لایه پنهان بازگشتی مجزا (هر لایه پنهان شامل بلوک‌های LSTM می‌باشد).
  - بین این دو لایه پنهان هیچ اتصالی وجود ندارد.
  - هر دو لایه پنهان به یک لایه خروجی متصل شده‌اند.



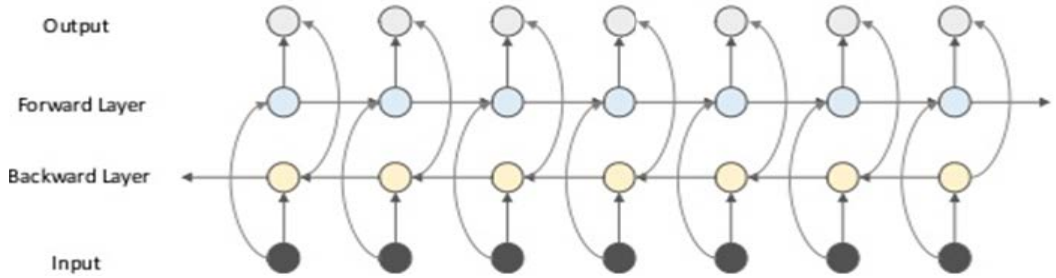
شبکه‌های عصبی LSTM دوطرفه (Bidirectional)

ایده اصلی

هر دنباله ورودی در دو جهت زمانی رو به جلو و از انتها به دو لایه پنهان بازگشتی مجزا داده شود

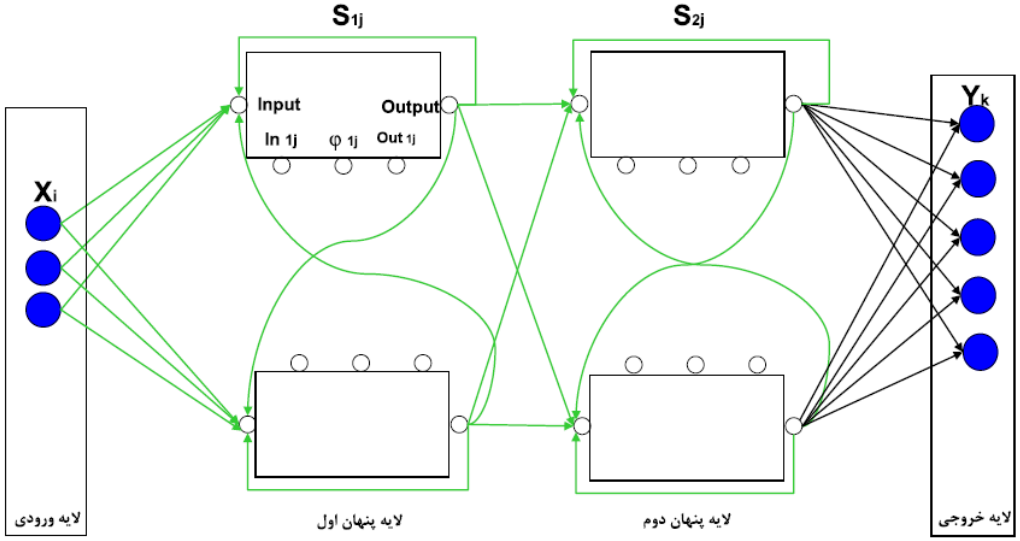
- فرض کنید دنباله آموزشی به صورت $X^T = (x_1, x_2, \dots, x_{T-1}, x_T)$ و دنباله هدف متناظر برابر $t^T = (t_1, t_2, \dots, t_{T-1}, t_T)$ باشد
- در هر مرحله بردار x_i را به لایه Forward و $x_{T-(i-1)}$ را به لایه Backward ارسال کرده و مقدار هد ف را بردار t_i قرار می‌دهیم.
- آموزش شبکه را با استفاده از الگوریتم مربوط به شبکه LSTM دنبال می‌کنیم

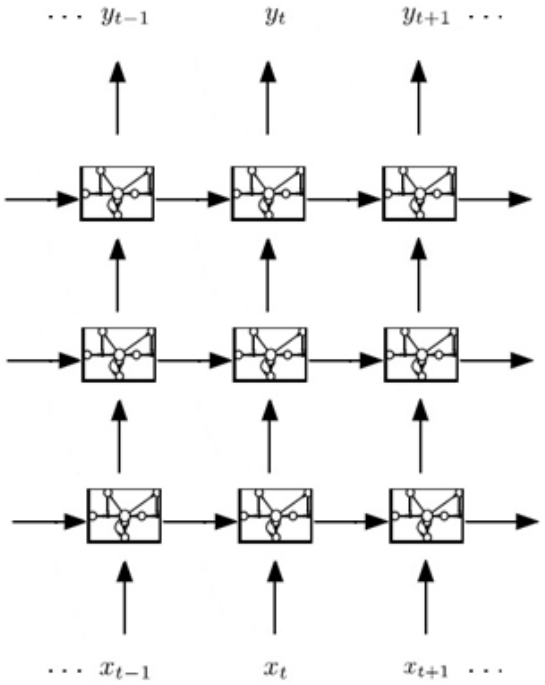
مقدار خالص رسیده به لایه خروجی جمع وزن‌دار مقدار خالص دو لایه Forward وBackward است



شبکه‌های عصبی LSTM: عمیق

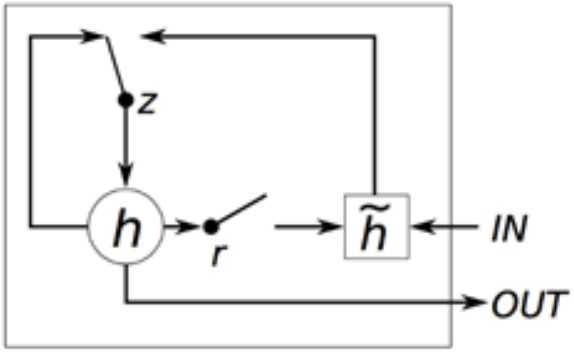
- شبکه عصبی با بیش از یک لایه مخفی
  - خروجی هر لایه پنهان ورودی لایه پنهان بالاتر
  - تقریب زننده جهانی
  - قابلیت یادگیری بیشتر

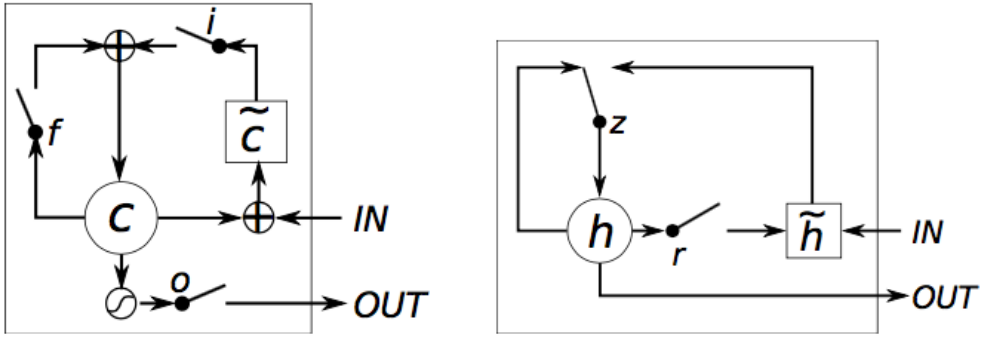




شبکه‌های عصبی LSTM: نسخه ساده شده

- شبکه عصبی واحد بازگشتی دروازه‌ای(GRU)
  - عدم وجود دروازه خروجی
  - دارای دو دروازه (LSTM دارای ۳ دروازه)
    - راه اندازی مجدد (reset) و بروزرسانی (update)
  - عبور تمام مقدار سلول به خروجی یا ورودی سایر بلوک‌ها



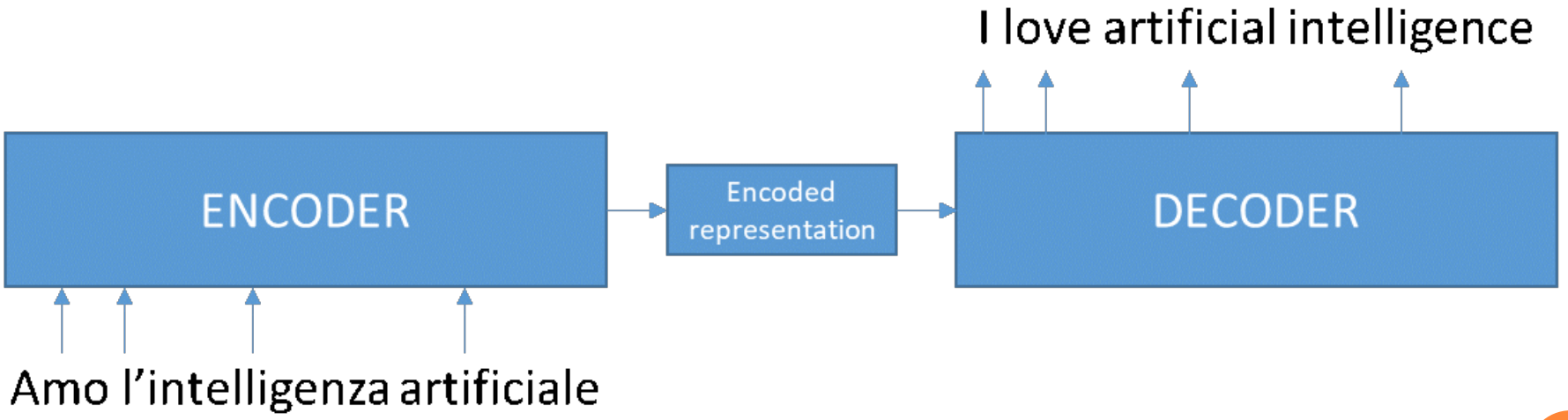


(a) Long Short-Term Memory (b) Gated Recurrent Unit

Figure 1: Illustration of (a) LSTM and (b) gated recurrent units. (a) i, f and o are the input, forget and output gates, respectively. c and $\tilde{c}$ denote the memory cell and the new memory cell content. (b) r and z are the reset and update gates, and h and $\tilde{h}$ are the activation and the candidate activation.

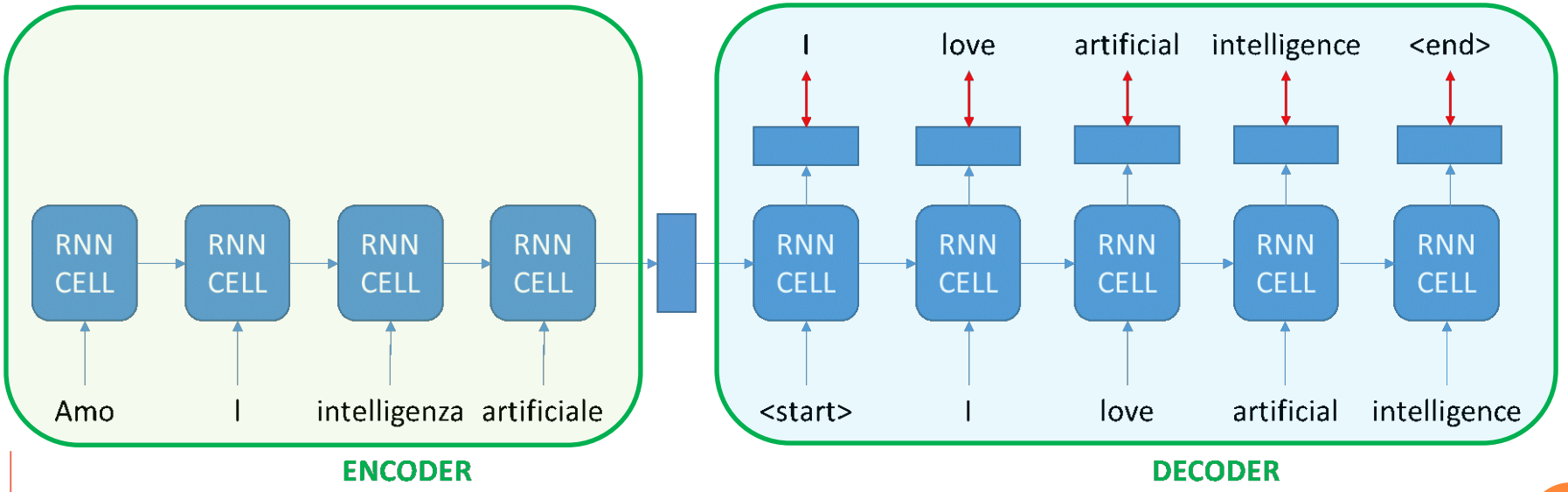
شبکه‌های بازگشتی: Seq2Seq ...

- نگاشت کردن دو دنباله به همدیگر
  - ترجمه ماشینی
  - چت بات
  - خلاصه سازی متن
  - زیرنویس ویدئو
  - ...



شبکه‌های بازگشتی: Seq2Seq

- نحوه کار کردن
  - رمزگذار (Encoder): کد کردن دنباله ورودی در یک بردار بافت (Context)
  - رمزگشا (Decoder): پیش بینی کلمه بعدی با دریافت بردار بافت و کلمه قبلی



شبکه‌های بازگشتی: سازوکار توجه . . .

- توجه (Attention) . . .
  - توجه بیشتر انسان به اطلاعات مهم در پردازش داده (تصویر، متن و ...)

- مثال: عنوان گذاری تصویر

A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.

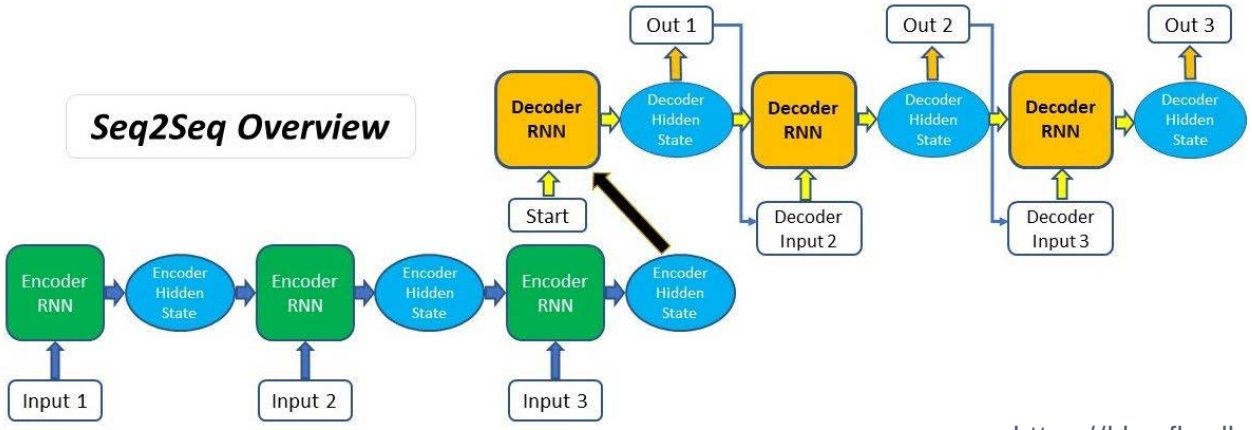
- مثال: تحلیل احساس

شبکه‌های بازگشتی: سازوکار توجه . . .

- توجه (Attention)
  - نواحی توجه در مساله عنوان گذاری تصویر
    - جمله *A woman is throwing a frisbee in a park*

شبکه‌های بازگشتی: سازوکار توجه . . .

- **مشکل**
  - **عدم امکان مدلسازی دقیق دنباله طولانی با یک بردار بافت به عنوان ورودی رمزگشا**
- **راه حل**
  - **در نظر گرفته خروجی همه حالت‌های میانی رمزگذار در فرایند رمزگشایی**
    - برای تولید هر خروجی توسط رمزگشا، از همه حالت‌های میانی رمزگذار استفاده می شود
      - وزن دادن (توجه) بیشتر به حالت‌های میانی مرتبط با خروجی جاری



https://blog.floydhub.com/attention-mechanism

شبکه‌های بازگشتی: سازوکار توجه . . .

سازوکار توجه . . .

- فرض: ترجمه متن n کلمه‌ای x در زبان مبدا به متن m کلمه‌ای y در زبان مقصد

$$\mathbf{x} = [x_1, x_2, \ldots, x_n]$$
$$\mathbf{y} = [y_1, y_2, \ldots, y_m]$$

- رمزگذار: یک BiLSTM با حالت‌های مخفی جلورو $\overrightarrow{h}_i$ و عقب‌رو $\overleftarrow{h}_i$
  - حالت کلی رمزگذار: اتصال دو دو حالت جلورو و عقب رو

$$\boldsymbol{h}_i = [\overrightarrow{\boldsymbol{h}}_i^\top ; \overleftarrow{\boldsymbol{h}}_i^\top]^\top, i = 1, \ldots, n$$

- حالت مخفی رمزگشا برای کلمه خروجی t=1,…,m

$$\boldsymbol{s}_t = f(\boldsymbol{s}_{t-1}, y_{t-1}, \mathbf{c}_t)$$

  - تابعی از حالت قبلی رمزگشا، کلمه خروجی قبلی و بردار بافت $\mathbf{c}_t$
    - بردار بافت: جمع وزن‌دار حالت‌های رمزگذار برای همه کلمات ورودی

$$\mathbf{c}_t = \sum_{i=1}^{n} \alpha_{t,i} \boldsymbol{h}_i$$

$$\alpha_{t,i} = \text{align}(y_t, x_i) = \frac{\exp(\text{score}(\boldsymbol{s}_{t-1}, \boldsymbol{h}_i))}{\sum_{i'=1}^{n} \exp(\text{score}(\boldsymbol{s}_{t-1}, \boldsymbol{h}_{i'}))}$$

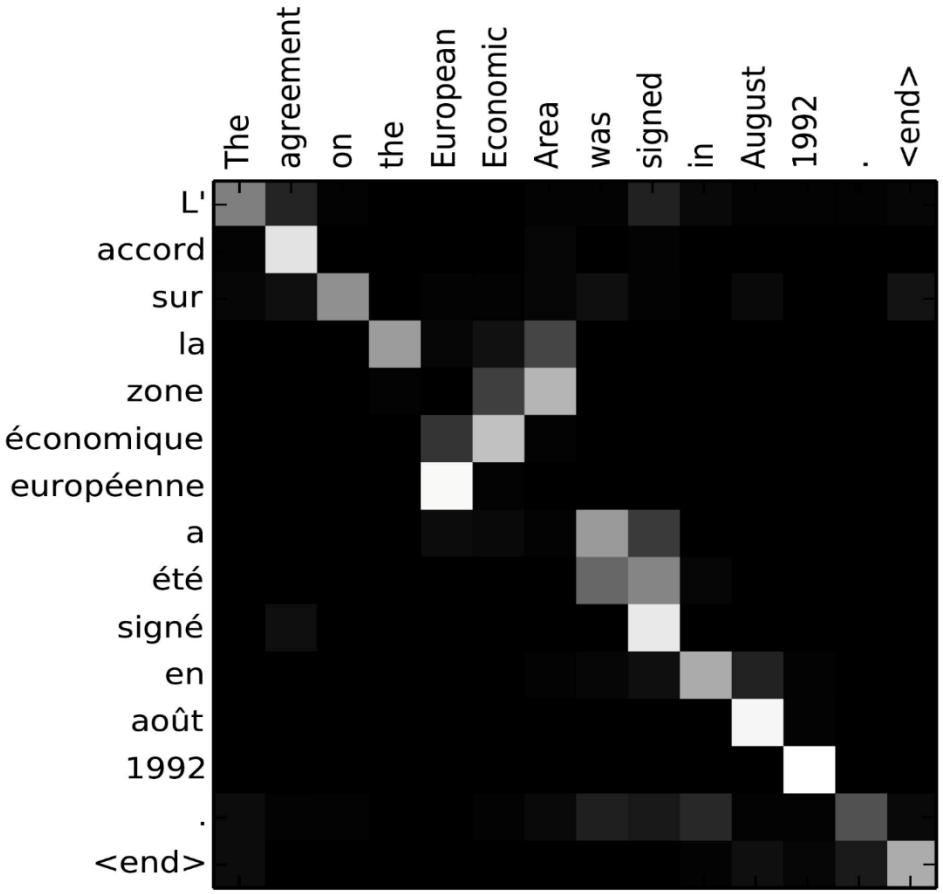
وزن‌های قابل آموزش با فرایند یادگیری

$$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \mathbf{v}_a^\top \tanh(\mathbf{W}_a[\boldsymbol{s}_t; \boldsymbol{h}_i])$$

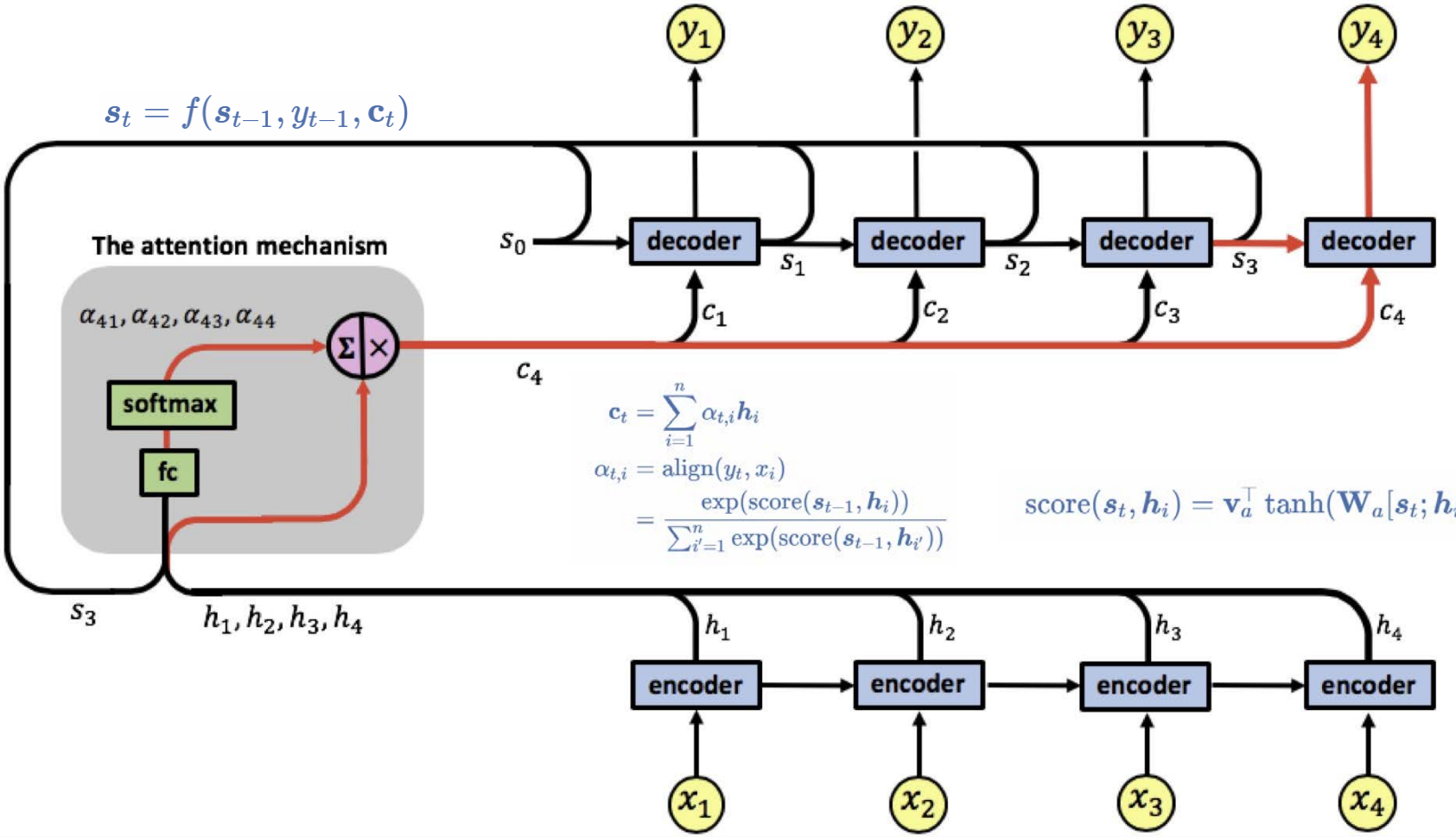
- وزن‌های $\alpha_{t,i}$= میزان توجه (تراز بودن) کلمه ورودی i با کلمه خروجی t

شبکه‌های بازگشتی: سازوکار توجه . . .

- سازوکار توجه . . .



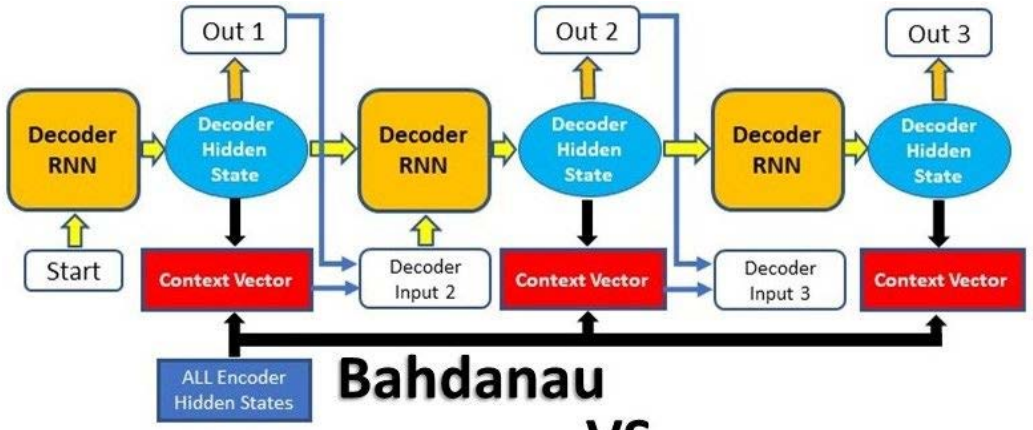
شبکه‌های بازگشتی: سازوکار توجه . . .



شبکه‌های بازگشتی: سازوکار توجه . . .

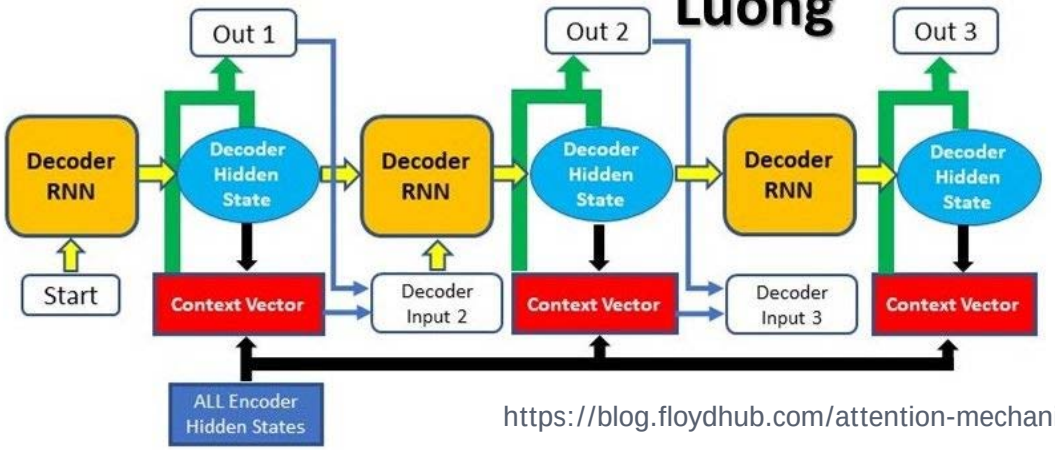
- انواع توجه

Additive Attention



Bahdanau VS Luong

Multiplicative Attention



https://blog.floydhub.com/attention-mechanism

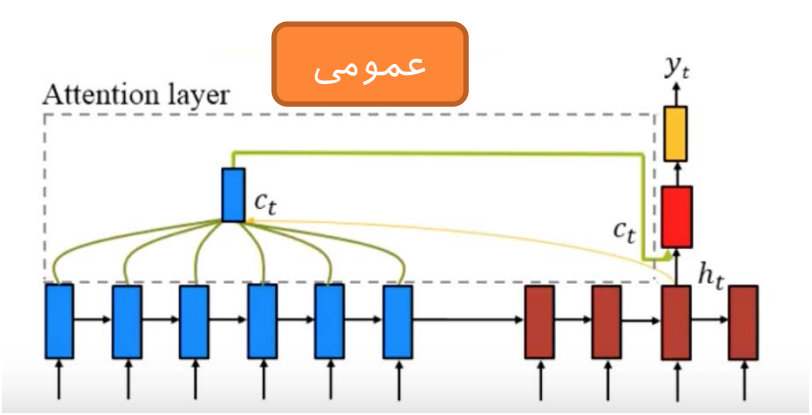
شبکه‌های بازگشتی: سازوکار توجه ...

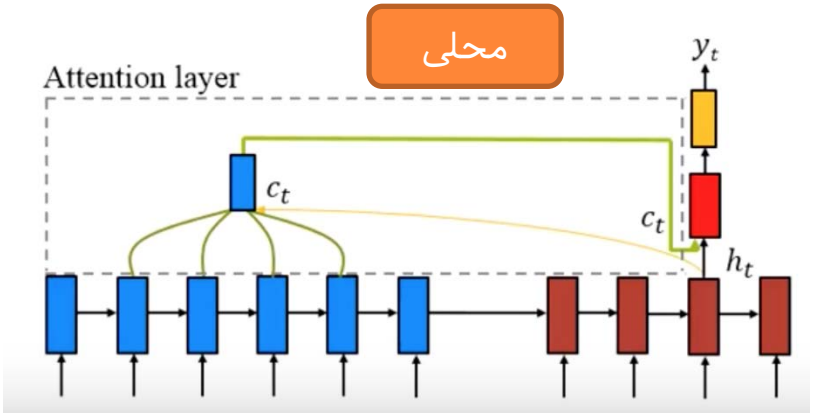
- روش‌های مختلف محاسبه امتیاز توجه

Name	Alignment score function	Citation
Content-base attention	$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \text{cosine}[\boldsymbol{s}_t, \boldsymbol{h}_i]$	Graves2014
Additive(*)	$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \mathbf{v}_a^\top \tanh(\mathbf{W}_a[\boldsymbol{s}_t; \boldsymbol{h}_i])$	Bahdanau2015
Location-Base	$\alpha_{t,i} = \text{softmax}(\mathbf{W}_a \boldsymbol{s}_t)$ Note: This simplifies the softmax alignment to only depend on the target position.	Luong2015
General	$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \boldsymbol{s}_t^\top \mathbf{W}_a \boldsymbol{h}_i$ where $\mathbf{W}_a$ is a trainable weight matrix in the attention layer.	Luong2015
Dot-Product	$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \boldsymbol{s}_t^\top \boldsymbol{h}_i$	Luong2015
Scaled Dot-Product(^)	$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \frac{\boldsymbol{s}_t^\top \boldsymbol{h}_i}{\sqrt{n}}$ Note: very similar to the dot-product attention except for a scaling factor; where n is the dimension of the source hidden state.	Vaswani2017

شبکه‌های بازگشتی: سازوکار توجه . . .

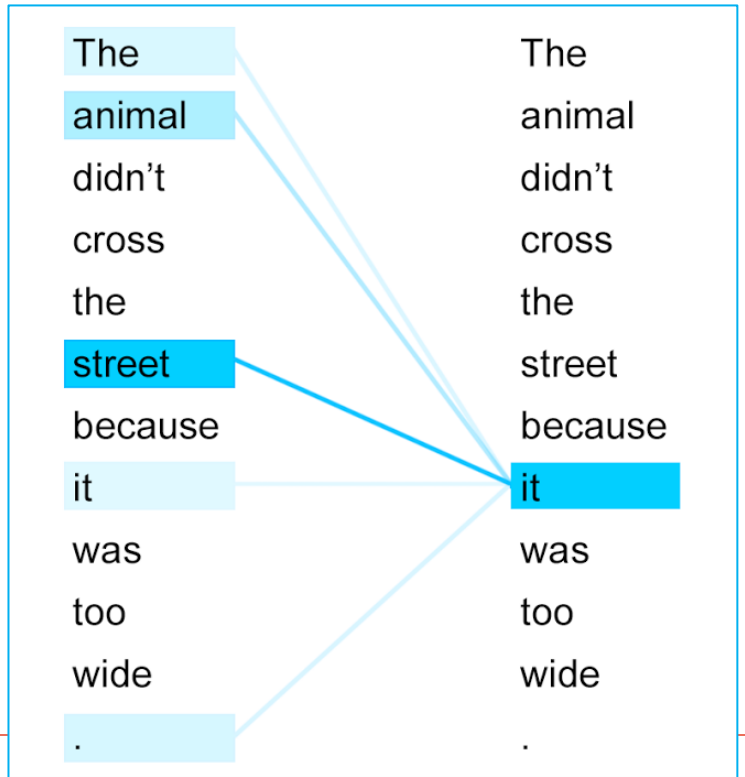
- **انواع توجه . . .**
  - **نرم (Soft) و سخت (Hard)**
    - نرم: یادگیری وزن‌ها و محاسبه آنها با در نظر گرفتن کل داده (کل تصویر یا کل جمله)
    - سخت: در نظر گرفتن تنها بخشی از داده در هر مرحله
  - **محلی (Local) و عمومی (Global)**
    - عمومی = توجه نرم = در نظر گرفتن کل داده در محاسبات وزن‌ها
    - محلی: پیش بینی فقط یک خروجی برای تراز کردن با ورودی جاری و سپس در نظر گرفتن یک پنجره اطراف این ورودی برای محاسبه بردار بافت

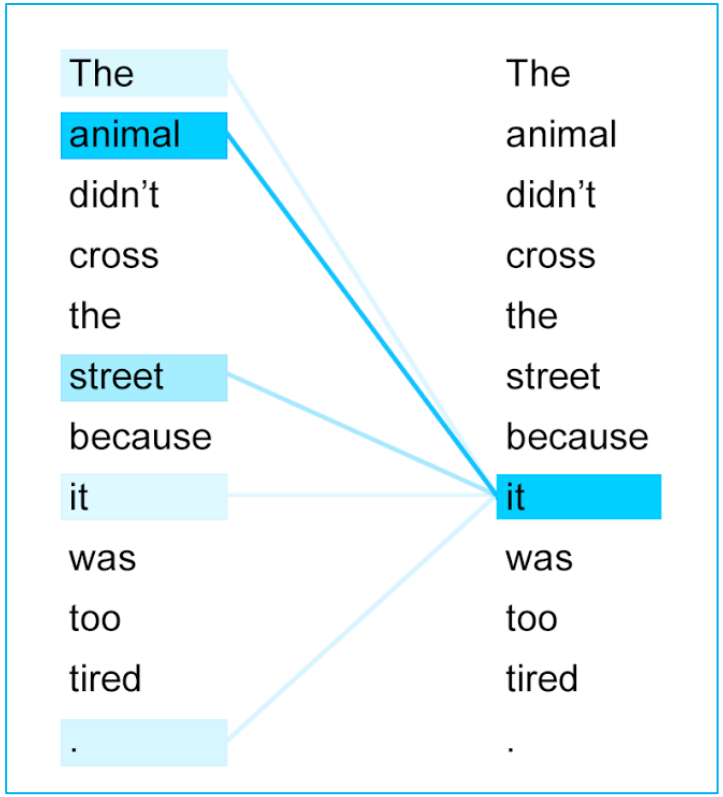




شبکه‌های بازگشتی: سازوکار توجه . . .

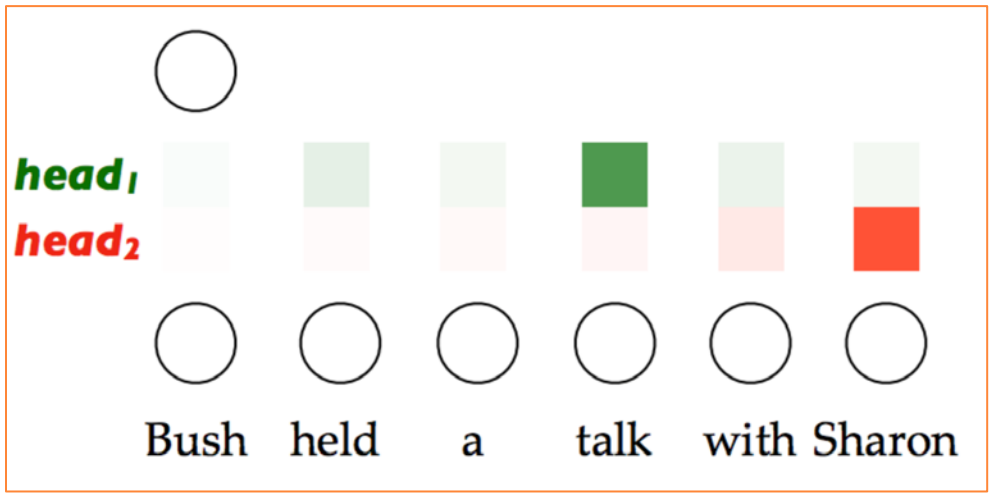
- انواع توجه . . .
  - توجه به خود (Self-Attention)
    - بیانگر میزان توجه مابین واحدهای مختلف یک دنباله (مانند کلمات یک جمله)
      - یاد گرفتن اینکه it معادل کدام کلمه است (مرجع ضمیر)

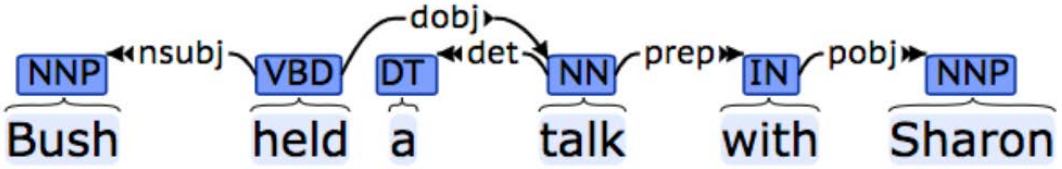




شبکه‌های بازگشتی: سازوکار توجه . . .

- انواع توجه . . .
  - توجه چند هسته‌ای (Multi-Head Attention)
    - حالت توسعه یافته Self-Attention که در آن چند لایه توجه نرم به صورت موازی اجرا شده و نتایج آنها با هم ادغام می‌شود

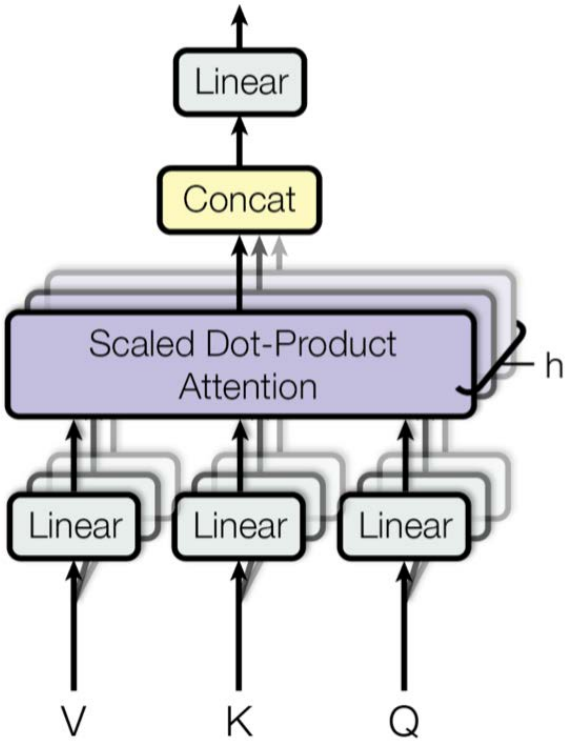




شبکه‌های بازگشتی: سازوکار توجه . . .

انواع توجه . . .

- **توجه چند هسته‌ای** (Multi-Head Attention)
  - اجرای چند Self-Attention به صورت موازی: وزن‌ها (Subspace)ی متفاوت
  - الحاق آنها به هم
  - ادغام با یک لایه



مثلا h=8

scaled dot-product attention

$$\text{Attention}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \text{softmax}(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{n}})\mathbf{V}$$

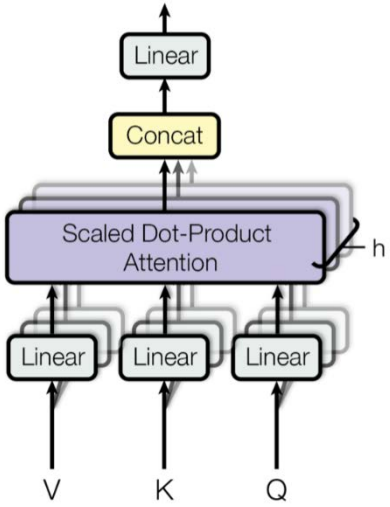
ابعاد Key

- ایده: ensembling

لایه وزنی خطی: تنظیم از طریق یادگیری

$$\text{MultiHead}(\mathbf{Q},\mathbf{K},\mathbf{V}) = [\text{head}_1;\ldots;\text{head}_h]\mathbf{W}^O$$
$$\text{where head}_i = \text{Attention}(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{K}\mathbf{W}_i^K, \mathbf{V}\mathbf{W}_i^V)$$

شبکه‌های بازگشتی: سازوکار توجه . . .



$$\text{Attention}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \text{softmax}(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{n}})\mathbf{V}$$

انواع توجه . . .

- **توجه چند هسته‌ای(Multi-Head Attention): مثال**
  - برای عبارت Action gets results
  - محاسبه برای هر کلمه نسبت به سایر کلمات
  - برای کلمه اول (Action)
    - بردار Query = کلمه Action (در ترجمه خروجی قبلی رمزگشا)
    - بردارهای Keys: همه کلمات (در ترجمه بردارهای حالت مخفی رمزگذار)
    - بردارهای Value: همه کلمات (در ترجمه بردارهای حالت مخفی رمزگذار)

n= 64

میزان توجه

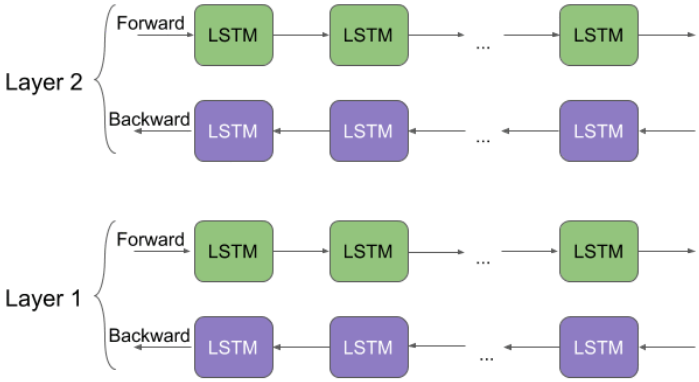
Word	q vector	k vector	v vector	score	score / 8	Softmax	Softmax * v	Sum
Action	q_1	k_1	v_1	$q_1 . k_1$	$q_1 . k_1 / 8$	x_{11}	$x_{11} * v_1$	z_1
gets		k_2	v_2	$q_1 . k_2$	$q_1 . k_2 / 8$	x_{12}	$x_{12} * v_2$	
results		k_3	v_3	$q_1 . k_3$	$q_1 . k_3 / 8$	x_{13}	$x_{13} * v_3$	

خروجی MHA

محاسبه به صورت مشابه برای بقیه کلمات

شبکه عصبی LSTM: کاربردها . . .

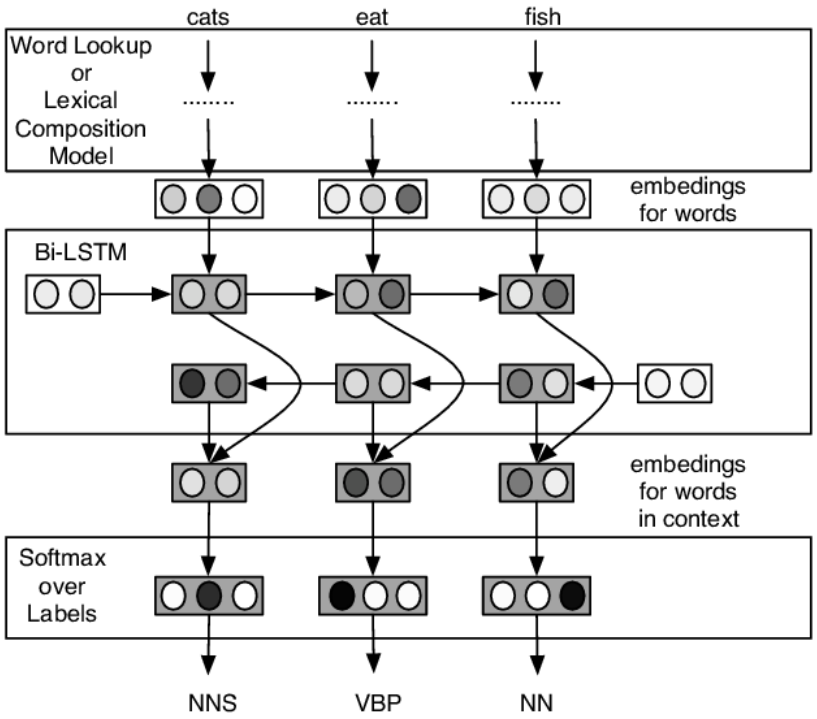
- مدل زبانی و بردار تعبیه کلمات . . .
  - استفاده از LSTM استاندارد
  - بردارهای تعبیه جدا برای کلمات Polysemy (ظاهر یکسان و معنی متفاوت)
    - شبکه Embeddings from Language Models (ELMo)

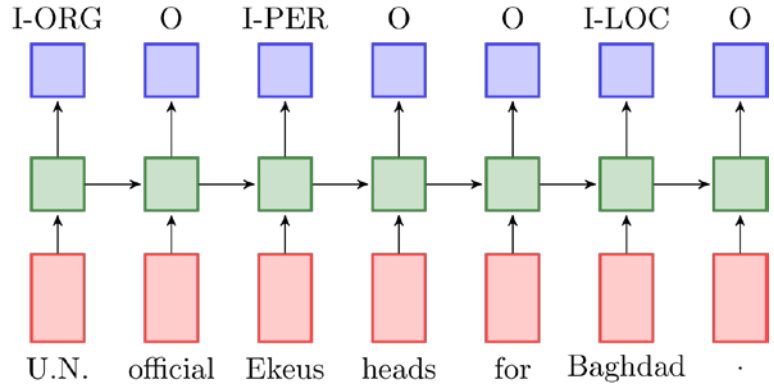


Peters, Matthew E., et al. "Deep contextualized word representations." *arXiv preprint arXiv:1802.05365* (2018).

شبکه عصبی LSTM: کاربردها . . .

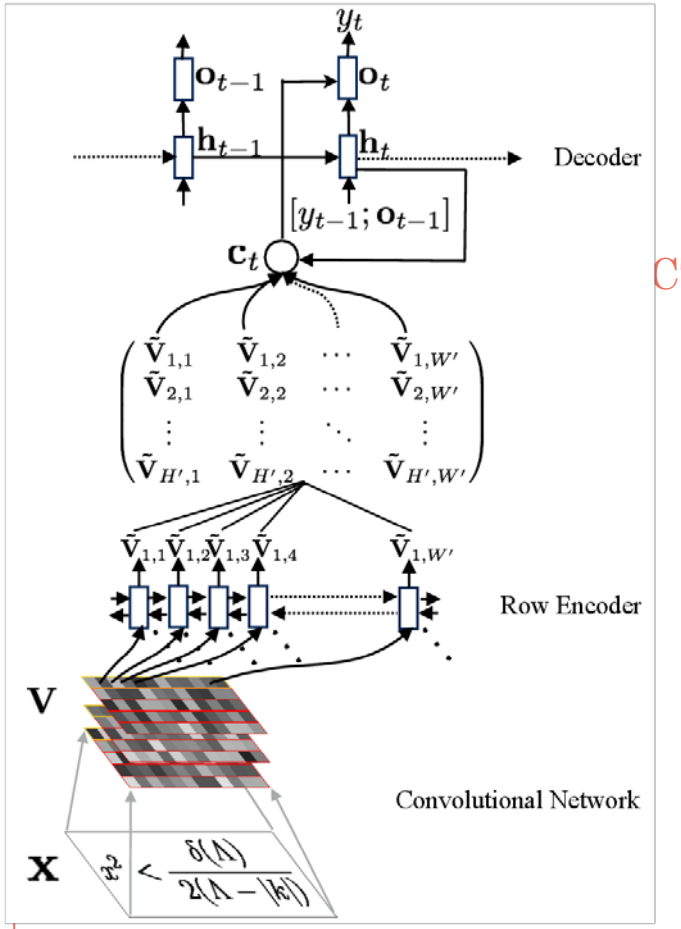
- برچسب زنی اجزای کلام (POS)/بازشناسی پدیده‌های اسمی (NER)
  - ورودی: بردار یک کلمه (مثل Word Vector)
  - خروجی: به تعداد برچسب‌ها (هر نرون یک برچسب)

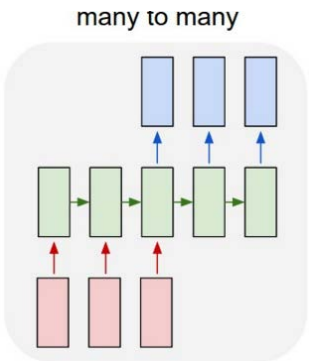




شبکه عصبی LSTM: کاربردها . . .

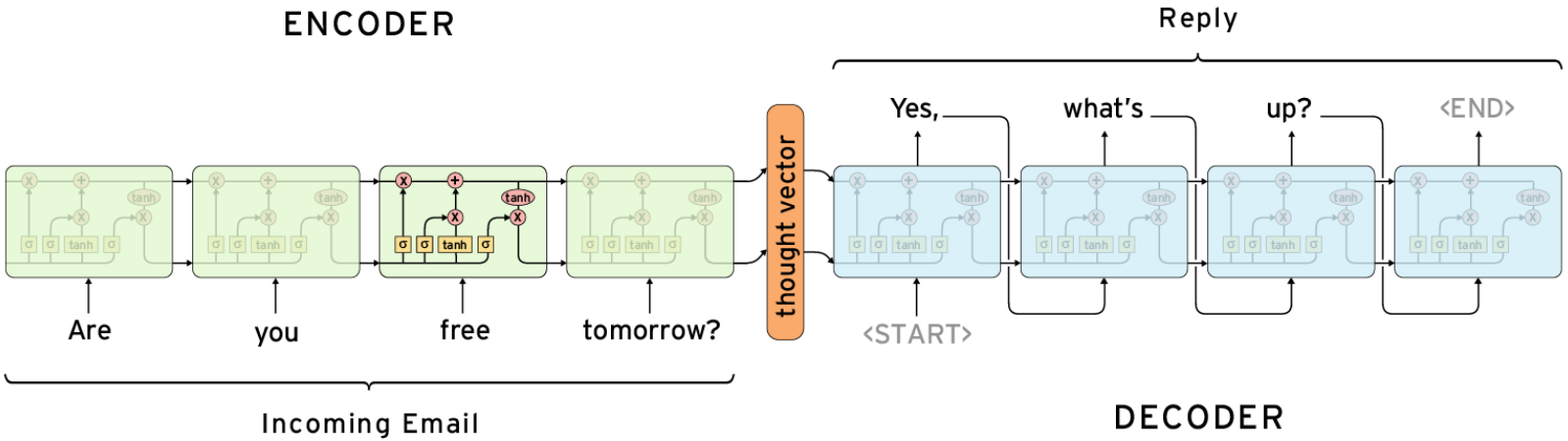
- بازشناسی دست خط/ نویسه‌های نوری (OCR)
  - ورودی: بردار مربوط به ویژگی یک فریم از تصویر
    - ویژگی های CNN
  - خروجی: به تعداد واحدهای بازشناسی شونده
    - هر نرون یک نویسه (کاراکتر)
    - نیاز به تبدیل دنباله برچسب فریم ها به دنباله نویسه = CTC

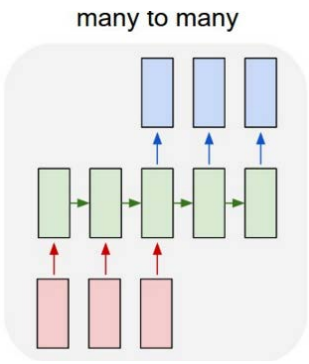




شبکه عصبی LSTM: کاربردها . . .

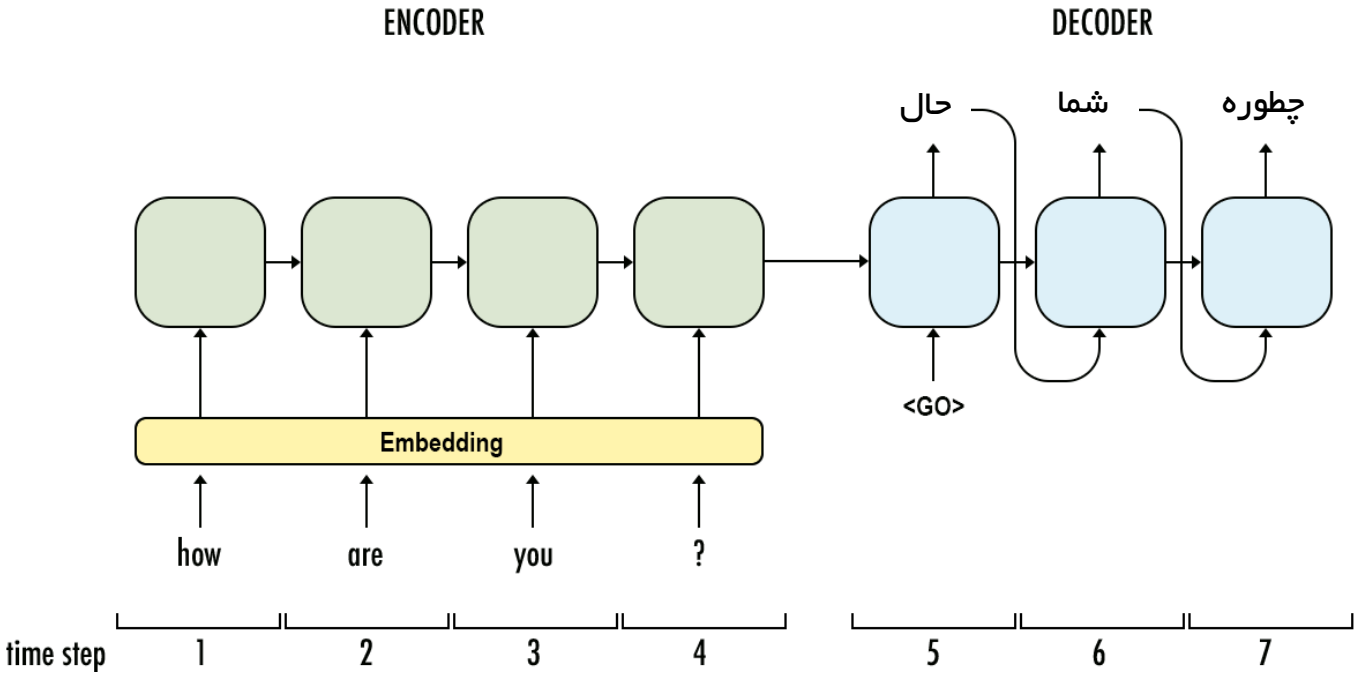
- چت بات
  - ورودی: دنباله کلمات ورودی (تبدیل هر کلمه به بردار با روشهای Word Vector)
  - خروجی: دنباله کلمات پاسخ





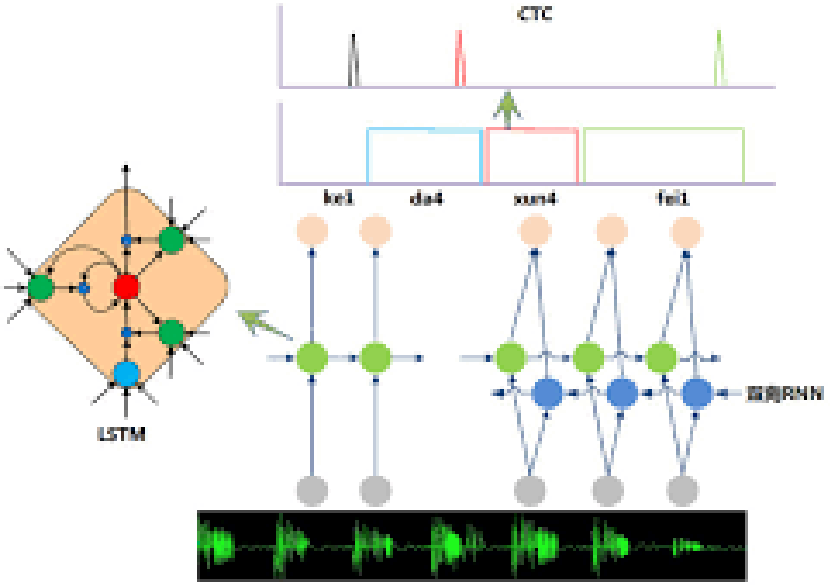
شبکه عصبی LSTM: کاربردها . . .

- ترجمه ماشینی
  - ورودی: بردار یک کلمه (مثل Word Vector) در زبان مبدا
  - خروجی: احتمال کلمات در زبان مقصد



شبکه عصبی LSTM: کاربردها . . .

- بازشناسی گفتار
  - ورودی: بردار مربوط به یک فریم گفتار
  - خروجی: به تعداد واحدهای بازشناسی شونده (هر نرون یک واج)
    - نیاز به روشی برای تبدیل دنباله برچسب فریم ها به دنباله واج = CTC



شبکه عصبی LSTM: تشخیص واج‌های فارسی . . .

- **داده‌ها: فارس‌دات**
  - شامل ۶۰۸۰ سیگنال صوتی
  - داده آموزش: ٪۹۵ کل داده (معادل ۵۶۹۸ سیگنال)
  - داده آزمون: ٪۵ کل داده (معادل ۳۸۲ سیگنال)
- **استخراج ویژگی**
  - روش مورد استفاده: MFCC
  - تعداد ضرایب هر فریم: ۳۹
  - طول فریم: ۱۶ میلی ثانیه
- **شبکه مورد استفاده: LSTM**
- **ساختار شبکه**
  - نرون‌های لایه ورودی: ۳۹
  - نرون‌های لایه خروجی: ۳۰ (تعداد واج‌های فارسی+ سکوت)
  - وزن‌های اولیه: تصادفی در بازه $\left[\frac{-1}{10}, \frac{1}{10}\right]$
  - نرخ یادگیری: ۰/۰۰۰۳
  - تعداد بلوک حافظه: ۱۵۰

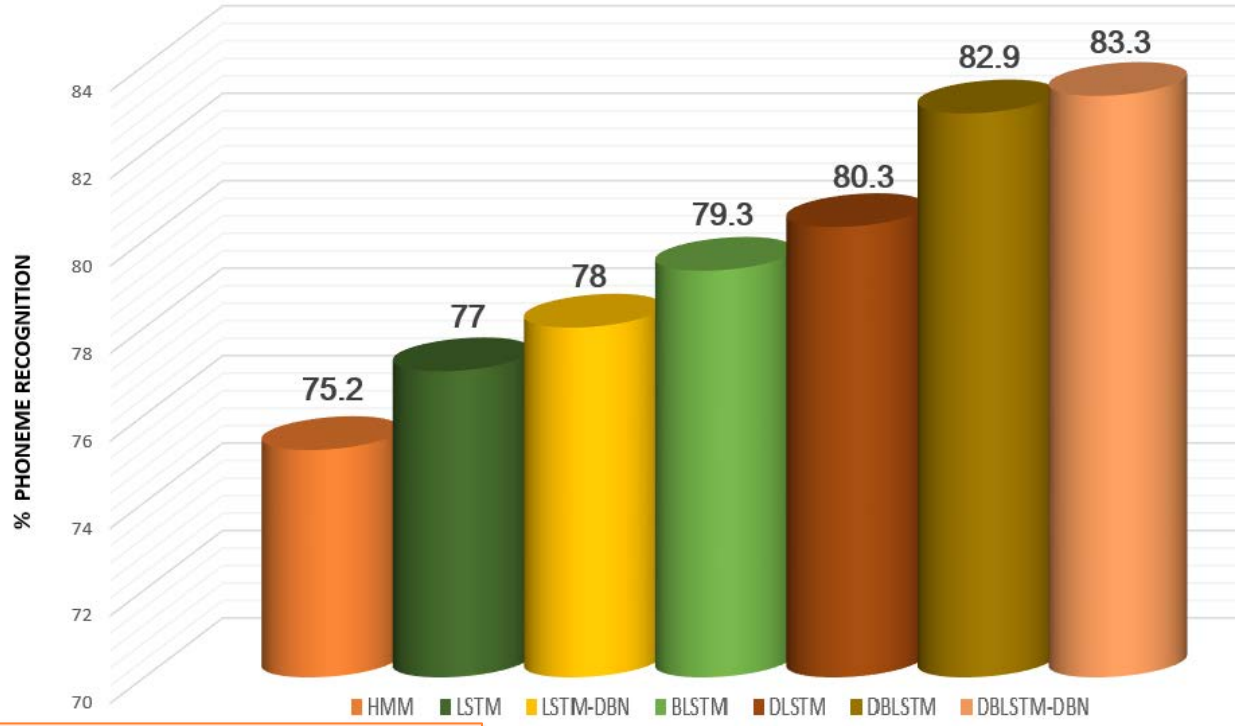
آرمیتا حجی‌مانی، استفاده از یادگیری عمیق برای بازشناسی گفتار فارسی، پایان‌نامه کارشناسی ارشد، دانشگاه تهران، ۱۳۹۵

شبکه عصبی LSTM دوطرفه: تشخیص واج‌های فارسی . . .

- **داده‌ها: فارس‌دات**
  - شامل ۶۰۸۰ سیگنال صوتی
  - داده آموزش: ۸۰٪ کل داده (معادل ۴۸۶۴ سیگنال)
  - داده آزمون: ۲۰٪ کل داده (معادل ۱۲۱۶ سیگنال)
- **استخراج ویژگی**
  - روش مورد استفاده: MFCC
  - تعداد ضرایب هر فریم: ۳۹
  - طول فریم: ۱۶ میلی ثانیه
- **شبکه مورد استفاده: Bidirectional LSTM**
- **ساختار شبکه**
  - نرون‌های لایه ورودی: ۳۹
  - نرون‌های لایه خروجی: ۳۰
  - وزن‌های اولیه: تصادفی در بازه $\left[\frac{-1}{10}, \frac{1}{10}\right]$
  - نرخ یادگیری: ۰/۰۰۰۳
  - تعداد بلوک حافظه: ۱۲۰

شبکه عصبی LSTM دوطرفه: تشخیص واج‌های فارسی

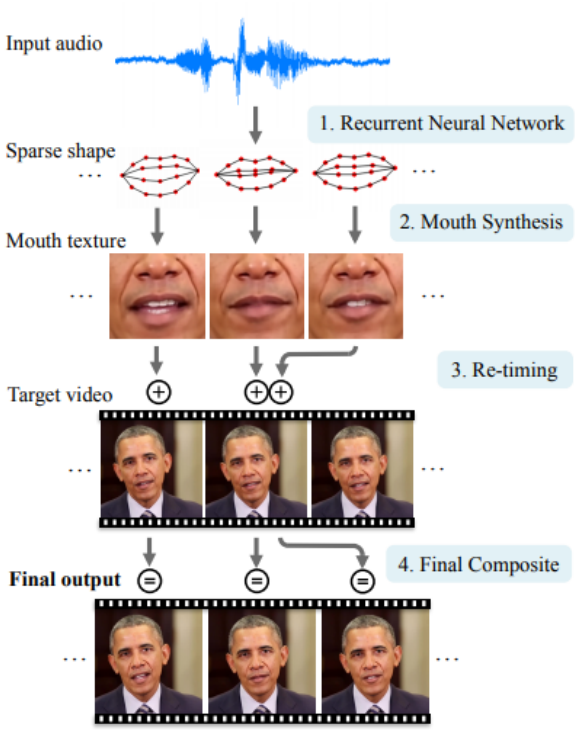
- دقت روی واج
  - کارایی بالاتر شبکه‌های دوطرفه نسبت به شبکه‌های یک‌طرفه
  - کارایی بالاتر شبکه‌های عمیق نسبت به شبکه‌های غیر عمیق



آرمیتا حجی‌مانی، استفاده از یادگیری عمیق برای بازشناسی گفتار فارسی، پایان‌نامه کارشناسی ارشد، دانشگاه تهران، ۱۳۹۵

شبکه عصبی LSTM: تولید سخنرانی ساختگی

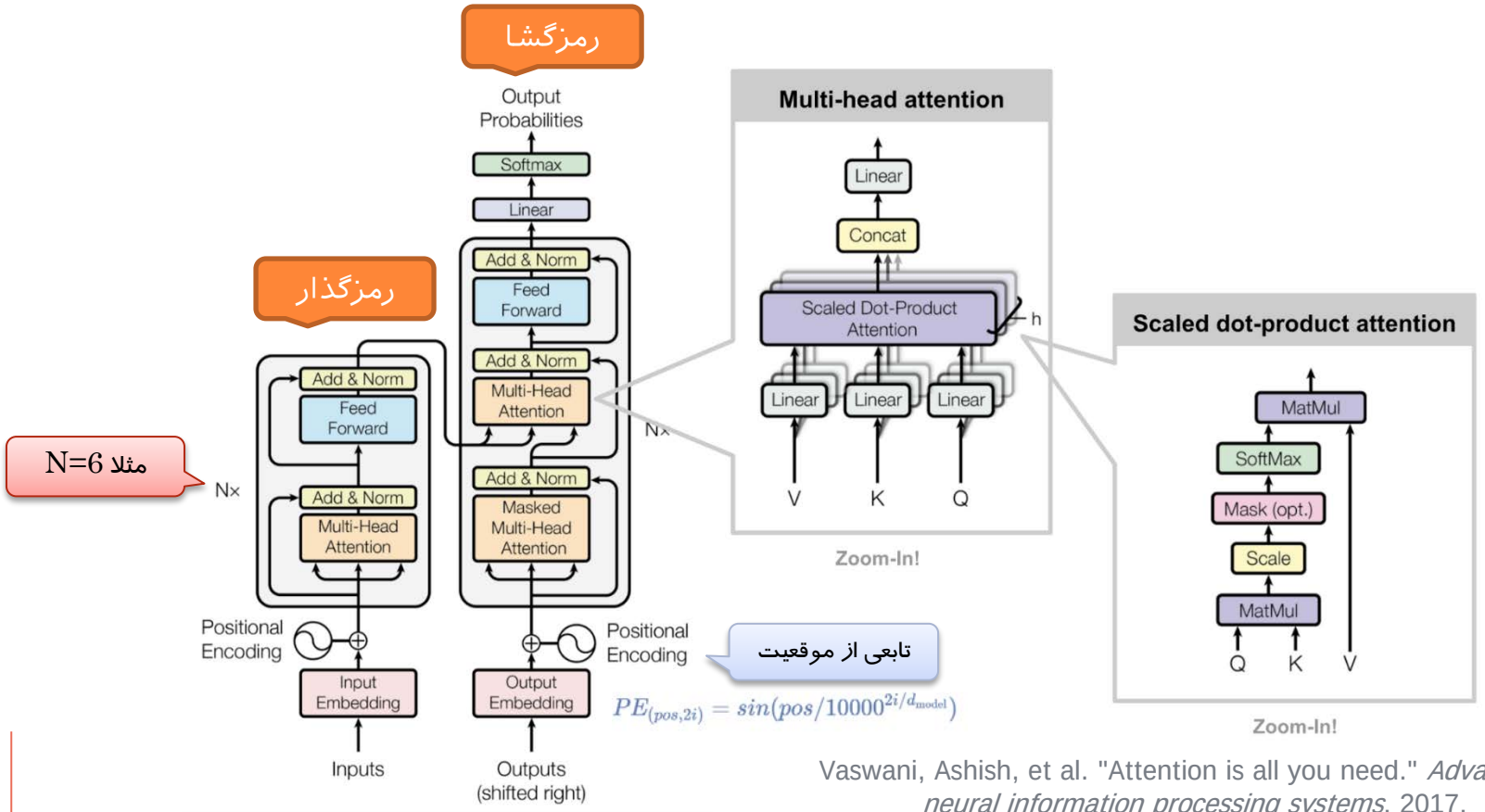
- تولید سخنرانی ساختگی برای اوباما: یادگیری حرکت لب با توجه به صدا



[Suwajanakorn, Supasorn, Steven M. Seitz, and Ira Kemelmacher-Shlizerman. "Synthesizing obama: learning lip sync from audio.", 2017]

شبکه عصبی: مبدل‌ها ...

- مبدل (Transformer)
  - تبدیل ورودی به خروجی و مدل کردن وابستگی بین آنها با سازوکار توجه (بدون RNN)

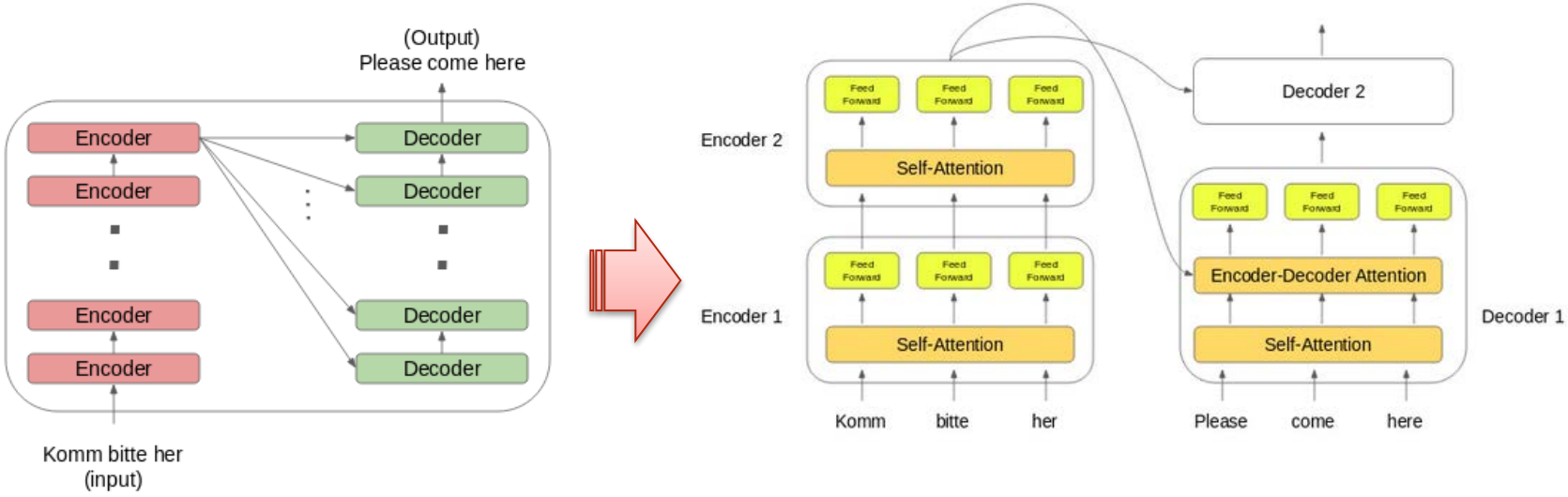


Vaswani, Ashish, et al. "Attention is all you need." *Advances in neural information processing systems*. 2017.

شبکه عصبی: مبدل‌ها ...

ساختار

- تعداد N (مثلا ۶) رمزگذار و رمزگشا پشته شده
- دنباله (همه کلمات) ورودی به اولین رمزگذار داده می‌شود
- خروجی هر رمزگذار به رمزگذار بعدی داده می‌شود
- خروجی آخرین رمزگذار به همه رمزگشاها داده می‌شود



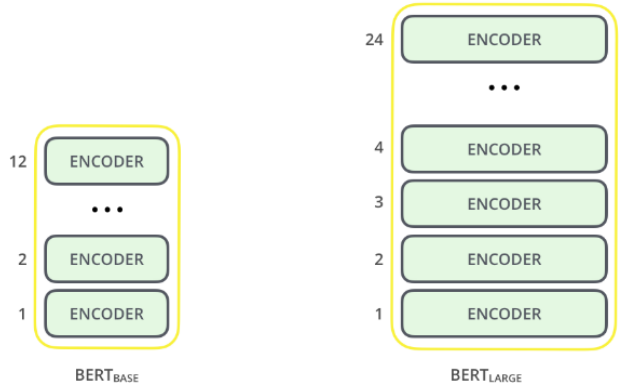
پیاده سازی در http://nlp.seas.harvard.edu/2018/04/03/attention.html

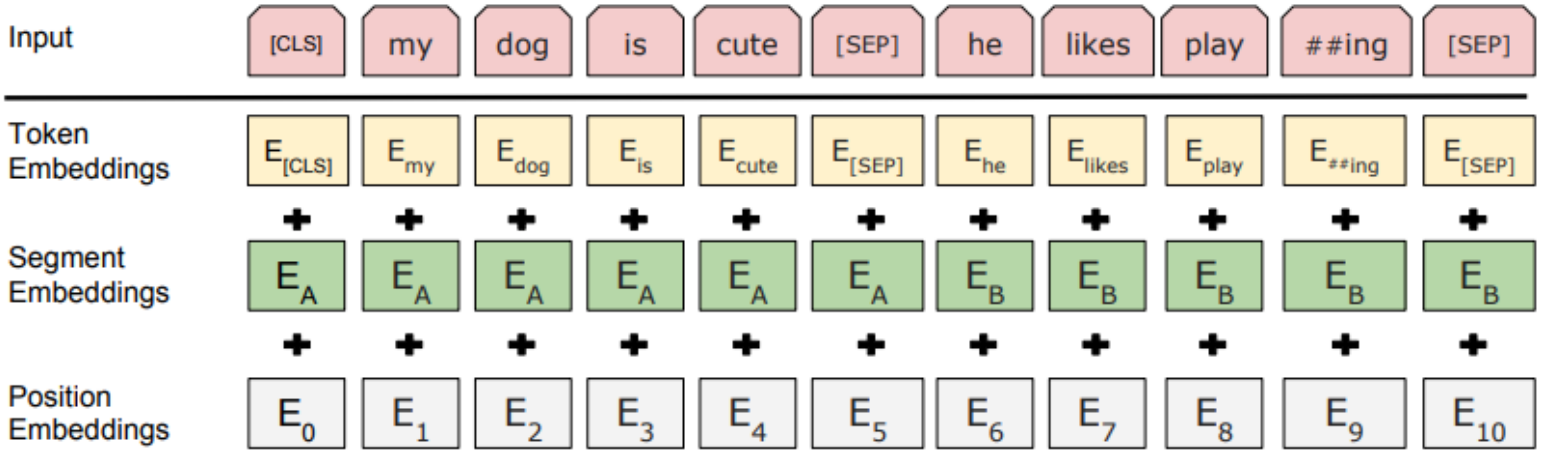
شبکه عصبی: مبدل‌ها ...

- بهبودها
  - رفع مشکل ثابت بودن طول دنباله ورودی (مشکل context fragmentation)
    - شبکه Transformer-XL: در نظر گرفتن بردار بافت قطعه قبلی به عنوان یکی دیگر از ورودی‌ها
  - مدل کردن ترتیب در دنباله به صورت مستقیم (مورد نیاز در یادگیری تقویتی)
    - Simple Neural Attention Meta-Learner (SNAIL)
  - شبکه Self-Attention GAN
    - مدل کردن وابستگی های خارج از محدوده فیلتر (وابستگی نواحی بزرگتر)

شبکه عصبی: کاربردهای مبدل‌ها . . .

- مدل زبانی و بردار تعبیه کلمات . . .
  - مدل Bidirectional Encoder Representations from Transformers (BERT)





شبکه عصبی: کاربردهای مبدل‌ها . . .

- پروژه Generative Pre-trained Transformer 3 (GPT-3)
  - مدل زبانی برای تولید متن طبیعی: تولید داستان، خلاصه سازی، شعر گفتن، کد نوشتن و ...
  - توسط OpenAI در ۲۰۲۰
  - دارای ۱۷۵ میلیارد پارامتر = بزرگترین شبکه عصبی ساخته شده تاکنون
    - هزینه آموزش شبکه = ۴.۶ میلیون دلار
    - آموزش روی حدود ۵۰۰ میلیارد واحد (کلمه) از متون مختلف (کتاب، اینترنت و ...)

https://gpt3examples.com
https://gpt3demo.com

شبکه عصبی: کاربردهای مبدل‌ها . . .

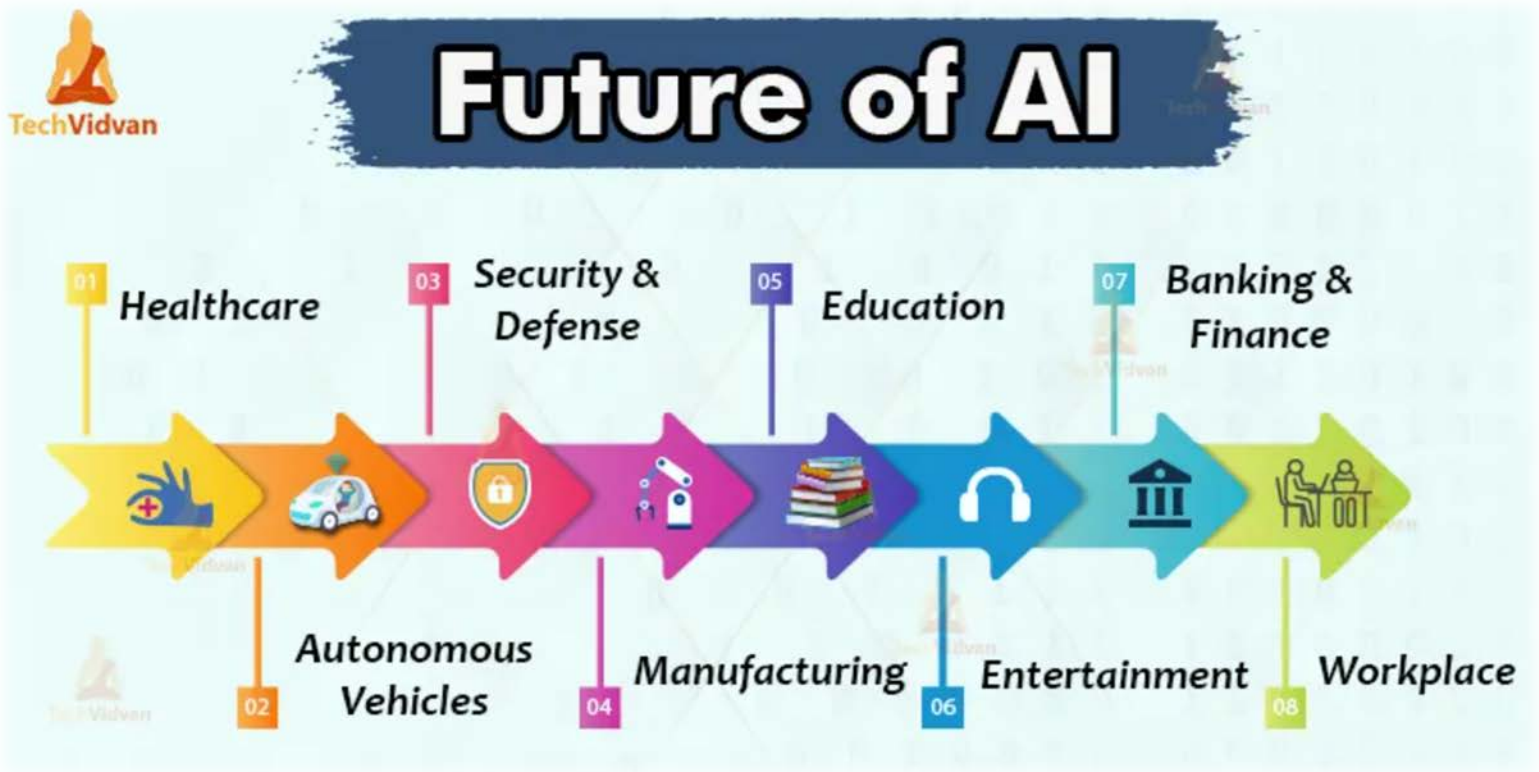
- پروژه Generative Pre-trained Transformer 3 (GPT-3)

شبکه عصبی: کاربردهای مبدل‌ها

روش غالب در بیشتر (!) کاربردهای مدل‌سازی دنباله

بسترهای یادگیری عمیق

Name	Platform	Written In	Cuda	Parallel Execution	Trained Model	RNN	CNN
Tensorflow	Linux, Window,MacOs, Rasbian,Mobile, Webapp	Python, C++, Cuda	Yes	Yes	Yes	Yes	Yes
Pytorch	Linux, Window, MacOs	Python, C++, Cuda	Yes	Yes	Yes	Yes	Yes
Keras	Linux, MacOs, window	Python	Yes	Yes	Yes	Yes	Yes
Mxnet	Linux, Window, Mac,Mobile, Webapp	C++, Python, R, Julia, Scala, Go, Perl	Yes	Yes	Yes	Yes	Yes
Deeplearning4j	Window, Linux,Mac, Mobile	Java, Scala, Cuda, C++, Perl, Python, Closure	Yes	Yes	Yes	Yes	Yes
Microsoft CNTK	Window, Linux	C++	Yes	Yes	Yes	Yes	Yes


TechVidvan
Future of AI
01
Healthcare
02
Autonomous Vehicles
03
Security & Defense
04
Manufacturing
05
Education
06
Entertainment
07
Banking & Finance
08
Workplace