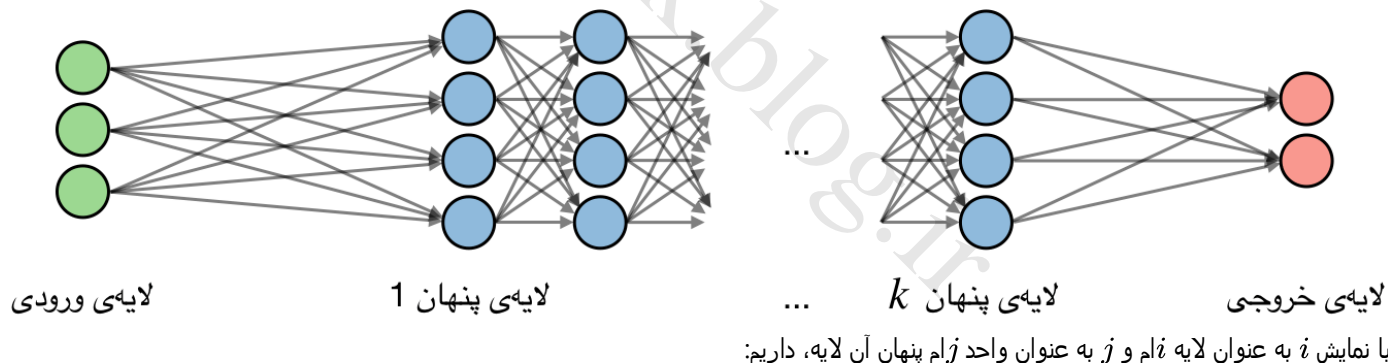


## راهنمای کوتاه یادگیری عمیق

### شبکه‌های عصبی

شبکه‌های عصبی دسته‌ای از مدل‌هایی هستند که با لایه‌بندی ساخته میشوند (ساختاری چند لایه دارند). شبکه‌های عصبی پیچشی ( کانولوشنی (CNN)) و شبکه‌های عصبی بازگشتی (RNN) انواع رایج شبکه‌های عصبی هستند.

معماری - واژه معماری در شبکه‌های عصبی در شکل زیر توصیف شده است:

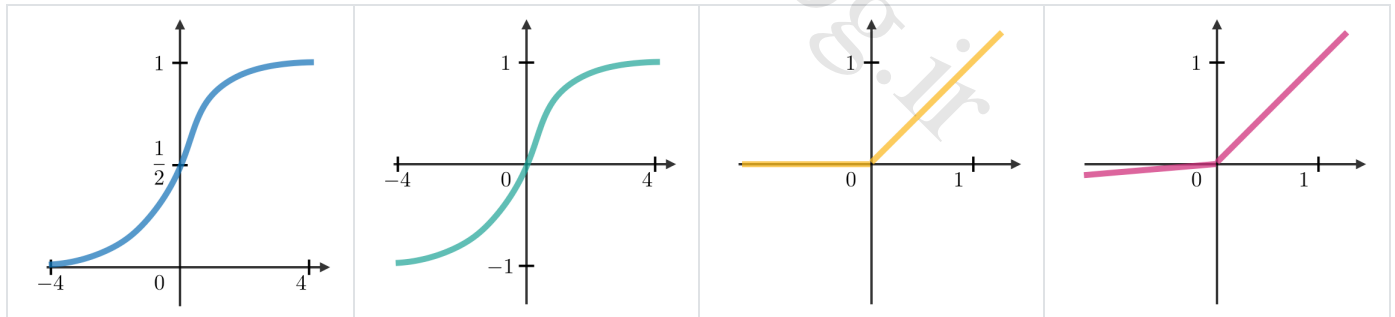


$$z_j^{[i]} = w_j^{[i]T} x + b_j^{[i]}$$

که به ترتیب  $w$ ،  $b$ ، و  $z$  وزن، پیش‌قدر، و خروجی لایه هستند.

تابع فعال‌سازی - توابع فعال‌سازی در انتهای واحد پنهان برای معرفی پیچیدگی غیر خطی به مدل استفاده می‌شوند. در اینجا رایج‌ترین آنها نمایش داده شده است:

سیگموئید	تانژانت هذلولوی	یکسو ساز	یکسو ساز نشتی‌دار
$g(z) = \frac{1}{1 + e^{-z}}$	$g(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$	$g(z) = \max(0, z)$	$g(z) = \max(\epsilon z, z)$ و $\epsilon \ll 1$



خطای آنتروپی متقاطع - در مضمون شبکه‌های عصبی، عموماً از تابع خطای آنتروپی متقاطع  $L(z, y)$  استفاده می‌شود که به صورت زیر تعریف می‌شود:

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

نرخ یادگیری - نرخ یادگیری اغلب با نماد  $\alpha$  و گاهی اوقات با نماد  $\eta$  نمایش داده می‌شود و بیانگر سرعت (گام) بروزرسانی وزن‌ها است که می‌تواند مقداری ثابت یا به سازگار شونده تغییر کند. محبوب‌ترین روش حال حاضر Adam نام دارد، متدی است که نرخ یادگیری را در حین فرآیند آموزش تنظیم می‌کند.

انتشار معکوس - انتشار معکوس روشی برای بروزرسانی وزن‌ها با توجه به خروجی واقعی و خروجی مورد انتظار در شبکه‌ی عصبی است. مشتق نسبت به وزن  $W$  توسط قاعده‌ی زنجیری محاسبه می‌شود و به شکل زیر است:

$$\frac{\partial L(z, y)}{\partial w} = \frac{\partial L(z, y)}{\partial a} \times \frac{\partial a}{\partial z} \times \frac{\partial z}{\partial w}$$

در نتیجه، وزن به صورت زیر بروزرسانی می‌شود:

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

بروزرسانی وزن‌ها - در یک شبکه‌ی عصبی، وزن‌ها به صورت زیر بروزرسانی می‌شوند:

• گام ۱: یک دسته از داده‌های آموزشی را تهیه می‌کنیم.

• گام ۲: الگوریتم انتشار مستقیم را برای بدست آوردن خطای مربوطه اجرا می‌کنیم.

• گام ۳: خطا را انتشار معکوس می‌دهیم تا گرادیان‌ها به دست بیایند.

• گام ۴: از گرادیان‌ها برای بروزرسانی وزن‌های شبکه استفاده می‌کنیم.

برون‌اندازی - برون‌اندازی یک روش برای جلوگیری از بیش‌برازش بر روی داده‌های آموزشی با حذف تصادفی واحدها در یک شبکه‌ی عصبی است. در عمل، واحدها با احتمال  $p$  حذف یا با احتمال  $1 - p$  حفظ می‌شوند.

**شبکه‌های عصبی پیچشی ( کانولوشنی )**

الزامات لایه کانولوشنی - با نمایش  $W$  اندازه توده‌ی ورودی،  $F$  اندازه نورون‌های لایه‌ی کانولوشنی،  $P$  اندازه‌ی حاشیه‌ی صفر، تعداد نورون‌های  $N$  که در توده‌ی داده شده قرار می‌گیرند برابر است با:

$$N = \frac{W - F + 2P}{S} + 1$$

نرمال‌سازی دسته‌ای - یک مرحله از فراعامل‌های  $\gamma$  و  $\beta$  که دسته‌ی  $\{x_i\}$  را نرمال می‌کند در زیر آمده است. نماد  $\mu_B$  و  $\sigma_B^2$  به میانگین و واریانس دسته‌ای که می‌خواهیم آن را اصلاح کنیم اشاره دارد که به صورت زیر است:

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

معمولاً بعد از یک لایه‌ی تمام‌متصل یا لایه‌ی کانولوشنی و قبل از یک لایه‌ی غیرخطی اعمال می‌شود و امکان استفاده از نرخ یادگیری بالاتر را می‌دهد و همچنین باعث می‌شود که وابستگی شدید مدل به مقداردهی اولیه کاهش یابد.

## شبکه‌های عصبی بازگشتی

انواع دروازه‌ها - انواع مختلف دروازه‌هایی که در یک شبکه‌ی عصبی بازگشتی معمولی به آنها برمی‌خوریم در زیر آمده‌اند:

دروازه‌ی خروجی	دروازه	دروازه‌ی فراموشی	دروازه‌ی ورودی
چه مقدار برای سلول آشکار کند؟	چه مقدار در سلول بنویسد؟	سلول را پاک کند یا خیر؟	در سلول بنویسد یا خیر؟

**LSTM** - یک شبکه‌ی حافظه‌ی کوتاه-مدت طولانی (LSTM) یک نوع از مدل‌های RNN است که مشکل ناپدید شدن (صفر شدن) گرادیان را با اضافه کردن «دروازه‌ی فراموشی» حل می‌کند.

## یادگیری تقویتی و کنترل

هدف یادگیری تقویتی برای یک عامل این است که یاد بگیرد در یک محیط چگونه تکامل یابد.

### تعاریف

فرایندهای تصمیم‌گیری مارکوف - یک فرآیند تصمیم‌گیری مارکوف (به اختصار MDP) شامل پنج‌تایی  $(S, A, \{P_{s,a}\}, \gamma, R)$  است به طوری که:

- $S$  مجموعه‌ی حالات است

- $A$  مجموعه‌ای از کنش‌ها است

- $\{P_{s,a}\}$  احتمالات انتقال وضعیت برای هر  $a \in A$  و  $s \in S$  هستند.

•  $\gamma \in [0, 1]$  ضریب تخفیف است.

•  $R : \mathcal{S} \rightarrow \mathbb{R}$  یا  $R : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$  تابع پاداشی است که الگوریتم سعی دارد آن را پیشینه بکند.

خطمشی - یک خطمشی  $\pi$  تابعی است  $\pi : \mathcal{S} \rightarrow \mathcal{A}$  که حالات را به کنش‌ها نگاشت می‌کند.

نکته: می‌گوییم ما در حال اجرای خطمشی  $\pi$  هستیم اگر به ازای وضعیت  $s$  کنش  $a = \pi(s)$  را اجرا کنیم.

تابع ارزش - برای سیاست  $\pi$  و وضعیت  $s$ ، تابع ارزش  $V^\pi$  را به صورت زیر تعریف می‌کنیم:

$$V^\pi(s) = E \left[ R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \dots \mid s_0 = s, \pi \right]$$

معادله‌ی بلمن - معادله‌ی بلمن بهینه‌ی تابع ارزش  $V^{\pi^*}$  مربوط به خطمشی بهینه‌ی  $\pi^*$  را مشخص می‌کند:

$$V^{\pi^*}(s) = R(s) + \max_{a \in \mathcal{A}} \gamma \sum_{s' \in \mathcal{S}} P_{sa}(s') V^{\pi^*}(s')$$

نکته: سیاست بهینه‌ی  $\pi^*$  برای وضعیت  $s$  به این صورت است که:

$$\pi^*(s) = \operatorname{argmax}_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} P_{sa}(s') V^*(s')$$

الگوریتم تکرار ارزش - الگوریتم تکرار ارزش دو گام دارد:

(۱) ارزش را مقداردهی اولیه می‌کنیم:

$$V_0(s) = 0$$

(۲) ارزش را با توجه به ارزش‌های قبلی تکرار می‌کنیم:

$$V_{i+1}(s) = R(s) + \max_{a \in \mathcal{A}} \left[ \sum_{s' \in \mathcal{S}} \gamma P_{sa}(s') V_i(s') \right]$$

یادگیری **Q** - یادگیری Q نوعی از یادگیری تقویتی بدون مدل برای تخمین Q است که به صورت زیر انجام می‌شود:

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left[ R(s, a, s') + \gamma \max_{a'} Q(s', a') - Q(s, a) \right]$$