# تمرین سری 6

Deep Learning

بکتاش انصاری

99521082

---

سوال۱)

**برای تشخیص گربه بودن و نبودن:**
مدل کانوولوشنی برای عکس گربه سمت چپ عملکرد بسیار خوبی خواهد داشت. چون مدل کانوولوشنی میتواند به راحتی فیچر های عکس گربه مانند گوش و ... را تشخیص دهد. برای عکس سمت راست نیز که انگار با اعمال یک فیلتر لبه یاب بر روی عکس گربه است. لایه کانولوشنی میتواند نتیجه خیلی مطلوبی نسبت به عکس سمت چپ نداشته باشد. دلیل آن هم این است که اگر ما سعی کنیم نتیجه یادگیری فیلتر های لایه های کانوولوشنی را برای عکس سمت چپ بصورت بصری نمایش دهیم میتوانیم متوجه شویم که خیلی از فیلترها چیزهایی را که یاد گرفته‌اند شامل رنگ های مختلف در نواحی مختلف است که عکس راست دارای این فیچرها نیست و عملا

فیلتر های کانوولوشنی نمیتوانند به خوبی عکس سمت چپ برای عکس سمت راست فیچر یاد بگیرند.
برای مدل مبتنی بر attention نیز یکی از ویژگی های آن این است که فیچرهای درآمده از عکس بدلیل ویژگی توجه به خودی که مدل دارد میتواند پترن‌ها و فیچرهای پیچیده را نیز به خوبی یاد بگیرد. نتیجه این مدل برای عکس سمت چپ که رنگی است خوب خواهد بود ( مانند کانوولوشنی) و برای عکس سمت راست نیز بهتر از کانوولوشنی خواهد بود بدلیل اینکه میتواند پترن های سخت تر که لوکال هم نیستند را به خوبی یاد بگیرد.

**برای مسئله انسان بودن یا نبودن:**
برای شکل سمت چپ که هر کدام از اندام ها در جای مناسب قرار دارند هر دو مدل میتواند عملکرد خوبی داشته باشد و تشخیص دهد که این چهره، چهره‌ی یک انسان است. اما برای عکس سمت راست که مکان اندام ها جابجا شده نتیجه attention مطلوب‌تر خواهد بود. زیرا مدل‌های attention میتوانند فیچرها و پترن‌هایی که فاصله زیاده از هم دارند را نیز به خوبی یاد بگیرند. ( بدلیل ویژگی self attention). و این باعث میشود که با اینکه مثلا دو چشم در کنار هم نیستند بتواند ارتباط و توجهی بین آن ها ایجاد کند و فیچر مناسب را استخراج کند. اما مدل کانوولوشنی بدلیل آنکه تنها میتواند فیچرهای محلی و local را بخوبی یاد بگیرد، نمیتواند نتیجه خیلی مطلوبی داشته باشد.

سوال۲)

الف)

برای محاسبه برخی معیارها مانند precision و Recall از این مقادیر استفاده میشوند که هرکدام عبارت‌اند از:
مورد اول FP: که به معنای False Positive است یعنی مدل کلاس ورودی را مثبت پیشبینی کرده در صورتی که این پیشبینی غلط است و کلاس درست منفی است.
مورد دوم TP: که به معنای True Positive است یعنی مدل کلاس ورودی را مثبت پیشبینی کرده و کلاس درست نیز مثبت است و پیشبینی مدل درست بوده است.
مورد سوم FN: که به معنای False Negative است یعنی مدل کلاس ورودی را منفی پیشبینی کرده که پیشبینی غلطی است و کلاس درست مثبت است.
مورد چهارم TN: که به معنای True Negative است یعنی مدل کلاس ورودی را منفی پیشبینی کرده که پیشبینی  درستی است و کلاس درست منفی است.

ب)

در این سوال که اشتباه تشخیص دادن یک شخص بی‌گناه به عنوان مجرم یا تشخیص ندادن یک فرد مجرم میتواند خیلی هزینه زیادی برای ما داشته باشد. تنها accuracy به عنوان معیار، مناسب نیست. زیرا برای ما مهم است که مدل بتواند از بین تمام دزد هایی که پیشبینی کرده است. به طور تدریجی دزد های واقعی را تشخیص دهد. ( یعنی آنهایی که دزد تشخیص داده و در اصل دزد نبوده اند را تشخیص دهد.) و همینطور از بین تمام دزدها توانسته چند تا دزد را تشخیص دهد. این معیارها توسط Precision و Recall برای ما مشخص میشوند. که ما باید بتوانیم یک trade off مناسبی بین این دو برای خودمان ایجاد کنیم. که میتوانیم به عنوان یک معیار مناسب از F1 score استفاده کنیم که شامل دوبرابر ضرب این دو مقدار تقسیم بر جمع آنها میباشد.

---

سوال۳)

الف)

در برخی روش‌ها که به آنها self supervised learning میگوییم قبل از آنکه مدل را برای تسک‌هایی مثل classification استفاده کنیم. ابتدا از دیتای موجود

بدون استفاده از لیبل سعی میکنیم فیچرهایی را استخراج کنیم و وزن‌های اولیه‌ای را بسازیم. یکی از این روش‌های SSL روش تخمین زاویه چرخش عکس است. به این شکل که از عکس یک عکس چرخش یافته ایجاد میکنه و از مدل میخواهیم تخمین بزند که این عکس نسبت به عکس جدید چه مقدار چرخیده است. مدل با آموزش بر روی این مسئله میتواند فیچرهای خیلی خوبی را یاد بگیرد و وزن‌های اولیه‌ای را بسازد. حال مدل اصلی خودمان را که وظیفه classification دارد با این وزن‌های اولیه ( pretraining weights) اجرا میکنیم تا آموزش ببیند. این کار باعث میشود مدل از یک موقعیت خوبی شروع کند و همگرایی را سریع کند و دقت را بالا ببرد.

ب)

یکی از روش های تولید embedding از روی توکن‌ها میباشد. که ابتدا بردارهایی با سایز کل توکن‌های ورودی میسازیم و سپس موقعیت هر توکن را در بردار embedding آن یک میکنیم و بقیه ایندکس‌ها را صفر میذاریم.
برای مثال اگر ما 100 توکن داشته باشیم. و کلمه سلام توکن دوم باشد. بردار آن شامل 100 مقدار خواهد بود که همگی به جز مقدار دوم صفراند و مقدار دوم یک است.
یکی از مشکلات آن این است که فاصله بین تمام توکن ها یکی است.

یعنی در اصل برای embedding ما انتظار داریم فاصله برداری هر دو embedding رابطه مستقیمی با شباهت معنایی دو توکن داشته باشد. که این انتظار در این روش برآورده نمیشود.

ج)

یکی دیگر از روش‌های ایجاد امبدینگ word2vec است. که یکی از روش‌های آن skip-gram میباشد. که در این روش ما در هر مرحله یکی از کلمات داخل متن را به صورت رندوم mask میکنیم و به عنوان target در نظر میگریم. و یک کلمه رندوم دیگر را به عنوان context در نظر میگیریم. و از مدل میخواهیم با توجه به توکن context پیشبینی کند که کلمه درست در جایگاه traget چه مقداری است. ما با این کار عملا بدون آنکه مسئله خاصی به مدل بدهیم و یا دیتای ما لیبل خاصی داشته باشد. تنها از مدل میخواهیم که به توجه به جایگاه کلمات در متن از روی آن‌ها فیچر استخراج کند. که مدل با این نوع آموزش میتواند برای هر کلمه وزن‌هایی را که همان فیچرها هستند ایجاد کند که هر وزن embedding آن کلمه خواهد بود. که این روش یک روش self-supervised است.

---

سوال۴)

الف)

یکی از روش هایی که در یک مقاله برای جست و جوی ابر پارامترهای معماری شبکه پیشنهاد شده است روش NAS است. در این روش چندین ماژول وجود دارد که آنها را توضیح میدهم.

ماژول اول trainer میباشد که آن به عنوان ورودی ابرپارامترهای مدل میگیرد و آن مدل را آموزش میدهد و از مدل آموزش داده شده یک دقت بر روی داده‌های eval میگیرد و خروجی میدهد. حال این مقدار دقت به عنوان Reward در نظر گرفته میشود و به عنوان ورودی به ماژول control داده میشود.

ماژول دوم controller میباشد که مقدار Reward را گرفته و بر اساس یک الگوریتم RL یک action ای را انجام میدهد. که این action شامل ابرپارامتر های جدید است.  در اصل action space ما لیستی از توکن‌هایی است که به عنوان ابرپارامتر در trainer استفاده میشود.

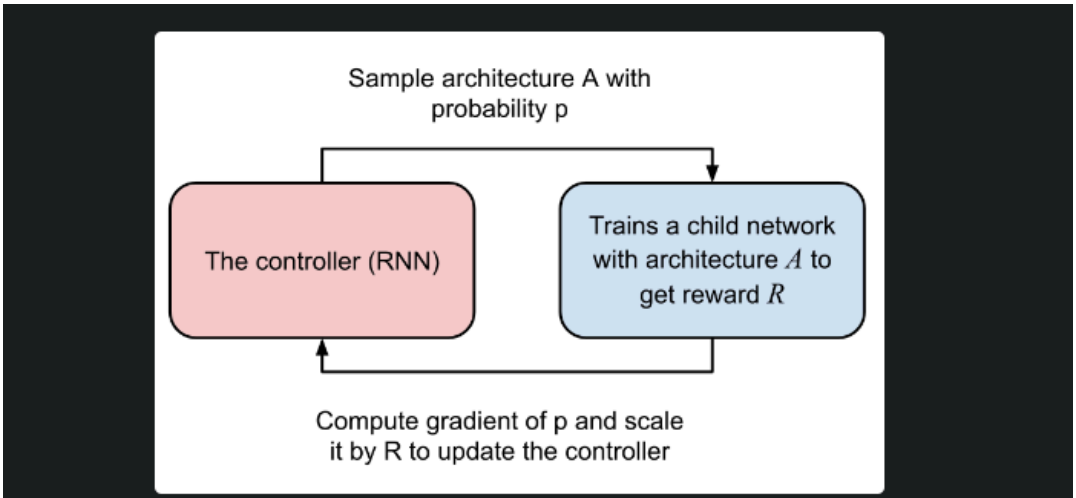


Fig. 8. A high level overview of NAS, containing a RNN controller and a pipeline for evaluating child models. (Image source: Zoph & Le 2017)

و controller توسط یک loss که یک RL loss میباشد. بهینه میشود.

$$\nabla_\theta J(\theta) = \sum_{t=1}^{T} \mathbb{E}[\nabla_\theta \log P(a_t | a_{1:(t-1)}; \theta) R]$$

یکی دیگر از روش‌هایی که میتوان به آن اشاره کرد که از RL استفاده کرده است.

MetaQNN

است که از Q-learning استفاده کرده است.

ب)

با توجه به روش گفته شده (NAS) در قسمت الف ابرپارامتر های مربوط به معماری شبکه عصبی میتوانند بهینه شوند تا مدل به دقت بهتری برسد. پس ما میتوانیم تعداد لایه های مدل را به عنوان یک ابرپارامتر به NAS بدهیم. اما تصویر ورودی یک ابرپارامتر نیست و نمیتوان آن را ابرمارامتر معماری شبکه در نظر بگیریم.

---

سوال۵)

از آنجایی که این دو مدل ( مدل مولد و مدل ممیز ) با هم از ابتدا شروع به آموزش میکنند و هر دو در ابتدا آموزش ندیده و ضعیف هستند و بر اساس توابع loss که مربوط به یکدیگر است. ( یعنی مدل مولد بر اساس مدل ممیز و مدل ممیز بر اساس مدل مبدل) همدیگر را بهبود میبخشند. ممکن است در epoch اول تابع loss ان ها مقدار x شود چون هر دو مدل ضعیف اند یک عکس ضعیف تولید کنند. و در epoch صدم چون هر دو مدل قوی اند loss آن ها همان x شوند اما چون هر دو آموزش دیده‌اند loss محاسبه شده براساس نتیجه یکدیگر است اما خروجی مبدل یک خروجی بسیار بهتری از epoch اول است.
یعنی در این مدل مقدار loss کمتر یا بیشتر لزوما به معنای بهتر بودن یا بدتر بودن خروجی نیست. و این رقابت و زمان آموزش است که باعث بهبود تدریجی میشود.

چون مقدار loss براساس یک معیار ثابت نیست و معیارها هی در حال تغییر است.

اگر به فرمول loss نگاه کنیم:

$$\min_{\theta_g} \max_{\theta_d} \left[ \mathbb{E}_{x \sim p_{data}} \log \underbrace{D_{\theta_d}(x)}_{\text{Discriminator output for real data x}} + \mathbb{E}_{z \sim p(z)} \log(1 - \underbrace{D_{\theta_d}(G_{\theta_g}(z))}_{\text{Discriminator output for generated fake data G(z)}}) \right]$$

در این فرمول خروجی بر اساس نتیجه تابع ممیز تولید میشود اما نکته اینجا است که خود تابع ممیز در حال بهبود است و برای عکس های بهتر لزوما loss کمتر نمیدهد. چون خودش هم سخت‌گیرانه تر عمل میکند.