پاسخ تمرین سوم **مهدیه نادری: ۹۸۵۲۲۰۷۶**

سوال اول) بهینه سازها

۱.الف مشکلات نرخ یادگیری بسیار بالا و روش تشخیص:

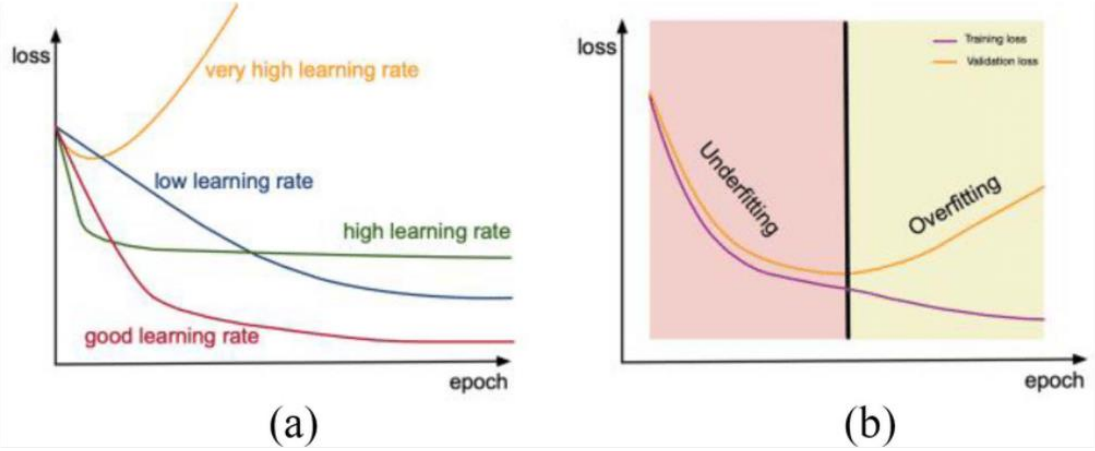
شبکه‌های عصبی یادگیری عمیق با استفاده از الگوریتم بهینه‌سازی گرادیان نزولی تصادفی آموزش داده می‌شوند.نرخ یادگیری یک هایپرپارامتر است که کنترل می‌کند هر بار که وزن‌های مدل به‌روزرسانی می‌شوند چقدر مدل را در پاسخ به خطای تخمینی تغییر دهیم. انتخاب نرخ یادگیری چالش برانگیز است زیرا یک مقدار بسیار کوچک ممکن است منجر به یک فرآیند آموزشی طولانی شود که ممکن است گیر کند، در حالی که ***یک مقدار بیش از حد بزرگ ممکن است منجر به یادگیری سریع مجموعه ای از وزنه های غیربهینه یا یک فرآیند آموزشی ناپایدار*** شود.

تاثیر نرخ یادگیری را می‌توان از طریق مقدار انتخابی برای آن سنجید.مقادیر کوچک‌تر آن به دوره‌های آموزشی بیشتری نیاز دارند، با توجه به تغییرات کوچک‌تری که در وزن‌ها در هر به‌روزرسانی ایجاد می‌شود، در حالی که مقادیر بزرگ‌تر منجر به تغییرات سریع و نیاز به دوره‌های آموزشی کمتری می‌شود.مقادیر خیلی زیاد نیز می‌تواند باعث شود که مدل خیلی سریع به یک راه‌حل غیربهینه همگرا شود، در حالی که مقادیر بسیار کوچک می‌تواند باعث گیرکردن فرآیند شود.

همان طور که در شکل می بینیم، در حالتی که نرخ یادگیری بالا باشد (نمودار سبز) خیلی سریع به یک نقطه بهینه می رسیم و از آن به بعد آموزش متوقف می شود.

همچنین نرخ یادگیری پایین (نمودار آبی) باعث می شود یادگیری با سرعت خیلی کمی انجام شود.

اگر نرخ یادگیری خیلی خیلی زیاد باشد، اصلا یادگیری صورت نمی گیرد و نمودار زرد رنگ را خواهیم داشت.



با ذخیره سازی مقدار loss و کشیدن نمودار آن بر حسب ایپاک میتوانیم بفهمیم نرخ یادگیری را خوب تنظیم کردیم یا نه یعنی اگر مقدار هزینه دائم در حال کاهش نباشد (نوسان داشته باشد) باید آن را کاهش دهیم.

استفاده از نرخ یادگیری بسیار بالا در شبکه‌های عصبی نیز ممکن است با مشکلاتی مواجه شود که این مشکلات ممکن است در عملکرد و کارایی شبکه تأثیرگذار باشند. برخی از این مشکلات عبارتند از:

1. **عدم استقراری :**نرخ یادگیری بسیار بالا ممکن است منجر به عدم پایداری و استقرار شبکه شود. شبکه ممکن است به سرعت به مقادیر بیش از اندازه‌ای برسد که موجب ناپایداری در آموزش و عملکرد آن می‌شود.
2. **پیچیدگی و مدل‌سازی ناصحیح :**استفاده از نرخ یادگیری بسیار بالا ممکن است باعث بیش‌برازش (overfitting) شود. این به معنای این است که شبکه، الگوهای خاص داده‌های آموزشی را به خوبی یاد می‌گیرد، اما نمی‌تواند به خوبی برای داده‌های جدید و ناشناخته عمل کند.
3. **پیش‌پردازش نادرست داده :**سرعت بالای یادگیری ممکن است باعث بشود که داده‌ها به نحو نادرستی پردازش شوند یا مدل به داده‌های نویزی و نامرتبط واکنش نشان دهد.
4. **ضربه‌های گذرای :**در صورتی که نرخ یادگیری بسیار بالا باشد، ممکن است شبکه به صورت ناگهانی واکنش نشان دهد که باعث گذر از نقاط بهینه و به وجود آمدن نوسانات غیرمطلوب در فرایند یادگیری شود.

برای تشخیص این مشکلات و رفع آن‌ها می‌توانید به روش‌های زیر توجه کنید:

1. **مشاهده نمودارهای آموزش و ارزیابی :**مشاهده نمودارهای مربوط به دقت شبکه در داده‌های آموزش و ارزیابی می‌تواند کمک کننده باشد. اگر دقت در داده‌های آموزش بسیار بالاست ولی در داده‌های ارزیابی کم است، این ممکن است نشانه‌ای از بیش‌برازش باشد.
2. **استفاده از تکنیک‌های کاهش نرخ یادگیری :**استفاده از تکنیک‌هایی مانند کاهش نرخ یادگیری به مرور زمان (learning rate decay)می‌تواند کمک کننده باشد تا شبکه به صورت تدریجی به نقاط بهینه برسد و از واکنش‌های ناگهانی جلوگیری کند.
3. **استفاده از اعتیاد به داده (data augmentation) و روش‌های تقویت داده :**استفاده از روش‌هایی که داده‌ها را تقویت و بهبود می‌دهند می‌تواند کمک کننده باشد تا شبکه بهتر و پایدارتر آموزش داده شود.
4. **استفاده از شبکه‌های با ساختار ساده‌تر :**استفاده از شبکه‌های با ساختار ساده‌تر و کنترل شده‌تر می‌تواند از بیش‌برازش جلوگیری کند.

به طور کلی، تعیین نرخ یادگیری مناسب و انجام آزمایش‌های مختلف بر روی شبکه می‌تواند بهترین راه برای یافتن تنظیمات بهینه و جلوگیری از مشکلات مربوط به نرخ یادگیری بسیار بالا باشد.

۱.ب مشکلات نرخ یادگیری بسیار پایین:

استفاده از نرخ یادگیری بسیار پایین در شبکه‌های عصبی می‌تواند به مشکلاتی منجر شود که ممکن است در ترجمه داده‌ها یا آموزش مدل‌های یادگیری عمیق باشد. برخی از مشکلاتی که ممکن است به وجود بیایند عبارتند از:

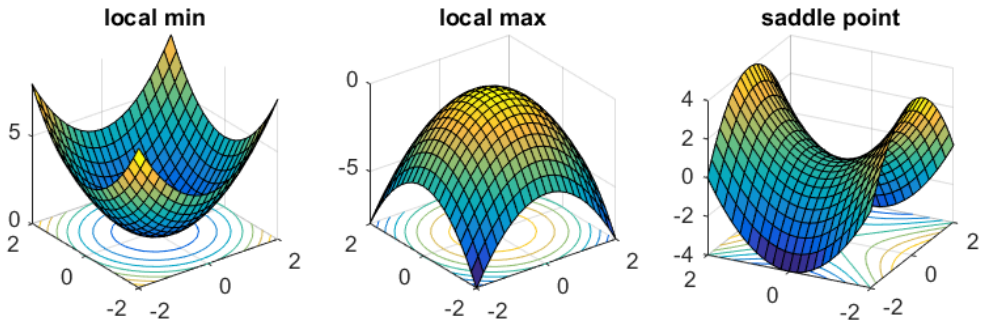
1. **کندی در آموزش مدل** :استفاده از نرخ یادگیری بسیار کم می‌تواند باعث کندی در فرآیند آموزش شبکه‌های عصبی شود. این موضوع می‌تواند منجر به افزایش زمان مورد نیاز برای آموزش مدل و یا گیر کردن مدل در مینیمم‌های محلی شود.
2. **عدم هم convergence**: نرخ یادگیری بسیار کم ممکن است باعث شود که مدل به سرعت به یک نقطه توقف در آموزش برسد و دیگر به سمت بهبود نرفته و در جایگاهی متمرکز شود که بهبود دادن کارایی مدل را دشوار می‌کند.
3. **عدم دستیابی به جواب بهینه** :استفاده از نرخ یادگیری بسیار پایین ممکن است باعث شود که مدل به جواب بهینه‌ای برای مسئله‌ای که در حال یادگیری است، نرسد یا به سختی به آن نزدیک شود.

برای تشخیص این مشکلات، می‌توانید از روش‌های زیر استفاده کنید:

1. **نظارت بر عملکرد مدل** :بررسی عملکرد مدل با معیارهای مختلفی مانند دقت، خطا، و کارایی در داده‌های ارزیابی. اگر کارایی مدل به طور مداوم بهبود نیابد، ممکن است نرخ یادگیری مناسب نباشد.
2. **رصد نمودارهای آموزش** :بررسی نمودارهای آموزش شبکه‌ی عصبی از جمله تغییرات در تابع هزینه (loss function) یا دقت مدل. نمودارهایی که به دلیل نرخ یادگیری پایین، تغییرات آنها به سمت جمود می‌روند، نشان‌دهنده مشکلات می‌باشند.
3. **تغییر نرخ یادگیری** :امتحان کردن نرخ‌های یادگیری مختلف و بررسی عملکرد مدل با آنها. اغلب برای یافتن بهترین نرخ یادگیری، ترکیبی از آزمون و خطا و استفاده از تکنیک‌های بهینه‌سازی مختلف استفاده می‌شود.

۱.پ نقطه زینی و بررسی آن در SGD و Adam :

نقطه زینی (Saddle point) یک نقطه در فضای تابع هدف یا تابع هزینه است که در آن، مشتقات جزئی تابع نسبت به متغیرها صفر است، اما در جهات مختلف، تابع هدف همزمان در نقطه زینی یا به سمت بالا (به عنوان اکسترمم محلی بیشینه) می‌رود و یا به سمت پایین (به عنوان اکسترمم محلی کمینه) می‌رود. این نقاط به عنوان یک نقطه کندوکاو برای الگوریتم‌های بهینه‌سازی می‌توانند باعث مشکلات در جستجوی بهینه‌سازی شوند. به عبارت دیگر، در اطراف یک نقطه زینی، گرادیان به صفر می‌رسد اما به دلایل هندسی یا ترکیبی از مقادیر مثبت و منفی در جهات مختلف، تابع هدف همگرا به کمینه یا بیشینه نمی‌شود و به جایی میانی ایستاده می‌شود.



الگوریتم بهینه‌سازی Adam در مواجهه با نقاط زینی (saddle points) دارای رفتار خاصی است. این الگوریتم از ترکیبی از روش‌های Momentumو RMSprop برای بهینه‌سازی استفاده می‌کند و در مواجهه با نقاط زینی می‌تواند رفتار متفاوتی داشته باشد:

1. **عکس‌العمل Adam در مواجهه با نقطه زینی:**
   - Adamبه دلیل استفاده از مفهوم میانگین مربعات گرادیان‌ها(Root Mean Square of gradients) ، در مواجهه با نقاط زینی ممکن است از جایگاه محلی خارج شده و به سمت کمینه محلی یا بهینه‌تر حرکت کند.
   - با استفاده از مفهوم گرادیان‌های مربعی(squared gradients) ، Adamمی‌تواند از نقطه زینی خارج شود و سریع‌تر به سمت بهینه محلی حرکت کند.
2. **مزایا و معایب:Adam**
   - **مزایا:**
     - Adamاز ترکیبی از تکنیک‌های Momentum و RMSprop استفاده می‌کند که می‌تواند به سرعت به نقاط کمینه هدایت شود.
     - این الگوریتم دارای نرخ یادگیری متغیر برای هر پارامتر است که می‌تواند به کارایی بهتر و سرعت بیشتر در آموزش منجر شود.
     - Adamدارای پارامترهایی است که به مدل امکان تنظیم بهتری را می‌دهد.
   - **معایب:**
     - ممکن است برای دیتاست‌های کوچک وجود نرخ یادگیری متغیر، باعث کاهش دقت و کارایی مدل شود.
     - برای تنظیم پارامترهای Adam نیاز به تجربه و تنظیم دقیق دارد.

در کل، Adamبه عنوان یک الگوریتم بهینه‌سازی مواجهه با نقاط زینی را با استفاده از میانگین مربعات گرادیان‌ها، معمولاً به سمت بهینه محلی حرکت می‌کند و از مزایای افزایش سرعت و کارایی در فرآیند آموزش برخوردار است. اما نیاز به تنظیمات دقیق و مناسب و نقطه زینی‌های خاص برای بهترین عملکرد ممکن دارد.

الگوریتم بهینه‌سازی Stochastic Gradient Descent (SGD) در مواجهه با نقاط زینی (saddle points) عکس‌العمل خاصی دارد. واقعیت این است که عملکرد SGD در این نقاط می‌تواند متفاوت باشد و ممکن است بهبود مدل در این نقاط را کاهش یا متوقف کند.

1. **عکس‌العمل SGD در مواجهه با نقطه زینی:**
   - SGDممکن است در نقاط زینی گیر کند و به دلیل گرادیان صفر، در این نقاط نتواند به سمت کمینه حرکت کند.
   - در برخی موارد، SGDممکن است به سمت یکی از جهت‌های گرادیان حرکت کند اما نتواند به کمینه محلی برسد.
2. **مزایا و معایب:SGD**
   - **مزایا:**
     - SGDساده و آسان برای پیاده‌سازی است و برای دیتاست‌های بزرگ می‌تواند بهینه باشد.
     - ممکن است در نقاطی که گرادیان غیرصفر است، به سرعت به سمت کمینه محلی حرکت کند.
   - **معایب:**

- مشکل اصلی SGD در نقاط زینی است. ممکن است در این نقاط گیر کند و نتواند به سمت بهینه محلی حرکت کند.
- SGDممکن است نیاز به تنظیم نرخ یادگیری داشته باشد و وابستگی بیشتری به این پارامتر داشته باشد.

در کل، SGDبه عنوان یک الگوریتم ساده بهینه‌سازی می‌تواند در نقاطی که گرادیان غیرصفر دارد به سرعت به سمت کمینه محلی حرکت کند، اما در مواجهه با نقاط زینی ممکن است به مشکل برخورد کرده و در جستجوی بهینه گیر کند. تنظیمات مناسب نرخ یادگیری و استفاده از تکنیک‌های دیگری مانند momentum می‌تواند کمک کننده باشد تا این مشکلات را حل کرد و عملکرد الگوریتم را بهبود بخشید.

۱.ت تفاوت کاهش هزینه شیب نزولی دسته ای و شیب نزولی دسته ای کوچک:

شکل سمت چپ: نمودار کاهش هزینه شیب نزولی دسته ای: چون به نسبت هموار تراست. یعنی میانگین کل داده‌ها روی آپدیت تاثیر داشتند.

شکل سمت راست: کاهش هزینه شیب نزولی دسته ای کوچک: چون در هر ایپاک تعدادی از داده ها تاثیر داشتند و با توجه به رندم بودن انتخاب آنها میانگین هزینه ها مقداری بالا پایین میشود اما درکل کاهش مییابد.

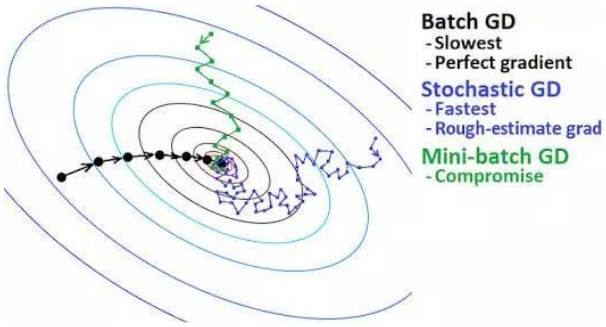
Batch Gradient descent :

در این الگوریتم روش و رویکرد بر مبنای بررسی تمام خطا ها در همه ی داده های آزمایشی است و زمانی مدل به روز رسانی و بهینه میشود که این محاسبات بر روی تمامی داده های آموزشی انجام شده باشد یعنی اگر تعداد داده های بالایی داشته باشیم باید همه ی داده ها را در حالت train به شبکه بدهیم بعد از آن دوباره مدل را بروز کنیم این صرفا یک پیچیدگی محاسباتی شدید برای ما ایجاد میکنه که در شکل زیر مشخص کرده ام.و علاوه بر این در این حالت فقط ما داده ها رو یکبار با محاسبه سنگین اپتیمایز کردیم و خب به صرفه نیست .

$$\text{for n} = 1 : \text{N}$$

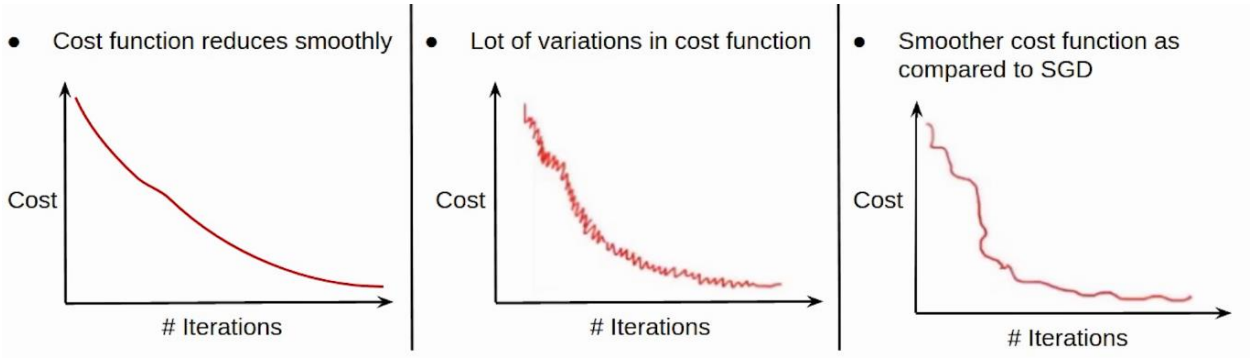
$$w_i^n := w_i^{n-1} - \eta \frac{\partial E_n(\text{w})}{\partial w_i}$$

Mini batch Gradient descent :



در این روش به جای اینکه بیایم و کل نمونه هارو بدیم یا یک دونه نمونه آموزش بدیم میایم و در هر دفعه یک Batch از داده هارو اموزش میدیم. مثلا اگه کل داده هامون N تاست میایم و ۲۰۰ تا ۲۰۰ تا اموزش میدیم .اینجا دیگه نگرانی نویزی شدن مسیرمون رو هم نداریم چون دیگه روی یک عدد نمونه حساب نمیکنیم و مطمعن تر داریم میریم جلو. که در شکل بالا هم تفاوت هایش مشخص شده است.

در نمودار های زیر مقایسه کامل تری داریم:

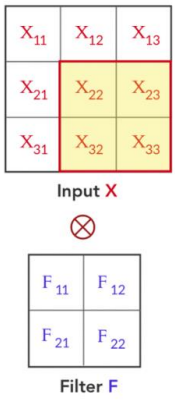


پاسخ تمرین سوم **مهدیه نادری: ۹۸۵۲۲۰۷۶**

در حالت Gradient Descent دسته‌ای (Batch GD) پارامترها را با استفاده از کل مجموعه داده به‌روزرسانی می‌کنیم، تابع هزینه در این حالت به آرامی کاهش پیدا می‌کند.
این به‌روزرسانی در حالت Stochastic GD (SGD) چنین آرام و صاف نیست. زیرا که ما پارامترها را بر اساس یک مشاهده به‌روز می‌کنیم، تعداد زیادی تکرار صورت می‌گیرد. همچنین ممکن است مدل شروع به یادگیری اطلاعات ناهمخوان نیز کند.
به‌روزرسانی تابع هزینه در حالت Mini-batch Gradient Descent نسبت به SGD صاف‌تر است. چرا که ما پارامترها را پس از هر زیرمجموعه از داده‌ها به‌روز می‌کنیم و نه بلافاصله پس از هر مشاهده.
منبع :
https://stats.stackexchange.com/questions/488017/understanding-mini-batch-gradient-descent
https://www.analyticsvidhya.com/blog/2021/03/variants-of-gradient-descent-algorithm/
https://virgool.io/@danialfarsy/%D8%A8%D8%B1%D8%B1%D8%B3%DB%8C-%D9%88-%D9%85%D9%82%D8%A7%DB%8C%D8%B3%D9%87-batch-gradient-descentmini-batch-gradient-descentstochastic-gradient-descent-n4yklzivliiw
https://scitech.blogsky.com/1399/12/12/post-832/%D9%85%D8%B3%D8%A7%DB%8C%D9%84%DB%8C-%D8%A7%D8%B2-%D8%A7%DA%A9%D8%B3%D8%AA%D8%B1%D9%85%D9%85-%D9%86%D8%B3%D8%A8%DB%8C-%D9%88-%D9%85%D8%B7%D9%84%D9%82-%D8%AA%D9%88%D8%A7%D8%A8%D8%B9-%DA%86%D9%86%D8%AF-%D9%85%D8%AA%D8%BA%DB%8C%D8%B1%D9%87
https://datayad.com/neural-networks-ep9/
http://viraai.com/%D8%AA%D8%A7%D8%AB%DB%8C%D8%B1-%D9%86%D8%B1%D8%AE-%DB%8C%D8%A7%D8%AF%DA%AF%DB%8C%D8%B1%DB%8C-%D8%A8%D8%B1-%D8%B9%D9%85%D9%84%DA%A9%D8%B1%D8%AF-%D8%B4%D8%A8%DA%A9%D9%87-%D8%B9%D8%B5%D8%A8%DB%8C/

سوال دوم:

باید از فرمول های زیر برای محاسبه خروجی کانولوشن استفاده کنیم.

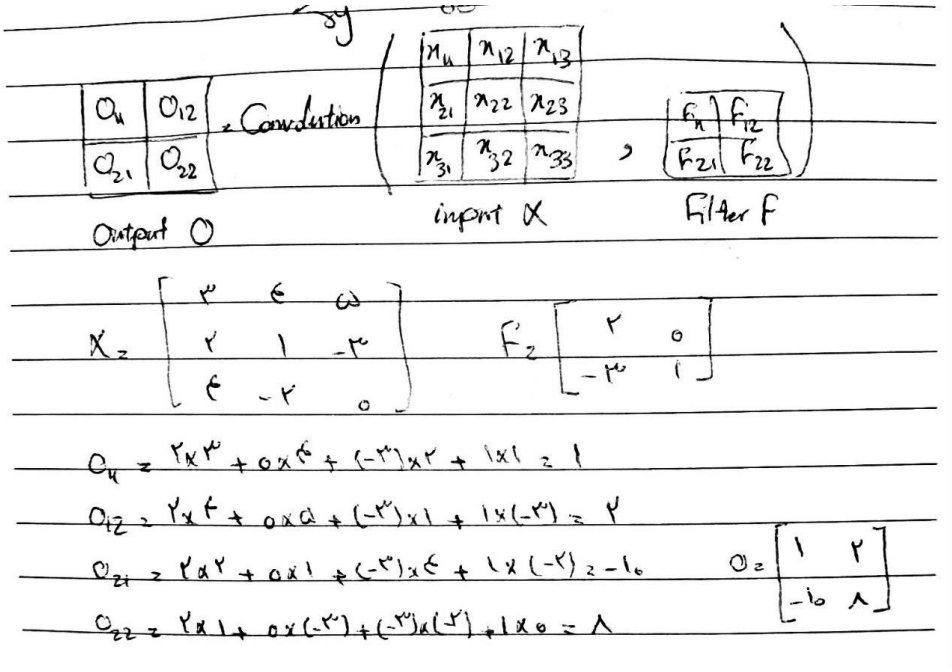


X_{11}	X_{12}	X_{13}
X_{21}	$X_{22}F_{11}$	$X_{23}F_{12}$
X_{31}	$X_{32}F_{21}$	$X_{33}F_{22}$

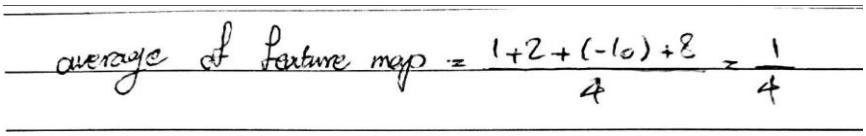
$$O_{11} = X_{11}F_{11} + X_{12}F_{12} + X_{21}F_{21} + X_{22}F_{22}$$
$$O_{12} = X_{12}F_{11} + X_{13}F_{12} + X_{22}F_{21} + X_{23}F_{22}$$
$$O_{21} = X_{21}F_{11} + X_{22}F_{12} + X_{31}F_{21} + X_{32}F_{22}$$
$$O_{22} = X_{22}F_{11} + X_{23}F_{12} + X_{32}F_{21} + X_{33}F_{22}$$

پاسخ تمرین سوم **مهدیه نادری: ۹۸۵۲۲۰۷۶**

خروجی لایه کانولوشنی را به صورت زیر حساب کردیم:



سپس باید Global Average Pooling step را روی خروجی حساب کنیم.



حالا نوبت پس انتشار خطا است.

شکل کلی شبکه عصبی به صورت زیر است.

ابتدا باید گرادیان نسبت به خروجی GAP را حساب کنیم.

در مشتق گیری از GAP چون تمام اعضا با هم جمع و سپس تقسیم بر ۴ شده اند. مشتق نسبت به هریک از اعضای O میشود ضریب آن و بقیه ی المنت ها مثل یک عدد ثابت درنظر گرفته میشوند و مشتقشان صفر میشود پس مشتق نسبت به هر کدام میشود ۰,۲۵ و چون ۴ عضو داریم یک ماتریس ۲در۲ که تمام اعضای آن ۰,۲۵ هستند خواهسم داشت.

برای محاسبه مشتق تابع لاس نسبت به O باید از قاعده chain rule استفاده کنیم و آن را در ۱ که صورت سوال گفته یعنی مشتق لاس نسبت به خروجی کل ضرب کنیم که چون ۱ است جواب همان میشود.

سپس گرادیان نسبت به X و نسبت به F را از فرمول های زیر بدست می آوریم. اثبات کامل آن در منبع گفته شده در سوال وجود دارد.

برای گرادیان نسبت به F از رابطه های زیر استفاده میکنیم و مشتق را نسبت به هر یک از المنت ها حساب میکنیم و در آخر کنار هم قرار میدهیم.

$$\frac{\partial L}{\partial F_{11}} = \frac{\partial L}{\partial O_{11}} * \frac{\partial O_{11}}{\partial F_{11}} + \frac{\partial L}{\partial O_{12}} * \frac{\partial O_{12}}{\partial F_{11}} + \frac{\partial L}{\partial O_{21}} * \frac{\partial O_{21}}{\partial F_{11}} + \frac{\partial L}{\partial O_{22}} * \frac{\partial O_{22}}{\partial F_{11}} \quad \frac{\partial L}{\partial F_{11}} = \frac{\partial L}{\partial O_{11}} * X_{11} + \frac{\partial L}{\partial O_{12}} * X_{12} + \frac{\partial L}{\partial O_{21}} * X_{21} + \frac{\partial L}{\partial O_{22}} * X_{22}$$

$$\frac{\partial L}{\partial F_{12}} = \frac{\partial L}{\partial O_{11}} * \frac{\partial O_{11}}{\partial F_{12}} + \frac{\partial L}{\partial O_{12}} * \frac{\partial O_{12}}{\partial F_{12}} + \frac{\partial L}{\partial O_{21}} * \frac{\partial O_{21}}{\partial F_{12}} + \frac{\partial L}{\partial O_{22}} * \frac{\partial O_{22}}{\partial F_{12}} \quad \frac{\partial L}{\partial F_{12}} = \frac{\partial L}{\partial O_{11}} * X_{12} + \frac{\partial L}{\partial O_{12}} * X_{13} + \frac{\partial L}{\partial O_{21}} * X_{22} + \frac{\partial L}{\partial O_{22}} * X_{23}$$

$$\frac{\partial L}{\partial F_{21}} = \frac{\partial L}{\partial O_{11}} * \frac{\partial O_{11}}{\partial F_{21}} + \frac{\partial L}{\partial O_{12}} * \frac{\partial O_{12}}{\partial F_{21}} + \frac{\partial L}{\partial O_{21}} * \frac{\partial O_{21}}{\partial F_{21}} + \frac{\partial L}{\partial O_{22}} * \frac{\partial O_{22}}{\partial F_{21}} \quad \frac{\partial L}{\partial F_{21}} = \frac{\partial L}{\partial O_{11}} * X_{21} + \frac{\partial L}{\partial O_{12}} * X_{22} + \frac{\partial L}{\partial O_{21}} * X_{31} + \frac{\partial L}{\partial O_{22}} * X_{32}$$

$$\frac{\partial L}{\partial F_{22}} = \frac{\partial L}{\partial O_{11}} * \frac{\partial O_{11}}{\partial F_{22}} + \frac{\partial L}{\partial O_{12}} * \frac{\partial O_{12}}{\partial F_{22}} + \frac{\partial L}{\partial O_{21}} * \frac{\partial O_{21}}{\partial F_{22}} + \frac{\partial L}{\partial O_{22}} * \frac{\partial O_{22}}{\partial F_{22}} \quad \frac{\partial L}{\partial F_{22}} = \frac{\partial L}{\partial O_{11}} * X_{22} + \frac{\partial L}{\partial O_{12}} * X_{23} + \frac{\partial L}{\partial O_{21}} * X_{32} + \frac{\partial L}{\partial O_{22}} * X_{33}$$

Derivatives of ∂L/∂F

$$\begin{bmatrix} \frac{\partial L}{\partial F_{11}} & \frac{\partial L}{\partial F_{12}} \\ \frac{\partial L}{\partial F_{21}} & \frac{\partial L}{\partial F_{22}} \end{bmatrix} = \text{Convolution} \left(\begin{bmatrix} X_{11} & X_{12} & X_{13} \\ X_{21} & X_{22} & X_{23} \\ X_{31} & X_{32} & X_{33} \end{bmatrix}, \begin{bmatrix} \frac{\partial L}{\partial O_{11}} & \frac{\partial L}{\partial O_{12}} \\ \frac{\partial L}{\partial O_{21}} & \frac{\partial L}{\partial O_{22}} \end{bmatrix} \right)$$

where

$$\begin{bmatrix} X_{11} & X_{12} & X_{13} \\ X_{21} & X_{22} & X_{23} \\ X_{31} & X_{32} & X_{33} \end{bmatrix} = \text{Input X} \qquad \begin{bmatrix} \frac{\partial L}{\partial O_{11}} & \frac{\partial L}{\partial O_{12}} \\ \frac{\partial L}{\partial O_{21}} & \frac{\partial L}{\partial O_{22}} \end{bmatrix} = \frac{\partial L}{\partial O} \text{ Loss gradient from previous layer}$$

∂L/∂F = Convolution of input matrix X and loss gradient ∂L/∂O

برای محاسبه گرادیان نسبت به X هم از فرمول های زیر استفاده میکنیم.

For every element of X_i

$$\frac{\partial L}{\partial X_i} = \sum_{k=1}^{M} \frac{\partial L}{\partial O_k} * \frac{\partial O_k}{\partial X_i}$$

$$\frac{\partial L}{\partial X_{11}} = \frac{\partial L}{\partial O_{11}} * F_{11}$$

$$\frac{\partial L}{\partial X_{12}} = \frac{\partial L}{\partial O_{11}} * F_{12} + \frac{\partial L}{\partial O_{12}} * F_{11}$$

$$\frac{\partial L}{\partial X_{13}} = \frac{\partial L}{\partial O_{12}} * F_{12}$$

$$\frac{\partial L}{\partial X_{21}} = \frac{\partial L}{\partial O_{11}} * F_{21} + \frac{\partial L}{\partial O_{21}} * F_{11}$$

$$\frac{\partial L}{\partial X_{22}} = \frac{\partial L}{\partial O_{11}} * F_{22} + \frac{\partial L}{\partial O_{12}} * F_{21} + \frac{\partial L}{\partial O_{21}} * F_{12} + \frac{\partial L}{\partial O_{22}} * F_{11}$$

$$\frac{\partial L}{\partial X_{23}} = \frac{\partial L}{\partial O_{12}} * F_{22} + \frac{\partial L}{\partial O_{22}} * F_{12}$$

$$\frac{\partial L}{\partial X_{31}} = \frac{\partial L}{\partial O_{21}} * F_{21}$$

$$\frac{\partial L}{\partial X_{32}} = \frac{\partial L}{\partial O_{21}} * F_{22} + \frac{\partial L}{\partial O_{22}} * F_{21}$$

$$\frac{\partial L}{\partial X_{33}} = \frac{\partial L}{\partial O_{22}} * F_{22}$$

Derivatives of ∂L/∂X using local gradients from Equation

$$\begin{bmatrix} \frac{\partial L}{\partial X_{11}} & \frac{\partial L}{\partial X_{12}} & \frac{\partial L}{\partial X_{13}} \\ \frac{\partial L}{\partial X_{21}} & \frac{\partial L}{\partial X_{22}} & \frac{\partial L}{\partial X_{23}} \\ \frac{\partial L}{\partial X_{31}} & \frac{\partial L}{\partial X_{32}} & \frac{\partial L}{\partial X_{33}} \end{bmatrix} = \text{Full Convolution} \left(\begin{bmatrix} F_{22} & F_{21} \\ F_{12} & F_{11} \end{bmatrix}, \begin{bmatrix} \frac{\partial L}{\partial O_{11}} & \frac{\partial L}{\partial O_{12}} \\ \frac{\partial L}{\partial O_{21}} & \frac{\partial L}{\partial O_{22}} \end{bmatrix} \right)$$

$\frac{\partial L}{\partial X}$ — Filter F — Loss Gradient $\frac{\partial L}{\partial O}$

∂L/∂X can be represented as 'full' convolution between a 180-degree rotated Filter F and loss gradient ∂L/∂O

پاسخ به صورت زیر خواهد بود.

$$z = \frac{o_{11} + o_{12} + o_{21} + o_{22}}{4} \Rightarrow \frac{\partial z}{\partial o_{ij}} = \frac{1}{4} \Rightarrow \frac{\partial z}{\partial o} = \begin{bmatrix} \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} \end{bmatrix}$$

number of elements $\leftarrow 4$

$$\frac{\partial L}{\partial o} = \frac{\partial L}{\partial z} \times \frac{\partial z}{\partial o} = \begin{bmatrix} 1/4 & 1/4 \\ 1/4 & 1/4 \end{bmatrix}$$

$$\frac{\partial L}{\partial F_{11}} = \frac{1}{4}(3+4+2+1) = \frac{10}{4} = 2.5 \qquad \frac{\partial L}{\partial F_{12}} = \frac{1}{4}(4+10+1-3) = 1.75$$

$$\frac{\partial L}{\partial F_{21}} = \frac{1}{4}(2+1+4-2) = 1.25 \qquad \frac{\partial L}{\partial F_{22}} = \frac{1}{4}(1-3-2+10) = -1$$

$$\frac{\partial L}{\partial F} = \frac{\partial L}{\partial o} \times \frac{\partial o}{\partial F} \Rightarrow \frac{\partial L}{\partial F} = \begin{bmatrix} 2.5 & 1.75 \\ 1.25 & -1 \end{bmatrix}$$

$$\frac{\partial L}{\partial x_{11}} = \frac{1}{4} \times 2 = 0.5 \qquad \frac{\partial L}{\partial x_{12}} = \frac{1}{4}(F_{12}+F_{11}) = \frac{1}{4}(2+0) = 0.5$$

$$\frac{\partial L}{\partial x_{13}} = \frac{1}{4} \times F_{12} = \frac{1}{4} \times 0 = 0 \qquad \frac{\partial L}{\partial x_{21}} = \frac{1}{4} \times F_{21} + \frac{1}{4} \times F_{11} = \frac{1}{4}(2-3) = -0.25$$

$$\frac{\partial L}{\partial x_{22}} = \frac{1}{4}(F_{22}+F_{21}+F_{12}+F_{11}) = \frac{1}{4}(4) = 1 \qquad \frac{\partial L}{\partial x_{23}} = \frac{1}{4}(F_{22}+F_{12}) = \frac{1}{4}(1+0) = 0.25$$

$$\frac{\partial L}{\partial x_{31}} = \frac{1}{4} F_{21} = \frac{1}{4}(-3) = -0.75 \qquad \frac{\partial L}{\partial x_{32}} = \frac{1}{4}(F_{22}+F_{21}) = \frac{1}{4}(-3+1) = -0.5$$

$$\frac{\partial L}{\partial x_{33}} = \frac{1}{4} \times F_{22} = \frac{1}{4}(1) = 0.25$$

$$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial o} \times \frac{\partial o}{\partial x} = \begin{bmatrix} 0 & 0.5 & 0.5 \\ -0.25 & 0 & 0.25 \\ -0.75 & -0.5 & 0.25 \end{bmatrix}$$

پاسخ تمرین سوم **مهدیه نادری: ۹۸۵۲۲۰۷۶**

سوال سوم:

الف)

Parameters for Conv1D layer 1:

$$\text{Parameters} = (\text{kernel size} \times \text{input channels} + \text{bias}) \times \text{number of filters}$$

Parameters for Dense layer 1:

$$\text{Parameters} = (\text{input size} + \text{bias}) \times \text{units}$$

لایه	ابعاد خروجی	توضیحات و محاسبات	تعداد پارامترهای قابل آموزش
Input Layer	(500, 7)	-	0
Convolutional Layer 1	(498, 16)	تعداد فیلترها = ۱۶ سایز هر فیلتر = ۳ چون کانولوشن یک بعدی است عدد بعد دوم ورودی یعنی ۷ تعداد کانال های آن حساب میشود. کرنل ها هم مربعی نیستند و خطی هستند. هر فیلتر هم یک بایاس دارد. (3*7+1)*16 = 352 باید ابعاد خروجی هم حساب کنیم چون برای لایه های بعدی لازم است. چون padding نداریم و سایز کرنل ۳ است از هر سمت ورودی ۱ واحد کم میشود یعنی در مجموع ۲ واحد کم میشود: ۵۰۰ - ۲ = ۴۹۸ تعداد کرنل ها ۱۶ تا است پس بعد دوم خروجی یا تعداد کانال ها ۱۶ میشود.	352
Max Pooling Layer 1	(249, 16)	تعداد ابعاد را نصف میکند ولی تعداد کانال تغییر نمیکند.	0
Convolutional Layer 2	(245, 32)	تعداد فیلترها = ۳۲ سایز هر فیلتر = ۵ تعداد کانال های ورودی این لایه ۱۶ است. هر فیلتر هم یک بایاس دارد. (5*16+1)*32 = 2592	2592
Max Pooling Layer 2	(122, 32)	تعداد ابعاد را نصف میکند ولی تعداد کانال تغییر نمیکند.	0
Convolutional Layer 3	(118, 64)	تعداد فیلترها = ۶۴ سایز هر فیلتر = ۵ تعداد کانال های ورودی این لایه ۳۲ است. هر فیلتر هم یک بایاس دارد. (5*32+1)*64 = 10304	10304
Flatten Layer	3776	تعداد ابعاد را نصف میکند ولی تعداد کانال تغییر نمیکند.	0
Dense Layer 1	128	تعداد نورون ها = ۱۲۸ هر نورون یک بایاس دارد. (3776+1)*128 = 483456	483456
Dense Layer 2	5	تعداد نورون ها = ۵ و هر نورون یک بایاس دارد. (128+1)*5 = 645	645

ب)

تفاوت اصلی بین) Conv2Dپیچیدگی 2 بعدی) و) Conv3Dپیچیدگی 3 بعدی) در ابعاد داده هایی است که روی آنها کار می کنند:

پاسخ تمرین سوم **مهدیه نادری: ۹۸۵۲۲۰۷۶**

Conv2D (پیچیدگی دو بعدی):

اساساً با داده‌های دو بعدی مانند تصاویر کار می‌کند، جایی که داده‌ها در ابعاد عرض و ارتفاع سازمان‌دهی می‌شوند، که اغلب به صورت عرض x ارتفاع x کانال نشان داده می‌شوند (به عنوان مثال، x224x3224 برای یک تصویر RGB).

معمولاً در وظایف بینایی رایانه برای پردازش تصویر، تشخیص اشیا، تشخیص تصویر و غیره استفاده می شود.

یک پنجره کشویی را روی داده‌های ورودی دو بعدی اعمال می‌کند و برای استخراج ویژگی‌ها، ضرب عنصر را با یک هسته (همچنین به عنوان فیلتر) انجام می‌دهد.

Conv3D (پیچیدگی سه بعدی):

روی داده‌های سه بعدی کار می‌کند و بعد دیگری فراتر از عرض و ارتفاع اضافه می‌کند، که معمولاً برای داده‌های ویدیویی یا داده‌های حجمی که عمق نیز در نظر گرفته می‌شود استفاده می‌شود (به عنوان مثال، فریم‌های ویدیویی که به صورت فریم x ارتفاع x عرض x کانال نشان داده می‌شوند).

کاربردها شامل تجزیه و تحلیل ویدئویی، تصویربرداری پزشکی (مانند سی تی اسکن، ام آر آی)، تجزیه و تحلیل داده های مکانی و زمانی و غیره است.

شبیه Conv2D است، اما در سه بعد فضایی (به عنوان مثال، x، y، (z، با در نظر گرفتن عمق علاوه بر عرض و ارتفاع، پیچش را انجام می دهد.

کاربردهای Conv3D :

تجزیه و تحلیل ویدیو Conv3D :به تجزیه و تحلیل ویژگی‌های مکانی-زمانی در ویدیوها برای کارهایی مانند تشخیص عملکرد، طبقه‌بندی ویدیو، و استخراج ویژگی‌های زمانی کمک می‌کند.

تصویربرداری پزشکی: به پردازش داده های پزشکی حجمی مانند سی تی اسکن یا ام آر آی کمک می کند و امکان استخراج ویژگی های سه بعدی را برای اهداف تشخیصی، تشخیص تومور، تقسیم بندی اندام و غیره فراهم می کند.

تجزیه و تحلیل داده‌های مکانی-زمانی: مدل‌های Conv3D را می‌توان در حوزه‌های مختلفی که با داده‌های مکانی-زمانی مانند پیش‌بینی آب‌وهوا، شبیه‌سازی دینامیک سیالات، و مطالعات زمین‌شناسی که درک الگوهای سه‌بعدی در طول زمان بسیار مهم است، استفاده کرد.

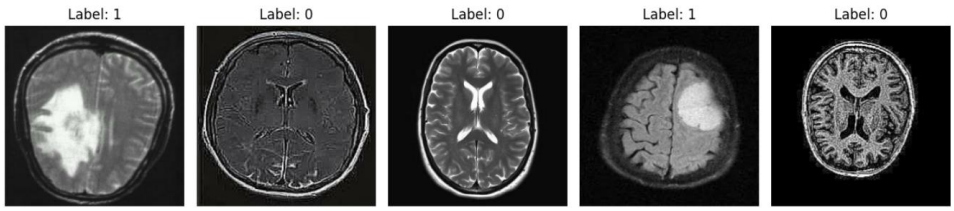
تشخیص ژست: تشخیص ژست ها یا حرکات در فضای سه بعدی، مانند تفسیر زبان اشاره یا ضبط حرکت، از معماری Conv3D سود می برد.

در اصل، در حالی که Conv2D بر روی داده‌های دو بعدی مانند تصاویر کار می‌کند، Conv3D این مفهوم را برای پردازش داده‌های حجمی یا مکانی-زمانی در سه بعد گسترش می‌دهد و کاربردهایی را در حوزه‌های مختلف پیدا می‌کند که با چنین ساختارهای داده‌ای پیچیده سروکار دارند.

سوال چهارم:

دیتاست را دانلود کردیم و در یک فولدر ذخیره و سئس از حالت فشرده خارج کردیم. و به دو بخش ولیدیشن و ترین تقسیم کردیم و یک پک کامل آن هم آماده کردیم که اگر لازم شد استفاده کنیم.

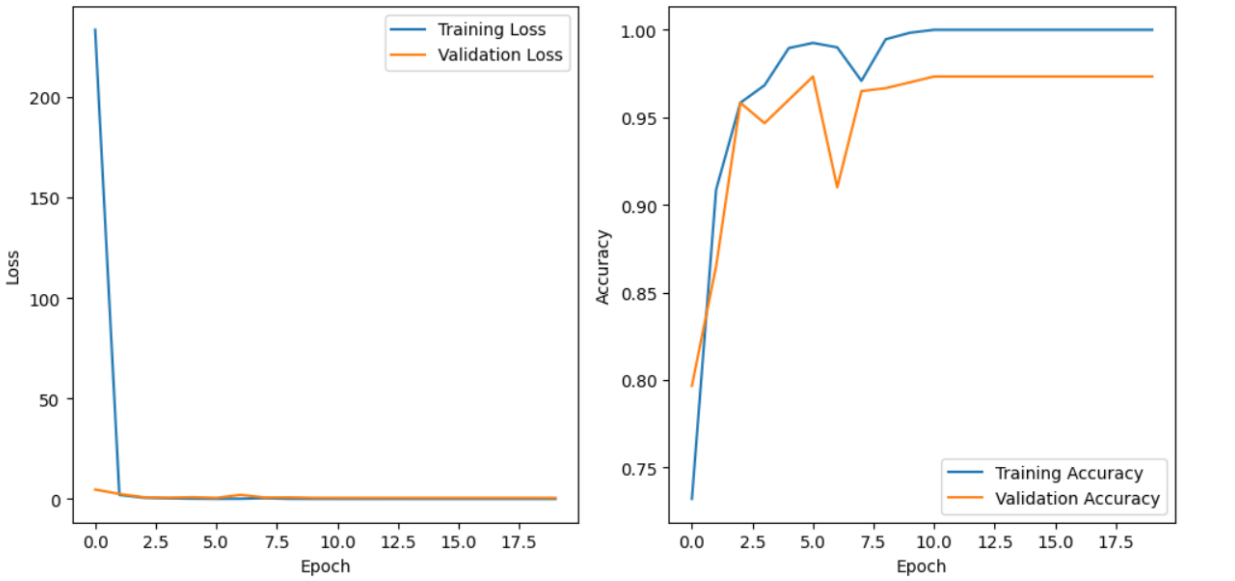
5 تا از تصاویر آن را به شکل زیر نمایش دادیم.



سپس مدل های خواسته شده را تعریف کرده و آموزش و تست کردیم و نمودارهای هرکدام را رسم کردیم.

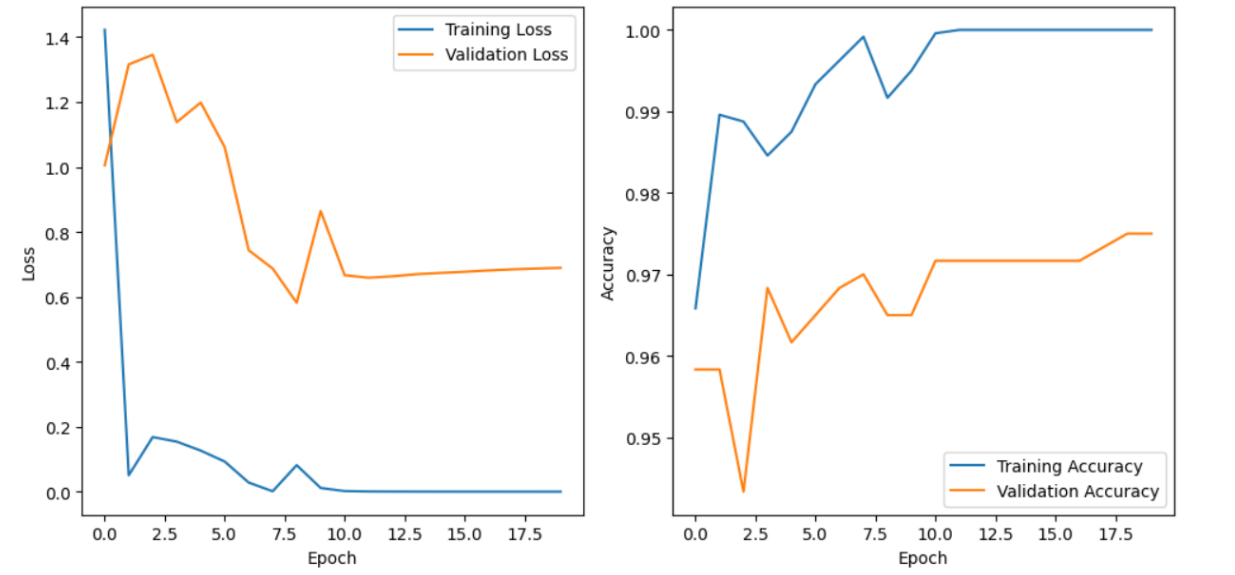
نمودار مدل سکوئنشیال:

```
Epoch 20/20
75/75 [==============================] - 4s 53ms/step - loss: 1.5093e-05 - accuracy: 1.0000 - val_loss: 0.4964 - val_accuracy: 0.9733
```



نمودار مدل فانکشنال:

```
Epoch 20/20
75/75 [==============================] - 4s 55ms/step - loss: 4.5532e-05 - accuracy: 1.0000 - val_loss: 0.6894 - val_accuracy: 0.9750
```



تعداد لایه ها و نوعشان در هر دو لایه یکسان بود. نتایج خیلی بهم نزیک هستند.

سوال پنجم:

لایه‌های هم‌گشتی در شبکه‌های عصبی کانولوشنال (Convolutional Neural Networks یا CNNs) ، برای پردازش تصاویر به‌خصوص در دسته‌بندی آن‌ها کاربرد فراوانی دارند.

پاسخ تمرین سوم **مهدیه نادری: ۹۸۵۲۲۰۷۶**

مثال عملی از ویژگی‌های منحصر به فرد لایه‌های هم‌گشتی:

یک مثال عملی از استفاده از لایه‌های هم‌گشتی در دسته‌بندی تصاویر می‌تواند شناسایی شیء یا ویژگی خاص در تصویر باشد. برای مثال، یک لایه هم‌گشتی می‌تواند ویژگی‌هایی مانند لبه‌ها، گوشه‌ها، الگوها و اشکال مختلف را در تصاویر شناسایی کند. این ویژگی‌ها سپس به لایه‌های بعدی منتقل می‌شوند تا بر اساس آن‌ها تصویر دسته‌بندی شود. این ویژگی‌ها باعث بهبود دقت و عملکرد دسته‌بندی می‌شوند زیرا اطلاعات مهم و با ارزش از تصویر استخراج می‌شوند.

چالش‌های لایه‌های هم‌گشتی:

با این حال، لایه‌های هم‌گشتی نیز می‌توانند چالش‌هایی را به وجود آورند. برخی از معایب و چالش‌های مرتبط با لایه‌های هم‌گشتی عبارتند از:

1. **اشتباه در استخراج ویژگی :**لایه‌های هم‌گشتی ممکن است ویژگی‌های غیرمفهومی را استخراج کنند یا ویژگی‌های مهم را از دست بدهند که باعث کاهش دقت در دسته‌بندی تصاویر می‌شود.
2. **پیچیدگی محاسباتی :**استفاده از لایه‌های هم‌گشتی ممکن است نیازمند منابع محاسباتی زیادی باشد که می‌تواند زمان‌بر و هزینه‌بر باشد.
3. **تراکم ویژگی‌ها :**در برخی موارد، لایه‌های هم‌گشتی ممکن است اطلاعات تکراری یا تراکم شده را ارائه دهند که موجب افزایش حجم داده و کاهش کارایی مدل می‌شود.
4. **دگرگونی فضای ویژگی :**در برخی حالات، فضای ویژگی ممکن است به‌طور ناخواسته دگرگون شود که می‌تواند منجر به از دست رفتن اطلاعات مفید و کاهش دقت دسته‌بندی شود.

این چالش‌ها می‌توانند تأثیر مستقیمی بر عملکرد و کارایی مدل‌های دسته‌بندی تصاویر داشته باشند. بهینه‌سازی و تطبیق مناسب با ویژگی‌های داده و شناسایی مسائل مرتبط می‌تواند به حل این مشکلات کمک کند و عملکرد مدل‌های دسته‌بندی را بهبود بخشد.

بله، البته که ویژگی‌های لایه‌های هم‌گشتی همراه با مزایا، معایب خاصی نیز دارند. یکی از معایب یا چالش‌هایی که ممکن است در لایه‌های هم‌گشتی ایجاد شود، پدیده‌ی تعمیم ناپذیری یا عدم قابلیت تعمیم مطلوب مدل در مواجهه با داده‌های جدید است.

مثال عملی) Overfitting :برازش بیش از حد(

یکی از چالش‌های اصلی لایه‌های هم‌گشتی می‌تواند مسأله‌ی برازش بیش از حد به داده‌ها باشد. این موضوع زمانی رخ می‌دهد که مدل به‌طور زیادی به داده‌های آموزشی وابسته می‌شود و ویژگی‌های خاص و تفاوت‌های جزئی داده‌های آموزشی را یاد می‌گیرد به گونه‌ای که نتواند به خوبی با داده‌های جدید یا داده‌هایی که قبلا مشاهده نکرده، کار کند.

برای مثال، فرض کنید یک مدل با لایه‌های هم‌گشتی برای تشخیص خودروهای موجود در تصاویر آموزش داده شود. اگر این مدل به گونه‌ای آموزش داده شود که وابستگی‌های بسیار دقیق به ویژگی‌های خاص مانند جزئیات کوچک، نوع خاصی از شیء مثل مدل، سال ساخت و ... را یاد بگیرد، این مدل ممکن است در برخورد با تصاویری که مشخصات دقیقی که در داده‌های آموزشی دیده نشده باشند، عملکرد ضعیفی داشته باشد. به عبارت دیگر، مدل به داده‌های آموزشی بسیار برازش شده است و از قابلیت تعمیم به داده‌های جدید برخوردار نمی‌باشد.

این مسأله اغلب با استفاده از روش‌هایی مانند استفاده از داده‌های ورودی بیشتر، استفاده از تکنیک‌های کاهش برازش بیش از حد (مثل Dropout) یا استفاده از تنظیمات متفاوت شبکه عصبی (مانند اندازه‌ی لایه‌ها، تعداد لایه‌ها و ...) قابل حل است.

پاسخ تمرین سوم **مهدیه نادری: ۹۸۵۲۲۰۷۶**

سوال ششم:

الف)

هدف از استفاده از فیلترهای ۱X۱ در شبکه‌های عصبی هم‌گشتی به کاهش تعداد نقشه‌های ویژگی و حفظ ویژگی‌های مهم می‌پردازد. این فیلترها عمق نقشه‌های ویژگی را کاهش می‌دهند اما به عنوان یک لایه پردازشی، اطلاعات معناداری را از ویژگی‌های قبلی استخراج می‌کنند. با اعمال یک فیلتر ۱X۱، تعداد کانال‌های ورودی کاهش می‌یابد (اگر فیلترهای اعمال شده قبلی بیشتر از یک کانال بوده باشند) و این کاهش تعداد کانال‌ها می‌تواند به کاهش محاسبات مورد نیاز در شبکه و افزایش سرعت آموزش کمک کند.

ب)

پس از اعمال یک فیلتر ۱X۱، نقشه ویژگی نهایی اطلاعاتی را ارائه می‌دهد که از ترکیب خطی ویژگی‌های موجود در نقشه ورودی به‌دست آمده است. این نقشه ویژگی جدید حاوی ترکیباتی از ویژگی‌های مختلف با وزن‌های متفاوت است.

پ)

نقشه ویژگی که از اعمال یک فیلتر ۱X۱ به‌دست می‌آید، با تصویر اصلی یا فیلترهای دیگر با اندازه‌ها و ویژگی‌های متفاوت می‌باشد. این نقشه ویژگی حاوی اطلاعات فشرده‌تری از نقشه‌های ویژگی قبلی است و بستگی به تعداد و نوع فیلترهای قبلی و وزن‌های آن‌ها دارد.

ت)

فیلترهای ۱X۱ در شبکه‌های عصبی عموماً در معماری‌های شبکه عصبی عمیق استفاده می‌شوند. مثلاً در شبکه‌های Inception که برای پردازش تصویر استفاده می‌شوند، از فیلترهای ۱X۱ به‌طور گسترده استفاده می‌شود.

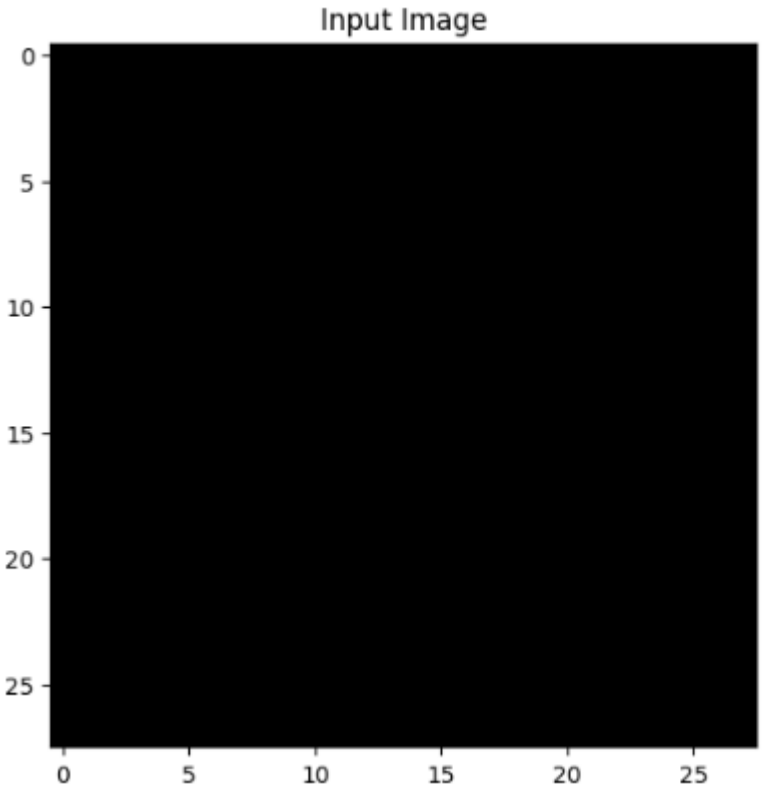
ث)

در برخی موارد، استفاده از فیلترهای ۱X۱ ممکن است مفید نباشد. اگر تعداد کانال‌های ورودی بسیار کم باشد و یا اگر به دلیل کاربرد خاصی نیاز به کاهش ابعاد نباشد، استفاده از این فیلترها ممکن است نتایج بهتری نداشته باشد. همچنین، اگر به‌دست آوردن ویژگی‌های جدید از ترکیبات مختلف از نظر محتوا یا تصاویری که دارای ابعاد مختلف هستند، مورد نظر نباشد، استفاده از این فیلترها ممکن است کمکی نکند.

ج)

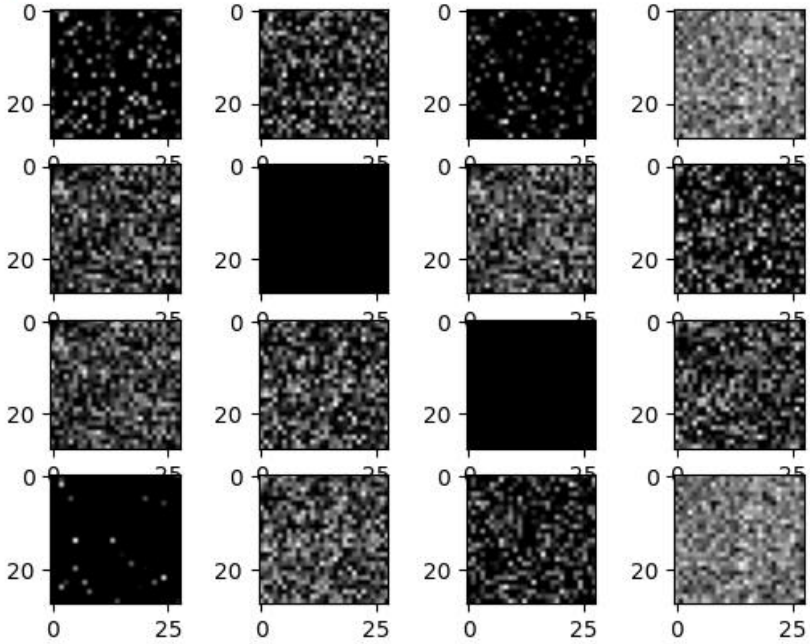
```python
import tensorflow as tf
import numpy as np
import matplotlib.pyplot as plt

# مدل را ایجاد کنید (با فرض کد مدل قبلی)
model = tf.keras.Sequential([
    tf.keras.layers.Input(shape=(28, 28, 3)),
    tf.keras.layers.Conv2D(filters=16, kernel_size=(1, 1), activation='relu'),
])

# نمونه‌ای از تصویر ورودی
input_image = np.random.rand(1, 28, 28, 3)

# محاسبه خروجی لایه
layer_output = model.predict(input_image)

# نمایش تصویر ورودی
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
plt.title('Input Image')
plt.imshow(input_image[0].astype('uint8'))  # نمایش تصویر ورودی


```

```
1/1 [==============================] - 0s 214ms/step
<matplotlib.image.AxesImage at 0x7be855198ca0>
```



```
# نمایش خروجی لایه
plt.subplot(1, 2, 2)
plt.title('Layer Output')

for i in range(layer_output.shape[-1]):
    plt.subplot(4, 4, i+1)
    plt.imshow(layer_output[0, :, :, i], cmap='gray')  # نمایش هر کانال به صورت تصویر خاکستری
plt.show()
```

```
<ipython-input-7-755d484b8810>:6: MatplotlibDeprecationWarning: Auto-removal of overlapping axes is d
  plt.subplot(4, 4, i+1)
```



سوال هفتم:

شبکه‌ی عصبیInception ، یک معماری عمیق برای شبکه‌های عصبی کانولوشنی است که از یک ترکیب از فیلترهای اندازه‌های مختلف و نوع مختلف عملیات کانولوشنی استفاده می‌کند. هدف این ماژول Inception ایجاد یک شبکه عصبی با ساختاری که بتواند به‌طور همزمان از ویژگی‌های مختلف تصویر استفاده کند و اطلاعات را به صورت موازی استخراج کند.

چگونه اندازه پارامتر گام در لایه های هم گشتی بر ابعاد فضای نگاشت ویژگی ها تاثیر می گذارد؟

ساختار Inception شامل استفاده از عملیات کانولوشن ۱x۱، ۳x۳، و ۵x۵ به صورت موازی است. این ماژول از یک لایه کانولوشن ۱x۱ برای کاهش عمق فضای ویژگی استفاده می‌کند. بنابراین، اندازه پارامتر گام نیز در اینجا نقش مهمی ایفا می‌کند که با استفاده از این فیلترهای مختلف، می‌تواند اطلاعات را به‌صورت همزمان از ویژگی‌های مختلف تصویر استخراج کرده و با کمک یکدیگر، ویژگی‌های کلی تصویر را شناسایی کند.

ویژگی های کلیدی و عملیات لایه های هم گشتی مورد استفاده در شبکه خود و اهمیت آنها در استخراج ویژگی را شرح دهید.

ویژگی‌های کلیدی لایه‌های هم‌گشتی مورد استفاده در این شبکه‌ها عبارتند از:

پاسخ تمرین سوم
مهدیه نادری: ۹۸۵۲۲۰۷۶

1. **Convolution Layers** (**لایه‌های کانولوشنی**) : این لایه‌ها برای استخراج ویژگی‌های تصویری از ورودی استفاده می‌شوند. با استفاده از فیلترهای کانولوشنی، ویژگی‌های مختلف تصویر استخراج می‌شوند.
2. **Pooling Layers** (**لایه‌های پولینگ**) : این لایه‌ها برای کاهش ابعاد ویژگی‌ها و حفظ اطلاعات مهم استفاده می‌شوند. معمولاً از عملیات Max Pooling یا Average Pooling برای این منظور استفاده می‌شود.
3. **Batch Normalization** (**نرمال‌سازی دسته‌ای**) : این عملیات برای استانداردسازی ورودی‌های لایه‌های مختلف و جلوگیری از مشکل مربوط به مشکل تفاوت مقیاس و تغییرات وزن‌ها استفاده می‌شود.
4. **Activation Functions** (**توابع فعال‌سازی**) : این توابع برای افزودن عدم خطیت به شبکه استفاده می‌شوند. توابعی مانند ReLU، Sigmoid و Tanh می‌توانند به عنوان توابع فعال‌سازی استفاده شوند.

برای پیاده‌سازی شبکه عصبی Inception برای دسته‌بندی تصویر با استفاده از مجموعه داده CIFAR-10 از کتابخانه TensorFlow یا Keras و از لایه‌ها و ماژول‌های Inception گرهای این کتابخانه‌ها استفاده کردیم. با استفاده از لایه‌هایConv2D ، MaxPooling2D و Activation Functionsاین شبکه را پیاده سازی کردیم.

```
# پیش‌پردازش داده‌ها
train_images, test_images = train_images / 255.0, test_images / 255.0

# ایجاد مدل Inception
input_shape = (32, 32, 3)  # ابعاد تصویر
num_classes = 10  # تعداد کلاس‌ها در مجموعه داده CIFAR-10

model = models.Sequential()

# لایه‌های اولیه شبکه Inception
model.add(layers.Conv2D(64, (1, 1), activation='relu', input_shape=input_shape))

# لایه‌های Inception Module
model.add(layers.Conv2D(64, (1, 1), activation='relu'))
model.add(layers.Conv2D(128, (3, 3), activation='relu'))
model.add(layers.Conv2D(256, (5, 5), activation='relu'))
model.add(layers.MaxPooling2D((3, 3), strides=(1, 1), padding='same'))

# لایه‌های آخر شبکه
model.add(layers.Flatten())
model.add(layers.Dense(512, activation='relu'))
model.add(layers.Dense(num_classes, activation='softmax'))

# کامپایل مدل
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

# آموزش مدل
history = model.fit(train_images, train_labels, epochs=3, validation_data=(test_images, test_labels))

# ارزیابی مدل بر روی داده‌های تست
test_loss, test_accuracy = model.evaluate(test_images, test_labels)

print(f"دقت مدل بر روی داده‌های تست: {test_accuracy * 100:.2f}%")
```

```
Downloading data from https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
170498071/170498071 [==============================] - 3s 0us/step
Epoch 1/3
 952/1563 [=================>............] - ETA: 38:34 - loss: 1.5777 - accuracy: 0.4343
```

مهدیه نادری: ۹۸۵۲۲۰۷۶