



دانشگاه صنعتی اصفهان

دانشکده برق و کامپیوتر

گزارش پژوهه کارشناسی

موضوع

طراحی و بررسی و پیاده‌سازی سیستم پرو کردن لباس به صورت مجازی با بهره‌گیری از شبکه‌های عصبی و مدل‌های تولید کننده

استاد:

دکتر محمدرضا حیدرپور

دانشجو:

زینب صادقیان ۹۵۲۹۳۶۳

ترم تابستان تحصیلی ۱۳۹۸-۹۹

فهرست مطالب

۶	مقدمه
۷	فصل اول
۷	چیست؟ Virtual Try-On
۷	چرا Virtual try-on از طریق AR مهم و محبوب است؟
۹	TriMirror
۱۱	فصل دوم- پیش نیازها
۱۱	اسپلاین صفحه نازک
۱۴	تغییر فرم
۱۶	نقاط هماهنگ
۱۸	تطابق دادن هندسی
۱۹	معماری تطابق هندسی
۲۰	استخراج ویژگی
۲۰	شبکه تطابق
۲۱	تطابق تجربی در تخمین هندسی کلاسیک
۲۱	لایه‌ی تطابق
۲۲	شبکه رگرسیون
۲۳	شبکه‌های عصبی غیر محلی
۲۳	فرمول‌سازی
۲۴	بلاک غیر محلی
۲۵	پیاده‌سازی بلاک غیر محلی
۲۵	پارس کردن و تخمین ژست بدن انسان
۲۶	openPose

۲۷	به شخص نگاه کنید
۲۹	بلاک های اکسپشن و آغاز (Inception)
۲۹	بلاک اینسپشن
۳۱	فاکتور گیری توسط کانولوشن های غیر متقاضان
۳۲	مرتب ساز کمکی
۳۳	کاهش سایز شبکه بندی به صورت بهینه
۳۴	معماری inception-v3
۳۴	بلاک اکسپشن
۳۴	کانولوشن جدایزیر به تفکیک عمق اوریجینال
۳۷	معماری کلی
۳۸	مدل های تولید کننده
۳۹	آموزش مدل های تولید کننده
۴۰	شبکه های مولد خصمانه
۴۲	تقریب: شبکه های عصبی خصمانه
۴۵	هزینه کانونی
۴۸	تفکیک کننده (PatchGAN) Markovian
۵۰	Unet
۵۱	شبکه های VGG
۵۲	هزینه های VGG
۵۳	فصل سوم - روش های موجود
۵۳	داده ها
۵۵	VTON
۵۵	نمایش فرد در VTON

۵۶CP-VTON
۵۷نمايش فرد در CP-VTON
۵۷ماژول هماهنگی هندسی
۵۷ماژول پوشیدن لباس
۵۹فصل چهارم - روش استفاده شده
۵۹داده های مسئله
۵۹روش به کار رفته
۶۰نمايش فرد در روش پياده سازی شده
۶۲ماژول تغيير حالت لباس
۶۳ماژول توليد نقشه هاي قطعه بندی M2
۶۵ماژول سنتز پوشیدن
۶۷فصل پنجم - نتایج و پياده سازی
۶۷جزئيات پياده سازی
۶۷ماژول تغيير فرم لباس
۶۷ماژول توليد نقشه هاي بخش بندی
۶۸نتایج کمی
۶۸نتایج کيفی
۷۲فصل ششم - جمع بندی و کارهای آينده
۷۳مراجع

مقدمه

sistem‌های امتحان کردن وسایل پوششی به صورت مجازی با هدف انتقال یک آیتم پوششی روی قسمت مشخصی از بدن پیشرفت‌های زیادی کرده است اما چالش‌ها همچنان برای ایجاد یک ظاهر درست و واقعی که ویژگی‌های بدن و فرد و لباس به خوبی حفظ شود، باقی مانده است. در این پروژه روش‌های موجود در این زمینه بررسی می‌شود و سپس روشی که بهترین نتیجه را داشته معرفی و تحلیل می‌شود. بخشی از این روش در این پروژه پیاده‌سازی شده است که نتایج آن را در انتهای مشاهده خواهیم کرد. لازم به ذکر است که این تنها قسمتی از یک راه طولانی برای ایجاد یک سیستم در لحظه‌ی پوشیدن لباس با استفاده از تکنولوژی واقعیت افزوده است. لذا این پروژه به عنوان نقطه آغازی برای پیاده‌سازی نهایی چنین سیستمی خواهد بود. پس از اعمال و تکمیل پیاده‌سازی این پروژه نیاز به همراه کردن و تنظیم کردن سیستم این پروژه با یک سیستم AR است. در این پروژه از انواع مختلف شبکه‌های عصبی و روش‌های آماری از جمله مدل‌های مولد خصمانه استفاده شده است.

۱ فصل اول

به منظور توضیح دقیق‌تر موضوع، در این بخش توضیح داده می‌شود مفهوم امتحان کردن اشیا به صورت مجازی^۱ چیست، چگونه استفاده می‌شود و پرداخت به آن چه لزومی دارد. همچنین مختصررا در باب اینکه شبکه‌های عصبی^۲ و یادگیری عمیق^۳ چگونه در این موارد استفاده می‌شود توضیحاتی داده می‌شود.

۱.۱ Virtual Try-On چیست؟

به صورت مختصر، virtual try-on روشی است که در آن مشتری یک محصول می‌تواند محصول مدنظر خود را از طریق تلفن همراه یا هر دستگاه مجهز به دوربین دیگر امتحان کند. تکنولوژی واقعیت افزوده^۴ یا AR این امکان را به وجود آورده است که افراد می‌توانند خود را در محصولی که می‌خواهند خریداری کنند از طریق گوشی هوشمند خود مشاهده کنند. این تکنولوژی مدتی است که در مواردی بین مردم محبوب است و در اپلیکیشن‌های مختلف (مانند Instagram، Snappchat) استفاده شده است و افراد می‌توانند عینک و زیورالات کوچک دیگر را امتحان کنند. این امر باعث شده است که تکنیک‌های تشخیص چهره^۵، مدل‌سازی سه بعدی^۶ و یادگیری ماشین روی عکس و ویدئو بهبود پیدا کند.

۲.۱ چرا Virtual try-on از طریق AR مهم و محبوب است؟

از دید مشتریان یک کسب و کار^۷ استفاده از AR راحت و سرگرم کننده است. همچنین از دید خود کسب و کار این امکان که مشتریان محصولات را قبل از خرید امتحان کنند می‌تواند ایجاد سودآوری کند. در کل مهم‌ترین ویژگی‌های AR که باعث می‌شود میزان خرید محصولات زیاد گردد را می‌تواند ۵ مورد زیر در نظر گرفت:

۱. تست محصول قبل از خرید: با وجود چنین سیستمی مشتری می‌تواند بررسی کند که مثلاً کفشی که قصد خرید آن را دارد متناسب با موارد دیگری ظاهری هست یا خیر.
۲. در هر مکانی می‌شود محصول را امتحان کرد: افراد می‌توانند بدون حضور در فروشگاه مواردی که می‌خواهند را امتحان کنند.
۳. صرفه جویی در زمان برای خرید
۴. مشورت کردن با افراد مختلف قبل از خرید محصول

¹ Virtual try-on

² Neural networks

³ Deep learning

⁴ Augmented reality

⁵ Face recognition

⁶ 3d-modeling

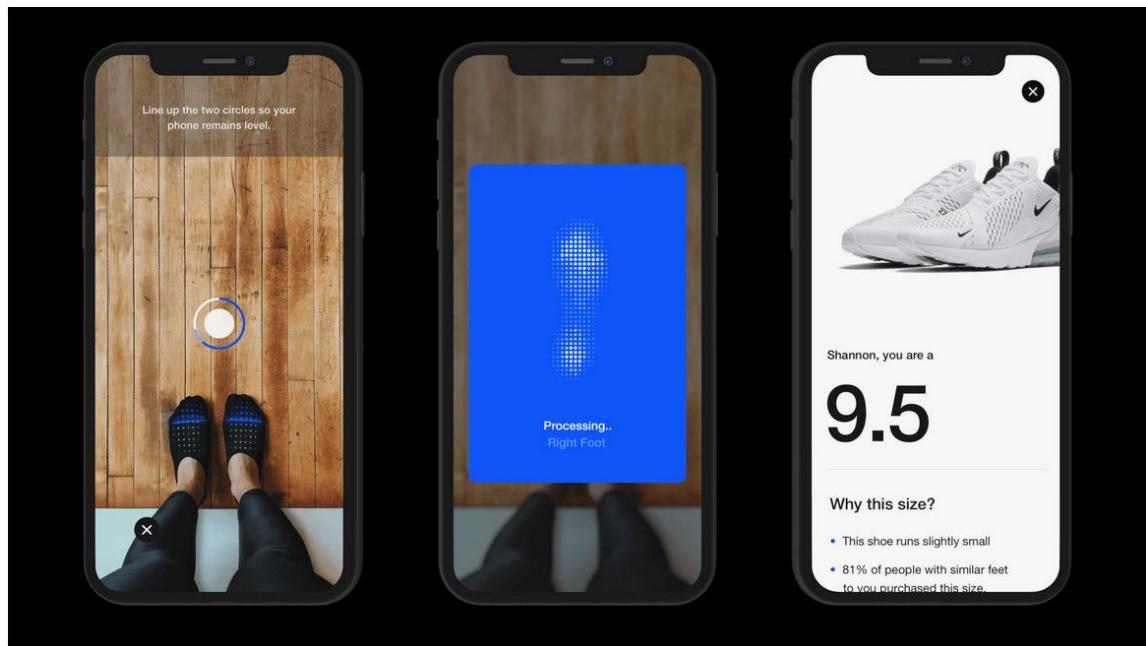
⁷ business

۵. افراد با هماهنگ کردن رنگ و مدل محصولی که قصد خرید آن را دارند، با محصولاتی که دارند خرید امن اری را تجربه خواهند کرد.

این سیستم برای خرید و امتحان کردن محصولات مختلفی مانند عینک و کفش و زیورآلات و برخی مدل‌های آرایشی به وجود آمده است که در زیر مثال‌هایی ذکر خواهد شد.

۱. اپلیکیشن شرکت نایک^۱ برای امتحان کردن کفش: Nike Fit

این اپلیکیشن به مشتریان این اجازه را می‌دهد که سایز مورد نظر برای کفشی که می‌خواهند خریداری کنند را یافته کنند و مشاهده کنند کفش در پای آن‌ها به چه صورت خواهد بود. این اپلیکیشن از تکنولوژی AR استفاده می‌کند تا پای فرد را اسکن کند.



شکل ۱-۱- تصاویری از عملکرد اپلیکیشن Nike Fit را مشاهده می‌کنیم.

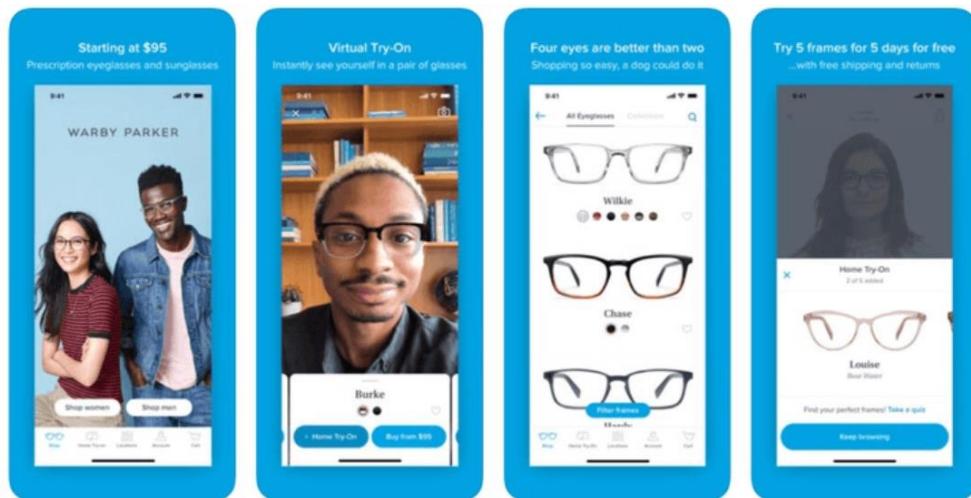
۲. برنده آرایشی مک^۲ امکان آرایش مجازی از طریق ویدئوهای یوتیوب^۳ را فراهم کرده است. افرادی که کanal یوتیوب این برنده آرایشی را دنبال می‌کردند این شانس را پیدا کردند تا آخرین محصولات آرایشی این برنده را به صورت مجازی و مستقیم از طریق ویدئو امتحان کنند. این برنده اولین برنده بود که از تکنولوژی جدید یوتیوب AR بهره برداشت.

¹ Nike

² M.A.C

³ Youtube

۳. برندهای مایکل کورس روی پلتفرم فیسبوک و از طریق تبلیغات روی این پلتفرم امکان مجازی تست کردن عینک‌های این برنده را برای افراد مختلف فراهم ساخت. کاربران فیسبوک با کلیک بر روی تبلیغات این برنده به صفحه‌ای می‌رفتند که می‌توانستند عینک‌ها را امتحان کنند.



شکل ۲-۱

با وجود این موارد هنوز پلتفرم مناسبی برای امتحان کردن لباس به صورت مجازی به وجود نیامده است. سایتها بی وجوه دارد که از طریق مدل‌سازی سه بعدی سعی بر اجرای اینکار دارند. از جمله این موارد می‌توان به sensemi و triMirror اشاره کرد.

۱.۲.۱ TriMirror

TriMirror یک اپلیکیشن در لحظه است که خود را به عنوان تکنولوژی اتاق پرو مجازی معرفی می‌کند. کاری که این برنامه انجم می‌دهد این است که افراد با وارد کردن سایز خود می‌توانند یک آواتار سه بعدی از خود ایجاد کنند، سپس لباس‌های مختلف که آن‌ها نیز به صورت سه بعدی مدل‌سازی شده‌اند را روی این آواتار ایمیشنی امتحان می‌کنند. این عملیات می‌تواند به صورت کامل نشان دهد لباس روی بدن فرد چگونه قرار خواهد گرفت و سایزبندی را کامل مورد بررسی قرار می‌دهد. لازم به ذکر است که این برنامه از الگوریتم‌های مدل‌سازی سه بعدی برای شیوه‌سازی و ساخت آواتار و لباس مورد نظر و روش‌های بهینه‌سازی GPU و CPU استفاده می‌کند.

^۱ Real-time



شکل ۱-۳

در نهایت چنین مدل سازی هایی هزینه های زیادی را در پی خواهد داشت لذا در این پروژه به دنبال ساخت یک سیستم دو بعدی هستیم که با تکنولوژی AR همراه باشد. در فصل سوم به تفضیل مقاله هایی که برای این سیستم های دو بعدی است را بررسی خواهیم کرد و در نهایت در فصل چهارم روشی که به کار گرفته شده و پیاده سازی شده است را بیان خواهیم کرد و در دو فصل آخر نتایج پیاده سازی و کارهایی که در آینده انجام خواهد شد را ذکر خواهیم کرد. در ادامه به بررسی تمامی مازول ها و پیش نیاز هایی که برای فهم مقاله ها و پیاده سازی روش انجام شده است می پردازیم.

۲ فصل دوم- پیش‌نیازها

در این فصل تمامی پیش‌نیازهایی که برای ایجاد سیستم نهایی مدنظر است را بررسی می‌کنیم و تا حد امکان توضیحات کافی در باب آن‌های مطرح می‌کنیم تا در کسیستم ساده‌تر شود.

۱.۲ اسپلاین صفحه نازک

فرض کنیم دو عکس داریم که می‌خواهیم یکی را طوری تغییر فرم^۱ دهیم که به دومی تبدیل شود. در این قسمت توضیحاتی در باب صفحات نازک^۲ داده می‌شود و روش اسپلاین صفحه نازک^۳ یا به اختصار TPS را بیان خواهیم کرد [1].

اسپلاین در ریاضیات یکتابع هموار چند جمله‌ای- چندضابطه‌ای است. در مسائل درون‌یابی معمولاً منظور از درون‌یابی اسپلاین^۴، پیدا کردن چندجمله‌ای درون‌یابی است. روش‌های مختلفی به منظور درون‌یابی هموار بین تعدادی نقاط کنترلی وجود دارد که اسپلاین صفحه نازک از این دست است. این روش، سطحی که از هر نقطه‌ی کنترلی عبور می‌کند را درون‌یابی می‌کند. می‌دانیم که یک مجموعه‌ای سه نقطه‌ای در فضای یک سطح صاف^۵ ایجاد می‌کند. در نتیجه می‌توان گفت نقاط کنترلی در حقیقت محدودیت‌های مکانی هستند که یک سطح خمیده ایجاد می‌کنند. سطح خمیده‌ای ایده‌آلی که می‌توان ایجاد کرد، سطح خمیده‌ای است که به کمترین میزان خم شده باشد. در شکل ۱-۲ چنین سطح خمیده‌ای را با ۷ نقطه‌ی کنترلی مشاهده می‌کنیم. این سطح باید از هر ۷ نقطه حتماً عبور کند.

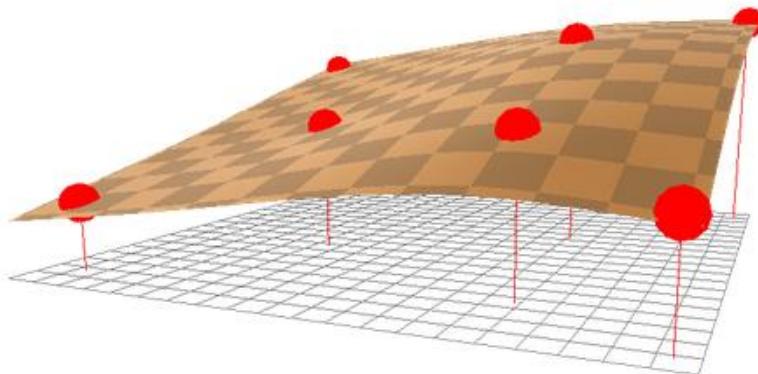
¹ Deform

² Thine plates

³ Thin-plate spline

⁴ Spline interpolation

⁵ flat



شکل ۱-۲- یک اسپلاین سطح نازک را نشان می دهد که از چندین نقاط کنترلی عبور می کند.

این حداقل سطح خم شده از طریق رابطه زیر به دست می آید:

$$f(x,y) = a_1 + a_2x + a_3y + \sum_{i=1}^n w_i U(|P_i - (x,y)|)$$

رابطه ۱-۲

سه عبارت اول رابطه ۲-۱، مربوط به قسمت خطی می شود که مربوط به صفحه‌ی صافی می شود که تمامی نقاط کنترلی را در بر می گیرد. این بخش می تواند با محاسبه کمترین مربعات، محاسبه گردد. عبارت آخر مربوط به نیروهای خم کننده است که توسط نقطه‌های کنترل کننده به سطح تحمیل می شود. برای هر نقطه کنترلی یک وزن W_i وجود دارد. عبارت $|P_i - (x,y)|$ فاصله بین نقطه کنترلی P_i و مکان (x,y) را نشان می دهد. این مقدار در تابع U مورد استفاده قرار می گیرد. تابع U به صورت زیر تعریف می شود:

$$U = r^2 \log r$$

رابطه ۲-۲

تا اینجا مقادیری که مشخص نیست و باید به دست آید وزن‌های W_i ضرائب a_1, a_2, a_3 است. تمامی وزن‌های W_i و کتور W را تشکیل می دهند و این مقادیر نا مشخص را می توان در رابطه ۳-۲ تعریف کرد:

$$L^{-1}V = (W|a_1 a_2 a_3)^T$$

رابطه ۳-۲

مقادیری که در دست داریم و مشخص هستند مکان نقاط کنترل (x_i, y_i) , و ارتفاع این نقاط V_i , است. مکان نقاط کنترلی را با ماتریس P و ارتفاع آنها را با وکتور Y که توسط صفر گسترش یافته است، تعریف می‌کنیم.

$$P = \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ \dots & \dots & \dots \\ 1 & x_n & y_n \end{bmatrix}$$

رابطه ۲-۴- ماتریس مکان نقاط کنترلی

$$Y = \begin{bmatrix} v_1 \\ v_2 \\ \dots \\ v_n \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

رابطه ۲-۵- ارتفاع نقاط کنترلی

اکنون ماتریس K را تعریف می‌کنیم. این ماتریستابع U را برای r_{ij} محاسبه می‌کند. $|Pi - Pj| = r_{ij}$ ، که فاصله‌ی بین دو نقطه‌ی کنترلی می‌باشد.

$$K = \begin{bmatrix} U(r_{11}) & U(r_{12}) & \dots \\ U(r_{21}) & U(r_{22}) & \dots \\ \dots & \dots & U(r_{nn}) \end{bmatrix}$$

رابطه ۲-۶- ماتریس

اکنون ماتریس L که در رابطه‌ی ۲-۳ استفاده شد، با ترکیبی از ماتریس K و P ساخته می‌شود.

$$L = \begin{bmatrix} K & P \\ P^T & 0 \end{bmatrix}$$

رابطه ۷-۲

برای اینکه ضرائب مجهول معادله‌ی $2 - 3$ را پیدا کنیم (یعنی ماتریس $(W | a_1 a_2 a_3)$ ، یا می‌توانیم L^{-1} را محاسبه کنیم یا سیستم خطی $Y = L(W | a_1 a_2 a_3)$ را حل کنیم. زمانی که مقادیر مجهول ماتریس $(W | a_1 a_2 a_3)$ یافت شد، به سراغ معادله‌ی $2 - 1$ می‌رویم و برای هر نقطه‌ی (x, y) ارتفاع $v = f(x, y)$ را به دست می‌آوریم.

تا اینجا درون‌یابی برای سطوح دو بعدی را بررسی کردیم، لازم به ذکر است که این درون‌یابی قابل تعمیم به ابعاد بزرگتر و n بعدی است. فقط لازم است که تعداد a ها را در رابطه‌ی $2 - 1$ بیشتر کنیم و از بردار¹¹ P بزرگتری استفاده کنیم. در نهایت هرچه نقاط کنترلی بیشتری داشته باشیم زمان بیشتری نیاز خواهد بود تا ارتفاع سطوح را درون‌یابی کنیم.

۱۰.۱.۲ تغییر فرم

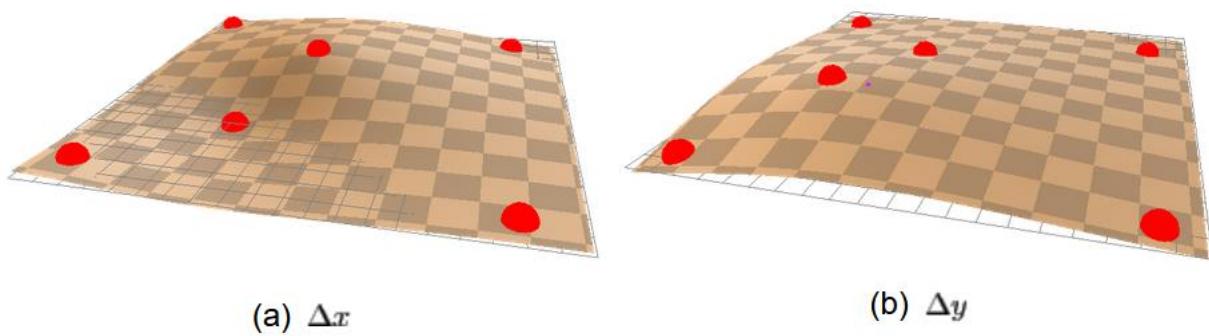
برای تغییر فرم یک عکس نیاز به دانستن مکان هر پیکسل در عکس A با مکان (x, y) چه مکانی در تصویر B ($x + dx, y + dy$) خواهد داشت. برای مثال در شکل $2 - 2$ ، با دانستن مکان لبخند فرد در عکس سمت چپ (x, y) باید بتوان مکان لبخند در عکس سمت (dx, dy) را محاسبه کرد. باید در نظر داشت که روش درون‌یابی گفته شده تنها ارتفاع سطح را محاسبه می‌کند. اما می‌دانیم که محور X ها و Y ها مستقل از هم می‌توانند درون‌یابی شوند. لذا یک سطح می‌تواند در نظر گرفت که dx و سطح دیگری را در نظر گرفت که dy را محاسبه کند.

¹¹ Vector



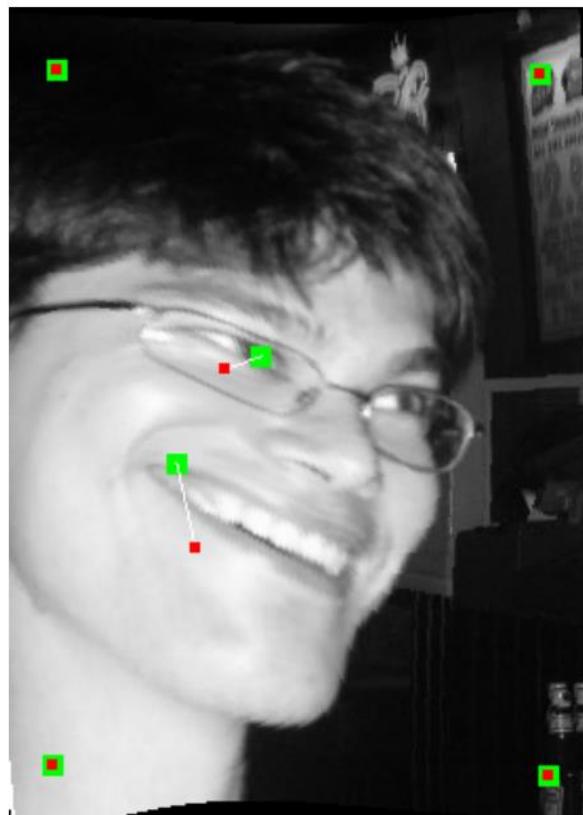
شکل ۲-۲

برای شکل ۲-۲ می‌توان ۶ نقطه در نظر گرفت، ۴ نقطه در گوشه‌های عکس، یک نقطه در چشم و یک نقطه در گوشه لب‌خند قرار می‌دهیم. گوشه‌های عکس تغییری نکرده‌اند، پس dx یا dy ای در این قسمت‌ها نخواهیم داشت و ارتفاعی ندارند. در حالی که برای نقطه لب‌خند، چون لب‌خند به سمت بالا است، dy ارتفاع گرفته است و dx نیز ارتفاع کمی گرفته است، چرا که لب‌خند ادکنی به سمت چپ نیز حرکت کرده است. برای نقطه‌ی چشم نیز چون چشم به سمت چپ حرکت کرده است dx ارتفاعی منفی گرفته است در حالی که dy ارتفاعی نزدیک به صفر دارد. می‌توان dx و dy را در سطوح شکل ۳-۲ نشان داد.



شکل ۳-۲

با داشتن دو سطح شکل ۲-۳ می‌توان مکان در هر نقطه تغییرات مکان (dx , dy) را محاسبه کرد. برای مثال در قسمت بینی تقریباً dx است در حالی که dy تغییرات چشمگیری داشته است. که یعنی بینی باید به سمت بالا حرکت کند اما تغییر خاصی در محور X نداشته باشد. درون یابی برای هر پیکسل شکل ۲-۴ نشان داده شده است.



شکل ۲-۲

دو سطح dx و dy نشان داده شده در شکل ۲-۳ کلیدهای اصلی هستند که تصویر A را به تصویر B تبدیل می‌کند. برای تبدیل تصویر A به تصویر B باید این سطوح پیدا شوند.

۲.۱.۲ نقاط هماهنگ

همان طور که گفته شد برای تبدیل تصویر A به تصویر B باید دو سطح صاف نازک پیدا شده در شکل ۲-۳ را پیدا کرد. برای یافتن ین سطح نیاز به نقاط کنترلی داریم و برای پیدا کردن این نقاط نیاز به دانستن تغییرات مکان در هر نقطه از عکس اول به عکس دوم داریم. لذا ما باید نقاط هماهنگ^۱ از یک عکس به عکس دیگر را پیدا کنیم.

^۱ Matching points

روش های زیادی برای پیدا کردن این نقاط هماهنگ وجود دارد. یکی از این روش ها استفاده از روش کمترین مربعات^۱ است. با داشتن محدوده W اطراف نقطه (x_0, y_0) در عکس A باید محدوده B کاملا مشابهی را در شکل B پیدا کنیم.

$$p(x, y) = \exp \left(- \sum_{i,j \in W} (A(x_0 + i, y_0 + j) - B(x + i, y + j))^2 / 2\sigma \right)$$

رابطه ۸-۲

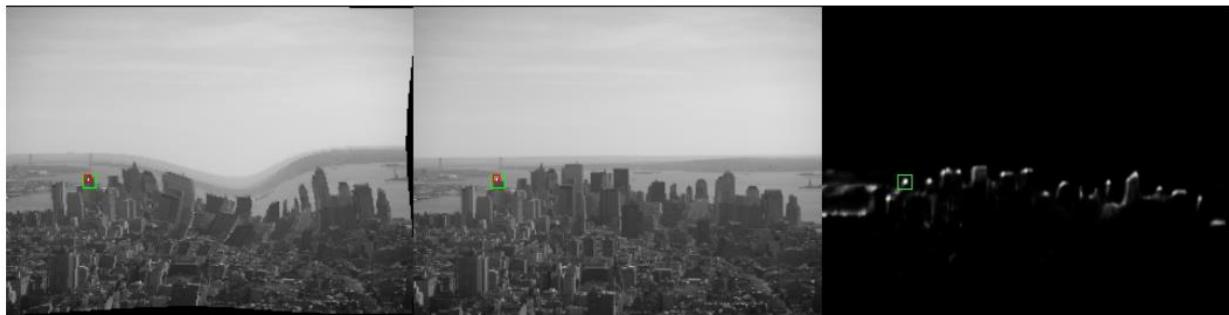
به این منظور می توان یک پنجره روی عکس B حرکت دهیم و تفاوت مربعات آن را با سطح کوچک در تصویر A محاسبه کنیم. این تفاوت نقاط هماهنگ در تصویر B را به ما می دهد.



شکل ۵-۲

برای مثال در شکل ۲ - ۵ اگر به دنبال منطقه سر ساختمان که در شکل سمت چپ مشخص شده است بگردیم، باید مقدار $p(x, y)$ را از طریق رابطه ۲ - ۸ محاسبه می گردد. مشاهده می شود که شباهت این قسمت با سر ساختمان ها در شکل سمت راست زیاد است. به این طریق می توان فهمید که این نقطه در شکل سمت چپ معادل کدام نقطه در شکل سمت راست است و ساختمان به کدام سمت حرکت کرده است.

^۱ Least square

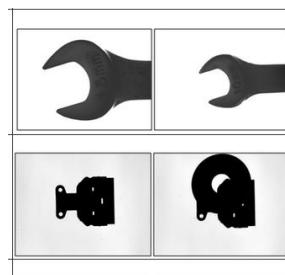


شکل ۶-۲

روش معمول دیگری که وجود دارد روش دستی است. یعنی به صورت دستی مشخص کنیم که هر نقطه چگونه و به کدام سمت جایه‌جا شده است.

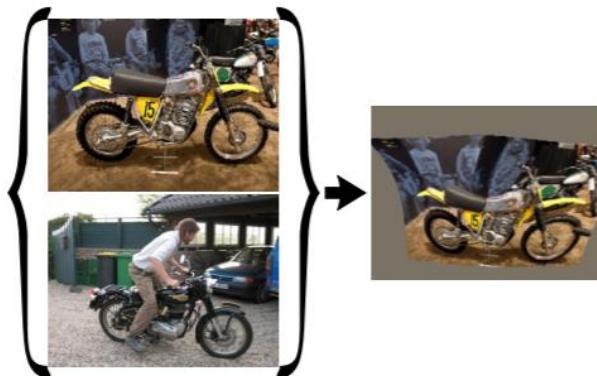
۳.۱.۲ تطابق دادن هندسی

تطابق هندسی^۱ در دو عکس مطابقت‌های هندسی را براساس ویژگی‌های هندسی پیدا می‌کند. این ویژگی‌ها شامل مواردی مانند لبه‌ها و خم‌ها و همچنین ویژگی‌های کلی تری مانند اشکال هندسی شامل دایره، مستطیل، مثلث و غیره که توسط خطوط، لبه‌ها و خم‌ها ایجاد می‌شود، است. در مواردی از تطابق هندسی استفاده می‌شود که شی‌ای در دو تصویر مختلف داشته باشیم که فقط از لحاظ اندازه تفاوت کرده یا شیء توسط جسمی بلاک شده است.



شکل ۷-۲

^۱ Geometric matching



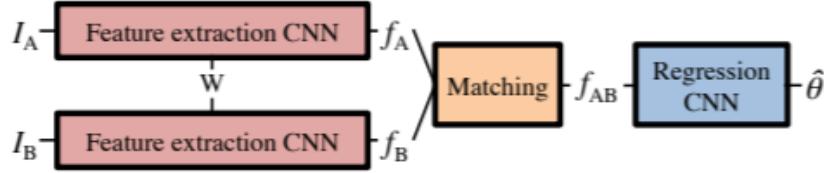
شکل ۴-۲- شبکه‌ی تخمین هندسی به صورت اتوماتیک دو عکس با ظاهرهای کاملاً متفاوت را با یکدیگر سازگار کرده است.

در این قسمت، یک شبکه عصبی جدید برای تخمین زدن پارامترهای لازم برای یک تبدیل هندسی که دو تصویر را به عنوان ورودی می‌گیرد معرفی می‌کنیم. این معماری به گونه‌ای طراحی شده است که از پایپلاین کلاسیک برای این منظور تقلید می‌کند. تفاوتی که با حالت کلاسیک دارد این است که از مازولهای مشتق پذیر استفاده می‌کند تا این عملیات تخمین قابل آموخته شدن^۱ شود [2].

پایپلاین کلاسیک شامل مراحل زیر بوده است:

۱. از هر دو عکس توصیف کننده‌های محلی استخراج می‌شوند. توصیف کننده‌هایی مانند SIFT یا تبدیل ویژگی مستقل از مقیاس از جمله این توصیف کننده‌ها هستند. SIFT یک الگوریتم در بینایی ماشین است که برای استخراج ویژگی‌های مشخص از تصاویر استفاده می‌شود. این توصیف کننده نسبت به مقیاس تصویر و چرخش ناوردادست. به صورت کلی در این الگوریتم ابتدا تصویر توسط فیلترهای گوسی در بازه‌ی ۱ تا ۲۰ محو می‌شود که خروجی آن تعدادی تصویر است. سپس این تصویرها از همسایگان خود کم می‌شوند تا یک سری جدید از تصویرها ایجاد شود.
۲. این توصیف کننده‌ها با هم تطبیق می‌یابند و مجموعه‌ای از شباهت‌های تجربی را ایجاد می‌کنند این شباهت‌ها در مرحله سه مورد استفاده قرار خواهد گرفت.
۳. در این مرحله از شباهت‌های تجربی مرحله دوم استفاده می‌شود تا پارامترهای مدل هندسی تخمین زده شود.

^۱ Trainable



شکل ۹-۲- نمودار معماری معرفی شده

معماری که در این بخش مطرح می‌شود، از پایپلاین کلاسیک به صورت زیر تقلید می‌کند:

۱. عکس‌های ورودی I_A و I_B وارد معماری که شامل لایه‌های کانولوشن^۱ است، می‌شوند و در نتیجه نقشه‌های ویژگی^۲ f_A و f_B استخراج می‌شوند که شبیه به توصیف کننده‌های محلی هستند.
۲. این نقشه‌های ویژگی با هم تطبیق می‌یابند و یک نقشه ویژگی شباهت‌های تجربی f_{AB} می‌شود.
۳. نقشه مرحله دوم وارد یک شبکه رگرسیون می‌شود که به صورت مستقیم، پارامترهای مدل هندسی، θ ، استخراج می‌شوند.

پس ورودی‌های این معماری دو عکس مختلف و خروجی آن پارامترهای مدل هندسی که می‌خواهیم از آن استفاده کنیم مثلاً تبدیل اسپلاین صفحات نازک، که یک وکتور ۶ بعدی است، خواهد بود. در ادامه هر مرحله در شکل ۲ - ۸ را بررسی و تحلیل می‌کنیم.

۱.۲ استخراج ویژگی

اولین مرحله از پایپلاین استخراج ویژگی است. برای این منظور یک معماری استاندارد شبکه عصبی کانولوشن یا CNN استفاده شده است. یک CNN همراه با یک لایه کاملاً متصل عکس ورودی را دریافت می‌کند و یک نقشه ویژگی^۳ $f \in R^{h \times w \times d}$ تولید می‌کند. این نقشه ویژگی می‌تواند به عنوان یک توری فضایی^۴ d بعدی، با اندازه W, x, h در نظر گرفته شود و گفت یک توصیف کننده محلی است. برای استخراج ویژگی از شبکه VGG-16 همراه با لایه‌های pool4 (قبل از لایه ReLU) استفاده شده است و نرمال‌سازی L2 پس از آن قرار گرفته است.

۱.۲ شبکه تطابق

ویژگی‌های عکس استخراج شده توسط شبکه استخراج ویژگی باید به یک تنسور تبدیل شوند و با هم ترکیب شوند و به عنوان ورودی شبکه رگرسیون به منظور تخمین پارامترهای تبدیل هندسی مورد استفاده قرار گیرند. ابتدا پیدا کردن

¹ Convolution layers

² Feature maps

³ Spatial grid

شباخت بین دو نقشه ویژگی به صورت کلاسیک را مطرح می‌کنیم. سپس روشی که قرار است به کار برده شود را توضیح می‌دهیم.

۱.۲ تطابق تجربی^۱ در تخمین هندسی کلاسیک

روش‌های کلاسیک با محاسبه‌ی شباخت بین تمامی جفت‌های توصیف کننده‌ها در دو عکس روش خود را آغاز می‌کنند. سپس تمامی توصیف کننده‌های ابتدایی و اصلی حذف خواهند شد زیرا اطلاعاتی که مورد نیاز و اصلی برای تخمین هندسی است از شباخت جفت توصیف کننده‌ها به دست می‌آید. در وهله‌ی بعدی این جفت‌ها با آستانه‌گذاری هرس خواهند شد یا تنها جفت‌هایی نگه داشته خواهند شد که بیشترین شباخت را به یکدیگر داشته باشند و نزدیک ترین همسایه به یکدیگر باشند. سپس تست دومین نزدیک‌ترین همسایه، جفت‌هایی که قدرت تطابق آن‌ها خیلی بیشتر از دومین بهترین تطبیق بین همان دو تصیف کننده است را هرس می‌کند.

۱.۲ لایه‌ی تطابق

لایه‌ی تطابق استفاده شده در این روش، سبکی مانند روش کلاسیک استفاده شده است. همانند روش کلاسیک فقط شباخت بین توصیف کننده‌ها و موقعیت‌های مکانی باید برای تخمین هندسی مورد استفاده قرار گیرد و خود توصیف کننده‌ها در نظر گرفته نمی‌شوند. برای این منظور یک لایه‌ی همبستگی^۲ که پس از آن یک لایه‌ی نرمال‌ساز قرار دارد استفاده می‌کنیم. ابتدا تمامی جفت‌های شباخت بین توصیف کننده‌ها در لایه‌ی همبستگی محاسبه می‌گردد. سپس امتیاز تشابه بین آن‌ها تحلیل می‌گردد و نرمال‌سازی می‌شود به طوری که تطابق‌های شبیه به هم به شدت وزن کمی دریافت خواهند کرد.

اگر جزئی‌تر نگاه کنیم، با داشتن نرمال‌سازی L2 نقشه‌های ویژگی $c_{AB} \in R^{h \times w \times d}$ نقشه وابستگی $f_A, f_B \in R^{h \times w \times (h \times w)}$ ، توسط لایه وابستگی به عنوان خروجی ایجاد می‌شود. در هر نقطه از این نقشه ضرب عددی هر جفت یک جفت از هر کدام از توصیف کننده‌ها $F_A \in f_A$ و $F_B \in f_B$ قرار دارد.

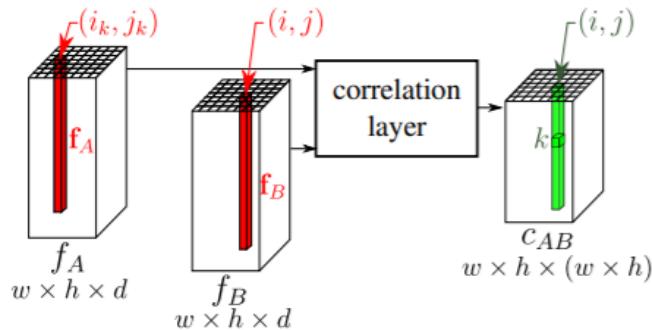
$$c_{AB}(i, j, k) = f_B(i, j)^T f_A(i_k, j_k)$$

که (i, j) و (i_k, j_k) مکان‌های منحصر به فرد هر ویژگی را در نقشه‌های w^* مشخص می‌کند و i_k ایندکس کمکی برای (i_k, j_k) است.

در شکل ۲-۹ نمودار لایه‌ی همبستگی نشان داده شد است. در نقشه همبستگی C_{AB} در یک مکان خاص (j, i) شباخت بین در همان نقطه و تمامی ویژگی‌های F_A قرار دارد.

¹ Tentative matches

² Correlation layer



شکل ۲ - ۱۰ - محاسبه نقشه همبستگی با ویژگی های CNN

همانطور که در روش کلاسیک برای تخمین شباهت انجام شد، پس اپردازش شباهت جفت ها به منظور حذف کردن تطابق های مشابه اهمیت زیادی دارد. به همین منظور یک نرمال سازی کanal به کanal روی هر مکان از نقشه همبستگی اعمال می کنیم تا نقشه همبستگی نهایی f_{AB} ایجاد شود. نرمال سازی توسط ReLU اجرا می شود تا همبستگی های منفی را به صفر تبدیل کند و پس از آن نرمال سازی L_2 قرار می گیرد. این کار در شرایطی که توصیف کننده f_B فقط با یک ویژگی از f_A ارتباط خوبی داشته باشد، نرمال سازی امتیاز تطابق را تقویت می کند. این عملیات دقیقاً مانند تطبیف نزدیک ترین همسایه در روش کلاسیک است. به علاوه در شرایطی که توصیف کننده f_B با چندین ویژگی از f_A تطابق پیدا کند، به علت وجود الگوهای تکراری یا درهم ریختگی، امتیاز های تطابق وزن کمتری پیدا خواهند کرد و این مانند تست دومین نزدیک ترین همسایه در روش کلاسیک است. باید در نظر داشت که هر دو عملیات های همبستگی و نرمال سازی نسبت به توصیف کننده های ورودی مشتق پذیرند که عملیات نشر به عقب^۱ را سهولت می بخشد.

۹.۱.۲ شبکه رگرسیون

نقشه همبستگی نرمال شده وارد یک شبکه رگرسیون می شود که به صورت مستقیم پارامترهای تبدیل هندسی مرتبط با دو عکس را تخمین می زند. در روش کلاسیک این مرحله شامل تخمین پارامترهای تبدیل از لیست شباهت های تجربی به دست آمده بود. سپس از محدودیت های هندسی محلی استفاده می شد تا لیست تطابق تجربی هرس شود و تنها تطابق هایی نگه داشته شود که با بقیه نتایج های در همسایگی مکانی یکپارچگی دارند. در نهایت تخمین نهایی با روش Hough voting صورت می گرفت.

دوباره به تقلید از روش کلاسیک یک شبکه عصبی استفاده می شود که دو بلوک لايه های کانولوشن را به صورت پشتی که پس از آن نرمال سازی دسته ای دارد، قرار می دهیم. پس از آن ReLU به منظور غیر خطی شدن و در نهایت یک لايهی کاملاً متصل اضافه می کنیم که پارامترهای تبدیل را محاسبه می کند. ایدهی پشت این شبکه عصبی این است که به صورت پایین به بالا این تخمین صورت می گیرد و این کار مانند روش کلاسیک hough voting است. لايه های ابتدایی برای تبدیل های

^۱ Backpropagation

کاندید احتمالی رای گیری می کنند و سپس در لایه های بعدی پردازش می شوند تا رای ها جمع بندی شوند. اولین لایه های کانولوشن همچنین همسایه های محلی را با یادگیری فیلترهایی که فقط اگر توصیف کننده های نزدیک در عکس A با توصیف کننده های نزدیک در عکس B تطابق یابند اجرا می شوند، ودار به اجماع کلی می کند.



شکل ۱۱-۲ - معماری شبکه‌ی رگرسیون.

۲.۲ شبکه‌های عصبی غیر محلی

هر دو عملیات‌های بازگشت^۱ و کونولوشن بلوک‌هایی هستند که یک همسایه‌ی محلی را پردازش می کنند. در این بخش عملیات‌های غیر محلی^۲ به عنوان خانواده‌ی بلوک‌های عمومی برای ثبت کردن وابستگی‌های رنج طولانی^۳ بررسی می شوند. ثبت کردن وابستگی‌های رنج طولانی اهمیت زیادی در شبکه‌های عصبی عمیق دارند. برای داده‌های ترتیبی مانند زبان و صوت عملیات‌های بازگشتی راه حل غالب برای مدل کردن وابستگی‌های رنج طولانی است. همچنین برای داده‌های تصویری وابستگی‌های فاصله زیاد^۴ از طریق پشته‌های عمیق عملیات‌های کانولوشن مدل‌سازی می شود. هر نوع این عملیات‌ها یک همسایگی محلی را پردازش می کند، یکی در راستای زمان و دیگری در راستای مکان. در نتیجه وابستگی‌های رنج بلند زمانی به دست می آیند که این عملیات‌ها به صورت مکرر انجام شوند و سیگنال‌ها به صورت پیشرونده داخل داده‌ها انتشار می یابند. اما این روش مشکلاتی دارد اول اینکه اینکار از لحاظ محاسبات به صرفه نیست، دوم اینکه اینکار موجب به وجود آمدن مشکلات بهینه‌سازی می شود؛ در نتیجه در این قسمت عملیات‌های غیر محلی تعریف می شوند که بری ذخیره و ثبت وابستگی‌های رنج-طولانی در شبکه‌های عصبی عمیق، کامپوننتی بهینه، ساده و عمومی هستند [3].

۱۰.۲.۲ فرمول‌سازی

یک عملیات غیر محلی عمومی در شبکه‌ی عصبی تعریف می کنیم:

$$\mathbf{y}_i = \frac{1}{\mathcal{C}(\mathbf{x})} \sum_{\forall j} f(\mathbf{x}_i, \mathbf{x}_j) g(\mathbf{x}_j). \quad (1)$$

رابطه ۹-۲

¹ Recurrent

² Non local

³ Long range dependency

⁴ Long distance

در معادله ۲ - ۹ شناسه برای مکان خروجی (در زمان، مکان، زمان-مکان) که پاسخ آن محاسبه می شود است و \hat{z} شناسه ای است که تمام مکان های ممکن را می شمارد. X سیگنال ورودی (عکس، ویدئو، توالی، اغلب ویژگی هایشان) و y سیگنال خروجی است که اندازه برابر X دارد. تابع دو متغیره f مقدار عددی بین ۰ و تمامی z ها را محاسبه می کند. تابع تک متغیره g نمایندگی سیگنال ورودی در مکان z را محاسبه می کند و پاسخ نهایی توسط فاکتور $(X)C$ نرمال سازی می گردد.

علت رفتار غیر محلی در معادله ۲ - ۹ این است که در این عملیات تمامی مکان ها (j) در نظر گرفته می شود. اگر بخواهیم با عملیات کانولوشن مقایسه کنیم، در کانولوشن جمع وزن دار در یک همسایگی محلی (برای یک کرنل یک بعدی با سایز ۳ داریم $i \leq j \leq i+1$) و در یک عملیات بازگشتی، مقدار در زمان i معمولاً براساس پله زمانی کنونی و آخرین پله زمانی است ($i = j$ یا $i = j-1$). عملیات غیر محلی با لایه ای کاملاً متصل تفاوت دارد. معادله ۲ - ۹ پاسخها را براساس رابطه ای بین مکان های مختلف محاسبه می کند، در حالی که لایه کاملاً متصل از وزن های یادگرفته شده استفاده می کند. در حقیقت برخلاف لایه ای کاملاً متصل در عملیات غیر محلی، رابطه ای بین x_i و z_j تابعی از داده هی ورودی نیست. به علاوه لایه کاملاً متصل نیاز به ورودی و خروجی با سایز مشخص دارد.

عملیات غیر محلی یک بلاک معطف است که می تواند در کنار لایه های بازگشتی یا کانولوشن به راحتی استفاده شود. این کمک می کند تا بتوان سلسه مراتبی ساخت که هر دو ویژگی های محلی و غیر محلی را با هم ترکیب می کند.

تابع های مختلفی می تواند برای f و g در معادله استفاده کرد. برای سادگی تابع g را به عنوان یک خطی نهفته می توان در نظر گرفت، $g(xj) = W_g x_j$ که W_g در آن ماتریس وزن هاست که یادگرفته می شود. همچنین برای f انتخاب های مختلفی مانند تابع گاسین^۱، تابع تعییه شده گاسین^۲ و ضرب داخلی^۳ و متصل کردن^۴ وجود دارد.

۲.۲.۲ بلاک غیر محلی

برای اینکه از عملیات غیر محلی توضیح داده شده استفاده شود، بلاک غیر محلی به صورت زیر تعریف می شود:

$$z_i = W_z y_i + x_i$$

که در این معادله y_i طبق معادله و x_i یک ارتباط باقی مانده را مشخص می کند. این ارتباط کمک می کند تا بلاک غیر محلی را در هر مدل از پیش یادگرفته، بدون آن که رفتار اولیه آن متفاوت شود، قرار دهیم [3].

¹ Gaussian

² Embedded gaussian

³ Dot product

⁴ Concatenation

۳.۲.۲ پیادهسازی بلاک غیر محلی

برای پیادهسازی این بلاک تعداد کانال‌های W_g و W_θ و W_ϕ (دو ماتریس آخر در تابع f استفاده می‌شوند) را برابر نصف کانال‌های X در نظر می‌گیریم و y_i را به صورت زیر تعریف می‌کنیم:

$$y_i = \text{softmax}(\hat{x}) P \sum_j f(x_i, \hat{x}_j) g(\hat{x}_j)$$

که در آن X یک نمونه از X است. همچنین W_z یک تابعی گذاری روی y_i به تفکیک مکان انجام می‌دهد.

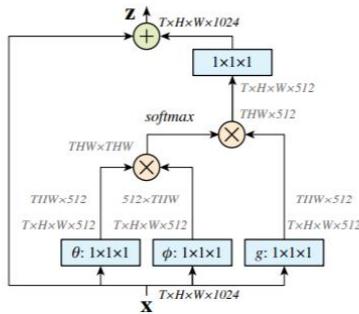


Figure 2. A spacetime **non-local block**. The feature maps are shown as the shape of their tensors, e.g., $T \times H \times W \times 1024$ for 1024 channels (proper reshaping is performed when noted). “ \otimes ” denotes matrix multiplication, and “ \oplus ” denotes element-wise sum. The softmax operation is performed on each row. The blue boxes denote $1 \times 1 \times 1$ convolutions. Here we show the embedded Gaussian version, with a bottleneck of 512 channels. The vanilla Gaussian version can be done by removing θ and ϕ , and the dot-product version can be done by replacing softmax with scaling by $1/N$.

شکل ۲ - ساختار یک بلاک غیر محلی در شکل مشاهده می‌شود.

۳.۲ پارس کردن و تخمین ژست بدن انسان

پارس کردن بدن انسان^۱ در سال‌های اخیر توجه زیادی را به علت پتانسیل زیاد تحقیقاتی که دارد، به خود جلب کرده است. هدف اینکار این است که عکس انسان را به قسمت‌های مختلف همراه با معانی مشخص (مثلاً دست، پا و ...) تقسیم و تکه تکه کرد به طوری که در کم بیشتری از جزئیات عکس داشته باشیم. همچنین تخمین ژست بدن انسان یک قسمت اصلی برای درک انسان‌ها در تصاویر و ویدئوها است. در کل الگوریتم‌های این زمینه می‌توان به سه دسته تقسیم کرد:

۱. پارس کردن لباس
۲. پارس کردن قسمت‌های مختلف بدن
۳. پارس کردن ژست بدن که شامل ژست دو بعدی، سه بعدی یا شکل بدن می‌شود.

¹ Human parsing

در این قسمت دو تا از روش‌های پاس کردن بدن انسان را که در ادامه به آن‌های نیاز داریم معرفی می‌کیم و به صورت مختصر روش استفاده از آن‌ها را بیان می‌کنیم.

۱.۳.۲ openPose

نام مقاله‌ای که در این بخش بررسی می‌شود تخمین دو بعدی چند-نفره در لحظه از طریق زمینه‌های قسمی وابستگی یا [4] است. در جامعه‌ی بینایی ماشین این کار اهمیت زیادی دارد زیرا:

۱. روشی آنی برای تخمین دو بعدی چند-نفره بر اساس روش پایین به بالا فراهم می‌کند. که این بر خلاف روش‌خای موجود است.

۲. نویسنده مقاله به صورت منبع-باز روش خود را منتشر کرده است همچنین کار خود را به حدی گسترش داده است که اولین سیستم آنی چند-نفره که مفاصل بدن انسان، دست، صورت و پا (به اندازه ۱۳۵ تا نقطه‌ی کلیدی) را ایجاد کرده است.

در این مقاله، نویسنده یک روش پایین به بالا استفاده می‌کند که در آن قسمت‌های مختلف بدن انسان توسط مدل مشخص می‌شوند و یک پارس کردن نهایی استفاده می‌شود تا نتایج تخمین ژست خروجی پیدا کند. اینکار پیچیدگی زمان اجرا را از تعداد افراد داخل عکس جدا می‌کند.

معماری این سیستم از چند بخش تشکیل شده است:

۱. اول عکس وارد یک شبکه‌ی پایه می‌گردد تا نقشه‌های ویژگی از آن استخراج شوند. در این مقاله نویسنده از ۱۰ لایه مدل VGG-19 استفاده کرده است.

۲. سپس این نقشه‌های ویژگی توسط چندین مرحله CNN پردازش می‌شوند تا دو چیز تولید شود:

a. مجموعه‌ای از نقشه‌های اطمینان قسمی^۱

b. مجموعه‌ای از زمینه‌های قسمی وابستگی^۲

۳. مجموعه‌ای از نقشه‌های اطمینان قسمی: مجموعه‌ای از نقشه‌های اطمینان دو بعدی برای مکان قسمت‌های مختلف بدن است. هر مفصل یک نقشه دارد.

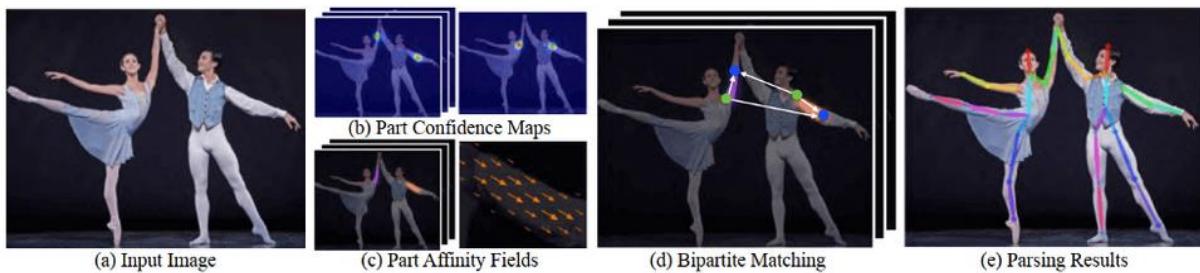
۴. مجموعه‌ای از زمینه‌های قسمی وابستگی: مجموعه‌ای از بردارهای دو بعدی است که درجه‌ی تجمع بین قسمت‌های مختلف را بررسی می‌کند.

¹ Open-source

² Part confidance maps

³ Part Affinity Fields (PAFs)

۵. در نهایت زمینه‌های قسمی وابستگی و نقشه‌های وابستگی توسط یک الگوریتم حریصانه پردازش می‌شوند تا ژست هر نفر در عکس را مشخص کنند.



شکل ۲-۱۳

پس به طور کلی:

ورودی‌های این کتابخانه: عکس پایه، مکان نقاط کلیدی به صورت فایل JSON و XML است.

خروجی‌های این کتابخانه: عکس پایه، مکان نقاط کلیدی به صورت فایل JSON و XML است.

```
{
  "version": 1.0,
  "people": [
    {
      "face_keypoints": [],
      "pose_keypoints": [
        179.1181102362205, 40.7272727272727, 0.907790213823318, 87.6850393700787, 91.2290909090909, 0.654397651553154, 49.3858267716535, 93.5563636363636, 0.493277914622102, 45.8582677165354, 153.6, 0.713929876685143, 29.9842519685039, 178.501818181818, 0.802610471844673, 127.748031496063, 87.5054545454545, 0.526449844241142, 142.614173228346, 159.650909090909, 0.792855083942413, 119.181102362205, 201.076363636364, 0.495145117864013, 42.8346456692913, 214.341818181818, 0.244036983649293, 0, 0, 0, 0, 0, 0, 92.7244094488189, 222.254545454545, 0.226115419587586, 0, 0, 0, 0, 0, 70.0472440944882, 31.1854545454545, 0.9622098035202, 89.9527559055118, 31.8836363636364, 0.945377916097641, 58.7086614173228, 36.77090909090909, 0.6546627222895622, 103.307086614173, 40.0290909090909, 0.842052660333054],
      "hand_right_keypoints": [],
      "hand_left_keypoints": []
    }
  ]
}
```

۲.۳.۲ به شخص نگاه کنید

مقاله‌ی بعدی که بررسی می‌شود به شخص نگاه کنید^۱ یا یادگیری حساس به ساختار و نظارت بر خود و یک معیار جدید برای تجزیه و تحلیل انسان است.

دیتاست‌هایی برای قسمت کردن بدن انسان هستند تعداد محدودی عکس دارند و تنوع حالت‌های مختلف انسان در آن‌ها محدود است. در این مقاله دیتاست و روشی ارائه شده است که از لحاظ تنوع، مقیاس و سختی پیشرفت زیادی داشته است [5]. دیتاست معرفی شده مقیاس بزرگی دارد که روی درک بدن انسان که ویژگی‌های مختلف زیادی دارد تمرکز می‌کند. این دیتاست ۵۰۴۶۲ عکس مختلف دارد و ۱۹ قسمت مختلف را در عکس تشخیص می‌دهد. در این دیتاست ژست‌های چالشی همراه با پشت‌زمینه‌های متنوع قرار دارد. ۱۹ قسمت مختلف در این روش کلاه، مو، عینک، لباس بالاتنه، پیراهن، کت، جوراب،

¹ Look into person, LIP

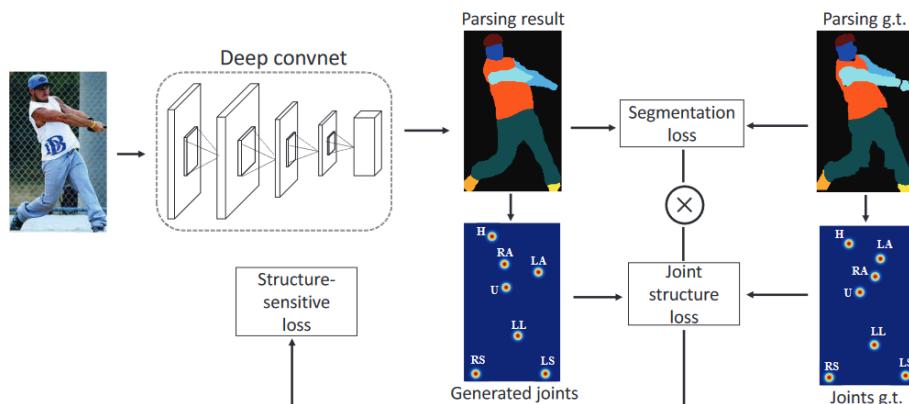
دستکش، شال، دامن، سوییشرت، صورت، بازوی راست، بازوی چپ، پای راست، پای چپ، کفش راست، کفش چپ و لیل برای پشتزمینه هستند.

Method	hat	hair	gloves	sunglasses	u-clothes	dress	coat	socks	pants	jumpsuits	scarf	skirt	face	l-arm	r-arm	l-leg	r-leg	l-shoe	r-shoe	Bkg	Avg
SegNet [3]	26.60	44.01	0.01	0.00	34.46	0.00	15.97	3.59	33.56	0.01	0.00	0.00	52.38	15.30	24.23	13.82	13.17	9.26	6.47	70.62	18.17
FCN-8s [21]	39.79	58.96	5.32	3.08	49.08	12.36	26.82	15.66	49.41	6.48	0.00	2.16	62.65	29.78	36.63	28.12	26.05	17.76	17.70	78.02	28.29
DeepLabV2 [4]	57.94	66.11	28.50	18.40	60.94	23.17	47.03	34.51	64.00	22.38	14.29	18.74	69.70	49.44	51.66	37.49	34.60	28.22	22.41	83.25	41.64
Attention [5]	58.87	66.78	23.32	19.48	63.20	29.63	49.70	35.23	66.04	24.73	12.84	20.41	70.58	50.17	54.03	38.35	37.70	26.20	27.09	84.00	42.92
DeepLabV2 + SSL	58.41	66.22	28.76	20.05	62.26	21.18	48.17	36.12	65.16	22.94	14.84	19.37	70.01	50.45	53.39	37.59	36.96	26.29	26.87	83.67	42.44
Attention + SSL	59.75	67.25	28.95	21.57	65.30	29.49	51.92	38.52	68.02	24.48	14.92	24.32	71.01	52.64	55.79	40.23	38.80	28.08	29.03	84.56	44.73

شکل ۱۴-۲ - مقایسه‌ی روش‌های مختلف

در شکل ۲-۱۴ روش‌های مختلف بر روی این دیتابست را مشاهده می‌کنیم.

دو روش آخر روشی است که در مقاله‌ی به شخص نگاه کنید به صورت خاص استفاده شده است که در آن از یک تابع هزینه‌ی جدید استفاده کرده است و نتایج بهتری به دست آورده است. این تابع هزینه، هزینه‌ی خود-ناظر حساس به ساختار یا SSL نامیده می‌شود. این هزینه کیفیت نتایج بخش‌بندی را از یک دیدگاه ساختاری مورد بررسی قرار می‌دهد. در حقیقت علاوه بر روش‌های به تفکیک پیکسل مفصل‌های تخمینی انسان به صورت مستقیم از عکس استخراج می‌شوند برای این که به صورت مستقیم نتایج بخش‌بندی معنایی سازگار با ساختار مفاصل انسان تولید شود، با وزن هزینه‌ی ساختاری به عنوان وزن هزینه‌ی بخش‌بندی معنایی برخورد می‌شود.



شکل ۲-۱۵ - هزینه‌ی SSL با شکل

برای استفاده از این روش به صفحه‌ی گیت‌هاب این مقاله می‌توان مراجعه کرد [6].

۴.۲ بلاک‌های اکسپشن و آغاز (Inception)

در این بخش هدف اصلی توضیح بلاک اکسپشن^۱ است. از آن جایی که ارتباطی بین بلاک اکسپشن و اینسپشن^۲ وجود دارد و اینسپشن پیش‌نیاز اکسپشن است ابتدا به توضیح بلاک اینسپشن می‌پردازیم.

۱۰.۴.۲ بلاک اینسپشن

شبکه‌ی اینسپشن یک نقطه‌ی مهم در ایجاد مرتب‌سازهای CNN بوده است. قبل از این بلاک بیشتر CNN‌ها فقط تعدادی لایه‌ی کانولوشن بوده‌اند که عمیق‌تر و عمیق‌تر می‌شدند و امید داشتیم که خوب اجرا شوند. در کل چهار ورژن مختلف برای بلاک اینسپشن وجود دارد. گوگل در توضیح ورژن چهار به مقایسه و تعریف مختصر هر کدام از این ورژن‌ها پرداخته است.

معماری اینسپشن عمیق به عنوان GoogLeNet تحت عنوان inception-v1 معرفی شد. سپس این معماری به روش‌های مختلف اصلاح گردید. ابتدا با معرفی نرم‌افزاری بسته‌ای^۳ (inception-v2) سپس به اضافه کردن فاکتورگیری در سومین تکرار^۴ که به عنوان inception-v3 [7] شناخته می‌شود.

پس از آن inception-v3 با ایده‌های فاکتورگیری مطرح می‌شود.

کانولوشن فاکتورساز: هدف کانولوشن فاکتورساز^۵ کاهش تعداد پارامترها بدون کم کارکرد شبکه است.

فاکتور کردن به کانولوشن‌های کوچک‌تر: یک کانولوشن ۵ در ۵ توسط دو کانولوشن ۳ در ۳ به صورت شکل ۱۶-۲ جایگزین می‌شود:

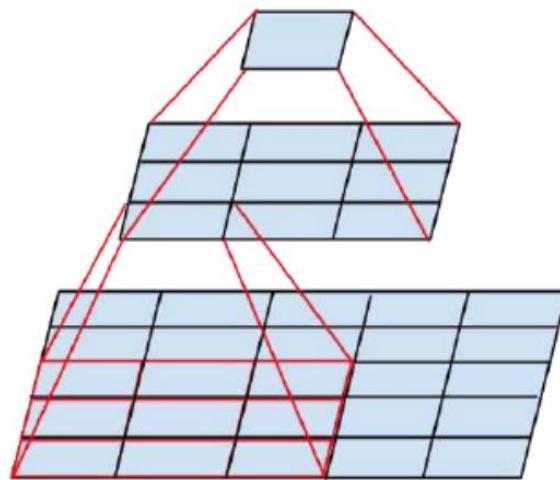
¹ Xception

² Inception

³ Batch normalization

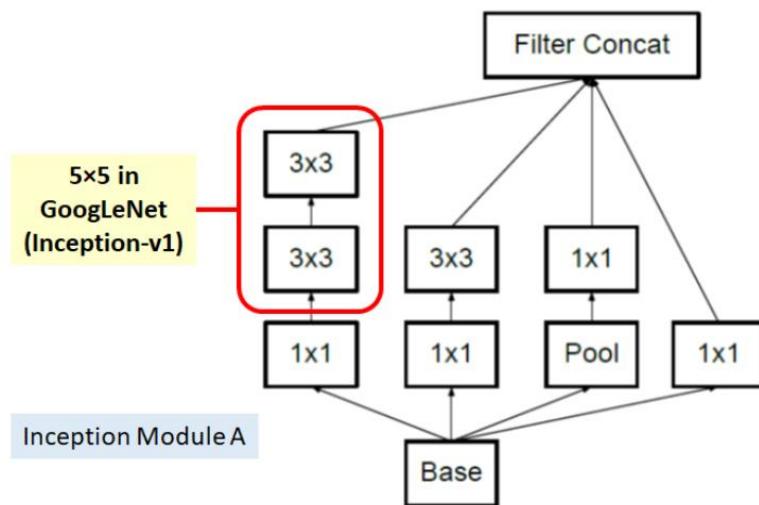
⁴ Iteration

⁵ Factorizing convolution



شکل ۱۶-۲

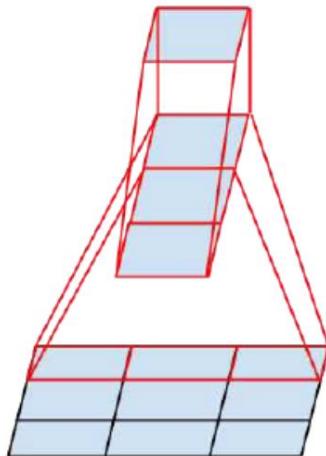
با استفاده کردن یک لایه‌ی ۵ در ۵ تعداد پارامترها ۲۵ خواهد بود در حالی که اگر دو لایه‌ی فیلتر ۳ در ۳ استفاده کنیم، تعداد پارامترها $3 \times 3 + 3 \times 3 = 18$ خواهد بود. در نتیجه پارامترها به اندازه ۲۸٪ کاهش می‌یابد. با این تکنیک یکی از ماثوله‌ای اینسپشن به صورت شکل ۱۷-۲ خواهد بود:



شکل ۱۷-۲

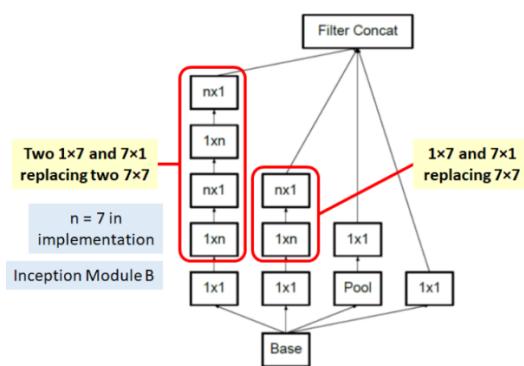
۲.۴.۲ فاکتور گیری توسط کانولوشن‌های غیر متقاض

یک لایه کانولوشن 3×3 و سپس یک لایه کانولوشن 1×3 به دنبال آن می‌تواند یک لایه کانولوشن 3×3 را جایگزین کند.



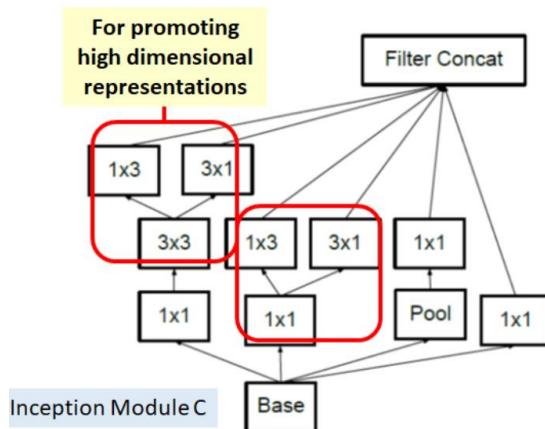
شکل ۱۶-۲

با استفاده کردن فیلتر ۳ در ۳ تعداد پارامترها ۹ تاس در حالی که اگر از یک فیلتر ۱ در ۳ استفاده کنیم تعداد پارامترها ۶ خواهد بود. پس ۳۳٪ از تعداد پارامترها کاهش یافت. سوالی که مطرح می‌شود این است که چرا از فیلترهای ۲ در ۲ استفاده نمی‌شود؟ پاسخ این است که اگر از دو فیلتر ۲ در ۲ استفاده شود تعداد پارامترها $2 \times 2 \times 2 = 8$ خواهد بود و تنها ۱۱٪ از تعداد پارامترها کاسته خواهد شد. با این تکنیک یکی دیگر از مازولهای اینسپشن به صورت شکل ۲-۱۹ خواهد بود:



شکل ۱۹-۲

ماژول بعدی برای ترقی دادن نمایش ابعاد بزرگتر به صورت شکل ۲۰-۲ تعریف می‌شود:

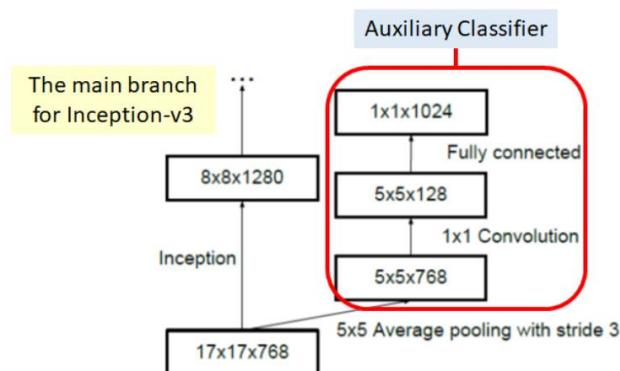


شکل ۲۰-۲

با این سه ماژول اینسپشن و فاکتورسازی تعداد پارامترها برای کل شبکه کاهش می‌یابد و احتمال تطابق زیاد^۱ کمتر می‌شود. در نتیجه شبکه می‌تواند عمیق‌تر شود.

۳.۴.۲ مرتب‌ساز کمکی

مرتب‌سازهای کمکی^۲ برای inception-v1 معرفی شده بودند اما تغیراتی برای inception-v3 وجود دارد. تنها یک مرتب‌ساز کمکی بر روی آخرین لایه‌ی 17×17 قرار می‌گیرد به جای اینکه دو مرتب‌ساز کمکی قرار گیرد.



شکل ۲۱-۲

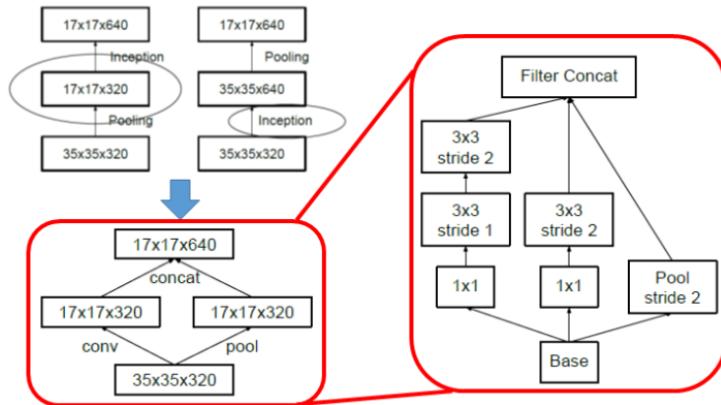
¹ Overfitting

² Classifier

هدف این مرتب‌ساز در inception-v3 با inception-v1 متفاوت است. در inception-v1 به این منظور استفاده می‌شد تا شبکه‌ی عمیق‌تری داشته باشیم در حالی که در inception-v3 به عنوان منظم‌کننده^۱ استفاده می‌شود. نرمال‌سازی دسته‌ای معرفی شده در inception-v2 نیز در مرتب‌ساز کمکی استفاده می‌شود.

۴.۴.۲ کاهش سایز شبکه‌بندی به صورت بهینه

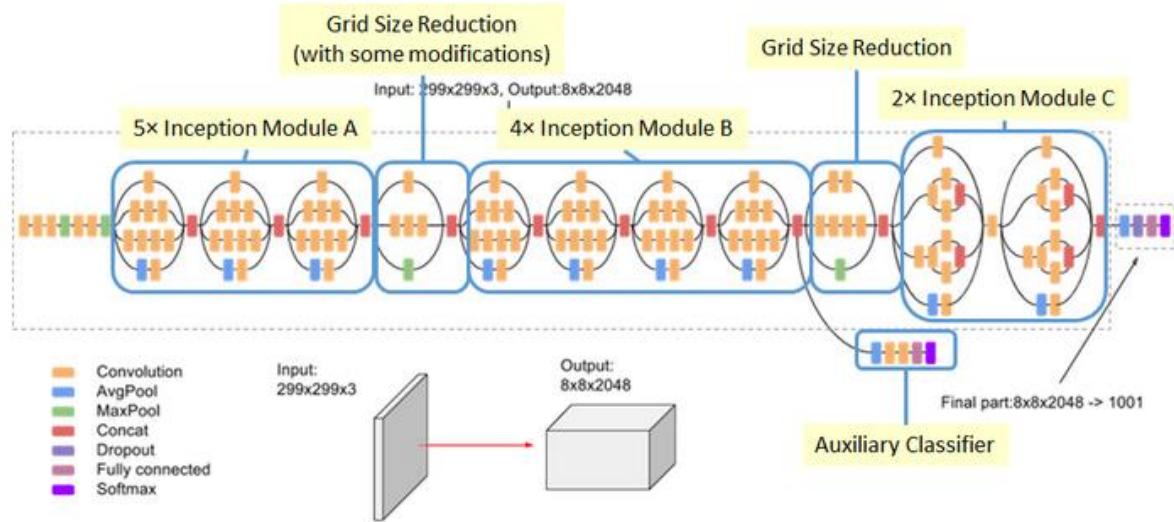
به صورت معمول کاهش سایز نقشه‌ی ویژگی با ماکس‌پولینگ^۲ صورت می‌گیرد. اما مشکلی که دارد این است که اگر یک لایه‌ی پولینگ سپس یک لایه‌ی کانولوشن وجود داشته باشد بسیار حرج‌صانه می‌شود یا اگر ابتدا لایه‌ی کانولوشن باشد سپس لایه‌ی پولینگ بسیار هزینه‌بر خواهد بود. کاهش بهینه‌ی سایز گرید به صورت زیر است:



شکل ۲۲-۲

¹ Regularize

² Max pooling

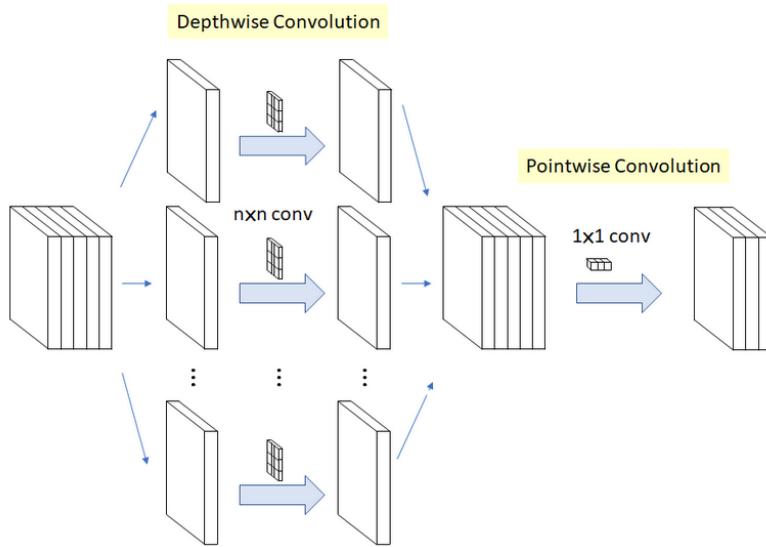


شکل ۲۳-۲

۶.۴.۲ بلاک اکسپشن

در این بخش بلاک اکسپشن را بررسی می کیم. اکسپشن در حقیقت مختصر اینسپشن شدید^۱ است [8].

۷.۴.۲ کانولوشن جداپذیر به تفکیک عمق اوریجینال



شکل ۲۴-۲

^۱ Extreme inception

پس از کانولوشن جداپذیر به تفکیک عمق به صورت اوریجینال یک کانولوشن به تفکیک نقطه^۱ قرار می‌گیرد.

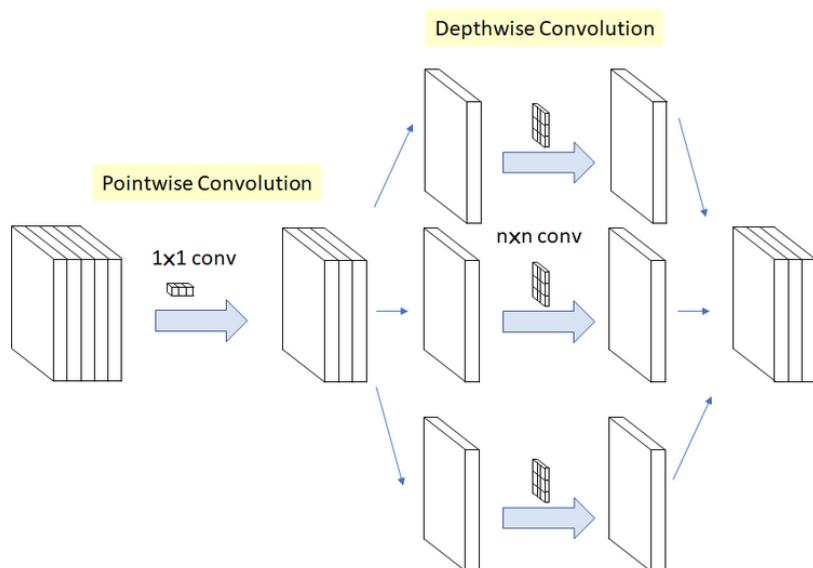
۱. کانولوشن به تفکیک عمق یک کانولوشن مکانی n در n به تفکیک کanal است. مثلاً در شکل بالا ۵ کanal داریم

سپس ۵ کانولوشن مکانی n در n خواهیم داشت.

۲. کانولوشن به تفکیک نقطه در حقیقت یک کانولوشن ۱ در ۱ به منظور تغییر ابعاد است.

در مقایسه با مدل‌های کانولوشن نیاز به اجرای کانولوشن در راستای تمامی کanal‌ها نداریم. در نتیجه تعداد اتصالات کمتر می‌شود و مدل سبک‌تر خواهد شد.

کانولوشن جداپذیر به تفکیک عمق تغییر یافته در اکسپشن



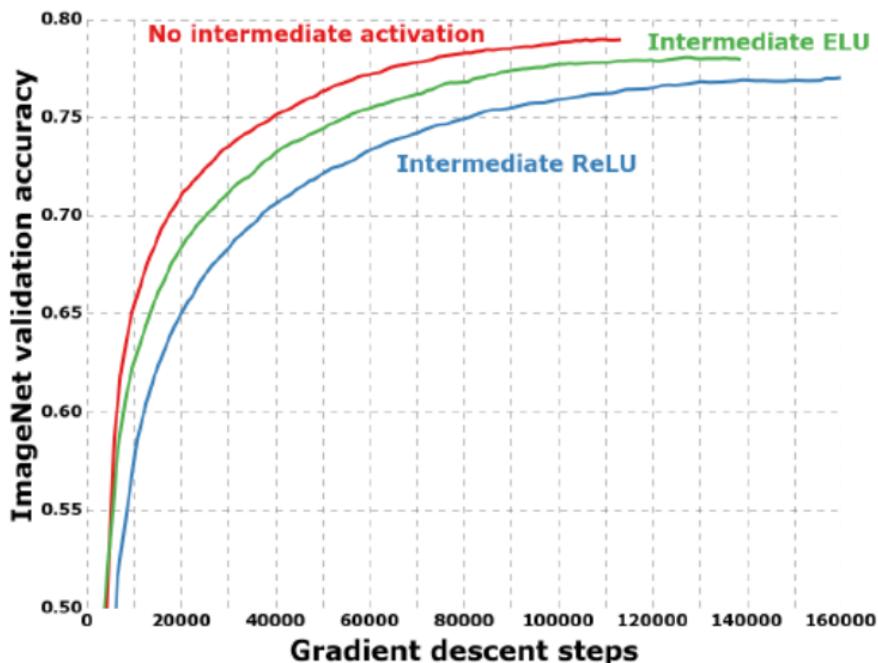
شکل ۲۰-۲

ساختار کانولوشن جداپذیر به تفکیک عمق تغییر یافته به این صورت است که ابتدا کانولوشن به تفکیک نقطه است و سپس پس از آن کانولوشن به تفکیک عمق وجود دارد. این تغییرات از ایده‌ی ماژول inception-v3 در ماژول گرفته شده است که ابتدا یک کانولوشن ۱ در ۱ قبل از هر کانولوشن مکانی n در n قرار بگیرد. در نتیجه دو تفاوت کوچک نسبت به حالت اوریجینال وجود دارد.

^۱ pointwise

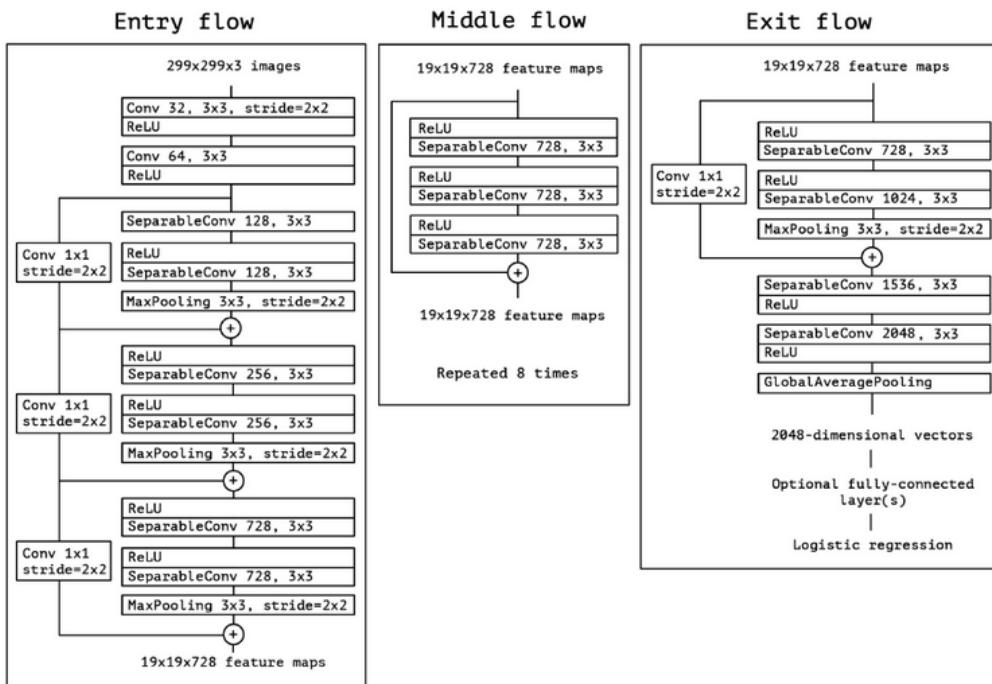
۱. ترتیب عملیات‌ها: همانطور که گفته شد کانولوشن جداپذیر به تفکیک عمق معمولاً به این صورت پیاده‌سازی می‌شود که ابتدا کانولوشن به تفکیک کانال مکانی اعمال می‌شود سپس کانولوشن ۱ در ۱ در حالی که در این حالت تغییریافته، ابتدا کانولوشن ۱ در ۱ اعمال می‌گردد سپس کانولوشن مکانی به تفکیک کانال صورت می‌گیرد.

۲. وجود یا عدم وجود غیر خطی بود: در مژول inception اصلی هیچ‌گونه عملیات غیر خطی پس از اولین عملیات وجود ندارد اما در مژول اکسپشن، کانولوشن جداپذیر به تفکیک عمق تغییر یافته، هیچ‌گونه ReLU واسط غیر خطی وجود ندارد.



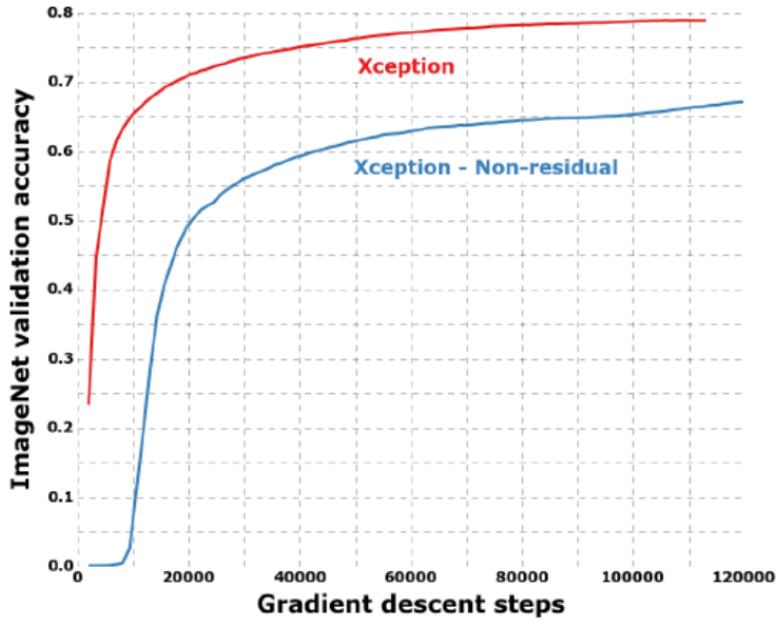
شکل ۲۶-۲

کانولوشن جداپذیر به تفکیک عمق تغییر یافته با تابع‌های مختلف فعال‌ساز امتحان شده‌اند. همانطور که در نمودار مشخص است اکسپشن بدون هیچ‌گونه فعال‌ساز خطی در مقایسه با آن‌هایی که از ReLU یا ELU استفاده کرده‌اند، بالاترین دقت را دارد.



۲۱۷-۲

همانطور که در شکل دیده می‌شود، SeparableConv همان کانولوشن جداپذیر به تفکیک عمق تغییر یافته است. با این بلاک‌ها به صورت مازول inception برخورد می‌شود و درون معماری کلی یادگیری عمیق قرار گرفته است. همچنین اتصالات residual (میانبر) در تمامی جریانات قرار گرفته است.



شکل ۲۱-۲

همانطور که دیده می شود، اکسپشن بدون میانبر دقیق بسیار کمتری نسبت به اکسپشن با میانبر دارد. در نتیجه اتصالات residual بسیار اهمیت دارند.

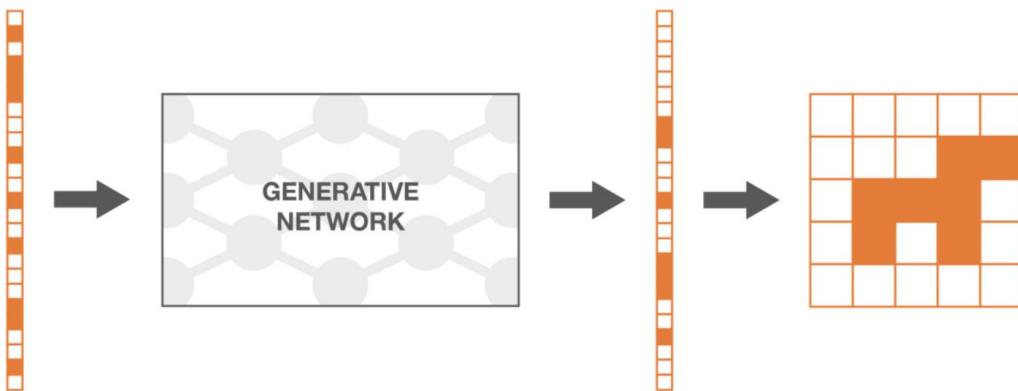
۲.۵ مدل‌های تولید کننده

فرض کنید می خواهیم یک عکس مربعی سیاه و سفید از سگ‌ها تولید کنیم که اندازه‌ی آن $n \times n$ است. می توان هر داده را تغییر اندازه داد تا یک وکتور با اندازه‌ی $n = N$ ایجاد شود و عکس سگ به صورت وکتور نشان داده شود. اما تمامی وکتورها با این اندازه شکل سگ را ایجاد نخواهند کرد بلکه آن‌هایی شکل سگ را ایجاد خواهند کرد که وکتور آن‌ها طبق یک توزیع احتمالاتی خاص در فضای N بعدی توزیع شده باشد. در نتیجه مسئله‌ی تولید یک عکس جدید از سگ را می توان معادل تولید یک وکتور جدید که از توزیع احتمالاتی سگ در فضای N بعدی پیروی می کند دانست. این همان مسئله‌ی تولید مقادیر رندوم با توجه به توزیع احتمالاتی خاص است. دو نکته اینجا حائز اهمیت است:

۱. توزیع احتمالاتی سگ در فضای N بعدی بسیار پیچیده و در یک فضای بسیار بزرگ است.
۲. اگر فرض کنیم چنینی توزیعی وجود دارد هنوز نمیدانیم چطور باید این توزیع را توصیف کرد.

با این حال تا حدی میدانیم چگونه به تعداد N مقادیر رندوم یکپارچه غیرهمبسته تولید کنیم. برای اینکار باید مقادیر رندوم N بعدی را تحت نتیجه‌ی یکتابع بسیار پیچیده که بر یک فضای N بعدی مقادیر رندوم اعمال شده است، توصیف کرد. مشخص

است که پیدا کردن تابع تبدیل به سادگی نیست. این تابع به صرعت مستقیم توصیف نمیشود و نیاز دارد تا از داده‌ها آموخته شود. مانند سایر موارد در این شرایط، تابع بسیار پیچیده به صورت طبیعی استفاده از شبکه‌های عصبی را نتیجه می‌دهد. در نتیجه ایده به این صورت می‌شود که تابع تبدیل توسط یک شبکه‌ی عصبی که مقادیر رندوم N بعدی ساده را ورودی می‌گیرد مدل‌سازی شود. خروجی این شبکه‌ی عصبی پس از آموزش یک متغیر N بعدی رندوم است که طبق توزیع درست احتمالاتی سگ خواهد بود. پس از اینکه معماری این شبکه طراحی شود، این شبکه آموزش داده می‌شود. دو نوع روش برای آموزش این شبکه‌های تولید کننده وجود دارد که یکی از آن‌ها شبکه‌های مولد خصمانه یا GAN است.



Input random variable
(drawn from a simple distribution, for example uniform).

The generative network transforms the simple random variable into a more complex one.

Output random variable
(should follow the targeted distribution, after training the generative network).

The output of the generative network once reshaped.

شکل ۲۹-۲

۱۰.۵.۲ آموزش مدل‌های تولید کننده

تا کنون نشان دادیم که تولید عکس جدید سگ می‌تواند با مسئله‌ی تولید و کنور رندوم در فضای N بعدی به طوری که توزیع احتمالاتی سگ پیروی کند جایگزین شود و نتیجه گرفتیم که می‌توان از شبکه‌ی عصبی برای مدل کردن تابع تبدیل استفاده کرد.

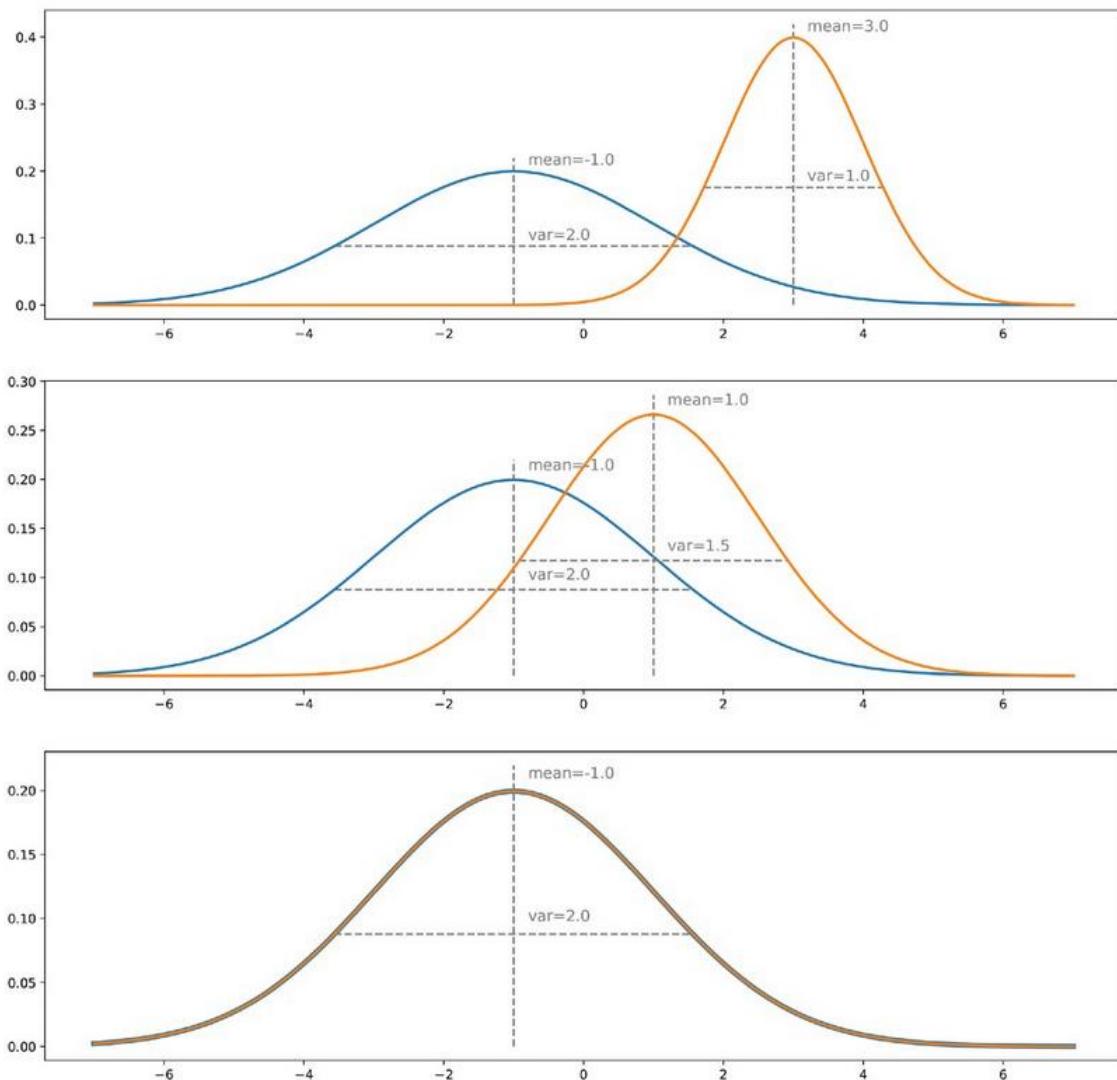
برای پیدا کردن تابع تبدیل درست باید شبکه‌ی عصبی را آموزش دهیم و بهینه کنیم. برای این منظور می‌توان از دو روش استفاده کرد: روش مستقیم و روش غیرمستقیم. روش آموزش مستقیم شامل مقایسه‌ی توزیع احتمالاتی حقیقی و توزیع احتمالاتی تولید شده و بازنثر اختلاف (خطا) در میان شبکه است. در روش غیرمستقیم مقایسه‌ی توزیع احتمالاتی حقیقی و توزیع احتمالاتی تولید شده به صورت مستقیم نخواهد بود. در این روش شبکه‌ی مولد به گونه‌ای آموزش دیده می‌شود که هر دو توزیع وارد یک عملیات منتخب می‌شود به طوری که پروسه‌ی بهینه‌سازی شبکه‌ی مولد با توجه به عملیات مذکور، توزیع

احتمالاتی تولید شده را به توزیع احتمالاتی حقیقی نزدیک می کند. این همان ایده‌ی پشت شبکه‌های تولید کننده خصمانه یا GAN است.

۲.۵.۲ شبکه‌های مولد خصمانه

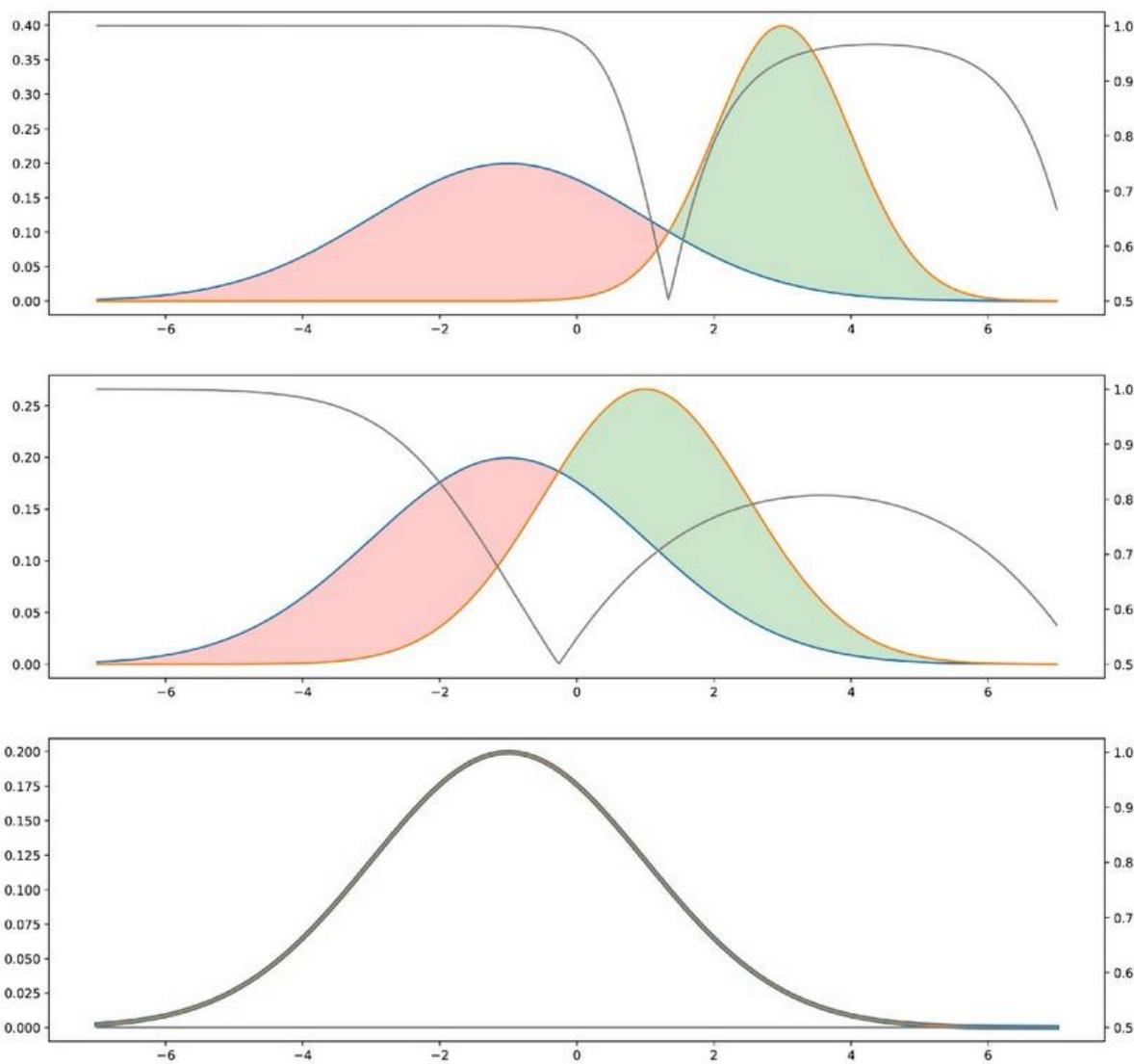
ایده‌ی جالب پشت شبکه‌های مولد خصمانه جایگزین کردن روش مقایسه‌ی مستقیم با روش غیرمستقیم است که نوعی عملیات بین دو توزیع را در نظر می گیرد و آموزش را متناسب با آن انجام می دهد. این عملیات در GAN یک جدا کننده بین نمونه‌های حقیقی و غیرحقیقی است. به نوعی می توان نام غیر جدا کننده رو به این جدا کننده داد! چون هدف این است که این جدا کننده تا جایی که می شود در کار خود شکست بخورد. پس در معماری GAN یک جدا کننده داریم که نمونه‌ی حقیقی و تولید شده را می گیرد و سعی می کند تا جایی که می شود آنها را کلاس بندی کند و یک مولد که آموزش می بیند تا جدا کننده را گول بزند.

فرض کنید که توزیع حقیقی داریم مثلا یکتابع گوسی یک بعدی و یک مولد می خواهیم که از این توزیع احتمالاتی نمونه گیری کند. آموزش مستقیم برای اینکه اختلاف بین دو توزیع تولید شده و حقیقی را کاهش دهد به صورت مکرر مولد را تغییر می دهد. در نهایت به یک توزیع تولیدی می رسد که مطابق با توزیع حقیقی است.



شکل ۳۰-۲

در روش غیر مستقیم باید یک جدا کننده در نظر بگیریم. اگر دو توزیع زیاد با هم اختلاف داشته باشند جدا کننده قابلیت این را دارد که با اطمینان زیاد بسیار از نقاطی که به عنوان ورودی می‌گیرد را به راحتی کلاس‌بندی کند. اگر بخواهیم جدا کننده را گول بزنیم باید توزیع تولیدی را به توزیع حقیقی نزدیک کنیم. در نتیجه جدا کننده به سختی می‌تواند کلاس را پیش‌بینی کند چون هر دو توزیع به هم بسیار نزدیک‌اند و برای هر نقطه شанс یکسانی برای حقیقی بودن و تولیدی بودن وجود دارد.



شکل ۲-۳۱

استفاده از یک جداکننده روش پیچیده‌ای است و مشخصاً جدا کننده ناشناخته است و باید آموزش دیده شود.

۳.۵.۲ تقریب: شبکه‌های عصبی خصم‌مانه

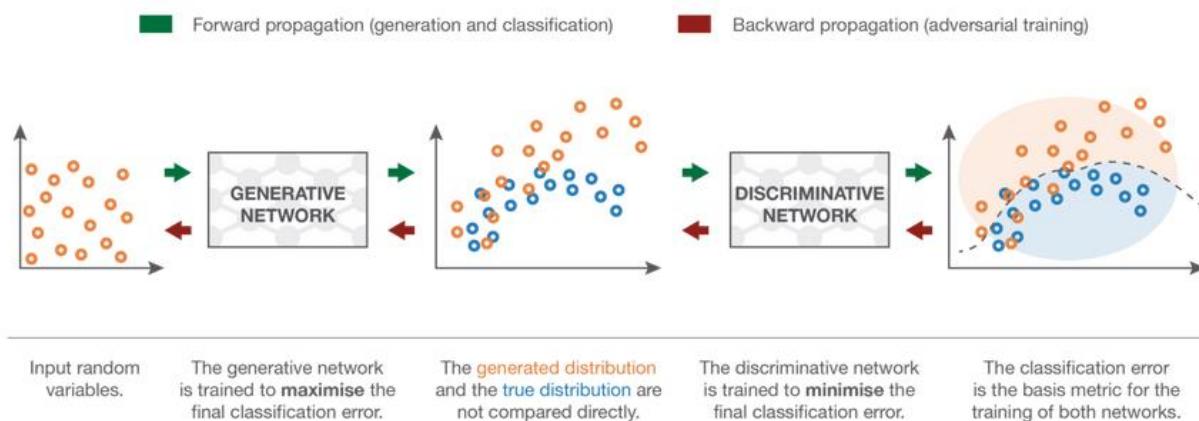
شبکه‌ی مولد یک شبکه است کهتابع تبدیل را مدل می‌کند. این شبکه به عنوان ورودی یک متغیر تصادفی ساده دریافت می‌کند و پس از آموزش باید متغیری که از همان توزیع پیروی می‌کند خروجی دهد. چون اینکار پیچیده‌ای است جداکننده با یک شبکه‌ی عصبی دیگر مدل می‌شود. این شبکه یک تابع جداساز را مدل می‌کند. این تابع یک ورودی (نقطه‌ای در فضای N بعدی) دریافت می‌کند و احتمال اینکه این ورودی حقیقی باشد را خروجی می‌دهد.

این دو شبکه می‌توانند با یکدیگر با اهداف کاملاً مخالف آموزش داده شوند:

۱. هدف تولید کننده گول زدن جداساز است لذا این شبکه آموزش داده می‌شود تا خطای نهایی کلاس‌بندی را کاهش دهد.

۲. هدف جداساز این است که نمونه‌های مصنوعی تولید شده را تشخیص دهد پس آموزش داده می‌شود تا خطای نهایی کلاس‌بندی را مینیمم کند.

پس در هر تکرار در مراحل آموزش وزن‌های شبکه‌های مولد به روزرسانی می‌شوند تا خطای کلاس‌بندی افزایش پیدا کند در حالی که وزن‌های شبکه‌ی جداساز به روزرسانی می‌شوند تا این خطای کاهش یابد (خطای گرادیان کاهشی بر روی پارامترهای جداساز).



شکل ۲_۳۲

این اهداف متفاوت نشان می‌دهد چرا به این نوع آموزش مدل‌های مولد خصمانه گفته می‌شود. هر دو شبکه تلاش می‌کنند بهتر و بهتر شوند و با یکدیگر رقابت می‌کنند. از دیدگاه تئوری بازی می‌توان به این ماجرا به عنوان تنظیمات minmax برای دو بازیکن یک بازی نگاه کرد.

در شکل شبکه‌های عصبی برای ایجاد شدن به دو چیز نیاز دارند: یکی معماری و دیگری تابع هزینه. تا اینجا معماری شبکه‌های مولد خصمانه بررسی شدند. این شبکه‌ها دو مدل شبکه را مشمول می‌شدند:

۱. شبکه‌ی تولید کننده (G) که ورودی رندوم Z را با چگالی p_Z دریافت می‌کند و خروجی ($x_g = G(z)$) که باید توزیع احتمالاتی هدف را دنبال کند ایجاد می‌کند.

۲. شبکه‌ی جداساز (D) که ورودی X را دریافت می‌کند که می‌تواند یک ورودی حقیقی (x_t با چگالی p_t) باشد یا یک ورودی مصنوعی و تولید شده (x_g با چگالی p_g که همان چگالی ایجاد شده p_z عبوری از G است) باشد و در نهایت احتمال ($D(x)$) که احتمال حقیقی بودن X را مشخص می‌کند خروجی دهد.

اکنون به تابع هزینه‌ی شبکه‌های مولد خصم‌مانه می‌پردازیم. اگر به یک نسبت به شبکه‌ی جداساز ورودی صحیح و تولیدی دهیم خطای این شبکه به صورت رابطه ۱۰-۲ توصیف می‌شود:

$$\begin{aligned} E(G, D) &= \frac{1}{2} \mathbb{E}_{x \sim p_t} [1 - D(x)] + \frac{1}{2} \mathbb{E}_{z \sim p_z} [D(G(z))] \\ &= \frac{1}{2} (\mathbb{E}_{x \sim p_t} [1 - D(x)] + \mathbb{E}_{x \sim p_g} [D(x)]) \end{aligned}$$

رابطه ۱۰-۲

هدف تولید کننده گول زدن جداساز است پس هنگام آموزش تولید کننده باید این خطا را افزایش دهیم در حالی که داریم سعی می‌کنیم این خطا را برای جداساز کاهش دهیم.

$$\max_G \left(\min_D E(G, D) \right)$$

رابطه ۱۱-۲

برای هر تولید کننده G بهترین حالت جداساز، جداسازی است که رابطه ۱۲-۲ مینیمم کند.

$$\mathbb{E}_{x \sim p_t} [1 - D(x)] + \mathbb{E}_{x \sim p_g} [D(x)] = \int_{\mathbb{R}} (1 - D(x)) p_t(x) + D(x) p_g(x) dx$$

رابطه ۱۲-۲

به منظور مینیمم کردن انتگرال بالا نسبت به D می‌توان تابع داخل انتگرال برای هر مقدار x مینیمم کرد. سپس بهترین جداساز برای تولید کننده ایجاد خواهد شد.

$$\mathbf{1}_{(p_t(x) \geq p_g(x))}$$

رابطه ۱۳-۲

سپس به دنبال G می‌گردیم که رابطه ۱۴-۲ ماقسیم کند.

$$\int_{\mathbb{R}} (1 - D_G^*(x)) p_t(x) + D_G^*(x) p_g(x) dx = \int_{\mathbb{R}} \min(p_t(x), p_g(x)) dx$$

رابطه ۱۴-۲

برای ماقسیم کردن این انتگرال نیز کافی است تابع داخل آن را برای هر x ماقسیم کنیم. از آنجایی که شدت p_t مستقل از تولید کننده G است تنظیماتی بهتر از رابطه ۱۵-۲ وجود ندارد.

$$p_g(x) \geq p_t(x)$$

رابطه ۱۵-۲

P_g شدت احتمال است که باید به ۱ انتگرال گیری شود در نتیجه برای بهترین G رابطه ۱۶-۲ خواهیم داشت.

$$p_g(x) = p_t(x)$$

رابطه ۱۶-۲

پس نشان دادیم که در یک حالت ایده‌آل با ظرفیت‌های نامحدود برای جداساز و تولید کننده نقطه‌ی بهینه برای تنظیمات خصم‌مانه به طوری است که تولید کننده چگالی یکسانی با چگالی صحیح ایجاد کند و جداساز نتواند بهتر از صحیح بودن غیر از چند مورد محدود باشد. در نهایت G معادله را نیز ماقسیم می‌کند که به وضوح می‌توان دید G می‌خواهد احتمال اشتباه کردن جداساز را زیاد کند [9].

۶.۲ هزینه کانونی

آزمایشگاه هوش مصنوعی فیسبوک مقاله‌ای به نام هزینه کانونی^۱ منتشر کرد. این هزینه، یک تابع هزینه جدید برای بهبود دقت متوسط در یک سیستم تشخیص شیء تک مرحله است.

¹ Focal loss

هدف سیستم تشخیص شیء در بینایی ماشین این است که اشیاء خاصی را در شکل پیدا کنند. این سیستم به دو صورت تک مرحله‌ای و دو مرحله‌ای پیاده‌سازی می‌شود. تفاوت اصلی در ای دو روش زمان پردازش است. روش تک مرحله‌ای سرعت بیشتری دارد اما نسبت به دو مرحله‌ای عملکرد ضعیف‌تری دارد. تشخیص دهنده‌های دو مرحله‌ای مکانی برای تشخیص اینکه شیء کجاست پیشنهاد می‌دهد سپس به دنیال اشیاء در جعبه‌های کاندید می‌گردد. این به معنای یافتن حدود ۱۰۰۰ - ۲۰۰۰ جعبه قبل از یافتن خود اشیاء است.

در یک ماشین خودکار تصویری که توسط سنسورها دیده می‌شود تصویری به صورت شکل ۳۳-۲ است.



شکل ۳۳-۲

برای تشخیص اینکه در اطراف چه خبر است باید از اشیاء در جلوی صفحه (اشیاء در فاصله‌ی چند فیتی) و پشت صفحه (مانند ساختمان‌های بلند) آگاه بود. در حقیقت سنسورها تلاش دارند پیش‌بینی کنند که آیتم‌هایی که مشاهده می‌کنند عضو جلوی تصویر یا پشت تصویر هستند تا بتوانند متوجه شوند به آیتم مذکور برخورد می‌کنند یا خیر.

در نتیجه یک تابع هزینه جدید که تمرکز شبکه‌ی عصبی را روی مثال‌هایی که کلاسشان غلط پیش‌بینی شده است، می‌برد معرفی می‌کنیم. به جای تلاش برای کاهش داده‌های پرت یا پیش‌بینی‌های که پیش‌بینی مدل در آن‌ها از واقعیت بسیر دور است، هزینه‌ی کانونی وزن یا تاثیر مقادیری که درست پیش‌بینی شده‌اند را کاهش می‌دهد. تابع هزینه فقط یک راه ریاضی برای نشان دادن این است که بگوییم یک حدس چقدر از مقدار حقیقی اش فاصله دارد. به صورت معمول از هزینه‌ی تقاطع-آنتروپی استفاده می‌شود. هزینه‌ی کانونی وزن وابسته به احتمال به هزینه‌ی تقاطع-آنتروپی اضافه می‌کند [10].

این روش برای بهبود شبکه‌های عصبی روش مناسبی است. یکی از مشکلاتی که هوش مصنوعی دارد این است که مدل‌های بد (مدل‌هایی که پیش‌بینی نادرست دارند) درباره‌ی پیش‌بینی‌ها یشان اطمینان دارند در حالی که مدل‌های خوب اینگونه نیستند.

$$\text{CE}(p, y) = \begin{cases} -\log(p) & \text{if } y = 1 \\ -\log(1 - p) & \text{otherwise.} \end{cases}$$

رابطه ۱۷-۲

$$\text{FL}(p_t) = -\alpha_t(1 - p_t)^\gamma \log(p_t).$$

رابطه ۱۸-۲

هزینه‌ی کانونی با این هدف است که زمانی که داده‌های زیادی در یک کلاس وجود دارد و داده‌های پراکنده در کلاس دیگر، پیش‌بینی را راحت‌تر کند. هزینه‌ی کانونی زمانی مفید است که نسبت [1:1000] یا بیشتر برای کلاس‌های غیرمتداول دارد [11].

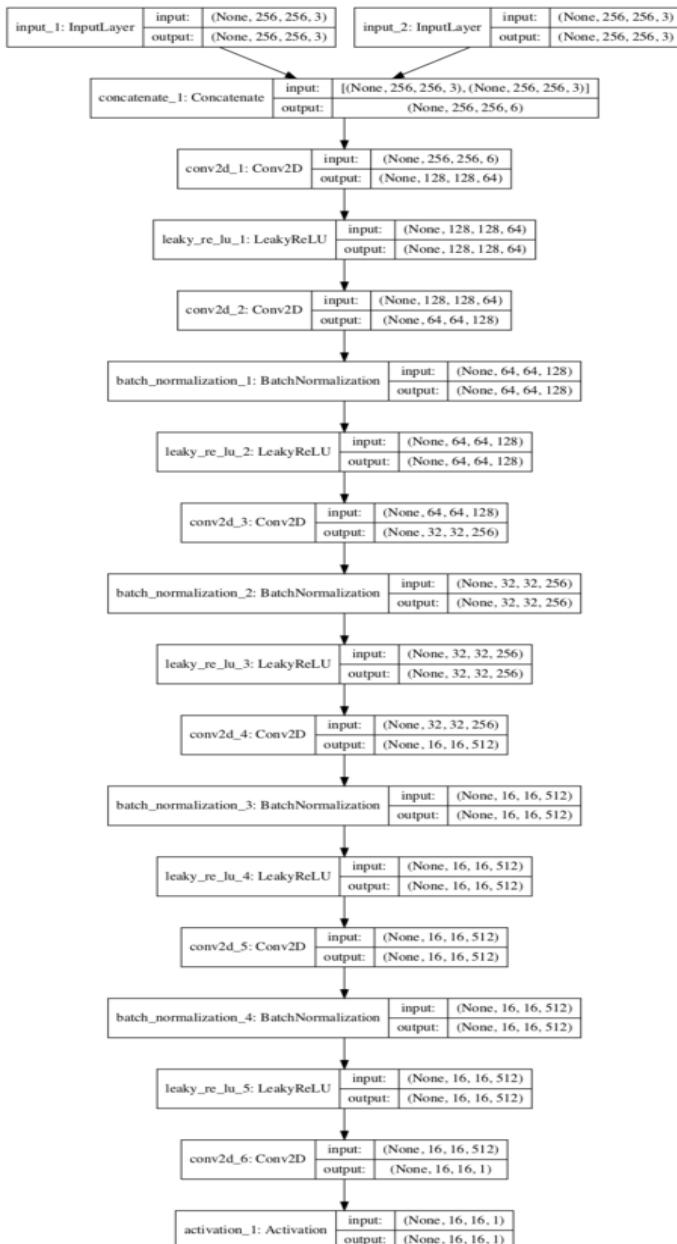
اگر همچین دسته‌بندی را به یک تشخیص دهنده‌ی شیء تک مرحله ورودی دهیم اغلب شبکه با هزاران شیء داخل پس‌زمینه اشباع می‌شود و برای پیش‌بینی اشیاء پیش‌زمینه چار مشکل می‌شود. در زمان آموزش، تشخیص دهنده‌ی شیء هزاران شیء پس‌زمینه رانگاه می‌کند و آن‌ها را به عنوان پس‌زمینه درست تشخیص می‌دهد اما اشیاء پیش‌زمینه را غلط تشخیص می‌دهد.

این دقیقاً مشکلی است که هزینه‌ی کانونی حل می‌کند. هزینه‌ی کانونی به اشیائی که به سختی دسته‌بندی می‌شوند وزن بیشتری می‌دهد و تاثیر آن‌ها را روی پیش‌بینی‌های ساده‌ی درست کاهش می‌دهد. از نظر ریاضی یک فاکتور مقیاسی به تابع هزینه‌ی تقاطع-آنتروپی اضافه می‌کند. این عدد در طول زمانی که اطمینان در پیش‌بینی بالا می‌رود کاهش می‌یابد (به سمت ۰ میل می‌کند).

به طور کلی اگر مدلی را روی داده‌ی عکس آموزش می‌دهیم که کلاس‌هایی با عدم تعادل بالا دارد می‌توان از هزینه‌ی کانونی در کنار هزینه‌ی تقاطع-آنتروپی استفاده کرد تا مدل با مثال‌های منفی سخت^۱ آموزش بیند به جای اینکه یک کلاس به شدت بررسی شود و درستی دلخواه زیاد به دست آید.

^۱ Hard negative examples

۷.۲ تکیک کننده (PatchGAN) Markovian



شکل ۷.۲

مشخص شده است که هزینه‌های L1 و L2 عکس‌های تار در مسائل تولید عکس ایجاد می‌کنند. اگرچه این هزینه‌ها نمی‌توانند سختی‌های فرکانس بالای عکس را نگه دارند در بسیاری از موارد فرکانس‌های کم را نگه می‌دارند. در بسیاری از مسائل لازم

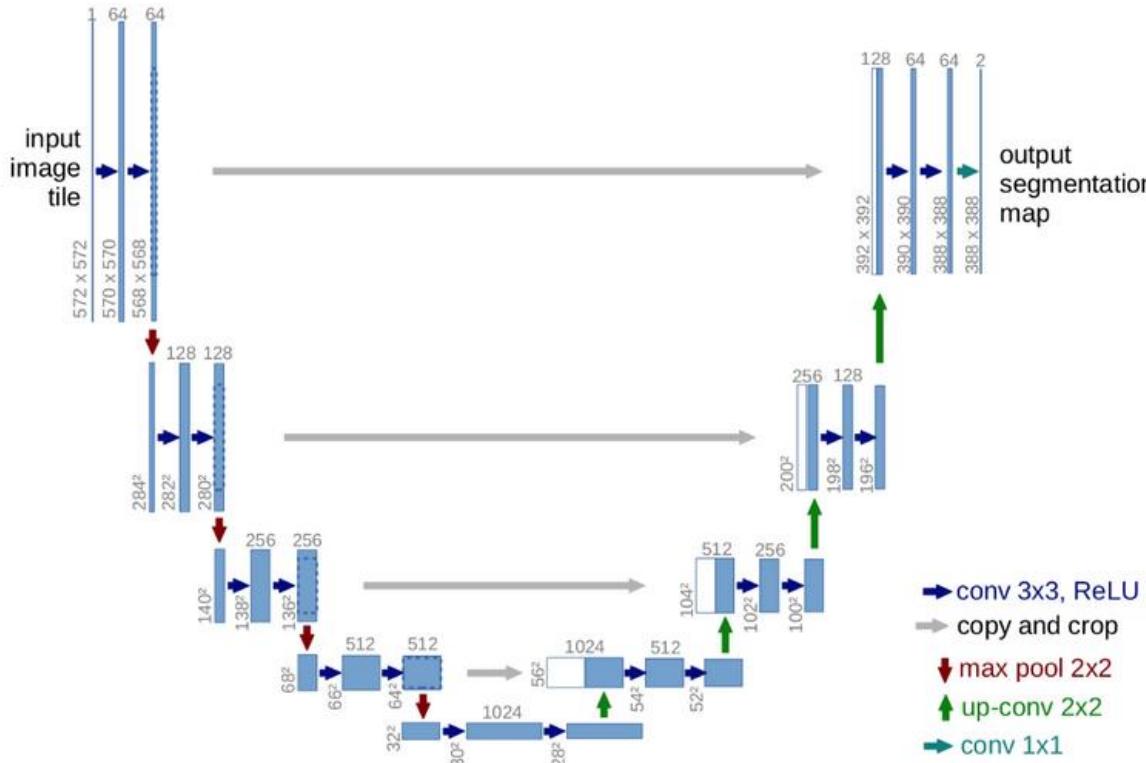
نیست که یک چهارچوب کاملاً جدید ایجاد شود تا فرکانس‌های کوچک به درستی حفظ شوند. هزینه‌ی L1 برای اینکار کافی است [12].

این مسئله باعث می‌شود تا جداکننده‌های GAN فقط روی مدل کردن ساختار فرکانس بالا تمرکز کنند و برای درستی فرکانس‌های کوچک روی L1 حساب کنند. برای مدل کردن فرکانس‌های بالا مناسب است تا روی ساختارها در تکه‌های عکس محلی تمرکز شود. در نتیجه معماری جدیدی برای جداکننده‌ی GAN طراحی شده است که فقط ساختار را در مقیاس تکه‌ها جریمه می‌کند. این جداکننده سعی می‌کند تا هر تکه‌ی $N \times N$ را به دسته‌ی واقعی یا مصنوعی تقسیم بندی کند. سپس این جداکننده به صورت کانولوشنی در سراسر تصویر اجرا می‌شود و تمامی پاسخ‌ها به منظور فراهم آوردن D مناسب را متوسط‌گیری می‌کند.

اندازه‌ی N می‌تواند بسیار کوچک‌تر یا به اندازه‌ی خود عکس باشد و هنوز تصاویر با کیفیت بالا ایجاد کند. این یک مزیت است چراکه PatchGAN با سایز کم پارامترهای کمتری دارد، سریع‌تر اجرا می‌شود و بر روی عکس‌های بسیار بزرگ نیز می‌تواند اجرا شود. این جداکننده می‌تواند به چشم هزینه‌ی مدل/بافت دیده شود.

Unet ۸.۲

[13] قابلیت این را دارد که از طریق کلاس‌بندی بر روی هر پیکسل مرزها را محلی و تفکیک کند در نتیجه در یک مسئله‌ی کلاس‌بندی ورودی و خروجی سایز یکسانی داشته باشند.



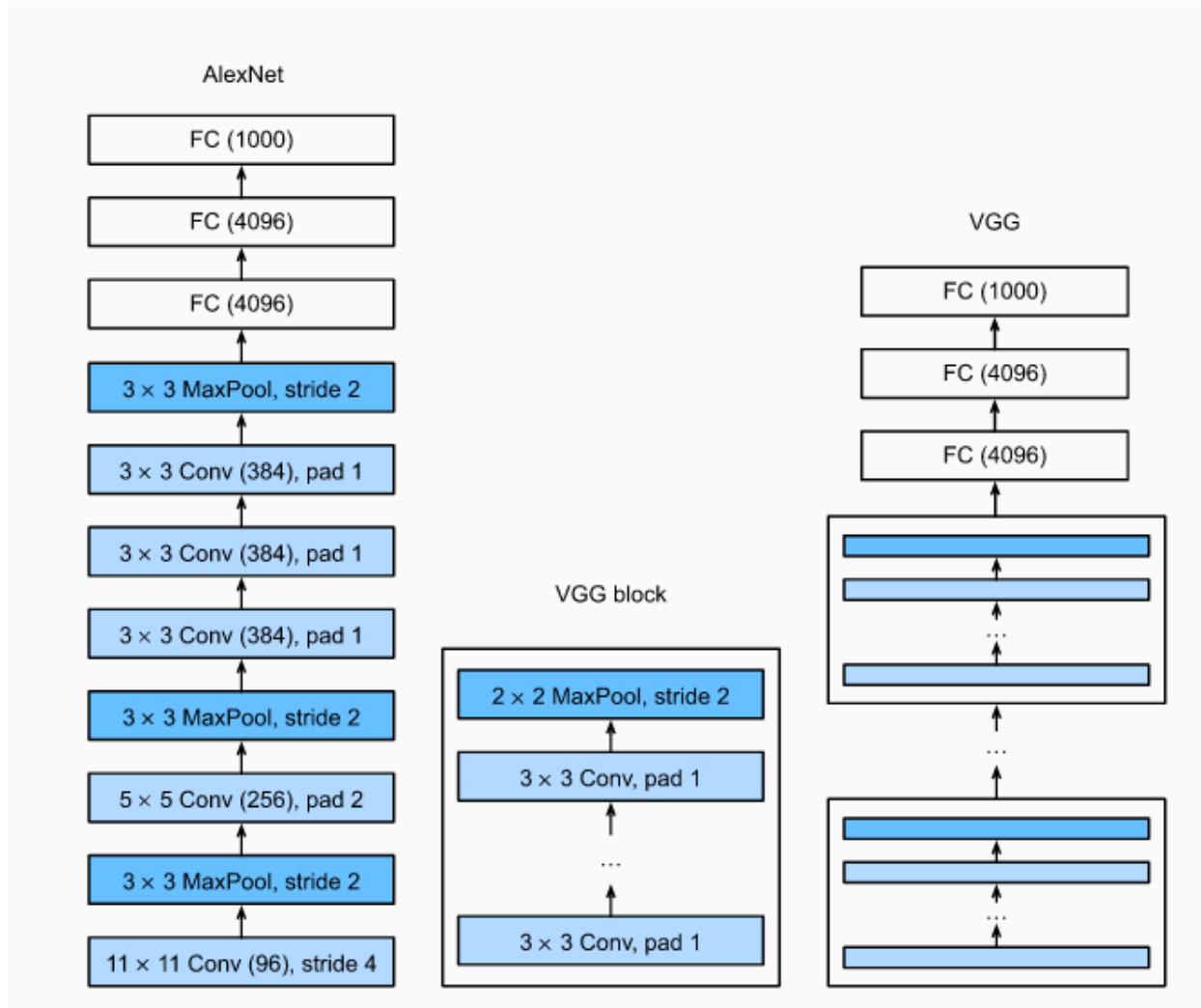
شکل ۲-۹

در شکل ۲-۹ معماری Unet را مشاهده می‌کنیم. این معماری از دو بخش عمدۀ تشکیل شده است:

۱. قسمت چپ که مسیر فرادردی گفته می‌شود که از پروسه‌ی کلی کانولوشن تشکیل شده است.
۲. قسمت راست که مسیر گسترش نامیده می‌شود که از لایه‌ی ترانهاده شده دو بعدی تشکیل شده است. هدف این بخش این است که خروجی با اندازه‌ی اولیه‌ی عکس ایجاد شود.

۹.۲ شبکه‌ی VGG

بلاک پایه‌ی سازنده‌ی CNN ها یک لایه‌ی کانولوشن، یک تابع غیر خطی مانند ReLU و یک لایه‌ی pooling است. یک بلاک VGG [14] از دنباله‌ای از لایه‌های کانولوشن که دنبالشان یک لایه‌ی ماکس پولینگ به منظور نمونه‌گیری قرار دارد.



شکل ۳۶-۲

اگر شبکه‌ی VGG ۵ بلاک کانولوشن داشته باشد که در دو تای اول یک لایه‌ی کانولوشن و در سه تای بعدی هر کدام دو لایه کانولوشن وجود داشته باشد، از آن جایی که این شبکه ۸ لایه‌ی کانولوشن و ۳ لایه‌ی کاملاً متصل دارد معمولاً ۱۱ نامیده می‌شود.

۱۰.۲ هزینه‌ی VGG

هزینه‌ی VGG تلاش می‌کند تا به شباهت ادراکی نزدیک‌تر شود. این هزینه بر اساس لایه‌های فعال‌ساز ReLU شبکه‌ی VGG از پیش آموزش دیده است. با $\varphi_{i,j}$ نقشه‌ی ویژگی درست شده توسط φ امین کانولوشن (بعد از فعال‌ساز) و قبل از VGG نامین لایه‌ی جمع‌شدن حداکثری^۱ درون شبکه‌ی VGG19 که فرض کرده‌ایم داریم مشخص می‌شود. سپس هزینه‌ی VGG بر اساس فاصله‌ی اقلیدسی بین نمایش ویژگی عکس دوباره ساخته شده‌ی $G_{\theta_G}(I_{LR})$ و عکس مرجع I_{HR} تعریف می‌شود.

$$l_{SRVGG/i,j} = \frac{1}{W_i, j H_i, j W_i, j} \sum x = \frac{1}{H_i, j} \sum y = \frac{1}{H_i, j} (\varphi_{i,j}(I_{HR})_{x,y} - \varphi_{i,j}(G_{\theta_G}(I_{LR}))_{x,y})^2$$

در رابطه‌ی ۱۹-۲ ابعاد نقشه‌های ویژگی درون شبکه‌ی VGG را مشخص می‌کنند.

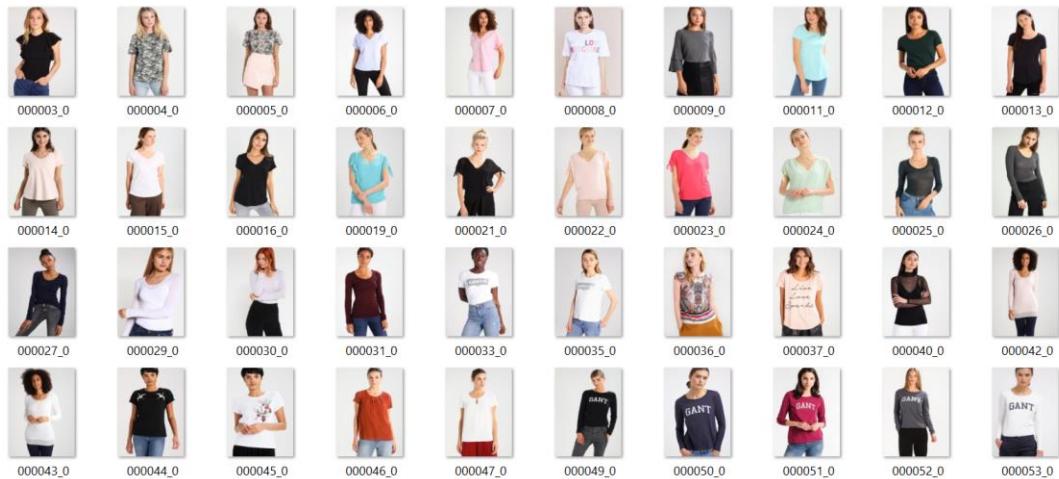
^۱ Max pooling

۳ فصل سوم - روش‌های موجود

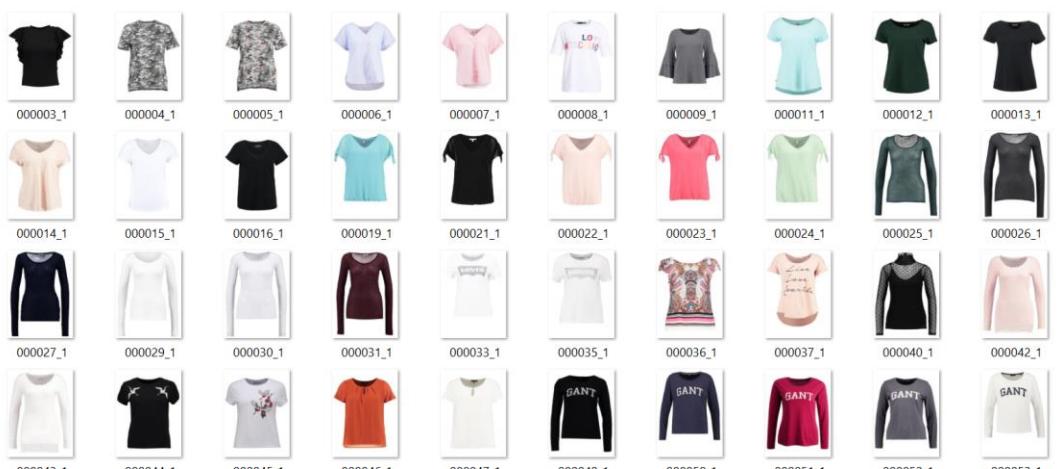
در این بخش تمامی روش‌های موجود در زمینه‌ی پوشیدن لباس به صورت مجازی بررسی می‌شود. ابتدا داده‌هایی که در این زمینه موجود است را به طوری کلی مشاهده‌می‌کنیم و سپس روش‌های موجود را به صورت خلاصه بررسی خواهیم کرد. لازم به ذکر است که از آنجایی که این کار روش نوینی است، روش‌های چندان زیادی در این زمینه موجود نبوده است و تمامی روش‌های ذکر شده مربوط به مقالات و کارهای ۲-۳ سال اخیر هستند.

۱.۳ داده‌ها

تصویری کلی از داده‌هایی که در اختیار قرار دارد در زیر مشاهده می‌شود:



شکل ۳-۱- داده هایی که از برای آموزش مدل داریم



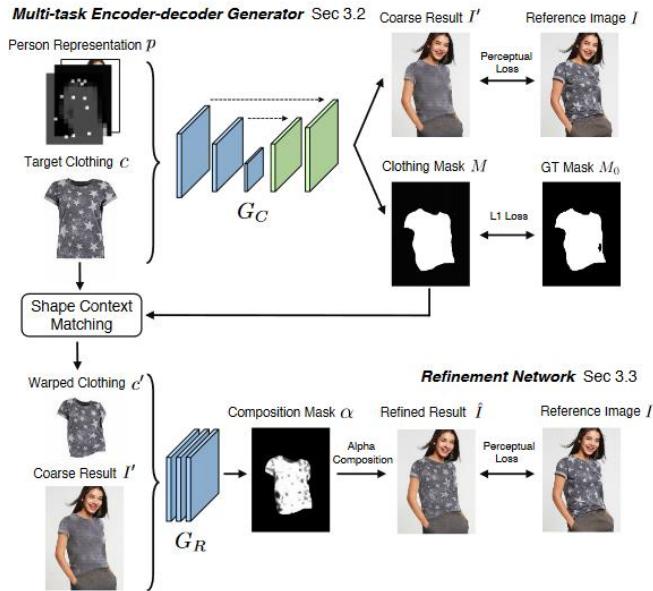
شکل ۲-۳- لباس هایی که در دیتابست لباس های هدف قرار دارد



شکل ۳-۳- نتیجه های اجرای الگوریتم به شخص نگاه کنید بر روی دیتابست

VTON ۲.۳

به صورت کلی روشی که در این سیستم استفاده می‌شود به این صورت است که ابتدا یک شمایی از لباس برای نمایش فرد ایجاد می‌شود. سپس عکس مرجع با یک معماری رمزگذار-رمزگشا که شرط نمایش فرد و عکس لباس هدف را دارد، سنتر می‌شود. تصویر خروجی از این معماری برای بهبود الگوهای ظاهری وارد یک شبکه‌ی اصلاح می‌شود [15]. تصویر کلی معماری به صورت زیر است:



شکل ۳-۴

۳.۳ نمایش فرد در VTON

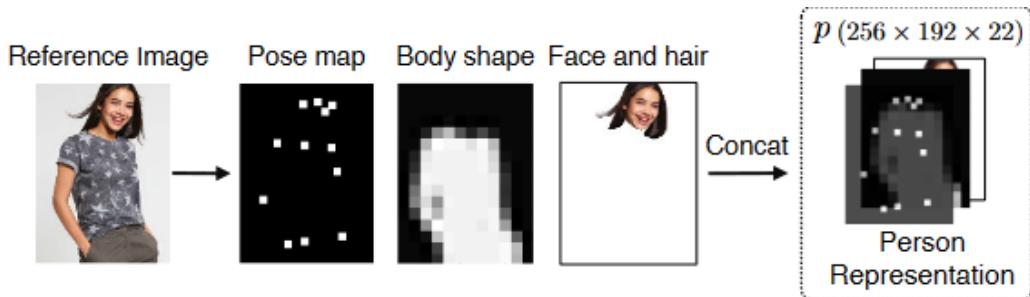
چالش اصلی سنتر پوشیدن مجازی لباس این است که عکس لباس هدف به گونه‌ای تغییر کند تا کاملاً متناسب با ژست شخص شود. در نتیجه یک شمای پوششی مناسب برای نمایش فرد در اینجا بررسی می‌شود که شامل ژست فرد، قسمت‌های مختلف بدن، صورت و مو است.

نقشه‌ی ژست: برای پیدا کردن ژست فرد از کتابخانه openpose استفاده شده است که ۱۸ نقطه‌ی کلیدی خروجی می‌دهد. در همسایگی هر نقطه تا شعاع ۱۱ در ۱۱ یک مستطیل سفید ایجاد شده است.

نمایش بدن انسان: با استفاده از روش توضیح داده شده در فصل دو به نام به شخص نگاه کنید قسمت‌های مختلف بدن در هر عکس همراه با حالت کلی بدن استخراج شده است. سپس این نقشه‌ی بخش‌بندی به یک عکس باینری یک کاناله تبدیل شده است که یک‌ها نشان‌دهنده بدن انسان هستند. سپس رزولوشن این عکس باینری کاهش پیدا کرده است.

بخش‌بندی صورت و مو: برای اینکه ویژگی‌های هر فرد حفظ شود، خصوصیات فیزیکی مانند صورت، رنگ پوست، مدل مو و غیره توسط روش به شخص نگاه کنید، بخش‌بندی و جدا شده‌اند.

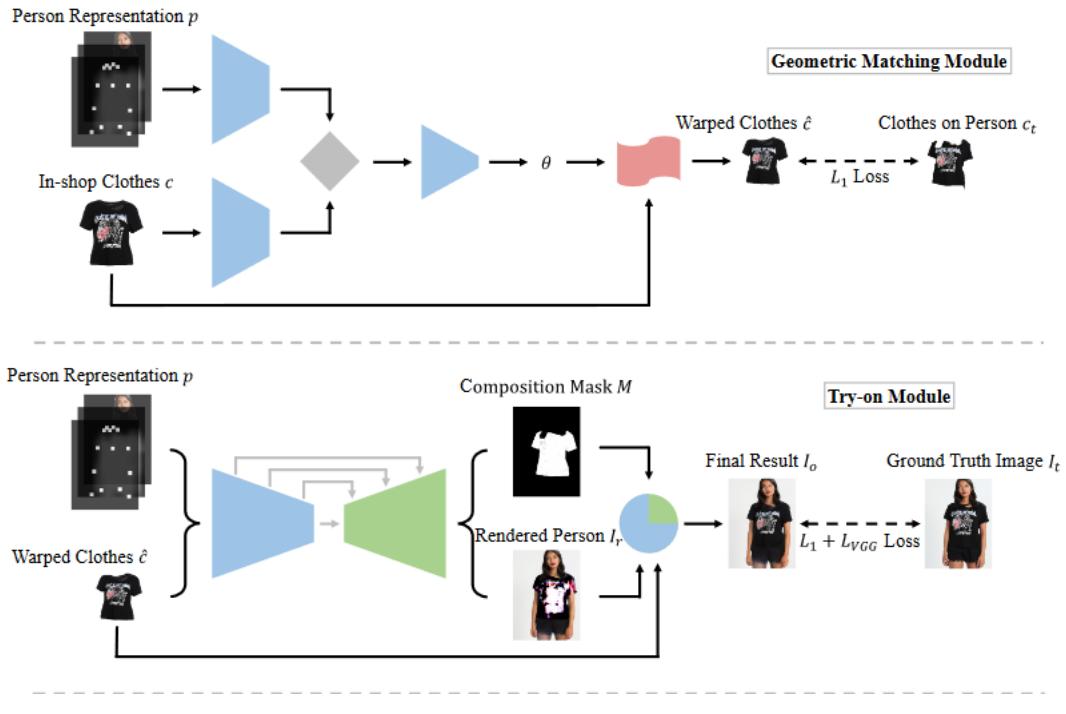
در نهایت رزولوشن هر سه بخش یکسان شده و به یکدیگر متصل می‌شوند.



شکل ۵-۳- نمایش فرد در مدل VTON

۴.۳ CP-VTON

این روش یک مازول هماهنگی هندسی معرفی می‌کند که به صورت مستقیم لباس C را متناسب با اطلاعات نمایش فرد p تغییر می‌دهد و عکس یک لباس چرخیده شده‌ی C را ایجاد می‌کند. این مازول یک شبکه‌ی عصبی است که به صورت مستقیم با هزینه‌ی L1 آموخته داده می‌شود. سپس خط لوله‌ی این روش با یک مازول سنتز try-on کامل می‌شود. در این مازول لباس چرخیده شده‌ی \hat{C} و تصویر فرد I_r به هم متصل می‌شوند [16]. تصویر کلی این معماری در شکل است:



شکل ۳-۶- ساختار کلی معماری *cp-vton*

۵. نمایش فرد در CP-VTON

مراحل نمایش فرد در این روش دقیقاً مانند روش توضیح داده شده در روش VTON است. این نمایش در هر دو مژول استفاده می‌شود.

۶. مژول هماهنگی هندسی

این مژول لباس C را به لباس چرخیده شده‌ی متناسب با ژست فرد داخل عکس تبدیل می‌کند. از ۴ بخش کلی تشکیل شده است که در قسمت هماهنگی هندسی فصل دو مفصل بررسی شد.

۷. مژول پوشیدن لباس

اکنون که لباس چرخیده شده متناسب با ژست ایجاد شده است هدف این مژول این است که لباس حاصل را با عکس فرد به هم مثل کند و نتیجه نهایی را خروجی دهد. یک روش ساده این است که به صورت مستقیم \hat{C} را روی عکس شخص پیست^۱ کنیم. نکته‌ی مثبتی که این کار دارد این است که تمامی ویژگی‌های لباس \hat{C} حفظ خواهد شد اما مشکلی که دارد این است که در نقاط مرزی لباس ظاهر غیر طبیعی ایجاد خواهد شد به علاوه سبب انسداد برخی از قسمت‌های بدن خواهد شد. راه دیگری که بسیار در تولید عکس بر اساس شرطی استفاده می‌شود، ترجمه‌ی ورودی به خروجی از طریق یک عبور یک طرفه از شبکه‌ای

¹ Paste

رمزگذار-رمزگشا مانند Unet است. این روش برای تولید عکس‌های بدون مرز مفید است. البته باید در نظر داشت امکان ندارد به بهترین صورت، لباس روی بدن شخص قرار گیرد. نبود تغییر حالت مکانی مستقیم می‌تواند باعث شود که خروجی‌های شبکه‌ی Unet ناواضح باشند.

ماژول پوشاندن لباس سعی می‌کند خوبی‌های هر دو روش را با هم ترکیب کند. با داشتن نمایش فرد p و لباس چرخیده شده \hat{C} که به یکدیگر متصل شده‌اند Unet همزمان عکس شخص I_r ایجاد می‌کند و ماسک ترکیبی M را پیش‌بینی می‌کند. سپس عکس شخص I_r و لباس چرخیده \hat{C} از طریق ماسک M با یکدیگر آمیخته می‌شوند و تصویر سنتز شده و نهایی Io را ایجاد می‌کنند.

$$I_o = M \odot \hat{c} + (1 - M) \odot I_r$$

رابطه ۱-۳

در فاز آموزش با داشتن نمونه‌ی (p, c, It) هدف این ماژول این است که اختلاف بین تصویر Io و تصویر حقیقی It را مینیمم کنیم. در این روش از روش معمول در شبکه‌های مولد شرطی استفاده شده است که برای تابع نهایی هزینه از ترکیبی از هزینه‌ی $L1$ و هزینه‌ی ادراکی VGG استفاده می‌شود. که هزینه‌ی ادراکی VGG به صورت زیر تعریف می‌شود:

$$\mathcal{L}_{VGG}(I_o, I_t) = \sum_{i=1}^5 \lambda_i \|\phi_i(I_o) - \phi_i(I_t)\|_1$$

رابطه ۲-۳

که $\phi_i(I)$ نقشه‌ی ویژگی عکس I را در i امین لایه در شبکه‌ی ادراکی Φ ، که یک شبکه‌ی 19-VGG و پیش آموزش دیده شده بر روی دیتابست ImageNet است را مشخص می‌کند. لایه‌ی $i \geq 1$ در حقیقت Conv1-2، Conv2-2، Conv3-2، Conv4-2، Conv5-2 را به ترتیب مشخص می‌کند. در راستای هدف نگهداری داشتن ویژگی‌ها ماسک ترکیبی M به گونه‌ای با این شده است تا لباس‌های چرخیده شده را تا حد امکان انتخاب کند. اینکار با اعمال نظم دهی $|M|_1 - |I|_1$ ، بر روی M صورت می‌گیرد. تابع هزینه‌ی کلی برای این ماژول به صورت زیر محاسبه می‌گردد:

$$\mathcal{L}_{TOM} = \lambda_{L1} \|I_o - I_t\|_1 + \lambda_{vgg} \mathcal{L}_{VGG}(\hat{I}, I) + \lambda_{mask} \|1 - M\|_1.$$

رابطه ۳-۳

۴ فصل چهارم – روش استفاده شده

در این قسمت به بررسی روش جدید که نتایج بهتری از دو مقاله بررسی شده در فصل ۳ داشته است می‌پردازیم. ابتدا به تفضیل روش استفاده شده را بررسی می‌کنیم سپس نتایج بخشی از مقاله که پیاده‌سازی شده است را در فصل بعد همراه با جزییات پیاده‌سازی بیان می‌کنیم و در آخر اینکه پژوهه به کدام سمت حرکت خواهد کرد را مورد بحث قرار می‌دهیم.

۱.۴ داده‌های مسئله

داده‌هایی که به آنها دسترسی می‌شده‌اند پیدا کرد داده‌های مقاله‌ی *cp-vton* به صورت تغییر سایز یافته است. این داده‌ها اندازه‌ی 192×256 دارند. به طور کلی ۱۹ هزار جفت عکس بالاتنه‌ی مختلف است که تعدادی از آنها به خاطر ناکامل بودن حذف شده‌اند. در کل ۱۴۰۰۶ جفت داده برای یادگیری و ۲۰۰۲ جفت داده برای تست در نظر گرفته شده است. در مجموعه تست لباس هدفی که قرار است پوشانده شود و لباس پوشیده شده به مدل داخل عکس یکی است. اما در داده‌های تست این موارد متفاوتند.

۲.۴ روش به کار رفته

با داشتن عکس لباس C و عکس مرجع I که شخص با لباس در آن قرار دارد که در آن شخص لباس متفاوتی استفاده کرده است. هدف نهایی سیستم این است که عکس جدید \hat{I} از شخص تولید شود که لباس C را پوشیده است به طوری که ژست و حالت شخص حفظ شده باشد. به صورت ایده‌آل داده‌های یادگیری برای سیستم باید به صورت سه تایی (\hat{I}, C, I) باشد. به

هر حال چنین ساختار داده‌ای غیر معمول است لذا داده‌های یادگیری را به صورت دوتایی (C, I) در نظر می‌گیریم. برای این سیستم از تصویر I نمایش فرد را به دست می‌آوریم و مدل بر اساس همان نمایش آموخته می‌شود [17].

این سیستم سه مژول مختلف دارد که در این پروژه مژول اول همراه با بخشی از مژول دوم پیاده‌سازی شده است.

۱. مژول تغییر فرم لباس: $\mu_1 = \hat{C}$ که لباس C را متناسب با نمایش p_1 به لباس تابخورده \hat{C} تبدیل می‌کند که متناسب با ژست فرد داخل عکس است.

۲. مژول تولید نقشه‌ی قطعه‌بندی^۱: $\hat{M}_S = M_2(p_2, \hat{C})$ که یک نقشه‌ی قطعه‌بندی جدید از بخش‌های بدن و قسمتی از بدن که توسط لباس هدف پوشیده شده است را تولید می‌کند. ورودی این مژول p_2 و \hat{C} است.

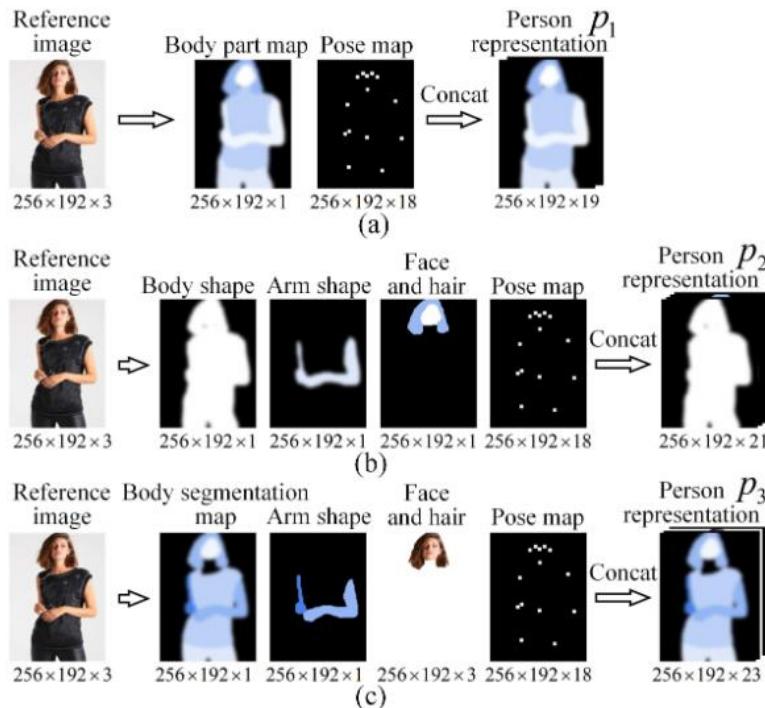
۳. مژول سنتر پوشیدن^۲: $\hat{I} = M_3(p_3, \hat{C})$ که عکس نهایی را سنتز می‌کند.

قسمت کلیدی این روش، نمایش شخص به سه صورت مختلف است که دو صورت اول p_1 و p_2 به صورت مستقیم از I گرفته شده‌اند در حالی که p_3 بر اساس I و \hat{C} پیش‌بینی می‌شود. p_3 اطلاعاتی شامل قطعه‌بندی قسمت‌های مختلف بدن و لباس هدف دارد. نمایش p_3 برای بازیابی جزئیات لباس و قسمت‌های بدن در سنتز نهایی عکس \hat{I} نقشی کلیدی دارد.

۴. نمایش فرد در روش پیاده‌سازی شده برای نگه‌داری ویژگی‌های لباس‌ها و بدن انسان روشنی برای استخراج سه مرحله از نمایش فرد ایجاد شده است.

¹ Segmentation map

² Try-on synthesis



شکل ۱-۴- سه مدل نمایش فرد که برای سه مأذول مورد نیاز برای سیستم است در این شکل نمایش داده شده است.

نمایش p_1 فرد: p_1 از دو بخش نقشه بدن به صورت یک کاناله و نقشه‌ی ژست به صورت ۱۸ کاناله تشکیل شده است. نقشه‌ی بدن که به صورت یک کاناله استفاده می‌شود شامل ۶ قسمت است که از روش "به شخص نگاه کنید" توضیح داده شده در فصل دو برای استخراج این ۶ قسمت استفاده شده است. البته در روش اصلی از روش دیگر برای این قسمت استفاده شده است. همچنین برای نقشه‌ی ژست از کتابخانه openpose توضیح داده شده در فصل دو استفاده شده است. ۱۸ نقطه‌ی کلیدی خروجی این کتابخانه را هر کدام را به صورت یک نقشه‌ی جدا در می‌آوریم. به این صورت که مکان هر نقطه را در عکس به صورت یک مستطیل ۱۱ در ۱۱ سفید در یک صفحه مشکی مشخص می‌کنیم. در نتیجه ۱۸ صفحه به این شکل خواهیم داشت.



۲- شکل - نقشه‌ی جداگانه مستطیل ۱۱ در ۱۱ که یکی از نقطه‌های کلیدی را مشخص می‌کند.



شکل ۳-۶- نقشه‌ی بخش‌بندی یک کاناله

نمایش p_2 فر5: از ۴ بخش نقشه‌ی شکل بدن به صورت یک کاناله، نقشه‌ی بازو یک کاناله، نقشه‌ی صورت و موی یک کاناله و نقشه‌ی ژست ۱۸ کاناله تشکیل شده است. نقشه‌ی ژست دقیقاً مانند p_1 است در حالی که بر جسب^۱‌های جدید معنایی از روش به شخص نگاه کنید استخراج می‌شوند.

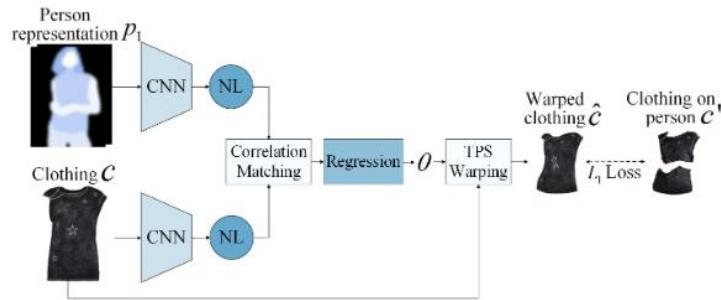
نمایش p_3 فر5: از ۴ بخش نقشه‌ی قطعه‌بندی بدن به صورت یک کاناله، نقشه‌ی بازو یک کاناله، نقشه‌ی صورت و موی سه کاناله و نقشه‌ی ژست ۱۸ کاناله تشکیل شده است. نقشه‌ی قطعه‌بندی در این بخش شامل لیل برای ۱۳ قسمت مختلف از فرد است در حالی که لباس هدف را پوشیده است و شامل قسمت بالایی و پایین لباس، کلاه، مو، دست‌کش، شال، صورت، بازوی چپ و راست، پای چپ و راست، کفش‌ها که توسط مازول تولید نقشه‌ی قطعه‌بندی تولید شده است می‌باشد.

هر کدام از این نمایش‌ها هدف خاصی را دنبال می‌کنند. p_1 برای ایجاد لباس تابیده شده استفاده می‌گردد، p_1 برای پیش‌بینی نقشه‌ی قطعه‌بندی و p_1 طرحی برای تولید عکس نهایی هدف ایجاد می‌کند.

۴.۴ مازول تغییر حالت لباس

این مازول لباس هدف C به لباس تغییر شکل یافته‌ی \hat{C} تبدیل می‌کند. ورودی‌های این مازول p_1 و C هستند. این مازول از یک مازول هماهنگ کننده و تطبیق‌دهنده‌ی هندسی استفاده می‌کند تا پارامترهای اسپلاین صفحه‌ی نازک را برای تغییر حالت C محاسبه کند. تفاوتی که با روش توضیح داده شده در فصل دو دارد این است که از بلاک‌های غیر محلی نیز در این مازول استفاده شده است تا یادگیری ویژگی‌ها و تطابق پیشرفت پیدا کند.

¹ label



شکل ۴-۴- معماری مژول تطابق هندسی یا مژول ۱ را مشاهده می کنیم.

این مژول با شبکه های عصبی کانولوشن شروع می شود و پس از آن بلاک های غیر محلی قرار می گیرد. این لایه ها ویژگی های p_1 و C را استخراج می کند که سپس با یکدیگر ترکیب می شوند تا یک نقشه هی شباهت ایجاد شود. این نقشه سپس وارد یک لایه رگرسیون می شود تا پارامتر های TPS برای تبدیل C به \hat{C} پیشینی شود. سپس کل این مژول با مینیمم کردن هزینه بین لباس تاب خورده و لباس حقیقت زمینی^۱ C' که از عکس اصلی قطعه بندی شده است آموخت داده می شود.

$$L(\hat{C}, C') = \|\hat{C} - C'\|_1$$

۴.۵ مژول تولید نقشه هی قطعه بندی M_2

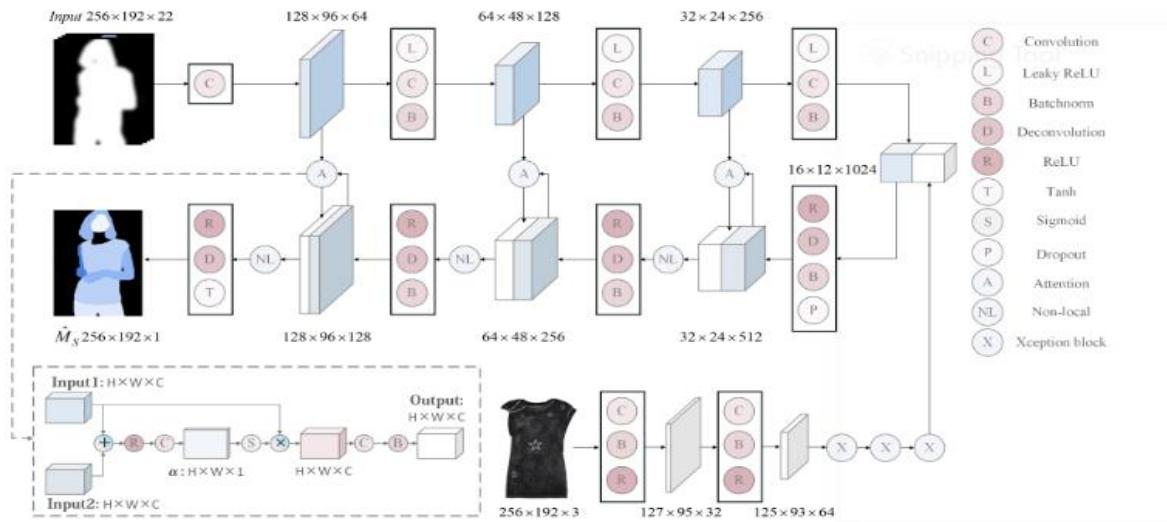
هدف این مژول این است که یک نقشه هی بخش بندی معنایی \hat{M}_S از شخص در حالی که لباس هدف را پوشیده است ایجاد کند. ورودی های این مژول p_2 و \hat{C} هستند. معماری M_2 از یک رمز گذار^۲ تشکیل شده است که اطلاعات را از p_2 و \hat{C} رمز گذاری می کند سپس رمز گشایی وجود دارد که بخش بندی معنایی \hat{M}_S را ایجاد می کند. این معماری یک زیر شبکه هی توجه^۳ دارد که مانند دروازه های عمل می کند تا داده های جریان رمز گذاری را از رمز گذار به رمز گشای بدهد. برای اینکه وابستگی های دامنه طولانی حفظ و دریافت شوند از عملیات های غیر محلی استفاده شده است.

با وجود اینکه \hat{C} ورودی این معماری است، برای برخی از اطلاعات جنس لباس هدف ممکن است پس از تغییر گم شده باشد. برای اینکه این اطلاعات بهتر حفظ شوند شاخه ای به این ساختار اضافه شده است تا ویژگی های لباس را از خود C مستقیما استخراج کند و به ویژگی های رمز گذار متصل کند.

¹ Ground truth

² encoder

³ attention



شکل ۵-۶ - مازول دوم - مازول تولید نقشه‌ی بخش‌بندی

مازول ($\widehat{M}_s = M_2(p_2, \widehat{c})$) با استفاده از داده‌های لباس هدف و عکس مرجع که زوج (c , I) را می‌سازند آموزش داده می‌شود. نقشه‌ی بخش‌بندی معنایی تولید شده با نقشه‌ی بخش‌بندی حقیقی که با روش به شخص نگاه کنید ایجاد شده است مقایسه می‌شود. می‌توان به این مازول به عنوان مدل تولید کننده‌ی شرطی^۱ نیز نگاه کرد. هزینه‌ی نهایی برای آموزش این مازول L_{SMGM} از هزینه‌ی کانونی روی هر یک از پیکسل‌های عملیات بخش‌بندی و هزینه‌ی خصم‌مانه^۲ به منظور تمیز نقشه‌های بخش‌بندی معنایی حقیقی از مصنوعی تشکیل شده است.

$$L_{fl} = -1NN \sum i = 1C \sum k = 1(1 - \widehat{y}_{ik}) \gamma y_{ik} \log(\widehat{y}_{ik}) \quad \text{رابطه ۴-۴}$$

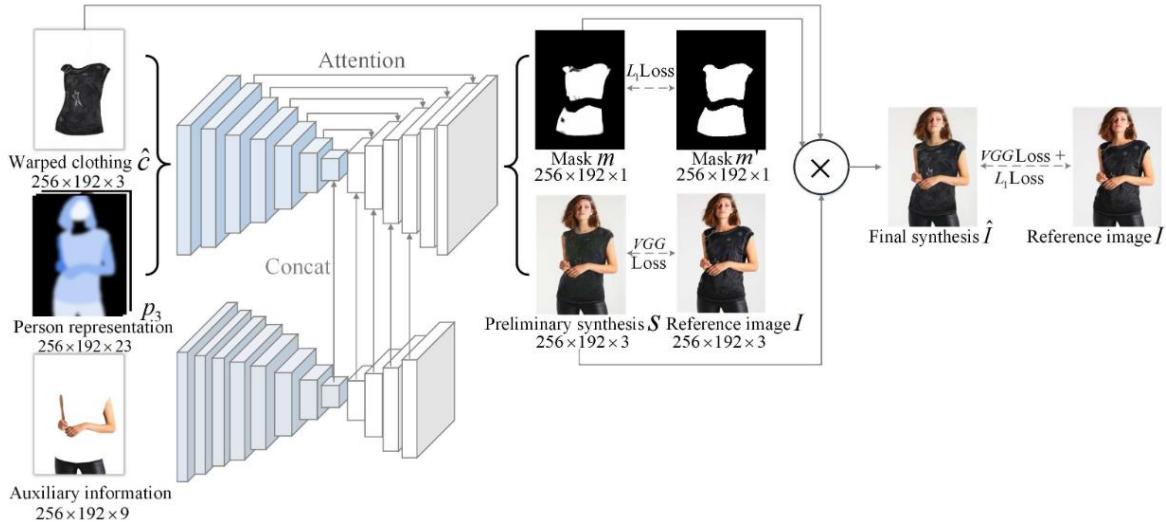
$$L_{cGAN} = E(x, y) [\log D(x, y)] + E(x, z) \left[\log \left(1 - D(x, G(x, z)) \right) \right] \quad \text{رابطه ۴-۵}$$

$$L_{SMGM} = \alpha L_{fl} + (1 - \alpha) L_{cGAN} \quad \text{رابطه ۴-۶}$$

که i و k ساخته‌های پیکسل‌ها و بخش‌بندی معنایی قسمت‌های مختلف بدن را مشخص می‌کند. y_{ik} بخش‌بندی معنایی حقیقی را مشخص می‌کند و \widehat{y}_{ik} احتمال پیش‌بینی شده را مشخص می‌کند. . رابطه ۴-۴ هزینه‌ی مدل تولید کننده‌ی شرطی drop out را مشخص می‌کند که در آن X داده‌ی ورودی (ترکیب p_2 و \widehat{c}) و y بخش‌بندی معنایی حقیقی و Z نویز به شکل [12]PatchGAN است [18]. جداکننده‌ای که برای آموزش این مدل تولید کننده استفاده شده است جداکننده‌ی [12]PatchGAN است.

¹ Conditional gan

² Adversarial loss



هدف این مژول سنتز کردن عکس \hat{I} براساس خروجی‌های دو مژول قبل است. به طور کلی از سه منبع اطلاعاتی استفاده می‌شود \hat{C} از مژول ۱ M_1 ، p_3 از مژول دوم، اطلاعات کمکی از شلوار و بازوها که از عکس اصلی استخراج شده است. معماری این مژول از دو بخش کلی تشکیل شده است. شاخه‌ی بالایی از یک U-net که براساس توجه است استفاده می‌کند و ویژگی‌های p_3 و \hat{C} را استخراج می‌کند. لایه‌ی پایینی از ۷ لایه‌ی رمزگذار تشکیل شده است که براساس ایده‌ی لایه‌ی اکسپشن توضیح داده شده در فصل دو طراحی شده است. سپس چهار لایه‌ی رمزگشا برای استخراج ویژگی‌های کمکی قرار داده شده است که این ویژگی‌ها به ویژگی‌های استخراج شده از لایه‌ی بالایی متصل خواهند شد.

ماژول سنتز کننده یک ماسک m را خروجی می‌دهد که محدوده‌ی لباس را در عکس هدف مشخص می‌کند. سنتز نهایی \hat{I} با متصل کردن S و \hat{C} با هدایت m به صورت زیر مشخص می‌گردد:

$$I = m \odot \hat{C} + (1 - m) \odot S$$

رابطه ۶-۴

که \odot ضرب ماتریسی المان به المان را مشخص می‌کند.تابع هزینه در این مژول از چهار بخش تشکیل شده است.

$$L_{TSM} = \lambda 1 L(m, m') + \lambda 2 L_{VGG}(S, I) + \lambda 3 L(\hat{I}, I) + \lambda 4 L_{VGG}(\hat{I}, I)$$

رابطه ۶-۵

بخش اول یعنی $L(m, m')$ هزینه‌ی بین ماسک^۱ لباس پیش‌بینی شده و m' حقیقی است. ماسک حقیقی از نقشه‌ی بخش‌بندی \hat{C} با حذف قسمت بازو به دست می‌آید. این تابع هزینه کمک می‌کند تا جایی که می‌شود جزیات لباس حفظ گردد. (\hat{I}, \hat{I}) هزینه‌ی بین عکس سنتز شده‌ی \hat{I} و عکس حقیقی I را محاسبه می‌کند.

علاوه بر تفاوت‌های شدت پیکسل‌ها می‌توان یک هزینه‌ی ادراکی بین دو عکس نیز در نظر گرفت که توسط ویژگی‌های استخراجی از مدل VGG به دست می‌آید. (I, I) هزینه‌ی ادراکی بین سنتز ابتدایی s و I و (\hat{I}, \hat{I}) هزینه‌ی ادراکی بین \hat{I} و I را محاسبه می‌کند. این امر باعث می‌شود عکس خروجی واقع‌گرایانه‌تر شود.

$$L(m, m') = \|m - m'\|_1$$

رابطه ۷-۴

$$L(\hat{I}, I) = \|\hat{I} - I\|_1$$

رابطه ۷-۵

$$L_{VGG}(s, I) = 5 \sum_i = 1 \lambda i \|f_i(s) - f_i(I)\|_1$$

رابطه ۸-۴

$$L_{VGG}(\hat{I}, I) = 5 \sum_i = 1 \lambda i \|f_i(\hat{I}) - f_i(I)\|_1$$

رابطه ۹-۴

^۱ Mask

۵ فصل پنجم - نتایج و پیاده‌سازی

در این فصل جزئیات پیاده‌سازی مژول‌ها مطرح می‌شود. لازم به ذکر است که مژول اول به طور کامل پیاده‌ستزی شده و آموزش دیده شده و تنها بخشی از مژول دوم پیاده‌سازی گردیده است.

۱.۵ جزئیات پیاده‌سازی

تمامی ورودی‌های تمامی مژول‌ها عکس‌های اندازه 192×256 بوده است. تمامی کدها با استفاده از کتابخانه پایتورچ^۱ زده شده است. محیط پیاده‌سازی کدها در فضای گوگل کولب^۲ با استفاده از gpu شرکت Nvidia بوده است.

۲.۵ مژول تغییر فرم لباس

این مژول با 200 هزار epoch با سایز دسته 4 آموزش دیده است. بهینه‌ساز این مژول، بهینه‌ساز آدام بوده است و مقادیر β_1 و β_2 به ترتیب برابر 0.5 و 0.999 بوده است. همچنین نرخ آموزش برابر 0.001 برای 100 هزار epoch اول در نظر گرفته شده است. سپس این مقدار به صورت خطی در 100 هزار epoch بعدی کاهش می‌یابد [17].

ساختار شبکه‌های کانولوشن در این مژول یکسان است و هر دو 6 لایه کانولوشن دارند. چهار لایه اول هر کدام 2 گامه هستند سپس دو لایه 1 گامه پس از آنها قرار گرفته است. پس از این لایه‌ها 4 لایه غیر محلی قرار گرفته‌اند. تعداد فیلترها به ترتیب برابر 64 , 128 , 256 , 512 بوده است. شبکه‌ی رگرسیون برای تخمین پارامترها از دو لایه کانولوشن 2 گامه و یک لایه کانولوشن یک گامه تشکیل شده است و در نهایت یک لایه کاملاً متصل قرار گرفته است.

۳.۵ مژول تولید نقشه‌ی بخش‌بندی

در این مژول برای پارامترهای رابطه $4-5$ ، $5 = 0.5$ در نظر گرفته می‌شود. این مژول با اندازه دسته 5 با 15 مرحله آموزش داده می‌شود. مژول تولید کننده چهار لایه رمزگذار و چهار لایه رمزگشا که اندازه فیلتر آن 4 در 4 است. تعداد فیلترها به ترتیب در لایه‌ای رمزگار 64 , 128 , 256 , 512 است. برای لایه‌ای رمزگشا تعداد کانال‌ها به ترتیب 512 , 256 , 128 , 1 است. لایه‌ای غیر محلی پس از لایه‌ای اتصال اضافه می‌شوند. شبکه‌های عصبی کانولوشن برای استخراج ویژگی‌های

¹ Pytorch

² Google colab

سطح بالا از لباس تغییر شکل یافته از دو لایه کانولوشن ۳ در ۳ و سه بلاک اکسپشن تشکیل شده‌اند که تعداد فیلترها به ترتیب ۳۲، ۶۴، ۱۲۸، ۲۵۶، ۵۱۲ است.

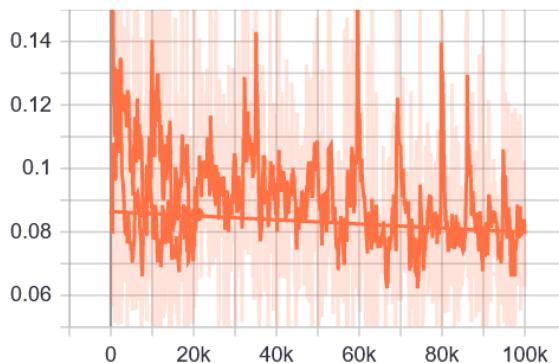
۴.۵ نتایج کمی

مقاله‌ی اصلی برای مقایسه‌ی کیفیت نهایی که شامل هر سه ماثول است تست A/B انجام داده است. به این صورت که به تعدادی از افراد نتایج خروجی سیستم ارائه شده در این گزارش و VTON داده شده است و از آن‌ها خواسته شده است که بهترین نتیجه را انتخاب کنند. همچنین این عملیات با cp-VTON نیز اجرا شده است که نتایج نهایی در جدول زیر آمده است.

Method	Human	Method	Human
VITON	32.13%	CP-VTON	22.62%
VTNFP	67.87%	VTNFP	77.38%

شکل ۱-۵ - مقایسه سه روش موجود مطرح شده در این پژوهه

metric



شکل ۲-۵ - میزان کاهش هزینه را در آموزش شبکه‌ی عصبی ماثول اول تا ۱۰۰ هزار مرحله مشاهده می‌کنیم. مشاهده می‌شود که این هزینه با افزایش مرحله کاهش پیدا می‌کند.

۵.۵ نتایج کیفی

در این قسمت تعدادی از نتایج کیفی پیاده‌سازی ماثول اول و ماثول دوم را مشاهده خواهیم کرد. تاثیر لایه‌های غیر محلی و ویژگی‌های غیر تغییر یافته در تولید نقشه‌ی بخش‌بندی در تصاویری که در ادامه می‌آیند مشخص شده است.



شکل ۳-۵- بخشی از نتایج خروجی شبکه‌ی عصبی مژول اول را در EPOCH و مرحله‌ی ۲۰ هزار و ۸۰۰ نشان می‌دهد. شکل اول سمت چپ لباسی است که باید تغییر پیدا کند، شکل سمت راست اول ژستی است که لباس باید به خود بگیرد و شکل وسط خروجی مدل است.



شکل ۴-۵- بخشی از نتایج خروجی شبکه‌ی عصبی مژول اول را در EPOCH و مرحله‌ی ۲۰ هزار و ۹۰۰ نشان می‌دهد. شکل اول سمت چپ لباسی است که باید تغییر پیدا کند، شکل سمت راست اول ژستی است که لباس باید به خود بگیرد و شکل وسط خروجی مدل است.



شکل ۵-۵- بخشی از نتایج خروجی شبکه‌ی عصبی مژول اول را در EPOCH و مرحله‌ی ۳۵ هزار و ۳۰۰ نشان می‌دهد. شکل اول سمت چپ لباسی است که باید تغییر پیدا کند، شکل سمت راست اول ژستی است که لباس باید به خود بگیرد و شکل وسط خروجی مدل است.



شکل ۵-۶- بخشی از نتایج خروجی شبکه‌ی عصبی مازول اول را در EPOCH و مرحله‌ی ۶۶ هزار و ۹۰۰ نشان می‌دهد. . شکل اول سمت چپ لباسی است که باید تغییر پیدا کند، شکل سمت راست اول ژستی است که لباس باید به خود بگیرد و شکل وسط خروجی مدل است.



شکل ۵-۷- تصویر سمت چپ عکس مرحوم و تصویر سمت راست لباس هدف است در شکل ۵-۷ نقشه‌ی بخش‌بندی تولیدی ناشی از مازول ۲ را مشاهده می‌کنیم.



شکل ۵-۸- نقشه‌ی بخش‌بندی تولیدی ناشی از مازول ۲ را برای شکل ۵-۷ مشاهده می‌کنیم.



شکل ۵-۹- تصویر سمت چپ عکس مرحوم و تصویر سمت راست لباس هدف است در شکل ۵-۱۰- نقشه‌ی بخش‌بندی تولیدی ناشی از مازول ۲ را مشاهده می‌کنیم.



شکل ۱۰-۵ - نقشه‌ی بخش‌بندی تولیدی ناشی از مازول ۲ را برای شکل ۹-۵ مشاهده می‌کنیم.

۶ فصل ۶ – جمع‌بندی و کارهای آینده

در این پژوهه از ابزارهای مختلف موجود در هوش مصنوعی، مدل‌های آموزش عمیق، شبکه‌های تولید‌کننده و روش‌های تغییر شکل عکس استفاده شده است. روش جدیدی که در این پژوهه ارائه شد در کاربردهای تست کردن اشیا به صورت مجازی مورد استفاده قرار می‌گیرد. این روش از سه مازول مختلف تشکیل شده بود مازول اول یک لباس تغییر حالت یافته ایجاد می‌کرد، مازول بعدی یک نقشه‌ی بخش‌بندی جدید از شخص هنگامی که لباس را پوشیده است ایجاد می‌کرد و در نهایت مازول سوم که مازول سنتز بود، خروجی دو مازول قبل را به یکدیگر متصل می‌کند. به منظور افزایش کیفیت خروجی نهایی این روش از ایده‌های جدید نسبت به دو روش کلی موجود تاکنون، استفاده کرده است.

در نهایت روش معرفی شده در فصل چهارم این پژوهه نسبت به دو روش مطرح شده در فصل سوم عملکرد بهتری را از خود نشان داده است. در این پژوهه تنها قسمتی از این روش پیاده‌سازی شده است که شامل مازول اول و قسمتی از مازول دوم معرفی شده در فصل چهارم است. امید است در آینده مازول سوم نیز به طور کامل پیاده‌سازی شود و سیستم نهایی برای هماهنگی با واقعیت افزوده ایجاد شود.

٧ مراجع

- [١] hlombaert, “thin plates,” <https://profs.etsmtl.ca/hlombaert/thinplates./>, 2017
- [٢] R. A. J. S. Ignacio Rocco, “Convolutional neural network architecture for geometric matching .”, 2017
- [٣] Xiaolong Wang, “Non-local neural networks ”,*IEEE Conferenceon Computer Vision and Pattern Recognition (CVPR)* ,١ ٤ ,p. 4, 2018 .
- [٤] Zhe Cao, “ Realtime multi-person 2d pose estimation using part affinity fields ”, *IEEE Conference on ComputerVision and Pattern Recognition* ,pp. 7291-7299, 2017 .
- [٥] Ke Gong, “Look into person: Self-supervised structure-sensitive learning and a new benchmark for human parsing ”,*IEEE Conference on Computer Visionand Pattern Recognition* ,p. 93 .
- [٦] Ke Gong, “ "Look into Person: Self-supervised Structure-sensitive Learning and A New Benchmark for Human Parsing",” https://github.com/Engineering-Course/LIP_SSL, 2017.
- [٧] C. Szegedy, “Rethinking the Inception Architecture for Computer Vision,” University College London, 2016.
- [٨] F. Chollet, “Xception: Deep Learning with depthwise separable convolutions ”, *conference on computer vision and pattern recognition* ,pp. 1251-1258, 2017 .
- [٩] GAN networks, “GANS,” <https://towardsdatascience.com/understanding-generative-adversarial-networks-gans-cd6e4651a29>.
- [١٠] sung-Yi Lin, “ Focal loss for dense object detection ”,*IEEE transactions on pattern analysis and machine intelligence* .٢٠١٨ ,
- [١١] sung-Yi Lin, “ Focal loss for dense object detection ”,*IEEE transactions on pattern analysis and machine intelligence* .٢٠١٨ ,

- [¹¹] Phillip Isola, “Image-to-image translation with conditional adversarial networks”, *the IEEE conference on computer vision and pattern recognition*, pp. 1125-1134, 2017 .
- [¹²] Olaf Ronneberger, “U-Net: Convolutional Networks for Biomedical Image Segmentation :arXiv:1505.04597.”
- [¹³] Simonyan, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” International Conference on Learning Representations, 2015.
- [¹⁴] Xintong Han, “Viton: An image-based virtual try-on network”, *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7543-7552, 2018 .
- [¹⁵] Bochao Wang, “Toward characteristic-preserving image-based virtual try-on network”, *the European Conference on Computer Vision(ECCV)* ,p. pages 589–604 .²⁰¹⁸,
- [¹⁶] Ruiyun Yu, “VTNFP: An Image-based Virtual Try-on Network with Body and ClothingFeature Preservation,” Department of computer science, University of California, Irvine, CA 92617, 2019.
- [¹⁷] Phillip Isola, “Image-to-image translation with conditional adversarial networks”, *IEEE conference on computer vision and pattern recognition*, pp. 1125-1134, 2017 .