

به نام خداوند جان و خرد

کزین برتر اندیشه برگذرد

پروژه اول - یادگیری عمیق پیشرفته

علی هدایت نیا

۸۱۰۱۰۵۰۶

فهرست

سوال (۱): دسته‌بندی و اعتبارسنجی چهره.....	۵
۱- مقدمه:.....	۶
۱.۱- مراحل سیستم تشخیص چهره.....	۶
۱.۱.۱- کاربردها و نگرانی‌ها.....	۹
۱.۲- دسته‌بندی و اعتبارسنجی چهره.....	۱۰
۱.۲.۱- بررسی ArcFace.....	۱۰
۱.۲.۲- تحلیل اکتشافی داده.....	۱۲
۱.۲.۳- پیش پردازش.....	۱۲
۱.۲.۴- بارگذاری داده.....	۱۵
۱.۲.۵- شبکه:.....	۱۶
۱.۲.۶- آموزش شبکه دسته‌بندی.....	۱۸
۱.۲.۷- سنجش شباهت اعتبارسنجی.....	۱۹
۱.۲.۸- نتایج.....	۱۹
۱.۲.۹- داده جدید.....	۲۳
۱.۲.۱۰- ادامه راه.....	۲۳
سوال (۲): دسته‌بندی تصاویر ماشین با Efficient-B+.....	۲۴
۲.۱- دسته‌بندی داده.....	۲۵
۲.۲- انتقال تجربه.....	۲۵
۲.۳- مجموعه داده.....	۲۵
۲.۳.۱- پیش پردازش داده‌ها.....	۲۶
۲.۴- بارگذاری داده.....	۲۷
۲.۵- شبکه.....	۲۸
۲.۶- تابع خط.....	۲۸
۲.۷- بهینه‌سازی و نرخ یادگیری.....	۲۹
۲.۸- نتایج.....	۲۹
۲.۹- نتیجه‌گیری:.....	۳۱
۲.۱۰- مصورسازی شبکه:.....	۳۲
خوداظهاری.....	۳۳

فهرست شکل‌ها

۷ شکل ۱- فیلترهای HARR
۷ شکل ۲- تشخیص چهره‌های موجود در تصویر
۸ شکل ۳- تصاویر نشانه‌گذاری شده چهره
۹ شکل ۴- تبدیل تصویر ورودی به بردار ویژگی منحصر به فرد
۱۰ شکل ۵- بدست آوردن میزان شباهت دو چهره با استفاده از Embedding های آن‌ها
۱۳ شکل ۶- پراکندگی (الف) تعداد و (ب) درصد درست و نادرست بودن مقایسه‌ها در داده‌های اعتبارسنجی
۱۴ شکل ۷- تصاویر مجموعه‌های (الف) آزمون و کنترل کیفیت (ب) آموزش مورد استفاده در آموزش طبقه‌بند
۱۵ شکل ۸- داده‌های اعتبارسنجی
۱۷ شکل ۹- اعتبارسنجی دو تصویر چهره و تایید شباهت یا عدم شباهت این دو چهره
۲۰ شکل ۱۰- تابع خطای آموزش طبقه‌بند
۲۰ شکل ۱۱- نمودار خطای آموزش و کنترل کیفیت (الف) ResNet۱۸، (ب) ResNet۱۵ و (ج) Gray+ResNet۱۸
۲۱ شکل ۱۲- نمودار ROC برای سه شبکه ResNet۱۸، Gray+ResNet۱۸ و ResNet۱۵
۲۶ شکل ۱۳- نمای کلی از داده‌های (الف) آموزش و (ب) آزمون مجموعه داده Stanford Car ۱۹۶
۲۸ شکل ۱۴- اعمال تبدیل‌های آموزش بر تعدادی از تصاویر
۳۰ شکل ۱۵- نمودار تابع خطا در (الف) شبکه پایه با پارامترهای ثابت، (ب) شبکه پایه با پارامترهای نسبتاً ثابت (لایه ۶ و ۷ و غیرثابت) و (پ) شبکه پایه با پارامترهای غیرثابت
۳۰ شکل ۱۶- نمودار دقت در (الف) شبکه پایه با پارامترهای ثابت، (ب) شبکه پایه با پارامترهای نسبتاً ثابت (لایه ۶ و ۷ و غیرثابت) و (پ) شبکه پایه با پارامترهای غیرثابت
۳۰ شکل ۱۷- نمودار تغییرات نرخ یادگیری در (الف) شبکه پایه با پارامترهای ثابت، (ب) شبکه پایه با پارامترهای نسبتاً ثابت (لایه ۶ و ۷ و غیرثابت) و (پ) شبکه پایه با پارامترهای غیرثابت
۳۲ شکل ۱۸- مصورسازی‌های مختلف داده ورودی بعد از آموزش مدل بر روی سه مدل با شبکه پایه EfficientNet-B۰ با (الف) پارامترهای ثابت، (ب) پارامترهای نسبتاً ثابت و (ج) پارامترهای غیرثابت

فهرست جداول

جدول ۱- تعداد داده‌های مجموعه‌های آموزش، کنترل کیفیت و آزمون.....	۱۲
جدول ۲- تبدیل‌های به کار رفته بر روی تصویر ورودی در آموزش طبقه‌بند.....	۱۵
جدول ۳- تبدیل‌های به کار رفته بر روی تصاویر ورودی در مرحله اعتبارسنجی.....	۱۶
جدول ۴- مقدار Hyper-Parameters استفاده شده برای آموزش مدل.....	۱۸
جدول ۵- بررسی معیارهای مختلف بر روی ساختارهای مختلف.....	۲۲
جدول ۶- تعداد داده‌های مجموعه‌های آموزش، کنترل کیفیت و آزمون در ۱۹۶ Standford Car	۲۵
جدول ۷- تبدیل‌های اعمال شده بر داده‌های آموزش Stanford Car ۱۹۶	۲۷
جدول ۸- تبدیل‌های اعمال شده بر داده‌های کنترل کیفیت Stanford Car ۱۹۶	۲۷
جدول ۹- دقیق داده‌های آزمون بر روی سه ساختار با شبکه‌های پایه ثابت، نسبتاً ثابت و غیر ثابت.....	۳۱

فهرست کدها

۱۸	کد ۱- بهینه‌سازی همزمان دو شبکه
۲۹	کد ۲- استفاده از برنامه ریز EfficientNet-B ₀ برای آموزش شبکه ReduceLROPlateau

سوال (۱): دسته‌بندی و

اعتبارسنجی چهره

۱-۱- مقدمه:

بر اساس دانشنامه اینترنی ویکیپدیا، سیستم تشخیص چهره به صورت زیر تعریف می‌شود:
((یک سیستم تشخیص چهره، سیستمی است که قادر به شناسایی یا تطبیق یک چهره از روی یک تصویر یا
یک فیلم، است.))

در واقع به طور دقیق‌تر، یک سیستم تشخیص چهره، یک نرمافزار داده‌شناسی زیستی^۱ است که با توجه به ویژگی‌های چهره فرد، قادر به متمایز کردن شخص از سایر افراد است. امروزه با گسترش، روش‌های هوش‌مصنوعی و یادگیری عمیق، به راه حلی هوشمند برای این مسئله، نزدیک شده‌ایم. اما به راستی، این سیستم چگونه کار می‌کند. در زیر به طور اجمالی به بررسی مراحل این سیستم می‌پردازیم.

۱.۱.۲- مراحل سیستم تشخیص چهره

یک سیستم تشخیص چهره را می‌توان به چند مرحله شکست که در زیر هر کدام از این مراحل را به طور خلاصه و مجزا بررسی می‌کنیم.

۱.۱.۲.۱- مرحله اول- شناسایی چهره^۲

در تشخیص چهره، چهره افراد در تصویر را می‌یابیم. برای شناسایی چهره، یک پنجره را بر روی تصویر می‌غلتانیم و چهره‌هایی را که در این پنجره قرار می‌گیرند، تشخیص می‌دهیم. تشخیص چهره در یک پنجره، بدین صورت است که یک سری ویژگی از تصویر درون این پنجره استخراج می‌کنیم و با استفاده از این ویژگی‌ها، تصمیم می‌گیریم که درون این پنجره، چهره‌ای هست یا نه؟ ویژگی‌هایی که از یک پنجره گرفته می‌شود، علاوه بر این که یافتن آن‌ها باید به سرعت انجام شود، همچنین باید قادر به تمایز وجود یا عدم وجود چهره باشند.

برای بدست آوردن این ویژگی‌ها راه‌های متفاوتی وجود دارد که با استفاده از گوشش‌های تصویر، جهت‌گیری‌ها را می‌یابند تا به یک سری ویژگی از تصویر برسند که با آن، در مورد وجود یا عدم وجود چهره درون این پنجره تصمیم گیری شود یا این که تصویر درون پنجره را به اجزاء مختلف صورت می‌شکنند و با استفاده از مکان قرارگیری این اجزاء پیش‌بینی شده، وجود چهره در تصویر را مورد بررسی قرار می‌دهند. راه حلی که مرسوم‌تر است بدین صورت است که برای یافتن ویژگی‌های تصویر درون پنجره از کانولوشن آن با یک سری فیلترهایی به نام فیلترهای HAAR استفاده می‌کنند. در زیر چند نمونه‌ای از این فیلترها را مشاهده می‌کنید که سطح سیاه مربوط به مقدار پایین و سطح سفید مربوط به مقدار بالا است. در شکل (۲) چند نمونه شناسایی چهره را می‌بینید که با همین روش، چهره تشخیص داده شده است.

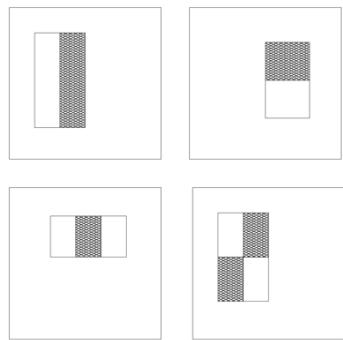
۱.۱.۲.۲- مرحله دوم- آنالیز چهره^۳

در این قسمت با استفاده از تصویر چهره افراد، چهره وی را آنالیز می‌کنیم، بدین‌گونه که با استفاده از فاصله اجزاء از یکدیگر می‌توان، چهره را نشانه گذاری کرد. در شکل (۳) چهره‌های نشانه گذاری شده را مشاهده می‌کنید.

Bioinformatics^۱

Face Detection^۲

Face Analysis^۳



شکل ۱- فیلترهای HARR

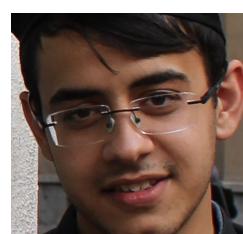
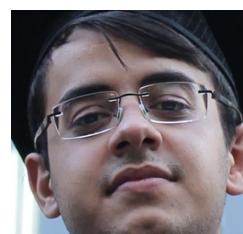
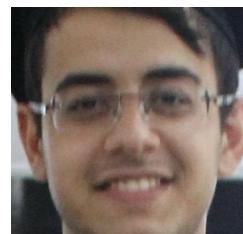
تصویر اصلی



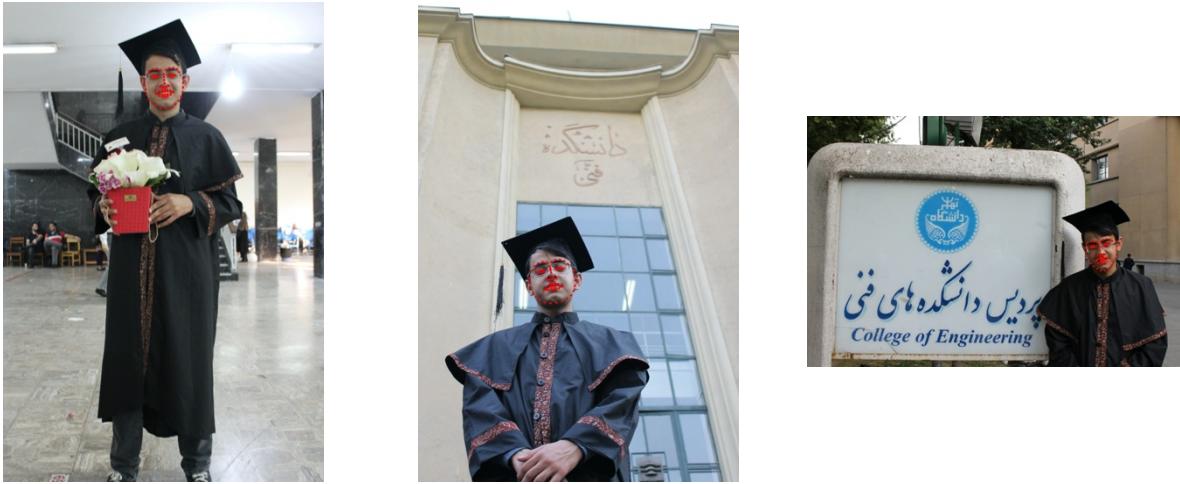
تصویر چهره شناسایی شده



چهره بریده شده



شکل ۲- تشخیص چهره‌های موجود در تصویر



شکل ۳- تصاویر نشانه‌گذاری شده چهره

۱.۱.۲.۳- مرحله سوم- تبدیل چهره به یک بردار ویژگی منحصر به فرد

پس از یافتن چهره فرد، آن را به یک بردار ویژگی تبدیل می‌کنیم. این بردار ویژگی باید به ازای هر فرد منحصر به فرد باشد و به‌گونه‌ای امضای فرد به حساب می‌آید. برای این منظور باید فاصله بردارهای ویژگی هر فرد با تصاویر خودش کم و با تصاویر سایر افراد زیاد باشد. یکی از چالش‌هایی که در این پروژه با آن روبرو هستیم، یافتن چنین بردار ویژگی است.

۱.۱.۲.۴- مرحله چهارم- مقایسه بردارهای ویژگی

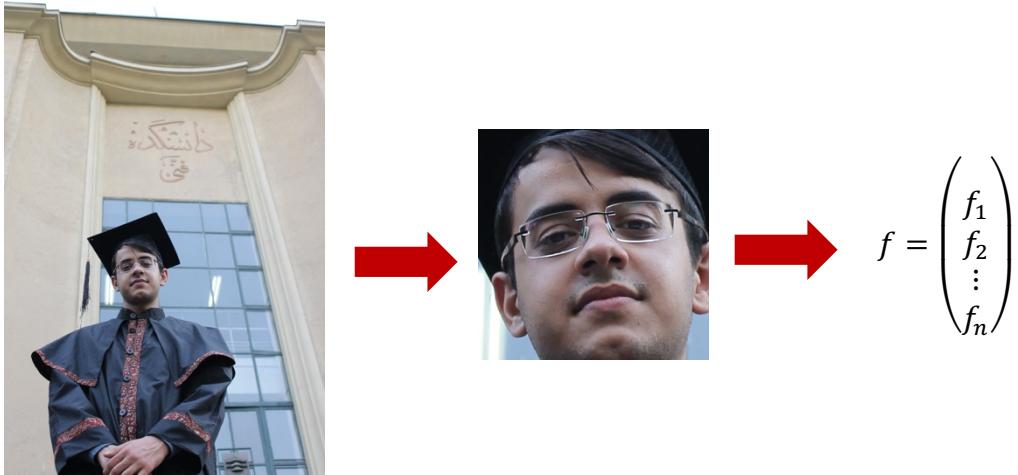
با یافتن بردار ویژگی منحصر به فرد، برای شناسایی تصویر ورودی، می‌توان شباهت بردار ویژگی مربوط به آن را با بردار ویژگی سایر افراد مقایسه کرد و اگر این بردار با بردار ویژگی فرد A بسیار شبیه بود، این تصویر نیز متعلق به فرد A است. برای مقایسه شباهت دو بردار می‌توان از انواع فاصله استفاده کرد. بدین صورت که هرچه فاصله کمتر باشد، دو بردار به یکدیگر شبیه‌تر هستند. برای این کار می‌توان فاصله‌های مختلفی را در نظر گرفت که در زیر به چند مورد از آن‌ها اشاره می‌کنیم:

- **فاصله اقلیدسی:** این فاصله به گونه‌ای نمایانگر اندازه تفاضل دو بردار است. فاصله اقلیدسی به صورت زیر تعریف می‌شود:

$$Euclidean(x, y) = \|x - y\|_2 = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

- **فاصله کسینوسی:** این فاصله بدین صورت است که اگر به تمام بردارها از مرکز نگاه کنیم، اندازه آن‌ها برایمان مهم نیست و تنها سوگیری آن‌هاست که اهمیت دارد. این فاصله به صورت زیر تعریف می‌شود:

$$\text{Cosine}(x, y) = \arccos\left(\frac{x \cdot y}{\|x\| \cdot \|y\|}\right)$$



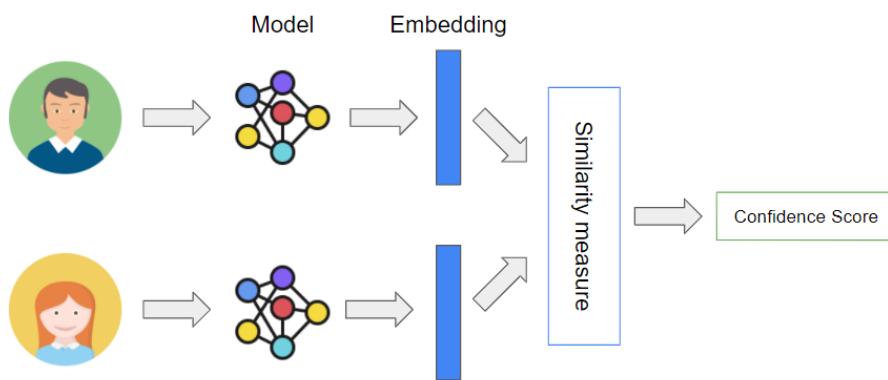
شکل ۴- تبدیل تصویر ورودی به بردار ویژگی منحصر به فرد

۱.۱.۳ - کاربردها و نگرانی‌ها

امروزه شاهد استفاده از سیستم‌های تشخیص چهره از شبکه‌های اجتماعی تا به مسائل امنیتی هستیم. حداقل استفاده‌ای که امروزه بیشتر افراد با آن سروکار دارند، باز کردن گوشه همراه با استفاده از سیستم تشخیص چهره است. اما همراه با افزایش کاربرد این سیستم‌ها در زندگی روزمره افراد، نگرانی‌هایی را نیز به همراه داشته است که از مهم‌ترین آن‌ها دغدغه حفظ حریم شخصی و امنیت است. به طور مثال اگر شرکتی برای ورود و خروج افراد از این سیستم‌ها استفاده کند، تشخیص اشتباه افراد، ممکن است مسئله‌ساز شود. همچنین در دسترس بودن چنین سیستم‌های به صورت عمومی باعث می‌شود که افرادی از این سیستم‌ها سوءاستفاده کنند و با دنبال کردن افراد، برای آن‌ها دردرس ایجاد کنند.

۱.۲.۱ - دسته‌بندی و اعتبار سنجی چهره^۴

سیستمی که برای تشخیص چهره استفاده می‌کنیم، بدین صورت است که ابتدا به سراغ یک مسئله دسته‌بندی تصاویر چهره افراد مختلف می‌رویم و یک شبکه عصبی را برای دسته‌بندی این تصاویر افراد، آموزش می‌دهیم. سپس با استفاده از این شبکه، آموزش داده شده، با استفاده از لایه‌های میانی این شبکه عصبی به ویژگی‌های مرتبط با تصویر چهره دست می‌یابیم که این ویژگی به گونه‌ای نمایانگر امضایی از تصویر چهره‌اند. با بدست آوردن این امضا از تصویر چهره، می‌توان دو چهره با یکدیگر مقایسه کرد و در مورد این که یک تصویر مربوط به کیست، با توجه به یک پایگاه داده، تصمیم‌گیری کرد. در واقع می‌توان گفت که در مسئله تشخیص چهره با این رویکرد، یافتن Embedding خوب، نقشی حیاتی دارد و Embedding‌ای خوب است که فاصله درون کلاسی آن کم و فاصله بین کلاسی آن بسیار زیاد باشد.



شکل ۵ - بدست آوردن میزان شباهت دو چهره با استفاده از Embedding‌های آن‌ها

برای بدست آوردن این Embedding‌ها در این پژوهه از مقاله ArcFace استفاده می‌کنیم. بنابراین، ابتدا به بررسی این ساختار می‌پردازیم.

۱.۲.۲ - بررسی ArcFace

مسئله دسته‌بندی تصاویر چهره همانند هر مسئله دسته‌بندی دیگر است و از آنجایی که در این مسئله به دنبال کم کردن فاصله درون کلاسی و افزایش فاصله بین کلاسی هستیم؛ بنابراین، باید در انتخاب تابع هزینه^۵ دقت کافی داشته باشیم. پیشتر از دو تابع هزینه Softmax و یا Triplet استفاده می‌شده است. اما این تابع‌ها مشکلاتی را به همراه دارند که در زیر به بررسی آن‌ها می‌پردازیم:

- تابع هزینه Softmax: با افزایش تعداد افراد، تعداد ستون‌های ماتریس وزن لایه نهایی نیز افزایش می‌یابد و همچنین این تابع هزینه تنها قادر به تمایز داده‌های مشاهده شده در داده‌های Train است و برای استفاده در مسائل به اصطلاح Open-set همچون تشخیص چهره مناسب نیست.

Face Classification & Identification⁴

Loss Function⁵

- تابع هزینه Triplet: این تابع هزینه سعی می‌کند که بردار ویژگی یک چهره را به بردار ویژگی هم گروهی آن نزدیک و از غیر هم گروهی آن دور کند که این باعث کند شدن فرآیند آموزش می‌شود.
- همانطور که مشاهده شد، در مسئله دسته‌بندی چهره‌ها نیازمند معرفی یک تابع هزینه هستیم که مشکلات توابع هزینه ذکر شده را برطرف کند.

۱.۲.۲.۱- بررسی تابع هزینه ArcFace

همانطور که پیشتر گفته شد، بدنبال یافتن یک Embedding مطلوب هستیم که داده‌های شبیه‌تر به یکدیگر نزدیک‌تر باشند و برای بدست آوردن این Embedding نیازمند معرفی یک تابع هزینه جدید هستیم. برای درک بهتر این تابع هزینه، خوب است که ابتدا یک نگاه کوتاه به تابع هزینه Softmax بیاندازیم:

$$L_{softmax} = -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{W_j^T x_i + b_{y_i}}}{\sum_{j=1}^n e^{W_j^T x_j + b_{y_j}}}$$

در رابطه فوق N نمایانگر اندازه Batch و n نشان‌دهنده تعداد دسته‌هاست. این رابطه به طور صریح، ویژگی‌های Embedding را بهینه نمی‌کند که ویژگی‌های درون یک گروه به یکدیگر شبیه و با خارج آن گروه، متفاوت باشند. پس برای حل این مشکل تابع هزینه زیر را معرفی می‌کنیم:

$$L_{ArcFace} = -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{s \cdot \cos(\theta_{y_i} + m)}}{e^{s \cdot \cos(\theta_{y_i} + m)} + \sum_{j=1, j \neq y_i}^n e^{s \cdot \cos(\theta_{y_j})}}$$

این تابع هزینه برای بدست آوردن Embedding مطلوب، سعی می‌کند به گونه‌ای بردارهای ویژگی را به یک کره نگاشت کند که نقاط هم گروه در کنار یکدیگر قرار گیرند. برای این منظور از فاصله زاویه بین گروه‌ها استفاده می‌کنیم و سعی می‌کنیم که فاصله زاویه‌ای هم گروهی‌ها به یکدیگر نزدیک و غیر هم گروهی‌ها از یک دیگر دور باشند. همچنین در این تابع هزینه سعی می‌کنیم که یک مقدار حاشیه‌ای^۶ بین کلاس‌های غیر هم گروه وجود داشته باشد. برای بدست آوردن این زاویه‌ها از رابطه ضرب داخلی بردارهای نرمال شده استفاده می‌کنیم:

$$W_j^T \cdot x_i = \|W_j^T\| \cdot \|x_i\| \cdot \cos(\theta_j) \Rightarrow \theta_j = \arccos \frac{W_j^T \cdot x_i}{\|W_j^T\| \cdot \|x_i\|}$$

رابطه فوق را می‌توان به قالبی کلی نیز تغییر شکل داد که در آن تابع هزینه CosFace و SphereFace هم در نظر گرفته شده باشد:

$$L_{ArcFace} = -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{s(\cos(m_1 \theta_{y_i} + m_2) - m_3)}}{e^{s(\cos(m_1 \theta_{y_i} + m_2) - m_3)} + \sum_{j=1, j \neq y_i}^n e^{s \cdot \cos(\theta_j)}}$$

Margin⁶

پس از یادگیری شبکه دسته‌بندی، از این شبکه برای بدست آوردن Embedding‌های تصویر چهره استفاده می‌کنیم و پس از آن با استفاده از یک تابع فاصله همچون تابع فاصله اقلیدسی یا تابع فاصله کسینوسی، میزان شباهت بین دو تصویر را می‌سنجیم و بر مبنای آن، یکتا بودن چهره‌ها را بررسی می‌کنیم.

۱.۲.۳- تحلیل اکتشافی داده^۷

داده‌هایی که در این پروژه استفاده شده‌اند، شامل دو تا دسته داده برای دسته‌بندی و اعتبارسنجی است که در زیر به طور مجزا به این دو دسته می‌پردازیم.

۱.۲.۳.۱- داده‌های دسته‌بندی

داده‌های دسته‌بندی شامل دو دسته آموزش^۸ و آزمون^۹ تقسیم می‌شود. مجموعه آموزش را به دو دسته آموزش و کنترل کیفیت^{۱۰} با نسبت به ترتیب ۰.۸ و ۰.۲ نسبت به کل، تقسیم می‌کنیم که این تقسیم‌بندی با توجه به مسئله قشر‌بندی^{۱۱} انجام شده است تا توزیع داده‌ها حفظ شود. استفاده از داده‌های کنترل کیفیت برای میزان کردن ابر پارامتر^{۱۲}‌های مدل استفاده می‌شود.

مجموعه آموزش	مجموعه کنترل کیفیت	مجموعه آزمون	تعداد گروه‌ها	تعداد داده‌های به ازای هر گروه
۱۰۰۰	۱۰۰۰	۱۰۰۰	۱۶	۴
۱۰۰۰	۱۰۰۰	۱۰۰۰	۴	۵

جدول ۱- تعداد داده‌های مجموعه‌های آموزش، کنترل کیفیت و آزمون

در شکل (۶) داده‌های آزمون و کنترل کیفیت و آزمون مورد استفاده برای آموزش طبقه‌بند را مشاهده می‌کنید. داده‌هایی که در اینجا به شبکه داده می‌شود، به اندازه ۱۱۲×۱۱۲ هستند.

۱.۲.۳.۱.۱- پیش پردازش

داده‌های آموزش و آزمون در دو دایرکتوری مجزا قرار دارند. در هر کدام از آن‌ها به ازای هر گروه یک دایرکتوری قرار دارد که در آن دایرکتوری تصاویر مربوط به یک گروه قرار دارد. برای این‌که در بار کردن در زمان آموزش و آزمون مدل، آسوده خاطر باشیم، یک Dataframe از این داده‌ها می‌سازیم که هر سطر شامل نام تصویر، نام گروه (دایرکتوری) و شماره یکتا مربوط به گروه که یک شماره بالاشمار از ۰ تا ۹۹۹ است. در داده‌های آموزش این Dataframe، علاوه بر این ویژگی‌ها به ازای هر سطر، یک ویژگی دیگر

Exploratory Data Analysis⁷

Training Set⁸

Test Set⁹

Validation Set¹⁰

Stratification¹¹

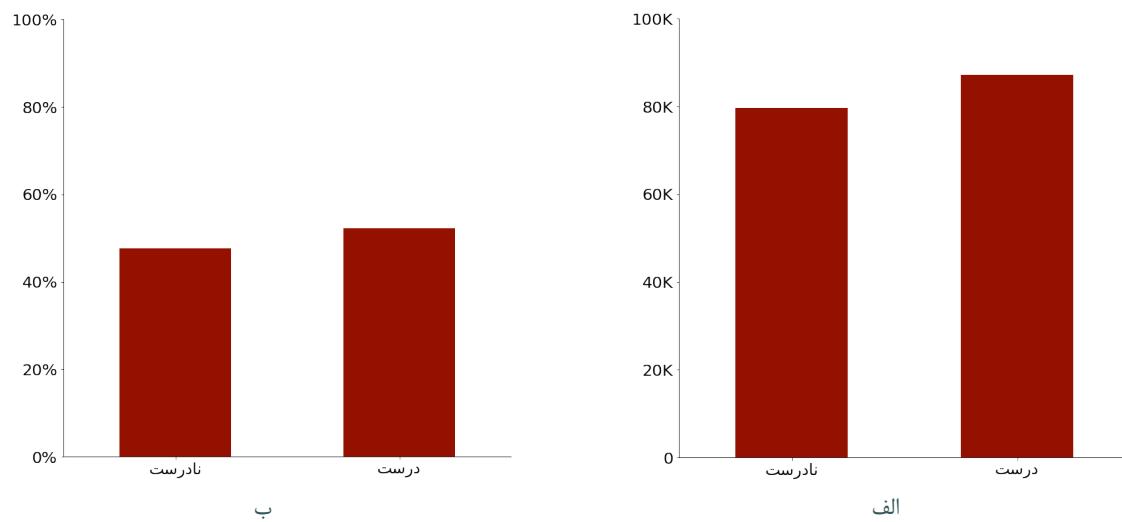
Hyperparameter¹²

داریم که نشان‌دهنده، حالت استفاده از این داده است که یکی از دو مقدار "train" یا "val" را دارد که به ترتیب نشان‌دهنده این است که این داده در آموزش و یا کنترل کیفیت استفاده می‌شود.

۱.۲.۳.۲ - داده‌های اعتبارسنجی

داده‌های اعتبارسنجی شامل یک سری تصویر به همراه یک فایل به نام "verification_dev.csv" است که هر سطر آن شامل اسم دو تصویر و نتیجه مقایسه آن دو تصویر است. در صورتی که دو تصویر مربوط به یک چهره باشند، مقدار این ستون ۱ و در غیر این صورت، مقدار آن ۰ است. در شکل (۷) یک نگاه کلی به این تصاویر را می‌بینید. اندازه این داده‌ها را نیز برای ورود به شبکه به ۱۱۲X۱۱۲ تغییر می‌دهیم.

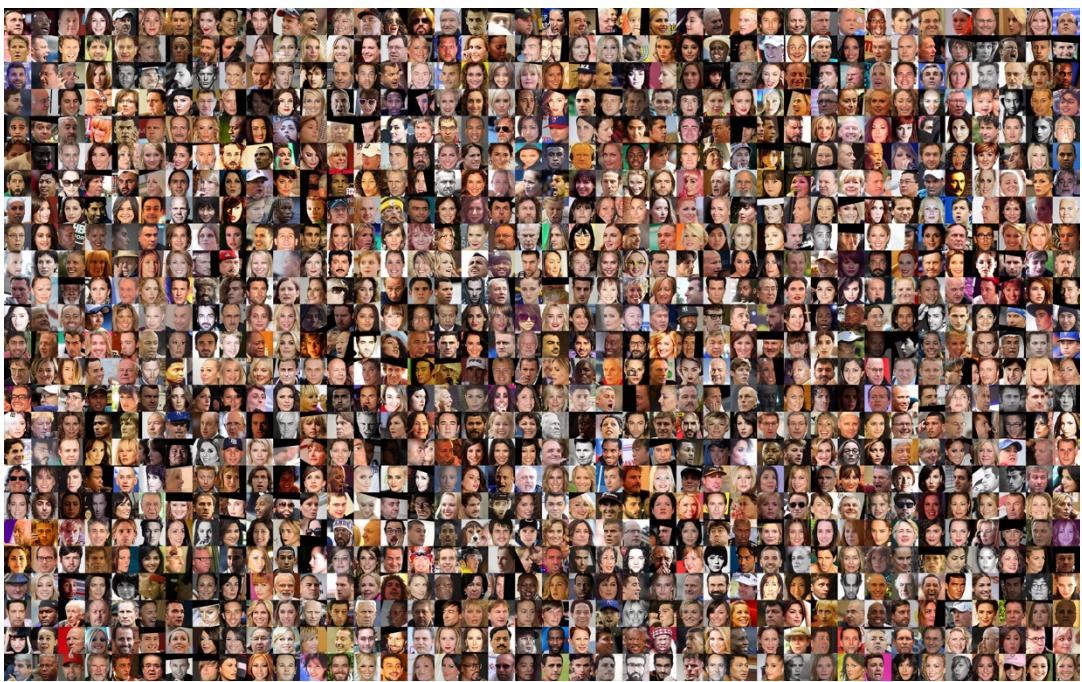
تعداد مقایسه‌های در مجموعه داده‌های اعتبارسنجی ۱۶۶۸۰۰ مقایسه است که پراکنده‌گی درست و نادرست بودن آن را در نمودار زیر مشاهده می‌کنید. همانطور که در نمودار دیده می‌شود، تقریباً جواب نیمی از مقایسه‌ها نادرست و جواب نیمه دیگر نیز درست است.



شکل ۶-پراکنده‌گی (الف) تعداد و (ب) درصد درست و نادرست بودن مقایسه‌ها در داده‌های اعتبارسنجی

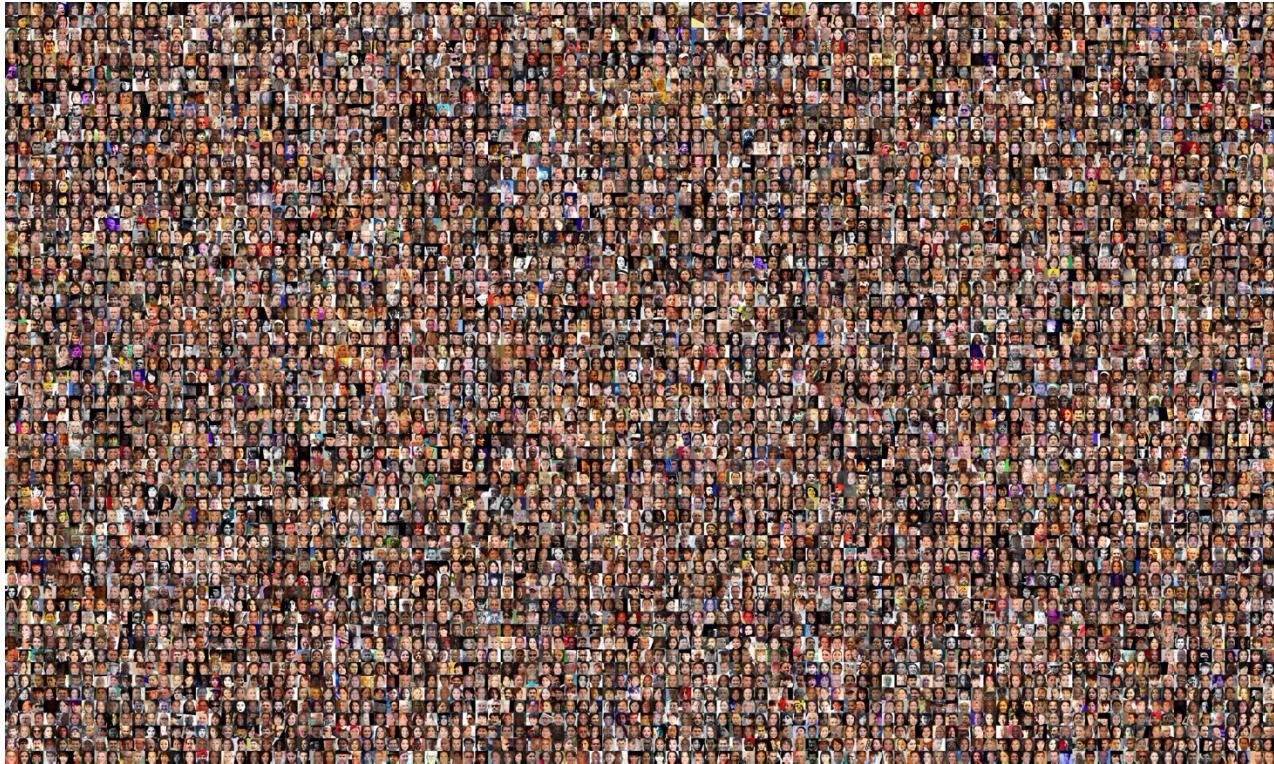


(الف)



(ب)

شکل ۷- تصاویر مجموعه‌های (الف) آزمون و کنترل کیفیت ب) آموزش مورد استفاده در آموزش طبقه‌بند



شکل ۸- داده‌های اعتبارسنجی

۱.۲.۴- بارگذاری داده

۱.۲.۴.۱- بارگذاری داده‌های دسته‌بندی^{۱۳}

Data Loader داده‌های دسته‌بندی را به صورت زوج داده و برچسب بر می‌گردانند که داده آن، تنسور تصویر و برچسب مربوط به آن، به صورت One-Hot است. همچنین بر روی این تصویر، یک سری تبدیل‌هایی اعمال می‌شوند که تبدیل‌هایی که برای آموزش طبقه‌بند استفاده شده است، به شرح ذیل هستند:

تصویر ورودی به اندازه مشخصی تبدیل می‌کنیم. اندازه‌ای که در اینجا استفاده شده است، به شرح ذیل هستند:
`transforms.Resize(input_size)`

با احتمال p تصویر را به صورت افقی معکوس می‌کند.
`transforms.RandomHorizontalFlip(p)`

تصویر ورودی را به `torch.Tensor` تبدیل می‌کند.
`transforms.ToTensor()`

تنسر ورودی را با میانگین و انحراف از معیار مشخص نرمالیز می‌کنیم.
`transforms.Normalize(mean, std)`

میانگین و انحراف از معیاری که در اینجا در نظر گرفته‌ایم، هر کدام به ترتیب ۰.۵ و ۰.۵ است.

جدول ۲- تبدیل‌های به کار رفته بر روی تصویر ورودی در آموزش طبقه‌بند

همچنین برای کنترل کیفیت مدل، از همان تبدیل‌های بالا به جز تبدیل دوم یا همان معکوس کردن افقی تصادفی، استفاده شده است. برای این تبدیل‌ها از تبدیل‌های موجود در `torchvision.transforms` استفاده کرده‌ایم.

همچنین لازم به ذکر است که نحوه بارگذاری داده‌ها در اینجا به این صورت است که با استفاده از فایل 'train.csv' و 'test.csv' که در مرحله پیش‌پردازش داده آن‌ها را تولید کرده بودیم، استفاده می‌کنیم.

۱.۲.۴.۲- بارگذاری داده‌های اعتبارسنجی^{۱۴}

برای بارگذاری داده‌های اعتبارسنجی، ابتدا دو تصویر را با توجه به آدرس آن‌ها که در فایل "verification_dev.csv" وجود دارد به همراه نتیجه تطابق یا عدم تطابق دو تصویر را می‌خوانیم و در نهایت به خروجی ۵ مورد، دو تصویر به همراه آدرس آن‌ها و خروجی تطابق یا عدم تطابق را برمی‌گردانیم. برگرداندن آدرس فایل‌ها برای استفاده در گزارش‌گیری در فاز اعتبارسنجی است. لازم به ذکر است که در این مرحله نیز همان تبدیل‌های زیر اعمال می‌شود:

تصویر ورودی به اندازه مشخصی تبدیل می‌کنیم. اندازه‌ای که در اینجا `transforms.Resize(input_size)` استفاده شده است، ۱۱۲×۱۱۲ است.

تصویر ورودی را به `torch.Tensor` تبدیل می‌کند.
تنسر ورودی را با میانگین و انحراف از معیار مشخص نرمایلز می‌کنیم.
`transforms.Normalize(mean, std)` میانگین و انحراف از معیاری که در اینجا در نظر گرفته‌ایم، هر کدام به ترتیب ۰.۵ و -۰.۵ است.

جدول ۳- تبدیل‌های به کار رفته بر روی تصاویر ورودی در مرحله اعتبارسنجی

۱.۲.۵- شبکه:

یادآوری این نکته خالی از لطف نیست که در این پروژه برای تشخیص چهره، ابتدا یک طبقه‌بند را آموزش می‌دهیم تا به پس از همگرایی تابع هزینه، به یک بردار ویژگی مناسب که بازگوکننده تصویر ورودی است، برسیم. سپس، با استفاده از بردارهای ویژگی که از این طبقه‌بند، برای دو تصویر ورودی به صورت مجزا بدست می‌آید، شبیه بودن یا نبودن دو تصویر چهره را بررسی می‌کنیم. در اینجا به صورت مجزا ساختارها و شبکه‌های استفاده شده در دو مرحله دسته‌بندی و اعتبارسنجی را بررسی می‌کنیم. لازم به ذکر است که بردار ویژگی نهانی که در اینجا به دنبال آن هستیم، یک بردار ۵۱۲×۱ است.

۱.۲.۵.۱- مسئله دسته‌بندی

برای دسته‌بندی داده‌ها، از شبکه ResNet استفاده کرده‌ایم. علت استفاده از این شبکه، نتایج بسیار خوب آن در مسائل دسته‌بندی تصاویر و همچنین عدم بوجود آمدن مشکلاتی همچون ناپدید شدن گرادیان^{۱۵} با افزایش عمق، بوده است. شبکه‌های ResNet، یک دسته از شبکه‌ها با عمق‌های مختلف است. به دلیل محدودیت منابع در دسترس، ساده‌ترین و کم‌عمر ترین شبکه این خانواده یعنی ResNet-18 را انتخاب کرده‌ایم. البته با توجه به این که در پیاده‌سازی ArcFace از شبکه ResNet-50 استفاده کرده بودند، شاید بهتر بود از این شبکه استفاده می‌کردیم اما محدودیت منابع، مرحله بعدی یعنی اعتبارسنجی را به شدت کند می‌کرد.

Verification Data Loader^{۱۴}

Vanishing Gradient^{۱۵}

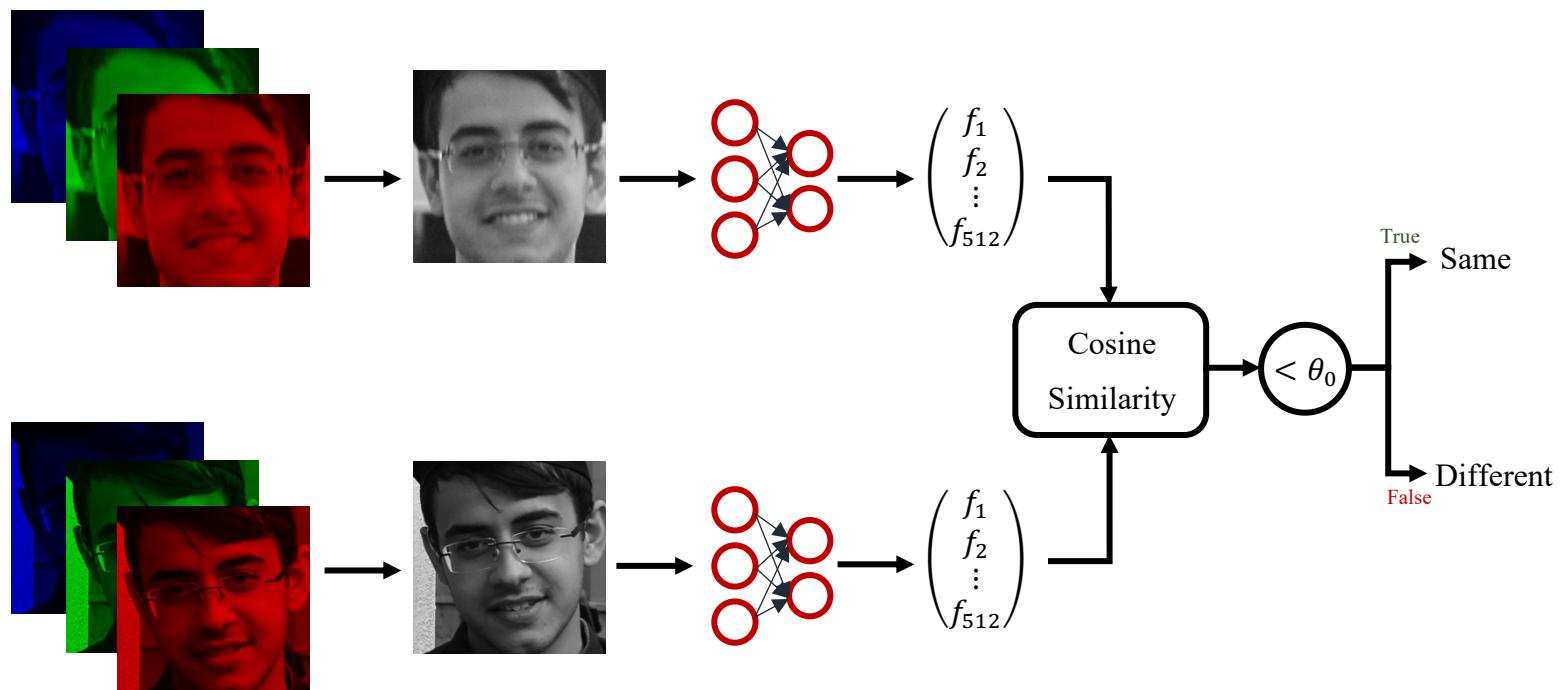
از آنجایی که در اینجا به دنبال یک بردار ویژگی ۵۱۲ بعدی هستیم، لایه آخر شبکه ResNet-18 را به یک لایه Fully Connected با خروجی ۵۱۲ تایی جایگزین می‌کنیم و پس از یک لایه به نام ArcFaceLayer (AFL) تعریف می‌کنیم که این لایه علاوه بر اینکه مسئول دسته‌بندی داده‌هاست، خروجی نیز به صورتی است که باعث سهولت در پیاده‌سازیتابع هزینه می‌شود. خروجی این لایه یک بردار ۱۰۰۰ بعدی است که به صورت زیر مقداردهی می‌شود:

$$out^i = \begin{pmatrix} out_1^i \\ out_2^i \\ \vdots \\ out_{1000}^i \end{pmatrix}_{1000 \times 1} \Rightarrow out_j^i = \begin{cases} s \times \cos(\theta_{y_i} + m) & j = y_i \\ s \times \cos(\theta_j) & \text{otherwise} \end{cases}$$

برای بدست آوردن θ ها از ضرب داخلی بردارهای نرمال شده وزن و بردار ۵۱۲ بعدی استفاده می‌کنیم. در رابطه بالا مقدار m و s را به ترتیب برابر ۰.۵ و ۶۴ قرار داده‌ایم.

۱.۲.۵.۲- مسئله اعتبارسنجی

در اینجا ابتدا، لایه نهایی را غیرفعال می‌کنیم تا به بردارهای ویژگی ۵۱۲ بعدی بررسیم و سپس با عبور دو تصویر، بردارهای ویژگی مربوط به آنها را می‌یابیم. سپس با استفاده از یک معیار فاصله که پیشتر به آنها اشاره شد، فاصله بین این دو بردار ویژگی را بدست می‌آوریم و این فاصله با شباهت رابطه عکس دارد؛ بدین معنا که هرچه فاصله بیشتر باشد، شباهت کمتر است و بر عکس. با بدست آوردن میزان شباهت دو تصویر، اگر این میزان شباهت از یک مقدار آستانه مشخص بیشتر باشد، این دو تصویر متعلق به یک فرد است و در غیر این صورت متفاوت است. برای یافتن این مقدار آستانه، از ROC کمک می‌گیریم.



شکل ۹- اعتبارسنجی دو تصویر چهره و تایید شباهت یا عدم شباهت این دو چهره

۱.۲.۶- آموزش شبکه دسته‌بندی

برای آموزش مدل از تابع هزینه CrossEntropy استفاده می‌کنیم که با توجه به خروجی شبکه دسته‌بندی، در واقع همان تابع هزینه ArcFace را بر روی مسئله دسته‌بندی ساده خواهیم داشت. همچنین برای بهینه‌سازی از الگوریتم Adam استفاده می‌کنیم. نکته‌ای که در اینجا باید به آن توجه شود، در اینجا دو شبکه رو توأم با یکدیگر آموزش می‌دهیم و برای این کار باید الگوریتم بهینه‌سازی را به صورت زیر نمونه‌گیری کنیم:

```
optimizer=optim.Adam([
    {'params': resnet.parameters()},
    {'params': arc_face_layer.parameters()}
], lr,
weight_decay
)
```

کد ۱- بهینه‌سازی همزمان دو شبکه

همچنین Hyper-Parameter‌هایی که برای آموزش شبکه استفاده کرده‌ایم، به شرح زیر است:

Hyper-Parameters	Value	
s	64	
m	0.5	
Epochs	100	
Batch Size	32	
Learning Rate	Epoch [1,20]	1e-1
	Epoch [21,30]	1e-2
	Epoch [31, 40]	1e-3
	Epoch [41, 60]	1e-4
	Epoch [61, 90]	1e-5
	Epoch [91, 100]	1e-6
Momentum (for SGD)	0.9	
Weight Decay	5e-4	

جدول ۴- مقدار استفاده شده برای آموزش مدل Hyper-Parameters

بنابر جدول (۴) آموزش برای ۱۰۰ دوره^{۱۶} انجام شده است که در برایر تعداد دوره‌هایی که در ؟، ArcFace را آموزش داده‌اند ۳۲هزار دوره)، بسیار ناچیز است. بنابراین، نمی‌توان انتظار چندان بالایی از این مدل داشت. علت اینکه این مقدار دوره آموزش داده شده است، این است که به علت کمبود منابع قابلیت آموزش با تعداد دوره بالا وجود نداشت. همچنین از عوامل دیگری که اثر منفی در کیفیت این بردار ویژگی خواهد داشت، شبکه انتخابی (ResNet18) است، چرا که در ؟، از شبکه ResNet50 استفاده شده است که عمق آن تقریباً سه برابر شبکه انتخابی ماست که این باعث کارایی بهتر مدل نسبت به مدل ما می‌شود.^{۱۷} Hyper-Parameter ها نیز مطابق مواردی است که در مقاله ArcFace ذکر شده است.

۱.۲.۶.۱- سنجش آموزش یافتن مدل

برای اینکه، از آموزش یافتن مدل مطمئن شویم، مقدار میانگین تابع هزینه در دوره‌های آموزش مختلف را رسم می‌کنیم. همانطور که مشاهده می‌کنید، مقدار این تابع نزولی است که این نشان‌دهنده آن است که مدل در حال یاد گرفتن است.

۱.۲.۷- سنجش شباهت اعتبارسنجی

برای سنجش شباهت دو تصویر در مرحله اعتبارسنجی، از تشابه کسینوسی^{۱۸} دو بردار ویژگی مربوط تصاویر استفاده کرده‌ایم. تشابه کسینوسی به صورت زیر تعریف می‌شود:

$$\text{CosineSimilarity} = \frac{x_1 \cdot x_2}{\|x_1\| \cdot \|x_2\|}$$

۱.۲.۸- نتایج

در اینجا نتایج را بر روی سه ساختار زیر بررسی می‌کنیم:

- این شبکه در واقع همان شبکه ResNet18 تصحیح شده است که به جای تصویر ۳ کانالی، تصویر ۱ کانالی سیاه و سفید^{۱۹} می‌گیرد.

ResNet18 •

ResNet50 •

۱.۲.۸.۱- نتایج طبقه‌بند

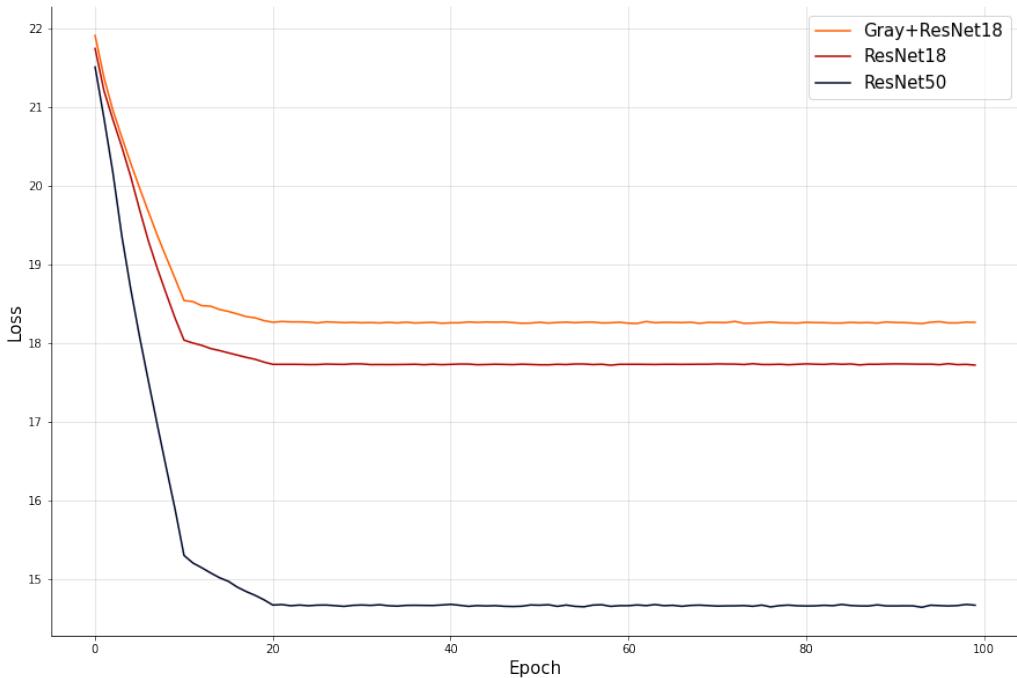
همانطور که از نمودارهای خطای زمان آموزش مدل پیدا است، دو هر سه مدل بعد از حدود ۲۰ دوره آموزش، کاملاً همگرا شده‌اند. همچنین از نمودار پیداست که با سیاه‌وسفید کردن تصویر، مقدار خطای افزایش می‌یابد که این امر قابل فهم است؛ زیرا با سیاه‌وسفید کردن تصویر، مقداری از اطلاعات مفید داده حذف می‌شوند. به طور مثال، یکی از ویژگی‌های مهم در تشخیص چهره رنگ پوست و

Epoch¹⁶

¹⁷ از جمله کارهایی که برای جلوگیری از این دو نقص می‌توان انجام داد، این است که به جای آموزش مدل از صفر، از مدل‌هایی که پیش‌تر آموزش یافته‌اند، استفاده کنیم و با استفاده از این مدل‌ها، مدل را برای داده‌های خودمان تنظیم (Fine-Tune) کنیم.

Cosine Similarity¹⁸

Gray Scale¹⁹

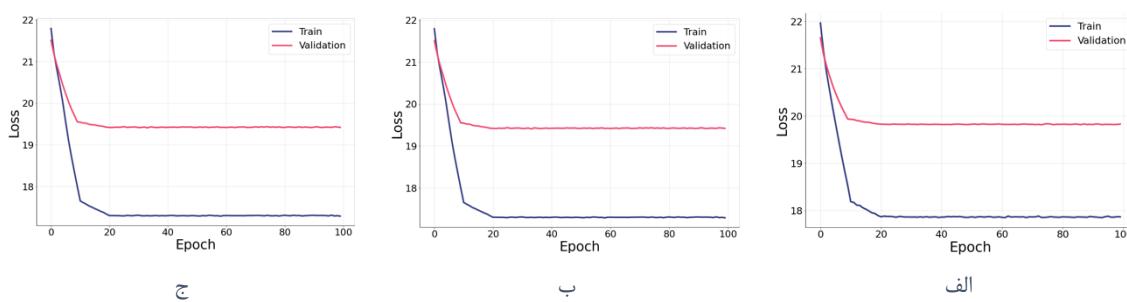


شکل ۱۰- تابع خطای آموزش طبقه‌بند

رنگ چشم است که با سیاه و سفید کردن تصویر، این ویژگی‌ها از بین می‌رود. همچنین از آن جایی مدل ResNet50 دارای پارامترهای بیشتری است، طبیعی است که نسبت به همتای ۱۸ لایه‌ای خود به خطای پایین‌تری همگرا شود.

۱۱.۲.۸.۲- نمودار خطای کنترل کیفیت

به نظر می‌رسد که مدل‌ها پس از گذشت ۲۰ دوره آموزش کاملاً همگرا شده‌اند و از آن‌جا به بعد تغییری در تابع خطای مشاهده نمی‌شود. البته باید توجه داشت که بعد از گذشت ۳۰ دوره آموزش، نرخ یادگیری به قدری کاهش می‌یابد که دیگر مدل توان تغییرات زیاد را ندارد و در واقع این کاهش نرخ یادگیری به گونه‌ای مانند یک تنظیم‌گر^{۲۰} عمل می‌کند. همچنین از جمله تنظیم‌گرهای دیگری که در مدل وجود دارد، می‌توان به لایه‌های Batch Normalization میانی اشاره کرد.



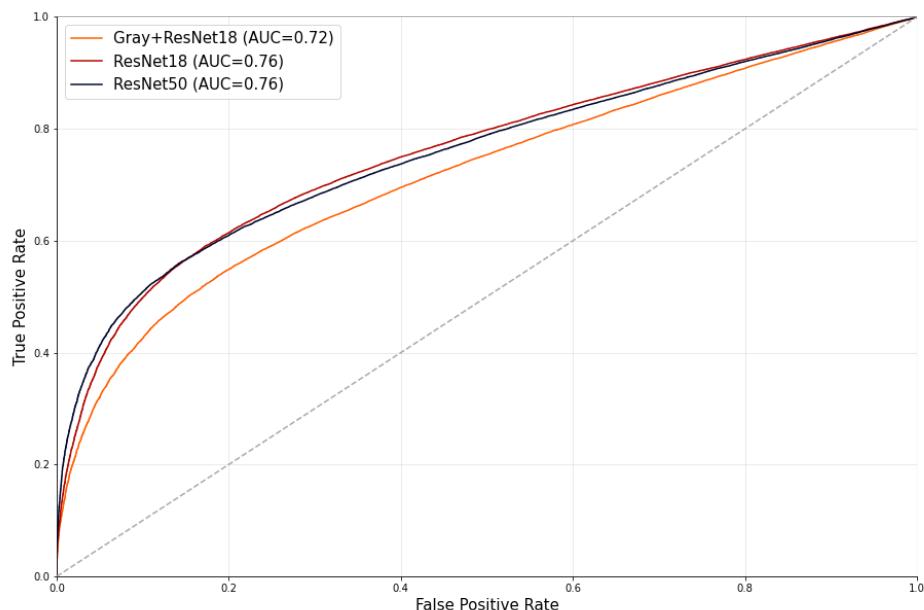
شکل ۱۱- نمودار خطای آموزش و کنترل کیفیت (الف) Gray+ResNet18 و (ب) ResNet18 و (ج) ResNet50

Regularizer^{۲۰}

۱۲.۸.۳ - نمودار ROC

نمودار ROC نموداری است که محور x آن نرخ FP و محور y آن نرخ TP را نشان می‌دهد. این نمودار در مسائل دو کلاسی نشان‌دهنده میزان تمایز دو توزیع کلاس‌های ۰ و ۱ است. هر چه دو توزیع کلاس‌های ۰ و ۱ از یکدیگر متمايز تر باشند، ROC از نیمساز ربع اول بیشتر فاصله می‌گیرد. معمولاً معیاری که برای بیان مقدار تمایز بین دو کلاس بیان می‌کنند، مساحت زیر این نمودار است که آن را AUC می‌نامند. هر چه مقدار AUC به یک نزدیکتر باشد، دو توزیع بهتر از یکدیگر متمايز هستند.

نمودار ROC برای سه شبکه فوق به صورت شکل (۱۱) است. طبق این نمودار به نظر می‌رسد که با سیاه و سفید کردن تصویر ورودی توزیع دو کلاس به یکدیگر نزدیکتر می‌شوند و اگر تصویر را سیاه و سفید نکنیم، وضعیت مدل بهتر است که همانطور که پیشتر توضیح داده شد، این امر قابل درک است و در اینجا به گونه‌ای نمایش داده می‌شود، زیرا با سیاه و سفید کردن تصویر معیارهای تمایز تصویر دو چهره کاهش یافته است. همچنین طبق این معیاری تفاوتی بین دو شبکه ResNet18 و ResNet50 وجود ندارد. البته گفتنی است که زمانی می‌توان گفت که واقعاً فرقی بین آن‌ها وجود ندارد که هر دو مدل متناسب با ظرفیت خود با داده کافی آموزش یابند.



شکل ۱۲ - نمودار ROC برای سه شبکه ResNet15 و ResNet18 و Gray+ResNet18

۱.۲.۸.۴- سایر معیارها

در اینجا خوب است علاوه بر معیارهایی که پیشتر بررسی شدند، کیفیت مدل را با یک سری معیار دیگر هم بسنجیم. در اینجا معیارهای زیر را بررسی می‌کنیم:

Accuracy :Accuracy • در واقع نسبت تشخیص درست خروجی را به ما می‌دهد. استفاده از Accuracy زمانی

مناسب است که داده نامتوازن نباشند. Accuracy به صورت زیر تعریف می‌شود:

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

Recall :Recall • نسبت درست تشخیص دادن در بین داده‌های درست را به ما می‌دهد. این معیار زمانی مناسب است

که از دست دادن یک کلاس صحیح فاجعه‌بار باشد. مثلاً دسته‌بندی سیب‌های سمی و غیرسمی. نحوه محاسبه این معیار به صورت زیر است:

$$Recall = \frac{TP}{TP + FN}$$

Precision :Precision • در واقع می‌گوید که چه نسبتی از داده‌هایی که درست تشخیص داده شده‌اند، واقعاً درست

بوده‌اند. استفاده از این معیار زمانی مناسب است که غلط بودن پیش‌بینی درست، زیان آور باشد. مثلاً پیش‌بینی سرمایه‌گذاری. این معیار به صورت زیر محاسبه می‌شود:

$$Precision = \frac{TP}{TP + FP}$$

F1-Score • زمانی که هم Precision برایمان مهم است، از این معیار استفاده می‌کنیم.

$$F1 - Score = \frac{2}{\frac{1}{Recall} + \frac{1}{Precision}}$$

	class	Precision	Recall	F1-Score	Threshold	Accuracy
Gray+ResNet18	0	0.62	0.79	0.70	0.158	66.9%
	1	0.75	0.56	0.64		
ResNet18	0	0.65	0.83	0.73	0.162	70.3%
	1	0.79	0.59	0.68		
ResNet50	0	0.64	0.85	0.73	0.124	70.2%
	1	0.81	0.56	0.66		

جدول ۵- بررسی معیارهای مختلف بر روی ساختارهای مختلف

۱.۲.۹- داده جدید

از آنجایی که در اینجا یک بردار ویژگی را یاد گرفته‌ایم، با آمدن داده‌ی جدید کافی است، این داده را از شبکه عبور داده و بردار ویژگی آن را با سایر بردارها مقایسه کنیم؛ بنابراین، با آمدن داده‌ی جدید، در این روش نیازمند آموزش مجدد مدل نیستیم.

۱.۲.۱۰- ادامه راه

در ادامه راه می‌توانیم، این تحقیق و پژوهش را بر روی عوامل مختلفی امتحان کنیم. از جمله این عوامل عبارتند از:

- **تابع خطا:** تابع خطایی که در اینجا استفاده کردیم، تابع خطا ArcFace است. در ادامه می‌توانیم توابع خطا دیگر همچون تابع خطای CosFace، SphereFace و یا Triplet را بررسی کرد.
- **بررسی تأثیر عمق:** برای این کار می‌توان از شبکه‌های عمیق‌تری استفاده کرد.
- **بررسی روش‌های اعتبارسنجی آنی^{۲۱}:** یکی از مسائل مهم این است که زمان پردازش تصویر، طولانی نباشد تا بتوان دو تصویر را به سرعت مقایسه کرد. پس افزایش سرعت برایمان مهم است که البته این مورد با عمق شبکه مصالحه دارد به گونه‌ای که برای افزایش سرعت باید تعداد پارامترها یا همان عمق شبکه را کاهش دهیم و این در حالی است که با کاهش عمق شبکه، دقت نیز ممکن است کاهش یابد.

سوال (۲): دسته‌بندی تصاویر

ماشین با Efficient-B0

۲.۱- دسته‌بندی داده

به فرآیندی که در آن تصویر را به صورت یک کل می‌بینیم و به آن تصویر یک برچسب انتساب می‌کنیم، دسته‌بندی تصاویر ۲۲ می‌گویند. به خاطر این که تنها یک برچسب به تصویر نسبت می‌دهیم، معمولاً در این تصاویر تنها یک عکس وجود دارد. در گذشته برای دسته‌بندی تصاویر، از تکنیک‌هایی استفاده می‌شد که به صورت دستی، یک سری ویژگی از تصاویر استخراج می‌کردند و سپس با استفاده از این ویژگی و مکان قرار گرفتن آن‌ها، تصاویر را دسته‌بندی می‌کردند. یا اینکه با استفاده از تشخیص لبه سعی در تشخیص یک شی خاص داشتند. امروزه با رشد شبکه‌های عصبی کانولوشنی، شبکه‌های بسیاری معرفی شده‌اند که نه تنها از روش‌های سنتی بهتر عمل می‌کنند، بلکه از عملکرد انسانی نیز پیشی گرفته‌اند. مشکلی که در شبکه‌های عصبی با آن رویرو هستیم، کمبود داده است؛ چرا که معمولاً فرآیند جمع‌آوری داده، هزینه‌بر و طاقت فرساست. بنابراین، از روشی به نام انتقال تجربه²³ استفاده می‌کنیم.

۲.۲- انتقال تجربه

در روش انتقال تجربه، از یک مدلی که پیشتر بر روی یک مجموعه داده معمولاً بزرگ‌تر، برای همان هدف ما یعنی دسته‌بندی تصاویر آموزش یافته است، به عنوان شبکه پایه‌ای خود در نظر می‌گیریم و تنها لایه نهایی را که وظیفه دسته‌بندی تصاویر را بر عهده دارد، با لایه‌ای مناسب با تعداد دسته‌های مجموعه داده خود، تعویض می‌کنیم. در واقع در این روش از شبکه پایه‌ای برای استخراج ویژگی استفاده می‌کنیم و بعد با یک لایه دسته‌بندی، مسئله دسته‌بندی مورد نظرمان را حل می‌کنیم. با این‌کار از مجموعه داده مبدأ به مجموعه داده مقصد انتقال تجربه کرده‌ایم. در این روش معمولاً از شبکه پایه‌ای را از بین شبکه‌های معروف بر روی پایگاه داده‌های بزرگ نسبتاً متناسب با تصاویر مجموعه داده خودمان، آموزش یافته است استفاده می‌کنیم. یکی از مجموعه داده‌هایی که معمولاً این مدل‌ها بر روی آن آموزش یافته‌اند، مجموعه داده ImageNet است که شامل میلیون‌ها تصویر است.

۲.۳- مجموعه داده

مجموعه داده‌ای که در اینجا استفاده می‌کنیم، مجموعه دارای دو مجموعه مجزای Stanford Car 196 است. این مجموعه داده می‌کنیم، مجموعه دارای دو مجموعه مجزای آموزش و آزمون است که در شکل ؟ یک نمای کلی از این دو مجموعه قابل رؤیت است. برای انتخاب صحیح ابرپارامترهای مدل، لازم است که مجموعه آموزش را به دو مجموعه آموزش و کنترل افزار کنیم. در این افزار، مهم است که توزیع داده‌های مجموعه آموزش و کنترل کیفیت یکسان باشد که به اصطلاح به آن ”Stratification“ می‌گویند. برای این‌کار به ازای هر دسته، ۰.۸ داده‌ها را به مجموعه آموزش و ۰.۲ باقی‌مانده را به مجموعه کنترل کیفیت داده‌ایم. در جدول (۶) تعداد داده‌های موجود در هر یک از این مجموعه‌ها مشخص شده‌است.

مجموعه آموزش	تعداد گروه‌ها	تعداد داده	تعداد تصاویر	میانگین تعداد داده‌های به ازای هر گروه	میانه تعداد داده‌ای به ازای هر گروه
مجموعه کنترل	۱۹۶	۶۵۹۸	۱۴	۳۳.۶۶	۳۴
کیفیت	۱۹۶	۱۵۴۶	۴	۷.۸۸	۸
مجموعه آزمون	۱۹۶	۸۰۴۱	۱۶	۴۱.۰۲	۴۲

جدول ۶ - تعداد داده‌های مجموعه‌های آموزش، کنترل کیفیت و آزمون در 196 Standford Car

Image Classification²²

Transfer Learning²³



شکل ۱۳- نمای کلی از داده‌های (الف) آموزش و (ب) آزمون مجموعه داده Stanford Car 196

۲.۳.۱- پیش پردازش داده‌ها

داده‌های مجموعه آموزش و آزمون به طور مجزا در دو دایرکتوری متفاوت وجود دارند که هر دایرکتوری شامل تصاویری با نام‌های مجزاست. همچنین این مجموعه شامل دو فایل مربوط به برچسب‌گذاری داده‌های داده‌های آموزش و آزمون مجموعه داده است. برای راحتی در این مرحله با خواندن این فایل‌ها، دو فایل “train.csv” و “test.csv” به نام‌های “train.csv” و “test.csv” ذخیره کرده‌ایم تا در گام بعد با خواندن این فایل به راحتی به داده‌ها دسترسی داشته باشیم. همچنین در این گام داده‌های آموزش و کنترل کیفیت را از یکدیگر جدا کرده‌ایم. این جداسازی بدین صورت است که در فایل “train.csv” یک ستون به نام “train/val” وجود دارد که مقدار آن یا

است و یا ”val“ که به ترتیب نشان‌دهنده تعلق آن فایل به مجموعه آموزش یا کنترل کیفیت دارد. همانطور که پیشتر ذکر شده در این جداسازی ۰.۰ داده‌های هر کلاس به مجموعه آموزش و ۰.۲ داده‌های هر کلاس به مجموعه کنترل کیفیت تخصیص یافته است.

۲.۴- بارگذاری داده

Data Loader داده‌ها را به صورت زوج تصویر و برچسب مربوط به آن بر می‌گرداند. همچنین برای افزایش قدرت مدل یک سری تبدیل‌هایی به این تصاویر قبل از بارگذاری اعمال می‌کنیم که برای داده‌های آموزش و کنترل کیفیت که این تبدیل‌ها به ترتیب در جدول‌های (۷) و (۸) قابل مشاهده است. تمام تبدیل‌هایی که بر روی داده‌های آموزش و کنترل کیفیت اعمال شده‌اند، تبدیل‌های موجود در `torchvision.transforms` هستند.

`transforms.Resize(input_width, input_height)` تصویر ورودی به اندازه مشخصی تبدیل می‌کنیم. اندازه‌ای که در اینجا استفاده شده است، 112×112 است.

`transforms.RandomHorizontalFlip(p)` با احتمال p تصویر را به صورت افقی معکوس می‌کند.
`transforms.RandomRotation(degrees)`

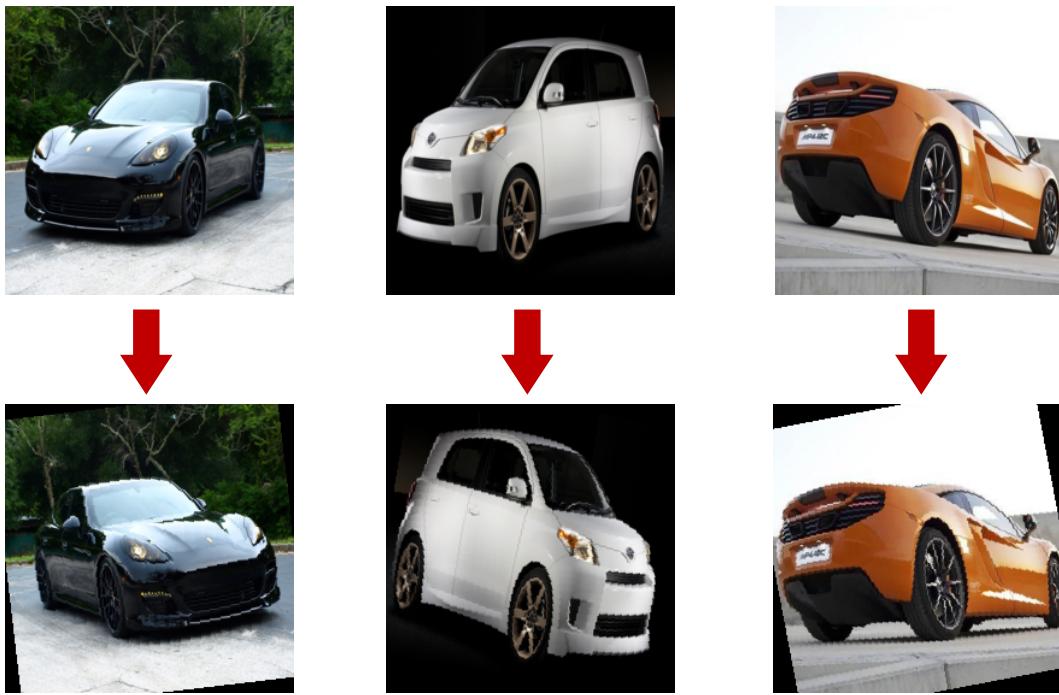
`transforms.ToTensor()` تصویر ورودی را به `torch.Tensor` تبدیل می‌کند.
`transforms.Normalize(mean, std)` تنسر ورودی را با میانگین و انحراف از معیار مشخص، نرمال می‌کنیم.
میانگین و انحراف از معیاری که در اینجا در نظر گرفته‌ایم، با توجه به داده‌های آموزش بوده است که مقدار آن به شرح زیر است:
$$\begin{cases} \mu = [0.471, 0.460, 0.455] \\ \sigma = [0.267, 0.266, 0.271] \end{cases}$$

جدول ۷- تبدیل‌های اعمال شده بر داده‌های آموزش Stanford Car 196

`transforms.Resize(input_size)` تصویر ورودی به اندازه مشخصی تبدیل می‌کنیم. اندازه‌ای که در اینجا استفاده شده است، 112×112 است.

`transforms.ToTensor()` تصویر ورودی را به `torch.Tensor` تبدیل می‌کند.
`transforms.Normalize(mean, std)` تنسر ورودی را با میانگین و انحراف از معیار مشخص، نرمال می‌کنیم.
میانگین و انحراف از معیاری که در اینجا در نظر گرفته‌ایم، همان میانگین و انحراف از معیاری است که برای داده‌های آموزش هم استفاده شده است.

جدول ۸- تبدیل‌های اعمال شده بر داده‌های کنترل کیفیت Stanford Car 196



شکل ۱۴ - اعمال تبدیلهای آموزش بر تعدادی از تصاویر

۲.۵ - شبکه

شبکه‌ای که در اینجا استفاده می‌کنیم، شبکه EfficientNet-B0 است که بر روی مجموعه داده ImageNet آموزش یافته است. تنها تغییری که در این شبکه ایجاد می‌کنیم، این است که لایه پایانی آن که وظیفه دسته‌بندی را بر عهده دارد، تغییر داده و به جای آن یک معماری متناسب با مجموعه داده خود قرار می‌دهیم. این معماری پایانی می‌تواند شامل یک یا چند لایه باشد. طبق بررسی‌هایی که انجام شد، در زمانی که وزن‌های شبکه پایه ثابت هستند، در صورتی که مدل تنها دارای یک لایه باشد، دقت مدل بر روی داده‌های کنترل کیفیت از ۶۱٪ بالاتر نمی‌رود. علت این دقت پایین این است که از آنجایی که وزن‌های شبکه پایه ثابت نگه داشته شده است، مدل برای دسته‌بندی تصاویر، از ویژگی‌هایی که از ImageNet آموخته است، استفاده می‌کند که برای جداسازی این ویژگی‌ها تنها یک لایه مناسب نیست و خطای بایاس مدل بسیار بالا می‌شود. اما اگر وزن‌های شبکه پایه را قابل آموختن قرار دهیم، به قدری این مشکل برطرف می‌شود ولی همچنان می‌توان گفت که خطای بایاس مدل بسیار بالاست. بدین خاطر، لایه دسته‌بندی پایانی را یک معماری سه لایه به همراه Dropout و BatchNormalization قرار دادیم. در این سه لایه، ابتدا ویژگی‌های بدست آمده از مدل پایه را به فضایی نصف قبل و پس از آن به فضایی ربع فضای ویژگی اولی، نگاشت کردیم. در نهایت نیز یک لایه برای دسته‌بندی متناسب با تعداد دسته‌های موجود در مجموعه داده، یعنی ۱۹۶ دسته قرار دادیم.

۲.۶ - تابع خطأ

تابع خطای که در اینجا استفاده کردیم، تابع خطای Label Smoothing Cross Entropy است. این تابع برای زمانی که با یک مسئله دسته‌بندی چندین کلاسی سروکار داریم، مناسب است. این تابع به جای این که سعی کند تا مدل مقدار ۱ را برای دسته درست و مقدار ۰ را برای دسته‌های نادرست تخمین بزنند، سعی در این دارد که مقدار ϵ -۱ را برای دسته درست و مقدار ϵ را برای

دسته نادرست پیش‌بینی کند و با این کار به گونه‌ای اثر تنظیم‌کنندگی^{۲۴} دارد و از بیش برآش^{۲۵} مدل جلوگیری می‌کنیم. این تابع به صورت زیر تعریف می‌شود:

$$\text{LabelSmoothingCrossEntropy} = (1 - \epsilon)ce(i) + \epsilon \sum_{j \neq i} \frac{ce(j)}{N}$$

در تعریف بالا، $ce(\cdot)$ همان Cross Entropy متعارف است. همچنین در تعریف فوق N ، همان تعداد کلاس‌هاست.

۲.۷- بهینه‌سازی و نرخ یادگیری

برای بهینه‌سازی از الگوریتم Adam استفاده کردہ‌ایم. در اینجا علاوه بر این که سعی کردہ‌ایم با استفاده از تابع خطای خطا، به صورت صریح از بیش برآش مدل جلوگیری کنیم، همچنین با استفاده از برنامه‌ریز^{۲۶} نرخ یادگیری به صورت ضمنی سعی در کاهش مقدار تابع خطای داریم. برنامه‌ریزی که در اینجا استفاده کردہ‌ایم، ReduceLROnPlateau است که به صورت زیر نمونه‌گیری شده‌است:

```
optimizer=torch.optim.lr_scheduler.ReduceLROnPlateau (   
    optimizer,  
    mode='max',  
    patience=3,  
    threshold=0.9,  
    min_lr=1e-6  
    verbose=True  
)
```

کد ۲- استفاده از برنامه ریز ReduceLROnPlateau برای آموزش شبکه EfficientNet-B0

در این برنامه ریز اگر به مدت ۳ دوره آموزش مقدار تابع خطای تغییر زیادی نکند، نرخ یادگیری را به مقدار ۱.۰ مقدار قبلی کاهش می‌دهیم.

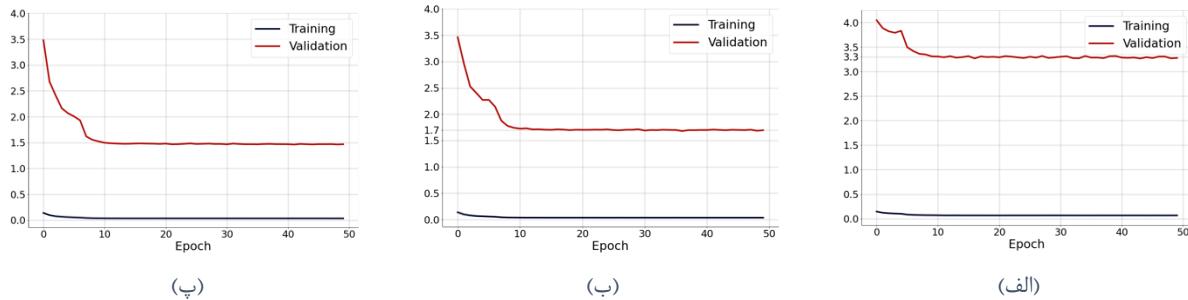
۲.۸- نتایج

با توجه به شکل (۱۵) با ثابت نگاه داشتن، وزن‌های شبکه پایه، همگرایی مدل به شدت سخت‌تر شده است. چرا که از یک شبکه ۳ لایه برای دسته بندی ویژگی‌های تصویر استفاده کردہ‌ایم که در این دسته بندی با توجه به پیچیدگی پایین مدل، خطای بایاس به شدت کم است و اگر وزن‌های تنها لایه‌های پایانی، مدل پایه را قابل آموختن قرار دهیم، پیچیدگی مدل بیشتر شده و خطای بایاس کاهش می‌یابد. البته لازم به ذکر است که با افزایش پیچیدگی، خطای واریانس نیز افزایش می‌یابد ولی این افزایش واریانس در مقابل کاهش بایاس بسیار ناجیز است. هنگامی که مدل پایه را به طور کامل قبل آموختن قرار می‌دهیم، افزایش مدل بیشتر و خطای بایاس کاهش و خطای واریانس افزایش می‌یابد که این بار هم اگرچه میزان کاهش بایاس از افزایش واریانس بیشتر بوده است اما نسبت به

Regularization^{۲۴}

Overfitting^{۲۵}

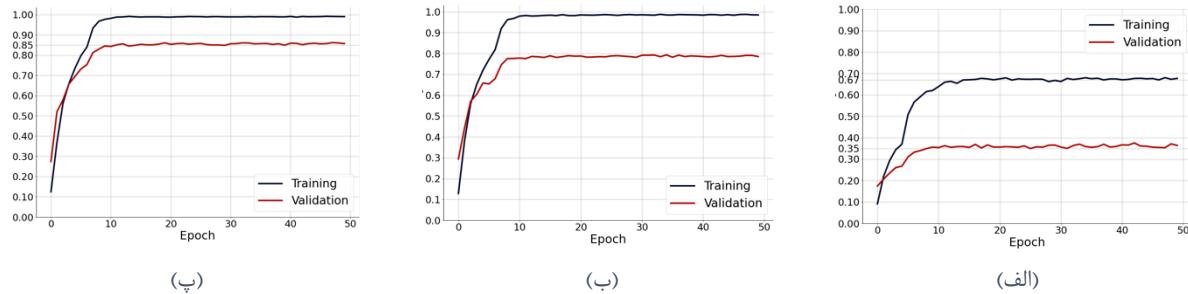
Scheduler^{۲۶}



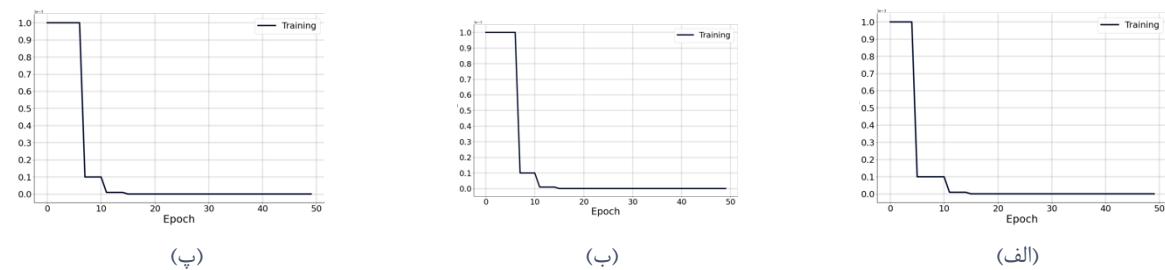
شکل ۱۵- نمودار تابع خطا در (الف) شبکه پایه با پارامترهای ثابت، (ب) شبکه پایه با پارامترهای نسبتاً ثابت (لایه ۶ و ۷ و ۸ غیرثابت) و (پ) شبکه پایه با پارامترهای غیرثابت

حالت قبل تفاوت چندانی نداشته‌ایم. لازم به ذکر است که خطای آموزش تقریباً به صفر رسیده است و با توجه به این که نرخ یادگیری نیز در دوره‌های پایانی به شدت کم است، تقریباً می‌توان گفت که هر سه مدل همگرا شده‌اند. با توجه به نمودار دقت در شکل (۱۶) نیز می‌توان به استدلال‌های فوق رسید. از آنجایی که مدل اول که در آن وزن‌های شبکه پایه غیر قابل-آموختن بودند، مدلی نسبتاً ضعیف برای مجموعه داده‌ما است، این مدل به یک بهینه کم کیفیت^{۲۷} رسیده است اما در حالت دوم و سوم که به ترتیب لایه‌های پایانی و در نهایت تمام لایه‌های را قابل آموختن کردیم، مشاهده می‌شود که اگرچه دقت بر روی داده‌های کنترل کیفیت حدود ۵٪ متفاوت است اما در مقابل حالت اول آن قدر چشمگیر نیست.

لازم به ذکر است که در این آزمایش همانطور که پیشتر گفته شده از برنامه‌ریز ReduceLROnPlateau برای تغییر مناسب نرخ آموزش استفاده شده است. تغییرات نرخ یادگیری در دوره‌های آموزش مختلف، در شکل (۱۷) قابل مشاهده است.



شکل ۱۶- نمودار دقت در (الف) شبکه پایه با پارامترهای ثابت، (ب) شبکه پایه با پارامترهای نسبتاً ثابت (لایه ۶ و ۷ و ۸ غیرثابت) و (پ) شبکه پایه با پارامترهای غیرثابت



شکل ۱۷- نمودار تغییرات نرخ یادگیری در (الف) شبکه پایه با پارامترهای ثابت، (ب) شبکه پایه با پارامترهای نسبتاً ثابت (لایه ۶ و ۷ و ۸ غیرثابت) و (پ) شبکه پایه با پارامترهای غیرثابت

دقت داده‌های ازمون	
ثابت	%۳۵.۲۸
نسبتاً ثابت	%۸۰.۰۴
غیر ثابت	%۸۵.۰۳

جدول ۹- دقتهای ازمون بر روی سه ساختار با شبکه‌های پایه ثابت، نسبتاً ثابت و غیر ثابت

۲.۹- نتیجه‌گیری:

با توجه به نتایجی که در این پژوهه به دست آمد، می‌توان این‌گونه استدلال کرد که برای آموزش یک شبکه با استفاده از روش انتقال تجربه، ثابت نگه داشتن شبکه پایه چندان مناسب نیست و بهتر است لایه‌های پایانی شبکه پایه را قابل آموختن قرار دهیم که این کار علاوه بر اینکه پیچیدگی مدل را افزایش می‌دهد، همچنین باعث می‌شود که مدل ویژگی‌های متناسب با مجموعه داده در دست را یاد بگیرد. اگر تمام لایه شبکه پایه را قابل آموختن قرار دهیم، پیچیدگی مدل به بیشینه مقدار ممکن خواهد رسید و خطای بایاس به شدت کاهش خواهد یافت. در اینجا نباید خطای واریانس را از یاد برد، چرا که با افزایش پیچیدگی، این خطای افزایش می‌یابد و امکان بیش‌برازش مدل وجود خواهد داشت. بنابراین، توصیه می‌شود که عمقی را که از انتهای شبکه پایه قابل آموختن قرار می‌دهیم، متناسب با اندازه مجموعه داده انتخاب شود. زمانی که تعداد داده‌های موجود در مجموعه برای هر دسته کم است، این عمق را کم انتخاب کنیم و اگر داده، به اندازه کافی وجود داشته باشد، تمام شبکه پایه را قابل آموختن قرار دهیم.

۲.۱۰- مصورسازی شبکه:

در شکل (۱۸) با مصورسازی‌های مختلف خروجی شبکه قابل بررسی است.



شکل ۱۸- مصورسازی‌های مختلف داده ورودی بعد از آموزش مدل بر روی سه مدل با شبکه پایه EfficientNet-B0 با (الف) پارامترهای ثابت، ب) پارامترهای نسبتاً ثابت و ج) پارامترهای غیر ثابت

خوداظهاری

در این پروژه در دو قسمت پیاده‌سازی کد و گزارش نویسی از منابع زیر به طور عمدۀ استفاده شده است:

کد سوال ۱: ●

○ پیاده‌سازی تابع خطا:

https://github.com/TreB1eN/InsightFace_Pytorch ■

کد سوال ۲: ●

○ پیاده‌سازی تابع خطا:

<https://github.com/matkalinowski/EfficientNet-on-Stanford-Cars-Dataset> ■

گزارش نویسی: ●

<https://www.baeldung.com/cs/euclidean-distance-vs-cosine-similarity> ○

<https://medium.com/@mygreatlearning/what-is-facial-recognition-technology-and-how-does-it-work-9fdefd335c3b> ○

<https://towardsdatascience.com/explaining-precision-vs-recall-to-everyone-295d4848edaf> ○

<https://medium.com/dataseries/how-does-resnet-improve-performance-caaa436f885b> ○

<https://towardsdatascience.com/an-overview-of-resnet-and-its-variants-5281e2f56035> ○

<https://towardsdatascience.com/roc-curve-and-auc-from-scratch-in-numpy-visualized-2612bb9459ab> ○

“ArcFace: Additive Angular Margin Loss for Deep Face Recognition”, Jiankang Deng, Jia Guo, Niannan Xue, Stefanos Zafeiriou, <https://arxiv.org/abs/1801.07698> ○

همچنین مدل‌های آموزش یافته در لینک‌های زیر موجود است:

سوال ۱: ●

https://drive.google.com/drive/folders/1ddwYg_UNAZ3DH6xh7VrrOKV5PapHZPx?usp=sharing

سوال ۲: ●

https://drive.google.com/drive/folders/1m7JbJjoth2_7C6-HT8GS9RYZ1_U5W-8C?usp=sharing