اعضاء گروه: فاطمه جمالو
تاریخ گزارش: ۲۴ تیر ۱۴۰۲

عنوان گزارش
گزارش نهایی پروژه درس یادگیری ژرف

۱ مقدمه

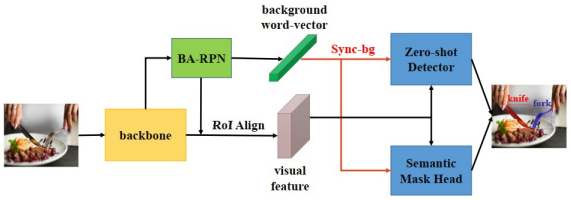
با افزایش حجم داده‌ها و پیچیدگی مسائل، روش‌های یادگیری بدون نمونه مانند (Zero-Shot Learning) به عنوان یک راه حل جذاب در حوزه‌ی یادگیری ماشین به شدت مورد توجه قرار گرفته‌اند. در این روش، هدف یادگیری استفاده از دانش موجود در مورد کلاس‌های دیده‌نشده برای تشخیص شیء است. یکی از روش‌های معروف برای یادگیری بدون نمونه، استفاده از Contrastive Learning است. در این روش، دو نمونه ورودی با هم مقایسه شده و برای آن‌ها یک بردار مشترک به عنوان نماینده تولید می‌شود. در این مقاله، یک روش زیبا و جدید برای یادگیری بدون نمونه با استفاده از Patch-wise Semantic و Greedy Infomaxارائه می‌شود. در این روش، از دانش موجود درباره یک کلاس دیده نشده در هنگام آموزش استفاده می‌شود و از روی آن، یک نمایش فضایی برای شیء ساخته می‌شود. سپس با استفاده از یادگیری مبتنی بر Contrastive Learning و Greedy Infomax، بردارهای نماینده بهبود داده می‌شوند و به شبکه اجازه می‌دهند که با دقت بیشتری برای تشخیص شیء استفاده شود. این روش با توجه به نتایج به دست آمده، می‌تواند راه‌حلی موثر برای یادگیری بدون نمونه باشد.

۲ مفاهیم و تعاریف پایه

۱.۲ یادگیری بدون نمونه (Zero-Shot Learning):

یکی از دلایلی که مدل‌های یادگیری ماشین به طور کلی هوشمندتر می‌شوند، وابستگی آن‌ها به استفاده از داده‌های برچسب‌خورده برای کمک به آن‌ها در تمایز بین دو شیء مشابه است. بدون این مجموعه داده‌های برچسب‌خورده، ایجاد مدل یادگیری ماشینی که بسیار کارآمد و قابل اعتماد باشد با موانع عمده‌ای روبرو خواهد شد. مجموعه داده‌های برچسب‌خورده در مرحله آموزش مدل بسیار مهم هستند. یادگیری عمیق به طور گسترده برای حل وظایفی مانند بینایی ماشین با استفاده از یادگیری نظارت شده صورت میگیرد. با این حال، مانند بسیاری از چیزهای دیگر در زندگی، با محدودیت‌هایی همراه است. دسته‌بندی نظارت شده نیاز به تعداد زیادی داده آموزشی برچسب‌خورده با کیفیت بالا دارد تا مدل قوی و قابل اعتمادی تولید کند. این بدان معناست که مدل دسته‌بندی نمی‌تواند با کلاس‌های ناشناخته کار کند. بدست آوردن مقادیر بالای داده برچسب‌گذاری‌شده با کیفیت بالا، سخت است و وقتی کلاس‌های بی‌شماری (به عنوان مثال گونه‌های حیوانات) وجود دارد که مدل باید آن‌ها را یاد بگیرد، کمکی نمی‌کند. یکی از راه‌های حل مشکل کاهش وابستگی مدل‌ها به داده‌های برچسب‌گذاری‌شده است. این انگیزه پشت یادگیری بدون نمونه وجود دارد، در آن مدل شما یاد می‌گیرد که چگونه کلاس‌هایی که پیش‌تر ندیده را دسته‌بندی کند. یادگیری بدون نمونه الگویی در یادگیری ماشین است که در آن یک مدل یادگیری عمیق پیش‌آموزش دیده شده است تا برای یک دسته از نمونه‌ها تعمیم داده شود. ایده پشت یادگیری بدون نمونه این است که چگونه انسان‌ها می‌توانند به طور طبیعی شباهت‌هایی بین کلاس‌های داده‌ای پیدا کنند و به همان شکل، ماشین را به شناسایی آن‌ها آموزش دهد. هدف اصلی یادگیری بدون نمونه ، به دست آوردن توانایی پیش‌بینی نتایج بدون هیچ نمونه آموزشی است؛ در حین آموزش، ماشین باید اشیاء را از کلاس‌هایی که در آموزش آن‌ها آموزش دیده نشده‌اند، شناسایی کند. یادگیری بدون نمونه بر اساس انتقال دانشی است که در نمونه‌های آموزشی وجود دارد، انجام می‌شود. یادگیری بدون نمونه برای یادگیری لایه‌های معنایی و ویژگی‌های واسطه‌ای پیشنهاد شده است، سپس آن را به پیش‌بینی یک کلاس جدید از داده‌های ناشناخته اعمال می‌کند. در مثال دسته‌بندی گونه‌های حیوانات، مدل شما ممکن است بتواند پیش‌بینی کند که تصویر در گوشه پایین سمت راست یک ”پاندا” است، حتی اگر در طول آموزش به صورت صریح یک مثال برچسب‌گذاری‌شده از ”پاندا”

ندیده باشد. در یادگیری بدون نمونه ، داده‌ها شامل موارد زیر می‌شوند: کلاس‌های دیده شده: این کلاس‌ها شامل کلاس‌هایی هستند



شکل ۱: کل معماری چارچوب تقسیم‌بندی نمونه صفر شات

که در طول آموزش، تصاویر برچسب‌گذاری شده برای آن‌ها وجود دارد. کلاس‌های ناشناخته یا دیده نشده: این کلاس‌ها شامل کلاس‌هایی هستند که در طول مرحله آموزش، تصاویر برچسب‌گذاری شده برای آن‌ها وجود ندارد. اطلاعات کمکی: این اطلاعات شامل توصیفات/ویژگی‌های معنایی/تعبیه کلمات برای هر دو کلاس دیده شده و ناشناخته در زمان آموزش است. این اطلاعات عملکرد پلی بین کلاس‌های دیده شده و ناشناخته را ایفا می‌کند (همانطور که در پست بعدی خواهیم دید). برای درک وضعیت مسئله در یادگیری صفر–شات، به معرفی یک نمادگذاری رسمی می‌پردازم. فرض کنید (S) مجموعه‌ای از داده‌های کلاس‌های دیده شده باشد. S شامل چندین تصویر (هر تصویر با (X) نمایش داده می‌شود)، برچسب‌های مرتبط (با (Y) نمایش داده می‌شوند) و اطلاعات کمکی (با (hy) نمایش داده می‌شوند) است. بردار hy به عنوان کدگذاری معنایی برای کلاس Y ارجاع داده میشود چون توصیف کلاس Y را با کلمات انجام میدهد. بردار دودویی به عنوان بردار ویژگی شناخته می‌شود و برای کدگذاری معنایی به طور گسترده ای استفاده می‌شود. با این حال، می‌توان کدگذاری‌های معنایی دیگری را نیز داشت. علاوه بر این، به صورت ریاضی می‌توانیم بگوییم $y \in Y^S$ که به معنی این است که برچسب کلاس می‌تواند یکی از کلاس‌های دیده شده (Y^S) مانند "گربه"، "سگ" و غیره باشد. به طور مشابه، $x \in X^S$ است که X^S مجموعه تمام تصاویر دیده شده است و $h \in A^S$ است که A^S مجموعه ویژگی‌های دیده شده است. این به صورت ریاضی به شکل زیر نوشته می‌شود:

$$S = \{(X, Y, hY) | x \in X^S, y \in Y^S, h \in A^S\} \quad (۱)$$

مجموعه‌ای از داده‌های دیده‌شده شامل تصویر X، برچسبYو رمزگذاری معناییhyرا دارد. تصویر یکی از تصاویر موجود در مجموعه X^S، برچسب یکی از برچسب‌های موجود در مجموعه Y^S و رمزگذاری از مجموعه A^S است. همچنین، مجموعه‌ای از داده‌های دسته‌ها با (U) نمایش داده می‌شود. در مرحله آموزش از داده‌های تصویری برچسب‌گذاری شده از دسته‌های دیده‌شده، به عبارت دیگر مجموعه (S)، استفاده می‌شود. مجموعه U از تک‌تک رکوردهای زیر تشکیل شده است:

$$U = \{(x, y, hy) | x \in X^U, y \in Y^U, h \in A^U\} \quad (۲)$$

در اینجا، X^U مجموعه‌ی تصاویر با تصاویر نامشخص است، Y^U کلاس برچسب‌های نامشخص و A^U مجموعه‌ی ویژگی‌های کلاس‌های نامشخص است. در مرحله آموزش نیز به توصیفات/صفات معنایی از هر دو دسته دیده‌شده و ندیده‌شده، به عبارت دیگر مجموعه $A^S \cup A^U$ دسترسی وجود دارد. یادگیری بدون داده آموزشی ترانسدوکتیو در حالت ترانسدوکتیو، علاوه بر داده‌های تصویری برچسب‌گذاری شده از دسته‌های دیده‌شده (یعنی مجموعه(S))، در مرحله آموزش نیز به تصاویر بدون برچسب از دسته‌های ندیده‌شده، یعنی مجموعه X^U، دسترسی برای مدل وجود دارد. همانند حالت استنتاجی، در مرحله آموزش نیز به توصیفات/صفات معنایی از هر دو دسته دیده‌شده و ندیده‌شده، به عبارت دیگر مجموعه $A^S \cup A^U$ دسترسی وجود دارد.یادگیری بدون داده آموزشی سنتی:در یادگیری بدون داده آموزشی سنتی، تصاویری که در زمان آزمون باید شناسایی شوند، فقط متعلق به دسته‌های ندیده‌شده، به عبارت دیگر $(\text{test classes}) \in Y^U$ هستند.یادگیری بدون داده آموزشی تعمیم‌یافته:در یادگیری بدون داده آموزشی تعمیم‌یافته، تصاویری که در زمان آزمون باید شناسایی شوند، ممکن است متعلق به دسته‌های دیده‌شده یا ندیده‌شده باشند، به عبارت دیگر $(\text{test classes}) \in Y^S \cup Y^U$ هستند.[۱]

۲.۲ یادگیری انتقالی(transfer learning):

شبکه‌های عصبی رفتار خود را به صورت سلسله مراتبی توسعه می‌دهند. هر شبکه عصبی از چندین لایه تشکیل شده است. پس از آموزش، هر یک از لایه‌ها برای شناسایی ویژگی‌های خاص در داده ورودی تنظیم می‌شوند. به عنوان مثال، در یک شبکه پیچشی

برای تشخیص تصویر، لایه‌های ابتدایی ویژگی‌های کلی مانند لبه‌ها، گوشه‌ها، دایره‌ها و گلوبول‌های رنگ را شناسایی می‌کنند. با عمق شدن در شبکه، لایه‌ها شروع به شناسایی چیزهای بیشتر واقعی مانند چشم، صورت و اشیاء کامل می‌کنند. در هنگام انتقال یادگیری، مهندسان هوش مصنوعی لایه‌های اول شبکه عصبی پیش‌آموزش‌دیده را منجمد می‌کنند. این لایه‌ها ویژگی‌های کلی را که در تمام دامنه‌ها مشترک هستند، شناسایی می‌کنند. سپس لایه‌های عمیق‌تر را برای تنظیم دقیق با مثال‌های خودشان و اضافه کردن لایه‌های جدید برای طبقه‌بندی دسته‌بندی‌های جدیدی که در مجموعه داده آموزشی آنها وجود دارد، فاین‌تیون می‌کنند. مدل‌های هوش مصنوعی پیش‌آموزش‌دیده و فاین‌تیون شده به ترتیب مدل‌های ''معلم'' و ''دانش‌آموز'' نیز خوانده می‌شوند. تعداد لایه‌های منجمد و فاین‌تیون شده بستگی به شباهت بین مدل‌های هوش مصنوعی مبدأ و مقصد دارد. اگر مدل هوش مصنوعی مقصد یک مسئله را حل کند که بسیار نزدیک به مدل مبدأ است، نیازی به فاین‌تیون لایه‌های مدل پیش‌آموزش‌دیده نیست. تنها کافیست یک لایه جدید به پایان شبکه اضافه شود و هوش مصنوعی برای دسته‌بندی دسته‌های جدید آموزش داده شود. این به عنوان ''استخراج ویژگی لایه‌های عمیق'' شناخته می‌شود. استخراج ویژگی‌های عمیق، همچنین در صورت وجود تعداد کمی از داده‌های آموزشی برای دامنه مقصد، ترجیح داده می‌شود. در صورتی که تفاوت‌های قابل توجهی بین مدل‌های هوش مصنوعی مبدأ و مقصد وجود دارد یا مثال‌های آموزشی فراوان هستند، مهندسان لایه‌های چندگانه از مدل هوش مصنوعی پیش‌آموزش‌دیده را بازمی‌کنند. سپس لایه طبقه‌بندی جدید را اضافه کرده و لایه‌های بازمی‌کنده را با مثال‌های جدید فاین‌تیون می‌کنند. این به عنوان ''استخراج ویژگی لایه‌های میانی'' شناخته می‌شود. در مواردی که تفاوت‌های قابل توجهی بین مدل‌های هوش مصنوعی مبدأ و مقصد وجود داردر صورتی که تفاوت‌های قابل توجهی بین مدل‌های هوش مصنوعی مبدأ و مقصد وجود دارد، مهندسان کل شبکه عصبی را بازمی‌کنند و مجدداً آموزش می‌دهند. این نوع انتقال یادگیری به عنوان ''فاین‌تیون کامل مدل'' شناخته می‌شود و نیاز به تعداد زیادی مثال آموزشی دارد. ممکن است به نظر برسد که گرفتن یک مدل پیش‌آموزش‌دیده و دوباره آموزش دادن تمام لایه‌های آن، بی‌معنی است. اما در عمل، این کار زمان و منابع محاسباتی را صرفه‌جویی می‌کند. قبل از آموزش، متغیرهای یک شبکه عصبی با اعداد تصادفی مقداردهی اولیه می‌شوند و با پردازش داده‌های آموزشی مقدارهایشان را تطبیق می‌دهند. مقادیر متغیرهای یک شبکه عصبی پیش‌آموزش‌دیده به میلیون‌ها مثال آموزشی تنظیم شده‌اند. بنابراین، آنها برای یک مدل هوش مصنوعی جدید که می‌خواهد روی مجموعه‌ای جدید از مثال‌هایی که حتی کمترین شباهتی با مدل هوش مصنوعی مبدأ دارند، نقطه شروعی بهتری هستند. انتقال یادگیری مشکلات زیادی را از بین می‌برد و به روشی کارآمد و هزینه‌بر برای آموزش مدل‌های هوش مصنوعی می‌پردازد. با این حال، این روش نیز مشکلاتی دارد. اگر یک شبکه عصبی پیش‌آموزش‌دیده دارای آسیب‌پذیری‌های امنیتی باشد، مدل‌های هوش مصنوعی که از آن به عنوان پایه برای انتقال یادگیری استفاده می‌کنند، این آسیب‌پذیری‌ها را به ارث خواهند برد. برای مثال، یک مدل پایه ممکن است در برابر حملات تهاجمی قابل اطمینان نباشد، که مثال‌های ورودی با دقت بالا و خاصیت‌های خاصی دارند که باعث تغییر رفتار هوش مصنوعی به شکل نامنظم می‌شوند. اگر یک فرد بدخواه، یک مثال تهاجمی برای یک مدل پایه توسعه دهد، حمله‌اش روی بیشتر مدل‌های هوش مصنوعی که از آن مشتق شده‌اند، کار خواهد کرد. محققان دانشگاه شیکاگو، دانشگاه سانتا کلارا و دانشگاه ویرجینیا تک، این موضوع را در یک مقاله ارائه شده در کنفرانس Usenix) (Security Symposiumدر سال گذشته نشان دادند. علاوه بر این، در برخی از حوزه‌ها، مانند آموزش هوش مصنوعی برای بازی‌ها، استفاده از انتقال یادگیری بسیار محدود است. این مدل‌های هوش مصنوعی بر روی یادگیری تقویتی آموزش می‌بینند، یک شاخه از هوش مصنوعی که بسیار پردازش محاسباتی دارد و نیاز به تلاش و خطا دارد. در یادگیری تقویتی، بیشتر مسائل جدید منحصر به فرد هستند و نیاز به مدل هوش مصنوعی و فرایند آموزش خودشان دارند. اما در کل، برای بیشتر برنامه‌های یادگیری عمیق، مانند طبقه‌بندی تصویر و پردازش زبان طبیعی، احتمالاً می‌توانید با یک دوز خوب از انتقال یادگیری هوشمند، مسیر خود را به سرعت پیش ببرید.[۲]

۳.۲ یادگیری نیمه نظارتی(Semi-Supervised Learning):

یادگیری نیمه‌نظارتی یک نوع یادگیری ماشینی است که در بین یادگیری نظارت شده و بدون نظارت قرار می‌گیرد. این روش از مقدار کمی از داده‌های برچسب‌گذاری شده و مقدار زیادی از داده‌های بدون برچسب برای آموزش مدل استفاده می‌کند. هدف یادگیری نیمه‌نظارتی یادگیری یک تابع است که بتواند بر اساس متغیرهای ورودی، متغیر خروجی را با دقتی بالا پیش‌بینی کند، به شکلی مشابه با یادگیری نظارت شده. با این حال، برخلاف یادگیری نظارت شده، الگوریتم بر روی مجموعه‌داده ای آموزش

داده می‌شود که حاوی داده‌های همراه با برچسب و بدون برچسب است. یادگیری نیمه‌نظارتی به خصوص زمانی مفید است که مقدار زیادی داده بدون برچسب در دسترس است، اما برچسب‌گذاری آنها گران‌قیمت یا دشوار است. فرضیاتی که الگوریتم‌های یادگیری نیمه‌نظارتی دنبال می‌کنند عبارتند از: فرضیه پیوستگی: الگوریتم فرض می‌کند که نقاطی که به یکدیگر نزدیک‌تر هستند، احتمالاً برچسب خروجی یکسانی دارند. فرضیه خوشه‌بندی: داده‌ها به خوشه‌های گسسته تقسیم می‌شوند و نقاطی که در یک خوشه هستند، احتمالاً برچسب خروجی مشترکی دارند. فرضیه منیفولد: داده‌ها تقریباً در یک منیفولد با ابعاد کمتر از فضای ورودی قرار دارند. این فرضیه امکان استفاده از فواصل و چگالی‌های تعریف شده بر روی منیفولد را فراهم می‌کند. یکی از مشکلات اساسی هر الگوریتم یادگیری نظارت شده این است که مجموعه‌داده باید به صورت دستی توسط یک مهندس یادگیری ماشین یا یک دانشمند داده برچسب‌گذاری شود. این فرآیند بسیار گران است، به ویژه زمانی که با حجم بزرگی از داده‌ها سر و کار داریم. همچنین، مشکل اساسی هر الگوریتم یادگیری بدون نظارت، محدود بودن طیف کاربردی آن است. برای مقابله با این مشکلات، مفهوم یادگیری نیمه‌نظارتی معرفی شده است. در این نوع یادگیری، الگوریتم بر روی ترکیبی از داده‌های برچسب‌گذاری شده و بدون برچسب آموزش داده می‌شود. به طور معمول، این ترکیب شامل مقدار بسیار کمی از داده‌های برچسب‌گذاری شده و مقدار بسیار زیادی از داده‌های بدون برچسب است. روش اصلی در این نوع یادگیری این است که ابتدا برنامه‌نویس با استفاده از یک الگوریتم یادگیری بدون نظارت، داده‌های مشابه را خوشه‌بندی می‌کند و سپس از داده‌های برچسب‌گذاری شده موجود برای برچسب‌گذاری بقیه داده‌های بدون برچسب استفاده می‌کند. موارد کاربرد معمول این نوع الگوریتم‌ها، ویژگی مشترکی با یکدیگر دارند و آن این است که دریافت داده‌های بدون برچسب نسبتاً ارزان است، در حالی که برچسب‌گذاری این داده‌ها بسیار گران است.[۳]

۳ روش پیشنهادی

۱.۳ تعریف مسئله:

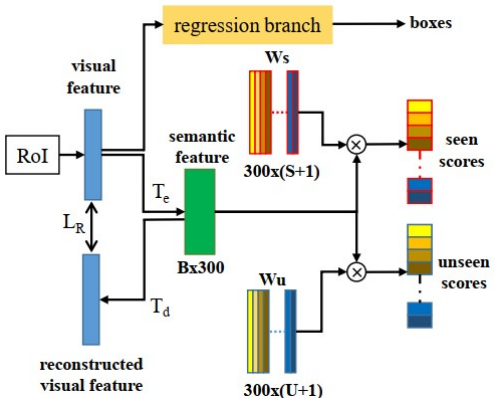
فرض کنید دو مجموعه‌ی کلاس غیر همپوشان (Cs) و(Cu) شامل تصاویر و بردارهای واژه از هر دو مجموعه‌ی کلاس داده شده باشد. مجموعه‌ی آموزش (Dtrain) از کلاس‌های Cs تشکیل شده است که شامل تصاویر xs و بردارهای واژه ws برای کلاس‌های دیده شده است. مجموعه‌ی آزمایش (Dtest) از کلاس‌های CsوCu ساخته شده است که شامل تصاویر x و بردارهای واژه w است، و در آن نمونه‌های دیده شده و نامشخص ممکن است در یک تصویر ظاهر شوند. در فرآیند آموزش، بافرض $Dtrain = d$داریم :

$$\theta = argmax\theta \sum_{i=1}^{d} logp(ysiCs|xsi, ws,) \tag{۳}$$

را برای آموزش شبکه به کار می‌بریم، که در آن از (Dtrain)که تنها شامل نمونه‌های کلاس‌های دیده شده است برای بهینه‌سازی پارامترهایθشبکه استفاده می‌شود. در فرآیند استنتاج، هدف به بافرض $Dtest = D$داریم :

$$argmax\theta \sum_{i=1}^{D} logp(ysiCs, yuiCu|xi, w,) \tag{۴}$$

تغییر می‌کند، که به معنی استفاده از شبکه آموزش‌دیده شده θبرای به دست آوردن نتایج دقیق تراکیب نمونه‌های کلاس‌های دیده شده و نامشخص است. به طور خلاصه،θرا از نمونه‌های دیده شده یاد می‌گیریم و از آن برای استنتاج نمونه‌های نامشخص استفاده می‌کنیم. تشخیص‌گر صفر نمونه(Zero-Shot Detector): شاخه‌ی طبقه‌بندی معنایی یک ساختار رمزگذار_رمزگشا است. در این ساختار، از Te برای رمزگذاری ویژگی بصری برای (RoI) ورودی به ویژگی معنایی استفاده می‌شود و از Td برای رمزگشایی ویژگی معنایی به ویژگی بصری در فرآیند آموزش استفاده می‌شود. تابع هزینه بازسازی LR برای کاهش خطا در بازسازی ویژگی بصری و ویژگی بصری بازساخته شده استفاده می‌شود که در معادله زیرتعریف شده است. از معیار خطای میانگین مربعات (مربع



شکل ۲: جزئیات برای آشکارساز شات صفر

دو نرم) بین هر عنصر در ویژگی بصری اصلی Oو ویژگی بصری بازساخته شده R استفاده می شود. تابع هزینه بازسازی LR به صورت زیر تعریف می‌شود:

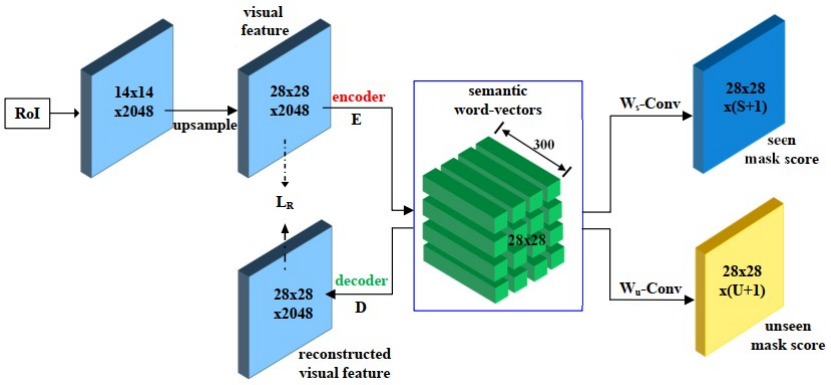
$$LR = \sum_{i} (Oi - Ri)^z \qquad (۵)$$

که در آن Oi و Ri به ترتیب بیانگر iامین عنصر در ویژگی بصری اصلی و بازساخته شده است

سر ماسک معنایی(Semantic Mask Head): ماژول رمزگشا و تابع هزینه بازسازی می‌توانند باعث شوند که شبکه یک تطبیق بین ویژگی بصری و معنایی بیشتری یاد بگیرد. در فرآیند استنتاج، Td حذف می‌شود و می‌توانیم با انجام ضرب ماتریسی بین ماتریس ویژگی معنایی و Ws و ماتریس ویژگی معنایی و Wu، امتیازهای کلاس‌های دیده شده و نامشخص را دریافت کنیم. در فرآیند آموزش، از رمزگذار E برای رمزگذاری ویژگی بصری به بردارهای واژگانی معنایی استفاده می‌شود. سپس از رمزگشا D برای بازگرداندن بردارهای واژگانی معنایی به ویژگی بصری بازساخته شده استفاده می‌شود و از تابع هزینه LR برای کمینه کردن تفاوت بین دو ویژگی بصری استفاده می‌شود. در فرآیند استنتاج، D حذف می‌شود. Ws-Conv و Wu-Conv هر دو لایه‌های کانولوشن ثابت هستند و ما از آنها برای انجام کانولوشن پیکسل به پیکسل بر روی بردارهای واژگانی معنایی برای به دست آوردن نتایج تشخیص نمونه‌های کلاس‌های دیده شده و نامشخص استفاده می‌کنیم. در این ساختار، از یک رمزگذار برای رمزگذاری ویژگی‌های بصری به بردارهای ویژگی معنایی استفاده می‌شود. سپس با استفاده از یک رمزگشا، بردارهای ویژگی معنایی به ویژگی‌های بصری بازساخته می‌شوند. این ساختار به عنوان سر شبکه برای تولید ماسک‌های معنایی برای شی‌ها استفاده می‌شود. با استفاده از این ماسک‌ها، می‌توان تصاویر را به شی‌های مختلفی تقسیم کرد و برای هر شی در تصویر، ماسک معنایی خود را به دست آورد.[۴]

۲.۳ پیش بینی انتها به انتها(End-to-End Learning):

یکی از مفاهیم ابتدایی تو هوش مصنوعی بازگشتی انتها به انتها است که تو شبکه‌های عمیق استفاده میشود یک شبکه عصبی بسیار عمیق روی داده‌های بسیار بعد بالا آموزش داده اگر این شبکه را با استفاده از پشتیبانی از پیش‌بینی انتها به انتها روی یک (GPU) آموزش دهید، مشکلاتی پیش خواهد آمد. شما باید قادر باشید تمام گراف محاسباتی شامل تمام وزن‌ها، فعالسازی‌ها و گرادیان‌ها را به یک بار در حافظه (GPU)جای دهید. شما ممکن است بتوانید این مشکل را با کاهش اندازه دسته‌ها بهبود ببخشید، اما این کار فقط تا حدودی موثر خواهد بود. محدودیت اصلی در اینجا این است که با پیاده‌سازی معمول پشتیبانی از پیش‌بینی انتها به انتها، نمی‌تواند به سادگی لایه‌های جداگانه شبکه را به صورت ناهمزمان آموزش داد. در عبور مستقیم، هر لایه برای انجام هر محاسبه به فعالسازی‌ها نیاز دارد. و در بازگشت با پشتیبانی از پیش‌بینی انتها به انتها، هر لایه برای به‌روزرسانی وزن خود باید منتظر گرادیان‌های پیش‌رو بعدی خود باشد. این قفل کردن به جلو و به عقب، نه تنهااز جداسازی آموزش بخش‌های جزئی شبکه جلوگیری می‌کند که می‌تواند مشکل حافظه را کاهش دهد، بلکه از آن جلوگیری می‌کند که آموزش بخش‌های جزئی



شکل ۳: سر شبکه‌ی ماسک معنایی که یک ساختار رمزگذار_رمزگشا است

شبکه را به صورت ناهمزمان به صورت موازی انجام داده و آموزش را بهبود بخشد.این مدل‌ها در کنار نتایج فوق‌العاده‌ای که نشون دادند، ضعف‌هایی هم دارند به داده‌های فراوانی نیاز دارند، در روش‌های سنتی ما به دانش پیشین فراوانی نیاز داریم اما در این مدل‌ها ما دانش پیشین رو نداریم و برای آموزش بهتر نیاز به داده‌های زیادی خواهیم داشت. در بسیاری از تسک‌ها هنگامی که حجم داده آموزش کم باشد مدل‌های سنتی عملکرد بهتری از خود نشان می‌دهند اما اگر حجم داده‌ها مناسب باشد مدل‌های پیش‌بینی انتها به انتها عملکرد بسیار بهتری از خود نشان می‌دهند. اگر تغییری در دیتاست صورت بگیرد مثلا ابعاد داده‌های ورودی تغییر کند، در مدل‌های سنتی فقط نیاز بود ماژول او تغییر کند اما در این مدل‌ها نیاز است تمام سیستم دچار تغییر گردد.

۳.۳ ساختار رمزگذار رمزگشا(Encoder-Decoder modle):

مدل ترانسفورمر اصلی از یک ساختار رمزگذار/رمزگشا استفاده می‌کند، مانند مدل‌های (seq2seq)قبلی. رمزگذار شامل لایه‌های رمزگذاری است که ورودی را به صورت تکراری و به ترتیب یکی پس از دیگری پردازش می‌کنند، در حالی که رمزگشا شامل لایه‌های رمزگشایی است که همین کار را با خروجی رمزگذار انجام می‌دهند. وظیفه هر لایه رمزگذار برای تولید رمزگذاری است که حاوی اطلاعاتی درباره قسمت‌های مرتبط ورودی با یکدیگر است. این رمزگذاری را به عنوان ورودی به لایه رمزگذار بعدی ارسال می‌کند. هر لایه رمزگشا هم کار برعکس را انجام می‌دهد، با گرفتن همه رمزگذاری‌ها و استفاده از اطلاعات متناسب متنی آنها برای تولید دنباله خروجی. برای این منظور، هر لایه رمزگذار و رمزگشا از یک مکانیزم توجه استفاده می‌کند. برای هر بخش ورودی، توجه وزن‌دهی را به اهمیت هر بخش دیگر می‌کند و از آنها برای تولید خروجی استفاده می‌کند. هر لایه رمزگشا یک مکانیزم توجه اضافی دارد که اطلاعات را از خروجی‌های رمزگشاهای قبلی برمی‌دارد، پیش از اینکه لایه رمزگشا از رمزگذاری استفاده کند. هر دو لایه رمزگذار و رمزگشا یک شبکه عصبی غذایی جلو را برای پردازش بیشتر خروجی دارند و شامل اتصالات باقیمانده و مراحل نرمال‌سازی لایه هستند.

رمزگذار(Encoder modle): هر رمزگذار شامل دو قسمت اصلی است: یک مکانیزم خودتوجهی و یک شبکه عصبی غذایی جلو. مکانیزم خودتوجهی، رمزگذاری ورودی را از رمزگذار قبلی دریافت کرده و وزن‌دهی اهمیت آنها به یکدیگر را برای تولید رمزگذاری خروجی انجام می‌دهد. شبکه عصبی غذایی جلو هر رمزگذاری را به صورت جداگانه پردازش می‌کند. این رمزگذاری‌های خروجی سپس به عنوان ورودی خود به رمزگذار بعدی و همچنین به رمزگشاها ارسال می‌شوند. رمزگذار اول، به جای رمزگذاری، اطلاعات موقعیتی و تعبیه‌های دنباله ورودی را به عنوان ورودی دریافت می‌کند. اطلاعات موقعیتی برای این است که ترانسفورمر بتواند از ترتیب دنباله استفاده کند، زیرا بخش دیگری از ترانسفورمر از این استفاده نمی‌کند. رمزگذار دوطرفه است. توجه می‌تواند بر روی نشانه‌های قبل و بعد از نشانه جاری قرار گیرد. برای در نظر گرفتن چند معناپذیری، از نشانه‌ها به جای کلمات استفاده می‌شود.

رمزگذاری موقعیتی(Positional encoding): رمزگذاری موقعیتی، یک نمایش برداری با اندازه ثابت است که موقعیت نشانه‌ها در دنباله هدف را فراهم می‌کند.

$$f : \mathbb{R} \to \mathbb{R}^d; d \in \mathbb{Z}, d > 0 \qquad (۶)$$

تعریف می‌شود، که در آن d یک عدد صحیح زوج مثبت است. رمزگذاری موقعیتی کامل – به شکل تعریف شده – با معادله زیر داده می‌شود:

$$(f(t)_{2k}, f(t)_{2k+1}) = (\sin(\theta), \cos(\theta)) \quad \forall k \in \{0, 1, \ldots, d/2 - 1\} \tag{۷}$$

که در آن $\theta = \frac{t}{r^k}$ ، $r = N^{2/d}$ و N یک پارامتر آزاد است که باید به طور قابل توجهی بزرگتر از بزرگترین k باشد که به عنوان ورودی به تابع رمزگذاری موقعیتی ورودی شده است. ، نویسندگان انتخاب کردند $N = 10000$ باشد. هنگامی که به صورت تابع مختلطی از نوع

$$f : \mathbb{R} \to \mathbb{C}^{d/2} \tag{۸}$$

نوشته شود، تابع ساده‌تری خواهد بود:

$$f(t) = \left(e^{it/r^k}\right)_{k=0,1,\ldots,\frac{d}{2}-1} \tag{۹}$$

که در آن $r = N^{2/d}$ است. دلیل اصلی انتخاب نویسندگان برای این تابع رمزگذاری موقعیتی، این است که به آن‌ها اجازه می‌دهد تا جابجایی‌ها را به عنوان تبدیلات خطی انجام دهند:

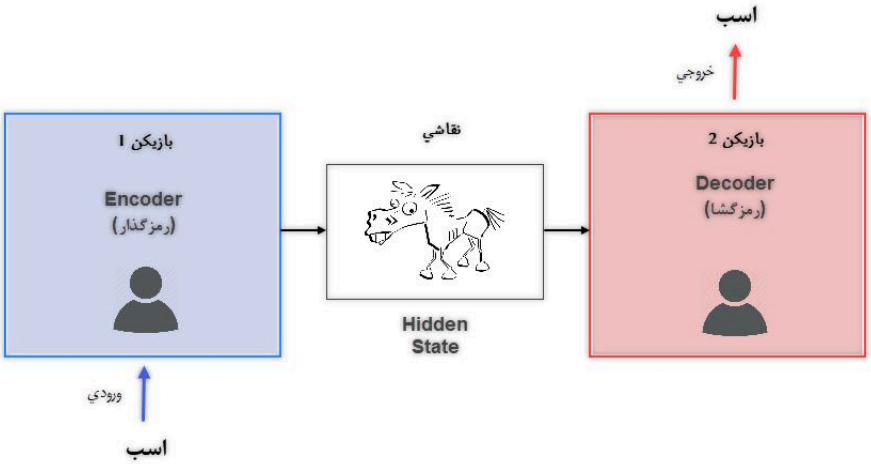
$$f(t + \Delta t) = \mathrm{diag}(f(\Delta t))f(t) \tag{۱۰}$$

که در آن $\Delta t \in \mathbb{R}$ فاصله‌ای است که می‌خواهیم جابجا کنیم. این امکان را به ترانسفورمر می‌دهد تا هر موقعیت رمزگذاری شده را بگیرد و رمزگذاری موقعیت n مرحله به جلو یا به عقب را با ضرب ماتریسی پیدا کند. با ترکیب خطی، هر پیچیدگی ممکن را می‌توان به عنوان تبدیلات خطی پیاده‌سازی کرد:

$$\sum_j c_j f(t + \Delta t_j) = \left(\sum_j c_j \,\mathrm{diag}(f(\Delta t_j))\right) f(t) \tag{۱۱}$$

برای هر ثابت cj. این امکان را به ترانسفورمر می‌دهد تا هر موقعیت رمزگذاری شده را بگیرد و مجموع خطی موقعیت‌های رمزگذاری شده همسایگان خود را پیدا کند. این مجموعه موقعیت‌های رمزگذاری شده، هنگامی که به مکانیزم توجه داده می‌شود، وزن‌های توجه را بر روی همسایگان خود ایجاد می‌کند، تقریباً مانند آنچه در یک مدل شبکه عصبی پیچشی اتفاق می‌افتد. با وزن‌دهی خطی، می‌توان به هر موقعیت رمزگذاری شده پرداخت و مجموعه‌ای از موقعیت‌های رمزگذاری شده همسایگان خود را به دست آورد. این مجموعه رمزگذاری شده، هنگامی که به مکانیزم توجه داده می‌شود، وزن‌های توجه را بر روی همسایگان خود ایجاد می‌کند، تقریباً مانند آنچه در یک مدل شبکه عصبی پیچشی اتفاق می‌افتد. با وزن‌دهی خطی به همسایگان، می‌توان این وزن‌های توجه را به دست آورد. به طور خلاصه، "ما فرض کردیم که این اجازه را به مدل می‌دهد که به راحتی بتواند به وسیله موقعیت نسبی توجه داشته باشد". در پیاده‌سازی‌های معمول، تمام عملیات بر روی اعداد حقیقی نه تعداد مختلط انجام می‌شود، اما از آنجا که ضرب مختلط به عنوان ضرب x۲ ماتریسی حقیقی قابل پیاده‌سازی است، این تفاوت نمادین است.

رمزگشا(Decoder modle): هر دیکدر از سه بخش اصلی تشکیل شده است: مکانیزم خود–توجه، مکانیزم توجه بر روی رمزگذاری‌ها و یک شبکه عصبی پیش‌رو. دیکدر به شکلی مشابه با رمزگذار عمل می‌کند، اما یک مکانیزم توجه اضافی در آن قرار دارد که به جای از دنباله ورودی، از اطلاعات مرتبط در رمزگذاری‌های تولید شده توسط رمزگذار استفاده می‌کند. این مکانیزم را هم می‌توان به عنوان توجه رمزگذار–دیکدر نامید. مانند رمزگذار اول، دیکدر اول اطلاعات موقعیتی و تعبیه‌های دنباله خروجی را به عنوان ورودی خود دریافت می‌کند، نه رمزگذاری. ترانسفورمر نباید از خروجی فعلی یا آینده برای پیش‌بینی خروجی استفاده کند، بنابراین بخشی از دنباله خروجی باید به صورت جزئی ماسک شود تا جریان اطلاعات برعکس جلوگیری شود. این امکان را برای تولید خودکار متن فراهم می‌کند. برای تمام سرهای توجه، توجه نمی‌تواند بر روی توکن‌های پسین قرار گیرد. آخرین دیکدر توسط یک تبدیل خطی نهایی و لایه softmax دنبال می‌شود تا احتمال‌های خروجی را بر روی واژگان تولید کند.



شکل ۴: یک ساختار رمزگذار–رمزگشا است

یادگیری خودنظارت شده(Self-Supervised Learning): یادگیری خودنظارت شده یک روش یادگیری عمیق است که در آن یک مدل با استفاده از داده‌های بدون برچسب پیش آموزش داده می‌شود و برچسب‌های داده به صورت خودکار تولید می‌شوند که در نسخه‌های بعدی به عنوان حقایق اولیه استفاده می‌شوند. ایده اصلی در یادگیری خودنظارت شده، ایجاد سیگنال‌های نظارتی از طریق درک داده‌های بدون برچسب است که در اولین نسخه به صورت بدون نظارت ارائه می‌شوند. سپس، مدل از بین برچسب‌های داده با اطمینان بالا که تولید شده‌اند، برای آموزش در نسخه‌های بعدی مانند یک مدل یادگیری نظارت شده با استفاده از الگوریتم پس‌انتشار گرادیان استفاده می‌کند. تنها تفاوت این است که برچسب‌های داده به عنوان حقایق اولیه در هر نسخه تغییر می‌کنند. یادگیری خودنظارت شده (SSL) به یک پارادایم یادگیری ماشین و روش‌های مربوط به آن برای پردازش داده‌های بدون برچسب به منظور به دست آوردن نمایش‌های مفیدی که می‌توانند در وظایف یادگیری پایین‌دستی مفید باشند، اشاره دارد. مهمترین ویژگی متد SSL این است که نیازی به برچسب‌های انسانی ندارد، به عبارت دیگر، برای آموزش از مجموعه داده‌های بدون برچسب استفاده می‌کند. پس از اینکه در مرحله نخست سیگنال‌های نظارتی (برچسب‌های تولید شده به صورت خودکار) یادگیری شوند، در مراحل دوم و بعدی برای برخی وظایف یادگیری نظارت شده استفاده می‌شوند. به همین دلیل، SSL به عنوان یک شکل میانی از یادگیری بدون نظارت و یادگیری نظارت شده توصیف می‌شود. روش SSL معمولاً بر اساس یک شبکه عصبی مصنوعی یا مدل دیگری مانند یک لیست تصمیم‌گیری استوار است. این مدل در دو مرحله یادگیری می‌کند. در ابتدا، با استفاده از یک وظیفه دسته‌بندی فرعی یا پیش‌بینی اولیه با استفاده از برچسب‌های خودساخته، پارامترهای مدل را مقداردهی اولیه می‌کند. سپس در مرحله دوم، وظیفه واقعی با استفاده از یادگیری نظارت شده یا بدون نظارت انجام می‌شود. وظایف فرعی دیگر شامل تکمیل الگو از الگوهای ورودی ماسک شده (شکاف‌های سکوت در گفتار یا بخش‌های تصویر ماسک شده با رنگ سیاه) هستند. تکنیک‌های یادگیری خودنظارت شده:

وظایف پیش‌متن Pretext tasks: وظایف پیش‌متنی، وظایف فرعی هستند که برای حل آن‌ها از ساختار داده بدون برچسب استفاده می‌شود، اما با وظیفه اصلی نیز ارتباط دارند. به عنوان مثال، مدل ممکن است بر روی یک وظیفه پیش‌متنی از پیش‌بینی چرخش یک تصویر آموزش داده شود تا عملکرد در وظیفه اصلی تصویربرداری بهبود یابد.

یادگیری مقایسه‌ای Contrastive learning: یادگیری مقایسه‌ای یک تکنیک یادگیری خودنظارت شده است که شامل آموزش یک مدل برای تمایز دادن بین نسخه نویزی داده و نسخه پاک آن است. مدل به یادگیری یک نمایش قوی از نویز آموزش داده می‌شود. در داده نمونه نیست. با این حال، بر خلاف یادگیری بدون نظارت، یادگیری با استفاده از ساختارهای داده بدون نظارت صورت نمی‌گیرد. یادگیری نیمه‌نظارتی، یک ترکیب از یادگیری نظارت شده و بدون نظارت است و تنها یک بخش کوچک از داده‌های یادگیری برچسب‌دار را می‌طلبد. در یادگیری انتقالی، یک مدل طراحی شده برای یک وظیفه، در یک وظیفه دیگر استفاده می‌شود. آموزش یک اتوانکودر به طور درونی به یک فرآیند خودنظارت شامل می‌شود، زیرا الگوی خروجی باید بهبود بازسازی الگوی ورودی بهینه شود. با این حال، در اصطلاح کنونی، عبارت ”یادگیری خودنظارت شده“ با وظایف دسته‌بندی مرتبط با آموزش پیش‌فرض مرتبط می‌شود و در این مورد، طراحی وظایف پیش‌فرض انسانی هستند و مشابه آموزش اتوانکودر کاملاً خودمحتوا نیستند. در یادگیری تقویتی، یادگیری خودنظارت شده از ترکیبی از ضایعات می‌تواند توصیف‌های انتزاعی را ایجاد

کند که تنها اطلاعات مهم در مورد وضعیت به صورت فشرده نگهداری می‌شوند.

۴.۳ یادگیری مقایسه‌ای(Contrastive Learning):

یادگیری مقایسه‌ای (Contrastive Learning)یک تکنیک است که با استفاده از اصل مقایسه نمونه‌ها با یکدیگر، ویژگی‌هایی که بین کلاس‌های داده مشترک هستند و ویژگی‌هایی که یک کلاس داده را از کلاس دیگری متمایز می‌کنند را یاد می‌گیرند و عملکرد وظایف بینایی را بهبود می‌بخشند. روش مقایسه‌ای با کمینه کردن فاصله بین دو نمایش از یک نقطه داده مشابه و بیشینه کردن فاصله بین نمایش‌های از نقاط داده مختلف، نمایش‌ها را یاد می‌گیرد. به طور معمول، فاصله بین داده‌های مثبت به حداقل و فاصله بین داده‌های منفی به حداکثر کاهش می‌یابد. روش (Contrastive Learning)عبارت است از یادگیری بازنمایی تصاویر از زوج ورودی‌های مشابه و غیرمشابه. بدین صورت که در طول یادگیری شبکه، انکودر تلاش می‌کند یک بازنمایی معقول را از تصاویر ایجاد کند و دیکودر نیز تا بر اساس تابع خطای (ContrastiveLearning)تصاویر غیرمشابه را از هم تفکیک کند.
فرض کنید X داده‌های ورودی باشد، که هر نمونه (X) دارای برچسب (y) است که کلاس متعلق به آن نمونه را نشان می‌دهد.
فرض کنید (f(x)) بردار ویژگی یا تعبیه‌ی x باشد که از یک شبکه عصبی به دست می‌آید. تابع زیان به صورت زیر تعریف می‌شود:

$$L = -log(exp(f(x_i)^T f(x_j))/(exp(f(x_i)^T f(x_j)) + \sum_{i=1}^{N} exp(f(x_i)^T f(x_k)))) \quad (۱۲)$$

در اینجا، (N) اندازه‌ی دسته است، (i) و (j) نشان‌دهنده‌ی نمونه‌های دسته هستند، (k) نشان‌دهنده‌ی نمونه‌های در همان کلاس با (xi) است و $[yi = yj]$ یک تابع نشانگر است که اگر (xi) و (xj) در یک کلاس باشند یک بازگشت یا آن‌گاه صفر است.
در این تابع از هر نقطه مرجع(xi) ، (N-1) نمونه مثبت (xj) و (N) نمونه منفی (xk) نمونه‌برداری می‌شود. در اولین بخش از تابع، احتمال (softmax) جفت مثبت (xi,xj) محاسبه می‌شود، در حالی که بخش دوم مجموع احتمالات سافتمکس (N) جفت منفی (xi,xk) را محاسبه می‌کند. این تابع فرآیند یادگیری را به سمت ایجاد تعبیه‌هایی که نمونه‌های مثبت را نزدیک‌تر از نمونه‌های منفی قرار می‌دهند و همچنین ایجاد فاصله‌ی مناسب بین انواع مختلف نمونه‌ها هدایت می‌کند.
تابع زیان مقایسه‌ایContrastive loss: با در اختیار داشتن لیستی از نمونه‌های ورودی $\{\mathbf{x}_i\}$، هر کدام دارای برچسب متناظر $y_i \in \{1, \ldots, L\}$ در بین L کلاس هستند. ما می‌خواهیم یک تابع $f_\theta(.) : \mathcal{X} \to \mathbb{R}^d$ را یاد بگیریم که x_i را به یک بردار تعبیه (embedding)تبدیل می‌کند، به طوری که نمونه‌هایی از همان کلاس، تعبیه‌های مشابهی داشته باشند و نمونه‌هایی از کلاس‌های مختلف، تعبیه‌های خیلی متفاوتی داشته باشند. بنابراین، ضایعات مقایسه‌ای، جفت ورودی (x_i, x_j) را می‌گیرد و در صورتی که از همان کلاس باشند، فاصله تعبیه را کمینه می‌کند و در غیر این صورت، فاصله را بیشینه می‌کند.

$$\mathcal{L}_{\text{cont}}(\mathbf{x}_i, \mathbf{x}_j, \theta) = 1[y_i = y_j] \|f_\theta(\mathbf{x}_i) - f_\theta(\mathbf{x}_j)\|_2^2 + 1[y_i \neq y_j] \max(0, \epsilon - \|f_\theta(\mathbf{x}_i) - f_\theta(\mathbf{x}_j)\|_2)^2 \quad (۱۳)$$

در اینجا، ϵ یک هایپرپارامتر است که حداقل فاصله بین نمونه‌های دو کلاس مختلف را تعریف می‌کند.[۵]

۱.۴.۳ تخمین تضادی نویز(Noise Contrastive Estimation):

تخمین تضاد نویز، به اختصار NCE، یک روش برای تخمین پارامترهای یک مدل آماری است. ایده این است که از رگرسیون لجستیک برای تفکیک داده‌های هدف از نویز استفاده شود.[۶] فرض کنید $\mathbf{x}$ نمونه هدف $\mathbf{x} \sim P(\mathbf{x}|C = 1; \theta) = p_\theta(\mathbf{x})$ باشد و $\tilde{\mathbf{x}}$ نمونه نویز $\tilde{\mathbf{x}} \sim P(\tilde{\mathbf{x}}|C = 0) = q(\tilde{\mathbf{x}})$ باشد. توجه کنید که رگرسیون لجستیک لاجیت (یعنی لگاریتم شانس) را مدل می‌کند و در این حالت ما می‌خواهیم لاجیت از یک نمونه u از توزیع داده‌ی هدف را به جای توزیع نویز مدل کنیم:

$$\ell_\theta(\mathbf{u}) = \log \frac{p_\theta(\mathbf{u})}{q(\mathbf{u})} = \log p_\theta(\mathbf{u}) - \log q(\mathbf{u}) \quad (۱۴)$$

بعد از تبدیل لاجیت به احتمالات با استفاده از سیگموئید $\sigma(.)$، می‌توانیم ضایعات تابع خطا چندجمله‌ای را به صورت زیر اعمال کنیم:

$$\begin{aligned} \mathcal{L}_{\text{NCE}} &= -\frac{1}{N}\sum_{i=1}^{N}\left[\log \sigma(\ell_\theta(\mathbf{x}_i)) + \log(1-\sigma(\ell_\theta(\tilde{\mathbf{x}}_i)))\right] \\ \text{where } \sigma(\ell) &= \frac{1}{1+\exp(-\ell)} = \frac{p_\theta}{p_\theta + q} \end{aligned} \tag{۱۵}$$

۲.۴.۳ چارچوب (InfoNCE) :

تابع هزینه InfoNCEدر CPC که الهام گرفته شده از NCE، از تابع هزینه خطاهای طبقه‌بندی متقابل برای شناسایی نمونه مثبت در مجموعه‌ای از نمونه‌های نویز بدون ارتباط استفاده می‌کند. با توجه به یک بردار متناظر $\mathbf{c}$، نمونه مثبت باید از توزیع شرطی $p(\mathbf{x}|\mathbf{c})$ انتخاب شود، در حالی که $N-1$ نمونه منفی از توزیع پیشنهادی $p(\mathbf{x})$ استخراج می‌شود، که مستقل از متناظر $\mathbf{c}$ است. به این منظور، همه نمونه‌ها را به عنوان $X = \{\mathbf{x}_i\}_{i=1}^N$ مشخص می‌کنیم که تنها یکی از آن‌ها، یعنی $\mathbf{x}_{\text{pos}}$، نمونه مثبت است. احتمال درست شناسایی نمونه مثبت به شرح زیر است:

$$p(C=\texttt{pos}|X,\mathbf{c}) = \frac{p(x_{\texttt{pos}}|\mathbf{c})\prod_{i=1,\dots,N;i\neq\texttt{pos}} p(\mathbf{x}_i)}{\sum_{j=1}^{N}\left[p(\mathbf{x}_j|\mathbf{c})\prod_{i=1,\dots,N;i\neq j} p(\mathbf{x}_i)\right]} = \frac{\frac{p(\mathbf{x}_{\texttt{pos}}|c)}{p(\mathbf{x}_{\texttt{pos}})}}{\sum_{j=1}^{N}\frac{p(\mathbf{x}_j|\mathbf{c})}{p(\mathbf{x}_j)}} = \frac{f(\mathbf{x}_{\texttt{pos}},\mathbf{c})}{\sum_{j=1}^{N} f(\mathbf{x}_j,\mathbf{c})} \tag{۱۶}$$

که تابع امتیازدهی $f(\mathbf{x},\mathbf{c}) \propto \frac{p(\mathbf{x}|\mathbf{c})}{p(\mathbf{x})}$ است.
تابع هزینه InfoNCE بهینه‌سازی منفی لگاریتم احتمال طبقه‌بندی درست نمونه مثبت است:

$$\mathcal{L}_{\text{InfoNCE}} = -\mathbb{E}\Big[\log \frac{f(\mathbf{x},\mathbf{c})}{\sum_{\mathbf{x}'\in X} f(\mathbf{x}',\mathbf{c})}\Big] \tag{۱۷}$$

حقیقت این که $f(x,c)$ نسبت چگالی $\frac{p(x|c)}{p(x)}$ را برآورد می‌کند، ارتباطی با بهینه‌سازی اطلاعات متقابل دارد. برای بیشتر کردن اطلاعات متقابل بین ورودی x و بردار متناظر c، داریم:

$$I(\mathbf{x};\mathbf{c}) = \sum_{\mathbf{x},\mathbf{c}} p(\mathbf{x},\mathbf{c})\log\frac{p(\mathbf{x},\mathbf{c})}{p(\mathbf{x})p(\mathbf{c})} = \sum_{\mathbf{x},\mathbf{c}} p(\mathbf{x},\mathbf{c})\log\frac{p(\mathbf{x}|\mathbf{c})}{p(\mathbf{x})} \tag{۱۸}$$

که عبارت لگاریتمی در رنگ آبی توسط f برآورد می‌شود. در وظایف پیش‌بینی دنباله، به جای مدل کردن مشاهدات آینده $p_k(\mathbf{x}_{t+k}|\mathbf{c}_t)$ به صورت مستقیم (که ممکن است نسبتاً پرهزینه باشد)، CPC یک تابع چگالی مدلسازی می‌کند تا اطلاعات متقابل بین $\mathbf{x}_{t+k}$ و $\mathbf{c}_t$ حفظ شود:

$$f_k(\mathbf{x}_{t+k},\mathbf{c}_t) = \exp(\mathbf{z}_{t+k}^\top \mathbf{W}_k \mathbf{c}_t) \propto \frac{p(\mathbf{x}_{t+k}|\mathbf{c}_t)}{p(\mathbf{x}_{t+k})} \tag{۱۹}$$

که $\mathbf{z}_{t+k}$ درمورد ورودی کد شده و $\mathbf{W}_k$ یک ماتریس وزن آموزش‌پذیر است.[۷]

۳.۴.۳ نمونه‌های منفی سخت(**Hard Negative Mining**):

نمونه‌های سخت منفی باید برچسب‌های متفاوتی با نمونه آنکور داشته باشند، اما ویژگی‌های تعبیه شده آن‌ها بسیار نزدیک به نمونه آنکور باشد. در صورت داشتن برچسب‌های حقیقی در مجموعه داده‌های نظارت شده، شناسایی نمونه‌های سخت منفی مربوط به وظیفه مورد نظر آسان است. برای مثال در آموزش تعبیه جمله، می‌توانیم جفت جملاتی که به عنوان "تناقض" در مجموعه

داده‌های NLIبرچسب گذاری شده‌اند، را به عنوان جفت‌های سخت منفی در نظر بگیریم (به عنوان مثال در روش SimCSE)، یا از نمونه‌هایی که بیشترین تطبیق با کلمات کلیدی را دارند و به عنوان نمونه‌های منفی در نظر گرفته می‌شوند (به عنوان مثال در روش DPR)، استفاده کنیم. اما در صورتی که بخواهیم در آموزش بدون نظارت از نمونه‌های سخت منفی استفاده کنیم، کار دشوار می‌شود. افزایش اندازه دسته یا اندازه حافظه به صورت ضمنی به افزایش تعداد نمونه‌های سخت منفی منجر خواهد شد، اما باعث افزایش بار سنگینی حافظه نیز خواهد شد. انحراف نمونه‌برداری در آموزش با تقابل مورد بررسی قرار داده شده است و روش Debias Loss را ارائه کردند. در محیط بدون نظارت، زیرا ما برچسب‌های حقیقی را نمی‌شناسیم، ممکن است به طور اتفاقی نمونه‌های منفی نادرستی را انتخاب کنیم. انحراف نمونه‌برداری می‌تواند باعث کاهش قابل توجهی در عملکرد شود. فرض کنید احتمال کلاس آنکور c یکنواخت باشد $\rho(c) = \eta^+$ و احتمال دیدن یک کلاس متفاوت $\eta^- = 1 - \eta^+$ باشد. • احتمال دیدن یک نمونه مثبت برای $\mathbf{x}$ برابر است با

$$p_x^+(\mathbf{x}') = p(\mathbf{x}'|\mathbf{h}_{x'} = \mathbf{h}_x) \tag{۲۰}$$

• احتمال دریافت یک نمونه منفی برای $\mathbf{x}$ برابر است با

$$p_x^-(\mathbf{x}') = p(\mathbf{x}'|\mathbf{h}_{x'} \neq \mathbf{h}_x) \tag{۲۱}$$

در هنگام نمونه‌برداری $\mathbf{x}^-$، ما نمی‌توانیم به $p_x^-(\mathbf{x}^-)$ واقعی دسترسی داشته باشیم و بنابراین $\mathbf{x}^-$ ممکن است با احتمال η^+ از کلاس آنکور (نامطلوب) نمونه‌برداری شود. بنابراین توزیع داده نمونه‌برداری واقعی به صورت زیر خواهد بود:

$$p(\mathbf{x}') = \eta^+ p_x^+(\mathbf{x}') + \eta^- p_x^-(\mathbf{x}') \tag{۲۲}$$

بنابراین ما می‌توانیم از $p_x^-(\mathbf{x}') = (p(\mathbf{x}') - \eta^+ p_x^+(\mathbf{x}'))/\eta^-$ برای نمونه‌برداری $\mathbf{x}^-$ برای کاهش انحراف در ضرر استفاده کنیم. با N نمونه $\{\mathbf{u}_i\}_{i=1}^N$ از p و M نمونه $\{\mathbf{v}_i\}_{i=1}^M$ از p_x^+، می‌توانیم انتظار دومین عبارت $\mathbb{E}_{\mathbf{x}^- \sim p_x^-}[\exp(f(\mathbf{x})^\top f(\mathbf{x}^-))]$ را در مخرج ضرر آموزش تقابلی برآورد کنیم:

(۲۳)

$$g(\mathbf{x}, \{\mathbf{u}_i\}_{i=1}^N, \{\mathbf{v}_i\}_{i=1}^M) = \max\left\{\frac{1}{\eta^-}\left(\frac{1}{N}\sum_{i=1}^N \exp(f(\mathbf{x})^\top f(\mathbf{u}_i)) - \frac{\eta^+}{M}\sum_{i=1}^M \exp(f(\mathbf{x})^\top f(\mathbf{v}_i))\right), \exp(-1/\tau)\right\}$$

که در آن τ دما و $\exp(-1/\tau)$ کمینه نظری $\mathbb{E}_{\mathbf{x}^- \sim p_x^-}[\exp(f(\mathbf{x})^\top f(\mathbf{x}^-))]$ است. ضرر تقابلی با روش دبیاس شده نهایی به صورت زیر خواهد بود:

$$\mathcal{L}_{\text{debias}}^{N,M}(f) = \mathbb{E}_{\mathbf{x}, \{\mathbf{u}_i\}_{i=1}^N \sim p;\, \mathbf{x}^+, \{\mathbf{v}_i\}_{i=1}^M \sim p^+}\left[-\log \frac{\exp(f(\mathbf{x})^\top f(\mathbf{x}^+)}{\exp(f(\mathbf{x})^\top f(\mathbf{x}^+) + N g(x, \{\mathbf{u}_i\}_{i=1}^N, \{\mathbf{v}_i\}_{i=1}^M)}\right] \tag{۲۴}$$

که در آن $\mathbf{x}$ و $\mathbf{x}^+$ نمونه‌های مثبت هستند که به ترتیب از p و p_x^+ نمونه‌برداری شده‌اند. در این روش دبیاس‌شده، ضرر تقابلی با توجه به احتمال واقعی نمونه‌برداری از هر کلاس محاسبه می‌شود و به این ترتیب، انحراف در ضرر به دلیل نمونه‌برداری نامطلوب از کلاس آنکور، کاهش می‌یابد.[۸] احتمال‌های نمونه‌برداری به گونه‌ای تغییر داده شدند که به نمونه‌های منفی سخت توجه بیشتری شود. برای این منظور، احتمال $p_x^-(x')$ به نسبت شباهت آن با نمونه آنکور، افزایش وزن داده شد. احتمال نمونه‌برداری جدید $q_\beta(x^-)$ به صورت زیر است:

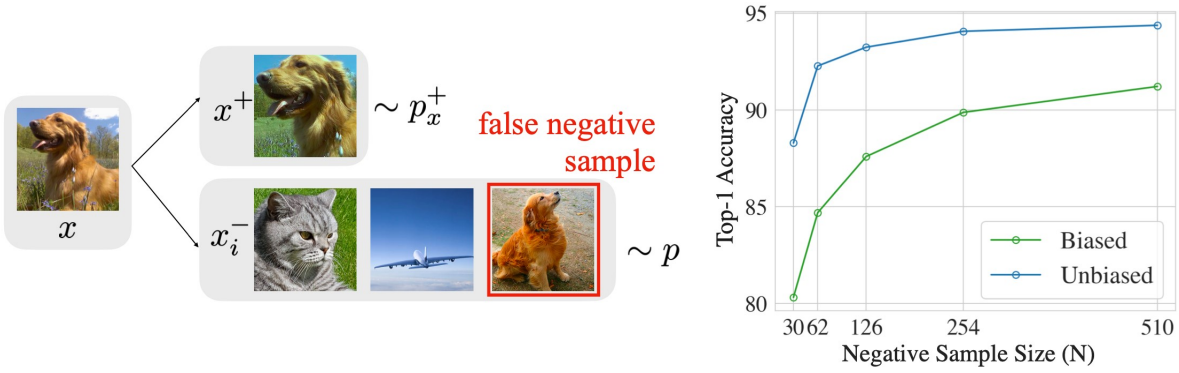
$$q_\beta(\mathbf{x}^-) \propto \exp(\beta f(\mathbf{x})^\top f(\mathbf{x}^-)) \cdot p(\mathbf{x}^-) \tag{۲۵}$$

که در آن β یک پارامتر هایپرمتری است که باید تنظیم شود. می‌توانیم انتظار دومین عبارت در مخرج ضرر تقابلی، یعنی $\mathbb{E}_{\mathbf{x}^- \sim q_\beta}[\exp(f(\mathbf{x})^\top f(\mathbf{x}^-))]$ را با استفاده از نمونه‌برداری مهمی تخمین بزنیم. هر دو ثابت تقسیم Z_β و Z_β^+ نیز به صورت تخمینی محاسبه می‌شوند. به این منظور، از فرمول نمونه‌برداری مهمی استفاده می‌شود:

(۲۶)

$$\mathbb{E}_{\mathbf{u} \sim q_\beta}[\exp(f(\mathbf{x})^\top f(\mathbf{u}))] = \mathbb{E}_{\mathbf{u} \sim p}[\frac{q_\beta}{p}\exp(f(\mathbf{x})^\top f(\mathbf{u}))] = \mathbb{E}_{\mathbf{u} \sim p}[\frac{1}{Z_\beta}\exp((\beta+1) f(\mathbf{x})^\top f(\mathbf{u}))]$$

$$\mathbb{E}_{\mathbf{v} \sim q_\beta^+}[\exp(f(\mathbf{x})^\top f(\mathbf{v}))] = \mathbb{E}_{\mathbf{v} \sim p^+}[\frac{q_\beta^+}{p}\exp(f(\mathbf{x})^\top f(\mathbf{v}))] = \mathbb{E}_{\mathbf{v} \sim p}[\frac{1}{Z_\beta^+}\exp((\beta+1) f(\mathbf{x})^\top f(\mathbf{v}))]$$



شکل ۵: انحراف نمونه‌برداری که به نمونه‌های منفی نادرست در یادگیری مقایسه‌ای اشاره دارد، می‌تواند منجر به کاهش شدید عملکرد شود.

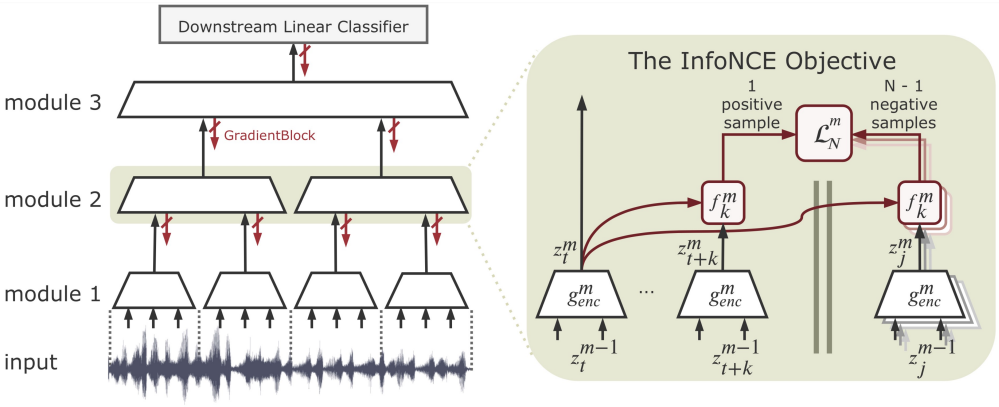
۴.۴.۳ چارچوب (Greedy InfoMax):

این روش در هر stage یک بازنمایی را با تقسیم شبکه به چندین ماژول یاد می گیرد. تابع خطا(Contrastive loss) در هر ماژول وجود دارد که ورودی را با نمونه های مشابه و غیرمشابه مقایسه می کند. پخش گرادیان ها فقط درون هر ماژول صورت می گیرد و بین ماژول ها انتقال نمی یابند. این روش الهام گرفته از یادگیری نورون های مغز است که براساس اطلاعات محلی عمل می کنند. برای نمایش ویژگیهای یادگرفته شده، از تابع خطا loss self supervised استفاده شده است. GreedyInfoMaxبا تقسیم شبکه به ماژول های مستقل، اطلاعات بین patch های مرتبط را افزایش می دهد. بخش انکودر که مسئول یادگیری بازنمایی دیتاست، به چند ماژول مختلف تقسیم شده است که در انتهای هرکدام یک تابع خطای contrastive استفاده شدهاست بنابراین در روند back propagation گرادیان در طول شبکه جریان نمییابد و بلکه هر یک از ماژولها بر اساس تابع خطای تعریف شده و به صورت حریصانه آموزش میبینند. اما باید توجه داشت که گاهی اوقات یادگیری حریصانهی ماژولهای ابتدایی شبکه نمیتواند عملکرد کلی شبکه را تضمین کند. بر این اساس به دلیل آنکه ماژولهای شبکه عصبی از یکدیگر اطلاعی ندارند، بنابراین نمیتوانیم از تمامی ظرفیت یک شبکهی عمیق استفاده کنیم. همچنین ایزولاسیون گرادیان موجب میشود تا لایههای مختلف از یکدیگر بازخوردی نداشته باشند؛ همانطور که گفته شد در حالت عادی لایه های ابتدایی بازنمایی خود را از دیتا بر اساس نظارت لایه ی نهایی بهبود میبخشند و در صورتی که این نظارت وجود نداشته باشد، بازنمایی ضعیف از دیتا در لایه های ابتدایی میتواند عملکرد شبکه را در لایه های نهایی کاهش دهد

در این روش، به جای آموزش مدل به صورت (end-to-end)، از یک تابع هزینه خودـنظارتی محلی استفاده می‌کنیم و از جریان گرادیان بین ماژول‌ها جلوگیری می‌کنیم. هر ماژول، با استفاده از یک تابع رمزگذار، نمایشی از خروجی ماژول قبلی را ایجاد می‌کند. در اینجا، هیچ گرادیانی بین ماژول‌ها جریان نمی‌یابد و برای این منظور از یک اپراتور مسدودکننده گرادیان به نام (GradientBlock(x)) استفاده می‌کنیم. لایه‌های رمزگذار enc gm داخل مدل خود را با استفاده از تابع (log-bilinear) و تابع هزینه محلی (InfoNCE) با اندازه‌گیری مشترک اطلاعات بین ورودی و خروجی هر ماژول آموزش می‌دهیم. برای محاسبه تابع هزینه محلی (InfoNCE)ما از یک مدل خودـرگرسیو استفاده نمی‌کنیم. به جای آن، این مدل را به عنوان یک ماژول جداگانه در بالای لایه‌های رمزگذاری قرار می‌دهیم و به صورت قطعی آموزش می‌دهیم. ماژول رمزگذار در هر ماژول، خروجی ماژول قبلی z_{m-1}^t را به یک نمایش z_m^t تبدیل می‌کند:

$$z_m^t = g_m^{enc}(\text{GradientBlock}(z_{m-1}^t)) \quad (۲۷)$$

لایه‌های رمزگذار g_m^{enc} با استفاده از یک مدل log-bilinear به فرم $f_{m,k}(z_m^{t+k}, z_m^t) = \exp(z_m^{t+k\top} W_{m,k} z_m^t)$ و هزینه



شکل ۶: رویکرد یادگیری InfoMax. Greedy (چپ) برای یادگیری خودنظم دادن نمایش‌ها، تعدادی ماژول را که ورودی به آن‌ها به شیوه معمول ارسال می‌شود، پشت سر هم قرار می‌دهیم، اما گرادیان‌ها به عقب جریان نمی‌یابند. به جای آن، هر ماژول با استفاده از یک هزینه محلی، به صورت فهمگرانه آموزش داده می‌شود. (راست) هر ماژول رمزگذار، نمایش خروجی ماژول قبلی را به یک نمایش جدید تبدیل می‌کند.

محلی InfoNCE آموزش داده می‌شوند:

$$L_m^N = -\sum_{k=1}^{K} \mathbb{E}_{z_m^{t+k}, z_m^t \sim p(z)} \left[\log \frac{\exp(f_{m,k}(z_m^{t+k}, z_m^t))}{\sum_{z_m^j \in X} \exp(f_{m,k}(z_m^j, z_m^t))} \right] \quad (۲۸)$$

در اینجا، $p(z)$ توزیع داده، X مجموعه نمونه‌های منفی و InfoNCEتابع هزینه‌ای است که اطلاعات مشترک بین نمایش فعلی و نمونه‌های منفی را اندازه‌گیری می‌کند. در طول آموزش، گرادیان‌ها با استفاده از یک اپراتور مسدودکننده گرادیان، که به شکل $\text{GradientBlock}(x) = x, \nabla\text{GradientBlock}(x) = 0$ تعریف می‌شود، از جریان بین ماژول‌ها جلوگیری می‌شود. این باعث می‌شود که اطلاعات یک ماژول برای به‌روزرسانی پارامترهای ماژول دیگری استفاده نشود.[۹]

۵.۴.۳ چارچوب (SimCLR):

چارچوب SimCLR ابتدا نمایش‌های عمومی تصاویر بر روی یک مجموعه داده برچسب نخورده را یاد می‌گیرد، و سپس می‌تواند به خوبی با مقدار کمی از تصاویر برچسب دار تنظیم شود تا عملکرد خوبی برای یک وظیفه طبقه‌بندی داده‌شده داشته باشد. نمایش‌های عمومی با به حداکثر رساندن توافق بین دیدگاه‌های مختلف تبدیل‌شده از یک تصویر یک‌سان و به حداقل رساندن توافق بین دیدگاه‌های تبدیل‌شده از تصاویر مختلف، در پی روشی به نام یادگیری تطبیقی، آموزش داده می‌شوند. به روز رسانی پارامترهای یک شبکه عصبی با استفاده از این هدف تطبیقی باعث نمایش دیدگاه‌های متناظر برای «جذب» یکدیگر می‌شود، در حالی که نمایش دیدگاه‌های غیر متناظر «دفع» یکدیگر می‌شوند. برای شروع، SimCLR به طور تصادفی مثال‌هایی را از مجموعه داده اصلی بیرون می‌کشد، و هر مثال را دو بار با استفاده از ترکیبی از افزایش‌های ساده (کراپ تصادفی، انحراف رنگ تصادفی، و تاری گاوسی) تبدیل می‌کند، و دو مجموعه از نماهای متناظر ایجاد می‌کند. دلیل این تغییرات ساده این است که (۱) ما می‌خواهیم نمایش «سازگار» همان تصویر را تحت تبدیل تشویق کنیم، (۲) از آنجا که داده‌های پیش آموزشی فاقد برچسب هستند، ما نمی‌توانیم دلیل قبلی را بدانیم که کدام تصویر شامل کدام طبقه شی است، و (۳) متوجه شدیم که این تحولات ساده برای شبکه عصبی برای یادگیری نمایش‌های خوب کافی هستند، هرچند سیاست تبدیل پیچیده‌تری را نیز می توان به کار برد. SimCLR سپس نمایش تصویر را با استفاده از یک متغیر شبکه عصبی کانولوشنی براساس معماری ResNet محاسبه می‌کند. سپس SimCLRیک تصویر غیر خطی از نمایش تصویر را با استفاده از یک شبکه کاملا متصل (به عنوان مثال MLP) محاسبه می‌کند، که ویژگی‌های نامتغیر را تقویت کرده و توانایی شبکه را برای شناسایی تبدیلات مختلف یک تصویر یک‌سان به حداکثر می‌رساند. ما از نزول گرادیان تصادفی برای به روز رسانی CNN و MLP به منظور به حداقل رساندن تابع اتلاف هدف مقابله‌ای استفاده می‌کنیم. پس از پیش‌آموزش بر روی تصاویر برچسب نخورده، یا می‌توانیم مستقیما از خروجی CNN به عنوان نمایش

یک تصویر استفاده کنیم، یا می‌توانیم آن را با تصاویر برچسب دار تنظیم کنیم تا عملکرد خوبی برای کارهای پایین‌دست داشته باشیم. این چارچوب با حداکثر کردن توافق بین نمایش‌های متفاوتاز یک نمونه با تحریک‌های متفاوت از همان نمونه، از طریق یک اتلاف مقایسه‌ای در فضای لاتانت، نمایش‌هایی برای ورودی‌های بصری یاد می‌گیرد. برای این کار، یک مینی‌بچ با N نمونه به صورت تصادفی نمونه‌برداری می‌شود و هر نمونه با دو عملگر تحریک داده شده و به همین ترتیب $2N$ نمونه تحریک‌شده به دست می‌آید.

$$\tilde{\mathbf{x}}_i = t(\mathbf{x}), \quad \tilde{\mathbf{x}}_j = t'(\mathbf{x}), \quad t, t' \sim \mathcal{T} \tag{۲۹}$$

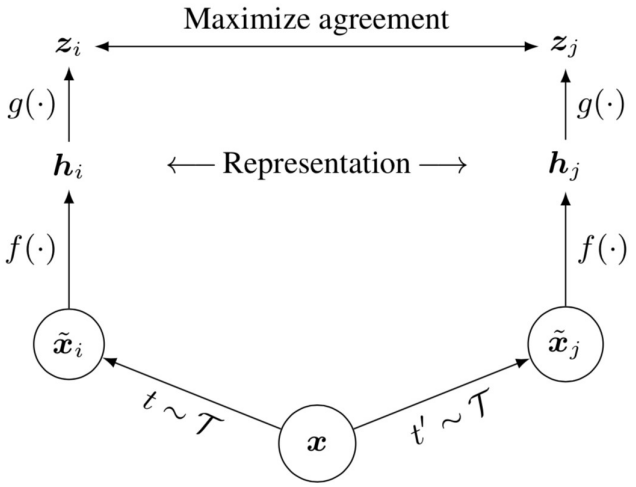
در اینجا، دو اپراتور تحریک داده به نام t و t' از همان خانواده تحریک‌ها $\mathcal{T}$ نمونه‌برداری می‌شوند. این تحریک‌ها شامل برش تصادفی، تغییر اندازه با چرخش تصادفی، اعوجاج رنگ و بلورگوسی هستند. با توجه به یک جفت مثبت، سایر $2(N-1)$ نقطه داده‌ای به عنوان نمونه‌های منفی در نظر گرفته می‌شوند. نمایش با استفاده از یک رمزگذار پایه $f(.)$ تولید می‌شود:

$$\mathbf{h}_i = f(\tilde{\mathbf{x}}_i), \quad \mathbf{h}_j = f(\tilde{\mathbf{x}}_j) \tag{۳۰}$$

از شباهت کسینوسی $\mathrm{sim}(.,.)$ برای تعریف اتلاف یادگیری مقایسه‌ای استفاده می‌شود. توجه کنید که این اتلاف بر روی یک لایه پروژکشن اضافی از نمایش $g(.)$ به جای فضای نمایش مستقیماً عمل می‌کند. اما تنها نمایش $\mathbf{h}$ برای وظایف پس‌رو استفاده می‌شود.

$$\mathbf{z}_i = g(\mathbf{h}_i), \quad \mathbf{z}_j = g(\mathbf{h}j) \; \mathcal{L}\mathrm{SimCLR}^{(i,j)} \quad = -\log \frac{\exp(\mathrm{sim}(\mathbf{z}i, \mathbf{z}j)/\tau)}{\sum k = 1^{2N} \mathbb{1}[k \neq i] \exp(\mathrm{sim}(\mathbf{z}_i, \mathbf{z}_k)/\tau)} \tag{۳۱}$$

در اینجا، $\mathbb{1}_{[k\neq i]}$ تابع نشانه‌گذاری است: ۱ اگر $k \neq i$ باشد و در غیر اینصورت ۰ است. سیم‌سی‌ال‌آر به یک اندازه‌گیری دسته بزرگ نیاز دارد تا تعداد کافی نمونه‌های منفی را جهت دستیابی به عملکرد خوب، در بر گیرد.



شکل ۷: ایک چارچوب ساده برای یادگیری مقایسه‌ای نمایش‌های بصری.

در SimCLR، یک تصویر غیر خطی مبتنی بر MLP قبل از محاسبه تابع تلفات برای هدف یادگیری مقابله‌ای به کار می‌رود، که به شناسایی ویژگی‌های ثابت هر تصویر ورودی و حداکثر کردن توانایی شبکه برای شناسایی تبدیلات مختلف یک تصویر یک‌سان کمک می‌کند. ما در آزمایش‌های خود دریافتیم که استفاده از چنین تصویر غیر خطی به بهبود کیفیت نمایش و بهبود عملکرد طبقه‌بندی کننده خطی آموزش‌دیده بر روی نمایش آموخته‌شده SimCLR تا بیش از ۱۰٪ کمک می‌کند. به طور جالب توجهی، مقایسه بین نمایش‌های استفاده‌شده به عنوان ورودی برای مدول تصویر MLP و خروجی از تصویر نشان می‌دهد که نمایش مرحله قبلی هنگامی که توسط طبقه‌بندی کننده خطی اندازه‌گیری می‌شود عملکرد بهتری دارد. از آنجا که تابع زیان برای

Algorithm 1 SimCLR's main learning algorithm.

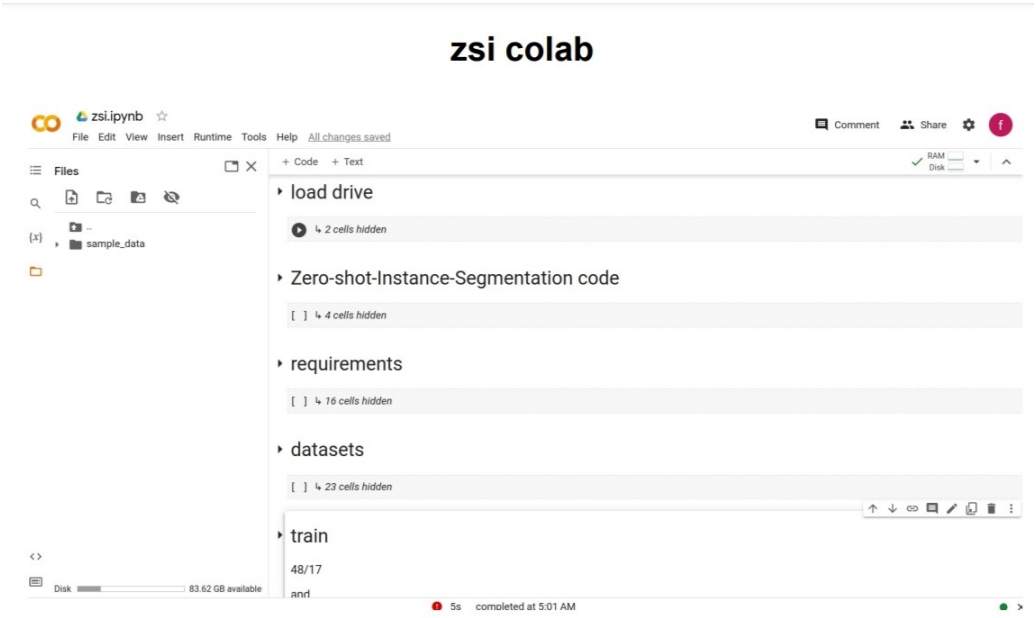
input: batch size N, constant τ, structure of f, g, $\mathcal{T}$.
for sampled minibatch $\{\boldsymbol{x}_k\}_{k=1}^{N}$ **do**
  for all $k \in \{1, \ldots, N\}$ **do**
    draw two augmentation functions $t \sim \mathcal{T}$, $t' \sim \mathcal{T}$
    # the first augmentation
    $\tilde{\boldsymbol{x}}_{2k-1} = t(\boldsymbol{x}_k)$
    $\boldsymbol{h}_{2k-1} = f(\tilde{\boldsymbol{x}}_{2k-1})$ # representation
    $\boldsymbol{z}_{2k-1} = g(\boldsymbol{h}_{2k-1})$ # projection
    # the second augmentation
    $\tilde{\boldsymbol{x}}_{2k} = t'(\boldsymbol{x}_k)$
    $\boldsymbol{h}_{2k} = f(\tilde{\boldsymbol{x}}_{2k})$ # representation
    $\boldsymbol{z}_{2k} = g(\boldsymbol{h}_{2k})$ # projection
  end for
  for all $i \in \{1, \ldots, 2N\}$ and $j \in \{1, \ldots, 2N\}$ **do**
    $s_{i,j} = \boldsymbol{z}_i^{\top} \boldsymbol{z}_j / (\|\boldsymbol{z}_i\| \|\boldsymbol{z}_j\|)$ # pairwise similarity
  end for
  define $\ell(i, j)$ **as** $\ell(i, j) = -\log \frac{\exp(s_{i,j}/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(s_{i,k}/\tau)}$
  $\mathcal{L} = \frac{1}{2N} \sum_{k=1}^{N} [\ell(2k-1, 2k) + \ell(2k, 2k-1)]$
  update networks f and g to minimize $\mathcal{L}$
end for
return encoder network $f(\cdot)$, and throw away $g(\cdot)$

شکل ۸: الگوریتم برای SimCLR.

اهداف مقایسه‌ای مبتنی بر خروجی افکنش است، تا حدی عجیب است که نمایش قبل از افکنش بهتر باشد. ما حدس می‌زنیم که هدف ما منجر به لایهی نهایی شبکه می‌شود تا نسبت به ویژگی‌هایی مانند رنگی که ممکن است برای کارهای پایین‌دست مفید باشد، نامتغیر شود. با هد افکنش غیر خطی اضافی، لایه نمایش قبل از این که سر افکنش قادر به حفظ اطلاعات مفیدتر در مورد تصویر باشد.[۱۰]

۴ چالش ها:

در هنگام پیاده سازی کد مربوط به مسئله یادگیری نمونه صفر ما با چالش هایی مواجه شدیم که کد پیاده سازی شده و تغییرات اعمال شده آن در فایل زیپ ضمیمه شده موجود میباشد.[۱۱]



شکل ۹: ZSIcolab

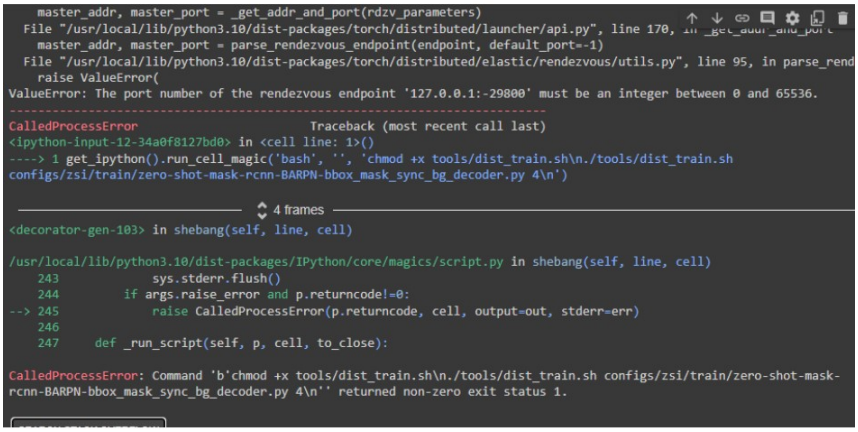
در کد باکس Zero-shot-Instance-Segmentationفایل های اجرا کد را از گیت دانلود می کنیم در گیت گفته شده است که یک محیط آناکونداanaconda بسازید و پکیج ها را نصب کنید فعالسازی محیط آناکوندا در کولبcolab یک سری مشکالت داره (خطا شماره ۱)به همین دلیل از دستورات معادل پیپpipاستفاده کردیم.

در سلولی که در شکل بعدی نشان داده شده باید ورژن پایتونی که می خواهیم استفاده کنیم را اتعیین کنیم که عدد ۲ را وارد کادر پایین می کنیم و Enter می زنیم تا برنامه در محیط پایتون ورژن ۷.۳ اجرا شود.

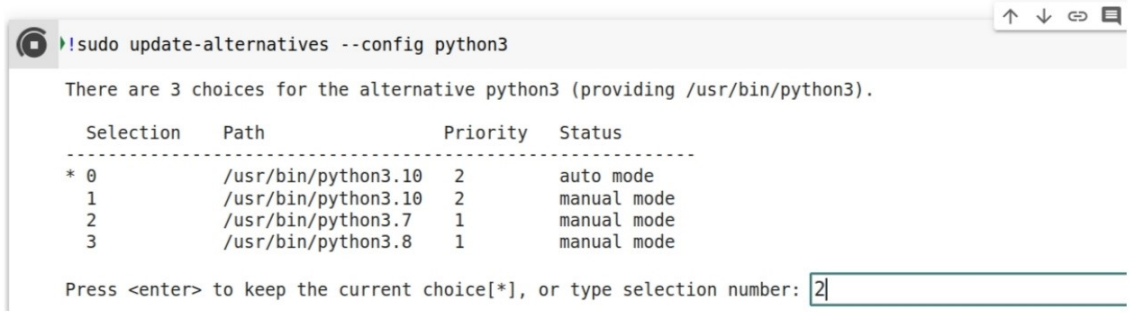
دیتاستی که در مسئله ما مورد استفاده قرار گرفته (COCO 2014) میاشد که آن را داخل برنامه دانلود می کنیم. این بخش از برنامه دارای سه بخش میباشد: .۱لینک توضیحات راجع به ساختار دیتاست .۲کپی کردن مجموعه داده ای از درایو .۳دانلود مجموعه داده ای چون دیتا ست را درداخل درایو به صورت فایل ذخیره شده نداریم . از بخش کپی از گوگل درایو رد می شویم و فقط بخش دانلود را اجرا می کنیم.

قسمت آموزش که طبق توضیحات مقاله دو ورژن دارد ۶۵/۱۵ و ۴۸/۱۷ قبل از آموزش باید یک سری اصالحات در کد انجام گیرد. خطا۲:

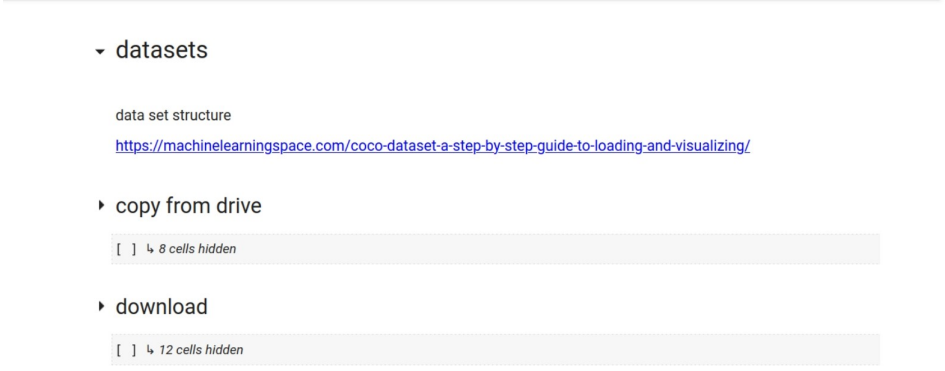
به دلیل عدم تطابق ورژن پکیج ها این خطا رخ داده است که برای اصلاح آن یکی از فایل های پکیج تورچ-ویژن را به صورت زیر تغییر میدهیم)در ورژن های قبل<PIL>از کلمه VERSION_PILLOW برای گرفتن ورژن استفاده میشد و در ورژن های جدید از___version___ استفاده می شود که در این فایل همین مورد اصالح می گردد.)



شکل ۱۰: خطای فعالسازی محیط آناکوندا در کولب



شکل ۱۱: تغییر ورژن پایتون قابل اجرا



شکل ۱۲: دانلود دیتاست و استخراج فایل های آن

in

```
~/usr/local/lib/python3.7/dist-packages/torchvision/transformers/functional.py
```

change line 5 to:

```
from PIL import Image, ImageOps, ImageEnhance, __version__ as PILLOW_VERSION
```

شکل ۱۳: خطای شماره دو

خطا شماره ۳:دستوری که برای اجرای مجموعه داده ای آموزش ۶۵/۱۵ وجود دارد اشتباه میباشد علت آن هم اشتباه وارد کرده اسم فایل است که رفع آن به صورتی که در شکل نشان داده شده است میباشد .

```
! ./ tools/dist_train.sh configs/zsi/65_15/train/zero-shot-mask-rcnn-BARPN-bbox_mask_sync_bg_65_15_decoder_notanh.py 4
```

شکل ۱۴: خطای شماره سه

خطا:۴ پورت عددی بین ۰ تا ۶۵۵۳۶ است که در کد این مقاله به صورت اشتباه مقدار ۲۹۸۰۰- قرار داده شده است که برای تغییر آن منفی مقدار دهی شده را در فایل زیر پاک می کنیم.
خطا۵:

_pickle.UnpicklingError: pickle data was truncated /usr/lib/python3.7/multiprocessing /semaphore_tracker.py:144: UserWarning:semaphore_tracker: There appear to be 11 leaked semaphores to clean up at shutdown len(cache))

از جمله چالش هایی که در این پژوهش وجود داشت اتمام محدودیت زمان اجرا فایل بر روی کولب بود که برای رفع آن دو راهکار وجو دارد: روش اول استفاده از ورژن پیشرفته تر کولب(Colab plus) میباشد روش دوم استفاده از سیستم gpu محلی با قابلیت رم بسیار بالا زیرا این مجموعه داده حجمی حدود ۱۲ گیگ را شامل میشود که بر روی کولب معمولی زمان محدود اجرا برای آن پایان میابد.

۵ آزمایش ها و مجموعه داده ای:

۱.۵ مجموعه داده(COCO)

جامعه تحقیقات دید کامپیوتری، برای آزمون عملکرد مدل‌های جدید و بهبودهای مدل‌های موجود، از مجموعه داده‌های استاندارد استفاده می‌کند. با این رویکرد، می‌توان کارایی مدل‌های مختلف را به طور کلی با یکدیگر مقایسه کرد تا نشان داد که یک مدل از لحاظ کارایی بیشتر یا کمتر از دیگری است. Common Objects in Context (COCO) نمونه‌ای از چنین مجموعه داده‌هایی برای بنچمارک کردن است، که به طور گسترده در جامعه تحقیقات دید کامپیوتری استفاده می‌شود. حتی برای پزشکان

in

`zsi/tools/dist_train.sh`

line 8 set port number to **29800**

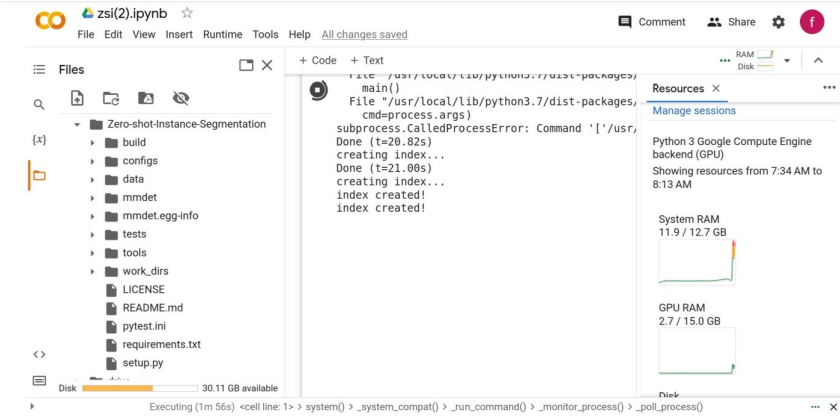
```
#!/usr/bin/env bash

PYTHON=${PYTHON:-"python"}

CONFIG=$1
GPUS=$2

$PYTHON -m torch.distributed.launch --nproc_per_node=$GPUS --master_port -29800 \
    $(dirname "$0")/train.py $CONFIG --launcher pytorch ${@:3}
```

شکل ۱۵: خطای شماره چهار



شکل ۱۶: اتمام محدودیت زمان اجرا فایل بر روی کولب

عمومی در این حوزه هم کاربرد دارد. در این پست، به مجموعه داده COCO می‌پردازیم و دلایل و اهداف ایجاد این مجموعه داده را توضیح می‌دهیم و ویژگی‌ها و معیارهای مجموعه داده را بررسی می‌کنیم. بیایید با صحبت درباره مجموعه داده COCO شروع کنیم. مجموعه داده(Common Objects in Context (COCO شرکت مایکروسافت استاندارد طلایی برای ارزیابی عملکرد مدل‌های دید کامپیوتری به روز است. COCO شامل بیش از ۳۳۰،۰۰۰ تصویر است، که بیش از ۲۰۰،۰۰۰ تصویر از آن‌ها برچسب‌گذاری شده است و در چندین دسته از اشیاء وجود دارد. COCOیک پروژه مشترک است که توسط حرفه‌ایان دید کامپیوتری از انواع مختلف موسسات معتبر اداره می‌شود، از جمله گوگل، کالتک و جورجیا تک. مجموعه داده COCOطراحی شده است تا نمایشی از اشیاءی که در زندگی روزمره ما با آن‌ها روبرو می‌شویم، از جمله وسایل نقلیه مانند دوچرخه، حیوانات مانند سگ و انسان‌ها، ارائه دهد. مجموعه دادهCOCOشامل تصاویر از بیش از ۸۰ دسته از "اشیاء" و ۹۱ دسته از "چیزهای" عمومی است، که بدین معناست که مجموعه داده می‌تواند برای بنچمارک کردن مدل‌های عمومی مفید باشد. علاوه بر این، مجموعه داده COCO شامل موارد زیر است: ۱۲۱،۴۰۸ تصویر ۸۸۳،۳۳۱ برچسب‌گذاری اشیاء ۸۰ کلاس داده نسبت تصویر متوسط ۶۴۰ × ۴۸۰ است شما می‌توانید از جستجوی معنایی داخل COCOبرای بهتر درک داده‌ها استفاده کنید یا به ترکیب کلاس‌ها نگاهی بیندازید. ما در COCOتصاویر عجیبی را پیدا کردیم که به عنوان یادآوری خوبی برای این است که همیشه باید یک درک عمیق از داده‌های آموزشی خود داشته باشید. مجموعه داده COCOبرچسب‌گذاری شده است، از این رو می‌توان از آن برای آموزش مدل‌های دید کامپیوتری نظارت شده استفاده کرد تا بتوانند اشیاء رایج در مجموعه داده را شناسایی کنند. مجموعه داده COCOهمچنین یک مجموعه داده پایه برای آموزش مدل‌های دید کامپیوتری در روش آموزش نظارت شده فراهم می‌کند. پس از آموزش مدل با استفاده از مجموعه داده COCO، می‌توان آن را برای یادگیری وظایف دیگر با یک مجموعه داده سفارشی، بهینه کرد. بنابراین، می‌توانید مجموعه داده COCOرا به عنوان یک تخته‌پرشتی تصور کنید: آن به شما کمک می‌کند تا یک مدل عمومی بسازید و می‌توانید با داده خودتان آن را سفارشی کنید تا عملکرد آن برای وظایف خاص بهبود یابد. در ویدئو زیر، به بحث درباره شروع یادگیری انتقالی از مجموعه داده COCOمی‌پردازیم. این ویدئو به بررسی اشیاء موجود در مجموعه داده COCOو نحوه نمایش مختلف اشیاء می‌پردازد. مجموعه داده COCO می‌تواند برای چندین وظیفه دید کامپیوتری استفاده شود. COCO به طور عمومی برای شناسایی اشیاء، تقسیم بندی معنایی و تشخیص نقاط کلیدی استفاده می‌شود. بیایید به هر یک از این نوع مسائل به تفصیل بپردازیم. اشیاء با یک جعبه محدود کننده و برچسب کلاس برچسب‌گذاری می‌شوند. این برچسب‌گذاری می‌تواند برای شناسایی محتوای یک تصویر استفاده شود. در مثال زیر، زرافه‌ها و گاوها در یک عکس از فضای باز شناسایی شده‌اند. در تقسیم بندی معنایی، مرز اشیاء با یک ماسک و کلاس اشیاء با یک برچسب کلاس برچسب‌گذاری می‌شوند. شما می‌توانید از این ویژگی استفاده کنید تا دقیق‌تر بفهمید که اشیاء مختلف در یک تصویر یا فیلم کجا قرار دارند. در تشخیص نقاط کلیدی، انسان‌ها با نقاط کلیدی علایق (مانند آرنج، زانو و غیره) برچسب‌گذاری می‌شوند. سپس می‌توانید از این ویژگی برای پیگیری حرکات خاص مانند این که یک شخص در حال ایستادن یا نشستن است، استفاده کنید. مجموعه داده COCO شامل بیش از ۲۵۰،۰۰۰ انسان با نقاط کلیدی برچسب‌گذاری شده است. مجموعه داده COCO در یک فرمت خاص به نام COCO JSON در دسترس است. COCO JSONبه طور گسترده خارج از مجموعه داده COCO استفاده نمی‌شود. به همین دلیل، اگر می‌خواهید داده‌های خود را به منظور توسعه COCO به کپی مجموعه داده اضافه کنید، ممکن است نیاز به تبدیل برچسب‌های موجود خود به COCO داشته باشید. به همین ترتیب، می‌توانید داده‌های COCO را به هر کدام از بسیاری از فرمت‌ها (مانند YOLO Darknet TXT) نیز تبدیل کنید، اگر این کاربردی برای مورد استفاده شما مناسب باشد. شرکت Roboflow برای انجام این کار به شما کمک می‌کند. به دلیل دسترسی به راهنمای تبدیل داده‌ها به و از فرمت COCO JSON، می‌توانید نحوه تبدیل داده‌ها به و از فرمت COCO را بررسی کنید.

۲.۵ کد رسمی مقاله

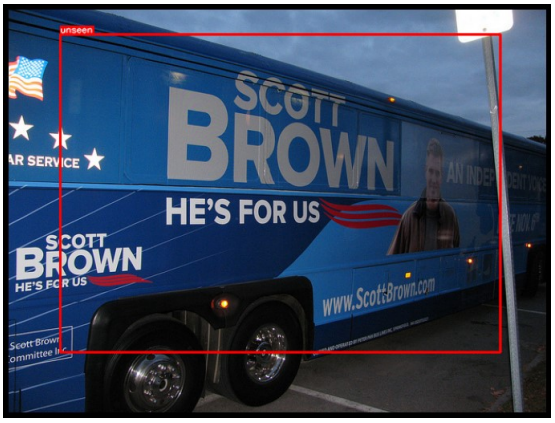
کد رسمی مقاله[۱۱] مورد استفاده قرار گرفت.

۱.۲.۵ تجزیه و تحلیل اجزای مختلف:

ما به بررسی مشارکت اجزای اصلی روش خود می‌پردازیم. ZSDبه معنی Zeroshot Detector ماست، SMH نشان دهنده Semantic Mask Head می‌باشد، و Encoderو Decoder در زیر SMH به ترتیب به ماژول‌های Encoder و Decoder

در Semantic Mask Head اشاره دارند. Det Decoder نشان می‌دهد که ماژول Decoder به Zeroshot Detector اضافه شده است و BARPN Syncbgبه BARPN و استراتژی پس‌زمینه هماهنگ شده اشاره دارد. نتایج برای دو نوع تقسیم‌بندی ۴۸/۱۷ و ۶۵/۱۵ اجرا شده‌اند.

۲.۲.۵ آزمایش:



شکل ۱۷: تشخیص شی دیده نشده (unseen object detection)

ما دو روش تقسیم کلاس‌های دیده شده و ندیده شده را ارائه می‌دهیم: تقسیم ۴۸/۱۷ و تقسیم ۶۵/۱۵، به این معنی که MS-COCO را به ترتیب به ۴۸ کلاس دیده شده با ۱۷ کلاس ندیده شده و ۶۵ کلاس دیده شده با ۱۵ کلاس ندیده شده تقسیم می‌کنیم. برای مجموعه آموزش، ابتدا تمام تصاویری را که شیء کلاس‌های دیده شده را در بخش آموزش MS-COCO دارند، انتخاب می‌کنیم. سپس تصویری که حاوی هر شیء کلاس ندیده شده‌ای باشد را از این مجموعه انتخاب شده حذف می‌کنیم تا تضمین شود که شیءهای ندیده شده در حین آموزش توسط شبکه مشاهده نشوند. برای مجموعه تست، تمام تصاویری که شیء کلاس‌های ندیده شده را در بخش اعتبارسنجی MS-COCOدارند، استخراج می‌کنیم. تصاویر در مجموعه تست ممکن است شیءهای دیده شده و ندیده شده را به طور همزمان داشته باشند.

```
!python tools/zsi_coco_eval.py results/zsi_48_17.bbox.json --ann data/coco/annotations/instances_val2014_unseen_
loading annotations into memory...
Done (t=0.07s)
creating index...
index created!
Loading and preparing results...
DONE (t=2.07s)
creating index...
index created!
Running per image evaluation...
Evaluate annotation type *bbox*
DONE (t=7.42s).
Accumulating evaluation results...
DONE (t=2.52s).
Average Precision  (AP) @[ IoU=0.40      | area=   all | maxDets=100 ] = 0.125
Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.114
Average Precision  (AP) @[ IoU=0.60      | area=   all | maxDets=100 ] = 0.099
Average Recall     (AR) @[ IoU=0.40      | area=   all | maxDets=100 ] = 0.574
Average Recall     (AR) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.539
Average Recall     (AR) @[ IoU=0.60      | area=   all | maxDets=100 ] = 0.483
```

(Segmentation)

شکل ۱۸: بررسی نتایج با متریک ارزیابی (evaluation metric)

۳.۵ نوآوری در حل مسئله:

ما برای بهبود عملکرد مدل از ساختار contrastiveاستفاده کردیم که برای ایجادتغییر میتوانیم از دوروش استفاده کنیم.در روش اول ایده رویکرد contrastive را دربین نمونه ها پیاده سازی میکنیم که برای این کار از روش greedy infomax استفاده میشود.روش contrastive learningیک روش یادگیری بدون نظارت است که در آن، به جای استفاده از برچسب های دسته‌بندی برای یادگیری، از تفاوت‌های بین دو نمونه برای آموزش استفاده می‌شود. رویکرد greedy infomax نیز یک روش برای یادگیری بدون نظارت است که با استفاده از اطلاعات موجود در ورودی، یک رمزگذار با دقت بالا را آموزش می‌دهد.برای پیاده‌سازی روش contrastive با استفاده از رویکرد greedy infomax، می‌توانید به شرح زیر عمل کنید:

۱. انتخاب دو تصویر تصادفی از مجموعه داده‌های خود به عنوان مثبت و منفی.

۲. استفاده از یک رمزگذار عمیق برای تبدیل تصاویر به بردارهای ویژگی.

۳. اعمال یک تابع هزینه contrastive برای دو بردار ویژگی مثبت و منفی. تابع هزینه contrastiveبا تفاوت بین دو بردار ویژگی کار می‌کند و سعی می‌کند تا فاصله بین بردارهای ویژگی تصاویر مثبت را کاهش دهد و فاصله بین بردارهای ویژگی تصاویر منفی را افزایش دهد.

۴. استفاده از روش greedy infomaxبرای آموزش رمزگذار. در این روش، از شبکه عصبی برای پیش‌بینی بیشینه اطلاعات ممکن در ورودی استفاده می‌شود. با استفاده از این روش، رمزگذار به طور خودکار یاد می‌گیرد که ویژگی‌های مهم و با ارزش را استخراج کند.

۵. انتخاب چندین جفت تصاویر مثبت و منفی تصادفی و اعمال تابع هزینه contrastiveبرای هر یک از آنها. این کار باید تعدادی جفت تصاویر مثبت و منفی تولید کند که برای هر یک از آنها تابع هزینه contrastive اعمال شود.

۶. اجرای الگوریتم gradient descentبرای به‌روزرسانی وزن‌های شبکه.

۷. بررسی دقت مدل با استفاده از داده‌های آزمون.

با این روش، می‌توانید یک مدل با دقت بالا برای تفکیک تصاویر مثبت و منفی را پیاده‌سازی کنید که بدون نیاز به برچسب‌های دسته‌بندی، فضای برداری مناسبی از تصاویر را به دست می‌آورد. این روش به طور گسترده‌ای در حوزه‌های مختلفی ازمینه هایی از جمله تشخیص صورت انسان، تشخیص شیء، تشخیص اشیاء و تولید خودکار شرح بر تصاویر مورد استفاده قرار می‌گیرد.در کد رسمی greedy infomaxابعاد تصاویر ورودی مسئله ۳۲*۳۲ میباشد و از طرفی در مجموعه داده ای MC COCO ابعاد تصاویر برابر۴۸۰*۶۴۰ است که این مسئله برای مدل های مسئله ما هنگام خواندن ورودی ها مشکل ایجاد میکند.بنابراین برای رفع این مشکل دو رویکرد پیش روی مسیرما قراردارد: روش اول افزودن یک لایه کانولوشنی که ابعاد تصاویر مجموعه داده ای را به ۳۲*۳۲ کاهش دهد. روش دوم که روش راحت تری است resize کردن ابعاد تصاویر قبل از تغذیه کردن آن ها به مدل شبکه . که برای این کار روش دوم انتخاب شد که برای این کار دورویکرد متداول وجود دارد .رویکرد اول استفاده از تابع torch.nn.Upsample میباشد که معمولا برای افزایش بعد تصاویر از آن در شبکه استفاده میشوداگر ابعاد خروجی مدل از ابعاد ورودی کمتر باشد بر روی داده هاdownsample انجام میشود.از طرفی طبق مطالعات و پژوهش های صورت گرفته استفاده از تابعtorch.nn.function.interpolate برای مدل های ارائه شده نتایج بهتری را درپی داشته است. درنتیجه در ماژول پیاده سازی شده infomax مدل torch.nn.function.interpolate را افزوده کردیم و ابعاد ورودی ها در تابع forward قبل از ورود به لایه های ابتدایی به مقدار ۳۲*۳۲ تغییر اندازه داده میشوند.بعد از این مراحل فایل جدید مربوط به مدل greedy infomax تغییر یافته را در فایل مربوط به کد zsi انتقال میدهیم.در مدل شبکه zsiخود یک بخش به نام infomax head اضافه میکنیم که خروجی نهایی این ماژول یک مقدار عددی مربوط به سه تابع زیان درون شبکه infomax میباشد که مقدار به دست آمده از آن را به تابع زیان اصلی zsi مقاله اصلی می افزاییم.در رویکرد دوم از روش InfoNCE استفاده میکنیم. در این رویکرد در واقع بر روی patch های مرتبط و غیر مرتبط روش contrastive را پیاده سازی میکنیم.روش InfoNCE یک الگوریتم یادگیری بدون نظارت برای بررسی تفاوت‌های بین دو بردار است. این الگوریتم برای بررسی تفاوت‌های بین دو بردار، از اطلاعات مشترک بین دو بردار استفاده می‌کند. روش contrastive نیز یک روش یادگیری بدون نظارت برای بررسی تفاوت‌های بین دو بردار است که با استفاده از تابع هزینه‌ای به نام Contrastive loss، برای یادگیری این تفاوت‌ها استفاده می‌شود.استفاده از روش InfoNCE و پیاده‌سازی روش contrastive بر روی patch ها، به منظور بررسی تفاوت‌های بین دو patch از یک تصویر القا می‌شود. برای این کار، ابتدا باید دو patchتصویر را انتخاب کرد و سپس میزان تفاوت بین آن‌ها را با استفاده از یکی از این دو روش بررسی کرد.برای پیاده‌سازی روش InfoNCEبر روی patchها، می‌توانیم از شبکه‌های عصبی مبتنی بر

رمزگذار استفاده کنید. در این روش، دو patch تصویر به شبکه داده می‌شوند و شبکه باید اطلاعات مشترک بین این دو patchرا برای بررسی تفاوت‌های آن‌ها استخراج کند. سپس با استفاده از تابع InfoNCE loss، این تفاوت‌ها یادگیری می‌شوند.برای پیاده‌سازی روش contrastive بر روی patchها، می‌توانیم از یک شبکه عصبی مبتنی بر رمزگذار استفاده کنیم و سپس با استفاده از تابع هزینه Contrastive loss، تفاوت‌های بین دو patch را یادگیری کنیم. در این روش، دو patchتصویر به شبکه داده می‌شوند و شبکه باید آن‌ها را به گونه‌ای برای مقایسه با یکدیگر تبدیل کند.در هر دو روش InfoNCE و contrastive، انتخاب داده‌های مناسب و تعیین تعداد و اندازه patchها نقش مهمی در دقت و کارایی روش دارد. همچنین، برای افزایش دقت روش‌های یادگیری بدون نظارت بر روی patchها، می‌توانیم از روش‌های پیش‌پردازش تصویر مانند افزایش تخمین تضادی و شارپن کردن تصویر استفاده کنیم.بعد از اعمال تغییرات انجام شده مدل های مربوط به هردو روش را برروی مدل zero shot learningاولیه پیاده سازی میکنیم و خروجی فایل حاوی ذکر شده برای هردو روش مطابق توضیحاتی که بالاتر ذکر شد مقدار عددی است که این مقدار عددی همان مقدارloss مدل ما میباشد که در نهایت مقدار آن در loss اصلی مدل zero shot ما قرار داده میشود و در نتیجه مدل ما با استفاده از این loss در کل معماری به روز رسانی میشود .در واقع نکته ای که برای انجام تمام این تغییرات وجود دارد این است که با توجه به ادبیات zero shot مدل اصلی ما یک ماژول به نام contrastive head به معماری zero shot قبل از visual feature های مسئله در معماری BA-RPN مسئله اصلی اضافه میکنیم. در پوشه اصلی مربوط به کد های مسئله دو فایل کد اجرایی به نام های زیر اضافه کردیم:

contrastive_head.py
contrastive_patch_head.py

همچنین در فایل مربوط به آموزش داده های train خروجی مربوط به کدام از این دو فایل را به تابع زیان اصلی (zero-shotloss) افزودیم.

۶ نتیجه گیری

درمقاله ما، با استفاده از روش‌های zero-shot learning و contrastive learning، تلاش کردیم تا دقت تشخیص تصاویر را در حوزه‌ی zeroshot learning افزایش دهیم. برای این منظور، از دو معماری zero shot detector و semantic mask head استفاده کردیم که با استفاده از دیکدر و انکودر توصیفات متنی، توانستیم لیبل‌گذاری دقیق‌تری را برای تصاویر بدست آوریم. بعد از پیاده‌سازی این دو معماری، برای بهبود دقت تشخیص تصاویر، از روش contrastive learning استفاده کردیم. با اعمال تابع هزینهcontrastive بر روی PATCH های مرتبط و غیر مرتبط، توانستیم فضای برداری مناسبی از تصاویر بدست آوریم که بهبود چشمگیری در دقت تشخیص تصاویر ایجاد کرد. همچنین، برای بهبود عملکرد رمزگذار در فرایند استخراج ویژگی، از دو روش greedy infomax و infoNCE بر روی PATCH های مرتبط و غیر مرتبط، به صورت جداگانه استفاده کردیم. این دو روش باعث بهبود دقت و کیفیت تشخیص تصاویر شدند. نتایج ما نشان داد که استفاده از روش‌های zero-shot learning و contrastive learning، می‌تواند بهبود چشمگیری در دقت و کیفیت تشخیص تصاویر ایجاد کند و می‌تواند در بسیاری از برنامه‌های کاربردی، مانند شناسایی فرآیندهای صنعتی و پزشکی، خدمات خودکار و تحلیل داده‌های بزرگ، مورد استفاده قرار گیرد.

مراجع

[۱] Akata، Z. and Schiele، B. Xian، Y. “Zero-shot learning-the good، the bad and the ugly،” in Proceedings of the IEEE conference on computer vision and pattern recognition، pp. ۴۵۸۲–۴۵۹۱، ۲۰۱۷.

[۲] He، Q. and Xiong، H. Zhu، H. Zhu، Y. Xi، D. Duan، K. Qi، Z. Zhuang، F. “A comprehensive survey on transfer learning،” Proceedings of the IEEE، vol. ۱۰۹، no. ۱، pp. ۴۳–۷۶، ۲۰۲۰.

[۳] Zhu X. and Goldberg، B. A. Introduction to semi-supervised learning. Springer Nature، ۲۰۲۲.

[۴] Cui، L. and Zhang، F. Qin، Y. Wu، J. Zheng، Y. “Zero-shot instance segmentation،” in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition، pp. ۲۵۹۳–۲۶۰۲، ۲۰۲۱.

[۵] Hinton، G. and Norouzi، M. Kornblith، S. Chen، T. “A simple framework for contrastive learning of visual representations،” in International conference on machine learning، pp. ۱۵۹۷–۱۶۰۷، PMLR، ۲۰۲۰.

[۶] Hyvärinen، A. and Gutmann M. “Noise-contrastive estimation: A new estimation principle for unnormalized statistical models،” in Proceedings of the thirteenth international conference on artificial intelligence and statistics، pp. ۲۹۷–۳۰۴، JMLR Workshop and Conference Proceedings، ۲۰۱۰.

[۷] Vinyals، O. and Li، Y. Oord، d. v. A. “Representation learning with contrastive predictive coding،” preprint arXiv arXiv:۱۸۰۷.۰۳۷۴۸، ۲۰۱۸.

[۸] Hard Jurie، F. and Herbin، S. Bucher، M. “Zero-shot based learning metric for mining negative classification،” in Computer Vision–ECCV ۲۰۱۶ Workshops: Amsterdam، The Netherlands، ۸–۱۰ October and ۱۵–۱۶، ۲۰۱۶، Proceedings، Part III، vol. ۱۴، pp. ۵۲۴–۵۳۱، Springer، ۲۰۱۶.

[۹] Veeling، B. and O’Connor، P. Löwe، S. “Putting an end to end-to-end: Gradient-isolated learning of representations،” in Advances neural information processing systems، vol. ۳۲، ۲۰۱۹.

[۱۰] Hinton، G. and Norouzi، M. Kornblith، S. Chen، T. “A simple framework for contrastive learning of visual representations،” in International conference on machine learning، pp. ۱۵۹۷–۱۶۰۷، PMLR، ۲۰۲۰.

[۱۱] zhengye۱۹۹۵، “GitHub - zhengye۱۹۹۵/Zero-shot-Instance-Segmentation: Zero-shot instance segmentation paper CVPR ۲۰۲۱ for code.” https://github.com/zhengye1995/Zero-shot-Instance-Segmentation، ۱۵-Jul-۲۰۲۳. [Accessed ۱۵-Jul-۲۰۲۳].