علوم رایانش و فناوری اطلاعات
نشریه علمی انجمن کامپیوتر ایران
مجلد ۲۰، شماره ۲، ۱۴۰۱
صفحات ۱۴-۲۵

تخمین زاویهٔ سر در شناسایی چهرهٔ انسان با استفاده از روش یادگیری خودنظارتی

مهدی پورمیرزایی[۱]، غلامعلی منتظر[۲،*]، سید ابراهیم موسوی[۳]

[۱] دانش‌آموختهٔ کارشناسی ارشد مهندسی فناوری اطلاعات، دانشگاه تربیت مدرّس، تهران، ایران، m.pooirmirzaie@modares.ac.ir

[۲] استاد مهندسی فناوری اطلاعات، دانشگاه تربیت مدرّس، تهران، ایران، montazer@modares.ac.ir

[۳] دانش‌آموختهٔ کارشناسی ارشد مهندسی فناوری اطلاعات، دانشگاه تربیت مدرّس، تهران، ایران، e.moosavi@modares.ac.ir

*نویسنده مسؤول

دریافت: ۱۴۰۰/۱۲/۱۵، بازنگری: ۱۴۰۱/۰۴/۲۰، پذیرش: ۱۴۰۱ /۰۶/۲۷

چکیده

یکی از عناصر مهم در تحلیل ژست افراد، تخمین زاویهٔ سر است؛ لیکن یکی از موانع اصلی برای این تخمین، هزینهٔ برچسب‌گذاری تصاویر است. برچسب‌گذاری زاویهٔ سر افراد در تصاویر مختلف فرایندی هزینه‌بر، زمان‌گیر و نیازمند دانش تاحدی تخصصی است. از همین رو تصاویر برچسب‌دار برای مسئلهٔ تخمین زاویهٔ سر در مقایسه با بقیهٔ مسائل بینایی رایانه محدود است. یکی از راه‌حل‌های جبران کمبود برچسب‌ها، استفاده از روش‌های خودنظارتی است. روش‌های خودنظارتی می‌توانند از داده‌های بدون برچسب (تصاویر چهرهٔ افراد)، به طریق پیش آموزش دادن شبکه‌های عصبی ژرف، ویژگی‌های مناسبی برای تخمین زاویهٔ سر استخراج کنند. در کنار پیش آموزش دادن شبکه‌های ژرف به روش یادگیری خودنظارتی، می‌توان از وظایف خودنظارتی به عنوان تابع هزینهٔ کمکی در کنار وظیفهٔ اصلی تخمین زاویهٔ سر استفاده کرد. این مقاله سعی دارد که تمایز استفاده از روش‌های یادگیری خودنظارتی برای تخمین زاویهٔ سر را نشان دهد. همچنین نشان داده می‌شود که با طراحی معماری یادگیری چند وظیفه‌ای از ترکیب توابع هزینهٔ بانظارت و خود نظارتی، میانگین خطای تخمین زاویهٔ سر تا ۲۹ درصد نسبت به روش پایه بانظارت و معماری HopeNet کاهش می‌یابد.

کلمات کلیدی: تخمین زاویهٔ سر، شناسایی چهره، یادگیری خودنظارتی، یادگیری بانظارت، یادگیری چندوظیفه‌ای، یادگیری ژرف.

۱. مقدمه

شناسایی چهره یکی از مهم‌ترین موضوعات در حوزهٔ بینایی رایانه‌ای است. این امر شامل مواردی همچون تشخیص‌چهره[۱]، تشخیص حالات چهره[۲]، تخمین سن[۳] با استفاده از چهره و تخمین زاویهٔ سر[۴] است. علاوه براین «تخمین زاویهٔ سر» نقشی حیاتی در بسیاری از کاربردهای عملی مانند پایش راننده [۱]، تعامل انسان با رایانه [۲]، تحلیل رفتار انسانی[۵] [۳] و تشخیص چهره [۴] ایفا می‌کند. به همین دلیل در بسیاری از کاربردها، به برآوردی از زاویهٔ سر نیاز داریم تا در برابر تغییرات محیطی مانند انسداد[۶] و روشنایی[۷] مقاوم باشد [۵] و به همین دلیل است که برآورد زاویهٔ سر و چهره در سال‌های اخیر توجه زیادی به خود جلب کرده است [۵–۷].

به طور کلی، دو روش برای تخمین زاویهٔ سر وجود دارد [۸]: روش نخست بر اساس تشخیص نقاط نواحی چهره[۸] است که در آن ابتدا نواحی چهره از روی تصویر چهره استخراج و بعد از تشخیص نقاط مهم[۹]، زاویهٔ سر محاسبه می‌شود. اگرچه روش‌های تشخیص نقاط مهم چهره با پیشرفت «یادگیری ژرف[۱۰]» بهبود چشم‌گیری یافته، اما تخمین زاویهٔ چهره با استفاده از این روش فرآیندی دو مرحله‌ای است و متعاقباً احتمال وجود خطا را هم افزایش می‌دهد. به عنوان مثال اگر مکان نقاط مورد نظر چهره به درستی شناسایی نشود، تخمین زاویه بسیار ضعیف انجام می‌شود [۶]. روش دوم، یادگیری از تصاویر به صورت انتها به انتها[۱۱] است که در آن شبکه‌های عصبی تلاش می‌کنند برای پیدا کردن زوایای مورد نظر، تنها از تصاویر چهره استفاده کنند. نتایج نشان داده که روش های دستهٔ دوم نسبت به دستهٔ نخست عملکرد به مراتب بهتری دارند [۶].

پیشرفت در یادگیری انتها به انتها ارتباط مستقیمی به تعداد برچسب‌ها دارد. از طرفی برچسب زدن برای زاویهٔ سر، پر هزینه، دشوار و زمان‌بر است و به همین دلیل تعداد داده‌های برچسب‌دار موجود در مقایسه با بسیاری از حوزه‌های بینایی رایانه‌ای بسیار کم‌تر است و همین موضوع تأثیری منفی بر نتایج آموزش به وسیلهٔ شبکه‌های یادگیری ژرف دارد. اخیراً پیشرفت‌های چشم‌گیری برای حل مشکل کمبود برچسب با استفاده از روش‌های یادگیری خودنظارتی[۱۲] انجام شده است و باعث شده که روش‌های بی‌نظارت دوباره محبوب شوند[۹–۱۱]. روش‌های خودنظارتی اغلب به عنوان یک مرحلهٔ پیش پردازش(پیش آموزش) برای مقداردهی اولیهٔ وزن‌ها در مسائل طبقه‌بندی استفاده می‌شوند و برای مسائل Fine-grained (مسائلی که در آن تصاویر تشابه زیادی با یکدیگر دارند)، مانند تخمین زاویهٔ سر، استفاده نشده است.

زمانی که روش‌های خودنظارتی به عنوان یک روش پیش‌آموزش[۱۳] برای تصاویر زاویهٔ سر استفاده می‌شود، شبکه در بهترین حالت می‌تواند ویژگی‌های نژادی[۱۴] و هویتی[۱۵] را از تصاویر یاد بگیرد که این اطلاعات برای شناسایی موقعیت سر کاربرد زیادی ندارد. همچنین این روش‌ها به حجم بسیار زیادی داده و پردازش‌های سنگین نیاز دارند. به همین دلیل با توجه به محدود بودن تصاویر کاربردی برای مسائل از نوع Fine-Grained در تخمین زاویهٔ سر، نمی‌توان از این روش‌ها استفاده کرد [۱۲]. در مطالعهٔ [۱۳] نشان داده شده که وظایف خودنظارتی «Pre-text» می‌توانند ویژگی‌های قابل قبولی را برای حل مسائل بانظارت مهیا کنند. در واقع لایه‌های میانی، ویژگی‌های ارزشمندتری را در مقایسه با لایه‌های نهایی برای مسئلهٔ بانظارت از خود نشان می‌دهند. در این روش‌ها، با حرکت از لایه‌های سطح پایین (لایه‌های نخستین) به سمت لایه‌های سطح بالا (لایه‌های آخر)، می‌توان افزایش و سپس کاهش را در ارزیابی خطی ویژگی‌ها برای مسئله بانظارت مشاهده کرد. به عبارت دیگر ویژگی‌های میانی برای یادگیری مسائل بانظارت بهتر از ویژگی‌های لایه‌های نهایی عمل می‌کنند. از سوی دیگر در مسائل یادگیری چند وظیفه‌ای[۱۶]، قبل از یک لایهٔ مشخص، دو وظیفه - دو سَر شبکهٔ ژرف - ویژگی‌های خود را با یکدیگر به اشتراک می‌گذارند و به یکدیگر کمک می‌کنند و پس از آن، ویژگی‌ها شروع به آسیب رساندن به یکدیگر می‌کنند. در مسائل یادگیری چند وظیفه‌ای از این مفهوم به عنوان همکاری و رقابت بین مسائل یاد می‌شود. در این حوزه پرسش نخست آن است که چرا روش‌های خودنظارتی Pre-text به عنوان سَرهای کمک‌کننده در نظر گرفته نمی‌شوند و از آن‌ها در قالب یادگیری چند وظیفه‌ای در بهبود یادگیری مسئلهٔ اصلی یا همان بانظارت استفاده نمی‌شود؟ پرسش دوم آن است که پس از انتخاب سَرهای خودنظارتی، چه لایه‌ای را می‌توان به عنوان بهترین سطح برای به اشتراک گذاری ویژگی‌ها در نظرگرفت؟

به طور کلی دو رویکرد اصلی برای استفاده از یادگیری خودنظارتی وجود دارد: ۱. استفاده از وزن‌های پیش آموزش داده شده ۲. استفاده از روش خودنظارتی به عنوان وظیفهٔ کمکی به طور همزمان در کنار وظیفهٔ اصلی (با نظارت) که یادگیری چند وظیفه‌ای ترکیبی[۱۷] نامیده می‌شود [۱۴]. در این پژوهش ابتدا شبکه عصبی چندوظیفه‌ای مبتنی بر معماری ResNet50 به همراه وظایف کمکی خود نظارتی ساخته می‌شود و سپس این نکته بررسی گردیده که وزن‌های استخراج شده چگونه به سرهای تخمین زاویهٔ سر کمک می‌کنند. مشخص شد که اگر یک وظیفهٔ با نظارت «آ» و یک وظیفه خودنظارتی «ب» وجود داشته باشد، بین نتایج استفاده از وزن‌های روش «ب» به عنوان روش پیش آموزش برای وظیفهٔ «آ» و استفاده از وظیفهٔ «ب» به عنوان وظیفهٔ کمکی «آ» تفاوت وجود دارد. همچنین مشخص شده است اگر یک وظیفهٔ خودنظارتی که برای تخمین زاویهٔ سر آموزش دیده، منجر به کاهش خطا می‌شود، آنگاه لزوماً آن وظیفه نمی‌تواند کارایی مناسبی به عنوان وظیفه‌ای کمکی داشته باشد. در ادامه تأثیر نوع وزن‌دهی اولیه در معماری HMTL بر روی میزان خطای زاویهٔ سر بررسی شده است. در HMTL مشخص شد با قرار دادن وزن‌های اولیهٔ مختلف - وزن‌های تصادفی، آموزش داده شده روی داده‌های ImageNet و پیش آموزش داده شده با روش‌های خودنظارتی - مقدار خطا در مقایسه با با نظارت کاهش می‌یابد.

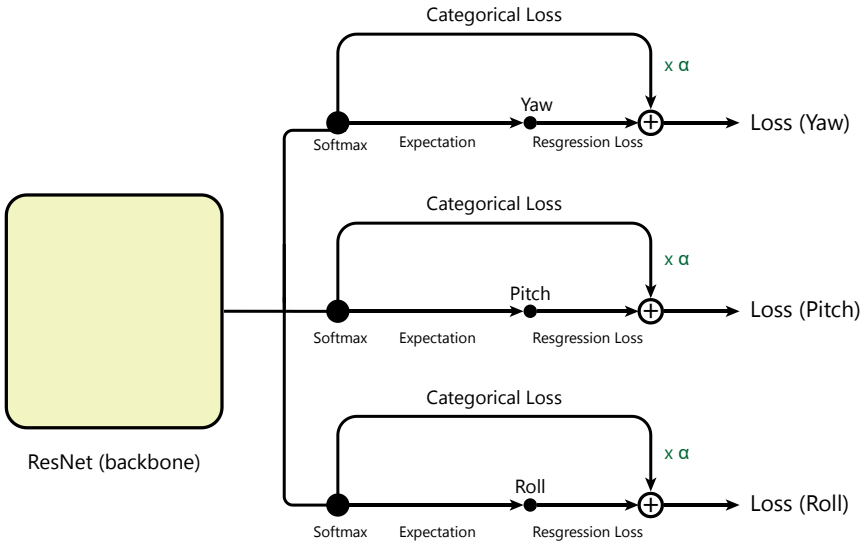
معماری HMTL بر روی داده‌های «AFLW2000 [۱۵]»، «BIWI [۱۶]»، «300W-LP [۱۷]» و «ETH-Xgaze [۱۸]» آموزش داده شده است. از پایگاه دادهٔ «ETH-XGaze» به این علت استفاده شده که در آن تصاویر هر سوژه به صورت جداگانه دسته‌بندی شده است. نکتهٔ مهم در مورد اضافه کردن سرهای کمکی خود نظارتی این است که می‌تواند در کنار دیگر روش‌های بانظارت برای تخمین زاویهٔ سر به کار گرفته شود. اگرچه توسط معماری طراحی شدهٔ HMTL، در استفاده از هریک از دو رویکرد خودنظارتی، کم‌ترین میزان خطا حاصل شده است، اما این معماری در فرآیند آموزش با وزن‌های تصادفی هم مؤثر بوده است. می‌توان مراحل انجام این فرایند را به صورت زیر خلاصه کرد:

الف. معماری بهینه برای روش HMTL با استفاده از دو وظیفهٔ کمکی[۱۸] خودنظارتی در کنار سرهای روش یادگیری بانظارت[۱۹] طراحی شده است. بدین منظور از روش‌های خود نظارت پازل‌کردن[۲۰] و نوع تغییر یافتهٔ چرخاندن[۲۱] تصویر استفاده شده است.

ب. تأثیر پیش آموزش وزن‌ها - وزن‌های داده‌های ImageNet و وزن‌های حاصل از روش خودنظارت - بر روی معماری اصلی ResNet50 و همچنین بر روی معماری چندوظیفه‌ای طراحی شده HMTL، بررسی شده است.

۲. پیشینهٔ تحقیق

در میان انواع مدل‌های یادگیری ژرف، شبکه‌های عصبی پیچشی[۲۲] کاربرد بیشتری در تخمین زاویهٔ سر داشته است [۱۹]. پس از معرفی معماری HopeNet [۶]، اکثر روش‌های پیشنهادی، تخمین زاویهٔ سر - که یک مسالهٔ رگرسیونی است - را با طبقه-بندی ترکیب کرده‌اند. به طور دقیق‌تر روش مذکور به ازای هر زاویهٔ سر، یک تابع هزینه در نظر می‌گیرد که هر تابع شامل دو بخش طبقه‌بندی و رگرسیون است. مقادیر رگرسیون از طریق امید ریاضی بخش طبقه‌بندی تخمین زاویهٔ سر محاسبه می‌شود (شکل ۱). در پژوهش [۵]، برای حل مشکل کمبود داده‌های آموزشی برای تخمین زاویهٔ سر، به جای استفاده از یک برچسب واحد به ازای هر تصویر، از توزیعی از برچسب-ها برای هر تصویر استفاده و سپس چندین تابع هزینه برای محاسبهٔ خطا در نظر گرفته شده است.



شکل ۱- ساختار شبکهٔ HopeNet برای مسئلهٔ تخمین زاویهٔ سر (برای هر زاویه دو تابع خطای پیوسته و گسسته ایجاد شده است)

در پژوهش [20]، برای تخمین زاویهٔ سر از دو وظیفهٔ طبقه‌بندی و رگرسیون استفاده شده است. به طوری که در مرحلهٔ نخست از طبقه‌بندی و در مرحلهٔ دوم، بر خلاف HopeNet، از میانگین کلاس‌های هر زاویهٔ سر برای یافتن مقادیر رگرسیون استفاده شده است.

در روش WHE-Net [۲۱]، تابع هزینه‌ای برای وظیفهٔ رگرسیون زاویهٔ اویلر Yaw پیشنهاد شده که تابع هزینهٔ در هم پیچیده‌ای[۲۳] ایجاد کردند تا دقت زاویهٔ Yaw در تخمین زاویهٔ سر را در محدوده ۳۶۰ درجه پوشش دهد. این تابع هزینه، به جای جریمه کردن مقادیر زاویه به طور مستقیم، حداقل زاویه لازم برای چرخش را جریمه می‌کند.

در پژوهش [۲۲] با استفاده از شبکهٔ عصبی گرافی[۲۴] ،که گروهی از شبکه‌های عصبی برای پردازش داده‌ها است، روشی پیشنهاد شده که تخمین زاویهٔ سر را به عنوان یک مسئله رگرسیون مبتنی بر گراف مدل‌سازی می‌کند. در این روش نقاط مهم چهره، گره‌های[۲۵] گراف و ارتباط بین هر گره، یال‌های[۲۶] گراف را تشکیل می‌دهند. در پژوهش [۲۳] روشی برای تخمین زاویهٔ سر برای تصاویر سه بعدی[۲۷] معرفی شده که در این روش نیازی به تشخیص و همچنین استخراج موقعیت دقیق نقاط کلیدی چهره نیست. در پژوهش [۲۴] از نوع خاصی از عملگرهای پیچشی به نام «کانولوشن ژرف جداپذیر[۲۸]» برای کاهش تعداد پارامترها و حجم محاسبات شبکه برای تخمین زاویهٔ سر استفاده شده است.

۲-۱. یادگیری خودنظارتی

یادگیری خودنظارتی زیرمجموعه‌ای از یادگیری بدون نظارت است که شامل دو مرحله است: ابتدا آموزش روی داده‌های بدون برچسب انجام می‌شود و سپس عملیات تنظیم دقیق[۲۹] وزن‌های پیش آموزش داده شده برای وظیفهٔ اصلی انجام می‌شود. به طور کلی می‌توان یادگیری خودنظارتی را به سه دستهٔ کلی تقسیم کرد:

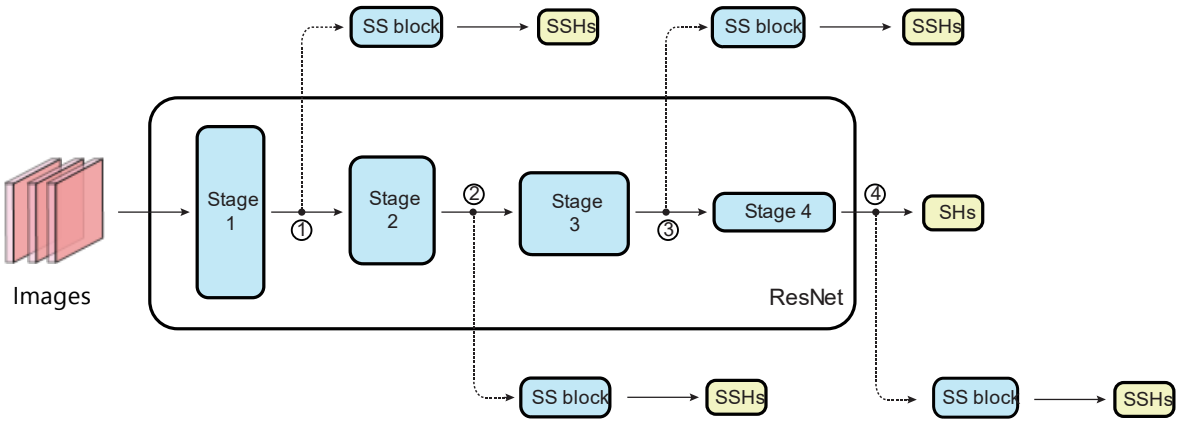
۱. یادگیری مغایرتی[۳۰]
۲. یادگیری غیرمغایرتی[۳۱]
۳. یادگیری وظایف Pretext

در سال‌های اخیر روش‌های یادگیری خودنظارتی از یادگیری وظایف Pretext که شامل مراحل: چرخاندن[۳۲]، رنگ‌آمیزی[۳۳] و پازل کردن[۳۴] است، به سمت یادگیری مغایرتی [۲۵] و یادگیری غیرمغایرتی [۹, ۱۰, ۲۱] پیش رفته‌اند. این روش‌ها با وجود نتایج خوبی که نشان‌داده‌اند، با معایبی مانند نیازمند بودن به حجم عظیمی از داده‌ها برای آموزش مواجه هستند [۱۲]. همچنین برای آموزش شبکه‌ها به روش آنان باید از دستهٔ[۳۵] با اندازهٔ بزرگ استفاده شود [۲۵].

تاکنون در دو پژوهش[۲۲, ۲۳] از یادگیری خودنظارتی برای تخمین زاویهٔ سر استفاده شده است که در پژوهش نخست [۲۷]، با استفاده از سه نوع تابع هزینه، تخمین را انجام می‌دهد و در پژوهش دیگر [۱۴]، با استفاده از افزودن روش خودنظارتی به عنوان دو وظیفهٔ کمکی خودنظارتی یادگیری Pretext، تشخیص هیجان چهره[۳۶] و تشخیص جنسیت[۳۷] و تخمین زاویهٔ سر انجام شده است. البته روش‌های خودنظارتی صرفاً به مرحلهٔ پیش آموزش محدود نمی‌شود و می‌توان آن‌ها را به عنوان آموزش مشترک[۳۸] در کنار یادگیری بانظارت هم استفاده کرد. به عنوان نمونه، در پژوهش [۲۸]، یک روش یادگیری نیمه‌نظارتی[۳۹] با استفاده از آموزش مشترک با یادگیری خودنظارتی پیاده‌سازی شده که وظیفهٔ آن، چرخاندن بوده است؛ در این تحقیق ۱۰ درصد داده‌ها برچسب‌دار و ۹۰ درصد آنان بدون برچسب بوده‌اند.

۳. روش کار

در این بخش روند طراحی معماری HMTL و وظیفه‌های خودنظارتی pre-text بیان می‌شود. شایان ذکر است در اینجا عبارات «خودنظارتی + بانظارت»، «HMTL» و «آموزش به کمک وظایف خودنظارتی کمکی» به جای یکدیگر استفاده شده‌اند و همگی به یک مفهوم اشاره دارند.



شکل ۲- بررسی شاخه‌های متفاوت خودنظارت و با نظارت (شماره‌ها نمایانگر محل جدا شدن شاخه‌هاست). معماری اصلی بر پایهٔ شبکهٔ ResNet50 در نظر گرفته شده است.

۳-۱. معماری چند وظیفه‌ای (HMTL)

در این مقاله از دو نوع معماری استفاده شده است: ۱. با نظارت؛ ۲. خودنظارتی + با نظارت (HMTL). مورد اول مشابه روش HopeNet و معماری دوم بر اساس افزودن وظایف خودنظارتی به معماری اصلی HopeNet در نظرگرفته شده است.

به منظور یافتن مناسب‌ترین معماری HMTL، چهار مکان در معماری ResNet [۲۹] که پایهٔ روش HopeNet است، برای محل جدا شدن هر کدام از شاخه‌های وظایف خودنظارتی، در نظر گرفته شده است. از آنجا که معماری ResNet50 دارای چهار بلوک اصلی از لایه‌های کانولوشن[۴۰] است، مکان‌های مورد نظر مطابق شکل ۲ بر روی محل خروجی بلوک‌ها انتخاب شده‌اند. برای هر کدام از وظایف چرخاندن و پازل-کردن، شاخهٔ خودنظارتی جدا شده از رمزگدار[۴۱] شامل سه لایهٔ کانولوشن همراه با «Normalization Batch» و تابع فعال‌سازی «Relu» و سپس یک لایهٔ متراکم[۴۲] به همراه تابع «Softmax» قرار داده شد. علاوه بر این محل جدا شدن شاخه‌های چرخاندن و پازل کردن و همچنین معماری آنان شبیه به یکدیگر طراحی شده‌اند.

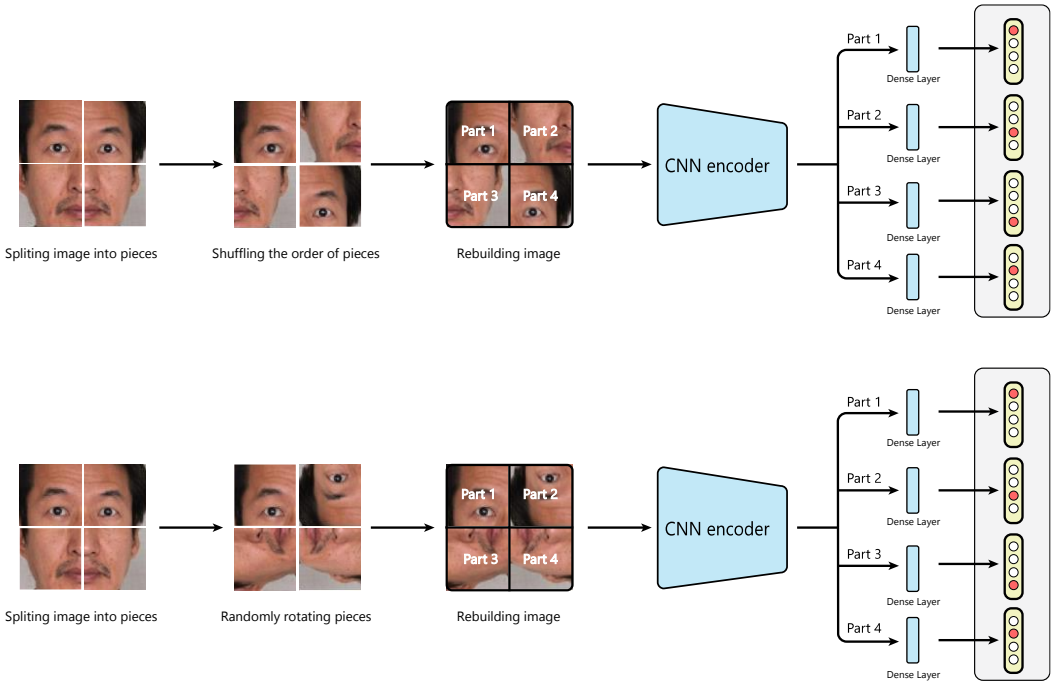
۳-۲. وظایف یادگیری خودنظارتی برای تخمین زاویهٔ سر

برای تخمین زاویهٔ سر، سه وظیفهٔ خودنظارتی در نظر گرفته شده است:

الف. پازل کردن: بر اساس پژوهش [۱۴]، در این مرحله نوع خاصی از روش پازل کردن استفاده شده است که در آن یک تصویر به چهار قسمت مساوی بریده می‌شود و سپس ترتیب قسمت‌های بریده شده به صورت تصادفی تغییر می‌کند و تصویر مورد نظر با ترتیب‌های جدید سر هم می‌شود. در این صورت شبکهٔ عصبی تلاش می‌کند که مکان درست هر ناحیه از تصویر پازل شده را پیش‌بینی کند. به عنوان نمونه، در حالت پازل کردن ۲×۲، چهار سر خودنظارتی (اشاره به هر ناحیه از تصویر) برای شبکه ایجاد می‌شود (شکل ۳).

ب. چرخاندن: در روش مرسوم چرخاندن، تصویر می‌تواند در چهار زاویهٔ n×۹۰ درجه چرخیده شود؛ (که n می‌تواند اعداد صفر تا سه باشد) [۳۰]. به تعبیر دیگر یک تصویر در چهار جهت چرخیده و عدد n به عنوان برچسب آن درنظر گرفته می‌شود؛ لذا با مسئله‌ای چهار کلاسه سروکار داریم. بر خلاف روش مرسوم، در این پژوهش از نوع دیگری از روش چرخاندن استفاده شده که در آن عمل چرخاندن بر روی هر ناحیه از تصویر برش خورده اعمال می‌شود. این بدان معنی است که هر ناحیه می‌تواند مستقل از دیگر نواحی به صورت تصادفی چرخانده شود (شکل ۳). بدین ترتیب مسئله به چهار مسئلهٔ چهارکلاسه تبدیل شده است.

ج. پازل کردن-چرخاندن: این روش از ترکیب دو روش قبل حاصل می‌شود. هر ناحیه از تصویر شامل دو وظیفه است؛ بنابراین به عنوان مثال، زمانی که از روش ۲×۲ پازل کردن-چرخاندن استفاده شود، هشت وظیفهٔ خودنظارتی ایجاد می‌شود و اما اگر تمام نواحی که ترتیب آنها به هم ریخته چرخانده شوند، به دلیل اغتشاش در اطلاعات ورودی، این احتمال وجود دارد که اطلاعات اصلی مرتبط به تخمین زاویهٔ سر از بین برود. به عبارت دیگر، ممکن است بیش از یک راه حل صحیح برای پیش‌بینی وجود داشته باشد. برای غلبه بر این مشکل، باید حداکثر دو قطعه از پازل به صورت تصادفی چرخانده شوند و بقیه بدون تغییر باقی بمانند. این کار به باقی ماندن اطلاعات وظایف اصلی در داخل تصاویر منجر می‌شود (یادآوری می‌شود که نواحی دست نخورده، برچسب صفر برای وظیفه چرخاندن داشته‌اند).



شکل ۳- مراحل پازل کردن (شکل بالا) و چرخاندن (شکل پایین) به همراه چگونگی ساخت سرهای شبکه و تشخیص درست برچسب‌های تولید شده

۳-۳. تخمین زاویهٔ سر

در این پژوهش تفاوت بین سه رویکرد در تخمین وضعیت سر بررسی شده است:

الف. استفاده از وزن‌های پیش آموزش در روش یادگیری خودنظارتی: در این حالت ابتدا وزن‌های بدنهٔ اصلی شبکه (مدل عمیق ResNet 50)، با استفاده از یادگیری خودنظارتی به روش pre-text آموزش داده می‌شود؛ سپس شبکهٔ پیش آموزش داده شده روی برچسب‌های زاویهٔ سر (Yaw، Pitch و Roll) مجدداً آموزش داده می‌شود.

ب. استفاده از وزن‌های پیش آموزش در روش یادگیری بانظارت. این حالت بسیار شبیه به حالت قبل است با این تفاوت که در اینجا از وزن‌های پیش آموزش دیده بر روی تصاویر ImageNet به همراه یادگیری بانظارت برای بدنهٔ اصلی شبکه استفاده می‌شود. استفاده از وزن‌های ImageNet روشی بسیار رایج در حوزهٔ بینایی رایانه است که به منظور آموزش مجدد شبکه بر روی تصاویر و برچسب‌های جدید استفاده می‌شود که به آن «انتقال یادگیری[۴۳]» گفته می‌شود [۳۱].

ج. استفاده از توابع هزینهٔ کمکی خودنظارتی (HMTL). این حالت، از وظیفه‌های کمکی خودنظارتی برای کمک به یادگیری بانظارت تخمین زاویهٔ سر استفاده می‌شود (رابطهٔ (۱)). معماری پایه‌ای این حالت، معماری HopeNet است که با افزوده شدن شاخه‌های خودنظارتی مطابق بخش ۳-۱، معماری HMTL مورد نظر ایجاد می‌شود. این معماری از وظایف خودنظارت در کنار بانظارت ایجاد می‌شود. از آنجا که هستهٔ معماری پیشنهادی در این پژوهش ناشی از شبکهٔ ResNet 50 است، علاوه بر وزن‌های تصادفی برای مقدار دهی شبکهٔ ResNet 50، از وزن‌های پیش آموزش مراحل الف و ب نیز می‌توان استفاده کرد. تمام وظایف کمکی خودنظارتی در اینجا از نوع طبقه‌بندی هستند.

$$\mathcal{L}_{Total} = \mathcal{L}_{SL} + \mathcal{L}_{SSL} = (\mathcal{L}_{Yaw} + \mathcal{L}_{Pitch}) + \left(\sum_j \mathcal{L}_{Puzzle_j} + \sum_j \mathcal{L}_{Rotation_j}\right) = \left(\left(\sqrt{\frac{1}{n}\sum_{j=1}^{n}(\hat{y}-y)^2}\right)_{yaw} + \left(\sqrt{\frac{1}{n}\sum_{j=1}^{n}(\hat{y}-y)^2}\right)_{pitch}\right) + \sum_{part_j}^{parts}\left(\left[\sum_i y_{i,part\,j}\log(\hat{y}_{i,part\,j})\right]_{puzzle} + \left[\sum_i y_{i,part\,j}\log(\hat{y}_{i,part\,j})\right]_{rotation}\right) \quad (۱)$$

که در این روابط:

L_{SL}: شامل دو تابع هزینهٔ RMSE برای سرهای وظایف با نظارت و L_{SSL}: توابع هزینهٔ Categorical Cross Entropy برای وظیفه‌های چرخاندن و پازل‌کردن خودنظارتی هستند.

همان‌گونه که در رابطهٔ (۱) نشان داده شده تابع خطا شامل مجموع توابع خطای سرهای خودنظارتی و با نظارت است. توابع خطای سرهای خودنظارتی پازل کردن و چرخاندن از جنس طبقه‌بندی («Cross entropy») و توابع خطای سرهای بانظارت از جنس رگرسیون («RMSE») است.

۴. پیاده‌سازی معماری

مطابق با قسمت قبل، معماری ResNet50 به عنوان رمزگذار شبکه لحاظ و ورودی شبکه ۲۲۴×۲۲۴×۳ قرار داده شد. در این روش برای آموزش از الگوریتم بهینه-سازی Adabelief [۳۲] با اندازهٔ دستهٔ ۶۴ استفاده شده است. پایگاه داده‌های مورد استفاده در این قسمت به شرح زیر هستند:

الف. 300W-LP [۱۷]. این پایگاه متشکل از تصاویر نسخهٔ گسترش یافتهٔ 300W است که در آن تصاویر چندین پایگاه مختلف از چهره افراد به همراه نقاط صورت در کنار یکدیگر گردآوری شده است. این پایگاه داده شامل ۶۱۲۲۵ تصویر است.

ب. AFLW2000 [۱۵]. این پایگاه شامل ۲۰۰۰ تصویر از پایگاه AFLW است که با مدل‌سازی سه بُعدی و استخراج نقاط دقیق صورت، زوایای سر با دقت بالا برچسب‌گذاری شده‌اند. لذا حاوی برچسب‌های سه زاویهٔ سر به صورت دقیق است به همین دلیل به عنوان تصاویر مجموعهٔ ارزیابی در این پژوهش در نظر گرفته شده است.

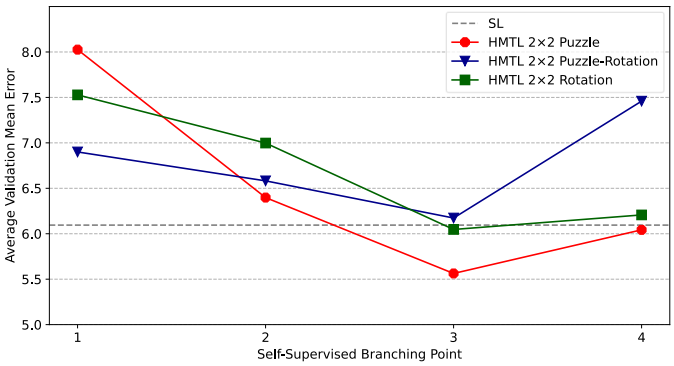
ج. BIWI [۱۶]. این پایگاه در محیطی آزمایشگاهی با ضبط ویدیوهای از نوع RGB-D از سوژه‌های انسانی مختلف در حالت های مختلف سر با استفاده از دستگاه «Kinect v2» جمع آوری شده است. این ویدئوها شامل حدود ۱۵۰۰۰ فریم است که در زوایای مختلف سر ایجاد شده است. این مجموعه داده معمولاً به عنوان معیاری برای تخمین زاویهٔ سر با استفاده از روش‌های تشخیص عمق استفاده می‌شود. البته در این پژوهش فقط از تصاویر رنگی برای تخمین زاویهٔ سر استفاده شده است.

د. ETH-XGaze [۱۸]. این پایگاه متشکل از بیش از یک میلیون تصویر با وضوح بالا از زوایای گسترده سر است. برای گردآوری داده‌های این مجموعه از ۱۱۰ شرکت‌کننده استفاده شده است و با تنظیمات سخت‌افزاری سفارشی و دقیق، تصاویر افراد ثبت شده است. این پایگاه در اولویت اول برای تخمین زاویهٔ نگاه ایجاد شده است. اما در کنار آن، برچسب‌های دو زاویهٔ Yaw و Pitch افراد محاسبه و در در دسترس قرار گرفته است.

۴-۱. یافتن بهترین معماری چند وظیفه‌ای

در ابتدا شبکهٔ ResNet50 و چهار مکان اشاره شده در قسمت ۳-۱ به عنوان مکان جدا شدن شاخه‌های خود نظارتی انتخاب شد. برای انجام این کار، مجموعه داده ETH-XGaze انتخاب و فقط از برچسب‌های زاویهٔ سر استفاده شد. دلیل استفاده از این مجموعه داده، دقت تفکیک تصاویر آن بر اساس هر سوژه است که می‌تواند به نشان دادن عملکرد روش در تعداد سوژه محدود کمک کند. این مجموعه داده فقط دارای محور Yaw و Pitch برای تخمین زاویهٔ سر است و محور Roll را شامل نمی‌شود. به دلیل حجم زیاد تصاویر این پایگاه، یک سوژه از مجموعهٔ آموزش این مجموعه داده مشتمل بر حدود ۱۰ هزار نمونه را برمی‌گزینیم. علت انتخاب این تعداد کم داده‌ها این است که با مقادیر کم داده، تفاوت عملکرد روش‌ها واضح‌تر خواهد بود. از سویی دیگر در گام ارزیابی شبکه، سه سوژه با شماره‌های ۱۰۸، ۱۰۹ و ۱۱۱ استفاده شد. همچنین برای جلوگیری از پدیدهٔ بیش‌برازش[۴۴]، از روش افزودنِ[۴۵] بزرگنمایی تصادفی و تغییر رنگ تصادفی و همچنین از روش Dropout [۳۳] استفاده شد. مقادیر Dropout برای سرهای با نظارت ۰/۴ و برای تمام سرهای خودنظارت ۰/۲ قرار داده شد. مطابق قسمت ۳-۱، برای هریک از وظایف چرخاندن و پازل کردن، شاخهٔ خودنظارتی جدا شده از رمزگذار شامل سه لایهٔ کانولوشن همراه با Batch Normalization و تابع فعال‌سازی Relu و سپس یک لایهٔ متراکم به همراه تابع Softmax لحاظ شد. محل جدا شدن شاخه‌های چرخاندن و پازل کردن و همچنین معماری آنان شبیه به یکدیگر طراحی شده‌اند. در این قسمت سعی شد که بهترین معماری خود نظارت + با نظارت پیدا شود.

در نهایت در هر گام آموزش شبکه، برچسب‌های خودنظارتی نمونه‌های درون هر دسته هنگام استفاده از روش پازل کردن-چرخاندن شبیه به $\{(X,Y) = (X,(R_i,P_i); i \in (1,\dots,4))\}$، روش پازل کردن $\{(X,Y) = (X,(P_i); i \in (1,\dots,4))\}$ و در نهایت برای روش چرخاندن نمونه‌های به صورت $\{(X,Y) = (X,(R_i); i \in (1,\dots,4))\}$ انتخاب شد. نتایج هر سه روش در چهار مکان انتخاب شده در جدول ۱ و شکل ۴ نشان داده شده است. برچسب‌های دو سر بانظارت شامل مقادیر Yaw و Pitch بودند که هر سر با نظارت توسط یک لایهٔ رگرسیون متراکم در بالای خروجی رمزگذار ایجاد شد. در همه روش‌ها ۱۱۰ چرخه[۴۶] برای آموزش در نظر گرفته شد؛ علاوه بر این در ابتدا آهنگ یادگیری[۴۷] روی ۰/۰۰۱ تنظیم و سپس در چرخه‌های ۳۰ و ۴۰، این ضریب در ۰/۱ ضرب شد.



شکل ۴- میانگین خطای میانگین Yaw و Pitch برای مکان‌های مختلف روش پازل کردن، چرخاندن و پازل کردن-چرخاندن ۲×۲ بر روی مجموعه دادهٔ ETH-Xgaze.

با توجه به نتایج مشخص شد که مکان نقطهٔ ۳ نسبت به سایر مکان‌ها میانگین خطای کمتری را در برداشته است. طبق جدول ۱، مشاهده می‌شود که خطای Yaw به‌طور چشمگیری در روش پازل کردن کاهش یافته است. با این حال، وظیفهٔ کمکی چرخاندن در میان سرهای بانظارت کمکی به کاهش خطا کمک نکرده است. به همین دلیل نقطه انشعاب شماره ۳ و وظیفهٔ پازل کردن به عنوان معماری نهایی HMTL برای مراحل بعد در نظر گرفته شد.

جدول ۱- میانگین خطای میانگین Yaw و Pitch برای مکان‌های مختلف روش پازل کردن، چرخاندن و پازل کردن-چرخاندن ۲×۲ بر روی مجموعه دادهٔ ETH-XGaze

روش	مکان جداشدن	Yaw (MAE)	Pitch (MAE)	میانگین
HMTL puzzling-rotation 2×2	1	5.075	8.725	6.9
HMTL puzzling-rotation 2×2	2	5.243	7.921	6.582
HMTL puzzling-rotation 2×2	3	5.243	7.921	6.582
HMTL puzzling-rotation 2×2	4	5.243	7.921	6.582
HMTL rotation 2×2	1	4.985	10.072	7.528
HMTL rotation 2×2	2	4.373	9.623	6.998
HMTL rotation 2×2	3	4.757	7.337	6.047
HMTL rotation 2×2	4	4.701	7.712	6.207
HMTL puzzling 2×2	1	6.478	9.575	8.026
HMTL puzzling 2×2	2	5.667	7.13	6.398
HMTL puzzling 2×2	3	4.272	6.852	5.563
HMTL puzzling 2×2	4	4.347	7.738	6.043
SL	-	5.338	6.852	6.095

۴-۲. یادگیری خودنظارتی برای تخمین زاویهٔ سر

همان گونه که اشاره شد، هدف این پژوهش، ارزیابی تأثیر سه مورد زیر بر روی تخمین زاویهٔ سر است:

۱. پیش آموزش به روش خودنظارتی؛
۲. پیش آموزش به روش بانظارت بر روی ImageNet؛
۳. اضافه شدن وظیفهٔ کمکی خودنظارتی به بانظارت.

در بخش قبل، مکان مناسب برای جدا شدن سرهای خود نظارت مشخص شد. از آنجا که معماری HopeNet اساس کار ما قرار داده شده است، سرهای (شاخه‌های) بانظارت بر روی قسمت شبکهٔ ResNet معماری HopeNet لحاظ شد. مطابق با تنظیمات مقالهٔ HopeNet، ضریب آلفای که مقدار حساسیت تابع هزینه بر روی خطای طبقه‌بندی نشان می‌دهد، ۲ در نظر گرفته می‌شود. علاوه براین مجموعه دادهٔ 300W-LP [17] برای آموزش تمام شبکه‌ها در نظر گرفته شده است. به منظور اعتبارسنجی نیز از مجموعه‌داده‌های AFLW2000 و BIWI استفاده شد. در همه روش‌ها ۱۱۰ چرخه برای آموزش در نظر گرفته شد. در ابتدا آهنگ یادگیری روی ۰/۰۰۱ تنظیم شد و سپس در چرخه‌های ۳۰ و ۴۰، در ۰/۱ ضرب شد.

در این بخش دو سطح از افزودن در نظر گرفته شد. سطح اول شامل بزرگنمایی تصادفی و تباین[۴۸] تصادفی؛ و در سطح دوم، تارکردن تصادفی، کاهش اندازهٔ تصادفی و «Cutout» نیز اضافه شد. مقادیر Dropout برای سرهای با نظارت ۰/۵ و برای تمام سرهای خودنظارت ۰/۲ قرار داده شد. از آنجا که مقادیر خطای سرهای با نظارت بزرگ‌تر از خودنظارت بودند، سرهای پازل کردن و چرخاندن در عدد ۵۰ ضرب شدند. تابع هزینهٔ معماری HMTL برای روش کمکی پازل کردن مطابق رابطهٔ (۲) لحاظ شده است:

$$\mathcal{L}_{Total} = \mathcal{L}_{SL} + \mathcal{L}_{SSL} = \mathcal{L}_{Cat-Reg} + \mathcal{L}_{puzzle} = (\mathcal{L}_{Cat} + \alpha\mathcal{L}_{Reg}) + \sum_{j} \mathcal{L}_{Puzzle_j}$$

$$= \left(-\sum_{i} y_{Cat_i} log(\hat{y}_{Cat_i}) + \alpha \sqrt{\frac{1}{n}\sum_{j=1}^{n} \left(y_{Reg_j} - E(\hat{y}_{Cat})_j\right)^2} \right)_{yaw}$$

$$+ \left(-\sum_{i} y_{Cat_i} log(\hat{y}_{Cat_i}) + \alpha \sqrt{\frac{1}{n}\sum_{j=1}^{n} \left(y_{Reg_j} - E(\hat{y}_{Cat})_j\right)^2} \right)_{pitch}$$

$$+ \left(-\sum_{i} y_{Cat_i} log(\hat{y}_{Cat_i}) + \alpha \sqrt{\frac{1}{n}\sum_{j=1}^{n} \left(y_{Reg_j} - E(\hat{y}_{Cat})_j\right)^2} \right)_{roll}$$

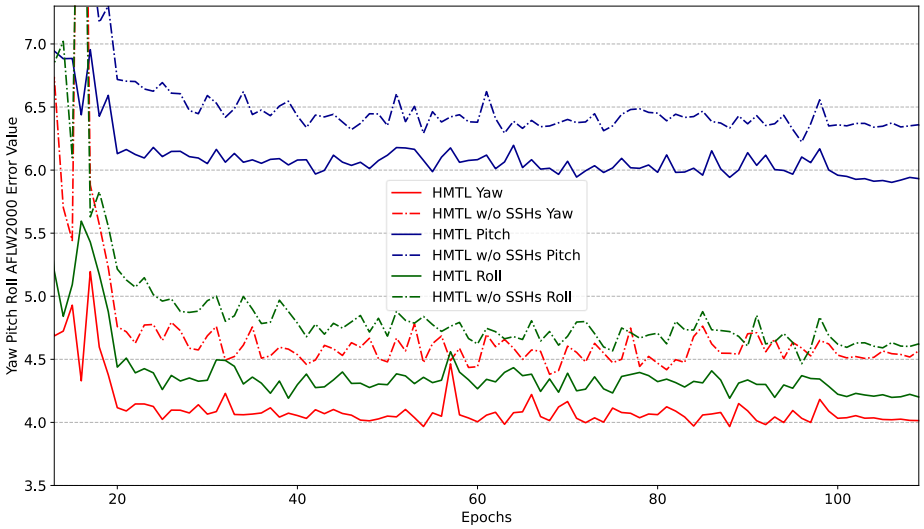
$$+ \sqrt{\frac{1}{n}\sum_{j=1}^{n} \left(y_{Reg_j} - E(\hat{y}_{Cat})_j\right)^2} \quad (۲)$$

که L_{Cat}: تابع هزینهٔ Categorical Cross Entropy برای طبقه‌بندی، L_{Reg}: توابع هزینهٔ RMSE برای سرهای Yaw، Pitch و Roll، $\hat{y}_{Cat}$: خروجی لایهٔ Softmax، y_{Cat}: مقدار حقیقی برچسب طبقه‌بندی برای سرهای Yaw، Pitch و Roll، α: ضریب توابع هزینهٔ رگرسیون نسبت به طبه‌بندی در روش HopeNet.

نکته مهم برای پیش آموزش دادن وزن‌ها این است که پیش آموزش به روش خود نظارتی روی رمزگذار (ResNet50) انجام شده است. نتایج در جدول‌های ۳ و ۴ بر روی مجموعه داده‌های BIWI و AFLW2000 نشان داده شده است. با توجه به این جدول مشاهده می‌شود وظایف کمکی خودنظارتی می‌تواند میانگین میزان خطا را کاهش دهد (شکل ۶). مطابق نتایج روش پازل کردن حتی با وزن‌های پیش آموزش داده شدهٔ چرخاندن، نتایج بهبود یافته است که این نشان دهنده تأثیر استفاده از هر دو رویکرد یادگیری نظارتی همراه با هم است. با این حال بر خلاف روش پازل کردن، اضافه کردن روش کمکی چرخاندن به بانظارت بهتر از روش بانظارت نبوده است. لازم به یادآوری است که برای روش بانظارت (SL) همان معماری HopeNet در نظر گرفته شده است.

۴-۳. تأثیر توابع خطای خودنظارتی در تخمین زاویهٔ سر

تحقیقات نشان می‌دهد ویژگی‌های محلی موجود در تصویر برای مسئلهٔ تخمین زاویهٔ سر مهم‌تر هستند[۳۴] . در روش پیشنهادی، ورودی به صورت دسته‌ای از ویژگی‌هایی فضایی محلی تصویر داده می‌شود و رابطه ویژگی‌های محلی با یکدیگر در تصویر نادیده گرفته می‌شود. به عبارتی نشان داده شد که ویژگی‌های محلی به تنهایی برای تخمین زاویهٔ سر می‌توانند بسیار مؤثر باشند حتی اگر ارتباط آنان حذف شده باشند. با این وجود پرسشی که مطرح می‌شود این است که ویژگی های فضایی محلی و کلی چقدر برای مسئلهٔ تخمین زاویهٔ سر مهم است؟ در ادامه سعی شده است از طریق انجام آزمایشی این موضوع مورد بررسی قرار گیرد. بدین منظور تصاویر سر به صورت تصادفی پازل شدند ولی برای آن‌ها برچسبی در نظر گرفته نشد. در این حالت ارتباط نواحی مختلف تصویر از بین برده شد تا ارزش نواحی محلی به تنهایی مورد ارزیابی قرار گیرند. از همین رو، تصاویر پازل شده به شبکهٔ بانظارت HopeNet داده شدند تا شبکه بر روی آنان آموزش ببینند. این حالت از آموزش بانظارت مشابه فرایند آموزش در HMTL است با این تفاوت که سرهای خودنظارت از آن حذف ولی ورودی همچنان تصاویر پازل شده است.



شکل ۵- مقایسه دو روش یادگیری بانظارت با تصاویر پازل شده (HMTL w/o SSHs)

با روش HMTL پیشنهادی. یادآوری شود که روش اول مشابه با روش دوم است، با این تفاوت که سرهای خودنظارتی و توابع خطای متناظر از آنان حذف شده است. در بقیه موارد هر دو روش در تنظیمات و شرایط یکسان آموزش دیده‌اند.

در واقع در این حالت اهمیت حضور سرهای خودنظارتی در سر بانظارت اصلی بررسی می‌شود. برای این کار روش پایهٔ بانظارت با دو ورودی تصاویر طبیعی و تصاویر پازل شده در کنار روش HMTL پیشنهادی در این پژوهش مقایسه شدند. مطابق حالت قبل، تصاویر مجموعهٔ 300W-LP برای آموزش و تصاویر AFLW2000 برای ارزیابی روش‌ها در نظر گرفته شده است. نتایج هر سه روش در جدول ۲ آمده است. همچنین در شکل ۵ تفاوت بودن و نبودن سرهای خودنظارتی در فرایند ارزیابی شبکه حین آموزش نسبت به سه زاویهٔ Yaw، Pitch و Roll نشان داده شده است.

مشاهده می‌شود که تصاویر پازل شده به تنهایی می‌توانند نسبت به حالتی که تصاویر پازل نشده‌اند باعث کاهش میانگین خطا در تخمین زاویهٔ سر شوند. از سویی، با اضافه شدن توابع خطای خودنظارتی (HMTL)، میانگین خطا باز هم کاهش می‌یابد. این موضوع نشان می‌دهد که هر دو عامل پازل کردن تصاویر و وجود سرهای خودنظارتی در کنار یکدیگر باعث رسیدن به کمترین میزان میانگین خطا شده است.

جدول ۲- مقایسه تأثیر پازل کردن تصاویر ورودی در حالت بانظارت و HMTL. (روش‌ها بر روی تصاویر 300W-LP آموزش دیده‌اند و بر روی تصاویر AFLW2000 ارزیابی شده‌اند)

روش	Yaw (MAE)	Pitch (MAE)	Roll (MAE)	میانگین
SL	6.221	5.569	3.984	5.258
HMTL 3×3 puzzling w/o SSHs	4.589	6.223	4.465	5.092
HMTL 3×3 puzzling	3.874	5.929	4.416	4.74

۵. نتایج

در این پژوهش نشان داده شد که چگونه می‌توان از روش‌های خودنظارتی برای تخمین زاویهٔ سر استفاده کرد. بدین منظور دو راهکار به کار گرفته شد: استفاده از یادگیری خودنظارتی به‌عنوان استخراج‌کننده ویژگی از پیش آموزش داده شده و با استفاده از آن به‌عنوان وظیفهٔ کمکی در کنار وظیفهٔ بانظارت. علاوه بر این نشان داده شد که رویکرد دوم میانگین خطای کمتری داراست. همچنین تلاش شد تا به دو پرسش زیر پاسخ دهیم:

الف. چگونه می توان از روش‌های خودنظارتی برای تخمین زاویهٔ سر (که یک مسئلهٔ Fine-Grained است)، استفاده کرد؟

ب. معماری شبکه به چه شکلی باید باشد تا هنگامی که وظیفهٔ کمکی خودنظارت اضافه می‌شود، بهترین عملکرد حاصل شود؟

برای نخستین پرسش از دو روش اصلی برای استفاده از روش‌های خودنظارتی استفاده شد: ۱. استفاده از وزن‌های روش خودنظارتی به عنوان وزن‌های از پیش آموزش داده شده. ۲. استفاده از روش خودنظارتی به عنوان وظیفه کمکی در کنار روش بانظارت.

هر دو روش توانستند باعث کاهش میانگین خطا شوند اما روش دوم نتایج بهتری را نشان داد. هنگامی که از روش خودنظارتی به عنوان کار کمکی در کنار روش بانظارت استفاده شود، اینکه از چه لایه‌ای به بعد از خودنظارت استفاده شود، بسیار مهم است. پژوهش‌های قبلی نشان می‌دهد که در فرآیند یادگیری چندوظیفه‌ای بعضی از سرهای کمکی می‌توانند به بقیهٔ شبکه کمک کنند و بعضی دیگر نیز می‌توانند باعث آسیب و کم کردن دقت نهایی شبکه شوند. البته دقیقاً مشخص نیست از چه لایه‌ای به بعد بهتر است از وظیفه کمکی استفاده شود، اما در این پژوهش با روش آزمون و خطا و با یافتن مکان مناسب، توانستیم دقت نهایی شبکه را افزایش دهیم.

اشاره به این نکته نیز ضروری است که به دلیل هزینهٔ محاسباتی بالا از روش «NAS[۴۹]» استفاده نشده است، در عوض چهار مکان جداسازی انتخاب شده و از هر کدام به بعد، سرهای بانظارت و خودنظارت از یکدیگر جدا شده‌اند.

آزمودن نقاط مختلف، بهترین نقطه برای تقسیم سرها را به دست می‌دهد. وظیفه‌های خودنظارتی که بر روی داده‌های بدون برچسب آموزش داده شدند، پازل کردن و چرخاندن بوده است. نتایج نشان می دهد که استفاده از سرهای خودنظارتی به عنوان وظیفهٔ کمکی در کنار روش با نظارت بهتر از زمانی است که از وزن‌های آموزش داده شدهٔ پایگاه دادهٔ ImageNet استفاده شود، اگرچه یکی از روش‌های خودنظارتی - چرخش تصاویر - اندکی میانگین خطای سرهای بانظارت شده را افزایش داده است. با وجود اینکه روش چرخاندن تصاویر نتوانسته بود به عنوان یک وظیفهٔ کمکی مؤثر باشد، ولی به عنوان یک روش پیش آموزش وزن‌های اولیه، بسیار مفید بوده است. رویکرد ترکیبی HMTL تنها برای مرحلهٔ آموزش بوده و در زمان ارزیابی یا زمان آزمایش، همهٔ شاخه‌های (سرهای) خودنظارتی را می توان از قسمت رمزگذار شبکه حذف کرد. این بدان معنی است که هیچ تغییری در معماری با نظارت و زمان استنتاج از شبکه ایجاد نخواهد شد.

همچنین در حین این پژوهش پرسش‌هایی نیز مطرح شد که برای تحقیقات آتی می‌توانند مورد توجه قرار گیرند:

ج. بهترین معماری برای روش خودنظارتی با سرهای کمکی کدام است؟

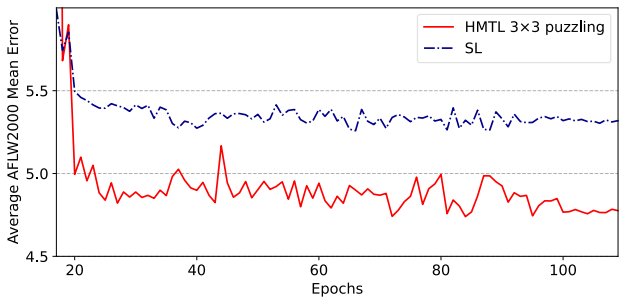
هرچند در این مقاله توانستیم تأثیر نقاط انشعاب متفاوت سرهای شبکه بر میانگین خطای روش بانظارت را نشان دهیم، اما همچنان در این زمینه ابهاماتی وجود دارد که کدام وظیفهٔ کمکی چقدر می‌تواند مفید باشد و کدام وظیفه چقدر می‌تواند آسیب رساند (انتقال منفی[۵۰]) که این موضوع می‌تواند توسط روش NAS به منظور طراحی بهترین معماری بررسی شود.

جدول ۳- نتایج بر روی مجموعه دادهٔ AFLW2000.

روش	سطح افزودن	وزن‌های پیش‌آموزش دیده	Yaw (MAE)	Pitch (MAE)	Roll (MAE)	میانگین
FAN (12 points) [35]	نامشخص	-	18.273	12.604	8.998	13.292
3DDFA [17]	نامشخص	-	5.4	8.53	8.250	7.393
RetinaFace R-50 (5 points) [36]	نامشخص	-	5.101	9.642	3.924	6.222
HopeNet [6]	≈2	-	6.47	6.559	5.436	6.155
Hybrid Coarse-Fine [37]	نامشخص	-	4.82	6.227	5.137	5.395
HPE-40 [20]	نامشخص	-	4.87	6.18	4.8	5.28
FSA-Caps-Fusion [34]	≈2	-	4.5	6.08	4.64	5.07
WHENet [21]	1	-	4.44	5.75	4.31	4.83
TriNet [38]	نامشخص	-	4.198	5.767	4.042	4.669
QuatNet [39]	نامشخص	-	3.97	5.61	3.92	4.5
SL	1	-	5.736	5.907	4.89	5.511
SL	1	ETH-XGaze SSL 2×2 rotation	5.86	5.541	4.113	5.171
SL	1	ImageNet	5.973	5.488	4.191	5.217
SL	2	-	6.221	5.569	3.984	5.258
SL	2	ETH-XGaze SSL 2×2 rotation	6.01	5.432	4.175	5.206
HMTL 2×2 puzzling	1	-	4.22	6.065	5.007	5.094
HMTL 3×3 puzzling	1	-	4.175	5.8	4.951	4.975
HMTL 3×3 puzzling	1	ETH-XGaze SSL 2×2 rotation	3.855	6.065	4.377	4.766
HMTL 3×3 puzzling	1	ImageNet	4.235	6.2	4.231	4.889
HMTL 3×3 puzzling	2	-	3.874	5.929	4.416	4.74
HMTL 3×3 puzzling	2	ETH-XGaze SSL 2×2 rotation	3.682	5.919	4.316	4.639
HMTL 2×2 rotation	1	-	5.998	5.604	4.31	5.304
HMTL 2×2 rotation	1	ETH-XGaze SSL 2×2 puzzling	5.77	5.657	4.104	5.177
HMTL 2×2 puzzling-rotation	1	-	5.554	6.211	5.312	5.692

جدول ۴- نتایج روش‌ها بر روی مجموعه دادهٔ BIWI

روش	وزن‌های پیش‌آموزش دیده	Yaw (MAE)	Pitch (MAE)	Roll (MAE)	میانگین
HopeNet [6]	-	4.810	6.606	3.269	4.895
QuatNet [39]	-	4.01	5.49	2.93	4.14
FSA-Caps-Fusion [34]	-	4.27	4.96	2.76	4.0
WHENet [21]	-	3.60	4.10	2.73	3.48
SL	-	4.322	5.94	3.113	4.458
SL	ImageNet	4.379	5.636	3.002	4.339
SL	ETH-XGaze SSL 2×2 rotation	4.298	5.891	2.782	4.323
HMTL 2×2 rotation	-	4.356	5.801	3.27	4.476
HMTL 3×3 puzzling	-	4.123	5.717	3.059	4.299
HMTL 3×3 puzzling	ImageNet	3.988	5.852	3.01	4.283
HMTL 3×3 puzzling	ETH-XGaze SSL 2×2 rotation	3.564	5.641	2.962	4.056



شکل ۶- مقایسه روش HMTL و بانظارت در میانگین خطای مجموعه دادهٔ AFLW2000 با سطح افزودن دو.

د. هنگام افزودن سرهای کمکی، بهتر است از چه تابع هزینه‌ای استفاده شود؟

وظیفه‌های مختلف در رویکرد چندوظیفه‌ای به معنی توابع هزینهٔ متفاوت است. به عنوان مثال توابع خطای رگرسیون و طبقه‌بندی در مقدار، سرعت یادگیری و کارایی با هم متفاوتند [۴۰]. زمانی که یک شبکه برای بیش از یک وظیفه آموزش داده می‌شود، توابع خطای مختلف در وظایف بایستی در یک تابع هزینهٔ واحد ادغام شوند و مدل آموزش ببیند تا این مقدار را به حداقل برساند. ساده‌ترین روش، وزن‌دهی دستی است که در این تحقیق مورد استفاده قرار گرفته است، اما روش‌های دیگری مانند میانگین خطای هندسی[۵۱] [۳۷] و وزن‌دهی با عدم قطعیت[۵۲] [۳۸] وجود دارند که می‌توانند در نتایج نهایی اثرگذار باشند.

ه. چگونه بهترین وظیفهٔ کمکی خودنظارتی را برای افزودن به وظیفهٔ بانظارتی انتخاب کرد؟

با پیشرفت‌های اخیر در یادگیری خودنظارتی، می‌توان فهمید که اضافه کردن سرهای کمکی (وظایف کمکی) با روش‌های جدید خودنظارتی به بانظارت، بهتر از روش‌های خودنظارتی وظایف pre-text عمل خواهند کرد. به خصوص در مورد روش‌های غیر مغایرتی که نیازی به زوج داده‌های منفی و مثبت برای آموزش ندارند [۱۰, ۲۱]. خصیصه‌های تولید شده به روش‌های خودنظارتی مانند روش‌های غیر مغایرتی، ممکن است ویژگی‌هایی سطح بالاتر و انتزاعی‌تری نسبت به روش بانظارت داشته باشند، همانطور که در روش DINO این موضوع مشاهده شده است [۱۰]. در حقیقت در این حالت سرهای بانظارت می‌توانند مانند وظیفه‌ای کمکی برای ساخت بهتر خصیصه‌های یادگیری خودنظارتی عمل کنند. با این حال، طراحی و ساخت روش‌هایی مانند یادگیری خودنظارتی وظایف pre-text، ساده تر و ارزان‌تر است که در مواردی به نتایج بسیار خوبی منتهی شده است [۴۱].

۶. نتیجه‌گیری

در این مقاله چگونگی استفاده از یادگیری خودنظارتی برای تخمین زاویهٔ مورد بحث و تحلیل قرارگرفت. نتایج نشان داد که در شرایط مختلف استفاده از یادگیری خودنظارتی می‌تواند نتایج متفاوتی به همراه داشته باشد. به تعبیری دیگر بعضی روش‌های خودنظارتی برای یادگیری انتقالی مناسب‌ترند و بعضی دیگر زمانی می‌توانند مؤثر باشند که به عنوان وظیفهٔ کمکی حین فرایند یادگیری بانظارت در نظر گرفته شوند که در این حالت مسئله به حالت یادگیری چند وظیفه‌ای تغییر شکل می‌یابد. از آنجا که در این رویکرد وظایف خودنظارتی و با نظارت در کنار یکدیگر استفاده می‌شود، در این پژوهش آن یادگیری چندوظیفه‌ای ترکیبی یا HMTL نام‌گذاری شده است. از طرفی نتایج این پژوهش مشخص کرد که برخی وظیفه‌های خودنظارتی مانند پازل کردن، زمانی می‌توانند مؤثرتر واقع شوند که به عنوان وظایف کمکی در کنار وظیفهٔ اصلی بانظارت، که در اینجا تخمین زاویهٔ سر بوده است در نظر گرفته شوند.

در نهایت با مقایسه و طراحی بهترین معماری HMTL برای تخمین زاویهٔ سر توسط دو وظیفهٔ خودنظارتی پازل کردن و چرخاندن، مطابق با معماری مشهور HopeNet، میانگین خطا در تخمین زاویهٔ سر به میزان قابل توجهی کاهش پیدا کرد.

مراجع

[1] D. Geronimo, A. M. Lopez, A. D. Sappa, and T. Graf, "Survey of pedestrian detection for advanced driver assistance systems," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 7, pp. 1239–1258, 2009.

[2] Z. Chen *et al.*, "A realistic face-to-face conversation system based on deep neural networks," 2019, doi: 10.1109/ICCVW.2019.00315.

[3] S. S. Mukherjee and N. M. Robertson, "Deep Head Pose: Gaze-Direction Estimation in Multimodal Video," *IEEE Trans. Multimed.*, 2015, doi: 10.1109/TMM.2015.2482819.

[4] K. Cao, Y. Rong, C. Li, X. Tang, and C. C. Loy, "Pose-Robust Face Recognition via Deep Residual Equivariant Mapping," 2018, doi: 10.1109/CVPR.2018.00544.

[5] Z. Liu, Z. Chen, J. Bai, S. Li, and S. Lian, "Facial pose estimation by deep learning from label distributions," 2019, doi: 10.1109/ICCVW.2019.00156.

[6] N. Ruiz, E. Chong, and J. M. Rehg, "Fine-grained head pose estimation without keypoints," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2018, pp. 2074–2083.

[7] R. Valle, J. M. Buenaposada, and L. Baumela, "Multi-Task Head Pose Estimation in-the-Wild," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2021, doi: 10.1109/TPAMI.2020.3046323.

[8] A. Sheka and V. Samun, "Knowledge Distillation from Ensemble of Offsets for Head Pose Estimation," *arXiv Prepr. arXiv2108.09183*, Aug. 2021.

[9] J. B. Grill *et al.*, "Bootstrap your own latent a new approach to self-supervised learning," *Adv. Neural Inf. Process. Syst.*, vol. 33, pp. 21271–21284, 2020.

[10] M. Caron *et al.*, "Emerging properties in self-supervised vision transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 9650–9660.

[11] X. Chen and K. He, "Exploring simple siamese representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 15750–15758.

[12] E. Cole, X. Yang, K. Wilber, O. Mac Aodha, and S. Belongie, "When does contrastive visual representation learning work?," *arXiv Prepr. arXiv2105.05837*, 2021.

[13] Y. M. Asano, C. Rupprecht, and A. Vedaldi, "A critical analysis of self-supervision, or what we can learn from a single image," *arXiv Prepr. arXiv1904.13132*, 2019.

[14] M. Pourmirzaei, G. A. Montazer, and F. Esmaili, "Using Self-Supervised Auxiliary Tasks to Improve Fine-Grained Facial Representation," *arXiv Prepr. arXiv2105.06421.*, May 2021.

[15] X. Yin, X. Yu, K. Sohn, X. Liu, and M. Chandraker,

[16] G. Fanelli, M. Dantone, J. Gall, A. Fossati, and L. Van Gool, “Random Forests for Real Time 3D Face Analysis,” *Int. J. Comput. vision, 101(3), pp.437-458.*, 2013, doi: 10.1007/s11263-012-0549-0.

[17] X. Zhu, Z. Lei, X. Liu, H. Shi, and S. Z. Li, “Face alignment across large poses: A 3d solution,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 146–155.

[18] X. Zhang, S. Park, T. Beeler, D. Bradley, S. Tang, and O. Hilliges, “ETH-XGaze: A Large Scale Dataset for Gaze Estimation Under Extreme Head Pose and Gaze Variation,” 2020, doi: 10.1007/978-3-030-58558-7_22.

[19] K. Khan, R. U. Khan, R. Leonardi, P. Migliorati, and S. Benini, “Head pose estimation: A survey of the last ten years,” *Signal Process. Image Commun.*, vol. 99, p. 116479, 2021.

[20] B. Huang, R. Chen, W. Xu, and Q. Zhou, “Improving head pose estimation using two-stage ensembles with top-k regression,” *Image Vis. Comput.*, 2020, doi: 10.1016/j.imavis.2019.11.005.

[21] Y. Zhou and J. Gregson, “WHENet: Real-time Fine-Grained Estimation for Wide Range Head Pose,” *arXiv Prepr. arXiv2005.10353.*, May 2020.

[22] M. Xin, S. Mo, and Y. Lin, “Eva-gcn: Head pose estimation based on graph convolutional networks,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1462–1471.

[23] V. Albiero, X. Chen, X. Yin, G. Pang, and T. Hassner, “img2pose: Face alignment and detection via 6dof, face pose estimation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 7617–7627.

[24] N. Dhingra, “LwPosr: Lightweight Efficient Fine Grained Head Pose Estimation,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, pp. 1495–1505.

[25] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. Hinton, “Big Self-Supervised Models are Strong Semi-Supervised Learners,” *Advances in neural information processing systems, 33, pp.22243-22255*. 2020.

[26] J. Zbontar, L. Jing, I. Misra, Y. LeCun, and S. Deny, “Barlow Twins: Self-Supervised Learning via Redundancy Reduction,” *arXiv Prepr. arXiv2103.03230*, 2021.

[27] S. K. Mustikovela *et al.*, “Self-Supervised Viewpoint Learning from Image Collections,” 2020, doi: 10.1109/CVPR42600.2020.00403.

[28] X. Zhai, A. Oliver, A. Kolesnikov, and L. Beyer, “S4L: Self-Supervised Semi-Supervised Learning,” *Proc. IEEE/CVF Int. Conf. Comput. Vis. (pp. 1476-1485).*, May 2019.

[29] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[30] S. Gidaris, P. Singh, and N. Komodakis, “Unsupervised representation learning by predicting image rotations,” *arXiv preprint arXiv:1803.07728*. 2018.

[31] F. Zhuang *et al.*, “A comprehensive survey on transfer learning,” *Proc. IEEE*, vol. 109, no. 1, pp. 43–76, 2020.

[32] J. Zhuang *et al.*, “AdaBelief optimizer: Adapting stepsizes by the belief in observed gradients,” *Advances in neural information processing systems, 33, pp.18795-18806*. 2020.

[33] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” *J. Mach. Learn. Res. 15(1), pp.1929-1958.*, 2014.

[34] T. Y. Yang, Y. T. Chen, Y. Y. Lin, and Y. Y. Chuang, “Fsa-net: Learning fine-grained structure aggregation for head pose estimation from a single image,” 2019, doi: 10.1109/CVPR.2019.00118.

[35] V. Kazemi and J. Sullivan, “One millisecond face alignment with an ensemble of regression trees,” 2014, doi: 10.1109/CVPR.2014.241.

[36] J. Deng, J. Guo, E. Ververas, I. Kotsia, and S. Zafeiriou, “Retinaface: Single-shot multi-level face localisation in the wild,” 2020, doi: 10.1109/CVPR42600.2020.00525.

[37] H. Wang, Z. Chen, and Y. Zhou, “Hybrid coarse-fine classification for head pose estimation,” *arXiv Prepr. arXiv1901.06778*, 2019.

[38] Z. Cao, Z. Chu, D. Liu, and Y. Chen, “A vector-based representation to enhance head pose estimation,” 2021, doi: 10.1109/WACV48630.2021.00123.

[39] H. W. Hsu, T. Y. Wu, S. Wan, W. H. Wong, and C. Y. Lee, “Quatnet: Quaternion-based head pose estimation with multiregression loss,” *IEEE Trans. Multimed.*, 2019, doi: 10.1109/TMM.2018.2866770.

[40] M. Crawshaw, “Multi-task learning with deep neural networks: A survey,” *arXiv preprint arXiv:2009.09796*. 2020.

[41] H. Bao, L. Dong, and F. Wei, “BEiT: BERT Pre-Training of Image Transformers,” *arXiv Prepr. arXiv2106.08254*, 2021.

پانویس‌ها

[1] Face detection
[2] Facial Landmarks detection
[3] Facial Age estimation
[4] Head Pose Estimation
[5] Human behavior analysis
[6] Occlusion
[7] Illumination
[8] Facial Landmarks
[9] Key-point
[10] Deep Learning
[11] End-to-End
[12] Self-Supervised Learning (SSL)
[13] Pre-training

[14] Racial
[15] Identity
[16] Multi-Task Learning (MTL)
[17] Hybrid Multi-Task Learning (HMTL)
[18] Auxiliary task
[19] Supervised Learning (SL)
[20] Puzzle
[21] Rotate
[22] Convolutional Neural Network
[23] Wrapped Loss function
[24] Graph Neural Network
[25] Vertexes
[26] Edges
[27] 3D
[28] Depthwise Separable Convolution
[29] Fine-tuning
[30] Contrastive Learning
[31] Non-Contrastive Learning
[32] Rotation
[33] Colorization
[34] Puzzling
[35] Batch
[36] Facial Emotion Recognition (FER)
[37] Gender Recognition
[38] Co-training
[39] Semi-Supervised Learning
[40] Convolution
[41] Encoder
[42] Dense
[43] Transfer Learning
[44] Overfiting
[45] Augmentation
[46] Epoch
[47] Learning rate
[48] Contrast
[49] Neural Architecture Search

بیوگرافی نویسندگان

مهدی پور میرزایی مدرک کارشناسی ارشد خود را در سال ۱۴۰۱ در رشته مهندسی فناوری اطلاعات از دانشگاه تربیت مدرس دریافت کرده و در حال حاضر پژوهشگر حوزهٔ بینایی ماشینی است. تمرکز وی بر استفاده از روشهای یادگیری نیمه نظارتی برای شناسایی احساسات چهره با استفاده از شبکه های عصبی ژرف است.

غلامعلی منتظر مدرک کارشناسی مهندسی برق را در سال ۱۳۷۱ از دانشگاه صنعتی خواجه نصیرالدین طوسی و مدرک کارشناسی ارشد و دکتری مهندسی برق را به‌ترتیب در سال‌های ۱۳۷۳ و ۱۳۷۷ از دانشگاه تربیت مدرس دریافت کرده و از سال ۱۳۷۸ به استخدام همین دانشگاه درآمده و درحال حاضر با مرتبهٔ استادی مهندسی فناوری اطلاعات در این دانشگاه به تدریس و تحقیق اشتغال دارد. زمینه‌های پژوهشی وی شامل طراحی سامانه‌های هوشمند در یادگیری الکترونیکی، رایانش نرم و سیاستگذاری هوشمند در علم و فناوری است.

سید ابراهیم موسوی مدرک کارشناسی ارشد خود را در سال ۱۴۰۱ در رشته مهندسی فناوری اطلاعات از دانشگاه تربیت مدرس دریافت کرده و در حال حاضر پژوهشگر حوزهٔ هوش مصنوعی است. تمرکز وی بر یادگیری ژرف و بینایی رایانه ای است.