دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)

دانشکده ریاضی و علوم کامپیوتر

استاد درس: دکتر مهدی دهقان

زمستان ۱۴۰۴

تشخیص چهره به کمک تجزیه HOSVD

درس جبرخطی عددی

محمدرضا شیخ الاسلامی ۴۰۲۱۳۴۲۳

چکیده

تشخیص چهره از دهه ۱۹۵۰ میلادی به‌عنوان یکی از مسائل مهم در حوزهٔ پردازش تصویر و تشخیص الگو مطرح شده است و پس از دهه ۱۹۹۰، با افزایش توان پردازشی رایانه‌ها، توجه پژوهشگران به این حوزه به‌طور چشمگیری افزایش یافته است. امروزه الگوریتم‌های تشخیص چهره نقش مهمی در کاربردهایی نظیر سیستم‌های امنیتی، کنترل دسترسی، نظارت تصویری و شبکه‌های اجتماعی ایفا می‌کنند و تحقیقات در این زمینه همچنان به‌صورت فعال ادامه دارد.

الگوریتم‌های تشخیص چهره عموماً بر این فرض استوار هستند که چهرهٔ هر فرد دارای ساختاری منحصربه‌فرد است و با استخراج الگوهای ساختاری موجود در تصاویر چهره می‌توان هویت افراد را شناسایی کرد. از این منظر، مسئلهٔ تشخیص چهره را می‌توان به‌عنوان یک مسئلهٔ یافتن الگو در داده‌های تصویری مدل‌سازی کرد که ابزارهای جبر خطی عددی و روش‌های کاهش بُعد نقش اساسی در حل آن دارند. یکی از رویکردهای کلاسیک در این زمینه، استفاده از تجزیهٔ مقدار منفرد (SVD) و الگوریتم Eigenfaces است که با نمایش تصاویر چهره به‌صورت بردار و استخراج زیرفضای کم‌بُعد، امکان تشخیص چهره را فراهم می‌سازد.

با وجود موفقیت روش‌های ماتریسی، تبدیل داده‌های ذاتاً چندبعدی تصویری به بردار، منجر به از دست رفتن بخشی از ساختار درونی داده‌ها می‌شود. به‌منظور رفع این محدودیت، در سال‌های اخیر استفاده از چارچوب جبر چندخطی و نمایش تانسوری داده‌ها مورد توجه قرار گرفته است. در این مقاله، با معرفی مفاهیم پایهٔ تانسورها شامل فولد و آنفولد، ضرب تانسور در مدهای مختلف و رتبهٔ تانسور، از تجزیهٔ مقدار منفرد مرتبه‌بالا (HOSVD) به‌عنوان تعمیمی از SVD برای تانسورها استفاده شده است و بر این اساس، الگوریتم Tensorfaces برای تشخیص چهره معرفی و تشریح می‌شود.

در بخش عملی، الگوریتم‌های Eigenfaces و Tensorfaces (به‌همراه روش‌های LBP و FisherFaces) بر روی دیتاست Extended Yale B پیاده‌سازی و ارزیابی شده‌اند. برای کاهش هزینهٔ محاسباتی، از هر یک از 28 کلاس، تعداد 64 تصویر انتخاب شده (در مجموع 1792 تصویر) و داده‌ها به‌صورت stratified با نسبت 80% آموزش و 20% آزمون تقسیم شده‌اند (1433 نمونهٔ آموزش و 359 نمونهٔ آزمون). عملکرد روش‌ها از نظر دقت تشخیص، زمان اجرا و حساسیت نسبت به هایپرپارامترها مورد مقایسه قرار گرفته است. نتایج نشان می‌دهد که FisherFaces با دقت 85.79% بهترین عملکرد را در میان روش‌های بررسی‌شده به دست آورده و در عین حال کمترین زمان اجرا (0.48s) را دارد. همچنین Tensorfaces مبتنی بر HOSVD با دقت 72.70% عملکرد قابل توجهی ارائه می‌دهد، اما به دلیل تجزیه‌ی تانسوری و اجرای چندین مرحله SVD دارای هزینهٔ زمانی بالاتری (40.66s) است. روش LBP نیز با دقت 71.59% و زمان 20.78s توازن نسبتاً مناسبی میان دقت و هزینهٔ محاسباتی نشان می‌دهد. در مقابل، الگوریتم Eigenfaces با دقت 42.62% حساسیت بیشتری نسبت به تغییرات نورپردازی نشان داده و در دیتاست‌هایی با تغییرات شدید روشنایی مانند Yale، عملکرد ضعیف‌تری نسبت به سایر روش‌ها دارد.

فهرست مطالب

۱ مقدمه ۵
۱.۱ مسئله تشخیص چهره و اهمیت آن ۵
۲.۱ کاربردهای تشخیص چهره ۵
۳.۱ چالش‌های اصلی در تشخیص چهره ۵
۴.۱ نقش کاهش بُعد در پردازش تصویر ۶
۵.۱ گذار از روش‌های ماتریسی به روش‌های تانسوری ۶

۲ تجزیه مقدار منفرد ۷
۱.۲ تعریف کلی SVD ۷
۲.۲ ارتباط SVD با مقادیر ویژه ۸
۳.۲ تفسیر شهودی و هندسی SVD ۸
۴.۲ کاربرد مقادیر منفرد در فشرده‌سازی تصاویر ۹
۵.۲ بازسازی و فشرده‌سازی تصاویر چهره با استفاده از SVD ۱۰

۳ الگوریتم Eigenfaces ۱۱
۱.۳ نمایش تصاویر چهره به‌صورت بردار ۱۱
۲.۳ معرفی الگوریتم Eigenfaces و تاریخچه آن ۱۱
۳.۳ ایده اصلی و نحوه عملکرد الگوریتم Eigenfaces ۱۱
۴.۳ کاربردها، مزایا و محدودیت‌های روش Eigenfaces ۱۲

۴ مفاهیم پایه تانسور ۱۳
۱.۴ تعریف تانسور و مرتبه تانسور ۱۳
۲.۴ نمایش داده‌های تصویری به‌صورت تانسور ۱۳
۳.۴ فولد و آنفولد تانسور ۱۴
۱.۳.۴ بازکردن تانسور در مد n ۱۴
۲.۳.۴ مثال عددی از بازکردن تانسور ۱۴
۳.۳.۴ فولدینگ ۱۵
۴.۳.۴ اهمیت بازکردن تانسور در الگوریتم‌های تانسوری ۱۵
۴.۴ ضرب تانسور در مدهای مختلف ۱۶
۱.۴.۴ تعریف ریاضی ۱۶
۲.۴.۴ ارتباط ضرب n-مد با آنفولد و فولد ۱۶
۳.۴.۴ خواص مهم ضرب n-مد ۱۶
۵.۴ رتبه تانسور و n-rank ۱۷
۶.۴ تفاوت رتبه تانسور و رتبه ماتریس ۱۸
۷.۴ انگیزه استفاده از نمایش تانسوری در تشخیص چهره ۱۸

۵ تجزیه مقدار منفرد مرتبه‌بالا (HOSVD) ۱۹
۱.۵ محاسبه HOSVD بر پایه بازکردن تانسور ۲۰
۲.۵ خواص تانسور هسته ۲۱
۳.۵ مقادیر منفرد در HOSVD ۲۱
۴.۵ HOSVD برش‌خورده ۲۱
۵.۵ تفسیر هندسی HOSVD ۲۱
۶.۵ کاربردهای HOSVD ۲۲

۶ الگوریتم Tensorfaces ۲۲
۱.۶ مدل‌سازی تانسوری داده‌های چهره ۲۲

۲.۶ تجزیه HOSVD تانسور داده ۲۳
۳.۶ بازنویسی تجزیه و تعریف تانسورهای کمکی ۲۳
۴.۶ تفسیر برش‌ها و نمایش خطی تصویر ۲۴
۵.۶ فاز آموزش ۲۵
۶.۶ مسئله تشخیص چهره و کاهش هزینه محاسباتی ۲۵
۷.۶ فاز آزمون ۲۶
۸.۶ مقایسه مفهومی Eigenfaces و Tensorfaces ۲۸

۷ پیاده‌سازی الگوریتم‌ها ۲۹
۱.۷ معرفی دیتاست ۲۹
۲.۷ نمایش تصاویر نمونه ۲۹
۳.۷ پیش‌پردازش تصاویر ۳۰
۴.۷ تقسیم‌بندی داده‌ها و تنظیمات اجرای آزمایش ۳۰
۵.۷ پیاده‌سازی الگوریتم Eigenfaces ۳۱
۱.۵.۷ نمایش و تحلیل Eigenfaces استخراج‌شده ۳۱
۲.۵.۷ بازسازی تصاویر چهره ۳۱
۳.۵.۷ نمونه‌های پیش‌بینی درست و نادرست ۳۲
۶.۷ پیاده‌سازی الگوریتم Tensorfaces مبتنی بر HOSVD ۳۳
۱.۶.۷ Tensorfaces استخراج‌شده ۳۳
۲.۶.۷ بازسازی تصاویر چهره ۳۴
۳.۶.۷ نمونه‌های پیش‌بینی درست و نادرست ۳۵
۷.۷ تنظیم و انتخاب هایپرپارامترها ۳۶
۱.۷.۷ هایپرپارامترهای Eigenfaces ۳۶
۲.۷.۷ هایپرپارامترهای Tensorfaces مبتنی بر HOSVD ۳۷
۳.۷.۷ ملاحظات عملی در انتخاب پارامترها ۳۷

۸ نتایج عددی ۳۸
۱.۸ معیارهای ارزیابی عملکرد ۳۸
۲.۸ دقت تشخیص الگوریتم‌ها ۳۸
۳.۸ زمان اجرای الگوریتم‌ها ۳۹
۱.۳.۸ تحلیل فنی هزینه‌ی محاسباتی Tensorfaces ۳۹
۴.۸ مقایسه‌ی جامع دقت و کارایی محاسباتی ۴۰
۵.۸ بررسی تأثیر هایپرپارامترها ۴۱
۱.۵.۸ Eigenfaces ۴۱
۲.۵.۸ Tensorfaces ۴۲
۶.۸ جمع‌بندی تحلیل نتایج ۴۵

۹ پیوست ۴۶
۱.۹ کد پایتون ۴۶
۲.۹ شبه‌کد الگوریتم‌ها ۴۶
۳.۹ تنظیمات آزمایش‌ها و پارامترها ۴۸

۱۰ منابع ۵۰

۱ مقدمه

تشخیص چهره یکی از مسائل مهم و پرکاربرد در حوزه‌های پردازش تصویر، بینایی ماشین و یادگیری ماشین است که از دهه ۱۹۵۰ میلادی مورد توجه پژوهشگران قرار گرفته است. با افزایش چشمگیر توان پردازشی رایانه‌ها از دهه ۱۹۹۰ به بعد، این حوزه رشد قابل‌توجهی داشته و همچنان یکی از موضوعات فعال پژوهشی به شمار می‌رود.

۱.۱ مسئله تشخیص چهره و اهمیت آن

هدف اصلی روش‌های تشخیص چهره، خودکارسازی فرآیند شناسایی هویت افراد با استفاده از الگوریتم‌های محاسباتی سریع و قابل اعتماد است. به‌طور کلی، فرض می‌شود که یک مجموعهٔ بزرگ از تصاویر شامل چهرهٔ افراد مختلف با تنوعی از حالات چهره، شرایط نورپردازی و زوایای دید در اختیار است. در سطح محاسباتی، هر تصویر دیجیتال به‌صورت یک ماتریس ذخیره می‌شود که هر درایهٔ آن متناظر با یک پیکسل از تصویر بوده و مقدار شدت روشنایی یا رنگ آن پیکسل را نشان می‌دهد. در تصاویر رنگی، این نمایش معمولاً شامل چند کانال رنگی (مانند RGB) است. بدین ترتیب، هر تصویر را می‌توان به‌صورت یک ماتریس حقیقی یا یک بردار با بُعد بسیار بالا مدل‌سازی کرد.

حال مسئلهٔ تشخیص چهره در این چارچوب به این صورت تعریف می‌شود که با دریافت یک تصویر جدید که در پایگاه داده موجود نیست، الگوریتم باید نزدیک‌ترین فرد موجود در مجموعهٔ داده را شناسایی کند. از دیدگاه آماری، این فرآیند نوعی مسئلهٔ تخصیص (Allocation) محسوب می‌شود؛ با این حال، در زمینهٔ پردازش تصویر، اصطلاح «تشخیص» (Recognition) توصیف دقیق‌تری از این فرآیند ارائه می‌دهد. اهمیت این مسئله در آن است که طراحی الگوریتم‌هایی که بتوانند در حضور تغییرات گستردهٔ ظاهری، عملکردی پایدار و دقیق داشته باشند، نیازمند مدل‌سازی مناسب ساختار داده‌های تصویری و استخراج الگوهای نهفته در آن‌ها است.

۲.۱ کاربردهای تشخیص چهره

الگوریتم‌های تشخیص چهره در سال‌های اخیر کاربردهای گسترده‌ای در زندگی روزمره یافته‌اند. از جمله مهم‌ترین این کاربردها می‌توان به موارد زیر اشاره کرد:

- سیستم‌های امنیتی و کنترل دسترسی، مانند باز کردن قفل تلفن‌های هوشمند یا ورود به سامانه‌های حساس
- شناسایی هویت افراد در دستگاه‌های خودپرداز (ATM) و سامانه‌های بانکی
- سیستم‌های نظارتی در فرودگاه‌ها و اماکن عمومی برای شناسایی افراد گمشده یا تحت تعقیب
- ثبت خودکار حضور و غیاب در سازمان‌ها و مراکز آموزشی
- شبکه‌های اجتماعی و سامانه‌های مدیریت تصاویر

در تمامی این کاربردها، سرعت، دقت و پایداری الگوریتم تشخیص چهره اهمیت ویژه‌ای دارد.

۳.۱ چالش‌های اصلی در تشخیص چهره

با وجود پیشرفت‌های قابل توجه در زمینهٔ تشخیص چهره، این مسئله همچنان با چالش‌های متعددی همراه است که طراحی الگوریتم‌های دقیق و پایدار را دشوار می‌سازد. یکی از دلایل اصلی این دشواری، ماهیت پیچیده و چندعاملی تصاویر چهره است؛ به‌طوری که هر تصویر می‌تواند تحت تأثیر عوامل مختلفی قرار گیرد که لزوماً ارتباطی با هویت فرد ندارند.

از جمله چالش‌های مهم در مسیر حل مسئلهٔ تشخیص چهره می‌توان به انتقال و تغییر اندازهٔ چهره در تصویر اشاره کرد. موقعیت قرارگیری چهره در تصویر و فاصلهٔ آن از دوربین می‌تواند باعث تغییر مقیاس و جابه‌جایی چهره شود

که این امر مقایسهٔ مستقیم تصاویر را با دشواری مواجه می‌کند. علاوه بر این، تفاوت‌های ظاهری در تصاویر مختلف نظیر تغییر زاویهٔ دید، وجود یا عدم وجود ریش، تغییر مدل مو و استفاده از عینک یا سایر پوشش‌ها می‌تواند ظاهر چهره را به‌طور قابل توجهی تغییر دهد، در حالی که هویت فرد ثابت باقی می‌ماند.

شرایط روشنایی محیط نیز یکی از عوامل بسیار تأثیرگذار در کیفیت تصاویر چهره محسوب می‌شود. تغییرات نور می‌تواند باعث ایجاد سایه‌ها یا تغییر کنتراست تصویر شده و ویژگی‌های استخراج‌شده از تصویر را تحت تأثیر قرار دهد. علاوه بر این، افزایش سن افراد در بازه‌های زمانی طولانی منجر به تغییرات تدریجی در ساختار چهره می‌شود که تشخیص هویت افراد را در تصاویر قدیمی و جدید دشوارتر می‌سازد.

در نتیجه، یک الگوریتم تشخیص چهرهٔ مطلوب باید قادر باشد ویژگی‌های مرتبط با هویت فرد را از این عوامل مزاحم تفکیک کند و نسبت به تغییرات ناشی از انتقال، مقیاس، زاویهٔ دید، روشنایی و تغییرات ظاهری مقاوم باشد. این نیاز، انگیزه‌ای قوی برای استفاده از روش‌های کاهش بُعد پیشرفته و مدل‌سازی‌های غنی‌تر، نظیر رویکردهای مبتنی بر تانسور و جبر چندخطی، فراهم می‌آورد.

۴.۱ نقش کاهش بُعد در پردازش تصویر

تصاویر دیجیتال معمولاً دارای ابعاد بسیار بالایی هستند و هر تصویر می‌تواند شامل هزاران یا حتی میلیون‌ها پیکسل باشد. کار با چنین داده‌های پرحجمی از نظر محاسباتی هزینه‌بر بوده و ممکن است منجر به کاهش کارایی الگوریتم‌ها شود.به همین دلیل، کاهش بُعد داده‌ها یکی از گام‌های کلیدی در پردازش تصویر و تشخیص چهره محسوب می‌شود.

روش‌های کاهش بُعد مبتنی بر جبر خطی، مانند تجزیهٔ مقدار منفرد ،(SVD) این امکان را فراهم می‌کنند که اطلاعات اصلی تصاویر در یک فضای کم‌بُعد نمایش داده شوند، به‌طوری که بخش عمده‌ای از ساختار و الگوی داده‌ها حفظ شود. این رویکرد نه‌تنها باعث کاهش پیچیدگی محاسباتی می‌شود، بلکه می‌تواند اثر نویز را نیز کاهش داده و دقت تشخیص را بهبود بخشد.

۵.۱ گذار از روش‌های ماتریسی به روش‌های تانسوری

در روش‌های کلاسیک، مجموعهٔ تصاویر چهره معمولاً به‌صورت ماتریس‌هایی از بردارهای تصویر مدل‌سازی می‌شوند که این رویکرد باعث از بین رفتن بخشی از ساختار چندبعدی ذاتی داده‌های تصویری می‌شود. روش‌های کلاسیک تشخیص چهره، نظیر الگوریتم Eigenfaces، معمولاً تصاویر را پس از تبدیل به بردار، با استفاده از ابزارهای جبر خطی ماتریسی تحلیل می‌کنند.هرچند این رویکردها موفقیت‌های قابل توجهی داشته‌اند، اما بردارسازی تصاویر منجر به نادیده گرفتن ساختار چندبعدی و چندعاملی داده‌ها می‌شود.

در مقابل، جبر چندخطی و جبر تانسورها چارچوبی قدرتمند برای تحلیل ساختار چندعاملی مجموعه تصاویر فراهم می‌کنند. نمایش داده‌ها به‌صورت تانسورهای مرتبه‌بالا این امکان را می‌دهد که عواملی نظیر پیکسل‌ها، حالات چهره و هویت افراد به‌صورت هم‌زمان مدل‌سازی شوند.به همین دلیل، روش‌های مبتنی بر تانسور، به‌ویژه تجزیهٔ مقدار منفرد مرتبه‌بالا (HOSVD)، ابزار مناسبی برای تفکیک ویژگی‌های سازندهٔ تصاویر چهره و تحلیل دقیق‌تر آن‌ها به شمار می‌روند. این چارچوب ریاضی، مبنای الگوریتم‌های پیشرفته‌تری نظیر Tensorfaces را تشکیل می‌دهد که در این پروژه مورد بررسی قرار گرفته‌اند.

۲ تجزیه مقدار منفرد

در این بخش، تجزیه مقدار منفرد[1] به‌عنوان یکی از مهم‌ترین ابزارهای جبر خطی عددی معرفی می‌شود. SVD هم از دیدگاه تئوری و هم از نظر کاربردی، نقش اساسی در تحلیل داده‌های با بُعد بالا ایفا می‌کند و در مسائلی نظیر کاهش بُعد، حذف نویز، فشرده‌سازی و به‌ویژه در تشخیص چهره (مانند روش Eigenfaces) به‌طور گسترده مورد استفاده قرار می‌گیرد.

۱.۲ تعریف کلی SVD

تجزیه مقدار منفرد یکی از بنیادی‌ترین روش‌ها برای تحلیل ساختار درونی ماتریس‌ها است. این تجزیه هر ماتریس حقیقی

$$A \in \mathbb{R}^{m\times n}$$

را به صورت حاصل‌ضرب سه ماتریس زیر نمایش می‌دهد:

$$A = U\Sigma V^T$$

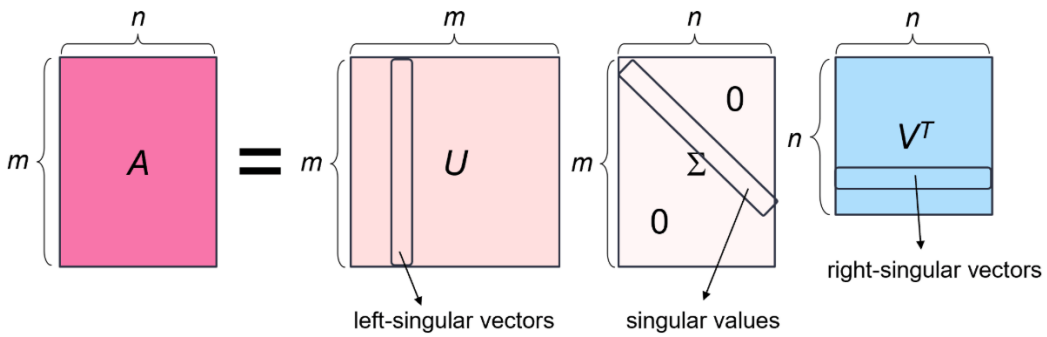
که در آن:

- $U \in \mathbb{R}^{m\times m}$ ماتریسی ارتوگونال است که ستون‌های آن **بردارهای منفرد چپ** نامیده می‌شوند،
- $\Sigma \in \mathbb{R}^{m\times n}$ ماتریسی قطری است که درایه‌های غیرمنفی روی قطر آن را **مقادیر منفرد** می‌نامند،
- $V \in \mathbb{R}^{n\times n}$ ماتریسی ارتوگونال است که ستون‌های آن **بردارهای منفرد راست** هستند.

مقادیر منفرد معمولاً به‌صورت نزولی مرتب می‌شوند:

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0,$$

که در آن $r = \mathrm{rank}(A)$ رتبه‌ی ماتریس است. وجود این تجزیه برای هر ماتریس حقیقی، حتی ماتریس‌های غیرمربعی، تضمین شده است.



شکل ۱: نمایش هندسی تجزیهٔ مقدار منفرد (SVD)

[1] Singular Value Decomposition (SVD)

۲.۲ ارتباط SVD با مقادیر ویژه

یکی از نکات مهم درباره‌ی بردارهای منفرد این است که آن‌ها ارتباط مستقیمی با بردارهای ویژه‌ی ماتریس‌های AA^T و A^TA دارند. اگر تجزیه مقدار منفرد ماتریس A به صورت

$$A = U\Sigma V^T$$

باشد، با ضرب کردن A در ترانهاده‌ی آن خواهیم داشت:

$$AA^T = (U\Sigma V^T)(V\Sigma^T U^T) = U\Sigma\Sigma^T U^T,$$

زیرا $V^TV = I$. در نتیجه:

$$AA^TU = U\Sigma\Sigma^T.$$

اگر ستون‌های ماتریس U را به صورت

$$U = [u_1\ u_2\ \dots\ u_m]$$

در نظر بگیریم، خواهیم داشت:

$$AA^Tu_i = \sigma_i^2 u_i.$$

بنابراین، بردارهای u_i بردارهای ویژه‌ی ماتریس AA^T هستند و σ_i^2 مقادیر ویژه‌ی متناظر با آن‌ها می‌باشند.

به‌طور مشابه، برای ماتریس A^TA داریم:

$$A^TA = V\Sigma^T\Sigma V^T,$$

که منجر به رابطه‌ی زیر می‌شود:

$$A^TA\,v_i = \sigma_i^2 v_i.$$

در نتیجه، بردارهای v_i بردارهای ویژه‌ی ماتریس A^TA بوده و σ_i^2 مقادیر ویژه‌ی متناظر آن‌ها هستند.

۳.۲ تفسیر شهودی و هندسی SVD

از دیدگاه هندسی، تجزیه SVD یک تبدیل خطی را می‌توان به سه عمل ساده تجزیه کرد:

۱. اعمال چرخش یا بازتاب در فضای ورودی توسط V^T و تعیین راستاهای اصلی،

۲. مقیاس‌دهی داده‌ها در راستاهای اصلی توسط Σ،

۳. اعمال چرخش یا بازتاب نهایی برای ورود به فضای خروجی توسط U.

همچنین می‌توان SVD را به صورت جمعی از مؤلفه‌های رتبه‌یک نوشت:

$$A = \sum_{i=1}^{r} \sigma_i\, u_i v_i^T.$$

در این نمایش، هر جمله‌ی $\sigma_i u_i v_i^T$ یک الگوی مستقل از داده را نشان می‌دهد و هرچه مقدار منفرد σ_i بزرگ‌تر باشد، سهم آن مؤلفه در بازسازی ماتریس A بیشتر خواهد بود. این ویژگی دلیل اصلی آن است که مؤلفه‌های اولیه اطلاعات بیشتری از داده را در خود ذخیره می‌کنند.

۴.۲ کاربرد مقادیر منفرد در فشرده‌سازی تصاویر

یکی از خواص کلیدی تجزیه مقدار منفرد، ارتباط مستقیم آن با انرژی داده (به معنای نرم فروبنیوس ماتریس) است که به صورت زیر بیان می‌شود:

$$\|A\|_F^2 = \sum_{i=1}^{p} \sigma_i^2.$$

این رابطه نشان می‌دهد که مجموع مربعات مقادیر منفرد برابر با انرژی کل داده است. در بسیاری از کاربردهای عملی، بخش عمده‌ی این انرژی در چند مقدار منفرد اول متمرکز می‌شود؛ به این معنا که مؤلفه‌های اولیه سهم بیشتری در بازنمایی داده دارند.

این ویژگی مبنای تقریب کم‌رتبه و کاهش بُعد داده‌ها را تشکیل می‌دهد. با نگه داشتن تنها t مقدار منفرد بزرگ‌تر، می‌توان تقریب رتبه-t از ماتریس A را به صورت زیر به‌دست آورد:

$$\tilde{A} = \sum_{i=1}^{t} \sigma_i \, u_i v_i^T.$$

این تقریب، بهترین تقریب کم‌رتبه از نظر کمینه‌سازی خطای فروبنیوس بوده و به‌طور گسترده در کاهش بُعد، حذف نویز و فشرده‌سازی داده‌ها مورد استفاده قرار می‌گیرد.

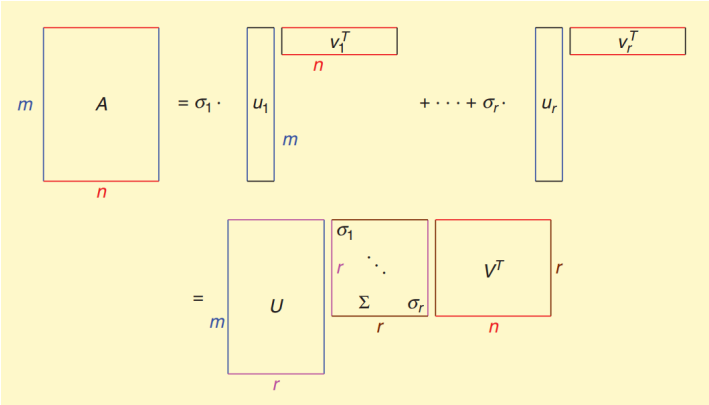
در پردازش تصویر، هر تصویر خاکستری را می‌توان به‌صورت یک ماتریس از شدت روشنایی پیکسل‌ها در نظر گرفت. اگر A ماتریس تصویر باشد و تجزیه SVD آن به‌صورت

$$A = \sum_{i=1}^{r} \sigma_i u_i v_i^T$$

محاسبه شود، با نگه داشتن تنها چند مؤلفه‌ی اول خواهیم داشت:

$$A \approx \sum_{i=1}^{t} \sigma_i u_i v_i^T, \qquad t \ll r.$$

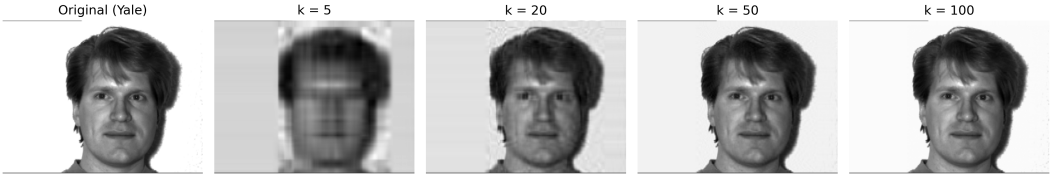
از آنجا که مقادیر منفرد بزرگ‌تر سهم بیشتری در بازسازی تصویر دارند، حذف مؤلفه‌های متناظر با مقادیر منفرد کوچک‌تر معمولاً منجر به حذف نویز و جزئیات کم‌اهمیت می‌شود. با افزایش اهمیت کیفیت تصویر فشرده‌شده، لازم است تعداد بیشتری از جفت‌های بردار منفرد و مقادیر منفرد نگه‌داری شود تا ماتریس بازسازی‌شده بیشتر به ماتریس اصلی A نزدیک گردد. بدین ترتیب، با انتخاب مناسب t می‌توان تعادلی میان کیفیت بازسازی تصویر و میزان فشرده‌سازی و پیچیدگی محاسباتی برقرار کرد.



شکل ۲: نمایش هندسی تجزیهٔ مقدار منفرد (SVD)

۵.۲ بازسازی و فشرده‌سازی تصاویر چهره با استفاده از SVD

برای نمایش اثر SVD در فشرده‌سازی تصاویر چهره، معمولاً تصویری شامل تصویر اصلی و چند بازسازی با مقادیر مختلف t (مانند $t = 5, 20, 50, 100$) ارائه می‌شود:



شکل ۳: تأثیر تعداد مؤلفه‌های SVD در بازسازی تصویر چهره

توضیح: همان‌طور که در شکل مشاهده می‌شود، با افزایش t کیفیت تصویر بازسازی‌شده بهبود می‌یابد، اما حجم اطلاعات و هزینه‌ی محاسباتی نیز افزایش پیدا می‌کند. این رفتار به‌صورت بصری نشان می‌دهد که مؤلفه‌های اولیه‌ی SVD بخش عمده‌ی اطلاعات تصویر چهره را در خود ذخیره می‌کنند.

۳ الگوریتم Eigenfaces

۱.۳ نمایش تصاویر چهره به‌صورت بردار

در مسائل تشخیص چهره، تصاویر دیجیتال به‌صورت داده‌هایی عددی در حافظه‌ی رایانه ذخیره می‌شوند. هر تصویر خاکستری را می‌توان به‌صورت یک ماتریس دوبعدی در نظر گرفت که هر درایه‌ی آن متناظر با شدت روشنایی یک پیکسل از تصویر است. در صورتی که تصویر دارای چند کانال رنگی باشد، می‌توان با ترکیب وزن‌دار کانال‌ها، تصویر را به مقیاس خاکستری تبدیل کرد تا پردازش ساده‌تری روی داده‌ها انجام شود.

برای به‌کارگیری روش‌های جبر خطی، لازم است هر تصویر دوبعدی به‌صورت یک بردار ستونی نمایش داده شود. این کار با کنار هم قرار دادن سطرها یا ستون‌های ماتریس تصویر انجام می‌گیرد. در نتیجه، هر تصویر چهره به‌صورت برداری در فضای با بعد بالا نمایش داده می‌شود و مجموعه‌ای از تصاویر پایگاه داده، مجموعه‌ای از بردارها در این فضا را تشکیل می‌دهند. این نمایش برداری، زمینه‌ی استفاده از روش‌های کاهش بُعد مانند تحلیل مؤلفه‌های اصلی را فراهم می‌کند.

۲.۳ معرفی الگوریتم Eigenfaces و تاریخچه آن

الگوریتم Eigenfaces یکی از نخستین و شناخته‌شده‌ترین روش‌های مبتنی بر جبر خطی برای تشخیص چهره است. این روش برای اولین بار توسط Sirovich و Kirby در سال ۱۹۹۰ با هدف فشرده‌سازی و بازسازی تصاویر چهره معرفی شد و سپس در سال ۱۹۹۱ توسط Turk و Pentland به‌عنوان راهکاری عملی برای مسئله‌ی تشخیص چهره مورد استفاده قرار گرفت.

Eigenfaces در واقع یک کاربرد مستقیم از الگوریتم تحلیل مؤلفه‌های اصلی (PCA) در حوزه‌ی پردازش تصویر است. ایده‌ی اصلی این روش بر این فرض استوار است که تصاویر چهره‌ی افراد مختلف، علی‌رغم تفاوت‌های ظاهری، در یک زیرفضای کم‌بعد از فضای کل تصاویر قرار می‌گیرند. با یافتن این زیرفضای کم‌بعد، می‌توان نمایش فشرده‌ای از چهره‌ها به دست آورد که اطلاعات اصلی را حفظ کرده و در عین حال ابعاد داده را به‌طور چشمگیری کاهش دهد.

۳.۳ ایده اصلی و نحوه عملکرد الگوریتم Eigenfaces

الگوریتم Eigenfaces بر این فرض اساسی استوار است که اگرچه تصاویر چهره‌ی افراد مختلف در فضای بسیار پُربُعدی نمایش داده می‌شوند، اما ساختار ذاتی این تصاویر در یک زیرفضای کم‌بعد نهفته است. به بیان دیگر، مجموعه‌ی تصاویر چهره را می‌توان به‌صورت نقاطی در یک فضای برداری با بعد بسیار بالا در نظر گرفت که در عمل روی یک زیرفضای کم‌بعد متمرکز شده‌اند. هدف اصلی ،Eigenfaces یافتن این زیرفضا و نمایش فشرده‌ی تصاویر در آن است.

در گام نخست، تصاویر چهره‌ی موجود در پایگاه داده پس از تبدیل به مقیاس خاکستری و نمایش برداری، مورد پیش‌پردازش قرار می‌گیرند. برای کاهش اثرات روشنایی کلی، میانگین تصاویر محاسبه شده و از تمام تصاویر کم می‌شود. این کار باعث می‌شود که تحلیل بعدی بیشتر بر تفاوت‌های ساختاری میان چهره‌ها متمرکز شود تا بر سطح کلی روشنایی تصاویر.

در ادامه، ماتریسی تشکیل می‌شود که ستون‌های آن بردارهای متناظر با تصاویر آموزش هستند. با اعمال تجزیه مقدار منفرد (SVD) یا به‌طور معادل تحلیل مؤلفه‌های اصلی (PCA) بر روی این ماتریس، مجموعه‌ای از بردارهای ویژه به دست می‌آیند که به آن‌ها چهره‌های ویژه (Eigenfaces) گفته می‌شود. این بردارها جهت‌هایی در فضای تصاویر هستند که بیشترین واریانس داده‌ها را در خود جای داده‌اند. به‌طور معمول، نخستین چهره‌های ویژه نمایانگر ساختار کلی صورت انسان بوده و مؤلفه‌های بعدی تغییرات جزئی‌تری مانند حالات چهره یا شرایط نوری را مدل می‌کنند.

هر تصویر چهره، چه از مجموعه‌ی آموزش و چه تصویر ورودی جدید، می‌تواند به‌صورت ترکیب خطی از تعداد محدودی از این چهره‌های ویژه تقریب زده شود. ضرایب این ترکیب خطی، مختصات تصویر در فضای چهره‌های ویژه را تشکیل

می‌دهند و در واقع نمایش فشرده‌ی تصویر محسوب می‌شوند. این نمایش کم‌بعد، اطلاعات اصلی تصویر را حفظ کرده و در عین حال پیچیدگی محاسباتی را به‌طور چشمگیری کاهش می‌دهد.

فرآیند تشخیص چهره در Eigenfaces بر مبنای مقایسه‌ی این نمایش‌های کم‌بعد انجام می‌شود. تصویر ورودی ابتدا به فضای چهره‌های ویژه نگاشته شده و سپس فاصله‌ی آن با تصاویر موجود در پایگاه داده یا نمایش‌های مرجع هر شخص محاسبه می‌گردد. معیار تصمیم‌گیری معمولاً فاصله‌ی اقلیدسی در فضای ویژگی‌هاست و تصویر ورودی به شخصی نسبت داده می‌شود که کمترین فاصله را داشته باشد. در صورتی که این فاصله از یک آستانه‌ی مشخص بزرگ‌تر باشد، تصویر به‌عنوان چهره‌ی ناشناخته در نظر گرفته می‌شود.

۴.۳ کاربردها، مزایا و محدودیت‌های روش Eigenfaces

روش Eigenfaces به دلیل سادگی مفهومی و کارایی مناسب، در بسیاری از کاربردهای اولیه‌ی تشخیص چهره مورد استفاده قرار گرفته است. از جمله کاربردهای آن می‌توان به سیستم‌های شناسایی هویت، کنترل دسترسی و بازیابی تصاویر چهره اشاره کرد. همچنین این روش نقش مهمی در توسعه‌ی الگوریتم‌های پیشرفته‌تر تشخیص چهره ایفا کرده است.

از مهم‌ترین مزایای Eigenfaces می‌توان به کاهش چشمگیر بُعد داده‌ها، کاهش حجم محاسبات و حساسیت کمتر به نویز اشاره کرد. انجام محاسبات در فضای کم‌بعد باعث افزایش سرعت و کارایی الگوریتم می‌شود.

با این حال، Eigenfaces دارای محدودیت‌هایی نیز هست. این روش نسبت به تغییرات شدید نور، زاویه دید، حالات چهره و افزایش سن حساس است. علاوه بر این، نمایش تصاویر به‌صورت بردار باعث از دست رفتن ساختار چندبعدی ذاتی داده‌های تصویری می‌شود. این محدودیت‌ها انگیزه‌ای برای توسعه‌ی روش‌های مبتنی بر جبر چندخطی و نمایش تانسوری داده‌ها فراهم کرده‌اند که در آن‌ها عوامل مختلف مانند شخص، حالت چهره و شرایط نوری به‌صورت هم‌زمان مدل‌سازی می‌شوند.

۴ مفاهیم پایه تانسور

۱.۴ تعریف تانسور و مرتبه تانسور

در جبر خطی کلاسیک، داده‌ها معمولاً به‌صورت بردار یا ماتریس نمایش داده می‌شوند. با این حال، در بسیاری از مسائل واقعی، به‌ویژه در پردازش تصویر و تشخیص الگو، داده‌ها دارای ساختاری چندبعدی هستند و نمایش آن‌ها تنها با بردار یا ماتریس منجر به از دست رفتن بخشی از اطلاعات ساختاری می‌شود. برای رفع این محدودیت، مفهوم **تانسور** به‌عنوان تعمیمی از بردار و ماتریس معرفی شده است.

تعریف ۱.۴. یک تانسور با درایه‌های حقیقی و ابعاد $n_1 \times n_2 \times \cdots \times n_k$ عنصری از فضای $\mathbb{R}^{n_1 \times n_2 \times \cdots \times n_k}$ است و به آن **تانسور مرتبه** k گفته می‌شود. هر درایه‌ی این تانسور با k اندیس مشخص می‌شود و به‌صورت $a_{i_1,i_2,\ldots,i_k}$ نشان داده می‌شود.

بردار و **ماتریس** حالت‌های خاصی از تانسورها هستند. اگر $k = 1$ باشد، تانسور متعلق به فضای $\mathbb{R}^{n_1}$ بوده و همان **بردار** است. اگر $k = 2$ باشد، تانسور متعلق به فضای $\mathbb{R}^{n_1 \times n_2}$ در نظر گرفته می‌شود که همان **ماتریس** است. برای $k \geq 3$، تانسور یک آرایه‌ی چندبعدی است که دیگر نمی‌توان آن را بدون از دست دادن ساختار ذاتی داده، به‌صورت یک ماتریس دوبعدی نمایش داد.

مرتبه[۲] **یا تعداد مدهای تانسور** برابر با تعداد ابعادی است که برای مشخص کردن هر درایه‌ی تانسور نیاز داریم. به‌عنوان مثال، اگر $\mathcal{A} \in \mathbb{R}^{2\times4\times3}$ باشد، آنگاه $\mathcal{A}$ یک تانسور مرتبه‌ی ۳ با سه مد است. هر درایه‌ی این تانسور به‌صورت $a_{i,j,k}$ نمایش داده می‌شود که اندیس‌های آن به‌ترتیب متناظر با مد اول، دوم و سوم هستند.

۲.۴ نمایش داده‌های تصویری به‌صورت تانسور

در پردازش تصویر، هر تصویر دیجیتال معمولاً به‌صورت یک آرایه از مقادیر شدت روشنایی ذخیره می‌شود که هر مقدار متناظر با یک پیکسل از تصویر است. در حالت تصاویر خاکستری، هر تصویر را می‌توان به‌صورت یک ماتریس در نظر گرفت. با این حال، زمانی که پایگاه داده شامل تصاویر افراد مختلف با حالات چهره یا شرایط نوری گوناگون باشد، نمایش ماتریسی قادر به حفظ ساختار چندعاملی داده‌ها نخواهد بود.
برای حل این مشکل، تصاویر می‌توانند به‌صورت **تانسور** مدل‌سازی شوند. این نمایش اجازه می‌دهد عوامل مختلف داده به‌طور هم‌زمان و بدون از دست رفتن وابستگی‌های ذاتی آن‌ها در نظر گرفته شوند.

فرض کنید یک پایگاه داده شامل اطلاعات زیر باشد:

- تصاویر خاکستری با P پیکسل،
- E حالت مختلف چهره یا شرایط نوری،
- N شخص متفاوت.

در این حالت، کل پایگاه داده را می‌توان به‌صورت یک تانسور مرتبه‌ی ۳

$$\mathcal{X} \in \mathbb{R}^{P\times E\times N}$$

نمایش داد.
در این نمایش:

- مد اول متناظر با پیکسل‌های تصویر،
- مد دوم متناظر با حالات مختلف چهره یا نور،
- مد سوم متناظر با هویت افراد است.

هر درایه‌ی $\mathcal{X}(i,j,k)$ شدت روشنایی پیکسل i-ام از تصویر مربوط به حالت j-ام از شخص k-ام را نشان می‌دهد.

[2] Order

۳.۴ فولد و آنفولد تانسور

در بسیاری از الگوریتم‌های مبتنی بر تانسور، از جمله تجزیه‌ی HOSVD، لازم است که یک تانسور چندبعدی به‌صورت یک ماتریس دوبعدی نمایش داده شود. این فرایند که **بازکردن تانسور**، **ماتریس‌سازی** [۳] یا **آنفولدینگ** [۴] نامیده می‌شود، به معنای مرتب‌سازی درایه‌های یک تانسور N-بعدی در قالب یک ماتریس است. عمل معکوس این فرایند، یعنی بازگرداندن ماتریس به ساختار تانسوری اولیه، **فولدینگ** [۵] نام دارد.

به‌طور کلی، یک تانسور را می‌توان به روش‌های مختلفی به ماتریس تبدیل کرد، اما در این پژوهش تنها از **بازکردن در مد** n**ام** استفاده می‌شود، زیرا این نوع بازکردن مستقیماً در تجزیه‌ی HOSVD به کار می‌رود.

۱.۳.۴ بازکردن تانسور در مد n

فرض کنید $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ یک تانسور مرتبه‌ی N باشد. بازکردن تانسور در مد nام که با $\mathcal{X}_{(n)}$ نمایش داده می‌شود، به‌گونه‌ای انجام می‌گیرد که **فیبرهای مد** n به‌صورت ستون‌های یک ماتریس کنار هم قرار گیرند. در نتیجه، ماتریس حاصل ابعاد

$$I_n \times \prod_{k \neq n} I_k$$

خواهد داشت.

به‌صورت صوری، هر درایه‌ی تانسور با اندیس‌های $(i_1, i_2, \ldots, i_N)$ در ماتریس بازشده به درایه‌ی (i_n, j) نگاشت می‌شود، که در آن:

$$j = 1 + \sum_{k \neq n} (i_k - 1)\, J_k, \qquad J_k = \prod_{m<k,\, m \neq n} I_m.$$

اگرچه این رابطه از نظر ریاضی دقیق است، اما درک آن از طریق مثال عددی بسیار ساده‌تر خواهد بود.

۲.۳.۴ مثال عددی از بازکردن تانسور

فرض کنید $\mathcal{X} \in \mathbb{R}^{3\times4\times2}$ دارای دو برش جلویی به‌صورت زیر باشد:

$$X_1 = \begin{bmatrix} 1 & 4 & 7 & 10 \\ 2 & 5 & 8 & 11 \\ 3 & 6 & 9 & 12 \end{bmatrix}, \qquad X_2 = \begin{bmatrix} 13 & 16 & 19 & 22 \\ 14 & 17 & 20 & 23 \\ 15 & 18 & 21 & 24 \end{bmatrix}.$$

بازکردن در مد اول: در بازکردن مد اول، فیبرهای مد اول به‌عنوان ستون‌های ماتریس در نظر گرفته می‌شوند. در این حالت:

$$\mathcal{X}_{(1)} \in \mathbb{R}^{3\times8}$$

و خواهیم داشت:

$$\mathcal{X}_{(1)} = \begin{bmatrix} 1 & 4 & 7 & 10 & 13 & 16 & 19 & 22 \\ 2 & 5 & 8 & 11 & 14 & 17 & 20 & 23 \\ 3 & 6 & 9 & 12 & 15 & 18 & 21 & 24 \end{bmatrix}.$$

بازکردن در مد دوم: در بازکردن مد دوم، فیبرهای مد دوم کنار هم قرار می‌گیرند. در این حالت:

$$\mathcal{X}_{(2)} \in \mathbb{R}^{4\times6}$$

[3] Matricization
[4] Unfolding
[5] Folding

و داریم:

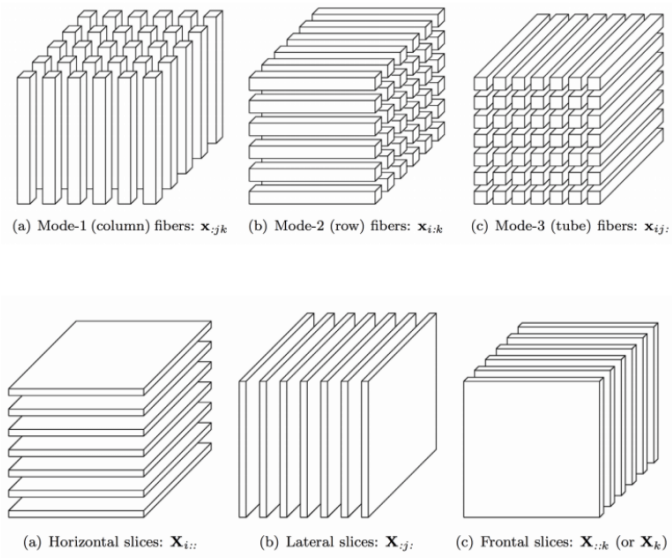
$$\mathcal{X}_{(2)} = \begin{bmatrix} 1 & 2 & 3 & 13 & 14 & 15 \\ 4 & 5 & 6 & 16 & 17 & 18 \\ 7 & 8 & 9 & 19 & 20 & 21 \\ 10 & 11 & 12 & 22 & 23 & 24 \end{bmatrix}.$$

بازکردن در مد سوم: در بازکردن تانسور در مد سوم، فیبرهای مد سوم به‌عنوان ستون‌های ماتریس در نظر گرفته می‌شوند. در این حالت:

$$\mathcal{X}_{(3)} \in \mathbb{R}^{2\times 12}$$

و بازکردن مد سوم به‌صورت زیر است:

$$\mathcal{X}_{(3)} = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & \cdots & 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 & 17 & \cdots & 21 & 22 & 23 & 24 \end{bmatrix}.$$



شکل ۴: نمونه‌ای از فیبرها و برش‌های یک تانسور مرتبه‌ی ۳

۳.۳.۴ فولدینگ

عمل فولدینگ عکس فرایند بازکردن (Unfolding) است و به معنای بازگرداندن ماتریس بازشده به ساختار تانسوری اولیه با ابعاد مشخص می‌باشد. به‌طور دقیق، اگر ماتریس بازشده‌ی $\mathcal{X}_{(n)}$ و ابعاد اولیه‌ی تانسور $\mathcal{X} \in \mathbb{R}^{I_1\times\cdots\times I_N}$ معلوم باشند، می‌توان با اعمال ترتیب معکوس اندیس‌ها، تانسور اولیه را بدون از دست دادن اطلاعات بازیابی کرد.
این فرایند تضمین می‌کند که عملیات بازکردن و بستن تانسور، در صورتی که ترتیب اندیس‌ها به‌صورت سازگار انتخاب شده باشد، یک تبدیل وارون‌پذیر است. بنابراین، Folding و Unfolding صرفاً تغییر نمایش داده‌ها هستند و هیچ‌گونه تغییر ذاتی در محتوای اطلاعاتی تانسور ایجاد نمی‌کنند.

۴.۳.۴ اهمیت بازکردن تانسور در الگوریتم‌های تانسوری

بازکردن تانسور نقش کلیدی در الگوریتم‌های تجزیه‌ی تانسوری دارد. به‌عنوان مثال، در تجزیه‌ی **HOSVD**، برای هر مد n، ابتدا ماتریس بازشده‌ی $\mathcal{X}_{(n)}$ ساخته شده و سپس تجزیه‌ی مقدار منفرد بر روی آن اعمال می‌شود. بدین ترتیب، بازکردن تانسور پل ارتباطی میان جبر خطی کلاسیک و آنالیز چندخطی داده‌ها محسوب می‌شود.

۴.۴ ضرب تانسور در مدهای مختلف

یکی از مهم‌ترین عملیات در جبر چندخطی، ضرب تانسور در یک ماتریس در امتداد یکی از مدها است که با عنوان n-mode product شناخته می‌شود. این عمل، تعمیم طبیعی ضرب ماتریس در بردار یا ماتریس در فضای تانسوری است و نقش اساسی در تجزیه‌های تانسوری، از جمله تجزیه‌ی HOSVD، ایفا می‌کند.

۱.۴.۴ تعریف ریاضی

فرض کنید $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ یک تانسور مرتبه‌ی N و $\mathbf{U} \in \mathbb{R}^{J \times I_n}$ یک ماتریس باشد. ضرب تانسور $\mathcal{A}$ در ماتریس $\mathbf{U}$ در مد n، با نماد $\mathcal{A} \times_n \mathbf{U}$ نمایش داده می‌شود و نتیجه‌ی آن تانسوری با ابعاد

$$\mathbb{R}^{I_1 \times \cdots \times I_{n-1} \times J \times I_{n+1} \times \cdots \times I_N}$$

خواهد بود، به‌طوری که درایه‌های آن به‌صورت زیر تعریف می‌شوند:

$$(\mathcal{A} \times_n \mathbf{U})_{i_1,\ldots,i_{n-1},j,i_{n+1},\ldots,i_N} = \sum_{i_n=1}^{I_n} a_{i_1,\ldots,i_n,\ldots,i_N}\, u_{j,i_n}.$$

۲.۴.۴ ارتباط ضرب n-مد با آنفولد و فولد

یکی از خواص بسیار مهم ضرب n-مد تانسور این است که این عملیات را می‌توان به‌صورت معادل یک ضرب ماتریسی ساده بر روی بازکردن تانسور در همان مد بیان کرد. این ویژگی نقش اساسی در پیاده‌سازی عملی تجزیه‌های تانسوری، به‌ویژه تجزیه‌ی HOSVD، ایفا می‌کند.
فرض کنید $\mathcal{A} \in \mathbb{R}^{I_1 \times \cdots \times I_n \times \cdots \times I_N}$ یک تانسور مرتبه‌ی N و $\mathbf{U} \in \mathbb{R}^{J \times I_n}$ یک ماتریس باشد. اگر

$$\mathcal{B} = \mathcal{A} \times_n \mathbf{U}$$

باشد، آنگاه

$$\mathcal{B} \in \mathbb{R}^{I_1 \times \cdots \times I_{n-1} \times J \times I_{n+1} \times \cdots \times I_N}.$$

اگر $\mathcal{A}_{(n)}$ آنفولد تانسور $\mathcal{A}$ در مد n باشد، رابطه‌ی بسیار مهم زیر برقرار است:

$$\boxed{\mathcal{B}_{(n)} = \mathbf{U}\,\mathcal{A}_{(n)}}$$

یعنی آنفولد تانسور حاصل از ضرب n-مد، برابر است با ضرب ماتریسی $\mathbf{U}$ در آنفولد مد-n تانسور اولیه.

در نتیجه، محاسبه‌ی ضرب n-مد را می‌توان در سه مرحله‌ی متوالی خلاصه کرد:

۱. آنفولد کردن تانسور $\mathcal{A}$ در مد n و تشکیل ماتریس $\mathcal{A}_{(n)}$،

۲. انجام ضرب ماتریسی $\mathbf{U}\mathcal{A}_{(n)}$،

۳. بازگرداندن نتیجه به ساختار تانسوری با استفاده از عمل فولدینگ.

به‌صورت فشرده، این فرآیند به شکل زیر نوشته می‌شود:

$$\boxed{\mathcal{A} \times_n \mathbf{U} = \mathrm{fold}_n\big(\mathbf{U}\,\mathcal{A}_{(n)}\big)}$$

۳.۴.۴ خواص مهم ضرب n-مد

ضرب n-مد دارای خواص مهم زیر است:

- اگر $m \neq n$ باشد، ترتیب ضرب اهمیتی ندارد:

$$\mathcal{A} \times_n \mathbf{U} \times_m \mathbf{V} = \mathcal{A} \times_m \mathbf{V} \times_n \mathbf{U}.$$

- اگر دو ضرب روی یک مد انجام شوند:

$$\mathcal{A} \times_n \mathbf{U} \times_n \mathbf{V} = \mathcal{A} \times_n (\mathbf{V}\mathbf{U}).$$

این خواص، پایه‌ی اصلی تعریف تجزیه‌ی HOSVD هستند که در آن تبدیل‌های متعامد به‌صورت جداگانه روی هر مد اعمال می‌شوند.

۵.۴ رتبه تانسور و n-rank

مفهوم رتبه در تانسورها تعمیمی از مفهوم رتبه‌ی ماتریس‌ها است، اما برخلاف حالت ماتریسی، رفتاری پیچیده‌تر دارد و دارای تعاریف متعددی است. در این بخش، ابتدا رتبه‌ی تانسور و سپس مفهوم n-rank معرفی می‌شوند.

رتبه‌ی تانسور: یک تانسور مرتبه‌ی N

$$\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$$

دارای رتبه‌ی یک است اگر بتوان آن را به‌صورت حاصل‌ضرب خارجی N بردار نوشت، یعنی:

$$\mathcal{A} = \mathbf{a}^{(1)} \circ \mathbf{a}^{(2)} \circ \cdots \circ \mathbf{a}^{(N)},$$

به‌طوری که هر درایه‌ی تانسور برابر باشد با

$$a_{i_1,i_2,\ldots,i_N} = a^{(1)}_{i_1}\, a^{(2)}_{i_2} \cdots a^{(N)}_{i_N}.$$

مفهوم تانسور رتبه‌ی یک مشابه مفهوم ماتریس‌های رتبه‌ی یک است که به‌صورت حاصل‌ضرب یک بردار ستونی و یک بردار سطری نمایش داده می‌شوند. بر این اساس، رتبه‌ی یک تانسور دلخواه $\mathcal{A}$ به‌صورت حداقل تعداد تانسورهای رتبه‌ی یک تعریف می‌شود که بتوان $\mathcal{A}$ را به‌صورت ترکیب خطی آن‌ها نوشت:

$$\operatorname{rank}(\mathcal{A}) = \min\left\{ R \;\middle|\; \mathcal{A} = \sum_{r=1}^{R} \mathbf{a}^{(1)}_r \circ \mathbf{a}^{(2)}_r \circ \cdots \circ \mathbf{a}^{(N)}_r \right\}.$$

n-rank تانسور: به‌منظور ساده‌سازی تحلیل، به‌جای رتبه‌ی کلی تانسور، مفهوم n-rank معرفی می‌شود. n-rank تانسور $\mathcal{A}$ برابر است با رتبه‌ی ماتریسی آنفولد کردن تانسور در مد n.
به‌صورت دقیق:

$$\operatorname{rank}_n(\mathcal{A}) = \operatorname{rank}\big(\mathcal{A}_{(n)}\big), \qquad n = 1, 2, \ldots, N.$$

در نتیجه، n-rank تانسور $\mathcal{A}$ به‌صورت یک بردار تعریف می‌شود:

$$\big(\operatorname{rank}_1(\mathcal{A}),\, \operatorname{rank}_2(\mathcal{A}),\, \ldots,\, \operatorname{rank}_N(\mathcal{A})\big).$$

این تعریف نشان می‌دهد که هر مد تانسور می‌تواند ساختار خطی متفاوتی داشته باشد و بنابراین تحلیل تانسور باید به‌صورت چندمدی انجام شود.

نقش n-rank در تجزیه HOSVD: در تجزیه‌ی HOSVD، هر بازکردن مد-n تانسور به‌طور جداگانه تحت تجزیه‌ی مقدار منفرد قرار می‌گیرد و ابعاد هسته‌ی تانسور مستقیماً به n-rank در هر مد وابسته است. به همین دلیل، n-rank نقش اساسی در کاهش بُعد، فشرده‌سازی و تحلیل داده‌های چندبعدی ایفا می‌کند.

۶.۴ تفاوت رتبه تانسور و رتبه ماتریس

در حالت ماتریسی، رتبه‌ی یک ماتریس به‌صورت تعداد بیشینه‌ی ستون‌ها (یا سطرهای) مستقل خطی تعریف می‌شود و همواره دارای خواص ساده و خوش‌رفتار است. به‌عنوان مثال، رتبه‌ی ماتریس را می‌توان به‌سادگی با استفاده از تجزیه‌ی مقدار منفرد یا محاسبه‌ی مقادیر ویژه تعیین کرد.
در مقابل، رتبه‌ی تانسور رفتاری بسیار پیچیده‌تر دارد. برخلاف ماتریس‌ها، برای تانسورها مفاهیم مختلفی از رتبه وجود دارد و این مفاهیم لزوماً معادل یکدیگر نیستند. علاوه بر این، رتبه‌ی یک تانسور ممکن است برحسب میدان اعداد حقیقی یا مختلط متفاوت باشد و در حالت کلی محاسبه‌ی رتبه‌ی دقیق تانسور یک مسئله‌ی NP-hard است.
تفاوت مهم دیگر آن است که در ماتریس‌ها رتبه‌ی ماتریس به‌طور مستقیم با ساختار زیرفضای سطری و ستونی مرتبط است، در حالی که در تانسورها هر مد می‌تواند دارای ساختار خطی متفاوتی باشد. به همین دلیل، به‌جای یک عدد یکتا، از برداری از رتبه‌ها موسوم به n-rank استفاده می‌شود که رتبه‌ی هر مد را به‌صورت جداگانه توصیف می‌کند.
این تفاوت بنیادی باعث می‌شود که تعمیم روش‌های کلاسیک ماتریسی به داده‌های چندبعدی به‌طور مستقیم امکان‌پذیر نباشد و نیاز به ابزارهای ویژه‌ی جبر چندخطی، مانند تجزیه‌ی HOSVD، احساس شود.

۷.۴ انگیزه استفاده از نمایش تانسوری در تشخیص چهره

در مسائل تشخیص چهره، داده‌ها ذاتاً دارای ساختار چندعاملی هستند. هر تصویر چهره نه‌تنها به هویت شخص وابسته است، بلکه تحت تأثیر عواملی نظیر حالت صورت، زاویه دید، شرایط نوری و حتی زمان ثبت تصویر قرار دارد. نمایش ماتریسی تصاویر، که در آن هر تصویر به‌صورت یک بردار بلند در نظر گرفته می‌شود، این ساختار چندبعدی را نادیده می‌گیرد.
نمایش تانسوری این امکان را فراهم می‌کند که هر یک از عوامل مؤثر بر تصویر چهره به‌عنوان یک مد مستقل مدل‌سازی شود. برای مثال، یک پایگاه داده‌ی چهره را می‌توان به‌صورت تانسوری با مدهای «پیکسل»، «حالت چهره» و «هویت شخص» نمایش داد. این نمایش، وابستگی‌های درونی بین تصاویر را به‌شکل طبیعی‌تری حفظ می‌کند.
از منظر الگوریتمی، استفاده از نمایش تانسوری موجب بهبود عملکرد روش‌های تشخیص چهره در شرایط واقعی می‌شود؛ زیرا تغییرات ناشی از نور یا حالت چهره به‌جای آن‌که به‌صورت نویز در نظر گرفته شوند، در مدهای جداگانه مدل می‌شوند. این موضوع به‌ویژه در روش‌هایی مانند Tensorfaces که مبتنی بر تجزیه‌ی HOSVD هستند، نقش کلیدی دارد.
در نهایت، انگیزه‌ی اصلی استفاده از نمایش تانسوری در تشخیص چهره، دستیابی به مدلی است که همزمان قادر به کاهش بُعد، حفظ ساختار داده‌ها و تفکیک عوامل مؤثر باشد؛ ویژگی‌هایی که روش‌های ماتریسی کلاسیک به‌تنهایی قادر به تأمین آن‌ها نیستند.

۵ تجزیه مقدار منفرد مرتبه‌بالا (HOSVD)

تجزیه مقدار منفرد مرتبه‌بالا[۶] که با نام تجزیه تاکر[۷] نیز شناخته می‌شود، نخستین بار توسط Tucker در سال ۱۹۶۳ معرفی شد. این تجزیه را می‌توان تعمیمی از تجزیه مقدار منفرد ماتریسی[۸] به تانسورهای مرتبه‌ی بالاتر دانست. هدف اصلی در این چارچوب آن است که یک تانسور چندبعدی به‌صورت ترکیبی از **تبدیلات متعامد** در هر مُد و یک **تانسور هسته** نمایش داده شود؛ به‌گونه‌ای که ساختار چندخطی داده‌ها حفظ گردد.

ایده‌ی اصلی: اگر در SVD ماتریسی، یک ماتریس به کمک دو پایه‌ی متعامد (برای سطر و ستون) و یک ماتریس مقادیر منفرد تجزیه می‌شود، در HOSVD نیز یک تانسور با چندین مُد (Mode) به کمک پایه‌های متعامد در **تمام مُدها** تجزیه می‌گردد. به همین دلیل، HOSVD را می‌توان تعمیم PCA/SVD به فضای چندخطی دانست؛ زیرا به‌جای یک فضای خطی واحد، زیرفضاهای متعامد متعددی متناظر با مُدهای مختلف داده تعریف می‌شود.

فرم استاندارد HOSVD فرض کنید

$$\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$$

یک تانسور مرتبه‌ی N باشد. آنگاه می‌توان $\mathcal{A}$ را به‌صورت زیر نوشت:

$$\mathcal{A} = \mathcal{S} \times_1 U^{(1)} \times_2 U^{(2)} \times_3 \cdots \times_N U^{(N)},$$

که در آن:

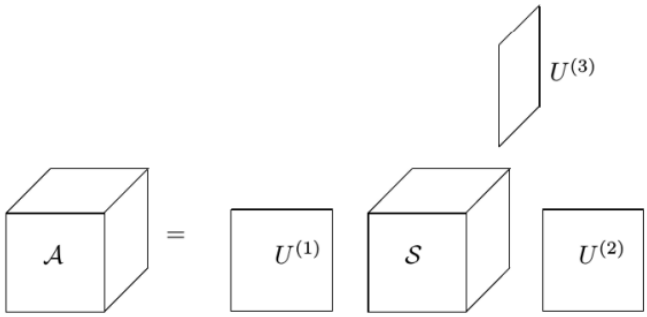
- $U^{(n)} \in \mathbb{R}^{I_n \times I_n}$ ماتریسی متعامد است، به‌طوری که

$$U^{(n)T} U^{(n)} = I.$$

  ستون‌های $U^{(n)}$ به‌عنوان **بردارهای منفرد مُد** n**ام** شناخته می‌شوند؛

- $\mathcal{S} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ تانسور هسته[۹] نام دارد و نقش مشابه ماتریس مقادیر منفرد را در SVD بازی می‌کند (البته در قالب یک ساختار چندبعدی).

در این رابطه، عملگر $\times_n$ **ضرب مُدی**[۱۰] است و به معنی ضرب تانسور در یک ماتریس روی بُعد (مُد) nام می‌باشد؛ به بیان دیگر، این عملگر امکان اعمال تبدیل خطی جداگانه روی هر مُد را فراهم می‌کند.



شکل ۵: نمایش هندسی تجزیه مقدار منفرد مرتبه‌بالا (HOSVD)

[6] Higher-Order Singular Value Decomposition (HOSVD).
[7] Tucker Decomposition.
[8] Singular Value Decomposition (SVD).
[9] Core Tensor
[10] Mode-n product

۱.۵ محاسبه HOSVD بر پایه بازکردن تانسور

محاسبه‌ی HOSVD به‌طور مستقیم بر پایه‌ی ارتباط این تجزیه با SVD ماتریسی انجام می‌شود. ایده‌ی اصلی آن است که یک تانسور چندبعدی با **بازکردن در هر مُد** به مجموعه‌ای از ماتریس‌ها تبدیل گردد و سپس SVD به‌صورت جداگانه روی هر یک از این ماتریس‌ها اعمال شود. به این ترتیب، پایه‌های متعامد متناظر با هر مُد استخراج شده و در نهایت تانسور هسته از طریق فرافکنی محاسبه می‌شود.

فرض کنید

$$\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$$

یک تانسور مرتبه‌ی N باشد. بازکردن تانسور در مُد nام، ماتریسی به‌صورت

$$A_{(n)} \in \mathbb{R}^{I_n \times (I_1 \cdots I_{n-1} I_{n+1} \cdots I_N)}$$

تولید می‌کند که در آن، تمام مؤلفه‌های تانسور به شکلی مرتب شده‌اند تا مُد nام سطرها را تشکیل دهد و سایر مُدها در ستون‌ها تجمیع شوند.

برای هر مُد $n = 1, \ldots, N$، تجزیه SVD ماتریس بازشده به‌صورت زیر محاسبه می‌شود:

$$A_{(n)} = U^{(n)} \Sigma^{(n)} V^{(n)T}.$$

پس از محاسبه‌ی تمامی ماتریس‌های متعامد $U^{(n)}$، تانسور هسته از طریق فرافکنی تانسور اولیه روی این زیرفضاها به‌دست می‌آید:

$$\mathcal{S} = \mathcal{A} \times_1 U^{(1)T} \times_2 U^{(2)T} \times_3 \cdots \times_N U^{(N)T}.$$

الگوریتم HOSVD

ورودی: تانسور $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$

خروجی: ماتریس‌های متعامد $U^{(1)}, U^{(2)}, \ldots, U^{(N)}$ و تانسور هسته $\mathcal{S}$

مراحل:

۱. برای هر مُد $n = 1, \ldots, N$:

(آ) بازکردن تانسور در مُد n و تشکیل ماتریس $A_{(n)}$؛

(ب) محاسبه‌ی SVD:

$$A_{(n)} = U^{(n)} \Sigma^{(n)} V^{(n)T};$$

(ج) ذخیره‌ی ماتریس $U^{(n)}$ (ستون‌های آن پایه‌ی مُد nام را می‌سازند).

۲. محاسبه‌ی تانسور هسته:

$$\mathcal{S} = \mathcal{A} \times_1 U^{(1)T} \times_2 \cdots \times_N U^{(N)T}.$$

۲.۵ خواص تانسور هسته

تانسور هسته $\mathcal{S}$ دو خاصیت مهم دارد که در تفسیر و کاهش بعد بسیار کاربردی هستند و نقش مشابه خواص بردارهای منفرد و مقدارهای منفرد در SVD را ایفا می‌کنند:

۱. **تمام‌متعامدی :(All-Orthogonality)** برش‌های تانسور هسته در هر مُد نسبت به یکدیگر متعامد هستند:

$$\langle \mathcal{S}_{i_n=\alpha}, \mathcal{S}_{i_n=\beta} \rangle = 0 \quad \forall \alpha \neq \beta.$$

این ویژگی مشابه عمود بودن بردارهای منفرد در SVD است و نشان می‌دهد مؤلفه‌های استخراج‌شده مستقل هستند.

۲. **مرتب‌سازی انرژی:** انرژی هر برش (با معیار نرم فروبنیوس) در هر مُد به‌صورت نزولی مرتب می‌شود:

$$\|\mathcal{S}_{i_n=1}\|_F \geq \|\mathcal{S}_{i_n=2}\|_F \geq \cdots \geq \|\mathcal{S}_{i_n=I_n}\|_F \geq 0.$$

بنابراین برش‌های اولیه مهم‌ترین مؤلفه‌های داده را حمل می‌کنند و معمولاً برای کاهش بُعد کافی هستند.

۳.۵ مقادیر منفرد در HOSVD

مقادیر منفرد تانسور در مُد nام به‌صورت نرم فروبنیوس برش‌های متناظر از تانسور هسته تعریف می‌شوند:

$$\sigma_i^{(n)} = \|\mathcal{S}_{i_n=i}\|_F, \quad i = 1, \ldots, I_n.$$

این مقادیر معمولاً به ترتیب نزولی مرتب می‌شوند:

$$\sigma_1^{(n)} \geq \sigma_2^{(n)} \geq \cdots \geq \sigma_{I_n}^{(n)} \geq 0.$$

در واقع $\sigma_i^{(n)}$ بیان می‌کند که مؤلفه‌ی iام در مُد nام چه میزان انرژی/اطلاعات را حمل می‌کند و به همین دلیل می‌توان از آن برای انتخاب رتبه‌های کاهش‌یافته استفاده کرد.

۴.۵ HOSVD برش‌خورده

[11]

در بسیاری از کاربردها تنها چند مؤلفه‌ی اول در هر مُد دارای انرژی قابل توجه هستند. در این حالت، با انتخاب

$$r_n < I_n, \quad n = 1, \ldots, N,$$

می‌توان فقط r_n بردار منفرد اول را نگه داشت و تجزیه‌ی کاهش‌یافته‌ی HOSVD را به‌صورت زیر نوشت:

$$\mathcal{A} \approx \tilde{\mathcal{S}} \times_1 \tilde{U}^{(1)} \times_2 \tilde{U}^{(2)} \times_3 \cdots \times_N \tilde{U}^{(N)}.$$

این تقریب مشابه SVD برش‌خورده است: با حذف مؤلفه‌های کم‌اهمیت، بعد و هزینه‌ی محاسباتی کاهش می‌یابد، در حالی‌که بخش عمده‌ای از اطلاعات داده حفظ می‌شود.

۵.۵ تفسیر هندسی HOSVD

HOSVD تعمیم طبیعی SVD به فضای تانسوری است. همان‌طور که SVD فضاهای سطر و ستون را متعامد می‌کند، HOSVD نیز زیرفضاهای متناظر با تمام مُدهای تانسور را به‌طور هم‌زمان متعامد می‌سازد. تانسور هسته نقش مشابه ماتریس Σ در SVD را دارد، با این تفاوت که به‌جای یک ماتریس قطری، یک تانسور چندبعدی است و **شدت برهم‌کنش مؤلفه‌ها در مُدهای مختلف** را نشان می‌دهد.

[11] Truncated HOSVD

۶.۵ کاربردهای HOSVD

تجزیه مقدار منفرد مرتبه‌بالا، به‌دلیل توانایی در استخراج ساختار چندخطی داده‌ها، در حوزه‌های مختلفی به‌کار گرفته می‌شود. مهم‌ترین کاربردهای آن عبارت‌اند از:

۱. **کاهش بُعد چندخطی** [۱۲]**:** در بسیاری از داده‌های چندبعدی (مانند داده‌های تصویری، ویدئویی، یا سنسوری)، کاهش بُعد در هر مُد به‌صورت جداگانه باعث حفظ بهتر ساختار داده نسبت به روش‌های بردارسازی و PCA می‌شود.

۲. **فشرده‌سازی داده‌های تانسوری:** با نگه‌داشتن مؤلفه‌های غالب در هر مُد (HOSVD برش‌خورده)، می‌توان نمایش فشرده‌ای از داده‌ها ساخت و حجم ذخیره‌سازی را به میزان قابل توجهی کاهش داد.

۳. **پردازش تصویر و بینایی ماشین:** از HOSVD در مسائل بازسازی تصویر، حذف نویز، تکمیل داده‌های گمشده و همچنین تحلیل ویژگی‌های چندعاملی استفاده می‌شود.

۴. **تشخیص چهره و مدل‌سازی چندعاملی:** در روش‌هایی مانند Tensorfaces، تصاویر چهره در قالب یک تانسور سازمان‌دهی می‌شوند تا عوامل مختلف (مانند هویت و حالت چهره) به صورت جداگانه مدل شوند و تشخیص نسبت به تغییرات محیطی مقاوم‌تر گردد.

۵. **یادگیری ماشین و تحلیل داده‌های چندراهه:** در تحلیل داده‌های چندراهه [۱۳] مانند داده‌های پزشکی، زیستی، شبکه‌های اجتماعی، و سیستم‌های توصیه‌گر، تجزیه‌های تانسوری از جمله Tucker/HOSVD به‌طور گسترده استفاده می‌شوند.

۶ الگوریتم Tensorfaces

الگوریتم Tensorfaces یک روش چندخطی برای تحلیل و تشخیص چهره است که بر پایه‌ی تجزیه مقدار منفرد مرتبه‌بالا (HOSVD) (یا N-mode SVD) بنا شده است. این الگوریتم نخستین بار توسط Vasilescu و Terzopoulos معرفی شد و ایده‌ی اصلی این روش آن است که داده‌های چهره به‌صورت **تانسوری** سازمان‌دهی شوند تا عوامل مؤثر (مانند هویت و حالت چهره) به شکل تفکیک‌شده و مستقل مدل شوند. این موضوع باعث می‌شود روش نسبت به تغییرات مزاحم مقاوم‌تر از روش‌های خطی کلاسیک مانند PCA/Eigenfaces باشد.

۱.۶ مدل‌سازی تانسوری داده‌های چهره

فرض کنید مجموعه‌ای از تصاویر چهره شامل n_p فرد باشد و از هر فرد در n_e حالت مختلف (برای مثال: خنثی، خندان، اخمو، با عینک و ...) تصویر ثبت شده باشد. هر تصویر در ابتدا یک ماتریس دوبعدی از شدت پیکسل‌هاست که با بردارسازی (چیدن ستون‌ها در امتداد هم) به یک بردار در فضای $\mathbb{R}^{n_i}$ تبدیل می‌شود؛ بنابراین n_i تعداد ویژگی‌های تصویر (تعداد پیکسل‌ها پس از بردارسازی) است.
در نتیجه، می‌توان کل داده‌ها را به صورت یک تانسور مرتبه‌سوم مدل کرد:

$$\mathcal{A} \in \mathbb{R}^{n_i \times n_e \times n_p}.$$

در این نمایش، هر مُد از تانسور متناظر با یکی از عوامل مؤثر در داده‌هاست: مُد اول مربوط به ویژگی‌های تصویری، مُد دوم بیانگر تغییرات ناشی از حالت (یا شرایط ثبت تصویر مانند نور) و مُد سوم بیانگر هویت افراد است. به‌طور مشخص:

- $\mathcal{A}(:,:,p)$ شامل تمام تصاویر فرد pام در حالت‌های مختلف است؛
- $\mathcal{A}(:,e,:)$ شامل تصاویر تمام افراد در حالت eام است.

[12] Multilinear Dimensionality Reduction
[13] Multi-way Data Analysis

این سازمان‌دهی تانسوری باعث می‌شود به جای ترکیب شدن تمام تغییرات در یک ماتریس بزرگ، عامل «حالت» و عامل «هویت» به صورت دو مُد مستقل مدل شوند و در مراحل بعدی امکان تفکیک و تحلیل این عوامل به صورت جداگانه فراهم گردد.

۲.۶ تجزیه HOSVD تانسور داده

از آنجا که معمولاً $n_e n_p \ll n_i$ است، تجزیه HOSVD تانسور $\mathcal{A}$ را به صورت زیر در نظر می‌گیریم. فرض کنیم تانسور داده $\mathcal{A}$ مطابق HOSVD به صورت زیر تجزیه شود:

$$\mathcal{A} = \mathcal{S} \times_i F \times_e G \times_p H,$$

که در آن:

- $\mathcal{S} \in \mathbb{R}^{(n_e n_p) \times n_e \times n_p}$ تانسور هسته است؛
- $F \in \mathbb{R}^{n_i \times (n_e n_p)}$ ماتریسی با ستون‌های **متعامد-یکه**[۱۴] است (یعنی $F^T F = I$)؛
- $G \in \mathbb{R}^{n_e \times n_e}$ و $H \in \mathbb{R}^{n_p \times n_p}$ ماتریس‌های متعامد[۱۵] هستند (یعنی $G^T G = I$ و $H^T H = I$)؛
- عملگرهای $\times_i, \times_e, \times_p$ ضرب مُدی بر روی مُد تصویر، حالت و شخص هستند.

دقت کنید که در اینجا $\times_i$ همان $\times_1$، $\times_e$ همان $\times_2$ و $\times_p$ همان $\times_3$ می‌باشد.

۳.۶ بازنویسی تجزیه و تعریف تانسورهای کمکی

برای تحلیل و تشخیص، مناسب است تجزیه HOSVD را به صورت‌هایی بازنویسی کنیم که نقش هر مُد در ساخت داده‌ها شفاف‌تر شود. مطابق تجزیه

$$\mathcal{A} = \mathcal{S} \times_i F \times_e G \times_p H,$$

ابتدا تانسور زیر را تعریف می‌کنیم:

$$\mathcal{D} = \mathcal{S} \times_i F.$$

در نتیجه می‌توان نوشت:

$$\mathcal{A} = \mathcal{D} \times_e G \times_p H.$$

در این بازنویسی، ابتدا اثر مُد تصویر (که بعد بسیار بزرگ دارد) وارد مدل می‌شود و سپس تغییرات مربوط به حالت و هویت به وسیله دو ضرب مُدی جداگانه اعمال می‌گردد.
برای بخش تشخیص، نمایش زیر نیز بسیار کلیدی است. اگر تعریف کنیم:

$$\mathcal{C} = \mathcal{S} \times_i F \times_e G,$$

آنگاه خواهیم داشت:

$$\mathcal{A} = \mathcal{C} \times_p H.$$

بنابراین $\mathcal{C}$ را می‌توان یک نمایش بینابینی دانست که اثر تصویر و حالت را در خود دارد و سپس با ضرب در H به داده‌های واقعی بازگشت می‌کنیم.

[14] Orthonormal columns
[15] Orthogonal

۴.۶ تفسیر برش‌ها و نمایش خطی تصویر

برای هر حالت e، برش‌های زیر را تعریف می‌کنیم:

$$A_e = \mathcal{A}(:, e, :) \in \mathbb{R}^{n_i \times n_p}, \qquad C_e = \mathcal{C}(:, e, :) \in \mathbb{R}^{n_i \times n_p}.$$

آنگاه از رابطه $\mathcal{A} = \mathcal{C} \times_p H$ نتیجه می‌شود:

$$A_e = C_e H^T, \qquad e = 1, \ldots, n_e.$$

اکنون ستون‌های ماتریس A_e را به‌صورت

$$A_e = \left[a_1^{(e)}\ a_2^{(e)}\ \cdots\ a_{n_p}^{(e)}\right],$$

و همچنین ستون‌های ماتریس H^T را به صورت

$$H^T = \left[h_1\ h_2\ \cdots\ h_{n_p}\right]$$

می‌نویسیم. در این صورت ستون p-ام رابطه $A_e = C_e H^T$ برابر است با:

$$a_p^{(e)} = C_e h_p.$$

از طرفی اگر ستون‌های C_e را به صورت

$$C_e = \left[c_1\ c_2\ \cdots\ c_{n_p}\right] \in \mathbb{R}^{n_i \times n_p},$$

و بردار h_p را به صورت

$$h_p = \begin{bmatrix} h'_1 \\ h'_2 \\ \vdots \\ h'_{n_p} \end{bmatrix} \in \mathbb{R}^{n_p},$$

بنویسیم، آنگاه خواهیم داشت:

$$a_p^{(e)} = \left[c_1\ c_2\ \cdots\ c_{n_p}\right] \begin{bmatrix} h'_1 \\ h'_2 \\ \vdots \\ h'_{n_p} \end{bmatrix} = h'_1 c_1 + h'_2 c_2 + \cdots + h'_{n_p} c_{n_p}.$$

بنابراین تصویر فرد p-ام در حالت e (یعنی $a_p^{(e)}$) را می‌توان به صورت یک ترکیب خطی از ستون‌های C_e نوشت؛ در نتیجه ستون‌های C_e نقش یک پایه (یا دیکشنری) برای حالت e-ام را ایفا می‌کنند و ضرایب این ترکیب خطی دقیقاً مؤلفه‌های بردار h_p هستند. از آنجا که بردار h_p به ستون p-ام از H^T (معادل سطر p-ام ماتریس H) مربوط است، می‌توان h_p را به عنوان نمایش (کُد) هویت فرد p-ام در نظر گرفت؛ نکته مهم این است که با تغییر حالت e، این بردار ثابت می‌ماند و تنها پایه‌ی C_e تغییر می‌کند.

۵.۶ فاز آموزش

در فاز آموزش هدف آن است که نمایش تانسوری داده‌ها ساخته شده و کمیت‌های لازم برای فاز آزمون استخراج و ذخیره شوند. به طور مشخص، در پایان این مرحله باید زیرفضای تصویری (ماتریس F) برای کاهش بعد، پارامترهای مربوط به حالت (G)، پارامترهای هویت (H) یا بردارهای h_p، و همچنین کمیت‌هایی که حل سریع مسائل کمترین مربعات را ممکن می‌کنند (مانند تجزیه‌های QR برای هر حالت) در اختیار باشند.
مراحل آموزش به صورت زیر است:

۱. **پیش‌پردازش داده‌ها:** تصاویر ابتدا هم‌ترازسازی و نرمال‌سازی شده و سپس با بردارسازی به فضای $\mathbb{R}^{n_i}$ منتقل می‌شوند.

۲. **ساخت تانسور داده:** پس از پیش‌پردازش، تصاویر در قالب تانسور

$$\mathcal{A} \in \mathbb{R}^{n_i \times n_e \times n_p}$$

سازمان‌دهی می‌شوند.

۳. **محاسبه HOSVD:** تجزیه

$$\mathcal{A} = \mathcal{S} \times_i F \times_e G \times_p H$$

محاسبه می‌شود و کمیت‌های $\mathcal{S}, F, G, H$ استخراج می‌گردند.

۴. **ذخیره بردارهای هویت:** بردارهای

$$H^T = [h_1 \; h_2 \; \cdots \; h_{n_p}]$$

به عنوان نمایش هویت افراد ذخیره می‌شوند.

۶.۶ مسئله تشخیص چهره و کاهش هزینه محاسباتی

فرض کنید تصویر ورودی ناشناخته به صورت بردار

$$z \in \mathbb{R}^{n_i}$$

داده شده است. اگر z متعلق به یکی از افراد پایگاه داده باشد، در حالت ایده‌آل باید بتوان آن را برای یکی از حالت‌ها به صورت یک ترکیب خطی از ستون‌های C_e نوشت:

$$z \approx C_e \alpha_e.$$

در صورتی که z دقیقاً تصویر فرد pام در حالت e باشد، آنگاه $\alpha_e = h_p$ خواهد بود. بنابراین نسخه اولیه تشخیص به شکل زیر است: برای هر $e = 1, \ldots, n_e$ مسئله کمترین مربعات

$$\min_{\alpha_e} \|C_e \alpha_e - z\|_2^2$$

حل می‌شود و سپس α_e با بردارهای هویت $\{h_p\}_{p=1}^{n_p}$ مقایسه شده و نزدیک‌ترین فرد تعیین می‌گردد.
با این حال، انجام این کار برای تمام حالت‌ها پرهزینه است، زیرا ماتریس C_e در فضای n_i تعریف شده و n_i معمولاً بسیار بزرگ است (تعداد پیکسل‌ها). برای کاهش هزینه محاسباتی، از تعریف

$$\mathcal{C} = \mathcal{S} \times_i F \times_e G$$

نتیجه می‌شود که برای هر حالت e می‌توان نوشت:

$$C_e = F B_e, \qquad B_e \in \mathbb{R}^{(n_e n_p) \times n_p},$$

که B_e یک ماتریس کم‌بعد است و از برش تانسور $(\mathcal{S} \times_e G)$ به دست می‌آید. این فاکتورگیری مهم است زیرا نگاشت F داده‌ها را از فضای بسیار بزرگ n_i به فضای کوچک‌تر $(n_e n_p)$ منتقل می‌کند و در نتیجه محاسبات اصلی در فضای کم‌بعد انجام می‌شوند.

برای تبدیل مسئله کمترین مربعات به فرم کم‌بعد، F را به یک ماتریس مربعی متعامد گسترش می‌دهیم:

$$\hat{F} = [F \;\; F_\perp], \qquad \hat{F}^T\hat{F} = I.$$

در این صورت داریم:

$$\|C_e\alpha_e - z\|_2^2 = \|FB_e\alpha_e - z\|_2^2 = \|\hat{F}^T(FB_e\alpha_e - z)\|_2^2.$$

اکنون عبارت داخل نُرم را باز می‌کنیم:

$$\hat{F}^T(FB_e\alpha_e - z) = \begin{bmatrix} F^T \\ (F_\perp)^T \end{bmatrix}(FB_e\alpha_e - z) = \begin{bmatrix} F^TFB_e\alpha_e - F^Tz \\ (F_\perp)^TFB_e\alpha_e - (F_\perp)^Tz \end{bmatrix}.$$

از آنجا که F ستون‌های متعامد-یکه دارد، $F^TF = I$، و همچنین چون $F_\perp$ عمود بر F است، داریم $(F_\perp)^TF = 0$. بنابراین:

$$\hat{F}^T(FB_e\alpha_e - z) = \begin{bmatrix} B_e\alpha_e - F^Tz \\ -(F_\perp)^Tz \end{bmatrix}.$$

پس:

$$\|C_e\alpha_e - z\|_2^2 = \|B_e\alpha_e - F^Tz\|_2^2 + \|(F_\perp)^Tz\|_2^2.$$

از آنجا که جمله دوم مستقل از e و α_e است، کمینه‌سازی عبارت اولیه معادل است با حل مسئله کم‌بعد زیر:

$$\min_{\alpha_e} \|B_e\alpha_e - \hat{z}\|_2^2, \qquad \hat{z} = F^Tz.$$

در نتیجه به جای کار در فضای n_i، مسئله در فضای (n_en_p) حل می‌شود که بسیار کوچک‌تر است و همین نکته سرعت الگوریتم را به شدت افزایش می‌دهد.

۷.۶ فاز آزمون

در فاز آزمون، برای تصویر ورودی z مراحل زیر انجام می‌شود:

۱. **فرافکنی تصویر ورودی:**

$$\hat{z} = F^Tz \in \mathbb{R}^{n_en_p}.$$

۲. **حل سریع کمترین مربعات برای هر حالت با QR:** برای هر حالت e، ماتریس B_e را یک بار تجزیه QR می‌کنیم:

$$B_e = Q_eR_e.$$

سپس مسئله کمترین مربعات

$$\alpha_e = \arg\min_{\alpha} \|B_e\alpha - \hat{z}\|_2^2$$

با استفاده از QR به حل دستگاه زیر کاهش می‌یابد:

$$R_e\alpha_e = Q_e^T\hat{z}.$$

۳. **تشخیص هویت:** پس از محاسبه α_e، بردار حاصل با بردارهای هویت ذخیره‌شده $\{h_p\}_{p=1}^{n_p}$ مقایسه می‌شود؛ به طور مثال اگر

$$\|\alpha_e - h_p\|_2 < \mathrm{tol},$$

آنگاه تصویر متعلق به فرد pام در نظر گرفته می‌شود.

الگوریتم نهایی تشخیص چهره با Tensorfaces

ورودی: تصویر تست $z \in \mathbb{R}^{n_i}$
خروجی: هویت تصویر (یا اعلام عدم تعلق به پایگاه داده)

مراحل:

۱. محاسبه فرافکنی: $\hat{z} = F^T z$.

۲. برای هر حالت e:

(آ) تجزیه QR: $B_e = Q_e R_e$.

(ب) حل: $R_e \alpha_e = Q_e^T \hat{z}$.

۳. یافتن فرد:

$$p^* = \arg\min_p \|\alpha_e - h_p\|_2.$$

۴. اگر $\|\alpha_e - h_{p^*}\|_2 < \mathrm{tol}$ آنگاه فرد p^* گزارش می‌شود، وگرنه تصویر به عنوان «خارج از پایگاه داده» در نظر گرفته می‌شود.

تفکیک صریح عوامل (هویت و حالت) در فضای تانسوری باعث می‌شود Tensorfaces نسبت به تغییرات حالت چهره مقاوم‌تر از روش‌های خطی کلاسیک باشد، در حالی‌که همچنان امکان کاهش بعد و محاسبات سریع فراهم است.

۸.۶ مقایسه مفهومی Eigenfaces و Tensorfaces

روش Eigenfaces مبتنی بر تحلیل مؤلفه‌های اصلی (PCA) است و تصاویر چهره را به‌صورت بردارهای یک‌بعدی در یک ماتریس نمایش می‌دهد. در این چارچوب، تغییرات نور، حالت و زاویه دید عملاً در یک فضای واحد و به‌صورت درهم‌تنیده مدل می‌شوند.
در مقابل، Tensorfaces یک روش چندخطی است که تغییرات مختلف را در مُدهای مستقل نمایش می‌دهد و به همین دلیل توانایی جداسازی عوامل مزاحم را دارد. تفاوت‌های مفهومی این دو روش به‌صورت زیر خلاصه می‌شود:

- Eigenfaces تغییرات نور، حالت و زاویه دید را به‌صورت درهم‌تنیده مدل می‌کند، در حالی که Tensorfaces این عوامل را از یکدیگر جدا می‌سازد.
- Tensorfaces نسبت به شرایط محیطی مختلف مقاوم‌تر است.
- Eigenfaces ساده‌تر و کم‌هزینه‌تر از نظر محاسباتی است.
- Tensorfaces برای داده‌های چندعاملی و واقعی مناسب‌تر است.

جدول ۱: مقایسه مفهومی روش‌های Eigenfaces و Tensorfaces

ویژگی	Eigenfaces	Tensorfaces
مدل ریاضی	تحلیل مؤلفه‌های اصلی (PCA)	تحلیل چندخطی مبتنی بر HOSVD
ساختار داده	ماتریس دوبعدی	تانسور چندبعدی
تعداد عوامل قابل مدل‌سازی	یک عامل غالب	چند عامل مستقل
نوع تجزیه	SVD ماتریسی	SVD N-mode (HOSVD)
تفکیک عوامل مزاحم	فاقد تفکیک	دارای تفکیک صریح مُدها
مقاومت در برابر تغییرات	پایین	بالا
پیچیدگی محاسباتی	کم	بیشتر از Eigenfaces
کاربرد در داده‌های واقعی	محدود	مناسب برای داده‌های چندعاملی
رابطه با PCA	روش پایه	تعمیم PCA به فضای تانسوری

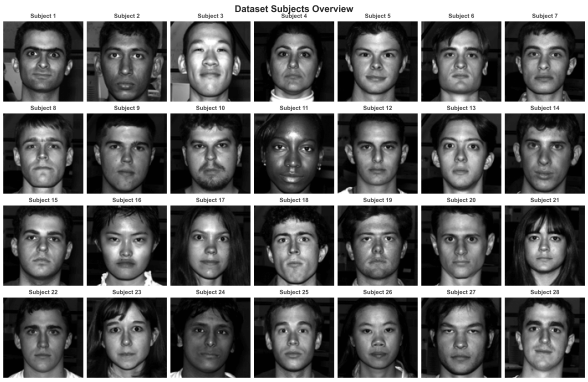
۷ پیاده‌سازی الگوریتم‌ها

۱.۷ معرفی دیتاست

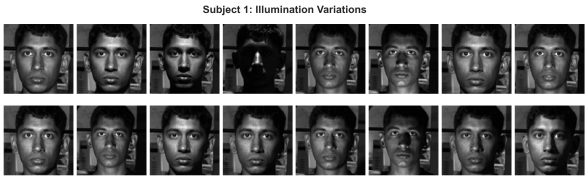
در این پروژه از دیتاست Extended Yale B (Cropped) استفاده شده است که یکی از مجموعه‌داده‌های مرجع در حوزه‌ی تشخیص چهره محسوب می‌شود. این دیتاست شامل تصاویر چهره‌ی **۲۸ فرد مختلف** بوده که برای هر فرد، **۹ وضعیت متفاوت زاویه‌ی قرارگیری (Pose)** و **۶۴ شرایط نوری مختلف** در نظر گرفته شده است. در نتیجه، مجموعاً **۱۶،۱۲۸ تصویر** در این مجموعه‌داده وجود دارد.
تصاویر با وضوح اولیه‌ی 192×168 پیکسل و به‌صورت سیاه‌وسفید ذخیره شده‌اند. تصاویر هر فرد در این دیتاست با تغییرات قابل توجه در زاویه و شدت نور ثبت شده‌اند، به‌گونه‌ای که در برخی تصاویر سایه‌های شدید و نواحی بسیار تاریک یا بسیار روشن مشاهده می‌شود. این ویژگی، دیتاست را به گزینه‌ای مناسب برای بررسی پایداری الگوریتم‌های تشخیص چهره در برابر تغییرات نوری تبدیل کرده است.
پس از بارگذاری داده‌ها، به‌منظور یکسان‌سازی ورودی‌ها و کاهش پیچیدگی محاسباتی، تمامی تصاویر به اندازه‌ی ثابت 96×96 پیکسل تغییر اندازه داده شده‌اند. این مرحله‌ی پیش‌پردازش، پیاده‌سازی الگوریتم‌های Eigenfaces و Tensorfaces را تسهیل می‌نماید.

۲.۷ نمایش تصاویر نمونه

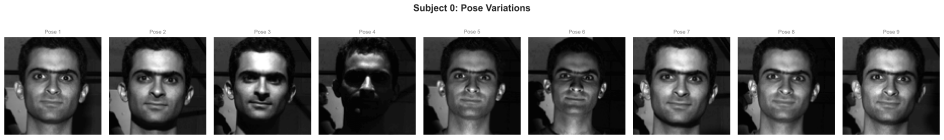
به‌منظور بررسی کیفی ساختار داده‌ها، چند نمونه تصویر از دیتاست نمایش داده می‌شود. این تصاویر به‌خوبی تنوع شدید شرایط نورپردازی و تفاوت میان افراد مختلف را نشان می‌دهند.
در شکل ۶، نمونه‌ای از تصاویر موجود در دیتاست در شرایط مختلف نمایش داده شده است.



(آ) افراد مختلف تحت شرایط نوری مشابه



(ب) یک فرد ثابت تحت شرایط نوری متفاوت



(ج) یک فرد ثابت تحت شرایط زاویه قرارگیری متفاوت

شکل ۶: نمونه‌هایی از تصاویر دیتاست

۳.۷ پیش‌پردازش تصاویر

با توجه به این‌که تصاویر مجموعه‌داده‌ی دانشگاه Yale همگی در مقیاس خاکستری ثبت شده‌اند و چهره‌ها در مرکز تصویر قرار دارند، مراحل پایه‌ای نظیر برش چهره و تبدیل به خاکستری از پیش در خود دیتاست انجام شده است. در این پروژه، پیش‌پردازش داده‌ها عمدتاً شامل دو بخش بوده است:

- **تغییر اندازه:** تمامی تصاویر به ابعاد 96×96 تبدیل شدند.
- **نرمال‌سازی شدت روشنایی:** هر تصویر ابتدا استانداردسازی شد (کم‌کردن میانگین و تقسیم بر انحراف معیار) و سپس به بازه‌ی $[0, 255]$ نگاشت شد تا مقیاس شدت روشنایی در تصاویر قابل مقایسه باشد.

در ادامه برای تحلیل کیفی داده‌ها، تصویر میانگین مجموعه‌ی آموزشی محاسبه شده و از تصویر نمونه کسر گردید. این مرحله باعث کاهش اثر روشنایی کلی و برجسته شدن الگوهای ساختاری چهره می‌شود. شکل ۷ تصویر اصلی، تصویر میانگین و تصویر مرکزسازی‌شده را نمایش می‌دهد.

شکل ۷: مراحل پیش‌پردازش تصاویر: (چپ) تصویر اصلی، (وسط) تصویر میانگین مجموعه‌ی آموزشی، (راست) تصویر پس از کم‌کردن میانگین.

۴.۷ تقسیم‌بندی داده‌ها و تنظیمات اجرای آزمایش

به‌منظور ارزیابی منصفانه‌ی عملکرد الگوریتم‌های مختلف، داده‌های مورد استفاده در این پروژه به دو مجموعه‌ی آموزشی و آزمون تقسیم شدند. تقسیم داده‌ها به‌صورت تصادفی و stratified انجام شده است تا توزیع نمونه‌های هر فرد در هر دو مجموعه تقریباً یکنواخت باقی بماند.
با وجود آن‌که دیتاست Extended Yale B در حالت کامل شامل 16128 تصویر است، در اجرای حاضر به‌منظور کاهش هزینه‌ی محاسباتی و امکان اجرای آزمایش‌های متعدد برای تحلیل هایپرپارامترها، برای هر فرد دقیقاً 64 تصویر انتخاب شده است. از آن‌جا که برای هر فرد در این دیتاست ۶۴ تصویر تحت شرایط نوری مختلف وجود دارد، این انتخاب باعث می‌شود تنوع نوری کامل هر فرد حفظ شود و در عین حال تعداد نمونه‌ها در تمامی کلاس‌ها برابر باقی بماند.
بنابراین با در نظر گرفتن 28 کلاس، مجموع داده‌های مورد استفاده شامل 1792 تصویر است:

$$28 \times 64 = 1792$$

سپس این داده‌ها با نسبت تقریبی 80% برای آموزش و 20% برای آزمون تقسیم شدند که خروجی این تقسیم‌بندی به صورت زیر است:

$$\text{Train} = 1433, \qquad \text{Test} = 359, \qquad \text{Classes} = 28$$

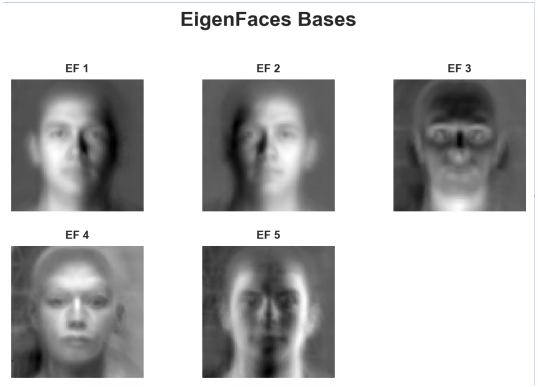
به‌طور متوسط، در این تقسیم‌بندی برای هر فرد حدود 51 تصویر آموزشی و 12 تصویر آزمون وجود دارد. این موضوع باعث می‌شود که هر کلاس دارای تعداد نمونه‌ی کافی برای آموزش باشد و ارزیابی الگوریتم‌ها نیز بر اساس داده‌های دیده‌نشده انجام گیرد.

۵.۷ پیاده‌سازی الگوریتم Eigenfaces

در این بخش، خروجی‌های حاصل از اجرای عملی الگوریتم Eigenfaces بر روی دیتاست Yale ارائه می‌شود. تمرکز این قسمت بر بررسی کیفی نتایج تصویری و نیز تحلیل اثر تعداد مؤلفه‌های اصلی k بر کیفیت بازسازی و دقت تشخیص است.

۱.۵.۷ نمایش و تحلیل Eigenfaces استخراج‌شده

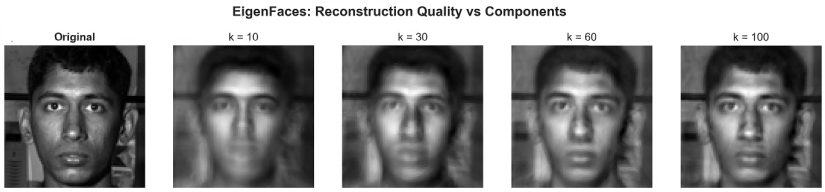
با اعمال الگوریتم Eigenfaces بر روی تصاویر آموزشی، بردارهای پایه‌ای استخراج می‌شوند که بیشترین واریانس موجود در داده‌ها را توصیف می‌کنند. این بردارها که با عنوان Eigenface شناخته می‌شوند، به‌صورت تصاویر دوبعدی قابل نمایش هستند و الگوهای غالب موجود در چهره‌ها را بازنمایی می‌کنند.
شکل ۸ نمونه‌ای از چند Eigenface استخراج‌شده را نشان می‌دهد. در این تصاویر، نواحی روشن و تیره بیانگر میزان تأثیر پیکسل‌های مختلف در تشکیل مؤلفه‌های اصلی هستند.



شکل ۸: نمونه‌ای از Eigenface های استخراج‌شده از تصاویر آموزشی

۲.۵.۷ بازسازی تصاویر چهره

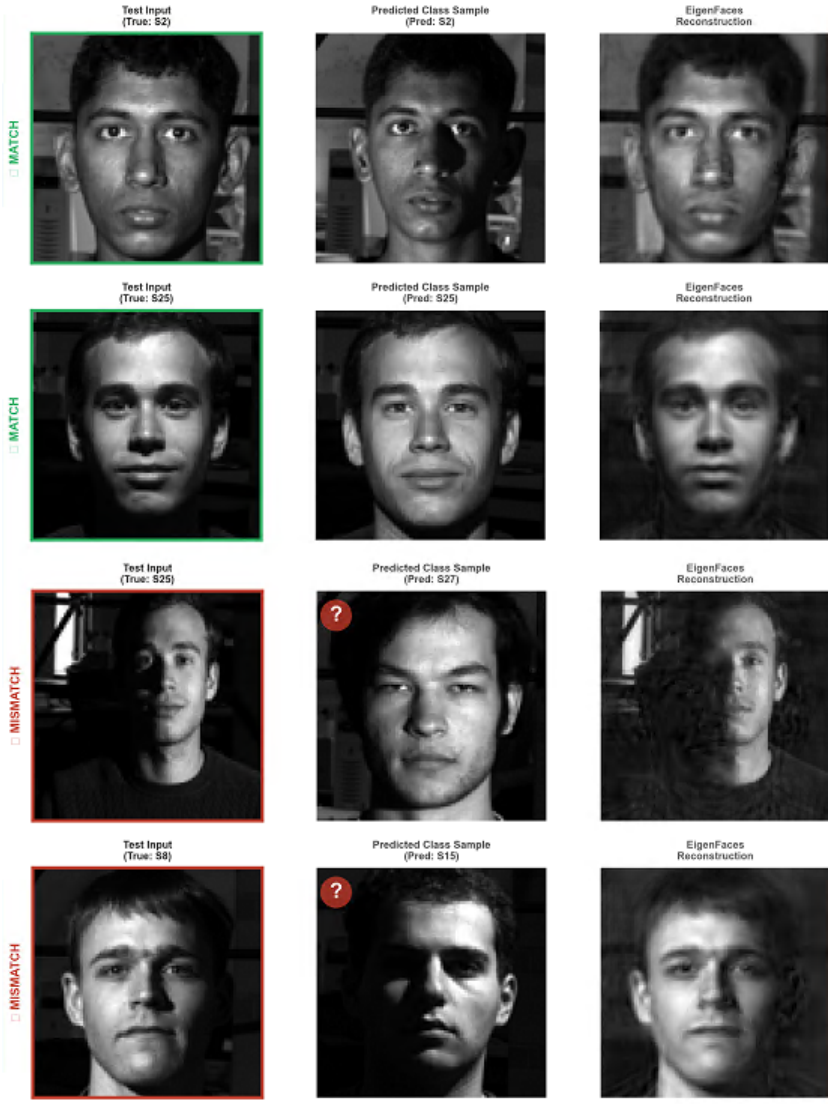
یکی از کاربردهای مهم الگوریتم Eigenfaces، بازسازی تقریبی تصاویر چهره با استفاده از تعداد محدودی از مؤلفه‌های اصلی است. در این فرآیند، تصویر آزمون ابتدا به فضای Eigenfaces فرافکنی شده و سپس تنها با استفاده از همین مؤلفه‌ها بازسازی می‌شود. شکل ۹ کیفیت بازسازی تصویر چهره را با استفاده از تعداد متفاوتی از مؤلفه‌های اصلی نشان می‌دهد. در این آزمایش، تصویر آزمون ابتدا به فضای Eigenfaces فرافکنی شده و سپس با استفاده از مقادیر مختلف k (تعداد Eigenface ها) بازسازی گردیده است. همان‌طور که مشاهده می‌شود، با افزایش تعداد مؤلفه‌های اصلی، جزئیات بیشتری از چهره در تصویر بازسازی‌شده حفظ می‌شود. در مقادیر کوچک k، ساختار کلی صورت قابل تشخیص است اما جزئیات ظریف مانند چشم‌ها و دهان با دقت کمتری نمایش داده می‌شوند. با افزایش k، کیفیت بازسازی بهبود یافته و تصویر بازسازی‌شده به‌تدریج به تصویر اصلی نزدیک می‌شود. این نتایج نشان می‌دهند که الگوریتم Eigenfaces قادر است با استفاده از تعداد محدودی مؤلفه‌ی اصلی، نمایشی فشرده و در عین حال معنادار از داده‌های چهره ایجاد کند.



شکل ۹: بازسازی تصویر چهره با استفاده از تعداد متفاوتی از Eigenface ها

۳.۵.۷ نمونه‌های پیش‌بینی درست و نادرست

به‌منظور بررسی کیفی رفتار الگوریتم Eigenfaces در مرحله‌ی تشخیص، نمونه‌هایی از تصاویر مجموعه‌ی آزمون که به‌درستی و به‌نادرستی طبقه‌بندی شده‌اند، به‌صورت بصری مورد تحلیل قرار گرفته‌اند. هدف از این بررسی، درک بهتر نحوه‌ی تصمیم‌گیری الگوریتم و شناسایی عوامل مؤثر در بروز خطاهای تشخیص است. در شکل ۱۰، برای هر نمونه دو تصویر به نمایش درآمده است. ستون سمت چپ تصویر ورودی آزمون را نشان می‌دهد، ستون میانی نمونه‌ای از تصاویر مجموعه‌ی آموزشی متعلق به کلاس پیش‌بینی‌شده توسط الگوریتم را نمایش می‌دهد، و ستون سمت راست تصویر بازسازی‌شده با استفاده از مؤلفه‌های اصلی Eigenfaces را ارائه می‌کند. در این شکل، نمونه‌های درست و نادرست به‌طور هم‌زمان نمایش داده شده‌اند و نمونه‌های نادرست با رنگ‌بندی متفاوت قابل تشخیص هستند. بررسی این نمونه‌ها نشان می‌دهد که در مواردی که تغییرات شدید نورپردازی وجود دارد یا شباهت ساختاری قابل توجهی میان چهره‌ی افراد مختلف دیده می‌شود، الگوریتم Eigenfaces ممکن است در تفکیک هویت‌ها دچار خطا شود. از سوی دیگر، در نمونه‌های پیش‌بینی درست، تصاویر بازسازی‌شده ساختار کلی چهره را به‌خوبی حفظ کرده‌اند که بیانگر استخراج مؤلفه‌های معنادار توسط مدل است.



شکل ۱۰: نمونه‌هایی از پیش‌بینی‌های درست و نادرست الگوریتم Eigenfaces (چپ) تصویر ورودی آزمون؛ (وسط) نمونه‌ای از کلاس پیش‌بینی‌شده؛ (راست) تصویر بازسازی‌شده با استفاده از Eigenfaces

۶.۷ پیاده‌سازی الگوریتم Tensorfaces مبتنی بر HOSVD

در این بخش، پیاده‌سازی عملی الگوریتم Tensorfaces مبتنی بر تجزیه‌ی مقدار منفرد مرتبه‌بالا [۱۶] بر روی دیتاست Yale ارائه می‌شود. برخلاف روش Eigenfaces که تصاویر را پس از بردارسازی در قالب یک ماتریس دوبعدی تحلیل می‌کند، الگوریتم Tensorfaces ساختار دوبعدی تصاویر را حفظ کرده و داده‌ها را به‌صورت یک تانسور چندبعدی مدل‌سازی می‌نماید. این رویکرد امکان استخراج هم‌زمان الگوهای مکانی و تغییرات میان نمونه‌ها را فراهم می‌سازد.
در پیاده‌سازی حاضر، به‌منظور مدل‌سازی صریح تغییرات نورپردازی، تصاویر آموزشی پس از نرمال‌سازی، در قالب یک تانسور مرتبه‌سوم به‌شکل

$$\mathcal{D} \in \mathbb{R}^{S \times L \times P}$$

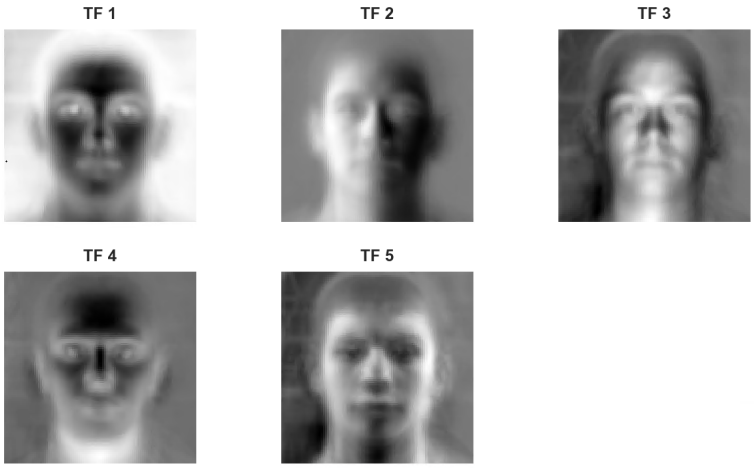
سازمان‌دهی شده‌اند، که در آن S تعداد افراد (Subjects)، L تعداد شرایط نوری (Lights) و $P = H \times W$ تعداد کل پیکسل‌های تصویر (پس از تغییر اندازه) می‌باشد. در این ساختار، مد اول بیانگر تغییرات مربوط به هویت افراد، مد دوم بیانگر تغییرات ناشی از شرایط نورپردازی، و مد سوم بیانگر اطلاعات پیکسلی (ساختار مکانی تصویر) می‌باشد.
پس از تشکیل این تانسور، تجزیه‌ی HOSVD بر روی آن اعمال شده و تانسور داده به‌صورت زیر تجزیه می‌شود:

$$\mathcal{D} = \mathcal{G} \times_1 U_S \times_2 U_L \times_3 U_P$$

که در آن U_S زیرفضای هویت (افراد)، U_L زیرفضای نور، و U_P زیرفضای پیکسلی را نمایش می‌دهد، و $\mathcal{G}$ تانسور هسته‌ی فشرده‌شده است که تعامل میان این مدها را در خود نگه می‌دارد.

۱.۶.۷ Tensorfaces استخراج‌شده

در شکل ۱۱ نمونه‌ای از پایه‌های پیکسلی استخراج‌شده از زیرفضای پیکسلی U_P نمایش داده شده است. این پایه‌ها که با عنوان Tensorfaces شناخته می‌شوند، چهره‌ی افراد واقعی نیستند، بلکه الگوهای آماری غالب در ساختار پیکسلی داده‌های چهره را بازنمایی می‌کنند.
نواحی روشن و تیره در این تصاویر بیانگر میزان مشارکت پیکسل‌ها در تشکیل زیرفضای کاهش‌یافته است. این پایه‌ها نقش مهمی در فرافکنی تصاویر به فضای ویژگی و بازسازی چهره‌ها ایفا می‌کنند.

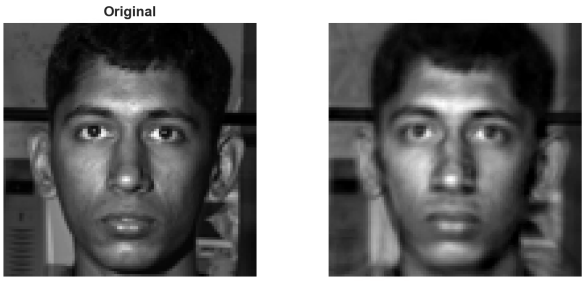


شکل ۱۱: نمونه‌ای از پایه‌های پیکسلی (Tensorfaces) استخراج‌شده با استفاده از تجزیه HOSVD.

[16] Higher-Order Singular Value Decomposition - HOSVD

۲.۶.۷ بازسازی تصاویر چهره

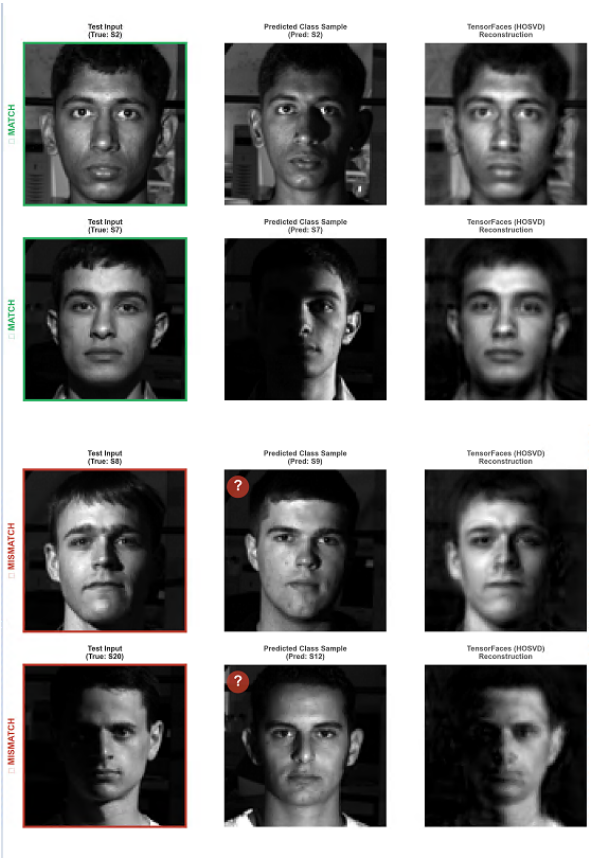
یکی از روش‌های ارزیابی کیفی عملکرد الگوریتم Tensorfaces، بازسازی تصاویر چهره با استفاده از مؤلفه‌های استخراج‌شده است. در این فرآیند، تصویر ورودی ابتدا به فضای کاهش‌یافته‌ی Tensorfaces فرافکنی شده و سپس با استفاده از زیرفضای پیکسلی U_P، در فضای پیکسلی بازسازی می‌شود.
شکل ۱۲ نمونه‌ای از تصویر اصلی و تصویر بازسازی‌شده توسط Tensorfaces را نشان می‌دهد. شباهت کلی ساختار صورت و حفظ الگوهای اصلی روشنایی در تصویر بازسازی‌شده نشان‌دهنده‌ی توانایی مدل در استخراج و حفظ اطلاعات مهم چهره است، هرچند برخی جزئیات ظریف به‌دلیل کاهش بعد از بین رفته‌اند.



شکل ۱۲: بازسازی نمونه تصویر چهره با استفاده از الگوریتم Tensorfaces (چپ) تصویر اصلی؛ (راست) تصویر بازسازی‌شده در فضای Tensorfaces

۳.۶.۷ نمونه‌های پیش‌بینی درست و نادرست

به‌منظور بررسی کیفی عملکرد الگوریتم Tensorfaces در مرحله‌ی تشخیص هویت، نمونه‌هایی از پیش‌بینی‌های درست و نادرست بر روی مجموعه‌ی آزمون به‌صورت بصری مورد تحلیل قرار گرفته‌اند. در شکل ۱۳، برای هر نمونه سه تصویر نمایش داده شده است: ستون سمت چپ تصویر ورودی آزمون، ستون میانی نمونه‌ای از تصاویر آموزشی متعلق به کلاس پیش‌بینی‌شده توسط الگوریتم، و ستون سمت راست تصویر بازسازی‌شده با استفاده از مدل Tensorfaces را نشان می‌دهد. بررسی این نمونه‌ها نشان می‌دهد که الگوریتم Tensorfaces در بسیاری از موارد، حتی در حضور تغییرات شدید نورپردازی، قادر به استخراج ویژگی‌های پایدار مرتبط با هویت افراد است. با این حال، در برخی نمونه‌ها، شباهت ساختاری میان چهره‌ها یا محدودیت رتبه‌های انتخاب‌شده در تجزیه HOSVD می‌تواند منجر به بروز خطا در تشخیص شود.



شکل ۱۳: نمونه‌هایی از پیش‌بینی‌های الگوریتم Tensorfaces

لازم به ذکر است که الگوریتم Tensorfaces در تقسیم‌بندی فعلی داده‌های آموزشی و آزمون، به دقت بالایی در تشخیص هویت دست یافته است، اما تمامی نمونه‌های آزمون به‌طور کامل و بدون خطا طبقه‌بندی نشده‌اند. این موضوع نشان می‌دهد که اگرچه مدل‌سازی چندخطی Tensorfaces پایداری بیشتری نسبت به Eigenfaces در برابر تغییرات شرایط تصویربرداری فراهم می‌کند، انتخاب مناسب رتبه‌های HOSVD و استفاده از طبقه‌بندهای پیشرفته‌تر می‌تواند نقش مهمی در بهبود بیشتر عملکرد نهایی مدل ایفا کند.

۷.۷ تنظیم و انتخاب هایپرپارامترها

عملکرد الگوریتم‌های Eigenfaces و Tensorfaces (مبتنی بر HOSVD) به‌طور قابل توجهی به انتخاب مناسب هایپرپارامترها وابسته است. این پارامترها مستقیماً بر میزان فشرده‌سازی داده‌ها، میزان حفظ اطلاعات تمایزدهنده‌ی چهره‌ها، پایداری روش در برابر تغییرات تصویربرداری (به‌ویژه نورپردازی)، و همچنین هزینه‌ی محاسباتی (زمان و حافظه) اثر می‌گذارند. ازاین‌رو، تنظیم صحیح آن‌ها نقش کلیدی در دستیابی به یک مدل کارآمد و قابل اتکا دارد.
در این پروژه، انتخاب هایپرپارامترها به‌صورت تجربی و بر اساس یک رویکرد چندمعیاره انجام شده است. به این معنا که به‌جای تمرکز صرف بر یک معیار (مانند دقت)، توازن میان معیارهای مختلف از جمله **دقت تشخیص**، **زمان اجرا**، **کیفیت بازسازی**، و **پایداری در برابر تغییرات نور** به‌طور هم‌زمان در نظر گرفته شده است. در ادامه، پارامترهای اصلی هر الگوریتم و منطق انتخاب آن‌ها شرح داده می‌شود.

۱.۷.۷ هایپرپارامترهای Eigenfaces

در الگوریتم Eigenfaces، مهم‌ترین هایپرپارامتر، تعداد مؤلفه‌های اصلی یا تعداد Eigenfaceهای مورد استفاده (k) است. این پارامتر تعیین می‌کند که داده‌ها در چه فضای کم‌بعدی نمایش داده شوند و چه میزان از ساختار آماری تصویر در این نمایش فشرده حفظ گردد. به بیان دیگر، k نقش مستقیمی در ظرفیت مدل برای بازنمایی جزئیات چهره‌ها دارد.
انتخاب مقدار مناسب برای k یک مسئله‌ی ایجاد توازن میان دو هدف متضاد است: از یک‌سو، کاهش بعد برای حذف نویز و کاهش پیچیدگی محاسباتی ضروری است؛ اما از سوی دیگر، کاهش بیش از حد می‌تواند باعث حذف ویژگی‌های هویتی و کاهش قدرت تشخیص شود. در صورت انتخاب مقادیر بسیار کوچک برای k، مدل تنها قادر به نمایش ساختار کلی چهره بوده و اطلاعات تفکیک‌کننده‌ی افراد از بین می‌رود. در مقابل، انتخاب kهای بسیار بزرگ، علاوه بر افزایش هزینه‌ی محاسباتی، می‌تواند موجب حساسیت بیشتر مدل نسبت به نویز و تغییرات ناخواسته‌ی تصویر (مانند سایه‌ها و تغییرات نور) گردد.
در این پروژه، انتخاب k بر پایه‌ی بررسی هم‌زمان **کیفیت بازسازی تصاویر**، **پایداری ویژگی‌ها در برابر تغییرات نوری**، و **کارایی محاسباتی** انجام شده است. بنابراین مقدار نهایی k به‌گونه‌ای انتخاب شده که در عین حفظ اطلاعات مؤثر برای تشخیص، از پیچیدگی غیرضروری جلوگیری شود.

۲.۷.۷ هایپرپارامترهای Tensorfaces مبتنی بر HOSVD

در الگوریتم Tensorfaces پیاده‌سازی‌شده در این پروژه، داده‌های آموزشی در قالب یک تانسور مرتبه‌سوم

$$\mathcal{D} \in \mathbb{R}^{S \times L \times P}$$

مدل‌سازی شده‌اند، که در آن S تعداد افراد (Subjects)، L تعداد شرایط نوری (Lights) و $P = H \times W$ تعداد کل پیکسل‌ها (پس از تغییر اندازه‌ی تصویر) است. بر این اساس، تجزیه HOSVD شامل سه رتبه‌ی برش‌خورده به‌ترتیب زیر می‌باشد:

- r_S: رتبه‌ی مد هویت افراد (Subject Mode)،
- r_L: رتبه‌ی مد شرایط نورپردازی (Light Mode)،
- r_P: رتبه‌ی مد پیکسل‌ها (Pixel Mode) که زیرفضای پیکسلی تصویر را تشکیل می‌دهد.

پارامتر r_P میزان فشرده‌سازی اطلاعات پیکسلی و ساختار مکانی چهره را کنترل می‌کند. انتخاب مقادیر کوچک‌تر برای این پارامتر منجر به کاهش ابعاد فضای ویژگی و افزایش سرعت محاسباتی می‌شود، اما در صورت انتخاب بیش از حد کوچک، ممکن است اطلاعات ساختاری مهم چهره از بین برود.
پارامترهای r_S و r_L به‌ترتیب ظرفیت مدل در نمایش تغییرات مربوط به هویت افراد و تغییرات ناشی از نورپردازی را تعیین می‌کنند. در صورت انتخاب مقادیر کوچک برای این رتبه‌ها، مدل ممکن است توانایی کافی برای جداسازی هویت‌ها یا مدلسازی تغییرات نوری را نداشته باشد، در حالی که انتخاب مقادیر بسیار بزرگ می‌تواند هزینه‌ی محاسباتی را افزایش داده و در برخی موارد خطر بیش‌برازش را نیز تشدید کند.
در این پروژه، برای تحلیل حساسیت الگوریتم Tensorfaces نسبت به هایپرپارامترها، هر یک از رتبه‌های r_S، r_L و r_P به‌صورت جداگانه تغییر داده شده و سایر پارامترها ثابت نگه داشته شده‌اند. نتایج این آزمایش‌ها نشان می‌دهند که انتخاب مقادیر متعادل برای رتبه‌ها نقش کلیدی در دستیابی به دقت مناسب و کنترل هزینه‌ی محاسباتی دارد.

۳.۷.۷ ملاحظات عملی در انتخاب پارامترها

در پیاده‌سازی حاضر، انتخاب هایپرپارامترها بر اساس ملاحظات عملی زیر انجام شده است:

- ایجاد توازن میان دقت تشخیص و هزینه‌ی محاسباتی (زمان آموزش و آزمون)،
- جلوگیری از بیش‌برازش از طریق محدودسازی ظرفیت مدل و کنترل رتبه‌ها،
- بررسی کیفی بازسازی تصاویر به‌عنوان معیاری برای ارزیابی میزان حفظ ساختار چهره،
- توجه به پایداری مدل در برابر تغییرات شدید نورپردازی که از ویژگی‌های اصلی دیتاست Yale است،
- و سازگاری تنظیمات با اندازه‌ی داده‌ها و امکان اجرای چندین آزمایش برای تحلیل‌های تجربی.

در مجموع، این مرحله نشان می‌دهد که انتخاب هایپرپارامترها نه‌تنها یک تنظیم عددی ساده، بلکه بخشی اساسی از فرآیند طراحی آزمایش است. تنظیم مناسب پارامترها می‌تواند پایداری و قابلیت تعمیم مدل را افزایش دهد، هرچند معمولاً این بهبود با هزینه‌ی محاسباتی بیشتر همراه خواهد بود.

۸ نتایج عددی

در این بخش، نتایج عددی حاصل از اجرای عملی الگوریتم‌های مختلف تشخیص چهره بر روی دیتاست Extended Yale B ارائه و تحلیل می‌شوند. در این مطالعه، چهار الگوریتم شامل Eigenfaces، LBP، FisherFaces و Tensorfaces مبتنی بر تجزیه‌ی HOSVD مورد پیاده‌سازی و مقایسه قرار گرفته‌اند.
تمامی آزمایش‌ها و نتایج گزارش‌شده بر روی سیستمی با پردازنده‌ی Intel Xeon E5-2697 v4 @ 2.30GHz با 32 هسته‌ی منطقی و 32GB حافظه‌ی اصلی انجام شده‌اند.
به‌منظور ارزیابی منصفانه‌ی عملکرد الگوریتم‌ها، داده‌ها به دو مجموعه‌ی آموزشی و آزمون تقسیم شده‌اند. در این تقسیم‌بندی، 80% از تصاویر برای آموزش مدل‌ها و 20% از داده‌ها برای ارزیابی عملکرد آن‌ها در نظر گرفته شده است. تقسیم داده‌ها به‌صورت تصادفی و stratified انجام شده است، به‌گونه‌ای که توزیع نمونه‌های هر فرد در هر دو مجموعه‌ی آموزشی و آزمون به‌طور یکنواخت حفظ شود. در نهایت، 1433 تصویر برای آموزش و 359 تصویر برای آزمون مورد استفاده قرار گرفته است.

۱.۸ معیارهای ارزیابی عملکرد

برای ارزیابی عملکرد الگوریتم‌های تشخیص چهره از معیارهای متداول در این حوزه استفاده شده است. مهم‌ترین معیار مورد استفاده، **دقت تشخیص (Accuracy)** است که نسبت تعداد نمونه‌های به‌درستی طبقه‌بندی‌شده به کل نمونه‌های موجود در مجموعه‌ی آزمون را نشان می‌دهد.
علاوه بر دقت، **زمان اجرای الگوریتم‌ها** نیز به‌عنوان معیاری برای سنجش کارایی محاسباتی در نظر گرفته شده است. زمان گزارش‌شده شامل مجموع زمان آموزش مدل و زمان پیش‌بینی بر روی مجموعه‌ی آزمون می‌باشد. این معیارها امکان مقایسه‌ی عددی میان روش‌های مختلف را از منظر **دقت** و **هزینه‌ی محاسباتی** فراهم می‌کنند.

۲.۸ دقت تشخیص الگوریتم‌ها

دقت تشخیص الگوریتم‌ها بر روی مجموعه‌ی آزمون دیتاست Yale محاسبه شده و نتایج آن در جدول ۲ ارائه شده است.

جدول ۲: دقت تشخیص الگوریتم‌ها بر روی مجموعه‌ی آزمون

Algorithm	Recognition Accuracy (%)
Eigenfaces	42.62
LBP	71.59
Tensorfaces (HOSVD)	72.70
FisherFaces	85.79

طبق جدول ۲، الگوریتم FisherFaces با دقت 85.79% بهترین عملکرد را در میان روش‌های بررسی‌شده به دست آورده است. این نتیجه قابل انتظار است، زیرا روش FisherFaces مبتنی بر LDA مستقیماً برای بیشینه‌سازی فاصله‌ی بین کلاس‌ها و کاهش پراکندگی درون‌کلاسی طراحی شده است؛ در نتیجه نسبت به تغییرات شدید نورپردازی، پایداری بیشتری در استخراج ویژگی‌های مرتبط با هویت افراد دارد.
در مقابل، الگوریتم Eigenfaces با دقت 42.62% ضعیف‌ترین عملکرد را نشان می‌دهد. علت این مسئله آن است که PCA مؤلفه‌های اصلی را بر اساس بیشترین واریانس کلی استخراج می‌کند، در حالی که در دیتاست Yale بخش قابل توجهی از واریانس ناشی از تغییرات نور و سایه است و لزوماً به هویت افراد مربوط نمی‌شود. بنابراین، مدل ممکن است ویژگی‌های وابسته به روشنایی را به‌جای ویژگی‌های هویتی یاد بگیرد و در تشخیص افراد دچار خطا گردد.
در میان روش‌های میانی، الگوریتم Tensorfaces با دقت 72.70% عملکردی نزدیک به LBP با دقت 71.59% ارائه داده است. این موضوع نشان می‌دهد که مدل‌سازی چندخطی داده‌ها و استفاده از ساختار تانسوری می‌تواند در

استخراج ویژگی‌های پایدارتر در برابر تغییرات تصویربرداری مفید باشد، هرچند که این روش همچنان از نظر دقت به FisherFaces نمی‌رسد.

۳.۸ زمان اجرای الگوریتم‌ها

زمان اجرای الگوریتم‌ها شامل زمان آموزش و پیش‌بینی بر روی مجموعه‌ی آزمون اندازه‌گیری شده است. نتایج در جدول ۳ ارائه شده‌اند.

جدول ۳: زمان اجرای الگوریتم‌ها (آموزش + آزمون)

Algorithm	Execution Time (s)
Eigenfaces	16.19
LBP	20.78
Tensorfaces (HOSVD)	40.66
FisherFaces	0.48

همان‌طور که در جدول ۳ مشاهده می‌شود، الگوریتم FisherFaces علاوه بر بالاترین دقت، کمترین زمان اجرا را نیز دارد. این ویژگی نشان می‌دهد که با کاهش بعد مؤثر داده‌ها و استفاده از LDA، فرآیند استخراج ویژگی و طبقه‌بندی بسیار سریع انجام می‌گیرد و این روش برای کاربردهای عملی و حتی بلادرنگ مناسب است.
در مقابل، الگوریتم Tensorfaces با زمان اجرای 40.66s پرهزینه‌ترین روش از نظر محاسباتی محسوب می‌شود. علت اصلی این موضوع، پیچیدگی بالای عملیات تجزیه‌ی تانسوری HOSVD و اجرای چندین SVD بر روی ماتریس‌های بزرگ حاصل از unfolding است. در نتیجه، هرچند Tensorfaces دقت مناسبی ارائه می‌دهد، اما هزینه‌ی محاسباتی بالای آن در کاربردهای محدودیت‌منابع می‌تواند چالش‌برانگیز باشد.
الگوریتم LBP نیز به دلیل استخراج ویژگی محلی برای کل تصویر، زمان اجرای بالاتری نسبت به Eigenfaces دارد. همچنین زمان Eigenfaces در این پیاده‌سازی به دلیل محاسبات PCA و هزینه‌ی پیش‌بینی در فضای ویژگی قابل توجه بوده است.

۱.۳.۸ تحلیل فنی هزینه‌ی محاسباتی Tensorfaces

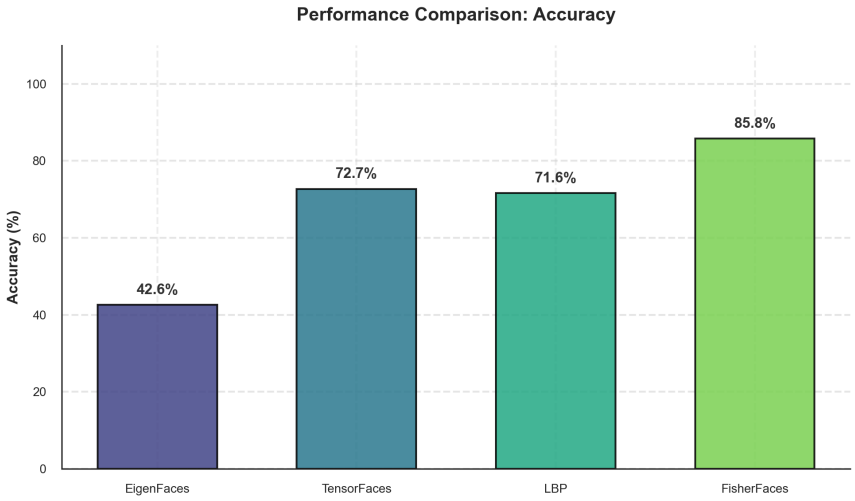
هزینه‌ی محاسباتی بالای الگوریتم Tensorfaces عمدتاً ناشی از مرحله‌ی تجزیه‌ی HOSVD است. در پیاده‌سازی حاضر، داده‌ها به‌صورت یک تانسور سه‌مدی سازمان‌دهی شده‌اند:

$$\mathcal{D} \in \mathbb{R}^{S \times L \times P},$$

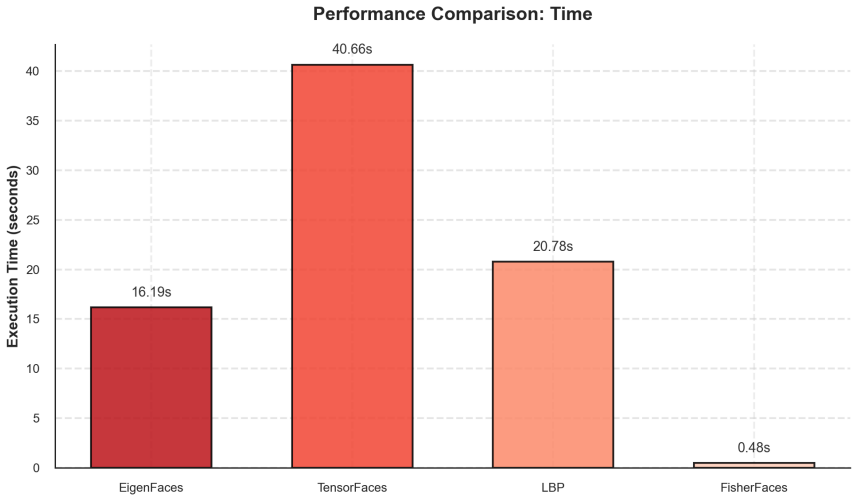
که در آن S تعداد افراد، L تعداد شرایط نوری، و $P = H \times W$ تعداد پیکسل‌ها (پس از بردارسازی تصویر) است. برای اجرای HOSVD، تانسور در هر مد unfold شده و SVD بر روی ماتریس‌های حاصل اعمال می‌گردد. برای مثال، در مد افراد ماتریسی با ابعاد $S \times (LP)$ و در مد نورپردازی ماتریسی با ابعاد $L \times (SP)$ تشکیل می‌شود. بنابراین، اعمال SVD بر روی این ماتریس‌های بزرگ می‌تواند هزینه‌ی زمانی قابل توجهی ایجاد کند.
به همین دلیل، هرچند Tensorfaces از نظر دقت عملکرد مناسبی دارد، اما زمان آموزش آن در مقایسه با سایر روش‌ها بیشتر است. در کاربردهای بلادرنگ یا سامانه‌های با محدودیت منابع، این هزینه‌ی محاسباتی می‌تواند عامل تعیین‌کننده باشد. یکی از راهکارهای پیشنهادی برای کاهش هزینه، استفاده از روش‌های Randomized SVD یا محدود کردن رتبه‌های برش‌خورده (r_S, r_L, r_P) به‌صورت بهینه است.

۴.۸ مقایسه‌ی جامع دقت و کارایی محاسباتی

به‌منظور تحلیل بهتر توازن میان دقت تشخیص و هزینه‌ی محاسباتی، نمودارهای تجمیعی دقت و زمان اجرای الگوریتم‌ها در شکل‌های ۱۴ و ۱۵ ارائه شده‌اند.



شکل ۱۴: مقایسه‌ی دقت تشخیص الگوریتم‌های مختلف



شکل ۱۵: مقایسه‌ی زمان اجرای الگوریتم‌های مختلف

مطابق شکل ۱۴، الگوریتم FisherFaces بهترین دقت را ارائه داده است، در حالی که Eigenfaces کمترین دقت را به خود اختصاص داده است. از طرف دیگر، طبق شکل ۱۵، الگوریتم Tensorfaces بیشترین زمان اجرا را دارد و بنابراین استفاده از آن در سیستم‌های بلادرنگ نیازمند منابع محاسباتی بالاتر است.

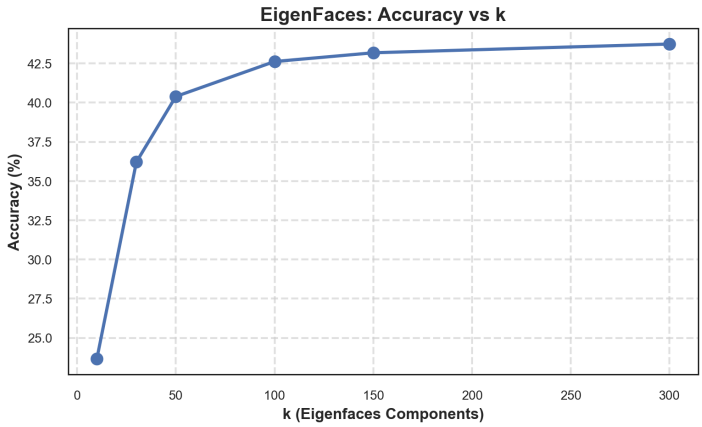
در مجموع، الگوریتم FisherFaces از نظر دقت و زمان اجرا، بهترین توازن را ارائه داده است. LBP و Tensorfaces دقت نسبتاً نزدیک دارند، اما LBP از نظر زمان اجرا مقرون‌به‌صرفه‌تر است. بنابراین انتخاب الگوریتم نهایی بسته به نیاز کاربرد (دقت بالا یا سرعت بالا) خواهد بود.

۵.۸ بررسی تأثیر هایپرپارامترها

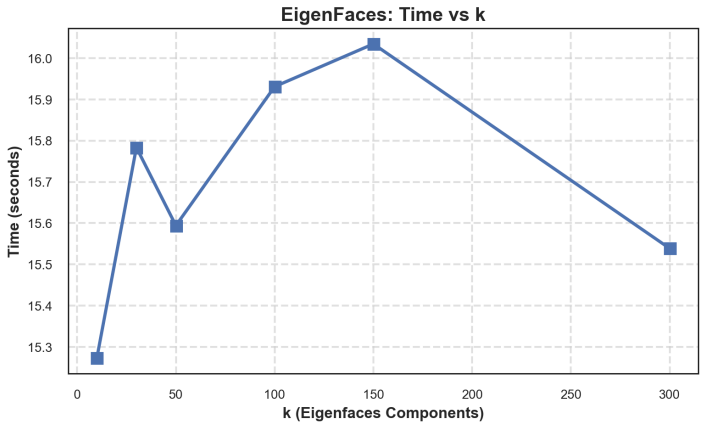
برای تحلیل دقیق‌تر رفتار الگوریتم‌ها و بررسی حساسیت آن‌ها نسبت به تنظیمات مختلف، تأثیر هایپرپارامترهای اصلی در الگوریتم‌های Eigenfaces و Tensorfaces به‌صورت تجربی بررسی شده است.

۱.۵.۸ Eigenfaces

در الگوریتم Eigenfaces، پارامتر اصلی، تعداد مؤلفه‌های اصلی یا تعداد Eigenfaceها (پارامتر k) است. در شکل ۱۶، تغییرات دقت و زمان اجرا بر حسب مقدار k نمایش داده شده است.



(آ) تغییرات دقت تشخیص بر حسب k



(ب) تغییرات زمان اجرا بر حسب k

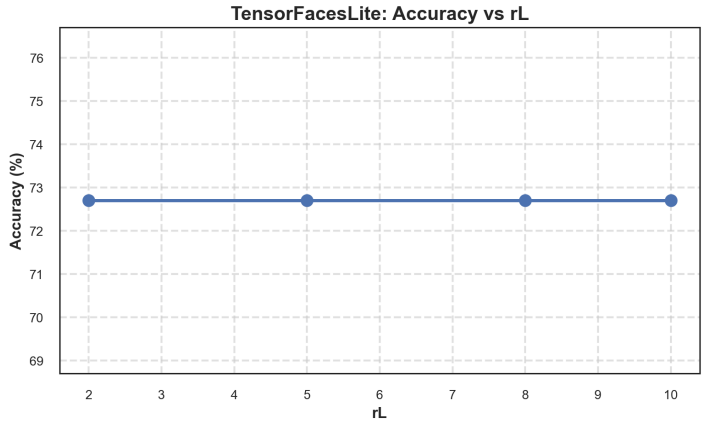
شکل ۱۶: تأثیر تعداد مؤلفه‌های اصلی (k) بر عملکرد Eigenfaces

مطابق شکل ۱۶آ، با افزایش k، دقت تشخیص در ابتدا رشد محسوسی دارد، زیرا ابعاد بیشتری از فضای ویژگی برای نمایش تغییرات چهره استفاده می‌شود. با این حال، پس از مقدار حدودی $k \approx 150$، منحنی دقت وارد ناحیه‌ی **اشباع** می‌شود و افزایش بیشتر k بهبود قابل توجهی ایجاد نمی‌کند. این مسئله نشان می‌دهد که اطلاعات مفید اصلی در تعداد محدودی از مؤلفه‌های PCA نهفته است و مؤلفه‌های بعدی اغلب شامل نویز یا تغییرات غیرضروری هستند.
از سوی دیگر، طبق شکل ۱۶ب، زمان اجرای الگوریتم با افزایش k تغییرات محدودی دارد و در بازه‌های مختلف تقریباً ثابت باقی می‌ماند. بنابراین انتخاب مقدار k بیشتر تابعی از توازن میان دقت و پیچیدگی مدل (و همچنین کیفیت بازسازی) خواهد بود تا افزایش محسوس هزینه‌ی زمانی.

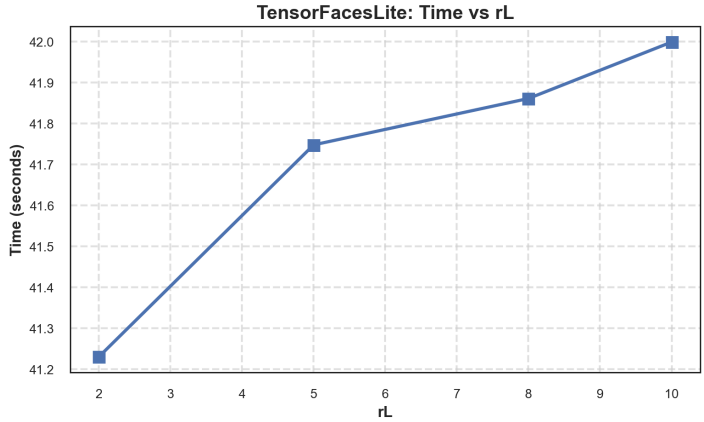
۲.۵.۸ Tensorfaces

در الگوریتم Tensorfaces پیاده‌سازی‌شده در این پروژه، پارامترهای اصلی، رتبه‌های برش‌خورده در تجزیه‌ی HOSVD هستند که شامل r_S، r_L و r_P می‌شوند. برای بررسی حساسیت عملکرد به این پارامترها، هر یک از این رتبه‌ها جداگانه تغییر داده شده و سایر پارامترها ثابت نگه داشته شده‌اند.

تحلیل پارامتر r_L نتایج مربوط به پارامتر r_L در شکل ۱۷ ارائه شده است.



(آ) دقت تشخیص بر حسب r_L



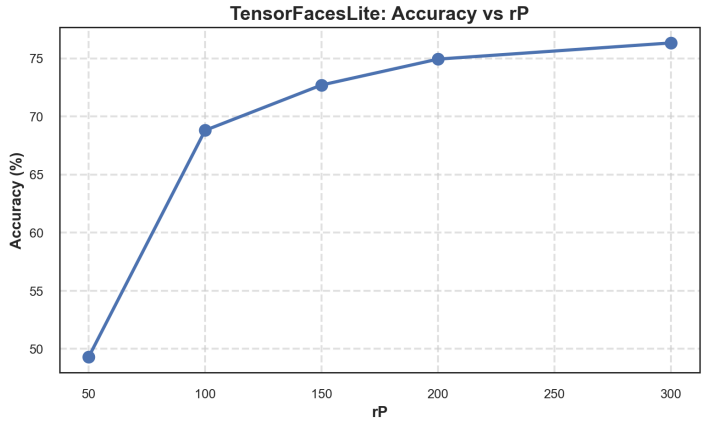
(ب) زمان اجرا بر حسب r_L

شکل ۱۷: تأثیر رتبه‌ی r_L بر عملکرد Tensorfaces

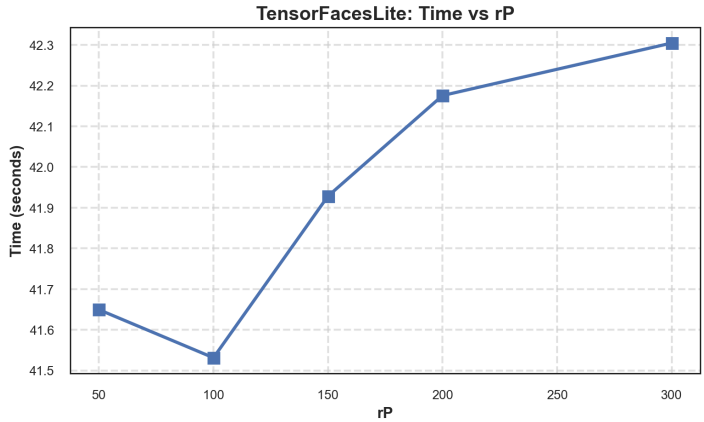
همان‌طور که در شکل ۱۷ مشاهده می‌شود، تغییر رتبه‌ی r_L (که بیانگر تعداد مؤلفه‌های انتخاب‌شده در مد نورپردازی است) در بازه‌ی بررسی‌شده تأثیر معناداری بر دقت تشخیص نداشته است. به‌عبارت دیگر، دقت مدل برای مقادیر مختلف r_L تقریباً ثابت باقی مانده و این موضوع نشان می‌دهد که در این تنظیمات، مدل پس از رسیدن به حداقل ظرفیت لازم برای مدل‌سازی تغییرات نوری، به حالت **اشباع** می‌رسد و افزایش بیشتر r_L باعث استخراج اطلاعات تمایزدهنده‌ی اضافی نمی‌شود.

از سوی دیگر، زمان اجرا با افزایش r_L افزایش جزئی و قابل انتظار دارد، زیرا افزایش رتبه در این مد مستقیماً باعث افزایش ابعاد زیرفضای نگه‌داشته‌شده و در نتیجه افزایش هزینه‌ی محاسباتی در مراحل فرافکنی و بازسازی می‌شود. بنابراین، انتخاب مقدار کوچک‌تر برای r_L (تا جایی که افت دقت ایجاد نشود) می‌تواند راهبرد مناسبی برای کاهش هزینه‌ی محاسباتی باشد.

تحلیل پارامتر r_P نتایج مربوط به پارامتر r_P در شکل ۱۸ ارائه شده است.



(آ) دقت تشخیص بر حسب r_P



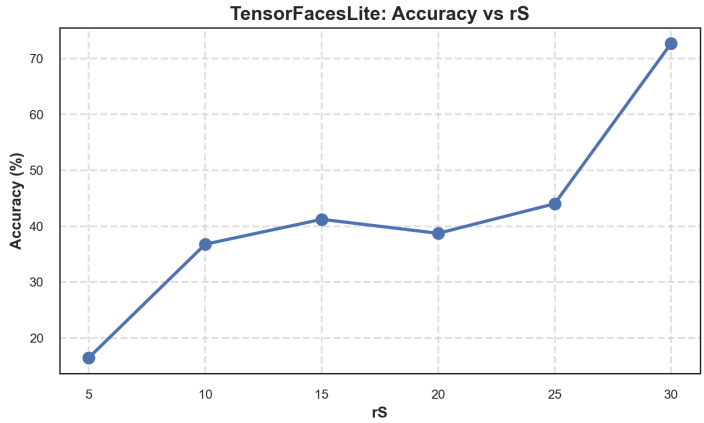
(ب) زمان اجرا بر حسب r_P

شکل ۱۸: تأثیر رتبه‌ی r_P بر عملکرد Tensorfaces

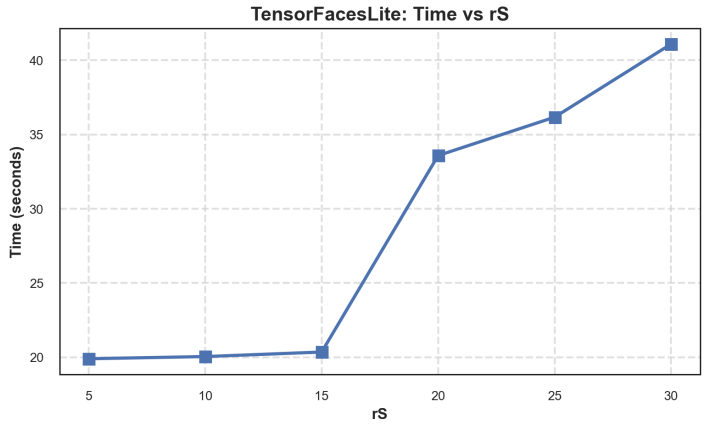
شکل ۱۸ نشان می‌دهد که پارامتر r_P (که رتبه‌ی مد پیکسل‌ها یا بُعد ویژگی‌های مکانی را کنترل می‌کند) نسبت به دو پارامتر دیگر، **اثرگذاری مستقیم‌تری بر دقت تشخیص دارد**. در مقادیر کوچک r_P، مدل قادر به حفظ جزئیات ساختاری کافی از چهره‌ها نیست و بنابراین افت قابل توجهی در دقت مشاهده می‌شود. با افزایش r_P، مدل اطلاعات مکانی بیشتری را در زیرفضای کاهش‌یافته حفظ کرده و دقت به‌صورت معناداری افزایش می‌یابد، تا جایی که در مقادیر بالاتر وارد ناحیه‌ی اشباع می‌شود.

از منظر هزینه‌ی محاسباتی، زمان اجرا در مقایسه با تغییرات دقت، افزایش بسیار ملایم‌تری دارد؛ زیرا در این پیاده‌سازی، هزینه‌ی اصلی مربوط به تجزیه‌ی اولیه و ساختار کلی تانسور است و افزایش r_P در محدوده‌ی بررسی‌شده تنها باعث افزایش محدود در ابعاد بردار ویژگی می‌شود. بنابراین می‌توان نتیجه گرفت که افزایش r_P در بازه‌های میانی، **بهترین توازن دقت/هزینه را ایجاد می‌کند** و نقش کلیدی در ارتقای عملکرد دارد.

تحلیل پارامتر r_S نتایج مربوط به پارامتر r_S در شکل ۱۹ ارائه شده است.



(آ) دقت تشخیص بر حسب r_S



(ب) زمان اجرا بر حسب r_S

شکل ۱۹: تأثیر رتبه‌ی r_S بر عملکرد Tensorfaces

پارامتر r_S نشان‌دهنده‌ی رتبه‌ی مد افراد/هویت‌ها است و در واقع ظرفیت مدل در نمایش تغییرات مرتبط با هویت افراد را تعیین می‌کند. طبق شکل ۱۹، افزایش r_S در ابتدا باعث رشد شدید دقت می‌شود، زیرا مدل امکان نمایش بهتر اختلافات بین افراد مختلف را پیدا می‌کند. این رفتار نشان می‌دهد که در مقادیر کوچک r_S، مدل **کم‌ظرفیت** بوده و فضای ویژگی برای تفکیک هویت افراد کافی نیست.

در ادامه، با رسیدن r_S به مقادیر بالاتر، افزایش دقت همچنان ادامه دارد، اما روند رشد به سمت اشباع می‌رود. این مسئله نشان می‌دهد که پس از سطح مشخصی از ظرفیت، اطلاعات اضافه‌شده بیشتر جنبه‌ی تکمیل جزئیات دارد و تأثیر آن بر تفکیک کلاس‌ها کمتر می‌شود.

از نظر زمان اجرا، نکته‌ی مهم این است که افزایش r_S باعث افزایش محسوس زمان محاسباتی می‌شود. این افزایش می‌تواند به دو عامل اصلی مرتبط باشد:

- افزایش ابعاد زیرفضای هویت و در نتیجه افزایش هزینه‌ی فرافکنی،
- و پیچیده‌تر شدن محاسبات مربوط به تانسور هسته و بازسازی.

بنابراین، r_S حساس‌ترین پارامتر از نظر ایجاد **رقابت بین دقت و زمان** محسوب می‌شود و انتخاب مقدار بهینه‌ی آن نقش تعیین‌کننده‌ای در کاربردپذیری مدل دارد.

۶.۸ جمع‌بندی تحلیل نتایج

نتایج عددی و نمودارها نشان می‌دهند که:

- الگوریتم FisherFaces بهترین گزینه از نظر دقت و سرعت اجرا است (دقت 85.79% و زمان 0.48s) و برای کاربردهای عملی و بلادرنگ مناسب می‌باشد.
- الگوریتم Tensorfaces اگرچه دقت خوبی ارائه می‌دهد (72.70%)، اما هزینه‌ی محاسباتی بالاتری نسبت به روش‌های سبک‌تر دارد (40.66s)، بنابراین در شرایطی مناسب‌تر است که منابع محاسباتی کافی در اختیار باشد.
- الگوریتم LBP توازن قابل قبولی میان دقت و هزینه دارد (دقت 71.59% و زمان 20.78s) و می‌تواند جایگزین مناسبی در سیستم‌های سبک‌تر باشد.
- الگوریتم Eigenfaces ساده و قابل پیاده‌سازی است، اما دقت پایین آن (42.62%) نشان می‌دهد که نسبت به تغییرات نورپردازی حساس بوده و برای دیتاست‌هایی مانند Extended Yale B که تغییرات شدید نور دارند، عملکرد مطلوبی ندارد.

در نهایت، می‌توان نتیجه گرفت که بسته به نیاز کاربرد، الگوریتم FisherFaces برای بیشینه‌سازی دقت و کارایی، و LBP برای شرایط محدودیت منابع، انتخاب‌های مناسبی هستند.

۹ پیوست

در این بخش، جزئیات تکمیلی مربوط به پیاده‌سازی، تنظیمات آزمایش‌ها و نتایج اضافی ارائه می‌شود. هدف از ارائه‌ی این پیوست، افزایش شفافیت فرآیند آزمایش‌ها و فراهم‌کردن امکان بازتولید نتایج برای خواننده است.

۱.۹ کد پایتون

پیاده‌سازی کامل الگوریتم‌های مورد بررسی در این پروژه با استفاده از زبان برنامه‌نویسی Python انجام شده است. در این پیاده‌سازی از کتابخانه‌های متداولی نظیر NumPy، Scikit-learn، Matplotlib، Scikit-image و Seaborn استفاده شده است. کد شامل بخش‌های اصلی زیر است:

- بارگذاری و پیش‌پردازش تصاویر دیتاست Extended Yale B،
- پیاده‌سازی الگوریتم‌های Eigenfaces، LBP، Fisherfaces و Tensorfaces،
- آموزش مدل‌ها و پیش‌بینی روی مجموعه‌ی آزمون،
- ترسیم نمودارهای دقت تشخیص و زمان اجرا،
- و تولید خروجی‌های تصویری برای تحلیل کیفی نتایج.

به‌منظور جلوگیری از طولانی‌شدن متن گزارش، کد کامل در مخزن جداگانه ارائه شده است.

۲.۹ شبه‌کد الگوریتم‌ها

الگوریتم Eigenfaces

۱: دریافت مجموعه‌ای از تصاویر آموزشی
۲: بردارسازی هر تصویر به یک بردار یک‌بعدی
۳: محاسبه‌ی تصویر میانگین: $\mu = \frac{1}{N}\sum_{i=1}^{N} X_i$
۴: مرکزسازی تصاویر: $X_{ci} = X_i - \mu$
۵: تشکیل ماتریس داده: $X = [X_{c1}, X_{c2}, ..., X_{cN}]$
۶: محاسبه‌ی ماتریس کوواریانس و استخراج مقادیر و بردارهای ویژه
۷: انتخاب k بردار ویژه با بزرگ‌ترین مقادیر ویژه
۸: فرافکنی تصاویر آموزشی به زیرفضای k-بعدی
۹: ذخیره‌ی بردارهای فرافکنی‌شده (Weights)
۱۰: برای هر تصویر آزمون:
۱۱: مرکزسازی تصویر آزمون: $x_c = x - \mu$
۱۲: فرافکنی به زیرفضای k-بعدی
۱۳: محاسبه‌ی نزدیک‌ترین همسایه با معیار فاصله‌ی اقلیدسی

الگوریتم Tensorfaces مبتنی بر HOSVD

۱: سازمان‌دهی داده‌های آموزشی به‌صورت تانسور: $\mathcal{D} \in \mathbb{R}^{S \times L \times P}$
۲: **تعریف ابعاد تانسور:**
۳: S: تعداد افراد (Subjects)
۴: L: تعداد شرایط نوری (Lights)
۵: $P = H \times W$: تعداد پیکسل‌ها پس از بردارسازی تصویر
۶: اعمال نرمال‌سازی یا مقیاس‌بندی مناسب روی داده‌ها
۷: اعمال تجزیه‌ی HOSVD و استخراج ماتریس‌های مدی:
۸: $U_S \in \mathbb{R}^{S \times r_S}$، $U_L \in \mathbb{R}^{L \times r_L}$، $U_P \in \mathbb{R}^{P \times r_P}$
۹: محاسبه‌ی هسته‌ی تانسور: $\mathcal{G} = \mathcal{D} \times_1 U_S^\top \times_2 U_L^\top \times_3 U_P^\top$
۱۰: برای هر نمونه‌ی آموزشی:
۱۱: استخراج نمایش فشرده در فضای ویژگی (بر اساس رتبه‌ها)
۱۲: برای هر نمونه‌ی آزمون:
۱۳: فرافکنی/بازنمایی در فضای ویژگی
۱۴: تعیین کلاس خروجی با معیار تصمیم‌گیری پیاده‌سازی‌شده در مدل

الگوریتم Local Binary Patterns Histograms (LBPH)

۱: دریافت مجموعه‌ی تصاویر آموزشی
۲: **for** هر تصویر I **do**
۳: برای هر پیکسل p:
۴: مقایسه مقدار p با همسایگان آن
۵: اگر مقدار همسایه $\geq$ مرکز باشد، ۱ و در غیر این‌صورت ۰
۶: تشکیل کد دودویی LBP برای p
۷: تقسیم تصویر LBP به بلوک‌های محلی
۸: محاسبه‌ی هیستوگرام LBP برای هر بلوک
۹: ترکیب هیستوگرام بلوک‌ها و تشکیل بردار ویژگی
۱۰: **for end**
۱۱: برای هر تصویر آزمون:
۱۲: استخراج LBP و بردار ویژگی
۱۳: تعیین کلاس با کمینه‌کردن فاصله‌ی اقلیدسی

الگوریتم Fisherfaces

۱: دریافت تصاویر آموزشی و برچسب کلاس‌ها
۲: بردارسازی تصاویر
۳: اعمال کاهش بعد با PCA برای جلوگیری از کمبود نمونه
۴: محاسبه‌ی ماتریس پراکندگی درون‌کلاسی S_W و بین‌کلاسی S_B
۵: حل مسئله‌ی بیشینه‌سازی نسبت S_B به S_W
۶: استخراج ماتریس انتقال: W_{FLD}
۷: فرافکنی تصاویر آموزشی به فضای LDA
۸: آموزش طبقه‌بند خطی روی ویژگی‌های LDA
۹: برای هر تصویر آزمون:
۱۰: فرافکنی با PCA و سپس LDA
۱۱: پیش‌بینی کلاس با طبقه‌بند خطی

۳.۹ تنظیمات آزمایش‌ها و پارامترها

تنظیمات آزمایش‌ها و پارامترهای اصلی

تمامی آزمایش‌ها با استفاده از تنظیمات یکسان، به‌منظور مقایسه‌ی منصفانه‌ی الگوریتم‌ها انجام شده‌اند. جدول ۴ تنظیمات اصلی آزمایش‌ها را خلاصه می‌کند.

جدول ۴: تنظیمات دقیق آزمایش‌ها و پارامترهای پیاده‌سازی‌شده

پارامتر	مقدار
دیتاست	Extended Yale B (Cropped)
تعداد کلاس‌ها (C)	28
حداکثر نمونه برای هر کلاس	64
تعداد کل نمونه‌های استفاده‌شده	$28 \times 64 = 1792$
نسبت آموزش/آزمون	%80 / %20 (Stratified)
تعداد نمونه‌های آموزش/آزمون	آموزش = 1433، آزمون = 359
بذر تصادفی (Seed)	42
اندازه‌ی تصویر	96×96
پیش‌پردازش	تغییر اندازه + کم‌کردن میانگین (Zero-Mean)
هایپرپارامترهای مدل‌ها	
طبقه‌بند	kNN ($k = 1$)
معیار فاصله	اقلیدسی (L_2)
تعداد مؤلفه‌های Eigenfaces (k)	100
تعداد مؤلفه‌های PCA در FisherFaces	50
رتبه‌های Tensorfaces	$(r_S, r_L, r_P) = (28,\ 10,\ 150)$
اندازه‌ی شبکه در LBP	8×8
تنظیمات LBP	(Uniform) $P = 8,\ R = 1$

جدول ۵: محدوده‌ی پارامترهای بررسی‌شده در تحلیل حساسیت مدل‌ها

مدل	شرح پارامتر	مقادیر بررسی‌شده
Eigenfaces	تعداد مؤلفه‌های اصلی (k)	$\{10,\ 30,\ 50,\ 100,\ 150,\ 300\}$
Tensorfaces	رتبه‌ی مد نورپردازی (r_L)	$\{2,\ 5,\ 8,\ 10\}$
Tensorfaces	رتبه‌ی مد پیکسل‌ها (r_P)	$\{50,\ 100,\ 150,\ 200,\ 300\}$
Tensorfaces	رتبه‌ی مد افراد (r_S)	$\{5,\ 10,\ 15,\ 20,\ 25,\ 30\}$

جدول ۶: تحلیل حساسیت Eigenfaces نسبت به تعداد مؤلفه‌ها (k)

k	Accuracy (%)	Time (s)
۱۰	23.68	15.2735
۳۰	36.21	15.7834
۵۰	40.39	15.5942
۱۰۰	42.62	15.9309
۱۵۰	43.18	16.0344
۳۰۰	43.73	15.5388

جدول ۷: تحلیل حساسیت Tensorfaces نسبت به رتبه نورپردازی (r_L) (با ثابت بودن $r_S = 30$ و $r_P = 150$)

r_L	Accuracy (%)	Time (s)
۲	72.70	41.2301
۵	72.70	41.7477
۸	72.70	41.8610
۱۰	72.70	41.9989

جدول ۸: تحلیل حساسیت Tensorfaces نسبت به رتبه پیکسل‌ها (r_P) (با ثابت بودن $r_S = 30$ و $r_L = 10$)

r_P	Accuracy (%)	Time (s)
۵۰	49.30	41.6495
۱۰۰	68.80	41.5313
۱۵۰	72.70	41.9284
۲۰۰	74.93	42.1756
۳۰۰	76.32	42.3046

جدول ۹: تحلیل حساسیت Tensorfaces نسبت به رتبه افراد (r_S) (با ثابت بودن $r_L = 10$ و $r_P = 150$)

r_S	Accuracy (%)	Time (s)
۵	16.43	19.8985
۱۰	36.77	20.0463
۱۵	41.23	20.3498
۲۰	38.72	33.5823
۲۵	44.01	36.1661
۳۰	72.70	41.0900

۱۰ منابع

[1] L. Eldén, *Matrix Methods in Data Mining and Pattern Recognition*, SIAM, 2019, Chap. 14.

[2] T. G. Kolda and B. W. Bader, "Tensor Decompositions and Applications," *SIAM Review*, vol. 51, no. 3, pp. 455–500, 2009. Available online: https://www.kolda.net/publication/TensorReview.pdf

[3] D. Brandoni, *(Thesis) Tensor Decompositions and Applications in Computer Vision*, Alma Mater Studiorum – Università di Bologna, A.Y. 2017–2018.

[4] M. Turk and A. Pentland, "Eigenfaces for Recognition," *Journal of Cognitive Neuroscience*, vol. 3, no. 1, pp. 71–86, 1991.

[5] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, "Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 711–720, 1997.

[6] T. Ahonen, A. Hadid, and M. Pietikäinen, "Face Description with Local Binary Patterns: Application to Face Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 12, pp. 2037–2041, 2006.

[7] M. A. O. Vasilescu and D. Terzopoulos, "Multilinear Analysis of Image Ensembles: TensorFaces," *European Conference on Computer Vision (ECCV)*, pp. 447–460, 2002.

[8] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman, "From Few to Many: Illumination Cone Models for Face Recognition under Variable Lighting and Pose," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 6, pp. 643–660, 2001.