فاز دوم پروژه

بهراد صمیمی400104267 ، آرین فضلی‌خانی 400101729

بخش 1)

برای این بخش از دیتاست اماده sportsMOT استفاده شده است که خودش دیتا ها را به بخش های اموزش و ولیدیشن و تست قسمت بندی و از طرفی در ابتدای نوت بوک section(1.2)&(1).ipynb داده های train برای fineTune کردن الگوریتم yolo اماده سازی شده اند و از طرفی در پروژه فاز 1 هم کد دیتالودر ان به دقت زده شده و اورده شده است.

بخش2.1)

کد این بخش در نوتبوک section(2.1)&(1).ipynb زده شده است.

زیر بخش 1)با توجه به اینکه SportsMOTیک دیتاست ویدئویی ورزشی است و نیاز به پردازش بلادرنگ دارد، YOLOگزینه بهتری نسبت به Faster R-CNNمحسوب می‌شود. در مسابقات ورزشی، تشخیص و رهگیری بازیکنان و توپ‌ها به‌صورت لحظه‌ای اهمیت زیادی دارد، بنابراین YOLOبه دلیل سرعت بالا و کارایی مناسب در محیط‌های ویدئویی، انتخاب بهتری برای این کار است .اگرچه دقت YOLO کمی کمتر از Faster R-CNNاست، اما با تنظیمات مناسب مثل استفاده از مدل های yolov8 و از طرفی fineTune کردن ان می‌توان دقت را افزایش داد و یک سیستم رهگیری بهینه داشت.

زیر بخش 2 و 4) بعد از fineTune کردن این مدل لاس ها در هر اپاک در نوت بوک قابل مشاهده است و تمامی معیار های ارزیابی شده و نمودار loss و دقت و دیگر معیار ها در پوشه result-fineTuneYolov8 قابل مشاهده است به طور دقیق

زیر بخش 3)در انتهای نوتبوک 2 تا عکس با مدل fineTune شده اورده شده که همانطور که مشاهده می شود فقط بازیکنان تشخیص داده شدند و داور و دیگر تماشاچیان در تشخیص نیستند و باکس ها دقیق است که نشان از اموزش خوب دارد(توضیحات تکمیلی در ویدئو)

بخش 2.2)

زیر بخش 1) رهگیری تک شیء در ویدئوهای ورزشی، مانند دنبال کردن توپ یا بازیکن، نیازمند روشی است که بتواند تغییرات ظاهری، تغییر مقیاس و حرکات سریع را به‌خوبی مدیریت کند **Siamese Networks** .به دلیل استفاده از مدلی که ایتخراج ویژگی انجام می دهد، قابلیت تعمیم‌پذیری بالایی برای رهگیری اشیا در شرایط پیچیده دارد. این روش با مقایسه ویژگی‌های استخراج‌شده از شیء در فریم‌های متوالی، می‌تواند موقعیت آن را با دقت بالا پیدا کند، حتی اگر تغییرات شدیدی در پس‌زمینه یا مقیاس آن رخ دهد. برخلاف روش‌های کلاسیک مانند **CSRT**که بیشتر به ویژگی‌های هندسی و فیلترهای همبستگی وابسته هستند، **Siamese Networks**قادر است در موقعیت‌هایی که توپ یا بازیکن دچار تغییرات چرخشی یا انسداد می‌شود، عملکرد بهتری ارائه دهد. بنابراین، این روش برای رهگیری اشیای متحرک سریع در ویدئوهای ورزشی، مانند توپ فوتبال یا بسکتبال، گزینه‌ای ایده‌آل است

زیر بخش 2)در نوتبوک گفته شده تمام خواسته ها انجام شده و ویدئو sample هم گذاشته شده و توضیحات تکمیلی در ویدئو report امده است.

زیربخش 3)

این الگوریتم در چالش های تغییر نور و تغییر مقیاس عملکرد خوبی دارد زیرا که این تغییرات در ویدئو های ورزشی کمتر هستند اما در چالش پوشیدگی مخصوصا وقتی که با بازیکنان دیگر پوشیده می شوند باعث ID Switch می شود که باعث می شود هدف درستی را دنبال نمی کند و این امر باعث پایین اوردن success rate و precision خواهد شد. (توضیحات هر یک از چالش ها در فاز 1 اورده شده است و در این بخش توضیح نداده ایم.)

زیر بخش 4)

در ابتدا با معیار ها success rate و precision بررسی شده برای ویدئو نمونه سپس بر روی 10 ویدئو تصادفی با انتخاب اولیه تصادفی این معیار ها برای الگوریتم بررسی شده اند.

زیربخش 5)

همانطور که گفته شد این الگوریتم به دلیل استفاده از ویژگی های ظاهری نقطه ضعف کمتری دارد اما در کل به هنگام پوشیدگی باعث می شود هدف اشتباهی را دنبال کند زیرا 2 تا بازیکن 1 تیم از دور و در دوربین بسیار شباهت دارند و این امر باعث شده که وقتی 2 بازیکن روی هم قرار می گیرند بعد از جدا شدن هدف اشتباهی را دنبال کند این الگوریتم که درواقع پدیده شبیه به ID switch رخ می دهد و این باعث ضعف الگوریتم در معیار های ارزیابی خواهد شد.

زیر بخش 6)

این بخش امتیازی هم در نوتبوک پیاده سازی شده است برای هدف نمونه در ویدئو نمونه.

بخش 2.3)

زیربخش 1)انتخاب ما برای این بخش الگوریتم byte Track است زیرا که برای دیتاست های ورزشی به این دلیل که معمولا شی ها سرعت بالاتری دارند ممکن است در برخی از frame ها خوب detect نشوند و الگوریتم تخصیص byte track به دلیل در نظر گرفتن box ها با confidence پایین انها را به خوبی ردیابی می کند و پایداری بیشری در ردیابی اجسم با سرعت بالا دارد. از طرفی برای تحلیل های بی درنگ نیز به دلیل سرعت بالایی که دارد گزینه مناسب تری است زیرا مانند الگوریتم های fairMOT و Deep SORT نیاز به یک شبکه عمیق در کنار ان نیز ندارد و در کل یک پکیج خوب به نسبت سرعت و دقت ارائه می دهد

زیربخش 2)به این صورت است که ابتدا الگوریتم detection روی یک فریم اجرا می شود و سپس box های استخراج شده (x, y, h, w) به همراه نمره اعتماد ان باکس به الگوریتم tracking داده می شود که انگاه الگوریتم ترکینگ الگورتیم تخصیص را اجرا می کند که این الگوریتم در الگوریتم انتخابی ما در بخش بعدی به طور کامل تر توضیح داده شده است

زیر بخش 3)در الگوریتم **ByteTrack**، فرآیند تخصیص اشیا بین فریم‌های مختلف به صورت دو مرحله‌ای انجام می‌شود. ابتدا، اشیایی که دارای (confidence score)بالا هستند، از خروجی تشخیص‌دهنده استخراج شده و با استفاده از الگوریتم **IoU-based Hungarian Matching**با ردگیری‌های

قبلی تطبیق داده می‌شوند. این مرحله فقط اشیای قابل‌اعتماد را در نظر می‌گیرد. پس از این مرحله، اشیایی که تطبیق نشده‌اند و ردگیری‌های باقی‌مانده وارد فاز دوم می‌شوند.

در مرحله دوم، **ByteTrack** برخلاف روش‌های سنتی، اشیای با نمره اطمینان پایین را نیز در نظر گرفته و مجدداً تلاش می‌کند تا آن‌ها را با ردگیری‌های باقی‌مانده تطبیق دهد. این کار باعث می‌شود اشیایی که به دلیل خطای تشخیص‌دهنده حذف شده‌اند، در فرآیند ردیابی باقی بمانند. برای این کار، از همان الگوریتم **Hungarian Matching** ولی این‌بار با در نظر گرفتن تمام اشیا، حتی آن‌هایی که نمره اطمینان پایین دارند، استفاده می‌شود. در نهایت، اشیایی که در هیچ‌یک از این دو مرحله تطبیق پیدا نکنند، به‌عنوان اشیای جدید ثبت شده یا از لیست ردگیری حذف می‌شوند، که باعث افزایش دقت و پایداری ردگیری در **ByteTrack** نسبت به روش‌های پیشین می‌شود.

حال خود الگوریتم Hungarian Matching به این صورت است که بین 2 تا فریم در بین تمام box ها می اید و ماتریس هزینه ها را می سازد و اگر به طور گرافی به ان نگاه کنیم و هر باکس را یک راس و بین هر 2 راس فاصله ان ها را بر اساس معیار هایی مانند IoU در نظر بگیریم سپس یک تطابق کامل یا مچینگ با کمینه وزن ها در بین این ها پیدا می کند و انگار پیشبینی می کند که هر باکس در فریم قبل به چه فریمی در اینده خواهد رفت.

زیربخش 4) این کار انجام شده و در فایل پروژه قرار داده شده است.

زیربخش5) این ارزیابی برای 5 تا از ویدئو ها انجام شده و کد قابلیت این را دارد روی هر تعداد ویدئو معیار ها را ارزیابی کند.

زیربخش 6) چالش‌های مانند تغییر شناسه(ID Switch) ، تغییر نور و پوشیدگی (Occlusion) در ByteTrack با مکانیزم تخصیص دو مرحله‌ای مدیریت می‌شوند. این الگوریتم ابتدا اشیای با نمره اطمینان بالا را به ردگیری‌های قبلی متصل کرده و سپس اشیای کم‌اطمینان را بررسی می‌کند تا از حذف نادرست آن‌ها جلوگیری شود، که باعث کاهش ID Switch می‌شود. در شرایط تغییر نور، چون ByteTrack تنها به اطلاعات مکانی و IoU متکی است و برخلاف روش‌هایی مانند FairMOT از ویژگی‌های بصری پیچیده استفاده نمی‌کند، ممکن است دقت آن کاهش یابد. در مواقع پوشیدگی، اگر یک شیء به‌طور موقت ناپدید شود اما دوباره ظاهر گردد، این الگوریتم با در نظر گرفتن موقعیت‌های قبلی و استفاده از تخصیص مجدد، احتمال از دست رفتن آن را کاهش می‌دهد، اما در صورت پوشیدگی طولانی‌مدت، امکان از دست رفتن شیء وجود دارد.

اما در کل به علت یکنواخت بودن دوربین در این کار انجام شده بیشترین دردسر برای تغییر شناسه است زیرا که بازیکنان یکدیگر را می پوشانند و تشخیص انها سخت می شود و بزرگترین مشکل الگوریتم روی تغییر شناسه است.

بخش 7) از الگوریتم DeepSORT برای الگوریتم دیگر استفاده کردیم و نتایج ان را روی ویدئو های مختلف گرفتیم و در خود نوتبوک با الگوریتم قبلی مقایسه کردیم.

تحلیل: در کل Byte Track نتیجه بسیار بهتری در هر 3 تا از معیار ها می دهد روی تمام ویدئو ها اما به دلیل استفاده DeepSORT از ویژگی های ظاهری معیار IDF1 در ابن الگوریتم ها با هم در رقابت است اما 2 معیار دیگر در الگوریتم ByteSORT بسیار بهتر عمل می کند.

(مقایسه نموداری و معیار ها در نوتبوک است و در ویدئو report قرار دارد)

بخش 3)(امتیازی)

بخش3.1)

زیر بخش 1)

مشکل: تغییر شناسه (ID Switch) درByteTrack

راه‌حل پیشنهادی: استفاده از یک مدل Re-Identification (ReID) همراه با تخصیص مبتنی بر یادگیری عمیق

الگوریتم پیشنهادی:
برای کاهش ID Switch درByteTrack، می‌توان یک ماژول ReID مبتنی بر یادگیری عمیق را به فرآیند تطبیق اضافه کرد. در حالت عادی، ByteTrackتنها از IoU**و** Hungarian Matching برای تخصیص استفاده می‌کند که در مواردی که اشیا به‌طور ناگهانی تغییر مکان دهند، دچار ID Switchمی‌شود. راه‌حل ما استفاده از شبکه Siamese با استخراج ویژگی‌های تعبیه‌شده (Embedding Features) برای مقایسه دقیق‌تر بین اشیا در فریم‌های متوالی است.

مراحل الگوریتم پیشنهادی:
۱ .**استخراج ویژگی‌های تعبیه‌شده :(Feature Embedding)** برای هر Bounding Boxکه از تشخیص‌دهنده) مثلاً (YOLO دریافت می‌شود، یک بردار ویژگی استخراج شود. این بردار با استفاده از یک شبکه Siamese**یا** Triplet Network که برای ReIDآموزش دیده، محاسبه می‌شود.
۲ .**ماتریس هزینه ترکیبی :**ماتریس هزینه تخصیص، علاوه بر معیارIoU ، شامل فاصله ویژگی‌های تعبیه‌شده Cosine Similarity) **یا** Euclidean Distance)نیز می‌شود. این ماتریس هزینه به‌جای اینکه فقط براساس IoUباشد، ترکیبی از شباهت مکانی و ویژگی‌های بصری را در نظر می‌گیرد.
۳ .**استفاده از الگوریتم یادگیری تقویتی :(Reinforcement Learning Matching)** به جای استفاده صرف از Hungarian Algorithm، می‌توان از مدل‌های یادگیری تقویتی مانند Deep Q-Network (DQN)برای بهینه‌سازی تخصیص اشیا بر اساس سابقه حرکت و شباهت بصری استفاده کرد.

بخش2)

بهینه‌سازی مدل (Model Optimization) پیش از تحویل آن به عنوان یک محصول، از دو جنبه کلیدی اهمیت دارد. نخست، بسیاری از مدل‌های Deep Learning در ابعاد بزرگ طراحی می‌شوند و اجرا و نگهداری آن‌ها می‌تواند منابع سخت‌افزاری و هزینه‌های بالایی تحمیل کند. دوم، در کاربردهای واقعی به‌ویژه سرویس‌های بلادرنگ، مدل باید پاسخ‌گویی سریع داشته باشد؛ در غیر این صورت، تجربه کاربر یا کارایی سیستم دچار افت جدی می‌شود. بنابراین، با انجام روش‌های Model Optimization می‌توان سرعت اجرا، حجم حافظه مصرفی، و هزینه‌های عملیاتی را تا حد زیادی کاهش داد و مدل را به شکل یک محصول قابل اتکا و کارآمد تحویل داد.

روش‌های مختلفی برای بهینه‌سازی مدل وجود دارد که می‌توان آن‌ها را در چند دسته کلی جای داد:

1. Pruning

در این روش، وزن‌های (Weights) کم‌اهمیت در مدل حذف می‌شوند تا تعداد پارامترها و عملیات محاسباتی کاهش یابد. مزیت اصلی Pruning این است که بدون تغییر معماری کلی مدل، می‌توان آن را سبک‌تر کرد. با این حال، اگر Pruning به‌درستی انجام

نشود یا شدت آن بالا باشد، ممکن است دقت مدل به‌طور محسوس افت کند. انواع مختلفی از Pruning وجود دارد، از جمله Weight Pruning حذف وزن‌های منفرد و Structured Pruning حذف کانال‌ها یا فیلترهای کامل.

2. Quantization

Quantization با کاهش دقت عددی پارامترها) مثلاً از Float32 به (Int8 باعث کوچک‌تر شدن مدل و تسریع محاسبات می‌شود. مزیت اصلی آن اجرای سریع‌تر و صرفه‌جویی در حافظه است. از سوی دیگر، در صورت استفاده از سخت‌افزارهایی که از Int8 پشتیبانی می‌کنند مانند برخی GPU ها یا TPU ها، می‌توان سرعت اجرا را بسیار افزایش داد. عیب اصلی این روش آن است که کاهش دقت محاسباتی ممکن است به افت کارایی مدل به‌ویژه در شبکه‌هایی که حساسیت بالایی به کمیت‌های عددی دارند، منجر شود.

3. Knowledge Distillation

در این رویکرد، یک مدل بزرگ به‌عنوان Teacher عمل می‌کند و یک مدل کوچک‌تر (Student) از خروجی‌های میانی یا نهایی Teacher یاد می‌گیرد. مزیت Knowledge Distillation آن است که مدل Student می‌تواند به کارایی نزدیک به Teacher برسد اما با تعداد پارامتر کمتر، زیرا بخشی از دانش مدل بزرگ را در خود ذخیره می‌کند. مشکل اصلی در این روش، نیاز به دو مرحله مجزا برای آموزش است) یک‌بار آموزش مدل Teacher و سپس آموزش مدل.(Student

4. Neural Architecture Search

هدف Neural Architecture Search یا به اختصار NAS ، یافتن معماری مناسب برای یک مدل است که در آن معیارهایی مانند سرعت اجرا، اندازه مدل و دقت در نظر گرفته می‌شوند. مزیت اصلی NAS این است که ساختار مدل می‌تواند به صورت خودکار و مبتنی بر داده و محدودیت‌های سخت‌افزاری انتخاب شود. اما نقطه‌ضعف این رویکرد، هزینه محاسباتی بالا در مرحله جستجوی معماری است که نیازمند منابع سخت‌افزاری گسترده و زمان قابل توجهی است.

5. Tensor Decomposition

در این روش، وزن‌های مدل) به‌ویژه در لایه‌های Convolutional یا (Dense به صورت فاکتورهای کم‌رتبه (Low-Rank) تجزیه می‌شوند تا تعداد کل عملیات کاهش پیدا کند. مزیت آن، امکان حفظ ساختار اصلی مدل و دستیابی به سرعت بالاتر است. اما اگر درجه کاهش رتبه خیلی زیاد باشد، خطای تقریب نیز بالا می‌رود و مدل ممکن است افت دقت چشمگیری نشان دهد.

6. Layer Fusion و Operator Fusion

در این روش‌ها، تعدادی از عملیات یا لایه‌های پشت سر هم در شبکه) مثلاً Convolution و (Batch Normalization با هم ادغام می‌شوند تا از بار اضافی ناشی از فراخوانی چندباره آن‌ها در زمان اجرا جلوگیری شود. مزیت اصلی آن، ساده‌شدن نمودار محاسباتی و بهبود زمان اجرا است. محدودیت آن نیز در شرایطی است که برخی لایه‌ها به دلایل عددی یا ساختاری قابل ادغام با یکدیگر نباشند.