گزارش پروژه Object Detection and Tracking

آریان فتحی

پروژه مورد نظر که از آن تحت عنوان MOT (Multi Object Tracking) یاد میشود یک تسک بسیار کاربردی و بعضا چالشی در بینایی کامپیوتر یاد میشود.

این پروژه از دو بخش Detection و Tracking تشکیل شده است که در هربخش چالش هایی وجود دارد که در نتیجه پروژه اثرگذار است.

وظیفه تشخیص آبجکت همانطور که از نامش پیداست به فرایند تشخیص و استخراج آجبکت های موجود و هدف در فریم اطلاق میشود که برای اینکار چندین مدل تا کنون معرفی شده است که معروف ترین آنها YOLO که برای اولین بار تحت مقاله ای با همین نام (You Look Only Once) معرفی شد یک شبکه عصبی با ساختار CNN است که یکی از ماژول های تشکیل دهنده آن Google net میباشد و تا کنون ورژن های مختلفی از آن معرفی و پیاده سازی شده که درهریک تفاوت های برجسته ای دیده میشود، یکی از پرکاربرد ترین و پراستفاده ترین این مدل ها Yolo v8 است که در کتابخانه ultralytics پیاده سازی شده است و قابل استفاده میباشد.

این مدل چندین پارامتر را به عنوان ورودی خود دریافت میکند که در عملکرد و تشخیص ابجکت ها تاثیر گذار است که در ادامه اشاره خواهد شد.

بخش دوم پروژه بررسی و کار کردن با tracking algorithm ها بود که دو مورد از آنها توسط مدل پشتیبانی میشوند، botsort که الگوریتم دیفالت مدل میباشد و bytetrack که الگوریتمی پرطرفدار و پرکارکرد میباشد.

این الگوریتم ها از چندین ابزار و فیلتر برای پیشبینی کردن مسیر ابجکت هایی که از فریم های قبلی درحال ردیابی هستند استفاده میکند که مهمترین آنها Kalman filter است که در سایتkalman.io میتوانید توتوریالی کامل از آن را مشاهده کنید، این فیلتر که در پیاده سازی این الگوریتم ها با فرض ثابت بودن سرعت تغییرات مکان ابجکت استفاده شده است، احتمال حضور ابجکت در فریم های بعدی در مختصات های مختلف فریم را پیشبینی میکند. همچنین بسته به نوع الگوریتم بکار رفته از شبکه عصبی برای feature extraction و ReID کردن ابجکت ها استفاده میشود. یا مثلا در الگوریتم بات سورت، از OptFlow که تغییرات نقاط موجود در صفحه را نسبت به فریم قبلی بررسی میکند نیز به کار رفته است، استفاده از تمام این ماژول ها کنار هم باعث میشود که پیشبینی و تشخیص حرکت مدل ها پیش از

پیش دقیق تر شود ولی باز هم بسته به تسک مورد نظر نیاز به تغییر و دستکاری پیاده سازی های موجود در کتابخانه ها داریم.

هدف از این پروژه هندل کردن محیط های به شدت پویا و occluded میباشد که بهترین تست برای این تسک ویدیو های بازی المپیک است.

در ادامه هایپرپارامتر هایی که برای tune شدن درنظر گرفته شده اند نیز اشاره میشود.

-version of yolo:

یکی از ویژگی های اثرگذار برای تشخیص ابجکت ها استفاده از ورژن مدل است، هرچقدر سایز مدل بزرگتر باشد تعداد لایه ها و پارامتر ها بیشتر بوده و دقت مدل افزایش میابد ولی سرعت پردازش نیز تحت تاثیر قرار میگیرد.

من برای دقت بیشتر از Yolov8x استفاده کردم که بزرگترین مدل موجود بود ولی نیاز به استفاده ازین مدل نبود.

-imgsz:

سایز فریم ورودی نیز یکی از تاثیرگذار ترین عوامل موجود برای تشخیص ابجکت های بی کیفیت و ریز در تسک است که سایز دیفالت آن ۶۴۰ بود و من از مقدار دو برابر آن یعنی ۱۲۸۰ استفاده کردم.

-Iou:

یکی از معیار هایی است که میزان انطباق یک باکس پیشبینی شده توسط مدل با ground truth label را به صورت تقسیم میزان اشتراک بر میزان اجتماع آنها محاسبه میکند را درنظر میگیرد، هرچه این مقدار بیشتر باشد، باکس ها دقیق تر شده و مرز های بهتری را برای هر ابجکت درنظر میگیرند ولی دقت کنید که ماکسیمم این مقدار باعث چندشماری یک ابجکت میشود. من در این پروژه از مقدار دیفالت آن یعنی ۰.۷ استفاده کردم.

در نهایت میرسیم به هایپرپارامتر های مورد بررسی در botsort، ازآنجایی که با botsort عملکرد بهتری دریافت کردم و هایپرپارامتر های این الگوریتم با bytetrackمشابه است از دوباره گویی پرهیز میکنم.

از مهم ترین پارامتر هایی که در پایستگی مسیر ها و از گم شدن ابجکت ها جلوگیری میکند میتوان به موارد زیر اشاره کرد:

- Track_high_thresh: لازم به ذکر است که دو مرز برای تشخیص ابجکت ها و دنبال کردن آنها درنظر گرفته شده است، بدین صورت که ابجکت هایی که دارای معیار confidence بالاتری از این مقدار هستند به عنوان ابجکت های دقیق تری درنظر گرفته شده و در پایپلاین الگوریتم فرایند متفاوتی تحت عنوانfirst association اجرا میشود، بدین صورت که جدا از معیار iou برای تطبیق دادن ابجکت های تشخیص داده شده و مسیر های موجود از فیچر های ReID نیز استفاده میشود. مقداری که برای هندل کردن ویدیوی بازی والیبال درنظر گرفتم، ۰.۳ است.
- Track_low_thresh اما زمانی که این امتیاز از مرز قبلی کمتر باشد ولی ازین معیار بیشتر باشد تحت عنوان second association بررسی میشود، در اصل تفاوت این دو الگوریتم نسبت به باقی الگوریتم ها در همین است که حتی به ابجکت هایی که زیاد دقیق نیستند ولی پیش زمینه هم محسوب نمیشوند نیز شانس دوباره داده میشود. این ابجکت ها فقط با iou بررسی میشوند. مقداری که برای تسک درنظر گرفتم ۰.۱ بود.
- New_track_thresh: این معیار نشان میدهد که اگر ابجکتی تشخیص داده شد ولی به مسیری نسبت داده نشده است ولی امتیاز ان ازین معیار بالاتر است، مسیری جدی برای آن درنظر گرفته میشود، کم کردن این معیار باعث میشود ابجکت های خیلی زیادی درتصویر درنظر گرفته شوند ولی شانس id switch را زیاد میکند که چندان دلچسب نیست، زیاد کردن این معیار باعث میشود مدل بیشتر روی ReID کردن تمرکز کند. مقدار مورد استفاده برای بهترین معیار: ۰.۷۸ بود.
- Match_thresh: این معیار میزان حساسیت برای تطابق دادن ابجکت های تشخیص داده شده در یک فریم با مسیر های موجود را نشان میدهد، هرچه این معیار پایین تر باشد ممکن است مدل ابجکت هایی که ذره حرکت های مشابه یا تطابق های مشابه تری داشته باشند را باهم اشتباه بگیرد و درستی مدل کمتر میشود درحالی که زیاد کردن این معیار تنها مسیر های بسیار مطمئن را لحاظ میکند. مقدار درنظر گرفته شده: ۰.۸۵

معیار های دیگری نیز در فایل های yaml این الگوریتم های موجود است ولی تاثیر زیادی در تسک من نداشتند، به طور کلی مشکل اصلی من ReID کردن ابجکت هایی بود که حرکت شتابدار زیادی داشتند و در چندین ثانیه از ویدیو خارج شده و دوباره با وضوح و زاویه متفاوتی وارد میشدند، مدل امکان پیشبینی همچین تغییرات زیادی را برای ابجکت ها درنظر نگرفته است. چالش های اصلی عمدتا حرکت های مشابه، ظاهر های مشابه، پوشیده شدن توسط موانع و خارج شدن از صحنه برای مدت طولانی است اما درنهایت توانستم به همه ی این مشکلات فائق آیم جز ReID کردن ابجکت هایی که برای مدت طولانی غایب بودند.

در نهایت نیز لینک پیاده سازی پروژه قرار گرفته است و ویدیو های بررسی و نتیجه نیز در زیپ قرار دارند.

https://colab.research.google.com/drive/1BX5jCxyyExlduJN_dlruMJ3Yblej78H?usp=sharing

منابع:

https://www.v7labs.com/blog/yolo-object-detection
https://arxiv.org/abs/1506.02640
https://www.youtube.com/watch?v=uMzOcCNKr5A
https://www.mdpi.com/2504-4990/5/4/83
https://medium.com/softplus-publication/video-object-tracking-with-yolov8-and-sort-library-e28444b189aa
https://www.datature.io/blog/introduction-to-bytetrack-multi-object-tracking-by-associating-every-detection-box
https://www.datature.io/blog/get-started-with-training-a-yolov8-object-detection-model
https://www.kalmanfilter.net/kalman1d.html