

به نام خدا



دانشگاه صنعتی شریف

دانشکده مهندسی برق، آزمایشگاه یادگیری عمیق

گزارش پروژه کارشناسی

موضوع

تسریع مدل آشکارساز در خلاصه سازی معنایی ویدیو

دانشجو

سجاد هاشم بیکی

استاد راهنمای پروژه

دکتر هدی محمدزاده

استاد درس پروژه

دکتر مهدی فردمنش

تیرماه 1403

تسريع مدل آشکارساز در خلاصه سازی معنایی ویدیو

چکیده

در این پروژه قصد داریم ابتدا به معرفی مفهوم خلاصه سازی ویدیو بپردازیم و مازول های مورد استفاده در آن، ردیاب و آشکارساز، را بررسی کنیم.

سپس به سراغ هدف اصلی پروژه که بهبود سرعت شبکه آشکارساز است میرویم و ایده های پیاده شده برای این امر را معرفی میکنیم.

واژه های کلیدی

آشکارسازی اشیاء، ردیابی اشیاء، بینایی ماشین، یادگیری عمیق، شبکه های کانوولوشنی

فهرست مطالب

4	مقدمه
5	خلاصه سازی معنایی ویدئو
7	تاریخچه خلاصه سازی معنایی ویدئو
8	آشکارسازی اشیا (Object Detection)
8	شبکه YOLO
10	ردیابی اشیا (Tracking)
11	بررسی معماری شبکه YOLOv5
16	بررسی ایده های تسریع مدل آشکارساز
17	بررسی و پیاده سازی ایده کانوولوشن در عمق (Depth-wise Convolution)
26	نتایج نهایی روش Depth Wised Convolution
30	جمع بندی
31	منابع و مراجع

مقدمه

امروزه با گسترش مدل های مبتنی بر یادگیری عمیق، میتوان کاربرد آنها را در جای جای زندگی روزمره مشاهده نمود. یکی از مباحث مهم در این حوزه، استفاده از شبکه های عمیق برای تجزیه و تحلیل محتوای تصویری میباشد.

کاربردهایی از جمله طبق بندی، مکان یابی، تشخیص ویژگی ها و مشخصات و... میباشد. استفاده از این مدل ها در عمل، همواره با چالش هایی مواجه بوده و هست. چالش هایی مانند کمبود داده برچسب دار برای آموزش و ارزیابی مدل ها، کمبود توان پردازشی برای آموزش شبکه های عمیق و همینطور استفاده از مدل ها به صورت آنی (Real time).

هدف اصلی ای که در این پروژه دنبال میشود بهبود مدل های آشکارساز موجود است، تا با کاهش حجم محاسبات و پارامترهای شبکه ها، بتوان در مرحله پیش بینی (Inference) به سرعت بالاتری دست یافت و در عین حال کمترین کاهش دقت و عملکرد را داشته باشیم.

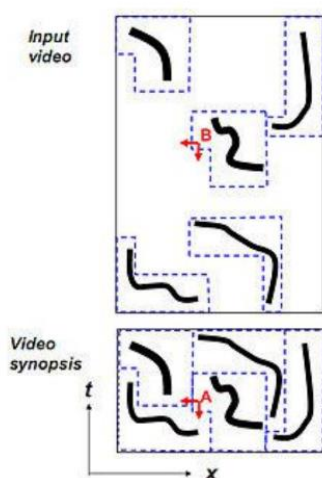
خلاصه سازی معنایی ویدئو

خلاصه سازی معنایی ویدئو به این مفهوم است که رویدادها را به صورت همزمان در فریم ها نشان دهیم و از لحاظ زمانی و مکانی از نهایت ظرفیت هر فریم استفاده کنیم، که این امر امکان مرور ساعتها ویدئو را در عرض چند دقیقه فراهم میکند.

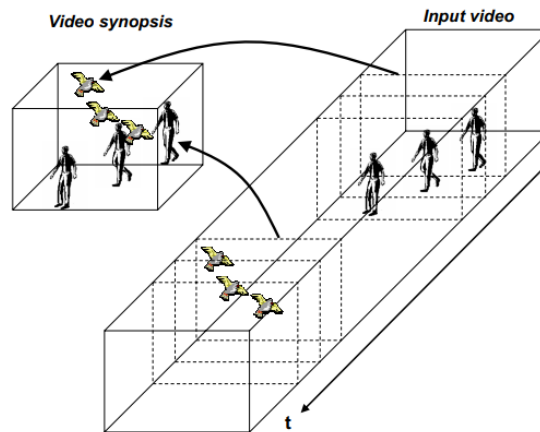
در این روش اشیاء در حال حرکت را شناسایی، ردیابی، و تجزیه و تحلیل می کنند. باوجود پیشرفت های تکنولوژی و افزایش رشد در استقرار دوربین های مدار بسته، مشاهده، تجزیه، و تحلیل فیلم های ضبط شده هنوز هم هزینه بر است .

خلاصه سازی ویدئو کاربردهای گوناگونی در تجزیه و تحلیل هوشمند ویدئوهای دوربینهای نظارتی و ترافیکی دارد. در این روش تصاویر چندین ساعت به چند دقیقه کاهش پیدا کرده و بر روی تمامی المانهای متحرک زمان تردد ثبت میگردد.

بطور خلاصه تمام اشیاء موجود در فریم ها جمع آوری شده و به یک محدوده زمانی کوتاه تر منتقل میشوند تا یک فیلم کوتاه تر و با حداکثر فعالیت در هر فریم بدست آید که باعث استفاده بهینه از فضای پیکسلی و همینطور خلاصه سازی در بعد زمان میشود.



شکل 1 - خلاصه سازی رویدادها و غنی کردن ابعاد فضایی-زمانی



شکل 2 - خلاصه سازی رویدادها و غنی کردن ابعاد فضایی-زمانی

که در نهایت یک فیلم خلاصه تولید می شود که در آن اشیا و فعالیت هایی که در زمان های مختلف اتفاق افتاده اند به طور همزمان نمایش داده می شوند. همانطور که اشاره شد دو تکنیک مهم در خلاصه سازی بکار میرود، که عبارتند از آشکارسازی اشیا و ردیابی اشیا. در ادامه به بررسی این دو مورد میپردازیم.





در شکل های بالا مفاهیم حذف فضاهای خالی در دو بعد فضایی و زمانی را میتوان مشاهده نمود که این ایده اصلی خلاصه سازی معنایی ویدیو میباشد.

تاریخچه خلاصه سازی معنایی ویدئو

فناوری خلاصه سازی ویدئو توسط پروفسور Shmuel Peleg از دانشگاه عبری اورشالیم در اسرائیل اختراع شد. شرکت BriefCam مجوز استفاده از این فناوری را از Yisum که مالک این ثبت اختراع است را دریافت کرد. در ماه می سال 2018 شرکت BriefCam توسط غول تصویربرداری دیجیتال ژاپنی، Canon Inc، به قیمت تقریبی 90 میلیون دلار خریداری شد.

این شرکت توانسته بازار انحصاری بسیار خوبی از طریق این ایده و توسعه آن فراهم کند. به دلیل پتنت ثبت شده شرکت دیگری حق استفاده از این تکنولوژی را ندارد، به همین دلیل شرکت های محدودی در سطح دنیا به استفاد از این فناوری به صورت رسمی پرداخته اند.

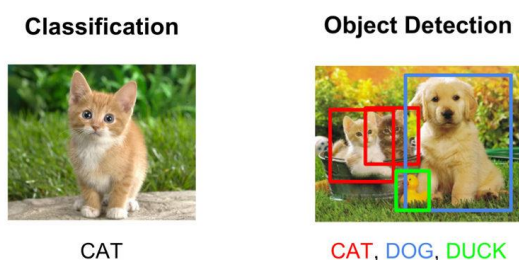
لازم بذکر است در چین و هند تلاش شده تا فناوری های مشابهی ارائه شود. در شکل زیر خلاصه ای از اطلاعات شرکت BriefCam ارائه شده است.

<div>Estimated Annual Revenue \$9M</div> <div>Agree? <input type="button" value="Yes"/> <input type="button" value="No"/></div>	<div><div>President & CEO Trevor Matz</div></div> <div>CEO Approval Rating 68/100</div> <div>Weigh In</div>	<div>OVERVIEW</div> <div><div>Founded:</div>2008</div> <div><div>Headquarters:</div>Newton, Massachusetts</div> <div><div>Status:</div>Private, Subsidiary of Canon, Inc.</div> <div><div>Industry Sector:</div>Internet Software</div> <div><div>SIC Code:</div>7372 NAICS listing »</div> <div><div>Links:</div><div>  </div></div>
---	--	--

شکل 3 - خلاصه ای از اطلاعات شرکت BriefCam

آشکارسازی اشیا (Object Detection)

منظور از آشکار سازی اشیا، پیدا کردن مکان اشیا با استفاده از چهار عدد (به طور مثال میتواند شامل مختصات مرکز کادر و اندازه طول و عرض کادر باشد) در تصویر و همین طور طبقه بندی آنها میباشد؛ در طبقه بندی صرفاً برچسب اشیا موجود در تصویر را پیش بینی میکنیم. در تصویر زیر تفاوت طبقه بندی و آشکار سازی مشهود است.

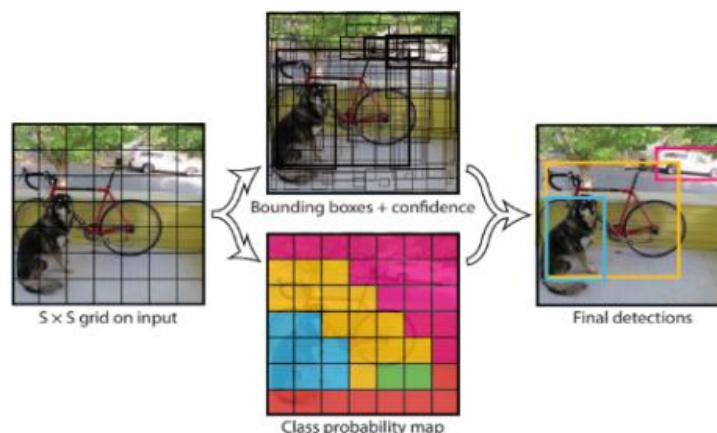


شکل 4 - تفاوت میان آشکار سازی و طبقه بندی اشیا

شبکه YOLO

در این پروژه برای آشکار سازی اشیا، از مدل های خانواده YOLO استفاده میکنیم. دلیل استفاده از این مدل ها سرعت بسیار بالای آنها در آشکار سازی اشیا نسبت به همتایان خود میباشد، که این امکان را به ما میدهند که بصورت آنی (Real Time) اشیا موجود در تصاویر و ویدیوها را آشکار سازی کنیم.

در ادامه به توضیح مختصری از نحوه عملکرد این شبکه میپردازیم.

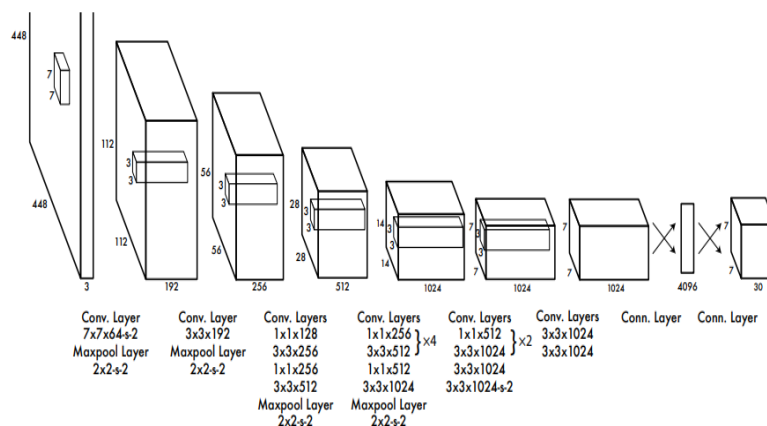


شکل 5- نحوه کلی شبکه بندی و تعداد خروجی های مدل به ازای هر تصویر

همانطور که از اسم این مدل پیداست، بارزترین ویژگی این شبکه نسبت به همتایان خود این است که تنها یکبار تصویر ورودی را پردازش میکند و تمام مقادیر لازم از جمله مختصات کادرها (آفست ها)، و احتمال کلاس ها را محاسبه میکند و همین امر موجب سرعت بسیار زیاد این مدل نسبت به مدلای مشابه شده است.

این مدل ابتدا تصویر ورودی را به یک شبکه $S \times S$ تقسیم می کند. اگر مرکز یک شی در یک سلول بیافتد، آن سلول مسئول تشخیص آن شی است. هر سلول B کادر را در نظر میگیرد. اگر هیچ جسمی در آن سلول وجود نداشته باشد، مقدار آن صفر میشود. در غیر این صورت امتیاز آن برابر با میزان IOU بین کادر پیش بینی شده و کادر واقعی میشود.

شبکه برای هر کادر چهار مقدار x, y, w, h و یک مقدار اطمینان پیش بینی می کند. مختصات (x, y) مرکز کادر را نسبت به مرزهای سلول ها نشان می دهد. عرض و ارتفاع نسبت به کل تصویر پیش بینی می شود. همچنین هر سلول C مقدار برای احتمال برچسب های کلاس ها پیش بینی می کند.

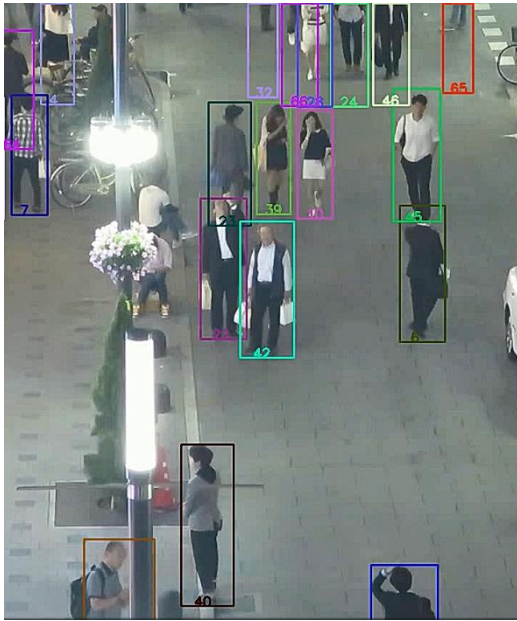


شکل 6 - ساختار لایه های کانولوشنی مدل YOLO

ردیابی اشیا (Tracking)

منظور از ردیابی پیدا کردن کادرهای مربوط به هر شی و تخصیص دادن یک شماره به آن شی در طول ویدیو میباشد. بطور خلاصه، پس از پیدا کردن کادرهای اشیا در فریمهای ویدیو با استفاده از مدل های آشکارساز، الگوریتم ردیابی ما به این صورت عمل میکند که با مشاهده دو فریم متوالی و کادر های متناظر با آن فریمها، باتوجه به موقعیت مکانی کادرها و فاصله آنها در دو فریم، شماره ای که کمترین خطا را ایجاد میکند را به کادر مربوطه تخصیص میدهد و این کار را برای همه فریم های متوالی انجام میدهد.

در نهایت کادرهایی داریم که در فریم های مختلف شماره هایی به آنها تخصیص داده شده است و انتظار داریم کادرهای مربوط به یک شی در طول ویدئو، یک شماره را گرفته باشند. برای درک بهتر این موضوع در ادامه دو فریم با فاصله زمانی چندثانیه ای قرار داده شده است که کادرها و شماره ی متناظر با آنها مشهود است.



شکل 7 - فریم ابتدایی



شکل 8 - فریم انتهایی

بررسی معماری شبکه YOLOv5

این شبکه از سه بخش اصلی تشکیل شده است. که در ادامه به بررسی بخش های مختلف آن میپردازیم. همانطور که در تصویر پایین مشاهده میکنید برای استفاده از ویژگی ها در سطوح مختلف ارتباطی جانبی میان سه بخش اصلی وجود دارد که موجب میشود در نهایت ویژگی هایی از تمام سطوح برای آشکارسازی نهایی داشته باشیم.

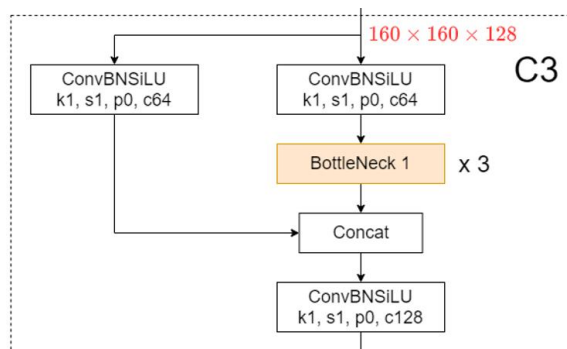
سه بخش اصلی این شبکه عبارتند از: بدنه (Backbone)، میانه (Neck) و سر (Head)

در ادامه به بررسی هر بخش به صورت مجزا میپردازیم.

- بدنه (Backbone)

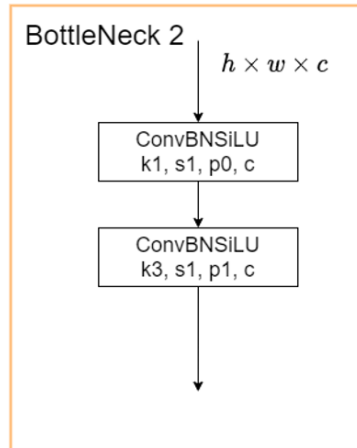
این قسمت وظیفه دریافت تصویر و اعمال بلوک های کانوولوشنی روی تصویر ورودی است. بدنه استفاده شده در این شبکه CSP-Darknet53 نام دارد. اجزای اصلی سازنده این بدنه، بلوک هایی به نام C3 هستند که هر کدام شامل چند بلوک کانوولوشنی میباشند.

هر بلوک کانوولوشنی شامل یک فیلتر کانوولوشنی، یک لایه نرمالیزشن و یک تابع فعال سازی میباشد که به ترتیب بر روی ویژگی ها اعمال میشوند. در شکل زیر میتوان نمایی از این ماژول را مشاهده کرد.

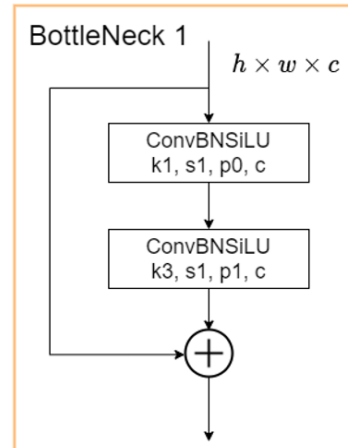


شکل 9 - نمایی از ماژول C3

یکی دیگر از اجزای مهم ماژول C3، بلوک هایی به نام Bottleneck هستند که در میان دو بلوک کانوولوشنی قرار میگیرند. در این شبکه دو نوع از بلوک های Bottleneck به کار رفته است. تفاوت میان این دو بلوک در وجود مسیر مستقیم و بدون وزن است که باعث جلوگیری از کاهش اندازه گرادینان (Gradient Vanishing) خواهد شد. در ادامه نمایی از دو بلوک Bottleneck را میتوان مشاهده نمود.



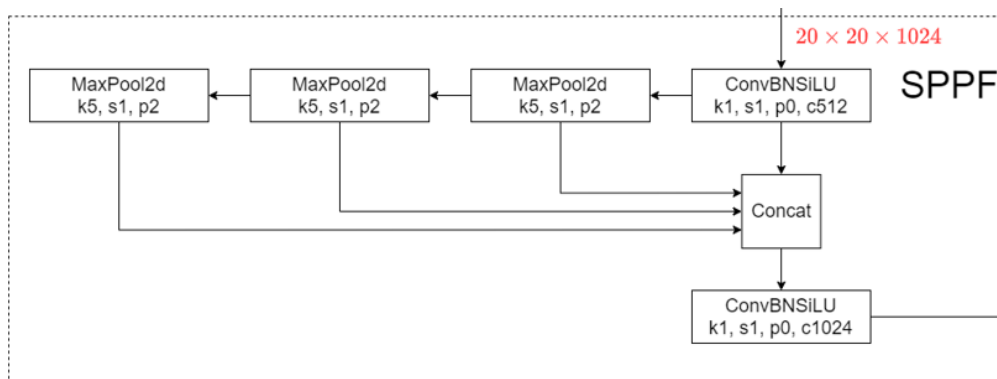
شکل 11 - نمایی از ماژول Bottleneck2



شکل 10 - نمایی از ماژول Bottleneck1

- گردن (Neck)

در این بخش از شبکه، هدف اتصال دو قسمت بدنه و سر به یکدیگر است. برای این منظور از بلوکی به نام SPPF استفاده شده است. ویژگی اصلی این بلوک استفاده از عملگرهای بیشینه گیری (MaxPooling) با ابعاد یکسان و به صورت متوالی است. در تصویر زیر نمایی از این بلوک را مشاهده میکنید.

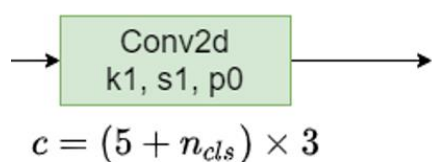


شکل 12 - نمایی از ماژول SPPF

- سر (Head)

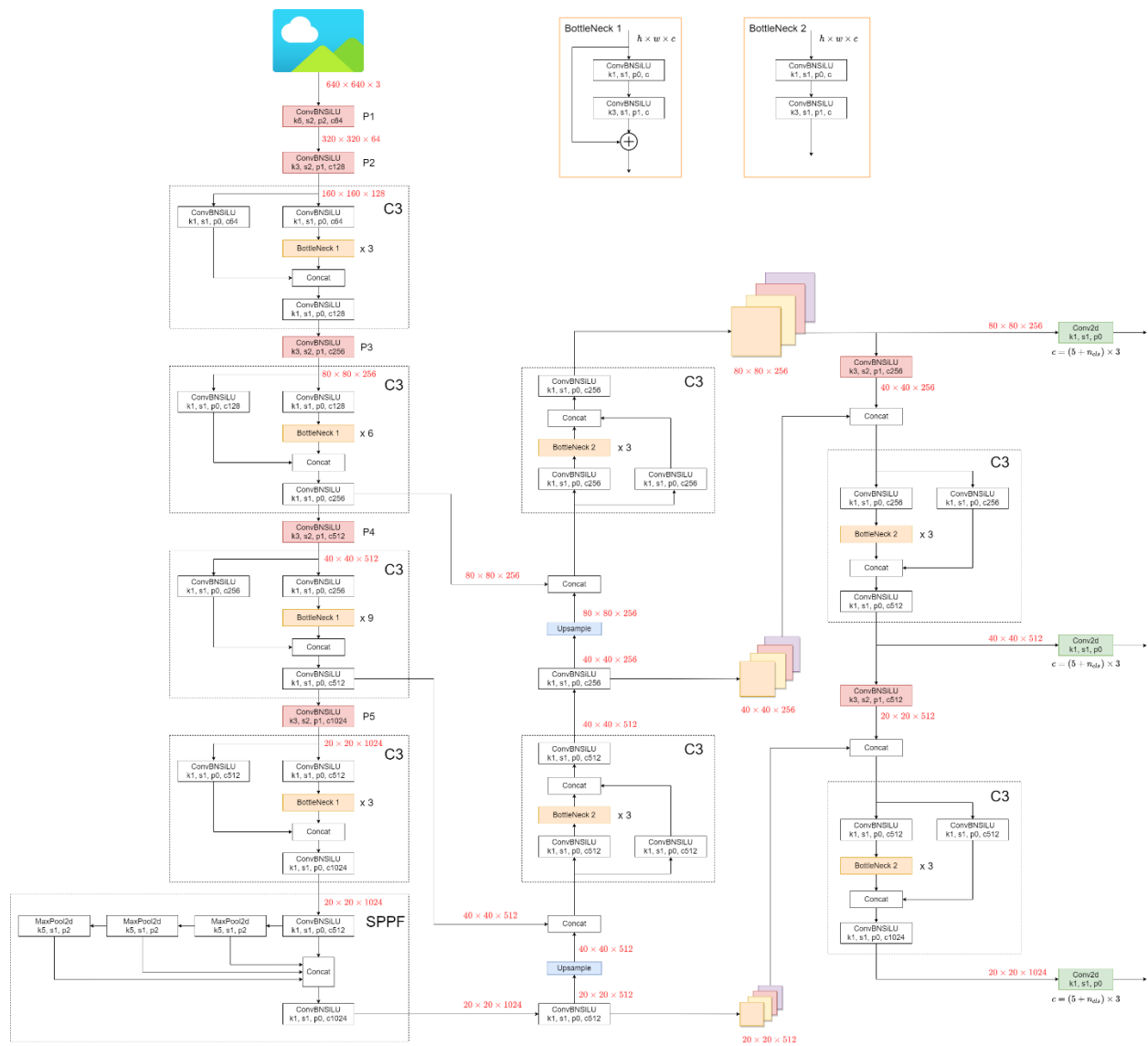
قسمت نهایی این شبکه Head نام دارد که وظیفه آن دریافت ویژگی های استخراج شده از بخش های قبل است و در نهایت برای هر کادر، پنج مقدار شامل مختصات طول و عرض مرکز، اندازه طول و عرض و همین طور احتمال حضور شی را پیش بینی میکند.

علاوه بر این مقادیر، میزان احتمال تعلق به هر کلاس را نیز محاسبه میکند. در تصویر زیر نمایی از آخرین بخش شبکه را مشاهده میکنید.



شکل 13 - نمایی از بلوک نهایی

و در نهایت با اتصال این سه بخش به شبکه نهایی می‌رسیم که در ادامه شبکه کامل را مشاهده می‌کنید.



شکل 14 - نمایی از شبکه YOLOv5

بررسی ایده های تسريع مدل آشکارساز

برای تسريع مدل، ابتدا تمام مدل های خانواده YOLO از نسخه اوليه تا پنجم بطور کامل بررسی شده و در ادامه ایده های مناسب را پیاده کرده و نتایج را نمایش داده ایم. پس از شناخت کامل معماری YOLOv5 دو ایده منتخب که از مقالات برگرفته شده اند بررسی و پیاده شده است.

هدف کلی در انتخاب و پیاده سازی ایده ها، کاهش میزان پارامترها و محاسبات شبکه و در نهایت افزایش سرعت پیش بینی میباشد و در عین حال بایستی کمترین کاهش دقت را داشته باشیم.

در تمام بخش های این پروژه دیتاست مورد استفاده MS COCO میباشد که صرفا از کلاس انسان استفاده کرده ایم. تعداد داده های مورد استفاده عبارتند از: 64000 داده آموزشی و 2600 داده برای ارزیابی کردن مدل.

لازم بذکر است برای تمامی مدل ها تعداد ایپاک های آموزش یکسان و برابر با ده در نظر گرفته شده است و ساینز دسته ها (Batch Size) برابر با 32 میباشد.

برای مقایسه عادلانه میان مدلها همگی تحت شرایط یکسان آموزش دیده اند و سرعت آنها اندازه گیری شده است. لازم بذکر است سخت افزار مورد استفاده برای آموزش و ارزیابی مدل ها، گرافیک Tesla P100 میباشد که به صورت رایگان از طریق پلتفرم کگل قابل دسترسی میباشد.

دو ایده مطرح شده هر کدام تعداد پارامترها و محاسبات را کاهش داده اند. اولین ایده برگرفته از مقاله شبکه MobileNet میباشد که در سریعترین مدل به دست آمده، به میزان 28 درصد از پارامترهای مدل و همچنین 24.4 درصد از حجم محاسبات را کاهش داده است. در نهایت این ایده توانسته است به میزان 16.7 درصد به سرعت مدل اضافه کند.

دومین ایده پیاده شده که باعث کاهش تعداد پارامترها در بلوک های C3 شده است، به میزان 12.5 درصد از پارامترها و 12.1 درصد از محاسبات مدل را کاهش داده است اما سرعت نهایی تفاوت چندانی با مدل اصلی ندارد و با وجود کاهش تعداد پارامترهای مدل، زمان پیش بینی شبکه برای هر تصویر تغییری نکرده است. در ادامه به صورت مجزا تمام ایده ها به طور کامل بررسی و نتایج آن گزارش شده است.

بررسی و پیاده سازی ایده کانولوشن در عمق (Depth-wise Convolution)

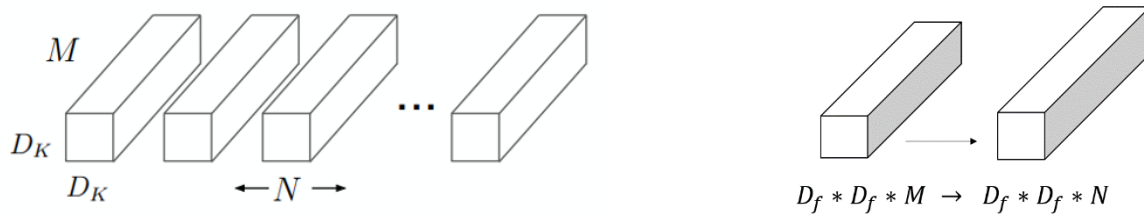
این ایده برای اولین بار در سال 2017 در مقاله شبکه MobileNet مطرح شد. پایه و اساس این ایده استفاده از عملگر کانولوشن استاندارد در دو مرحله است که این امر باعث کاهش تعداد پارامترهای بلوک های کانولوشنی خواهد شد.

برای درک بهتر تفاوت میان دو عملگر کانولوشن استاندارد و در عمق مثال زیر را بررسی میکنیم. ابتدا فرض میکنیم یک دسته ویژگی (FeatureMap) به ابعاد $D_f * D_f * M$ در دسترس داریم و میخواهیم یک فیلتر کانولوشنی استاندارد به ابعاد $D_k * D_k * M$ و به تعداد N بر روی آن اعمال کنیم.

با توجه به اینکه ابعاد طول و عرض دسته ویژگی را ثابت در نظر گرفته ایم و کانولوشن اعمالی از نوع همانی (Same) میباشد، واضح است که ابعاد دسته ویژگی خروجی $D_f * D_f * N$ خواهد بود.

با یک محاسبه سرانگشتی میتوان یافت که تعداد عملیات لازم یا در واقع هزینه محاسباتی (Computational Cost) این فیلتر برابر با $D_k * D_k * M * N * D_f * D_f$ خواهد بود.

در تصاویر زیر شماتیکی از نحوه عملکرد عملگر کانولوشن استاندارد را مشاهده میکنید. عمق هر فیلتر برابر با عمق دسته ویژگی ورودی و تعداد فیلترها برابر با عمق دسته ویژگی خروجی میباشد.



شکل 16 - فیلتر کانولوشن استاندارد

شکل 15 - نمایی از دسته ویژگی های ورودی و خروجی از فیلتر کانولوشن استاندارد

حال همان دسته ویژگی با ابعاد $D_f * D_f * M$ را در نظر بگیرد. اگر بخواهیم از این دسته ویژگی ورودی، به دسته ویژگی خروجی به ابعاد $D_f * D_f * N$ برسیم، همانند کانولوشن استاندارد، و در عین حال از عملگر کانولوشن در عمق استفاده کنیم، ابتدا M فیلتر مجزا با ابعاد $D_K * D_K$ و با عمق یک را بر روی دسته ویژگی ها اعمال میکنیم.

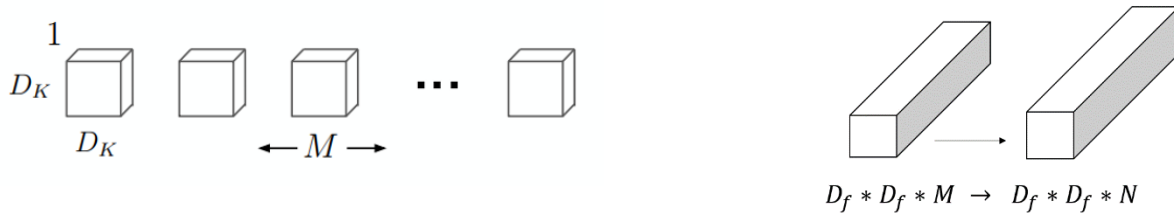
سپس در مرحله دوم، دسته ویژگی های مرحله قبل که ابعاد $D_f * D_f * M$ دارند را با استفاده از فیلترهای یک در یک، $1 * 1 * M$ ، با عمق M کانولوشن انجام میدهیم. بدیهی است که تعداد این فیلتر های یک در یک برابر با N باید باشد تا در نهایت به دسته ویژگی با ابعاد $D_f * D_f * N$ برسیم.

برای محاسبه هزینه محاسباتی این عملگر همانند قبل عمل میکنیم. در مرحله اول که به میزان M فیلتر مجزا با ابعاد $D_K * D_K$ و با عمق یک داریم، میزان محاسبات، تعداد ضرب های انجام شده، برابر با

$D_f * D_f * M * D_K * D_K$ میباشد. در مرحله دوم که به تعداد N فیلتر یک در یک و با عمق M داریم، میزان محاسبات برابر با $D_f * D_f * M * N$ خواهد بود.

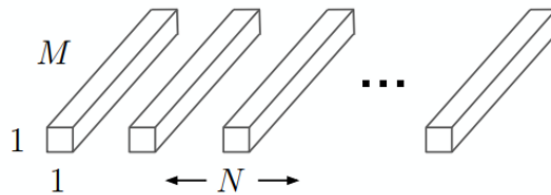
در نهایت هزینه محاسباتی برای این عملگر برابر با مجموع محاسبات دو مرحله یعنی $M * N * D_f * D_f + D_K * D_K * M * D_f * D_f$ خواهد بود.

برای درک بهتر شمایی کلی از این عملگر در ادامه قابل مشاهده است.



شکل 17 - نمایی از دسته ویژگی های ورودی و خروجی از فیلتر کانوولوشن در عمق

شکل 18 - اولین مرحله از فیلتر کانوولوشن در عمق



شکل 19 - دومین مرحله از فیلتر کانوولوشن در عمق

همانطور که ملاحظه شد، میتوان یافت که در حالت دوم، کانوولوشن در عمق، میزان محاسبات فیلتر کردن به میزان $\frac{1}{N} + \frac{1}{D_k^2}$ کاهش یافته است. پارامتر N همان تعداد فیلترها یا تعداد لایه های دسته ویژگی خروجی میباشد و پارامتر D_k ابعاد فیلتر است که در این شبکه برابر با 3 میباشد.

$$\frac{M * N * D_f * D_f + D_k * D_k * M * D_f * D_f}{D_k * D_k * M * N * D_f * D_f} = \frac{1}{N} + \frac{1}{D_k^2}$$

معادله 1 – مقایسه هزینه محاسباتی دو فیلتر استاندارد و در عمق

حال پس از آشنایی کامل با عملگر کانوولوشن در عمق و مقایسه آن با عملگر کانوولوشن عادی به سراغ پیاده سازی آن بر روی شبکه YOLOv5 میرویم. لازم بذکر است شبکه مورد استفاده در این پروژه از نسخه Nano می باشد که سبک ترین و سریع ترین نسخه در میان نسخه های دیگر خانواده YOLOv5 می باشد.

نکته هائز اهمیت آن است که ایده کانوولوشن در عمق را برای هر شبکه ای که در آن از فیلترهای کانوولوشنی استفاده شده است را میتوان به کار برد و تعداد پارامترهای آن شبکه را کاهش داد و این ایده صرفاً مختص خانواده YOLO نمی باشد.

همانطور که در بخش آشنایی با معماری شبکه YOLOv5 مطرح شد، این شبکه دارای سه بخش اصلی است. در هر بخش بلوک هایی قرار دارند که میتوان در آنها فیلترهای کانوولوشن استاندارد را با کانوولوشن در عمق جایگزین کرد. با اینکار چند مدل سبکتر که پارامترهای کمتری دارند به دست آمده است.

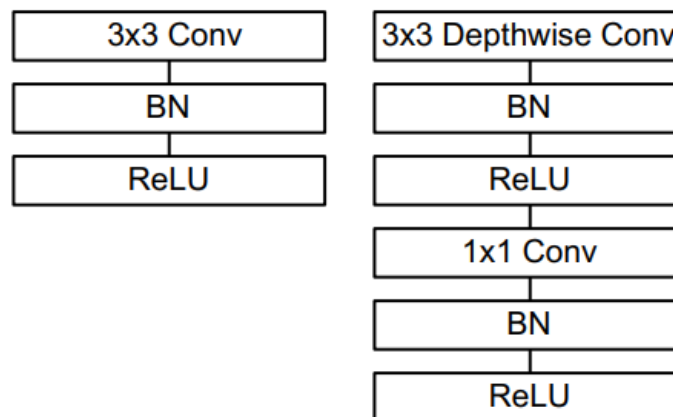
در اولین مدل صرفاً قسمت بدنه (Backbone) را تغییر داده ایم یا به اصطلاح Depth-wised کرده ایم. برای دومین مدل علاوه بر بخش بدنه، بخش انتهایی (Head) را نیز تغییر داده ایم. در ادامه هر مدل را به صورت مجزا بررسی و میزان تغییرات در سرعت و دقت آنرا گزارش میکنیم.

در ساختار شبکه YOLOv5 میان هر دو بلوک C3 که در بالا معرفی شده اند، یک بلوک کانوولوشنی قرار دارد که دو بلوک C3 را به یکدیگر متصل میکند. این بلوک کانوولوشنی دارای سه بخش فیلتر کانوولوشنی، لایه نرمالیزشن و همینطور یک تابع فعالساز می باشد.

فیلتر استفاده شده در این بلوک، از ابعاد $3 * 3$ می باشد که این فیلترها قرار است با فیلترهای کانوولوشنی در عمق جایگزین شوند.

عمل جایگزینی به این صورت است که پس از هر مرحله ی فیلتر کانوولوشنی در عمق یک لایه نرمالیزشن و تابع فعالسازی قرار میدهیم. در شکل زیر شمایی از تبدیل یک بلوک کانوولوشنی استاندارد به کانوولوشنی در عمق مشهود می باشد.

در این بلوک بسته به تعداد فیلترهای بکار رفته نسبت کاهش پارامترها برابر با $\frac{1}{N} + \frac{1}{D_k^2}$ می باشد. در این شبکه پارامتر D_k برابر با 3 و N همان تعداد لایه های یک فیلتر است. برای مثال اگر تعداد فیلترهای یک بلوک کانوولوشنی برابر با 32 باشد، پارامترها و محاسبات حدود 0.14 برابر خواهند شد.



شکل 20 - نحوه تبدیل یک بلوک استاندارد به در عمق (Depth-wise)

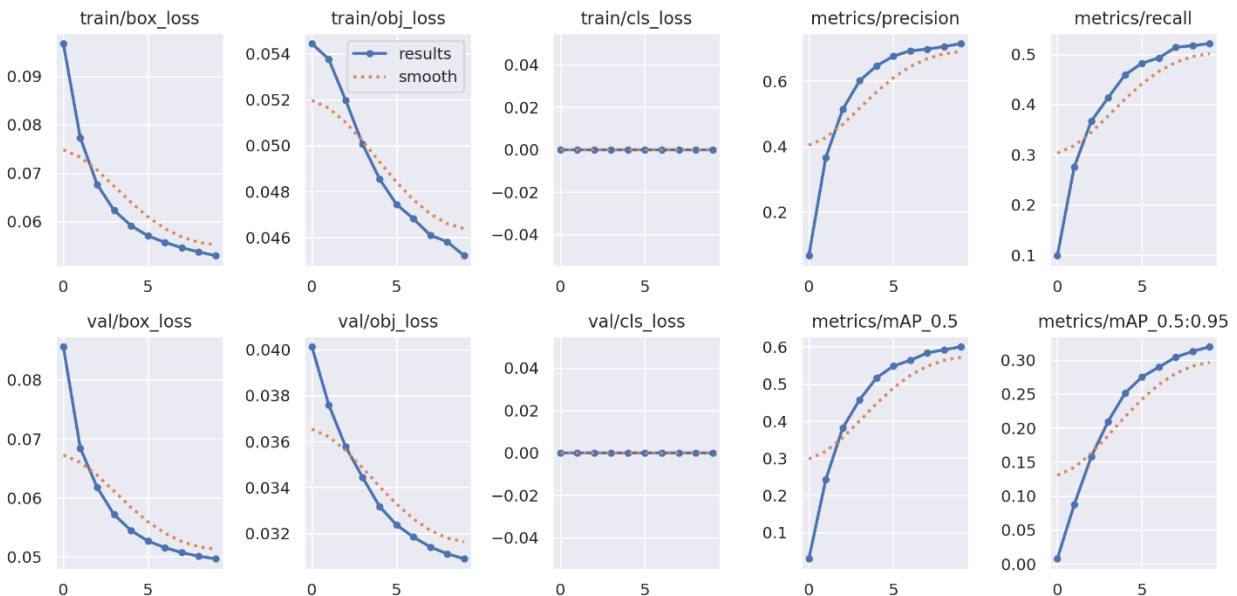
- مدل Depth-wise Backbone

در این مدل همانطور که بیان شد صرفاً بخش بدنه اصلاح شده است و عملگر کانوولوشن استاندارد با کانوولوشن در عمق جایگزین شده است.

این امر باعث شد تعداد پارامترهای شبکه از تعداد 1.76 میلیون به 1.41 میلیون برسد. این حجم از کاهش پارامتر سبب شد حجم محاسبات از 4.1 GFLOPs به 3.3 برسد.

سرعت این مدل نسبت به مدل اصلی، از 714 فریم بر ثانیه به 769 فریم بر ثانیه رسیده است که حدود 7 درصد سرعت مدل افزایش یافته است. از لحاظ دقت مدل نیز حدود 5.6 درصد در معیار mAP50 کاهش داشته ایم.

در جدول زیر میتوان عملکرد مدل را در مقایسه با مدل اورجینال مشاهده کرد. نمودار دقت و خطای مدل نیز در ادامه پیوست شده است.

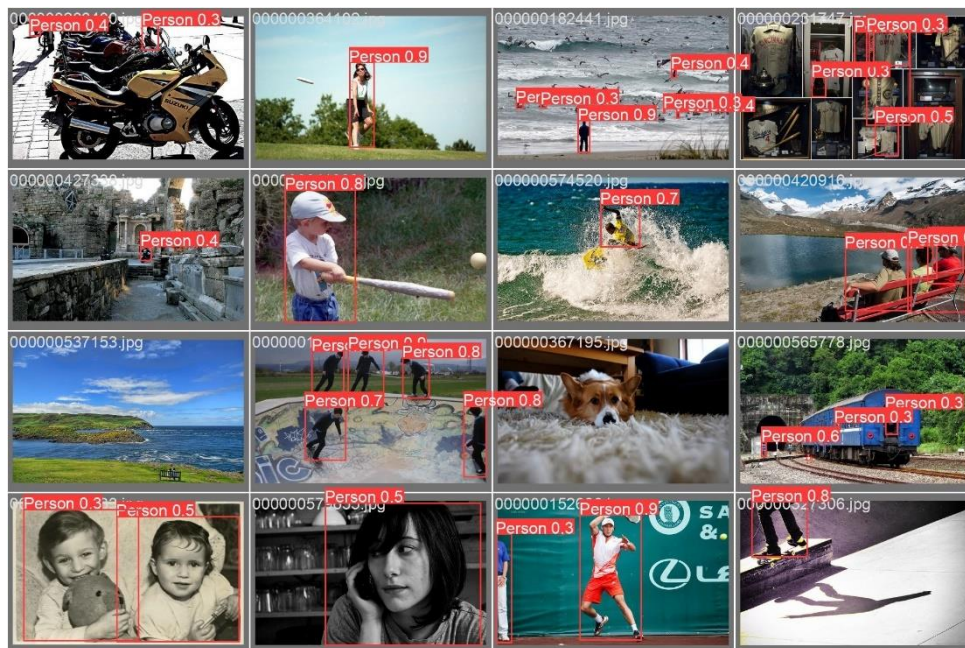


نمودار 1 - عملکرد مدل DW-BB در طول ده اپیک آموزش و ارزیابی

Model	mAP_val 50-95	mAP_val 50	Speed CPU(ms)	Speed GPU(ms)	Params (M)	GFLOPs @640	FPS CPU	FPS GPU
Original	35.8	64.5	93.2	1.4	1.760	4.1	10	714
DW-BB	32.6	60.9	89.6	1.3	1.415	3.3	11	769

جدول 1 - مقایسه دو مدل اصلی و Depth-wised Backbone

در ادامه چند نمونه از خروجی های مدل بر روی یک دسته 16 تایی از دیتای ارزیابی مشهود است. این نتیجه برای بهترین مدل که در واقع شامل وزن های ایپاک دهم است میباشد.



شکل 21 - چند نمونه از خروجی مدل Depth-wised Backbone

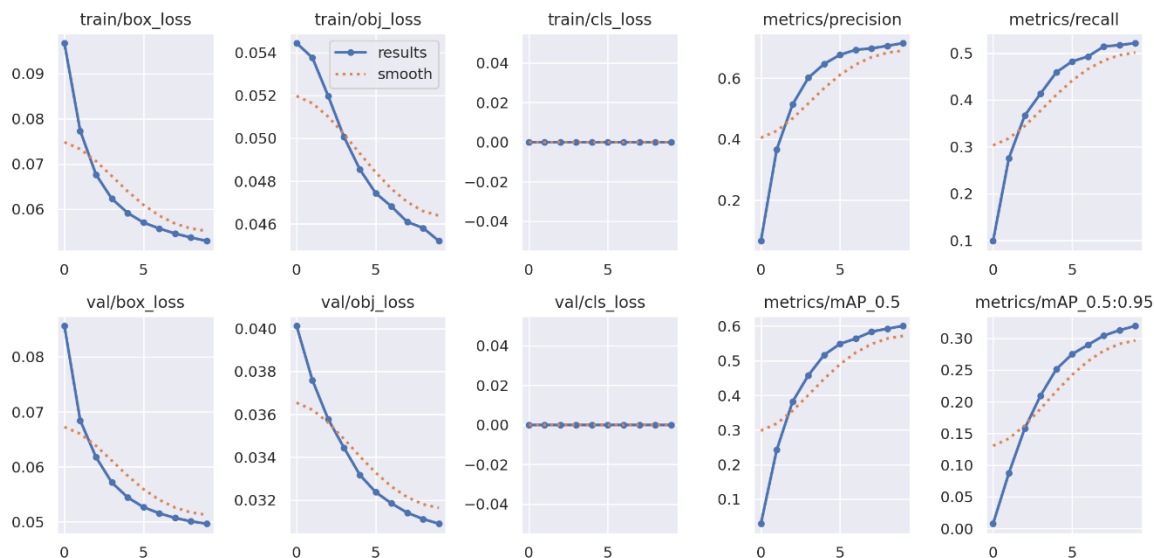
- مدل Depth-wise Backbone & Head

در این مدل علاوه بر بخش بدنه، بخش انتهایی نیز اصلاح شده است و بلوک کانولوشن استاندارد با بلوک کانولوشن در عمق جایگزین شده است.

این جایگزینی سبب شد تعداد پارامترهای شبکه از 1.76 میلیون به 1.25 میلیون برسد. این حجم از کاهش پارامترها باعث کاهش حجم محاسبات از 4.1 GFLOPs به 3.1 GFLOPs شد. با این اندازه از کاهش محاسبات سرعت مدل از 714 فریم بر ثانیه به 833 فریم بر ثانیه رسید.

در نهایت دقت مدل نیز حدود 6.8 درصد در معیار mAP50 کاهش یافته است و سرعت مدل نیز حدود 16.7 درصد افزایش یافته است (معادل با 119 فریم بر ثانیه).

در جدول زیر میتوان عملکرد مدل را در مقایسه با مدل اورجینال مشاهده کرد. نمودار دقت و خطای مدل نیز در ادامه پیوست شده است.

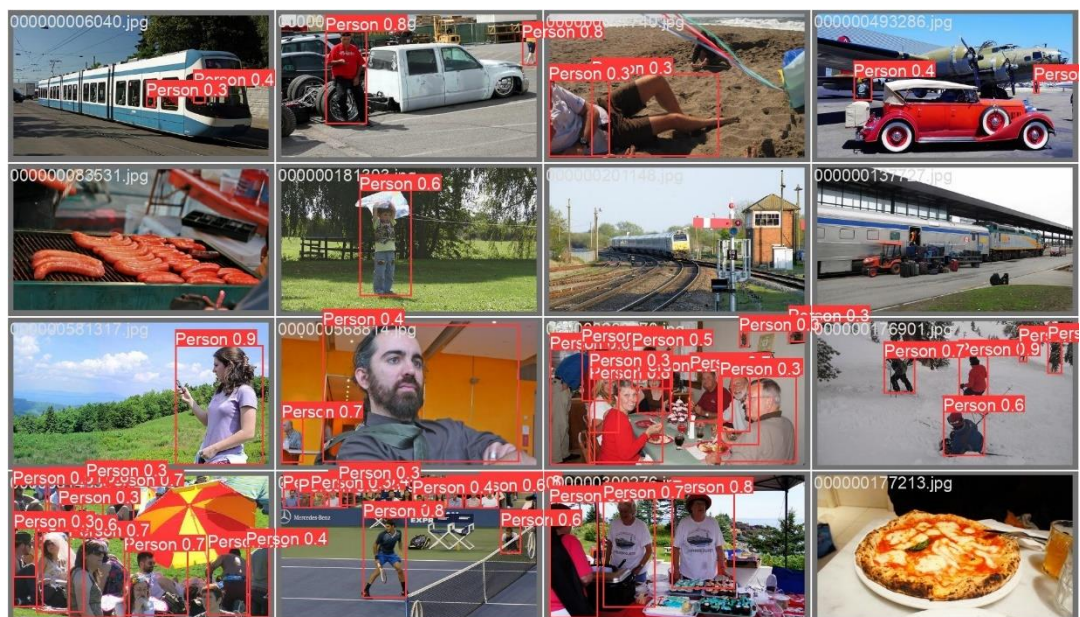


نمودار 2 - عملکرد مدل DW-BB&Head در طول ده اپیک آموزش و ارزیابی

Model	mAP_val 50-95	mAP_val 50	Speed CPU(ms)	Speed GPU(ms)	Params (M)	GFLOPs @640	FPS CPU	FPS GPU
Original	35.8	64.5	93.2	1.4	1.760	4.1	10	714
DW- BB&Head	32.0	60.1	83.7	1.2	1.253	3.1	11	833

جدول 2 - مقایسه دو مدل اصلی و Depth-wised Backbone & Head

در ادامه چند نمونه از خروجی های مدل بر روی یک دسته 16 تایی از دیتای ارزیابی مشهود است. این نتیجه برای بهترین مدل که در واقع شامل وزن های ایپاک دهم است میباشد.



شکل 22 - چند نمونه از خروجی مدل Depth-wised Backbone&Head

نتایج نهایی روش Depth Wised Convolution

همانطور که گفته شد این ایده بر روی دو بخش اصلی معماری شبکه YOLOv5 پیاده شد که منجر به دو مدل Depth-wised BB و مدل Depth-wised BB&Head شد. در ادامه سرعت و دقت دو مدل را در مقایسه با یکدیگر و همچنین در مقایسه با مدل اورجینال بررسی خواهیم کرد.

- مدل Depth-wised Backbone

در مدل Depth-wised BB تعداد پارامترها حدود 20 درصد نسبت به مدل اصلی کاهش یافته است و از مقدار 1.76 میلیون پارامتر به مقدار 1.41 میلیون پارامتر رسیده است.

این کاهش تعداد پارامترها باعث شده است میزان محاسبات شبکه از مقدار 4.1 GFLOPs به 3.3 GFLOPs برسد، یعنی حدود 19 درصد از هزینه محاسباتی شبکه کم شده است.

این کاهش تعداد پارامترها منجر به افزایش 7.7 درصدی سرعت شده است. در واقع سرعت پردازش مدل از 714 فریم بر ثانیه به 769 فریم بر ثانیه رسیده است.

لازم بذکر است این سرعت بدست آمده بر روی گرافیک پلتفرم کگل که دارای کانفیگ P100 میباشد محاسبه شده است و همین طور واحد پردازشی مرکزی (CPU) این سیستم نیز یک پردازنده Intel Xeon 2.2 GHz میباشد.

سرعت مدل بر روی این واحد پردازنده برابر با 89.6 میلی ثانیه میباشد که نسبت به مدل اصلی حدود 4 میلی ثانیه سریع تر شده است. از لحاظ معیارهای دقت مدل نیز حدود 5.6 درصد در معیار mAP50 کاهش داشته ایم که از 64.5 در مدل اصلی به 60.9 در این مدل رسیده ایم.

در معیار mAP50-95 نیز از مقدار 35.8 درصد به 32.6 درصد رسیده ایم که شاهد افت حدود 8.9 درصدی در این معیار بوده ایم.

- مدل Depth-wised Backbone & Head

در آخرین مدل یعنی مدل Depth-wised BB&Head تعداد پارامترها حدود 29 درصد نسبت به مدل اصلی کاهش یافته است و از مقدار 1.76 میلیون پارامتر به مقدار 1.25 میلیون پارامتر رسیده است.

این کاهش تعداد پارامترها باعث شده است میزان محاسبات شبکه از مقدار GFLOPs 4.1 به GFLOPs 3.1 برسد، یعنی حدود 24 درصد از هزینه محاسباتی شبکه کم شده است.

این کاهش تعداد پارامترها منجر به افزایش 16.7 درصدی سرعت شده است. در واقع سرعت پردازش مدل از 714 فریم بر ثانیه به 833 فریم بر ثانیه رسیده است.

سرعت مدل بر روی این واحد پردازنده برابر با 83.7 میلی ثانیه میباشد که نسبت به مدل اصلی حدود 9.5 میلی ثانیه سریع تر شده است. از لحاظ معیارهای دقت مدل نیز حدود 6.8 درصد در معیار mAP50 کاهش داشته ایم که از 64.5 در مدل اصلی به 60.1 در این مدل رسیده ایم.

در معیار mAP50-95 نیز از مقدار 35.8 درصد به 32.0 درصد رسیده ایم که شاهد افت حدود 10.6 درصدی در این معیار بوده ایم.

- مقایسه دو مدل DW-BB و DW-BB&Head

در نهایت میان دو مدل جدید اصلاح شده، مدل اول تنها حدود 0.8 درصد در معیار mAP50 عملکرد بهتری نسبت به مدل دوم داشته است. اما مدل دوم از لحاظ سرعت به مراتب بهتر از مدل اول می باشد. چرا که تنها با از دست دادن کمتر از یک درصد دقت، حدود 8.3 درصد سریع تر از مدل اول می باشد.

از لحاظ عملکرد زمانی بر روی واحد پردازشی مرکزی (CPU) نیز مدل دوم حدود 6 میلی ثانیه نسبت به مدل اول سریع تر است. در جدول زیر میتوان نتایج بدست آمده برای این دو مدل را مشاهده کرد.

Model	mAP_val 50-95	mAP_val 50	Speed CPU(ms)	Speed GPU(ms)	Params(M)	GFLOPs @640	FPS CPU	FPS GPU
DW-BB	32.6	60.9	89.6	1.3	1.415	3.3	11	769
DW-BB&Head	32.0	60.1	83.7	1.2	1.253	3.1	11	833

جدول 3 - مقایسه مشخصات و عملکرد دو شبکه تسريع شده

و در نهایت تمامی نتایج بدست آمده اعم از سرعت مدل بر روی کارت گرافیک و واحد پردازشی CPU، دقت مدل، حجم محاسبات و تعداد پارامترهای مدل را میتوان در جدول زیر مشاهده نمود.

Model	mAP_val 50-95	mAP_val 50	Speed CPU(ms)	Speed GPU(ms)	Params(M)	GFLOPs @640	FPS CPU	FPS GPU
Original	35.8	64.5	93.2	1.4	1.760	4.1	10	714
DW-BB	32.6	60.9	89.6	1.3	1.415	3.3	11	769
DW-BB&Head	32.0	60.1	83.7	1.2	1.253	3.1	11	833

جدول 4 – مقایسه مشخصات و عملکرد سه شبکه مورد آزمایش

جمع بندی

در این پروژه هدف اصلی که دنبال شد افزایش سرعت مدل اشکار ساز بود که با توجه به ایده های مطرح شده به آن دست پیدا کردیم و توانستیم با اندکی کاهش دقت تا 120 فریم بر ثانیه به سرعت مدل اضافه کنیم.

با توجه به ایده های پیاده شده، به دو مدل تسریع شده رسیدیم که در مدل اول حدود 7.7 درصد نسبت به مدل اصلی سریع تر شده است و این مقدار در مدل دوم حدود 16.7 درصد میباشد. از لحاظ دقت عملکرد مدلای به دست آمده نیز، مدل اول حدود 8.9 درصد و مدل دوم حدود 10 درصد کاهش داشته اند.

باتوجه میزان افزایش سرعت در مدل دوم، میتوان این کاهش دقت را ارزشمند دانست. چراکه باتوجه بیش برداش مدل بر روی محیط کاری خود، میتوان به راحتی این کاهش دقت را جبران کرد.

در ادامه سعی داریم با ترکیب این مدل با ایده های افزایش دقت مانند ایده مطرح شده در پایان نامه آقای پرویزی و استفاده از مدل GYOLO به مدلی دست پیدا کنیم که علاوه بر سریع تر بودن، دقت از دست رفته را نیز جبران کند.

امید است در ادامه کار بتوان از این پروژه برای ادامه کارهای تحقیقاتی و همین طور استفاده در صنعت بهره برد.

- [Video Summarization by Learning from Unpaired Data, CVPR 2019](#)
- [Making a Long Video Short: Dynamic Video Synopsis, CVPR 2006](#)
- [ByteTrack: Multi-Object Tracking by Associating Every Detection Box, ECCV 2022](#)
- [Brief Cam](#)
- [You Only Look Once: Unified, Real-Time Object Detection, CVPR 2016](#)
- [YOLOv3: An Incremental Improvement](#)
- [YOLO9000: Better, Faster, Stronger](#)
- [YOLOv4: Optimal Speed and Accuracy of Object Detection](#)
- [Scaled-YOLOv4: Scaling Cross Stage Partial Network](#)