بسمه تعالی

بررسی شبکه ی یولو از ابتدا تا به امروز

ر.ز

شهریور۱۴۰۱

چکیده

در این گزارش به بررسی شبکه ی معروف و پرکاربرد یولو از ورژن اول تا ورژن اخیر (ورژن هفتم تا زمان نوشتن این گزارش) می پردازیم. روش کار بررسی و خلاصه نویسی از مقالات اصلی –در صورت وجود- برای هر ورژن می باشد. علاوه بر یولو۱ تا ۷، ورژن های دیگری مانند YOLOX, YOLOR, PPYOLO نیز موجود می باشند اما خواسته ناظر پروژه در این گام بررسی یولو سری اول تا هفتم می باشد و لذا در این گزارش آورده نشدند.

ورژن اول یولو: You only look once

ورژن اول یولو توسط ردمان در سال ۲۰۱۵ با نام شما فقط یک بار نگاه می کنید ارائه شد(Redmon, 2015) . یولو به مساله تشخیص شی به دید رگرسیون نگاه کرده و با دادن ویژگی های استخراج شده از تصویر به یک شبکه عصبی، کادر و احتمال کلاس برای آن شی را پیش بینی می کند. چون یولو از یک شبکه استفاده می کند لذا آموزش و تست آن سریع تر است.در زمان ارائه نسخه ی سریع آن به سرعت 150 fps روی تایتان ایکس و دقت 52.7mAp -دو برابر مدل های مشابه بلادرنگ- دست یافت. نسخه ی عادی یولو نیز دارای دقت 63.4 mAP و سرعت 45 fps می باشد. پردازش بلادرنگ ویدیو با 25 ms تاخیر توسط یولو قابل پیاده سازی است. دموی پیاده سازی و مقایسه با دیگر مدل های هم دوره این پروژه در سایت https://pjreddie.com/darknet/yolo/ قابل مشاهده است.

بر خلاف شبکه ی DPM -رهیافت sliding window- و R-CNN –رهیافت proposal regions- شبکه یولو کل تصویر را در هنگام آموزش و تست می بیند. لذا تعداد خطاهای پس زمینه نصف R-CNN است. هم چنین نسبت به دیگر شبکه ها با فاصله زیادی تعمیم پذیری بیشتری دارد، بدین معنا که میتوان آن را با تصاویر طبیعی آموزش داد و سپس روی تصاویر هنری تست کرد.

تشخیص یکپارچه

شبکه یولو از کل تصویر برای پیش بینی هر کادر (bounding box) استفاده می کند و قابلیت آموزش end-to-end دارد. این شبکه تصویر ورودی را به گرید S*S تقسیم کرده و اگر مرکز یک شی در یکی از سلول ها قرار گیرد آن سلول مسئول تشخیص آن شی است. هر سلول B کادر و سطح اطمینان مربوطه را پیش بینی می کند، یعنی هر کادر دارای پنج پیشبینی است: مختصات مرکز کادر نسبت به مرزهای سلول گرید، طول و عرض کادر نسبت به کل تصویر و سطح اطمینان که نشانگر IOU بین پیش بینی و واقعیت است. هم چنین هر سلول احتمال شرطی وجود شی در سلول را برای هر C کلاس پیش بینی می کند. با این توصیفات تنسور پیشبینی نهایی دارای سایز S*S * (B*5 +C) می باشد.

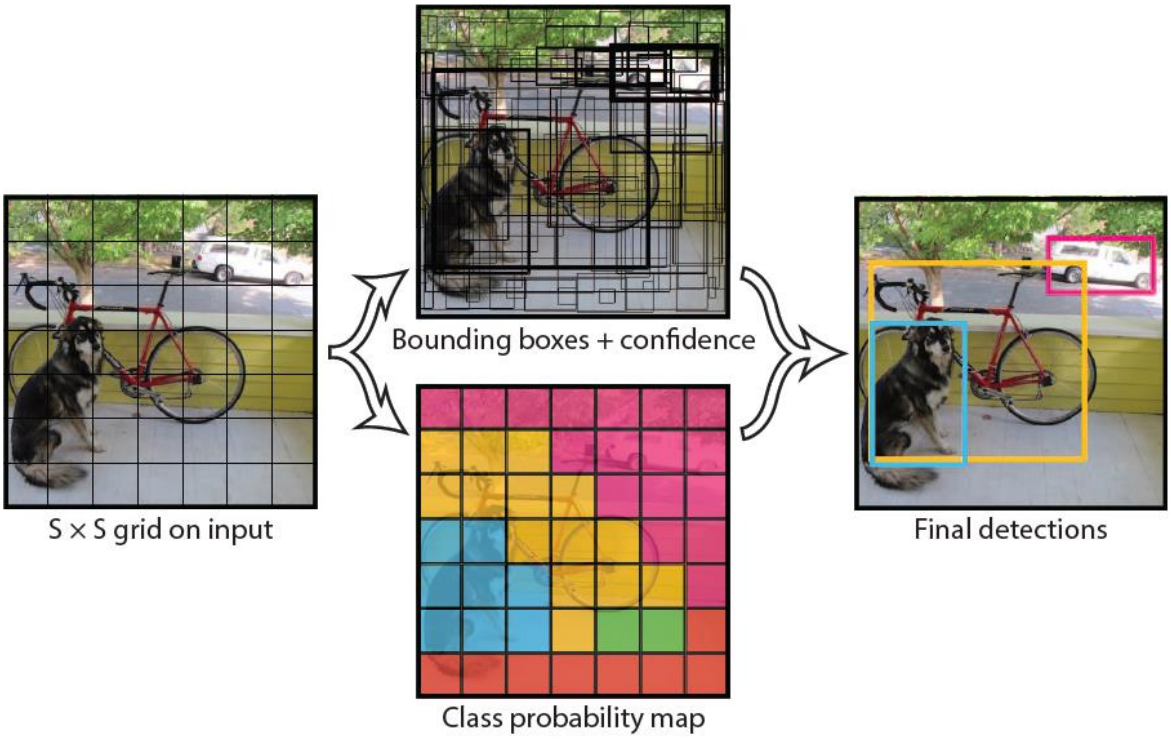
معماری شبکه و آموزش

با الهام از شبکه GoogleNet از ۲۴ لایه کانولوشنی و سپس دو لایه تمام متصل استفاده شده است، بجای ماژول اینسپشن از یک لایه کاهشی 1*1 و سپس conv 3*3 استفاده شده است. برای ورژن سریع تعداد لایه های کانولوشنی به ۹ عدد کاهش می یابد.

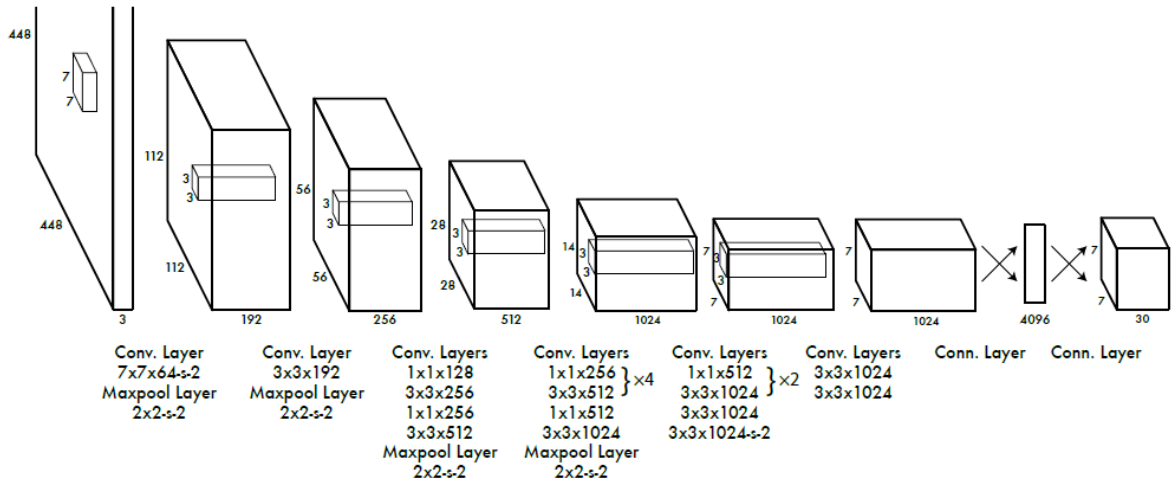
برای آموزش شبکه ابتدا ۲۰ لایه روی داده های ۱۰۰۰ کلاسی ImageNet به مدت یک هفته آموزش داده شده است. کادرها نسبت به بعد تصویر نرمالیزه می شوند به طوری که ابعاد آن ها بین صفر و یک باشد. هم چنین برای بهبود دقت روی اشیای کوچک از رزولیشن 448*448 بجای224*224 استفاده شده است. لایه آخر از تابع فعال سازی خطی و دیگر لایه ها از leaky ReLu استفاده می کنند. آموزش روی داده های پاسکال2007,2012 voc با بچ سایز ۶۴ و ۱۳۵ ایپاک انجام شده است.

برای جلوگیری از بیشبرازش از لایه دراپ اوت قبل از فولی کانکتید با نرخ 0.5 و آگمنتیشن تصاویر -۲۰ درصد تغییر مقیاس و انتقال و تغییر تصادفی اشباع و اکسپوز تا ضریب 1.5- استفاده شده است.

این مدل دارای محدودیت هایی می باشد به عنوان مثال در تعداد اشیای تشخیص داده شده مجاور هم دارای محدودیت است. هم چنین چون کادر ها در فاز آموزش یادگرفته می شوند نسب به ابعاد و یا نسبت های جدید خطا دارد. منبع اصلی خطا محل یابی اشتباه است.



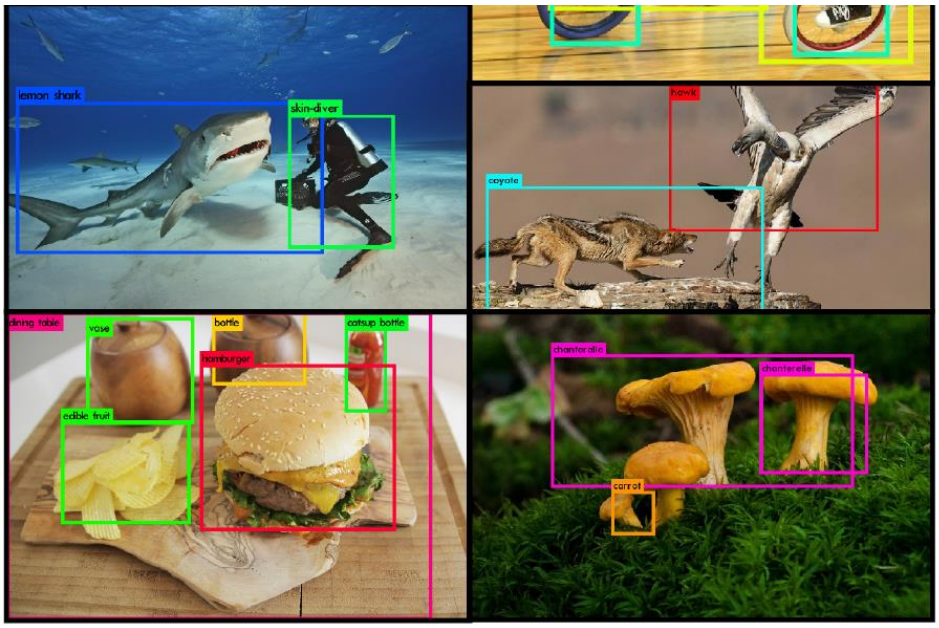
شکل ۱ گرید بندی تصویر ورودی و پیش بینی کادر، سطح اطمینان و احتمال کلاس برای هر سلول



شکل ۲ معماری شبکه یولو ورژن اول

جدول1 مقایسه ی یولو۱ با DPM و RCNN از نظر سرعت و دقت

Real-Time Detectors	Train	mAP	FPS
100Hz DPM [30]	2007	16.0	100
30Hz DPM [30]	2007	26.1	30
Fast YOLO	2007+2012	52.7	**155**
YOLO	2007+2012	**63.4**	45
Less Than Real-Time			
Fastest DPM [37]	2007	30.4	15
R-CNN Minus R [20]	2007	53.5	6
Fast R-CNN [14]	2007+2012	70.0	0.5
Faster R-CNN VGG-16[27]	2007+2012	73.2	7
Faster R-CNN ZF [27]	2007+2012	62.1	18
YOLO VGG-16	2007+2012	66.4	21



شکل ۳ اجرای مدل یولو۲ روی چند نمونه تصویر

ورژن دوم یولو: YOLO-9000

در سال ۲۰۱۶ ورژن دوم یولو که قابلیت تشخیص ۹۰۰۰ شی را داشت –و بهمین خاطر پسوند ۹۰۰۰ در نام آن وجود دارد- ارائه شد(Joseph Redmon, Ali Farhadi, 2016) . پیاده سازی و دادگان پروژه به طور آزاد در http://pjreddie.com/yolo9000/ قابل دسترس است.

از آنجایی که دادگان برای طبقه بندی و یا تگ کردن تصاویر بیشتر و آسان تر در دسترس است، در این مدل با دیدگاه سلسله مراتبی و آموزش جوینت سعی شده از این دادگان در تشخیص اشیا استفاده شود. یعنی با دیدن تصویری که دارای لیبل دیتکشن است، شبکه دیتکشن آموزش می بیند، و با دیدن تصویر با لیبل طبقه بندی، تنها بخش متناظر با طبقه بند آموزش می بیند. برای حل مشکل تفاوت لیبل ها نیز از WordNet استفاده شده است

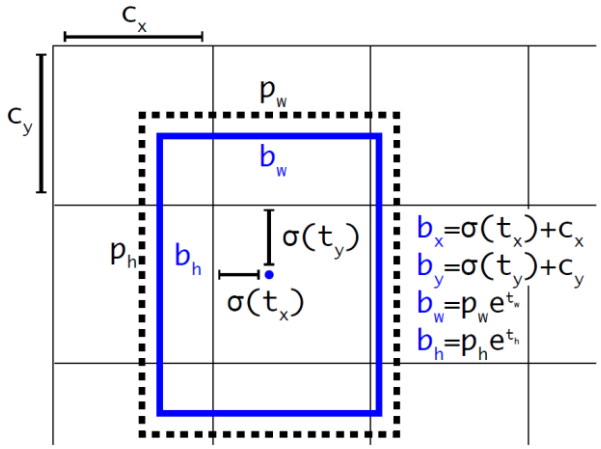
تغییرات نسبت به ورژن اول یولو

همان طور که گفته شد، ورژن اول یولو دارای معایبی از جمله خطای بالای محل یابی و ریکال پایین می باشد. با استفاده از بچ نرمالیزاسیون علاوه بر افزایش دو درصدی دقت، دیگر نیازی به رگولاسیون توسط دراپ اوت نیز نیست. هم چنین با افزایش رزولیشن ورودی شبکه (از ۲۲۴ به ۴۴۸) در مرحله ی طبقه بندی دقت چهار درصد افزایش یافت.

برای داشتن تعداد فرد سلول، رزولیشن به ۴۱۶ کاهش داده شده است. تعداد فرد برای گرید باعث می شود در مرکز تصویر سلول داشته باشیم که با تصاویر حاوی شی بزرگ سازگاری بهتری دارد. با حذف یک لایه پولینگ، داونسمپل از ۶۴ به ۳۲ کاهش یافته و رزولیشن نقشه ویژگی به ۱۳*۱۳ افزایش می یابد.

دو لایه فول کانکتید نهایی حذف شده و در نتیجه شبکه یولو۲ تماما کانولوشنی است. برای پیش بینی کادر نیز از انکورهای از قبل تعریف شده استفاده می شود. با این رهیافت بجای حدود ۱۰۰ باکس، حدود هزار باکس برای هر تصویر خواهیم داشت و دقت هم کمی (0.3%) کاهش می یابد اما در عوض ریکال از ۸۱ به ۸۸ درصد افزایش می یابد. برای بهبود عملکرد، بهتر است ابعاد انکور هوشمندانه تعیین شوند، بدین منظور از خوشه بندی k-means استفاده شده است. مقدار k برابر پنج و از مفهوم IOU برای تعریف فاصله استفاده شده است. چالش دیگری که استفاده از انکور بوجود می آورد، ناپایداری مدل بویژه در تکرارهای اولیه می باشد. چرا که انکور ها هر مقداری می توانند داشته باشند. برای حل این چالش با استفاده از رهیافت مشابه ورژن اول، محل انکورها را نسبت به سلول تعریف می کنیم. در شکل ۴ ارتباط کادر پیش بینی شده با مرکز خوشه انکور از پیش تعریف شده نشان داده شده است.

برای دستیابی به رزولیشن بیشتر و در نتیجه تشخیص بهتر اشیا ریزتر، الگوریتم های faster_RCNN و SSD از اعمال شبکه پروپوزال بر روی نقشه ویژگی های مختلف برای بدست آوردن رنج رزولیشن استفاده می کنند. اما یولو۲ از یک لایه passthrough -که ویژگی ها را از لایه قبلی منتقل می کند- استفاده می کند. برای تطابق ابعاد و کانکتید کردن رزولیشن ها از رهیافت مشابه identity در ResNet، یعنی پشته سازی ویژگی های همسایه در کانال های مختلف، استفاده شده است. لایه 512*26*26 به 1024*13*13 تبدیل می شود تا قابلیت کانکتید شدن با نقشه ویژگی نهایی (13*13) را داشته باشد.

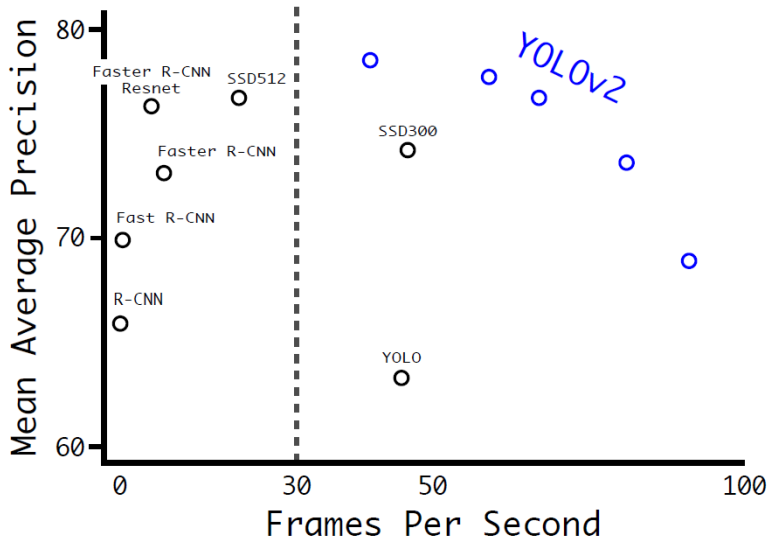


شکل ٤ پیش بینی مرکز و ابعاد کادر بر حسب مرکز خوشه (انکر از قبل تعریف شده)

چون تنها از لایه های کانولوشنی و پولینگ استفاده شده است، می توان ریسایز تصاویر را on-fly انجام داد و لذا برای مقاوم شدن نسبت به تغییر سایز، مدل را به صورت چند سایزی از ۳۲۰ تا ۶۰۸ و به صورت مضارب ۳۲ (مقدار ضریب داونسمپل) آموزش داد. هر ده ایپاک سایز تصاویر ورودی با یکی از این اعداد به طور تصادفی تغییر می کند.

در بحث دقت شبکه یک مصالحه بین سایز تصویر ورودی (و در نتیجه سرعت مدل) و دقت بدست آمده وجود دارد، به طوری که در رزولیشن بالا به دقت 78.6 (بالاترین دقت بین مدل های هم دوره) می رسد. در شکل ۵ مصالحه سرعت-دقت برای یولو، یولو۲، F-RCNN و SSD مشاهده می شود.

در مدل یولو۲ از شبکه ای مشابه vgg2 با سرعت بیشتر (و دقت کمی کمتر) به نام Darknet19 استفاده شده است که جزییات آن را در شکل ۶ مشاهده می کنید. هم چنین در شکل ۷ خلاصه ای از تغییرات اعمال شده به مدل و اثر آن ها بر روی دقت و در نتیجه مسیر توسعه از مدل یولو۱ به یولو۲ بررسی شده است.



شکل ۵ مصالحه ی بین دقت و سرعت برای یولو، RCNN و SSD روی voc 2007

Type	Filters	Size/Stride	Output
Convolutional	32	3×3	224×224
Maxpool		$2 \times 2/2$	112×112
Convolutional	64	3×3	112×112
Maxpool		$2 \times 2/2$	56×56
Convolutional	128	3×3	56×56
Convolutional	64	1×1	56×56
Convolutional	128	3×3	56×56
Maxpool		$2 \times 2/2$	28×28
Convolutional	256	3×3	28×28
Convolutional	128	1×1	28×28
Convolutional	256	3×3	28×28
Maxpool		$2 \times 2/2$	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Maxpool		$2 \times 2/2$	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	1000	1×1	7×7
Avgpool		Global	1000
Softmax			

شکل ٦ جزییات ساختار شبکه DarkNet19 به کار رفته در مدل یولو۲

	YOLO								YOLOv2
batch norm?		✓	✓	✓	✓	✓	✓	✓	✓
hi-res classifier?			✓	✓	✓	✓	✓	✓	✓
convolutional?				✓	✓	✓	✓	✓	✓
anchor boxes?				✓	✓				
new network?					✓	✓	✓	✓	✓
dimension priors?						✓	✓	✓	✓
location prediction?						✓	✓	✓	✓
passthrough?							✓	✓	✓
multi-scale?								✓	✓
hi-res detector?									✓
VOC2007 mAP	63.4	65.8	69.5	69.2	69.6	74.4	75.4	76.8	**78.6**

شکل ۷ مسیر اعمال تغییرات به ورژن دوم از ورژن اول برای یولو و دقت بدست آمده پس از اعمال هر تغییر

YOLOv3

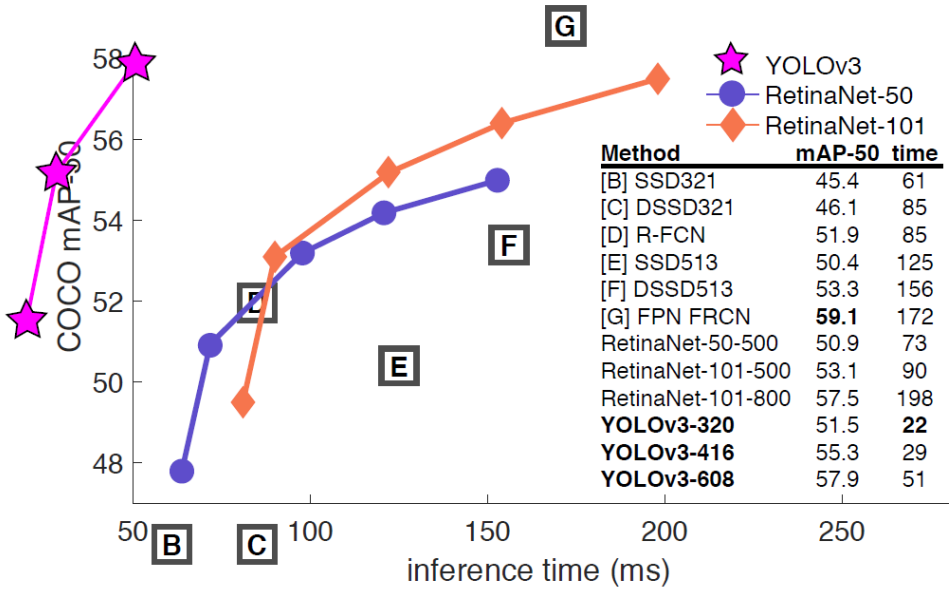
مقاله اصلی ارائه شده توسط Redmon –سازنده دو ورژن قبلی- یک تکنیکال ریپورت است و حتی در بسیاری موارد ساختار غیر رسمی پیدا می کند (Joseph Redmon, Ali Farhadi, 2018) .

یولو۳ کادرها را در سه مقیاس پیش بینی می کند و از مفهومی مشابه FPN: Feature Pyramid Network برای استخراج ویژگی استفاده می کند. هم چنین بک بون شبکه از دارکنت۱۹ به دارکنت۵۳ تغییر یافته است.

در شکل ۸ معماری بک بون دارکنت و در شکل ۹ مقایسه دقت و سرعت یولو۳ با دیگر شبکه ها مشاهده می شود.

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

شکل ۸ معماری شبکه Darknet-53 بکار رفته در یولو۳



شکل ۹ مقایسه یولو۳ با شبکه های رقیب مانند Retina, SSD

YOLOv4

در سال ۲۰۲۰ توسط Bochkovskiy ورژن چهارم یولو ارائه شد(Bochkovskiy, 2020) . در این ورژن با روش هایی مانند :

- Weighted-Residual-Connections (WRC)
- Cross-Stage-Partial-connections (CSP)
- Cross mini-Batch Normalization (CmBN)
- Self-adversarial-training (SAT)
- Mish-activation
- Mosaic data augmentation,
- DropBlock regularization
- CIoU loss

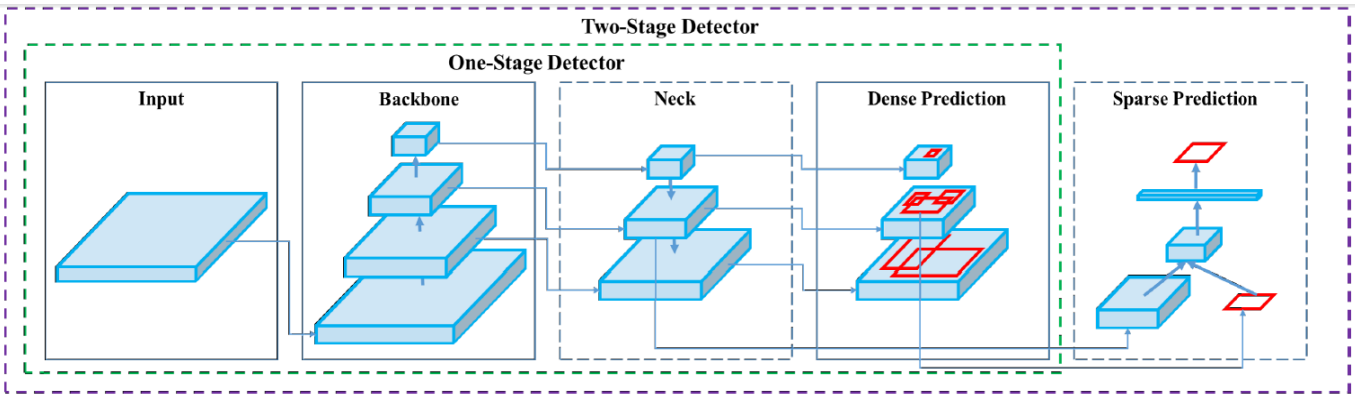
برای افزایش دقت و رسیدن به مقدار 43.5% AP روی دیتاست MS COCO با سرعت 65 FPS روی Tesla V100 استفاده شده است. کد در گیت هاب https://github.com/AlexeyAB/darknet قابل دسترس است. هم چنین یک گام با ارزش که در این ورژن برداشته شده است حذف نیاز به چندین GPU برای آموزش مدل است. یولو۴ می تواند روی تنها یک GPU مرسوم آموزش ببیند.

مرور ساختار مدل های تشخیص اشیا

در شکل ۱۰ معماری دیتکتور های تک و دو مرحله ای مشاهده می شود. بخش Input شامل تصویر، پچ، image Pyramid و ... می تواند باشد. Backbone نیز معماری های معروفی مانند VGG, ResNet, DarkNet53 برای GPU و SquuzeNet, MobileNet, ShuffleNet برای CPU ا ست. بخش Head را می توان به دو د سته یک و دو مرحله ای که هر کدام دو حالت با و بدون انکور دارند، تقسیم کرد. معروف ترین مدل دو مرحله ای خوانواده R-CNN می باشد که از نوع دارای انکور است. مدل دو مرحله ای بدون انکور نیز توسعه یافته است مانند RepPoints. مدل های یک مرحله ای معروف مانند YOLO, SSD, RetinaNet می باشند که دارای انکور هستند. نوع بدون انکور مدل های تک مرحله ای نیز موجود است مانند:CenterNet, CornerNet, FOCS . به مدل های تک و دو مرحله ای، پیش بینی dense و sparse نیز می گویند.

معمولا بین بخش بکبون و هد، لایه هایی با هدف جمع آوری ویژگی از مراحل گوناگون قرار می گیرد که در اینجا گردن (neck) نامیده می شوند. مانند:

Feature Pyramid Network (FPN), Path Aggregation Network (PAN), BiFPN, NAS-FPN



شکل ۱۰ معماری شبکه های تشخیص اشیا تک مرحله ای و دو مرحله ای

آموزش بهتر مدل

برای آموزش بهتر مدل و رسیدن به دقت بالاتر (معمولا به ازای صرف زمان و محاسبات بیشتر) معروف ترین روش آگمنتیشن است. مثلا آگمنتیشن فوتونیکی (تغییر سطح روشنایی، کانتراست، هیو، اشباع و نویز) و هندسی (تغییر مقیاس، برش، آینه و چرخش تصادفی) و یا حتی روش هایی مانند حذف بخشی از تصویر و یا پر کردن آن با مقدار صفر (برای مدل سازی وجود مانع) به کار می رود. گاهی نیز آگمنتیشن به طور ترکیبی روی چند تصویر انجام می شود مانند روش MixUp. از style transfer GAN نیز می توان برای کاهش بایاس تسکتچر تصاویر استفاده کرد.

یک چالش دیگر عدم تعادل بین دادگان کلاس های گوناگون است که برای حل آن focal loss استفاده شده است. روش label smoothing نیز با تبدیل لیبل های سخت one-hot به لیبل های نرم، باعث مقاوم تر شدن مدل می شوند. برای بهبود رگرسیون bbox (چه در حالت بدون انکور که مختصات مرکز و طول و عرض و یا مختصات دو گوشه کادر پیش بینی می شوند و چه در حالت با انکور که مقادیر آفست متناظر پیش بینی می شوند) معیارهایی بر مبنایم IoU loss ارائه شده است. مثلا GIoU شکل و جهت شی را علاوه بر میزان هم پوشانی در نظر می گیرد، DIoU فواصل مراکز را در نظر میگیرد و یولو۴ از CIoU loss که ناحیه هم پوشان، فاصله بین مراکز و نسبت ابعاد را در نظر می گیرد، استفاده می کند.

ماژول های افزونه و پس پردازش

برخی تکنیک ها و ماژول ها منجر به افزایش قابل توجه دقت مدل (به ازای کمی افزایش زمان اجرا) می شوند. به عنوان مثال برای بهبود میدان رسپکتیو از ماژول هایی مانند: SPP,ASPP و RFB می توان استفاده کرد. ماژول SPP از Spatial Pyramid Matching (SPM) ریشه گرفته است به طوری که SPM را در شبکه های CNN مورد استفاده قرار داده و بجای بخش bag of word از max pol استفاده کرده است اما از آنجایی که یک

بعدی ا ست، نمی تواند برای لایه تماما مت صل مورد ا ستفاده قرار گیرد لذا ردمان در یولو۳ آن را قابل ات صال به خروجی ماکس پول با ابعاد کرنل 1,5,9,13 و گام یک بهبود داد که باعث بهبود AP 2.7% تنها با افزایش 0.5% محاسبات شد. روش RFB دارای چندین کانولوشن دیلیتید k*k با نرخ دیلی k و گام یک می باشد که منجر به افزایش AP_{50} 5.7% روی COCO با هزینه 7% محاسبات بیشتر در شبکه SSD می شود.

ماژول دیگر، ماژول توجه ا ست که به دو د سته برح سب کانال مانند SE[1] و برح سب نقطه مانند SAM[2] تق سیم بندی می شود. SE بیشتر برای CPU مناسب است چرا که برای GPU حدود ده درصد اضافه محاسبات دارد اما SAM روی GPU تقریبا اضـافه باری ندارد و روی CPU نیز تنها 0.1% افزایش محاسـبات دارد و دقت رزنت را 0.5% بهبود می دهد.

برای تجمیع ویژگی ها یک ایده عملی قدیمی ا ستفاده از skip connection می با شد. از آنجایی که پیش بینی چند مقیاسـه مانند FPN معروف شـده اند، ماژول های گوناگونی برای تجمیع ویژگی ها از سـطوح مختلف ابداع شده اند مانند:SAFM, ASFF و BiFPN که مورد اخیر از ات صالات reidual با ورودی چند گانه وزن دهی شده برای جمع کردن ویژگی از مقیاس های گوناگون استفاده می کند.

در سال ۲۰۱۰ تابع فعالساز ReLu برای حل م شکل محو گرادیان (در توابع tanh و sigmoid) ارائه شد. سپس P/L-ReLu برای حل م شکل صفر بودن تابع در مقادیر کمتر از صفر پی شنهاد شدند. برای شبکه های کوانتیده ReLu6 و hard Swish مناسب می باشند. SELU برای خود-نرمالیزاسیون ابداع شد. هم چنین Swish و mish هر دو به طور پیوسته مشتق پذیر می باشند.

مر سوم ترین پس پردازش شبکه های ت شخیص ا شیا، NMS می با شد که کادرهای با نمره پایین را حذف می کند (برای شبکه های بدون انکور لازم نیست). Soft NMS حالت کاهش اطمینان نا شی از وجود مانع را در نظر می گیرد و DIoU NMS علاوه بر آن فواصل از مرکز را نیز لحاظ می کند.

جدول2 اثر PANet,SPP,RBF,SAM بر روی شبکه ResNeXt-50

Model	AP	AP_{50}	AP_{75}
CSPResNeXt50-PANet-SPP	42.4%	64.4%	45.9%
CSPResNeXt50-PANet-SPP-RFB	41.8%	62.7%	45.1%
CSPResNeXt50-PANet-SPP-SAM	**42.7%**	**64.6%**	**46.3%**
CSPResNeXt50-PANet-SPP-SAM-G	41.6%	62.7%	45.0%
CSPResNeXt50-PANet-SPP-ASFF-RFB	41.1%	62.6%	44.4%

[1] Squeeze and Excitation
[2] Spatial Attention Module

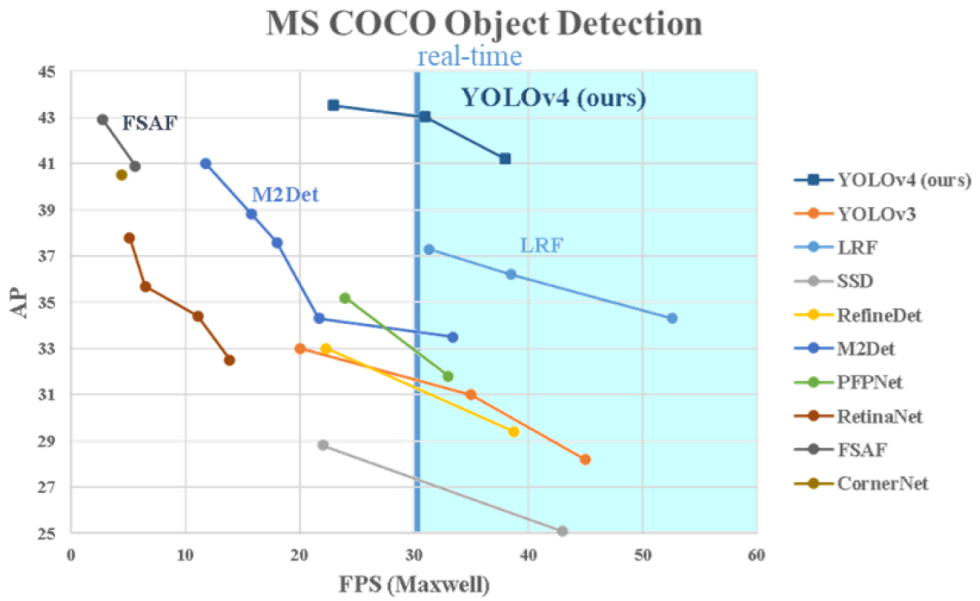
جدول3 اثر دیتا آگمنتیشن، لیبل اسموسینگ و تابع فعالساز روی دقت شبکه ResNeXt-50

MixUp	CutMix	Mosaic	Bluring	Label Smoothing	Swish	Mish	Top-1	Top-5
							77.9%	94.0%
✓							77.2%	**94.0%**
	✓						**78.0%**	**94.3%**
		✓					**78.1%**	**94.5%**
			✓				77.5%	93.8%
				✓			**78.1%**	**94.4%**
					✓		64.5%	86.0%
						✓	**78.9%**	**94.5%**
	✓	✓		✓			**78.5%**	**94.8%**
	✓	✓		✓		✓	**79.8%**	**95.2%**

در یولو۴ از دارکنت۵۳ برای بکبون، از SPP برای افزایش زاویه دید، از PANet به عنوان مسیر تجمع (گردنه مدل) و از هد یولو۳ انکوردار استفاده شده است. برای دیتا آگمنتیشن روش جدید Mosaic که چهارتصویره است استفاده شده است. به علاوه نرمالیزاسیون بچ نیز از چهار تصویر برای محاسبه آمارگان استفاده می کند که باعث کاهش نیاز به بچ سایز بزرگ می شود. هم چنین روش Self Adversarial Training (SAT) نیز در دو مرحله برای دیتاآگمنتیشن معرفی شده است. اثر دیتا آگمنتیشن و لیبل اسموسینگ در جدول3 نشان داده شده است.

در بکبون یولو۴ از دیتاآگمنتیشن CutMix و Mosaic، از رگولارایزرDropBlock، نرم سازی لیبل کلاس ها، Mish activation، Cross Stage Partial Connection (CSP) و MiWRC استفاده شده است.

و در آخر برتری یولو۴ از حیث دقت و سرعت در مقایسه با دیگر شبکه ها در شکل ۱۱ مشاهده می شود.



شکل ۱۱ مقایسه دقت و سرعت یولو۴ با دیگر شبکه های معروف و مهم

یولو ورژن پنجم

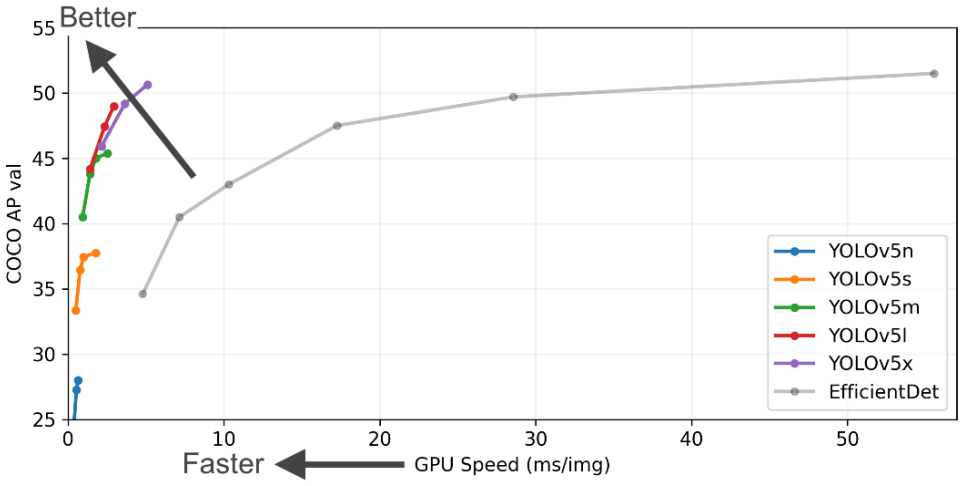
حدود یک ماه پس از انتشار یولو۴، یولو پنج توسط Glen Jocher –CEO of Ultrylistic- ارائه شد. کمپانی وی (Ultrylistic) ورژن های مختلف یولو را روی پایتورچ پیاده سازی کرده است و از آنجایی که پایتون و پایتورچ معروف تر بوده و نصب و استفاده از آن راحت تر است مورد پذیرش جامعه هوش مصنوعی بیشتر قرار گرفت. هم چنین Jocher مبدع دیتا آگمنتیشن Mosaic مورد استفاده در یولو۴ می باشد. تا جایی که نگارنده اطلاع دارد مقاله ای برای یولو۵ توسط Jocher ارائه نشده است و منبع این بخش یک پایان نامه تحقیقاتی (Thuan, 2021) در مورد یولو است.

معماری یولو۵ بسیار مشابه یولو۴ می باشد:

- بک بون: focus, CSP Net
- گردن: SPP, PANet
- هد: یولو۳ هد به همراه GIoU-loss

یک تفاوت درخشان بین یولو۵ و ۴ مربوط به پیش بینی کادر می باشد. همان طور که اشاره شد در ورژن های قبلی برای پیش بینی سایز انکورها از k-means استفاده می شد که در نتیجه وابستگی به دیتاست اولیه ایجاد می کرد. در یولو۵ پروسس انتخاب ابعاد انکور به طور اتوماتیک در خود یولو انجام می شود و متناسب با هر دیتاست هنگام آموزش این ابعاد بدست می آیند.

داکیومنت این ورژن در گیت هاب به طور مفصل آمده است: https://github.com/ultralystics/yolov5

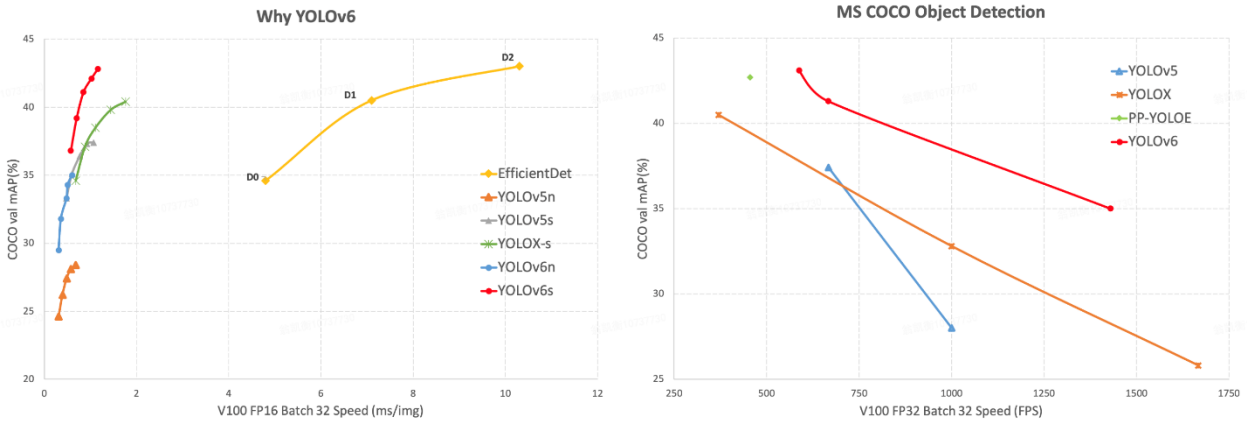


شکل ۱۲ نتایج (دقت و سرعت) یولو۵ (سایزهای مختلف) و مقایسه با EfficentNet

جدول4 ابعاد تصویر ورودی، mAP، سرعت و تعداد پارامتر برای یولو۵ (در سایزهای مختلف)

Model	size (pixels)	mAP^{val} 0.5:0.95	mAP^{val} 0.5	Speed CPU b1 (ms)	Speed V100 b1 (ms)	Speed V100 b32 (ms)	params (M)	FLOPs @640 (B)
YOLOv5n	640	28.0	45.7	**45**	**6.3**	**0.6**	**1.9**	**4.5**
YOLOv5s	640	37.4	56.8	98	6.4	0.9	7.2	16.5
YOLOv5m	640	45.4	64.1	224	8.2	1.7	21.2	49.0
YOLOv5l	640	49.0	67.3	430	10.1	2.7	46.5	109.1
YOLOv5x	640	50.7	68.9	766	12.1	4.8	86.7	205.7

یولو ورژن ششم



شکل ۱۳ دقت و سرعت یولو۶ در مقایسه با یولو۵ و یولو x

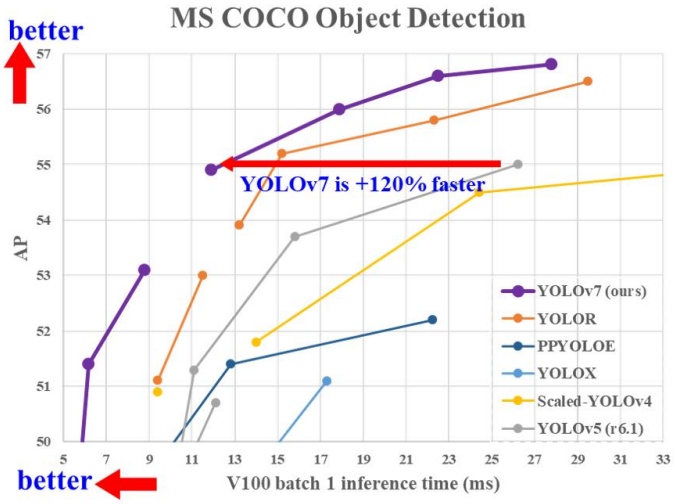
مشـابه ورژن پنجم، تا بحال مقاله ای برای آن به طور رسـمی ارائه نشـده اسـت و فقط دارای ریپو در گیت می باشـد[۳]. طبق اعلام ارائه دهنده یولو۶ نانو به دقت 35 mAP روی COCO val2017 با سـرعت 1242 FPS روی T4 با تسـنورآرتی (FP16 for bs32) دسـت پیدا کرده اسـت. یولو۶ ورژن s نیز به مقادیر متناظر 43.1 mAP و 520 FPS رسیده است. مقایسه با یولو۵ و یولو x در شکل ۱۳ مشاهده می شود.

برای یولو۶ دو ویژگی متمایز را می توان نام برد: طراحی متنا سب با سخت افزار برای بک بون و گردنه معماری، هد دیکوپل شده توسط SIoU Loss .

Model	Size	mAP^{val} 0.5:0.95	$Speed^{V100}$ fp16 b32 (ms)	$Speed^{V100}$ fp32 b32 (ms)	$Speed^{T4}$ trt fp16 b1 (fps)	$Speed^{T4}$ trt fp16 b32 (fps)	Params (M)	FLOPs (G)
YOLOv6-n	416 640	30.8 35.0	0.3 0.5	0.4 0.7	1100 788	2716 1242	4.3 4.3	4.7 11.1
YOLOv6-tiny	640	41.3	0.9	1.5	425	602	15.0	36.7
YOLOv6-s	640	43.1	1.0	1.7	373	520	17.2	44.2

شکل ۱۴ ابعاد تصویر ورودی، دقت، سرعت و تعداد پارامتر ها برای ورژن های مختلف یولو۶

[3] https://github.com/meituan/YOLOv6



شکل ۱۵ مقایسه دقت و سرعت یولو۷ با یولو۴، یولوx، یولو۴و۵

یولو ورژن هفتم

ورژن هفتم یولو برخلاف دو ورژن قبلی، علاوه بر ریپو در گیت[4] دارای مقاله مفصل (Chien-Yao Wang, Alexey Bochkovskiy et al, 2022) می باشد. این مدل به دقت 56.8% mAP دست یافته که از تمامی مدل های هم دوره خود بیشتر است. این مدل بیشتر بر روی بهبود سه مورد متمرکز شده است: تابع لاس، لیبل گذاری و روش های آموزش. قبل از اینکه به سراغ معماری مدل برویم دو تکنیک که در کارهای مشابه انجام شده است و مورد توجه مانیز می باشد را به طور مختصر توضیح می دهیم:

ری-پارامتریزاسیون مدل

این تیکنیک را می توان به عنوان یک روش آنسامبل در نظر گرفت که ماژول های مختلف در مرحله اینفرنس با یک دیگر ترکیب می شوند و به دو دسته سطح ماژول و سطح مدل تقسیم می شود. در سطح مدل می توان چندین مدل یکسان را روی دادگان متفاوت آموزش داد و سپس روی وزن های بدست آمده میانگین گیری کرد. در سطح ماژول –که امروزه بیشتر مورد توجه محققان است- مدل در فاز آمورش به چندین شاخه شکسته شده و در فاز اینفرنس دوباره به مدل معادل تجمیع می شود.

تغییر مقیاس (scaling) مدل

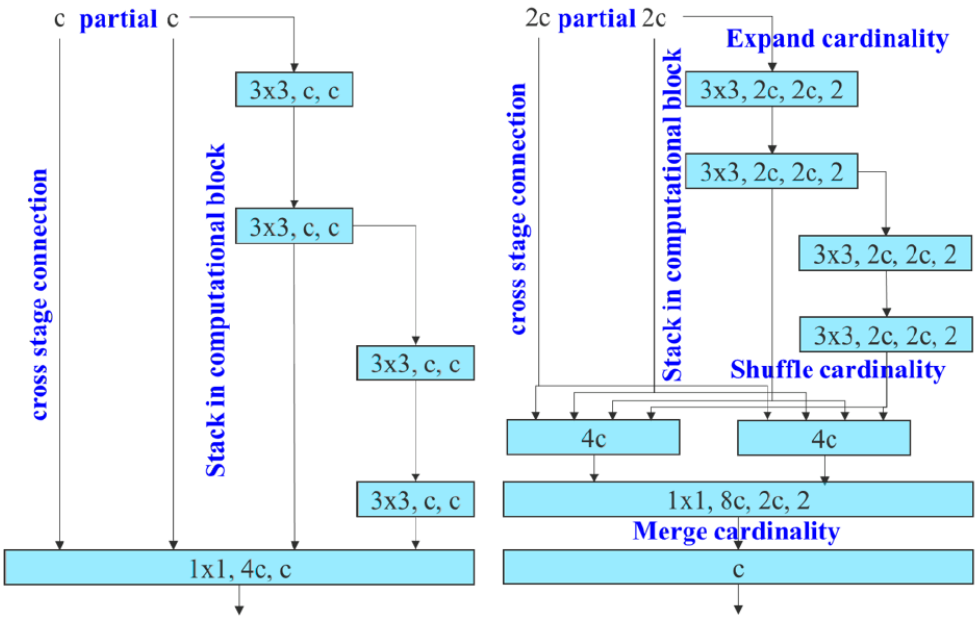
[4] https://github.com/WongKinYiu/yolov7

تغییر مقیاس مدل تلاشی برای برازش مدل با سخت افزارهای مختلف است و معمولا از پارامترهایی مانند ابعاد تصویر، عمق (تعداد لایه)، عرض (تعداد کانال) و مرحله (استیج: تعداد هرم ویژگی) استفاده می کند. یکی از روش های مرسوم تغییر مقیاس، جستجوی معماری شبکه (Neural Architecture search: NAS) می باشد.

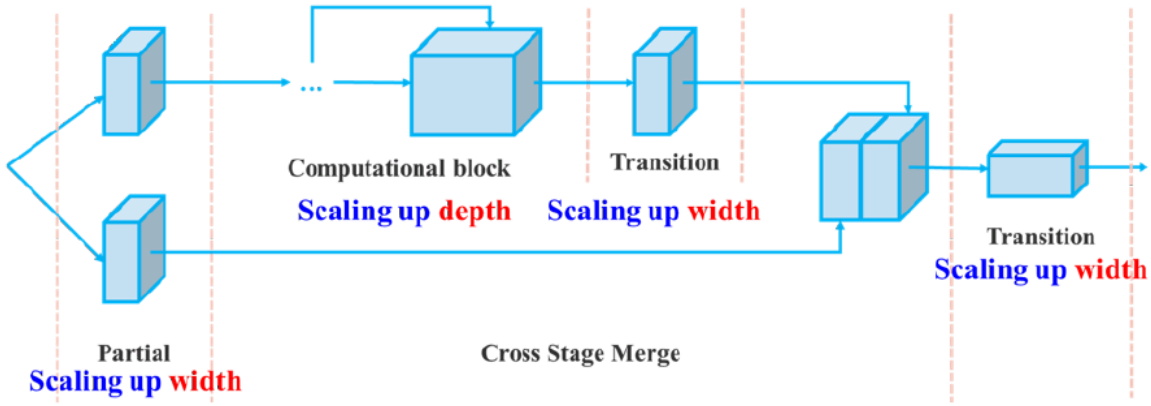
معماری یولو۷

برای بهبود معماری از E-ELAN: Extended Efficient Layer Aggregation Network استفاده شده است. این بلوک محاسباتی بر مبنای ELAN ساخته شده است. مقاله مربوط به ELAN هنوز ارائه نشده است! اما تمرکز آن روی بهبود دقت و سرعت با در نظر گرفتن عواملی هم چون: هزینه دسترسی حافظه، نسبت ورودی/خروجی، تعداد شاخه ها، اپراتور های element-wise و توجه به طولانی ترین و کوتاه ترین مسیر گرادیان، می باشد. با استفاده از گروه کانولوشنی، کانال ها و کاردینالیتی شبکه گسترش می یابد. در شکل ۱۶ ELAN و مدل توسعه یافته آن مشاهده می شود.

تکنیک تغییر مقیاس همانطور که اشاره شد برای دستیابی به سرعت و دقت مختلف متناسب به سخت افزار در دسترس می باشد. اما نکته مهم این است که شبکه های مبتنی بر concatenate نیاز به در نظر گرفتن عمق و عرض با هم دارند. بدین معنی که تغییر یکی بدون توجه به اثرات آن بر ابعاد منجر به بدتر شدن نتایج می شود. در شکل ۱۷ تغییر مقیاس بکار رفته با لحاظ این نکته مشاهده می شود.



شکل ۱۶ بلوک ELAN و گسترش یافته آن که در شبکه یولو۷ استفاده شده است



شکل ۱۷ تغییر مقیاس با توجه به این نکته که برای شبکه های مبتنی بر concatenate نیاز به در نظر گرفتن عمق و عرض به طور توامان می باشد.

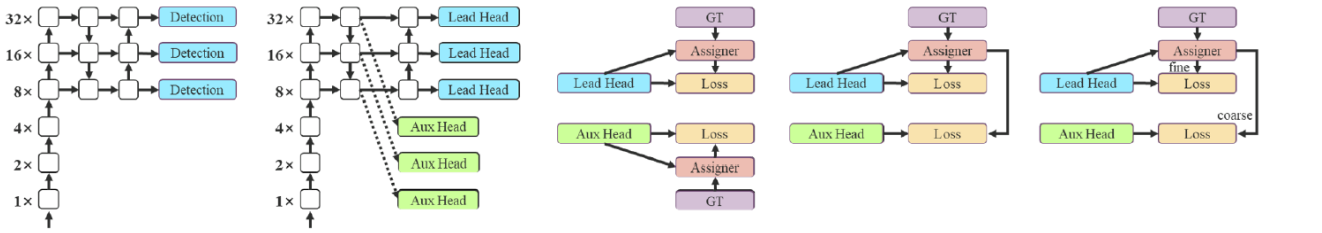
ری-پارامتریزاسیون برنامه ریزی شده برای کانولوشن

مدل RepConv بر روی VGG نتایج خوبی نشان داده است. این مدل با ترکیب کانکشن های ایدنتیکال و کانولوشن 1*1 و 3*3 در فاز آموزش و سپس حذف افزونه ها در فاز اینفرنس موجب افزایش دقت بدون کاهش سرعت می شود. اما برای مدل هایی مانند ResNet و DenseNet موجب کاهش شدید دقت می شود. نگارنده مقاله معتقد است وجود اتصال ایدنتیکال در RepConv به باقی مانده در RenNet و به کنارهمگذاری در DenseNet آسیب می زند و لذا یک مدل RepConvN که بدون اتصال ایدنتیکال می باشد ارائه کرده است.

سوپرویژن و تخصیص لیبل

تکنیک سوپرویژن عمیق برای آموزش شبکه ها به این صورت است که هد های کمکی اضافه به لایه های میانی –که ما آن را auxiliary head می نامیم- و شبکه کم عمق با لاس کمکی –که ما آن را lead head می نامیم- استفاده می کند. برای تخصیص لیبل نیز از رهیافت soft label و در نظر گرفتن پیش بینی شبکه به همراه حقیقت زمینه ای موجود (ترجمه ی ground truth!) استفاده می کند. مساله ای که مطرح می شود چگونگی تخصیص لیبل نرم به هد کمکی و هد رهبر می باشد که تا کنون در مقاله ای به آن پاسخ داده نشده است. در یولو۷ روش تخصیص لیبل به طوری است که هر دو هد کمکی و رهبر توسط پیشبینی هد رهبر هدایت می شوند. به بیان دیگر شبکه از پیش بینی هد رهبر به عنوان راهنمای تولید لیبل های سلسله مراتبی بزرگ –برای هد کمکی- به کوچک –برای هد رهبر- استفاده می کند. در شکل ۱۸ بلوک دیاگرام این روش مشاهده می شود.

تکنیک های دیگری نیز برای بهبود شبکه به کار رفته است مانند استفاده از نرمالیزاسیون بچ مستقیما متصل به شبکه کانولوشنی، ترکیب Implicit Knowledge از YOLOR با نقشه ویژگی کانولوشنی و استفاده از EMA - تکنیک mean teacher- برای اینفرنس نهایی مدل.



شکل ۱۸ استفاده از روش سوپرویژن عمیق و تخصیص لیبل از بزرگ به کوچک

نتایج آزمایشات

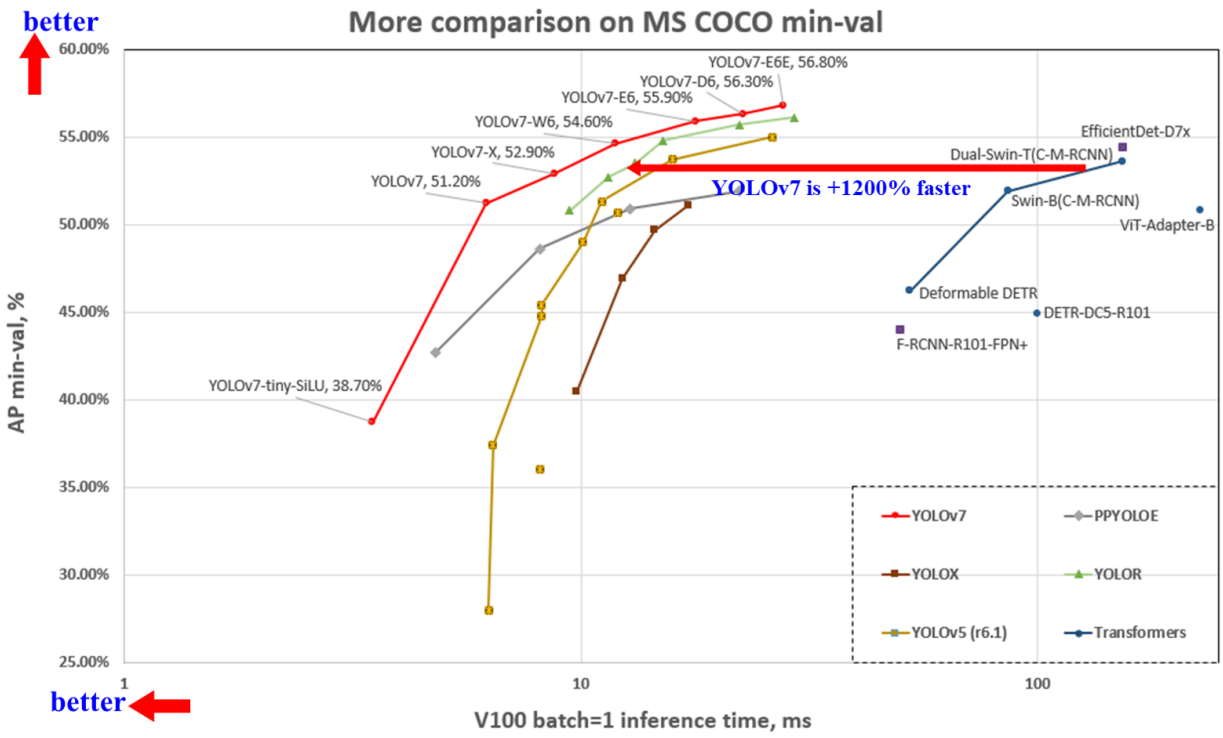
مدل پایه YOLOv7 برای GPU در سه حالت Edge با پسوند tiny، نرمال و فضای ابری با پسوند W6 ارائه شده است. مدل تغییر مقیاس یافته برای حالت نرمال با پسوند X– و برای فضای ابری با D6, E6– مشخص می شود. هم چنین برای E6 از روش E-ELAN نیز ا ستفاده شده ا ست که در آن حالت با نام YOLOv7-E6E م شخص می شـود. برای مدل لبه tiny از تابع فعالسـاز ReLu و برای دیگر حالات از SiLu اسـتفاده شـده اسـت. جدول مقایسـه با YOLOR به عنوان بیس، درجدول 5 و مقایسـه با YOLOX, YOLOv5, PPYOLO در جدول6 مشاهده می شود.

جدول5 مقایسه دقت، پارامترها و هزینه محاسباتی بین ورژن های گوناگون YOLOR و YOLOV7

Model	#Param.	FLOPs	Size	AP^{val}	AP^{val}_{50}	AP^{val}_{75}	AP^{val}_{S}	AP^{val}_{M}	AP^{val}_{L}
YOLOv4 [3]	64.4M	142.8G	640	49.7%	68.2%	54.3%	32.9%	54.8%	63.7%
YOLOR-u5 (r6.1) [81]	46.5M	109.1G	640	50.2%	68.7%	54.6%	33.2%	55.5%	63.7%
YOLOv4-CSP [79]	52.9M	120.4G	640	50.3%	68.6%	54.9%	34.2%	55.6%	65.1%
YOLOR-CSP [81]	52.9M	120.4G	640	50.8%	69.5%	55.3%	33.7%	56.0%	65.4%
YOLOv7	36.9M	104.7G	640	**51.2%**	**69.7%**	**55.5%**	**35.2%**	**56.0%**	**66.7%**
improvement	-43%	-15%	-	+0.4	+0.2	+0.2	+1.5	=	+1.3
YOLOR-CSP-X [81]	96.9M	226.8G	640	52.7%	**71.3%**	57.4%	36.3%	57.5%	68.3%
YOLOv7-X	71.3M	189.9G	640	**52.9%**	71.1%	**57.5%**	**36.9%**	**57.7%**	**68.6%**
improvement	-36%	-19%	-	+0.2	-0.2	+0.1	+0.6	+0.2	+0.3
YOLOv4-tiny [79]	6.1	6.9	416	24.9%	42.1%	25.7%	8.7%	28.4%	39.2%
YOLOv7-tiny	6.2	5.8	416	**35.2%**	**52.8%**	**37.3%**	**15.7%**	**38.0%**	**53.4%**
improvement	+2%	-19%	-	+10.3	+10.7	+11.6	+7.0	+9.6	+14.2
YOLOv4-tiny-3l [79]	8.7	5.2	320	30.8%	47.3%	32.2%	**10.9%**	31.9%	51.5%
YOLOv7-tiny	6.2	3.5	320	**30.8%**	**47.3%**	**32.2%**	10.0%	**31.9%**	**52.2%**
improvement	-39%	-49%	-	=	=	=	-0.9	=	+0.7
YOLOR-E6 [81]	115.8M	683.2G	1280	55.7%	73.2%	60.7%	40.1%	**60.4%**	69.2%
YOLOv7-E6	97.2M	515.2G	1280	**55.9%**	**73.5%**	**61.1%**	**40.6%**	60.3%	**70.0%**
improvement	-19%	-33%	-	+0.2	+0.3	+0.4	+0.5	-0.1	+0.8
YOLOR-D6 [81]	151.7M	935.6G	1280	56.1%	73.9%	61.2%	**42.4%**	60.5%	69.9%
YOLOv7-D6	154.7M	806.8G	1280	56.3%	73.8%	61.4%	41.3%	60.6%	70.1%
YOLOv7-E6E	151.7M	843.2G	1280	**56.8%**	**74.4%**	**62.1%**	40.8%	**62.1%**	**70.6%**
improvement	=	-11%	-	+0.7	+0.5	+0.9	-1.6	+1.6	+0.7

جدول 6 مقایسه ورژن های گوناگون YOLOX, PPYOLO, YOLOv5, YOLOR, YOLOv7

Model	#Param.	FLOPs	Size	FPS	AP^{test} / AP^{val}
YOLOX-S [21]	9.0M	26.8G	640	102	40.5% / 40.5%
YOLOX-M [21]	25.3M	73.8G	640	81	47.2% / 46.9%
YOLOX-L [21]	54.2M	155.6G	640	69	50.1% / 49.7%
YOLOX-X [21]	99.1M	281.9G	640	58	51.5% / 51.1%
PPYOLOE-S [85]	7.9M	17.4G	640	208	43.1% / 42.7%
PPYOLOE-M [85]	23.4M	49.9G	640	123	48.9% / 48.6%
PPYOLOE-L [85]	52.2M	110.1G	640	78	51.4% / 50.9%
PPYOLOE-X [85]	98.4M	206.6G	640	45	52.2% / 51.9%
YOLOv5-N (r6.1) [23]	1.9M	4.5G	640	159	- / 28.0%
YOLOv5-S (r6.1) [23]	7.2M	16.5G	640	156	- / 37.4%
YOLOv5-M (r6.1) [23]	21.2M	49.0G	640	122	- / 45.4%
YOLOv5-L (r6.1) [23]	46.5M	109.1G	640	99	- / 49.0%
YOLOv5-X (r6.1) [23]	86.7M	205.7G	640	83	- / 50.7%
YOLOR-CSP [81]	52.9M	120.4G	640	106	51.1% / 50.8%
YOLOR-CSP-X [81]	96.9M	226.8G	640	87	53.0% / 52.7%
YOLOv7-tiny-SiLU	6.2M	13.8G	640	286	38.7% / 38.7%
YOLOv7	36.9M	104.7G	640	161	51.4% / 51.2%
YOLOv7-X	71.3M	189.9G	640	114	53.1% / 52.9%
YOLOv5-N6 (r6.1) [23]	3.2M	18.4G	1280	123	- / 36.0%
YOLOv5-S6 (r6.1) [23]	12.6M	67.2G	1280	122	- / 44.8%
YOLOv5-M6 (r6.1) [23]	35.7M	200.0G	1280	90	- / 51.3%
YOLOv5-L6 (r6.1) [23]	76.8M	445.6G	1280	63	- / 53.7%
YOLOv5-X6 (r6.1) [23]	140.7M	839.2G	1280	38	- / 55.0%
YOLOR-P6 [81]	37.2M	325.6G	1280	76	53.9% / 53.5%
YOLOR-W6 [81]	79.8G	453.2G	1280	66	55.2% / 54.8%
YOLOR-E6 [81]	115.8M	683.2G	1280	45	55.8% / 55.7%
YOLOR-D6 [81]	151.7M	935.6G	1280	34	56.5% / 56.1%
YOLOv7-W6	70.4M	360.0G	1280	84	54.9% / 54.6%
YOLOv7-E6	97.2M	515.2G	1280	56	56.0% / 55.9%
YOLOv7-D6	154.7M	806.8G	1280	44	56.6% / 56.3%
YOLOv7-E6E	151.7M	843.2G	1280	36	56.8% / 56.8%



شکل ۱۹ مقایسه YOLOv7 با YOLOR, PPYOLO, YOLOv5 و ترنسفورمر ها

فهرست منابع

Bochkovskiy, A. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection.

Chien-Yao Wang, Alexey Bochkovskiy et al. (2022). YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors.

Joseph Redmon, Ali Farhadi. (2016). YOLO9000: better, faster, stronger.

Joseph Redmon, Ali Farhadi. (2018). YOLOv3: an incremental improvment.

Redmon, J. (2015). you only look once: Unified, Real-Time Object Detection.

Thuan, D. (2021). *Evolution of YOLO algorithm and YOLOv5: the state of the art object detection algorithm.*