YOLOV7

برای آموزش اولیه شبکه yolov7 از لینک آموزشی که همراه با تعریف پروژه برای من ارسال شده بود استفاده کردم.

https://www.youtube.com/watch?v=bgAUHS1Adzo&list=PL4sqgbpSjuJjZz_YyNx5e2jPrR9kun2l8&index=10

معیار ارزیابی:

برای مقایسه‌ی آموزش‌های مختلف از mAP استفاده می‌شود. در لینک github زیر کدهای برای محاسبه mAP قرار داده شده است. نیازمندی اجرای این کد سه مورد می‌باشد، تصاویر تست، فایل txt که شامل object های موجود در تصویر است و در نهایت فایل txt که شامل objectهای تشخیص داده شده توسط شبکه است.

https://github.com/Cartucho/mAP

مجموعه داده:

برای آموزش از همان مجموعه داده های گفته شده در ویدیو استفاده شد که لینک آن در زیر قرار داده شده است.

https://www.kaggle.com/datasets/andrewmvd/car-plate-detection

این مجموعه داده‌ها شامل 433 تصویر با ماسک پلاک‌های موجود در تصویر است. پلاک‌های این مجموعه داده شامل پلاک ملی ایران نمی باشد.

https://www.kaggle.com/datasets/skhalili/iraniancarnumberplate

دیگر مجموعه داده شامل 444 تصویر که شامل پلاک‌های ملی ایران می‌باشد همراه با ماسک پلاک‌های موجود در تصویر است.

این تصاویر تحت augmentation زیر قرار گرفته‌اند.

- ✓ Flip Horizontal
- ✓ shear

در مجموعه 1529 تصویر برای train، 148 تصویر برای validation و 70 تصویر برای test در نظر گرفته شده است.

روش‌های آموزش:

Transfer learning یا یادگیری انتقالی تکنیکی در یادگیری ماشین است که در آن از یک مدل از پیش آموزش دیده که بر روی یک مجموعه داده بزرگ آموزش داده شده به عنوان نقطه شروع برای حل یک مساله جدید یا مشابه استفاده می‌شود. این رویکرد به ویژه در مواردی می‌تواند مفید باشد که مجموعه داده هدف کوچک یا شبیه به مجموعه داده از پیش آموزش دیده باشد.

برای بهبود عملکرد آموزش در مدل yolov7 با استفاده از یادگیری انتقالی می‌توان آخرین لایه‌های head را تغییر داد تا مدل را با مساله خاص خود تطبیق دهید. درادامه چند پیشنهاد وجود دارد:

تغییر تعداد کلاس‌ها: مدل yolov7 در حال حاضر در 80 کلاس آموزش دیده است. برای مساله تشخیص شی متفاوت با تعداد کلاس‌های متفاوت، می‌توان تعداد کانال‌های خروجی در آخرین لایه را برای مطابقت با مساله با تعداد کلاس خاص تغییر داد.

آموزش اول:

اولین روش آموزش را، بر اساس تغییر کلاس‌ها انجام می‌دهیم برای اینکار تعداد کلاس‌ها در فایل yolov7.yaml را که در مسیر yolov7\cfg\training قرار دارد براساس مساله خود تغییر می‌دهیم. همچنین اینکه در فایل bata.yaml نیز باید تعداد کلاس‌ها را به یک تغییر داد.

nc: 1 # number of classes

این آموزش بر اساس وزنی که از مجموعه داده coco موجود است مورد آموزش قرار گرفته است همانطور که در بالا گفته شد این نیز یه نوع Transfer learning است.

```
!python train.py --batch 16 --cfg cfg/training/yolov7.yaml --
epochs 55 --data data.yaml --
weights 'runs/train/exp5/weights/best.pt' --device 0
```

mAP به دست آمده از آموزش مدل در این روش بر روی داده‌های تست برابر 96.02٪ می‌باشد.

آموزش دوم:

Fune-tune کردن لایه‌های backbone:

Backbone در یک شبکه عصبی به بخشی از شبکه اشاره دارد که داده‌های ورودی را پردازش کرده و ویژگی‌هایی از آن را استخراج می‌کند. Backbone مدل yolov7 بر اساس DarkNet-53 است که بر روی مجموعه داده

ImageNet آموزش داده شده است. وقتی از Backbone یک مدل از پیش آموزش داده استفاده می‌کنید، می‌توان وزن‌های لایه‌های Backbone را در طول آموزش اولیه مدل yolov7، freeze کرد این کار می‌تواند به حفظ ویژگی‌های آموخته شده در Backbone کمک کند و در عین حال به لایه‌های yolov7 اجازه می‌دهد تا اشیا خاص مساله جدید را شناسایی کند.

هنگامی که لایه‌های yolov7 آموزش داده شدند، می‌توان با باز کردن لایه‌های Backbone و ادامه آموزش مدل بر روی مجموعه داده خود، Backbone را به خوبی تنظیم کنید. این می‌تواند به Backbone کمک کنه تا ویژگی‌های پیچیده‌تری را که مختص مساله است را شناسایی کند.

برای آموزش بر اساس freeze کردن لایه‌های Backbone دوبار مجموعه داده را مورد آموزش قرار می‌دهیم، از آنجایی که لایه‌های Backbone در مدل yolov7، 50 لایه می‌باشد ابتدا 50 لایه اول را freeze می‌کنیم.

```
!python train.py --batch 16 --cfg cfg/training/yolov7.yaml --
epochs 55 --data data.yaml --weights 'yolov7.pt' --device 0 --
freeze 50
```

سپس از بهترین وزن به دست آمده از مرحله‌ی قبل استفاده کرده و با باز کردن همه‌ی لایه‌ها به آموزش مجدد می‌پردازیم.

```
!python train.py --batch 16 --cfg cfg/training/yolov7.yaml --
epochs 55 --data /content/gdrive/MyDrive/yolov7/ANPR_ir-
1/data.yaml --weights 'runs/train/exp2/weights/best.pt' --
device 0
```

mAP به دست آمده از آموزش مدل در این روش بر روی داده‌های تست برابر 88.83٪ می‌باشد. اگر چه تصور می‌رفت این مدل نتایج بهتری نسبت به مدل اول داشته باشد ولی اینگونه نبود. از آنجایی که این مدل دوبار و هر بار در 55 epochs مورد آموزش قرار گرفت احتمال میرود نتایج ضعیف آن به دلیل overfit شدن به مجموعه-داده آموزشی باشد.

آموزش سوم:

Fine-tune کردن anchor boxes:

anchor boxes های مورد استفاده در مدل yolov7 برای تشخیص اشیا در طیف خاصی از اندازه‌ها و نسبت ابعاد بهینه شده است. اگر مساله تشخیص شی اشیایی است که به طور قابل توجهی بزرگتر یا کوچکتر از اشیا موجود در مجموعه داده coco هستند، بهتر است برای بهبود دقت تشخیص، anchor boxes را بر اساس مساله

تنظیم کرد. در لینک زیر توضیحاتی برای پیدا کردن anchor boxes مناسب مساله آورده شده است. به دلیل کمبود وقت نتوانستم این مساله را بررسی کنم.

https://towardsdatascience.com/training-yolo-select-anchor-boxes-like-this-3226cb8d7f0b

از طرفی در لینک زیر گفته شده است که برای yolov5 و yolov7 پارمتری در آموزش به نام autoanchor وجود دارد که که anchorها را براساس مجموعه داده به روز رسانی می‌کند که به صورت پیش فرض فعال است اگر بخواهیم آن را غیر فعال کنیم پارمتر noautoanchor-- را در آموزش تنظیم می‌کنیم. این آموزش بر اساس تنظیم کردن این پارامتر انجام شده است.

https://github.com/ultralytics/yolov5/issues/2092

```
!python train.py --batch 16 --cfg cfg/training/yolov7.yaml --
epochs 55 --data /content/gdrive/MyDrive/yolov7/ANPR_ir-
1/data.yaml --weights 'yolov7.pt' --device 0 --noautoanchor
```

mAP به دست آمده از آموزش مدل در این روش بر روی داده‌های تست برابر 91.64٪ می‌باشد همانطور که انتظار می‌رفت در مقایسه با مدل دوم که مشابه این مدل ولی با اعمال autoanchor بوده است نتایج ضعیف‌تری به دست آمده است.

استفاده از یک optimizer متفاوت:

مدل yolov7 از بهینه‌ساز Adam استفاده می‌کند. براساس مساله‌ی خاص می‌توان از بهینه‌سازهای مختلفی مثل SGD و RMSprop برای بهبود عملکرد مدل استفاده کرد.

نرخ یادگیری:

می‌توان با نرخ‌های یادگیری اولیه متفاوتی مانند 0.001، 0.005 یا 0.1 آزمایش و بررسی کرد کدام یک برای مساله plate بهتر عمل می‌کند.

استفاده از مقادیر مختلف Momentum:

SGD momentum یا Adam beta1 می‌تواند بر سرعت همگرایی و دقت نهایی مدل تاثیر بگذارد. می‌توانید مقادیر مختلف momentum مانند 0.9، 0.95 یا 0.99 را امتحان کرد و بررسی کرد کدام یک برای مساله plate بهتر عمل می‌کند.

استفاده از مقادیر مختلف Weight Decay:

Weight Decay می‌تواند به جلوگیری از overfit مدل کمک کند. می‌توان با مقادیر مختلف Weight Decay مانند 0.0001، 0.001 یا 0.01 آزمایش کرد و بررسی کرد کدام یک برای مساله plate بهتر عمل می‌کند.

Batch Size

Batch Size می‌تواند بر سرعت همگرایی و استفاده از حافظه فرآیند آموزش تأثیر بگذارد. می‌توان با اندازه‌های مختلف Batch مانند 8، 32 یا 64 آزمایش کنید و بررسی کرد کدام یک برای مساله plate بهتر عمل می‌کند.

به طور کلی، می‌توان با ترکیب مختلف این گزینه‌ها عملکرد بهتری برای مدل به دست آورد. که در اینجا به بررسی چند مورد پرداختیم.

می‌توان پارامترهای گفته شده در بالا را تغییر داد تا به بهترین نتیجه برای مساله خود رسید. به دلیل کمبود وقت این مسال را بررسی نکردم.

Number of ground-truth objects per class در این مجموعه داده 73 می‌باشد.

ردیف	روش آموزش	epochs	mAp	Number of detected objects per class
1	روش اول	55	**96.02%**	82 (tp:71, fp:11)
2	روش دوم	110	88.83%	80 (tp:66, fp:14)
3	روش سوم	55	91.64%	75 (tp:67, fp:8)