

تمرین سری سوم - DL

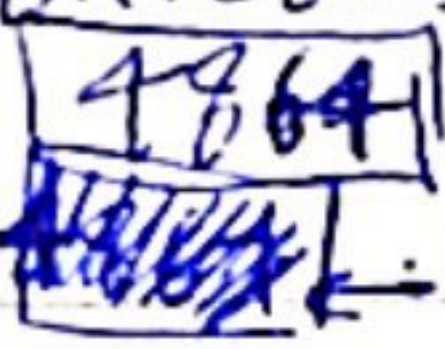
پرینال طاهری - 99106352

سؤال 1

1) $K \times K \times M \times N + N$: مقدار پارامترها

$H \times W \times K \times K \times M \times N$: هزینه محاسباتی

Padding : stride = 2 Img : $128 \times 128 \times 3$ (ب)

$n\text{-Param} = K \times K \times M \times N + N = 5 \times 5 \times 3 \times 64 + 64 = 4864$ (1) 

ابعاد خروجی : $\frac{\text{اندازه لبه} + 2 \times \text{پدینگ} - \text{استراید}}{2} + 1$

$$= \frac{128 - 5 + 2 \times 2}{2} + 1 = 64 = W_{out} = H_{out}$$

Cost : $N \times M \times K \times K \times H_{out} \times W_{out}$

$$= 64 \times 3 \times 5 \times 5 \times 64 \times 64 = 19660800$$

$n\text{-Param} = 5 \times 5 \times 64 \times 128 + 128 = 204928$ (2)

$$\text{dim} = \frac{64 - 5 + 2 \times 2}{2} + 1 = 32$$

$$\text{Cost} = 32 \times 32 \times 5 \times 5 \times 64 \times 128 = 209715200$$

$\text{Param} = 5 \times 5 \times 128 \times 256 + 256 = 819456$ (3)

$$\text{dim} = \frac{32 - 5 + 2 \times 2}{2} + 1 = 16$$

$$\text{Cost} = 16 \times 16 \times 5 \times 5 \times 128 \times 256 = 209715200$$

Layer	dim-out	n-Param	Comp. Cost
1	64x64x64	4,800 4,800	19,660,800
2	32x32x128	2,4928	20,715,200
3	16x16x256	819,456	20,715,200

ب - 2: Receptive Field

layer 1 \rightarrow R.F = 5x5

layer 2 \rightarrow stride = 2 \rightarrow R.F = (5-1)x2+5=13

layer 3 \rightarrow 13 + (5-1)x2 = 21

$$DSC: n_param = F \times F \times C_{in} + C_{in} + C_{in} \times C_{out} \quad (2)$$

$$Comp. Cost = F \times F \times C_{in} \times H_{out} \times W_{out} + C_{in} \times C_{out} \times H_{out} \times W_{out} + C_{out}$$

ممکن است تعداد پارامترهای DSC بیشتر از حالت استاندارد شود مخصوصاً

با تعداد کانال کم اما به صورت کلی محاسبات آن از حالت استاندارد

سریعتر است.

layer 1 \rightarrow Param = 4800, Cost = 19,660,800

layer 2 \rightarrow Param = 5x5x64+64+64x128+128

\rightarrow Param = 9792

Cost : 5x5x64x32x32+64x128x32x32

\rightarrow Cost = 10,027,008

layer 3 →

$$\text{Param} = 5 \times 5 \times 128 + 128 + 128 \times 256 + 256 = \boxed{36,352}$$

$$\text{Gst} = 5 \times 5 \times 128 \times 16 \times 16 + 128 \times 256 \times 16 \times 16 =$$

$$\rightarrow \text{Gst} = \boxed{9,207,808}$$

Layers	Param 1	Param 2	Gst 1	Gst 2
1	4,864	4864	19,660,800	~
2	204,928	9792	209,715,200	10,927,008
3	819,456	36,352	209,715,200	9,207,808

(5)

مقادیر الف :

$$\text{فرم} : 16 \times 16 \times 256 = 65,536$$

$$\rightarrow \text{FC params} = 65,536 \times 200 + 200 = \boxed{13,107,400}$$

$$\text{Total param} = 4,864 + 204,928 + 819,456 + 13,107,400$$

$$= \boxed{13,924,948} \rightarrow \text{FC} \approx 94\%$$

مقادیر ب :

$$\text{FC params} = 13,107,400$$

$$\text{Total param} = 4864 + 9792 + 36,352 + 13,107,400$$

$$= \boxed{13,124,924}$$

← تستر هم باید برای (FC) است - به جای flattening و فلتینگ از Global Avg. Pooling استفاده کنید و هر Feature map را به یک مقدار تبدیل کنید

← 256 مقادیر به FC داده می شود / در حالت دیگر می توان از bottleneck استفاده کرد.

سوال ②

الف)

ResNet با اضافه کردن حفره‌های لایه قبلی به صورت مستقیم به لایه بعدی باعث می‌شود

مسئله $\text{gradient vanishing}$ این بود زیرا در مرحله backpropagation یک

مستقیم به لایه قبل دارد.

Dense Net با الهام از ResNet ساخته شده با این تفاوت که ورودی

به هر لایه از تمام لایه‌های قبل از فرستادن است و به عبارتی اینکه لایه‌های قبل را

با لایه بعدی جمع کند با آکس Concatenate می‌کند. این کار باعث می‌شود

از feature map های لایه قبل استفاده کند و مقدار را بیشتر می‌کند.

اما $\text{Computational Cost}$ آن بیشتر است.

ب)

مسئله $\text{vanishing gradient}$ به این دلیل رخ می‌دهد که در مرحله backpropagation

مقدار گرادیان بسیار کم می‌شود و باعث می‌شود که مدل نتواند به خوبی

یاد بگیرد.

به دلیل اینکه در Dense Net به هر feature map های لایه‌های قبل

دسترسی وجود دارد، گرادیان راحت‌تر می‌تواند به لایه‌های قبل برسد.

مذکور می‌شود که این است که به دلیل اینکه هر لایه به تمام feature map های

لایه‌های قبل دسترسی دارد، مقدار زیادی از ویژگی‌ها را به اشتراک می‌دهد

و این باعث می‌شود که هر لایه همه ویژگی‌ها را از اول یاد بگیرد.

ج)

در شبکه‌هایی که نیاز به لایه‌های زیاد و عمیق است مورد استفاده است. برای

مثال در دسته‌بندی ImageNet که نیاز به لایه‌های زیاد است.

و برای هر Modality از یک Feature Extractor مقادیر استخراج می‌کنیم
سپس Feature ها را ترکیب کرده و از لایه‌های FC رد می‌کنیم.

1- Feature Extractor:

برای تصویر می‌توانیم از DenseNet استفاده کنیم زیرا به خوبی می‌تواند ویژگی‌ها را
تجزیه و استخراج کند.

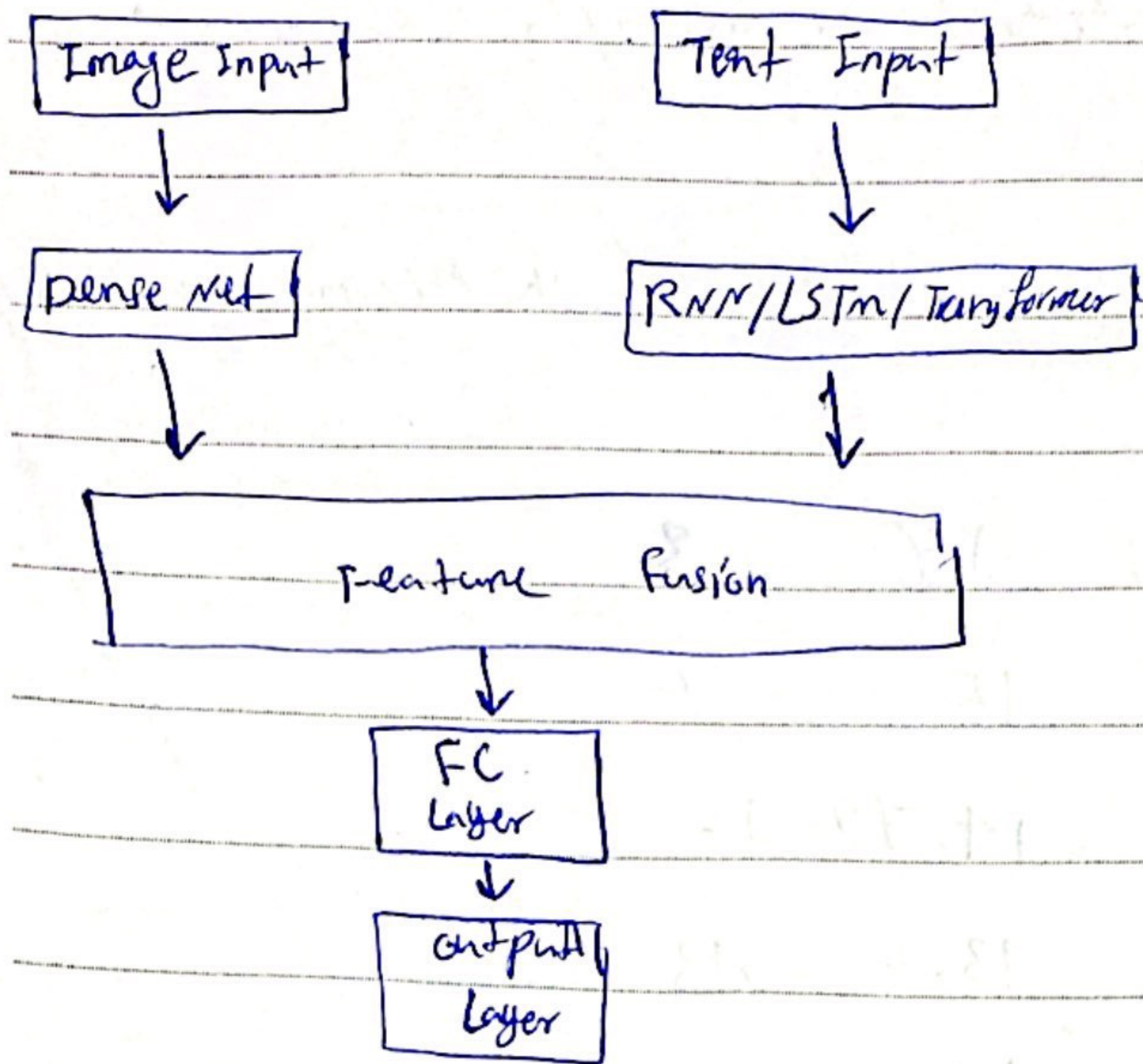
برای متن از ساختارهایی مانند RNN یا Transformer استفاده می‌کنیم زیرا در موقعیت
متن نیاز است که مفاهیم را استخراج کند و وابستگی‌های متوالی را بفهمد. از
LSTM یا GPT می‌توانیم استفاده کنیم.

2- Feature Fusion:

برای ترکیب ویژگی‌ها می‌توانیم به صورت ساده آن‌ها را Concatenate کنیم یا
از attention mechanism استفاده کنیم که اهمیت هر ویژگی را تفهیم و وزن
مقادیر به هر کدام بدهد.

3- Classification:

سپس از ترکیب ویژگی‌ها، آن‌ها را از یک یا چند لایه FC رد می‌کنیم و در لایه آخر
بابت به نتایج (Classification و Regression) یک لایه خروجی قرار
می‌دهیم.



سوال 3

الف) U-Net از دو بخش Encoder و Decoder تشکیل شده است. Encoder، ویژگی‌های high-level را تقویر، استخراج و نیاز به downsampling در حالی که Decoder تقویر را بازسازی می‌کند (upsampling). در این میان ویژگی‌های اصلی زوایا و خطوط از این ورودی به دلیل downsampling. Skip Connection باعث شود spatial resolution بین encoder، decoder یکسان بماند و کیفیت تقویر بازسازی شده را بهتر کند. در این حالت decoder از لایه‌های encoder و عددی برابر از feature map که encoder یاد گرفته استفاده می‌کند. با ابعادها کا هفت برابر کند. در حقیقت backpropagation نیز به دلیل ارتباط مستقیم بین انکودر و دیکودر، به دلیل بهتر یاد می‌گیرد.

(ب)

Random Deformation به صورت کلی به تغییرات غیر خطی و تصادفی در داده ها

ورود اشاره دربر داشته

Elastic Deformation، در این حالت تغییرات محلی در نرم به صورتی اعمال می شود که به کشیدگی است و می تواند زاویه دید مختلف را به سازه بدهد.

Affine transformation: مانند مدوش و اسکیل کردن

در تغییرات دید مانند اضافه کردن نویز یا مسک کردن بخشی از تصویر.

این کار باعث افزایش ریسک و منجر به overfitting می شود.

همچنین ایجاد این تغییرات باعث می شود مدل به تغییرات کوچکی حساس نباشد و نتایج را به درون و بیرون های اصلی ندارد.

(ج) $\text{Input} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$ $\text{filter} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$

$\text{output size} = \text{input size} + (\text{filter size} - 1)$

$= 2 + 2 - 1 = 3$

~~$\begin{bmatrix} 1 & 2 & 3 \\ 3 & 4 & 5 \\ \dots & \dots & \dots \end{bmatrix} + \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} = \begin{bmatrix} 1 & 4 & 4 \\ 6 & 20 & 16 \\ 9 & 24 & 16 \end{bmatrix}$~~

$\begin{bmatrix} 1 & 2 & 3 \\ 3 & 4 & 5 \\ \dots & \dots & \dots \end{bmatrix} + \begin{bmatrix} 0 & 2 & 1 \\ 0 & 6 & 8 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 3 & 6 & 0 \\ 2 & 12 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 4 & 6 \\ 0 & 12 & 16 \end{bmatrix}$

$= \begin{bmatrix} 1 & 4 & 4 \\ 6 & 20 & 16 \\ 9 & 24 & 16 \end{bmatrix}$

Yolo 1:

$$B=2, C=80$$

الف)

$$\rightarrow \text{depth} = B \times 5 + C = 90$$

هر bounding box دارای 4 ویژگی و اسکال (x, y, h, w) اسکال + confidence score

Yolo 3:

$$B=3, C=80$$

$$\rightarrow \text{depth} = B \times (C+5) = 255$$

ولای تقاربت این در این اسکال که در Yolo 3 مقدار bounding box

به ازای هر cell امتزاس پیدا کرده است همچنین به دلیل multiscale prediction

در Yolo 3 که در 3 رزولوشن دیتکت می کند عمق مدل بیشتر شده است.

ب)

Yolo 3 به جای استفاده از Softmax که فرض می کند هر شیء دقیقاً

به یک کلاس تعلق دارد از sigmoid استفاده می کند و به هر کلاس یک

احتمال نسبت می دهد. همچنین در Yolo 3 Categorical Cross-Entropy

و Binary Cross-Entropy تعریف شده است.

همچنین در Yolo 3 برای objectness score اسکال که به هر شیء به اسکال

هر شیء متعلق به یک کلاسی است ~~می~~ ^{برای} می کند که آیا داخل bounding box

هم شیء وجود دارد یا خیر.

ج) برای جلوگیری از تشخیص تکراری و حذف آن از الگوریتمی به اسم $non-maximum suppression$ استفاده شده است که تعیین می‌کند نتایج بالایی برای هر شیء، فقط شود. فرایند NMS به این صورت است که ابتدا به هر $bounding box$ یک $confidence score$ نسبت داده می‌شود و آن‌ها را می‌کند و در از بین نتایج به‌ترتیب می‌بندد انتخاب می‌شوند پس آن‌ها را حذف می‌کند و IoU که همپوشانی هر کدام است را می‌سازد می‌کند و اگر همپوشانی از مقدار $threshold$ بود آن‌ها را حذف می‌کند.

د) این رویکرد با استفاده از $Fully Convolutional Network$ (FCN) $upscaling$ باعث می‌شود بتواند در مقیاس‌های مختلف تصویر را بازسازی کند. در نهایت $Anchor box$ های از پیش تعیین شده در این دو مدل ترکیب می‌کند تا شبکه مستقل از اندازه تصویر، مکان، ابعاد اشیاء را تشخیص دهد.

این ایده به این صورت پیاده‌سازی شده است که در زمان آموزش، اندازه داده‌های ورودی پس از هر $iteration$ به صورت تصادفی تغییر داده می‌شود و به تغییر اندازه تصویر، $grid cell$ ها نیز تغییر می‌کند.

به این دلیل سودمند است که اگر مدل بتواند تصویر با ابعاد مختلف را در ورودی بگیرد، همپوشانی با نتایج به‌شمارش می‌تواند اندازه تصویر را تعیین دهد؛ برای سرعت بیشتر تصویر را کوچک‌تر در زمان وقت بیشتر تصویر را بزرگ‌تر می‌کند.

(5)

در Yolov1 شبکه به صورت مستقیم حقیقات bounding box را پس می
دهد و مقدار موقعیت ها را در یک محدودیت و در نظر گرفته به فیل موقعیت ها
ابعاد است. این شبکه های گسترده
همچنین پس بینی زمان را درست و تقریباً گرفته می شود و مرکز باکس دقیقاً داخل
cell از آن مربوط به آن می قرار می گیرد.
راه حل:

در yolov2، Anchor box معرّفی شد که به جای پس بینی دقیق
ابعاد حقیقات، ابعاد و موقعیت های باکس ها که یک سری باکس از
پس بینی شده اند را تقسیم می کند.
همچنین به جای انجم یک به پس بینی مرکز باکس، از ~~استفاده~~ استفاده
شده که حقیقات مرکز باکس نیست آید.

(6)

1- استفاده از معیار جدید (Darknet-53) در پس بینی که در نتیجه قوی از 19-Darknet
استفاده شد. این معیار از 53 لایه کانولوشنی به جای 19 لایه
استفاده می کند و همچنین 111 از Resnet از ~~Resnet~~ Residual Connection
استفاده می کند.

2 multi-scale prediction

بدولت سیستم ها قبل از تمییز استفاده می کند. خروجی با وضوح بالا برابر با سایر
وضوح متوسّط و متوسّط را - وضوح پایین و بی بزرگ

3- استفاده از sigmoid و softmax + استفاده از Logistic Regression
Subo