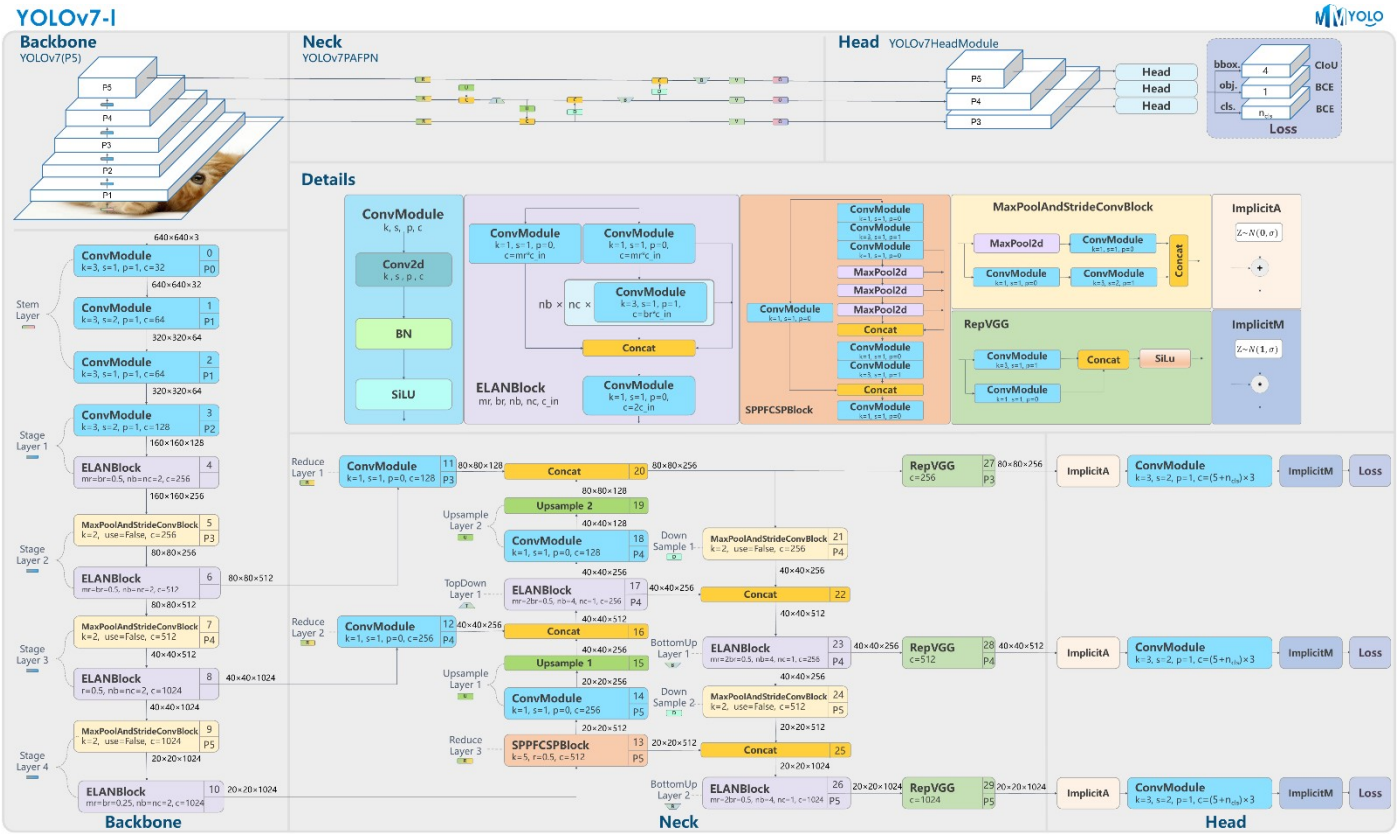
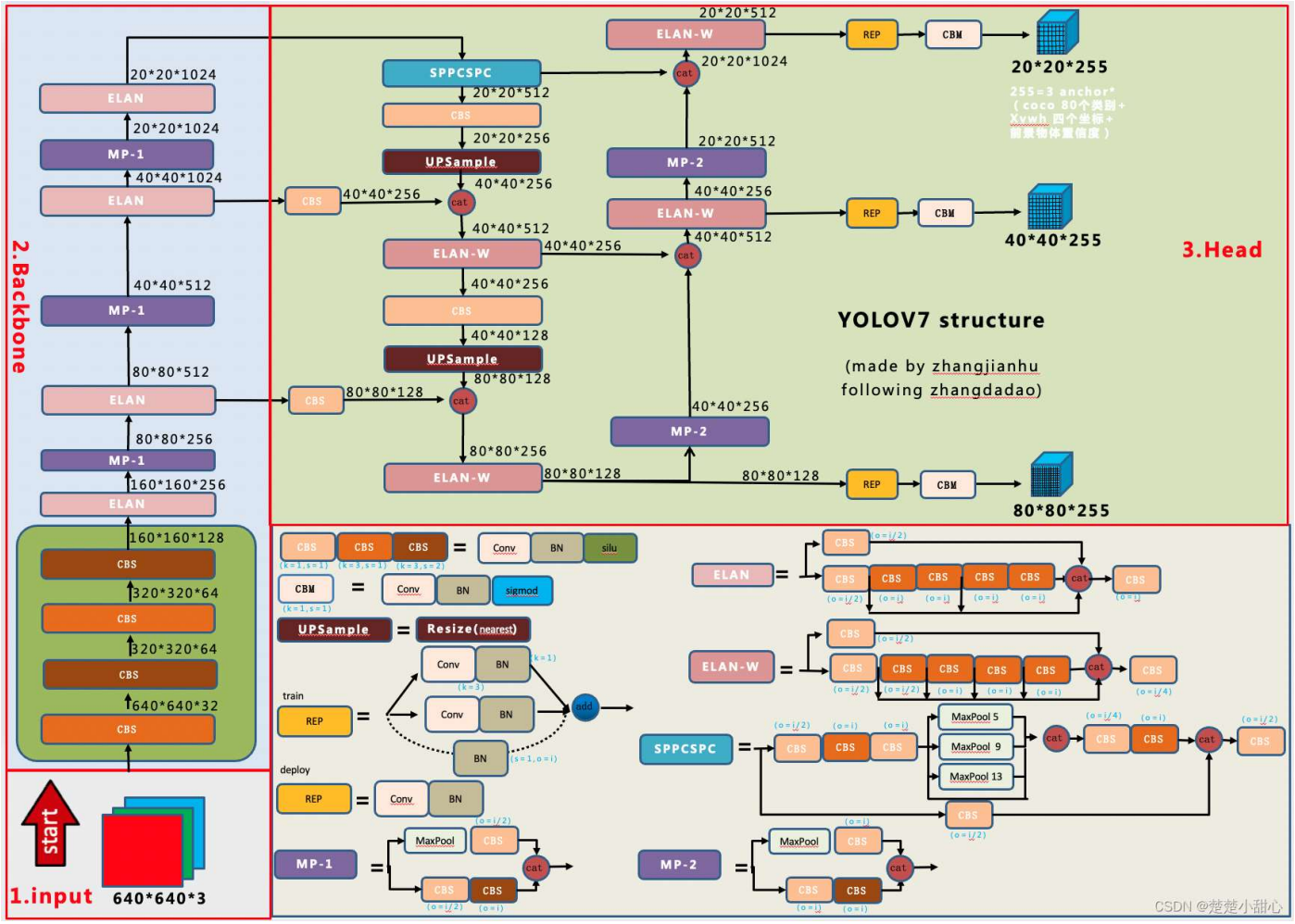
YOLOv7


YOLOv7-l
Backbone
YOLOv7(P5)
Neck
YOLOv7PAFPN
Head
YOLOv7HeadModule
Loss
Details
ConvModule
Conv2d
BN
SiLU
ELANBlock
SPPFCSPBlock
MaxPoolAndStrideConvBlock
RepVGG
ImplicitA
ImplicitM
Backbone
Neck
Head


2.Backbone
3.Head
YOLOV7 structure
(made by zhangjianhu
following zhangdadao)
SPPCSPC
ELAN-W
MP-1
MP-2
UPSample
20*20*255
40*40*255
80*80*255
1.input
640*640*3
start
CSDN @楚楚小甜心

From YOLOv7 authors (same as YOLOv4!):

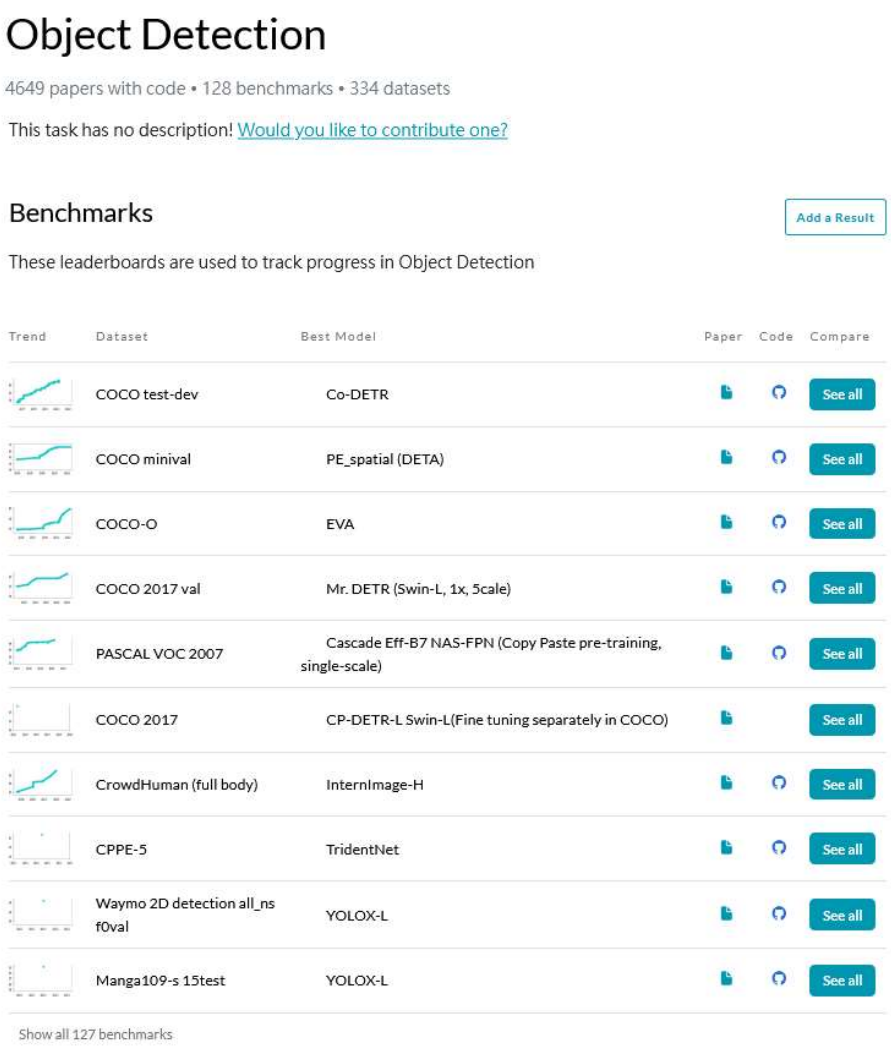
Being able to become a state-of-the-art (SOTA) real-time object detector usually requires the following characteristics:

1. a faster and stronger network architecture; (**backbone**)
2. a more effective feature integration method; (**neck**)
3. a more accurate detection method; (**head**)
4. a more robust **loss function**;
5. a more efficient **label assignment method**; and
6. a more efficient **training method**.

YOLOv7 focuses on improvements in 4, 5, 6.

What is State of the Art (SOTA)?

State of the Art (SOTA) 就是指目前技術的最優算法，並以此為衡量標準 (search "Browse State-of-the-Art" and you can find the SOTA in each field)



Object Detection

4649 papers with code • 128 benchmarks • 334 datasets

This task has no description! Would you like to contribute one?

Benchmarks

Add a Result

These leaderboards are used to track progress in Object Detection

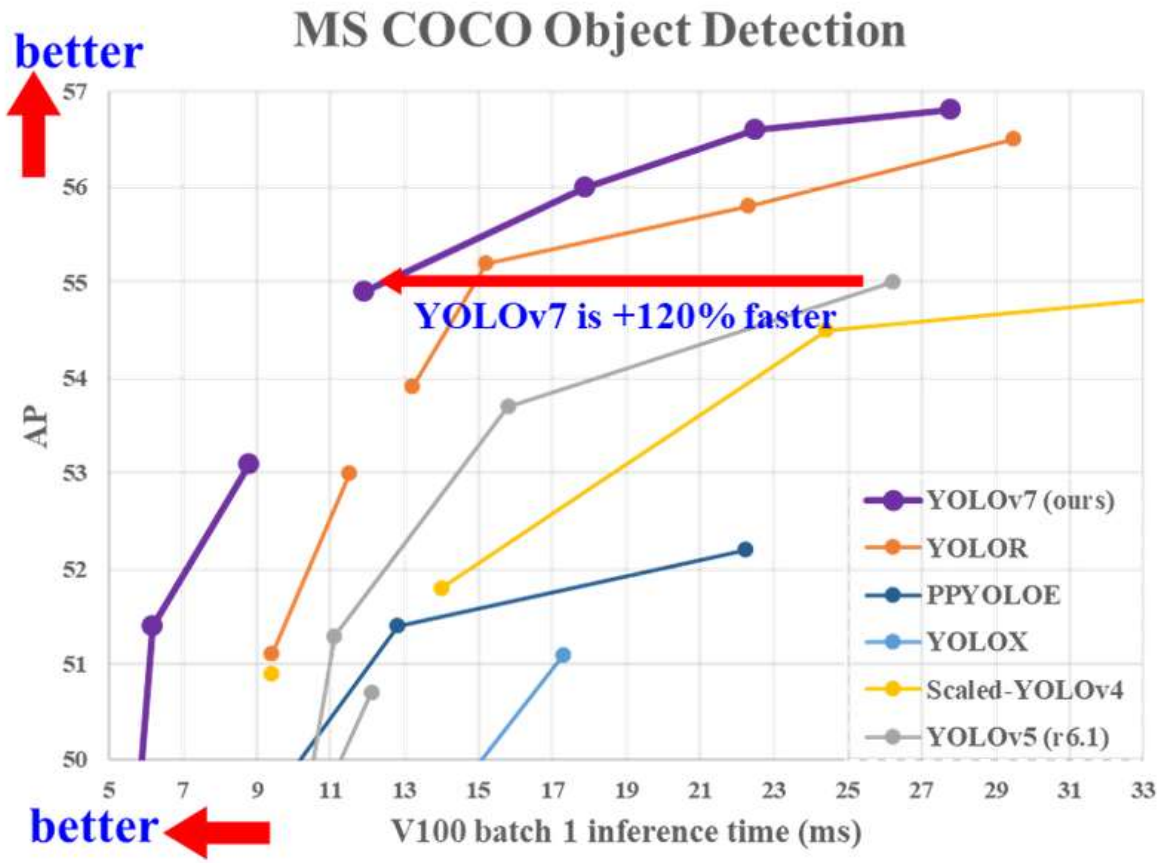
Trend	Dataset	Best Model	Paper	Code	Compare
	COCO test-dev	Co-DETR			See all
	COCO minival	PE_spatial (DETA)			See all
	COCO-O	EVA			See all
	COCO 2017 val	Mr. DETR (Swin-L, 1x, 5cale)			See all
	PASCAL VOC 2007	Cascade Eff-B7 NAS-FPN (Copy Paste pre-training, single-scale)			See all
	COCO 2017	CP-DETR-L Swin-L(Fine tuning separately in COCO)			See all
	CrowdHuman (full body)	InternImage-H			See all
	CPPE-5	TridentNet			See all
	Waymo 2D detection all_ns f0val	YOLOX-L			See all
	Manga109-s 15test	YOLOX-L			See all

Show all 127 benchmarks

https://paperswithcode.com/task/object-detection

YOLOv7 的主要貢獻如下：

- 使用了一些 **trainable bag-of-freebies tricks** 優化網路訓練，包括 model re-parameterization、dynamic label assignment
- 進一步分析 **model re-parameterization**，並提出 planned re-parameterized convolution module，使其能應用於各種模型架構
- 進一步分析 **dynamic label assignment**，並提出兩種 label assignment 策略來處理及分配不同分支
- 模型縮放與擴展 **(extend、compound scaling model)** 以有效運用參數量和計算量
- 跟之前的 SOTA real time 物件偵測模型相比降低了 40% 參數量、50% FLOPs，並有更快的推理運算速度及準確率。此外，所有模型皆是 **train from scratch** (previous YOLO: ImageNet pretrained)



Backbone: ELAN, E-ELAN[1]

Neck: CSPSPP + (ELAN, E-ELAN) PAN

Head: YOLOR

Recap from YOLOv4:

1. What is Bag of Freebies?

---

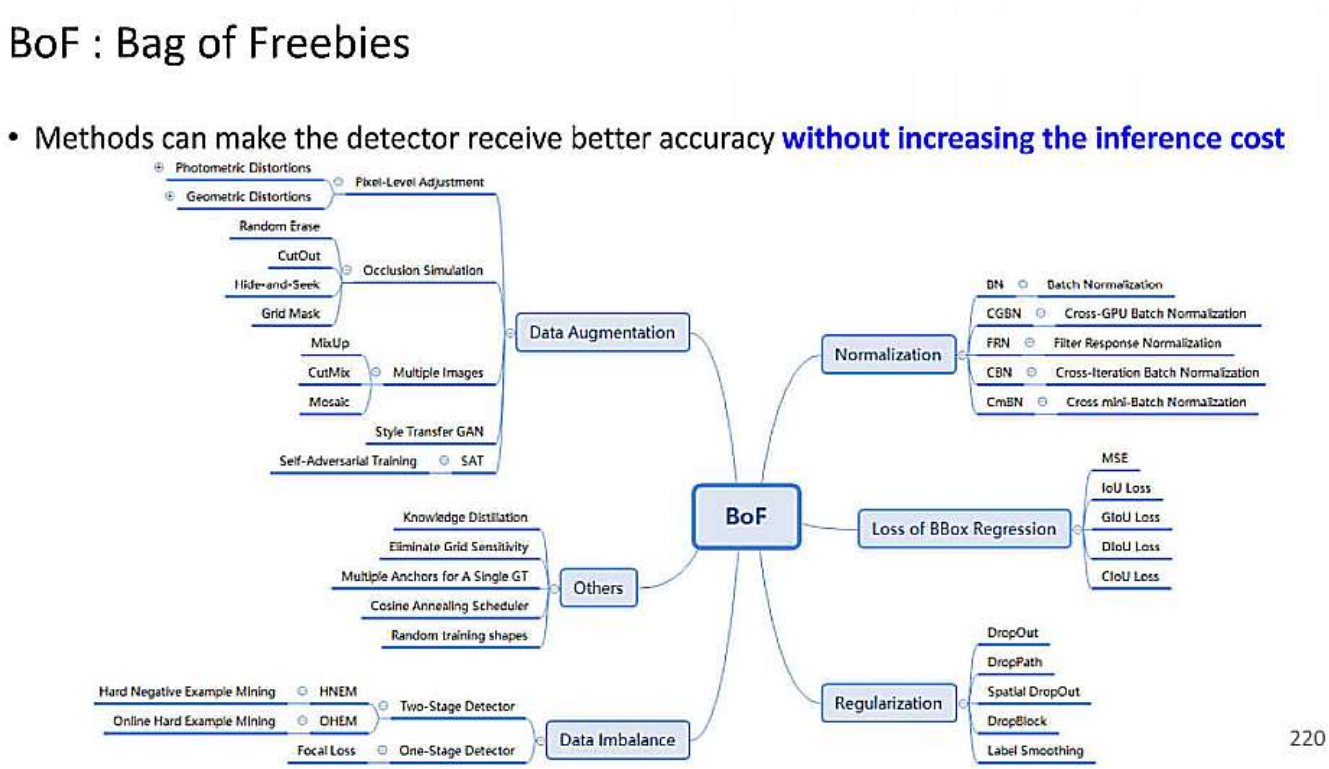
[1] https://github.com/WongKinYiu/yolov7/issues/458

In my opinion ELAN, E-ELAN are only parts of the backbone, but it is more a matter of how you define backbone.

Note: E-ELAN is only used in YOLOv7-E6E, while other variants use ELAN to replace E-ELAN.

2. What is Bag of Specials?

1. **Bag of Freebies (BoF)** are methods that only change the training strategy or only increase the training cost **without increasing the inference cost**, in order to achieve better accuracy.



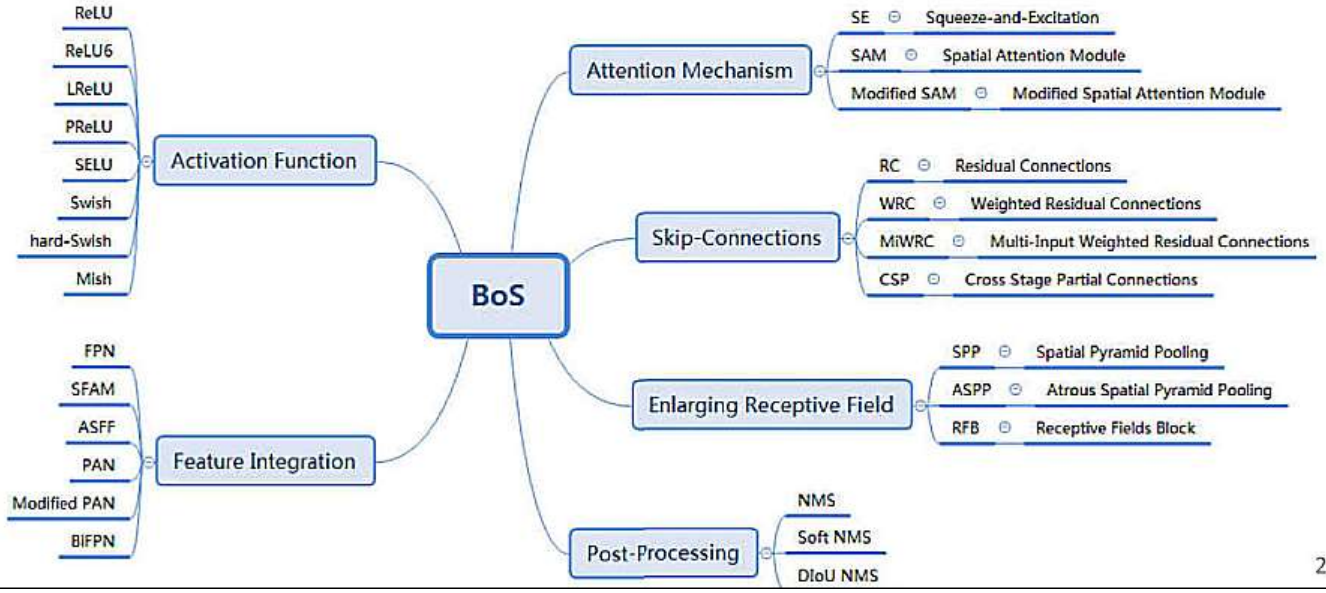
One common practice from BoF is **data augmentation**. The purpose of data augmentation is to increase the variability of the input images, so that the designed object detection model has higher robustness to the images obtained from different environments. It is especially helpful when the dataset is small and lack diversity.

2. **Bag of Specials (BoS)** are methods that only **increase the inference cost** by a small amount but can significantly improve the accuracy of object detection.

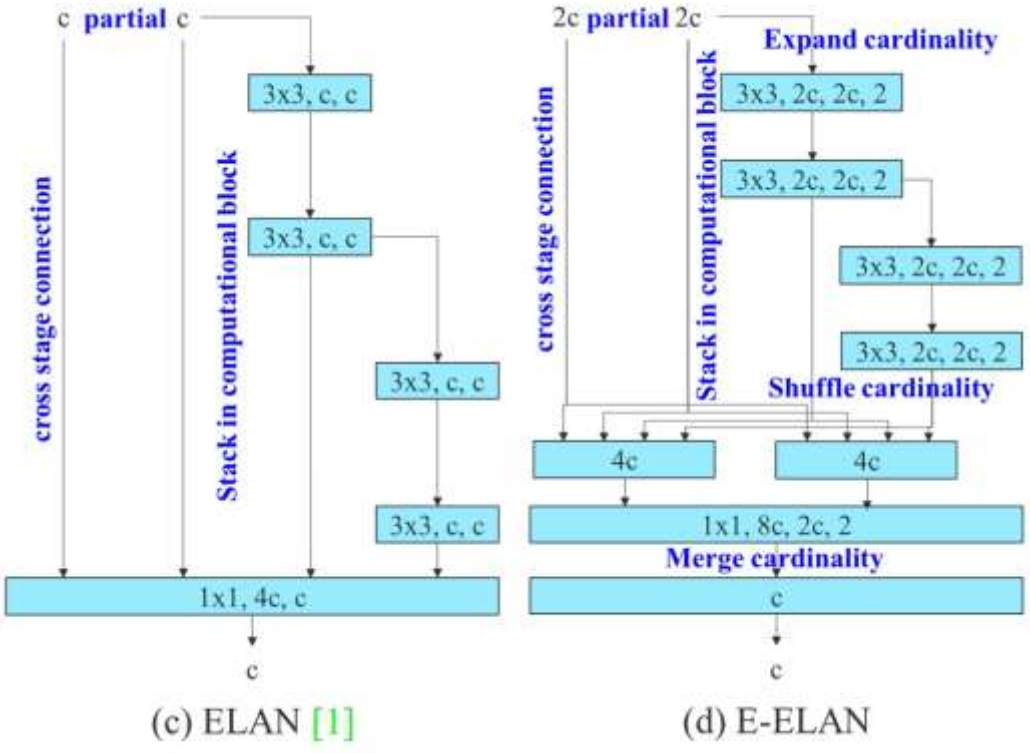
Examples of Bag of Specials are in the below image.

BoS : Bag of Specials

- Methods that only **increase the inference cost by a small amount** but can significantly improve the accuracy of object detection



New architecture: **Extended Efficient Layer Aggregation Networks (E-ELAN)**



Main idea behind designing E-ELAN: a more efficient network with acceptable model performance tradeoff.

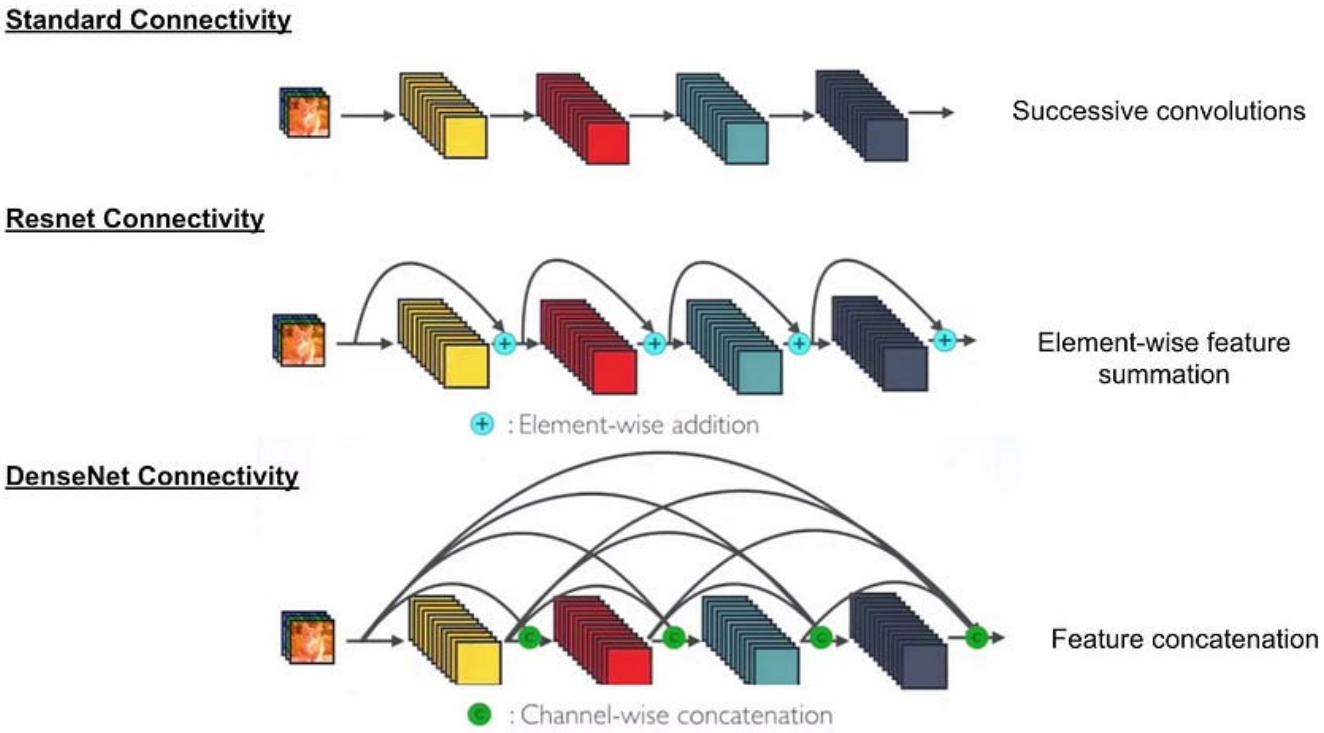
An example from YOLOv5 backbone: CSPDarkNet53

To explain why E-ELAN has such architecture, we can go through some network evolution before E-ELAN:

ResNet → DenseNet → VovNet → CSPVovNet → ELAN → E-ELAN

Note: CSPVovNet, ELAN, and E-ELAN are proposed by the author of YOLOv7.

ResNet → DenseNet



Differences between ResNet and DenseNet:

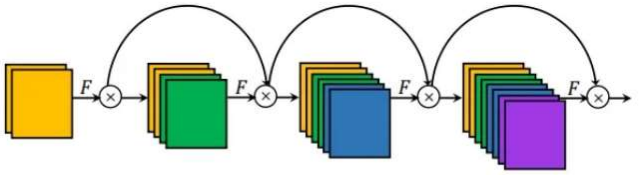
ResNet uses **element-wise addition**, while DenseNet uses **channel-wise concatenation** of feature maps.

Advantages of DenseNet: combines different scale of information from feature maps, way **fewer parameters** as feature concatenation reduces the need for extra kernels. **Preserve input information** without information loss. **Stronger gradient flow** as the error signal can be propagated to layers nearer to input more directly.

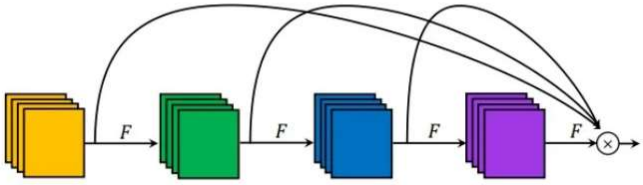
Disadvantage of DenseNet: Each layer of feature map uses all previous feature maps for computation, thus **high memory access cost (MAC)**. Number of channels grows linearly when the network goes deeper, using 1x1 convolution would help reduce channels.

DenseNet → VovNet[2]

[2] VoVNet stands for Variety of View Network.



(a) Dense Aggregation (DenseNet)



(b) One-Shot Aggregation (VoVNet)

Differences between ResNet and DenseNet:

DenseNet: **Dense Aggregation**; VoVNet: **One-Shot Aggregation**.

One-Shot Aggregation: VoVNet aggregates its feature map in the last layer at once, instead of every layer.

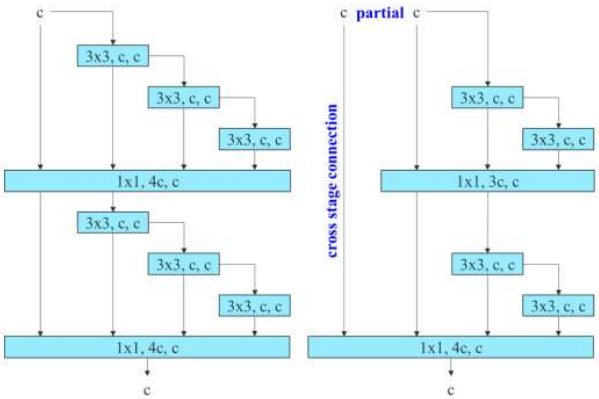
Advantages of using One-Shot Aggregation: reducing MAC and improving GPU computation efficiency.

VovNet → CSPVovNet

This "CSP" is the CSP we learnt in YOLOv4 (CSPDarknet-53, from CSPNet).

Refer to YOLOv4 notes for the advantages of using Cross Stage Partial Network.

Authors of YOLOv7: the architecture of CSPVoVNet also analyzes the gradient path, in order to enable the weights of different layers to learn more diverse features. The gradient analysis approach described above makes inferences faster and more accurate.



(a) VoVNet [39] (b) CSPVoVNet [79]

CSPVovNet → ELAN

ELAN 针对的问题是在 model scaling (more explanations in the next section) 中模型表现变差的问题。作者以 VoVNet 和 ResNet 做对比。**VoVNet 在叠加更多 block 时表现要比 ResNet 更差**，作者分

析是因为 VoVNet 结构中**存在过多的 transition layers (layers in which the size of feature map/ number of channels changes)**，这导致在叠加 block 时最短梯度路径 (the shortest gradient path) 不断增加，从而使得 block 增加时训练难度上升。因此，作者通过**适当删除** transition layer 来改善网络表现。(背後概念其實是為網絡做減法，除掉一些可能令結果變差的的網絡層。)

More about gradient path:

https://blog.csdn.net/weixin_43334693/article/details/130478338 (Section 3.1 (c))

https://blog.csdn.net/jiaoyangwm/article/details/128523467 (Section 2)

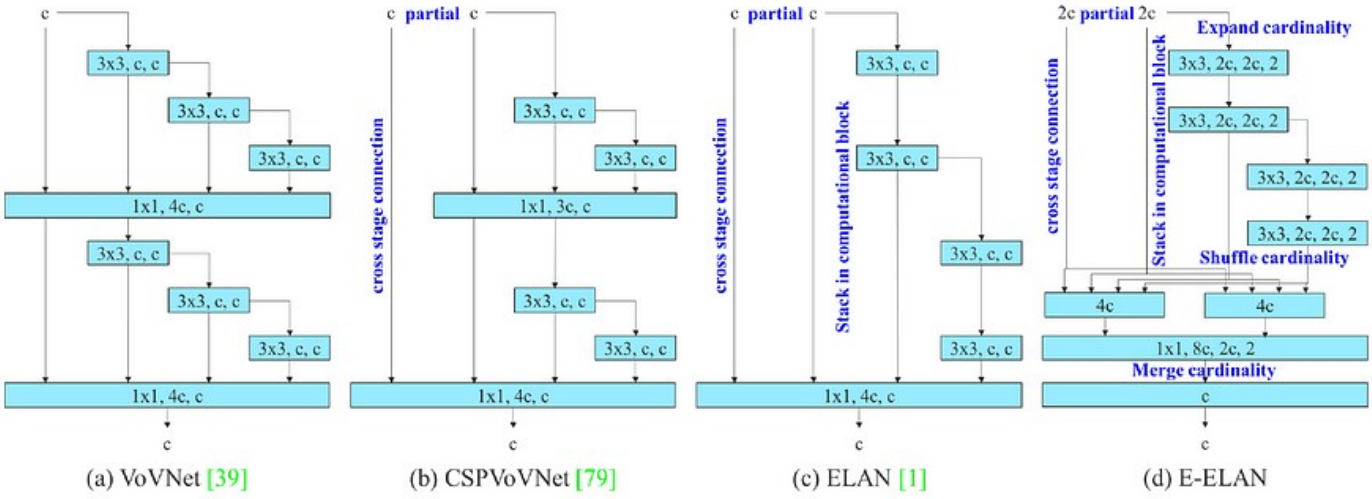


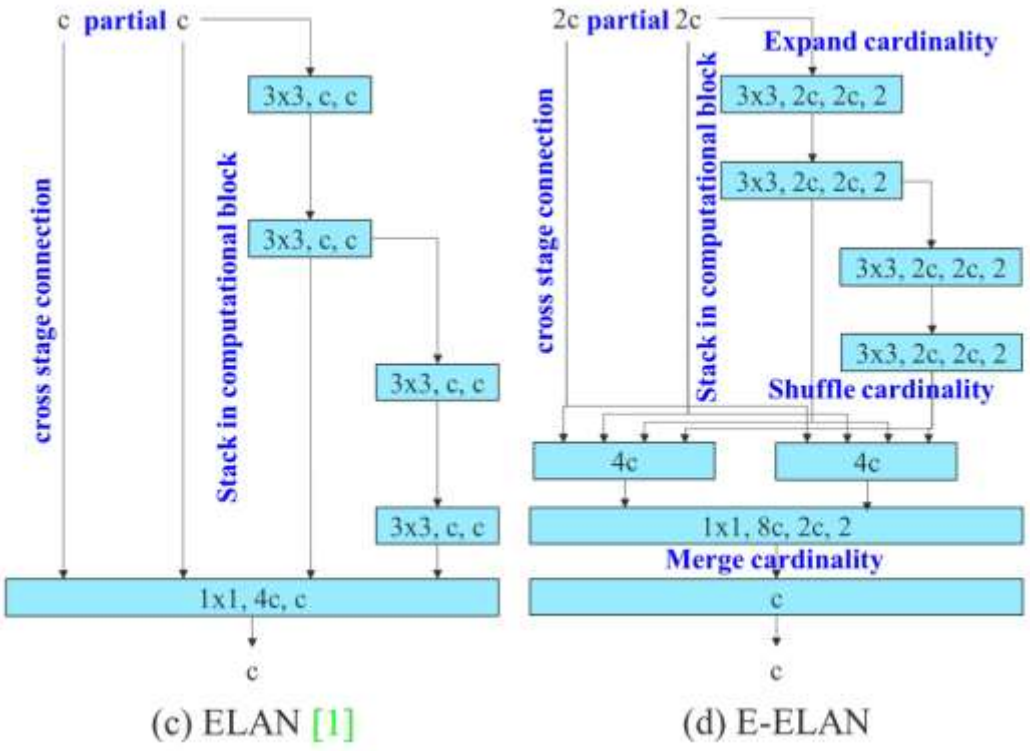
Figure 2: Extended efficient layer aggregation networks. The proposed extended ELAN (E-ELAN) does not change the gradient transmission path of the original architecture at all, but use group convolution to increase the cardinality of the added features, and combine the features of different groups in a shuffle and merge cardinality manner. This way of operation can enhance the features learned by different feature maps and improve the use of parameters and calculations.

Note: (3x3, 2c, 2c, 2) corresponds to (kernel size, input channel, output channel, no. of groups)

ELAN → E-ELAN

由於大型的 ELAN 架構在梯度路徑長度和 computational block 的堆疊數量已達到穩定的狀態，若是**再堆疊更多的 computational block 可能會導致參數利用率下降**。

因此提出的新架構 E-ELAN (d) 在基於 ELAN 的架構上加入了 **expand, shuffle, merge cardinality 方法**，能夠持續加強模型的學習能力，同時不破壞原本的梯度路徑。此外，E-ELAN 僅改變 computational block 架構，**並沒有更動 transition layer**。



Expand

採用 **Group convolution** 擴展所有 computational block 的 channel 及 cardinality，並且這些 computational block 中的**每層 layer 都應用相同的 group 參數和 channel 倍數**。這樣的操作使得模型有較強的可擴展性 (模型能將輸入的資訊增放到更高維度的空間做訓練，並且不同組別能學習不同方面的特徵)。

Cardinality 是指分組的數量，由 ResNeXt 提出。有助於多 GPU 模型分散運算、減少參數量、約束相鄰 kernel 間的相關性，避免模型過擬合。

作者在论文中引入了一个叫作基数（cardinality）的超参数，指独立路径的数量，这提供了一种调整模型容量的新思路。实验表明，通过扩大基数值（而不是深度或宽度），准确率得到了高效提升。作者表示，与 Inception 相比，这个全新的架构更容易适应新的数据集或任务，因为它只有一个简单的范式和一个需要调整的超参数，而 Inception 需要调整很多超参数（比如每个路径的卷积层内核大小）。
这个全新的结构有三种等价形式，fig3.a 就是前面所说的aggregated residual transformations。fig3.b 则采用两层卷积后 concatenate，再卷积，有点类似 Inception-ResNet，只不过这里的 paths 都是相同的拓扑结构。fig 3.c采用的是grouped convolutions，采用32个 group，每个 group 的输入输出 channels 都是4，最后把channels合并。作者在文中明确说明这三种结构是严格等价的，并且用这三个结构做出来的结果一模一样。

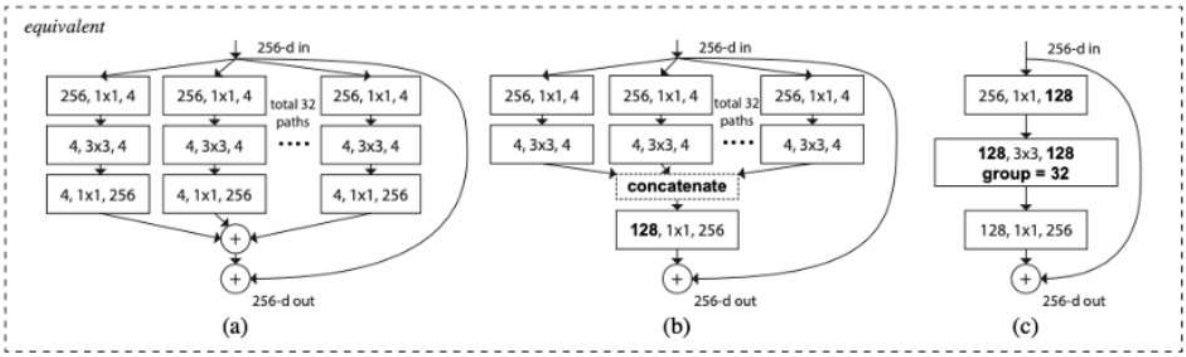


Figure 3. Equivalent building blocks of ResNeXt. **(a)**: Aggregated residual transformations, the same as Fig. 1 right. **(b)**: A block equivalent to (a), implemented as early concatenation. **(c)**: A block equivalent to (a,b), implemented as grouped convolutions [23]. Notations in **bold** text highlight the reformulation changes. A layer is denoted as (# input channels, filter size, # output channels).

Note: YOLOv7 uses approach (c) (group convolution) to increase cardinality.

Group convolution: 將卷積層進行分組(g groups)，然後採用不同的卷積核再對各個組進行卷積，最後把結果 concat 起來。Group convolution is first mentioned in AlexNet.

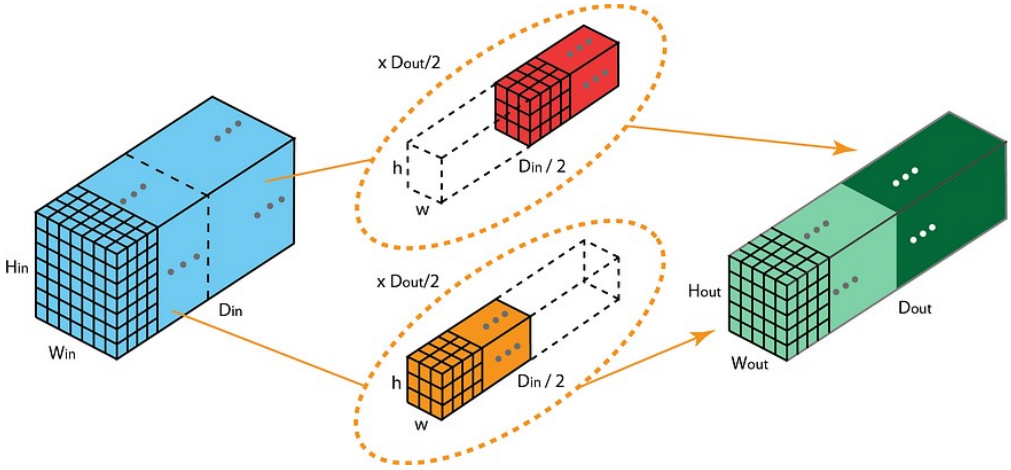
Group convolution is like **a method in between traditional convolution and depthwise convolution** in terms of **how many channels do each kernel handle**.

Number of groups g in group convolution in YOLOv7: $g = 2$

e.g. $D_{in} = 64, g = 2$, kernel size = 3×3

Each group is of dimension $(H_{in}, W_{in}, \frac{64}{2}) = (H_{in}, W_{in}, 32)$.

2 kernels with dimension $(3, 3, 32)$.



Advantage: increase cardinality, 可以減少運算量, by how many? By $1/g$.

Where does such reduction of computation come from?

Each kernel only handles parts of the input channels, so each group convolution can be regarded as separate traditional convolution with smaller size.

Limitation: Group convolution 中不同 groups 之間的資訊完全獨立，convolution kernels 只作用在各自的 group 中，**channels 接起來的時候也不會混合資訊**，限制了模型表徵能力。

That's why we have **Shuffle** next after Expand in E-ELAN.

Shuffle

將 computational block 的輸出進行 channel shuffle，假設 group 參數為 g，就將其 channel shuffle 為 g 組，再分別進行 concatenation。

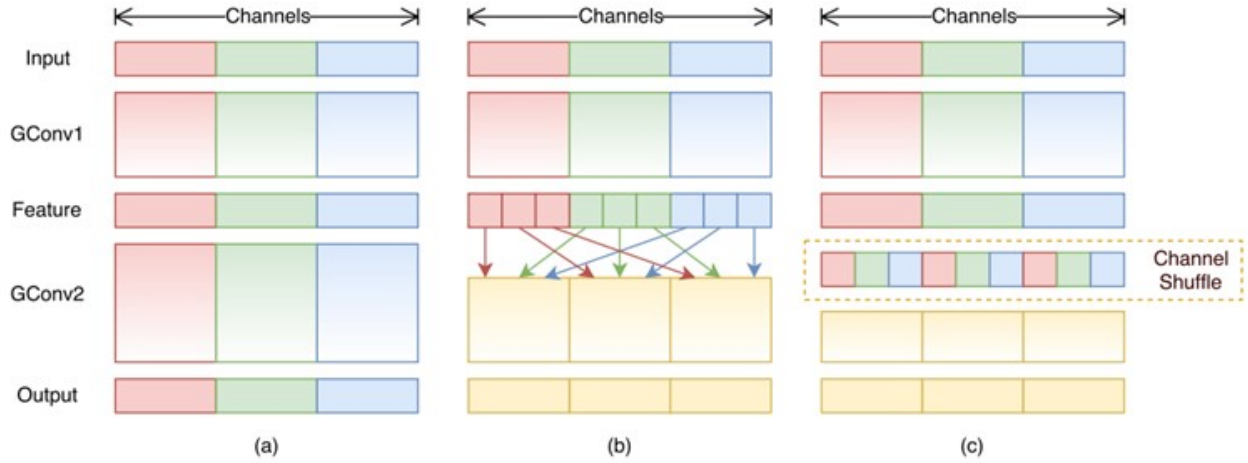
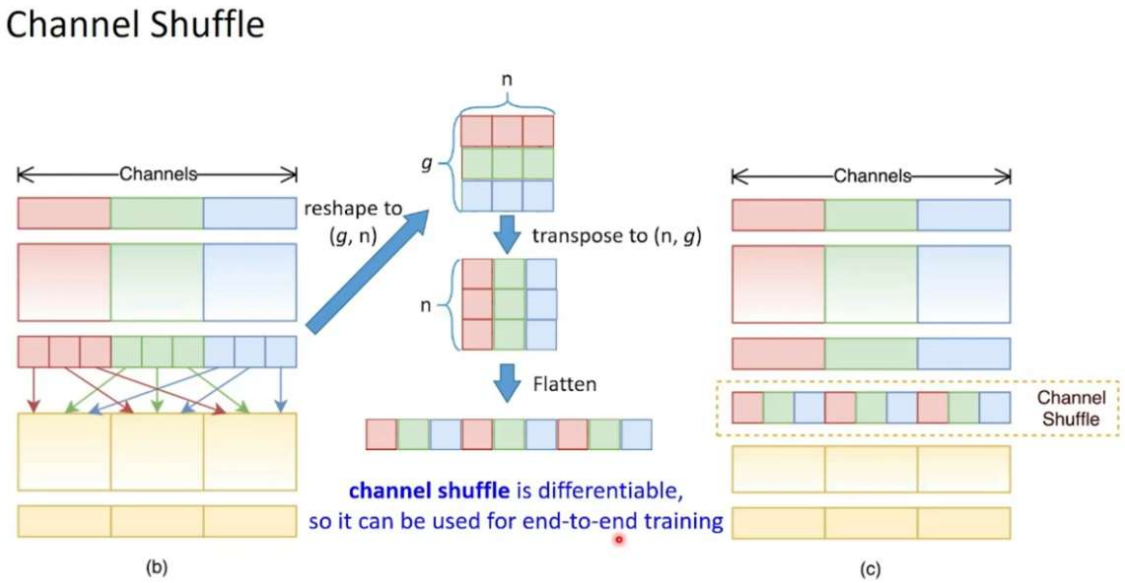


Figure 1. Channel shuffle with two stacked group convolutions. GConv stands for group convolution. a) two stacked convolution layers with the same number of groups. Each output channel only relates to the input channels within the group. No cross talk; b) input and output channels are fully related when GConv2 takes data from different groups after GConv1; c) an equivalent implementation to b) using channel shuffle.

Question: In practice how do we perform channel shuffling on our feature map? (let say the size of our feature map from is $(n, 416,416)$, where n is the number of channels of the feature map, and we have g groups.)

針對分為g個 groups、每個 group 有n 個 channels 的 group convolution 的輸出張量，執行：

1. 將總共$g \times n$ 個 channels 的輸出張量 reshape 為(g, n)
2. 將張量 transpose
3. 將張量 flatten



Advantage of channel shuffling after **Expand**: channel shuffling allows information exchange (feature) across groups from **Expand**.

In terms of information exchange across channels, we can also use 1×1 convolution to achieve, but **performing 1×1 convolution is computationally more expansive than channel shuffling**, as channel shuffling only involves data movement without any arithmetic computation.

Question: Does the g in group convolution necessarily be the same as the g in channel shuffling?

Yes, as this allows every group in the next layer to utilize all features learnt from all previous groups.

Merge cardinality: 最後再以**相加 (add)** 的操作將這些分組的卷積層進行**融合**。

Note from the author of YOLOv7: For E-ELAN architecture, since our edge device do not support group convolution and shuffle operation, we are forced to implement it as an **equivalence architecture**, which is shown in Figure A3.

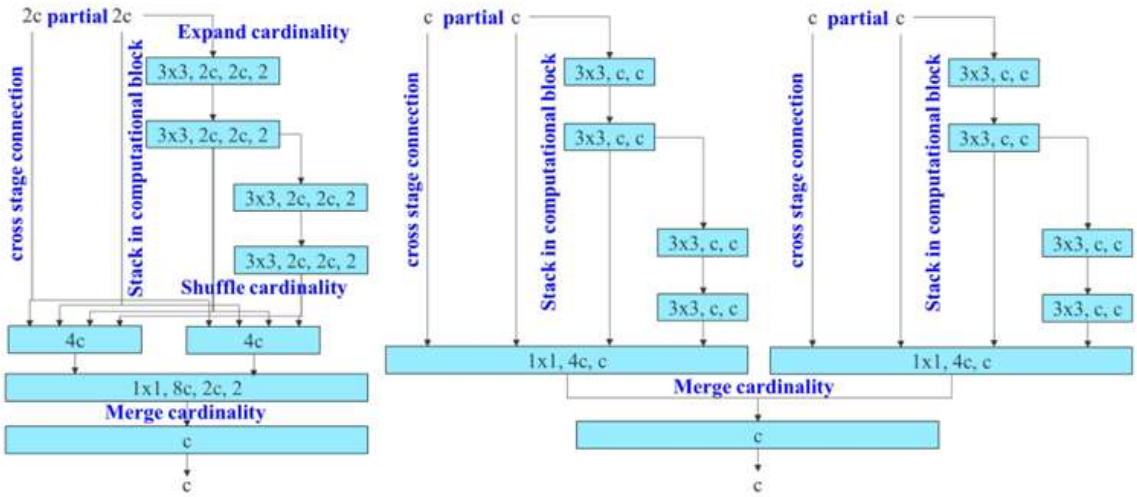
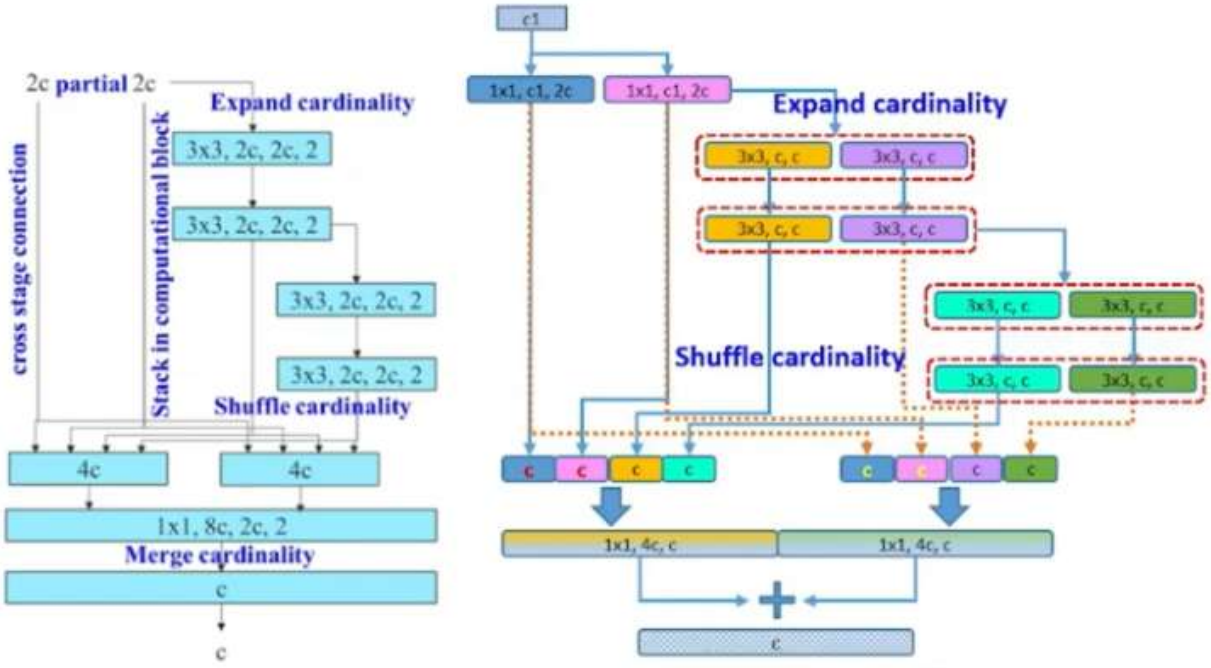
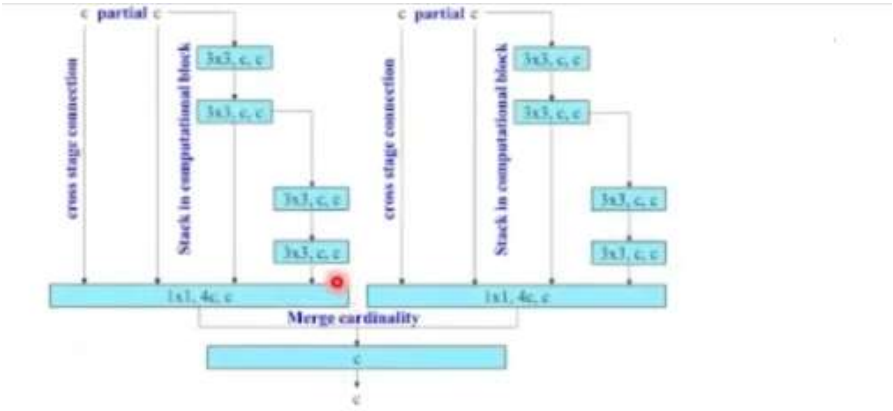


Figure A3. Equivalence E-ELAN.

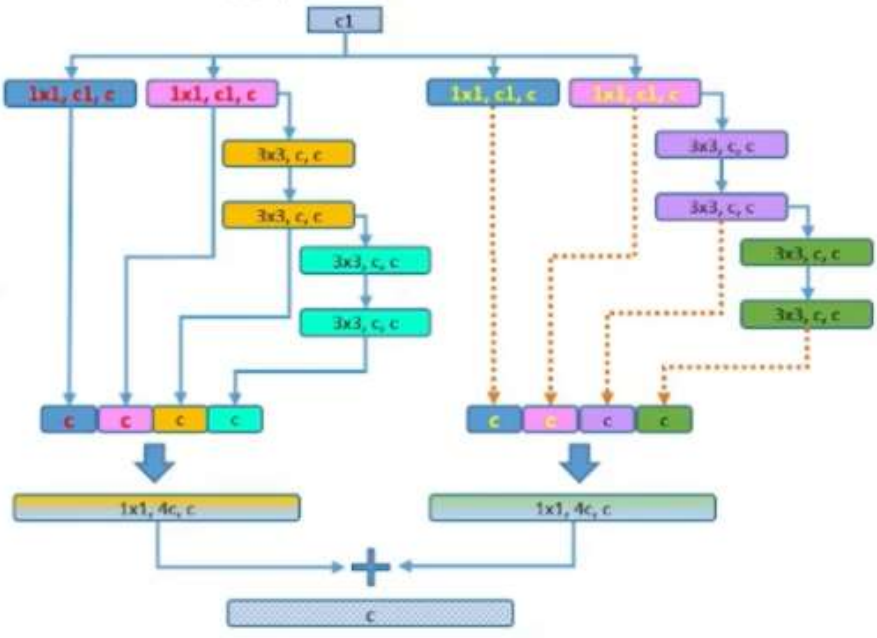
Q: Are they really equivalent? See the next two figures for comparisons.



(d) E-ELAN



(b) Equivalence E-ELAN



Equivalence E-ELAN

They are really equivalent!

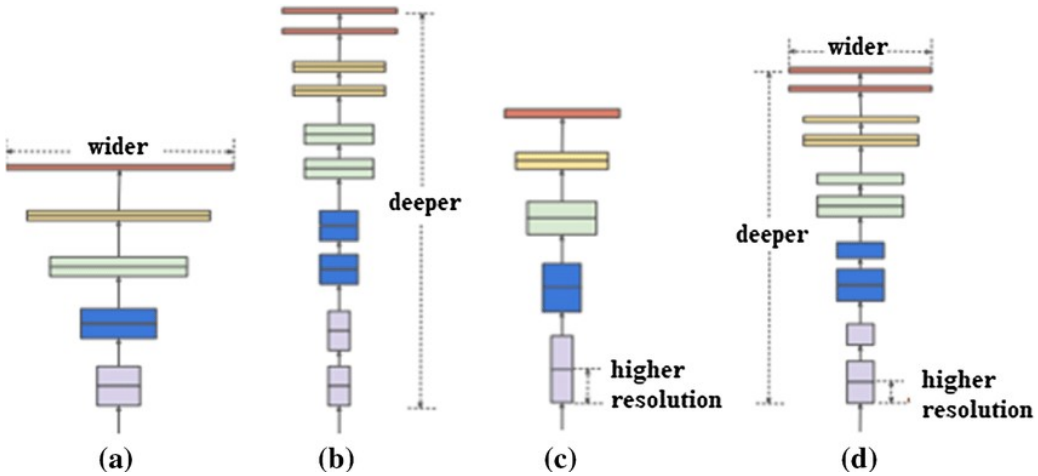
Model Scaling

Model scaling is a way to scale up or down an already designed model and make it fit in *different computing devices*, and make a good **trade-off** for the amount of network parameters, computation, inference speed, and accuracy.

The inference speed of a designed model is not only affected by the amount of computation and model size, but more importantly, the **limitation of peripheral hardware resources** must be considered. Therefore, when performing tiny model scaling, we must also consider factors such as **memory bandwidth**, **memory access cost (MACs)**, and **DRAM traffic**.

(extracted from *Scaled-YOLOv4: Scaling Cross Stage Partial Network*)

Scaling factors: **resolution** (size of input image), **depth** (number of layer), **width** (number of channel), and **stage** (number of feature pyramid).



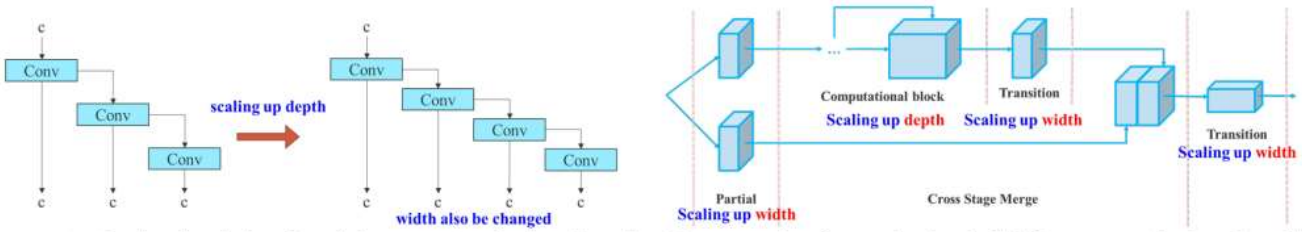
Different model scaling: (a) width scaling, (b) depth scaling, (c) resolution scaling and (d) compound scaling

Observation from the author of YOLOv7:

Checking the literature, we found that almost all model scaling methods **analyze individual scaling factor independently,** and even the methods in the compound scaling category also **optimized scaling factor independently**.

Problem: We want to analyze the impact of each scaling factor on the number of parameters and computation, but **for concatenation-based models, we cannot easily analyze that**. (Why?)

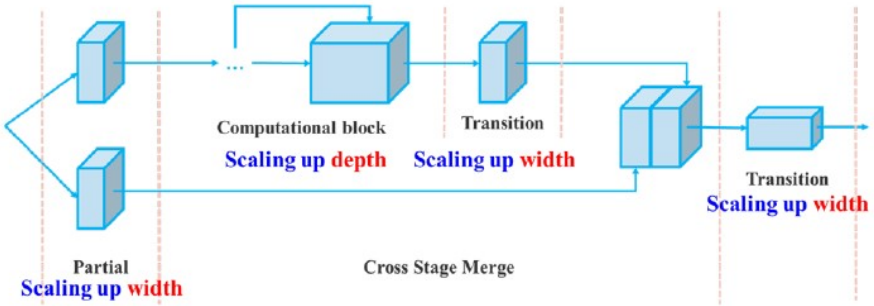
The figure below explains this problem in details.



(a) concatenation-based model (b) scaled-up concatenation-based model (c) compound scaling up depth and width for concatenation-based model

Figure 3: Model scaling for concatenation-based models. From (a) to (b), we observe that when depth scaling is performed on concatenation-based models, the output width of a computational block also increases. This phenomenon will cause the input width of the subsequent transmission layer to increase. Therefore, we propose (c), that is, when performing model scaling on concatenation-based models, only the depth in a computational block needs to be scaled, and the remaining of transmission layer is performed with corresponding width scaling.

為了解決這個問題，作者提出了 compound model scaling 方法。下圖為該方法的操作，當對於 computational block 的 depth 進行縮放時，計算縮放後 computational block 的輸出通道變化量，並以該變化量對 transition layer 進行 (main idea is that it provides a way to calculate the scaling ratio)



(c) compound scaling up depth and width for concatenation-based model

How can compound scaling up depth and width solve the problem for concatenation-based model?

Compound scaling 可以同時**維持最初模型的 property 與最優的結構**。

Trainable bag-of-freebies

1. Planned re-parameterized convolution: RepConv

Main idea of model re-parameterization: improving the model performance and reducing the frequency of memory access, while the inference cost remains more or less the same. Common model re-parameterization methods fall into these two levels: model-level and module-level.

📌model-level

常見的做法有兩種：

1. 用不同的訓練資料訓練多個相同的模型，再平均這些模型的權重。
2. 將訓練過程中多個不同迭代的權重進行加權平均。

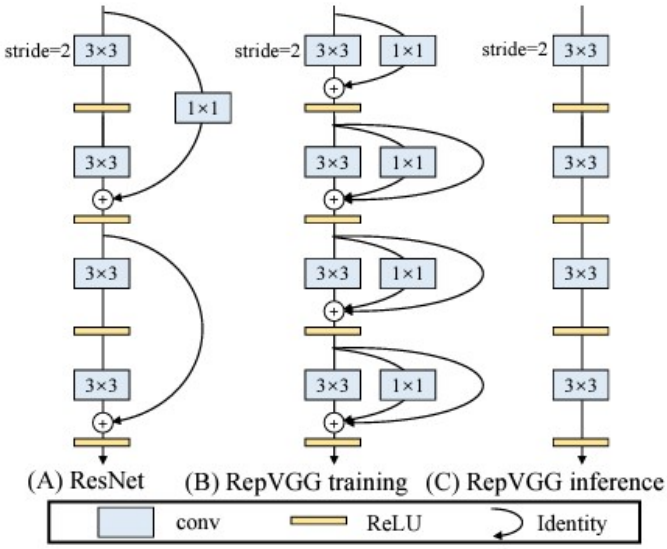
📌module-level

module-level ensemble method 透過等價轉換多個 module 為單個 module（即 model compression），將訓練及推論階段上的模型進行解耦，因此這**兩個階段上的模型會具有不同的架構**。

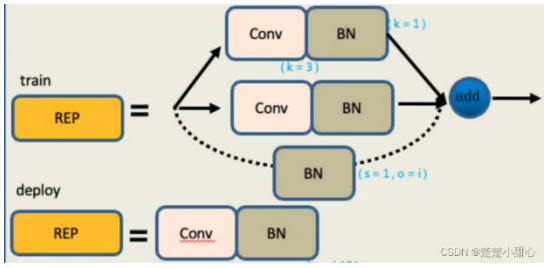
RepConv:

RepConv is originated from RepVGG, with the core strategy to re-parameterize convolution blocks. During training, the block in RepConv is multi-branches (RepConv actually combines 3 × 3 convolution, 1 × 1 convolution, and identity connection in one convolutional layer), but during inference, RepConv is re-parameterized to plain network[3].

RepVGG implementation:



YOLOv7 RepConv implementation:



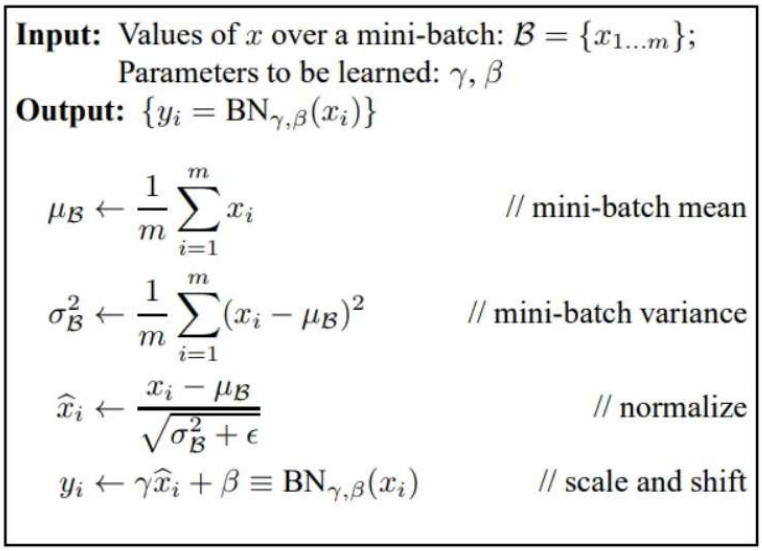
For activation function, YOLOv7 uses SiLU instead of ReLU. The branches are the same as that in RepVGG.

具體上 RepConv 的模型重參數化操作可以分為兩個步驟：

- **Folding BN**: 將 batch normalization 的 (μ, σ, γ, β) 折進 convolution 的 weights & biases 中 。實作上，接在 convolution 後的 BN ，通常在 inference 時都可以被整合到 convolution 的 weights 與 biases 內：與 deviation 相關的 (σ, γ) 變成 weights 的 scale ，而 (μ, σ, γ, β) 變成 biases。
- **Reparameterization to plain**: 將 3x3 convolution 、1x1 convolution 與 residual path 合成一個 3x3 convolution 。RepVGG 論文中雖然使用了這三個組合，但其實方法並不限制於特定 kernel size 。

實際的 BN 是在訓練過程中統計 (μ, σ, γ, β) ，然後對 latent features 進行正規劃。

[3] The idea of "plain network" is first proposed in *Deep Residual Learning for Image Recognition* by Kaiming He. It follows two simple design rules: (i) for the same output feature map size, the layers have the same number of filters; and (ii) if the feature map size is halved, the number of filters is doubled so as to preserve the time complexity per layer.



Batch normalization 實際運作的行為。(資料來源)

實作上，接在 convolution 後的 BN ，通常在 inference 時都可以被整合到 convolution 的 weights 與 biases 內：與 deviation 相關的 (σ, γ) 變成 weights 的 scale ，而 (μ, σ, γ, β) 變成 biases。

$$\mathrm{bn}(\mathrm{M}, \boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{\gamma}, \boldsymbol{\beta})_{:,i,:,:} = (\mathrm{M}_{:,i,:,:} - \boldsymbol{\mu}_i)\frac{\boldsymbol{\gamma}_i}{\boldsymbol{\sigma}_i} + \boldsymbol{\beta}_i$$

$$\mathrm{W}'_{i,:,:,:} = \frac{\boldsymbol{\gamma}_i}{\boldsymbol{\sigma}_i}\mathrm{W}_{i,:,:,:}, \quad \mathbf{b}'_i = -\frac{\boldsymbol{\mu}_i\boldsymbol{\gamma}_i}{\boldsymbol{\sigma}_i} + \boldsymbol{\beta}_i$$

在把 BN 折進 convolution 後，我們可以進一步地把 1x1 convolution 與常用於建立 residual 的 identity path ，也改寫成只有中心有數值的 3x3 convolution。

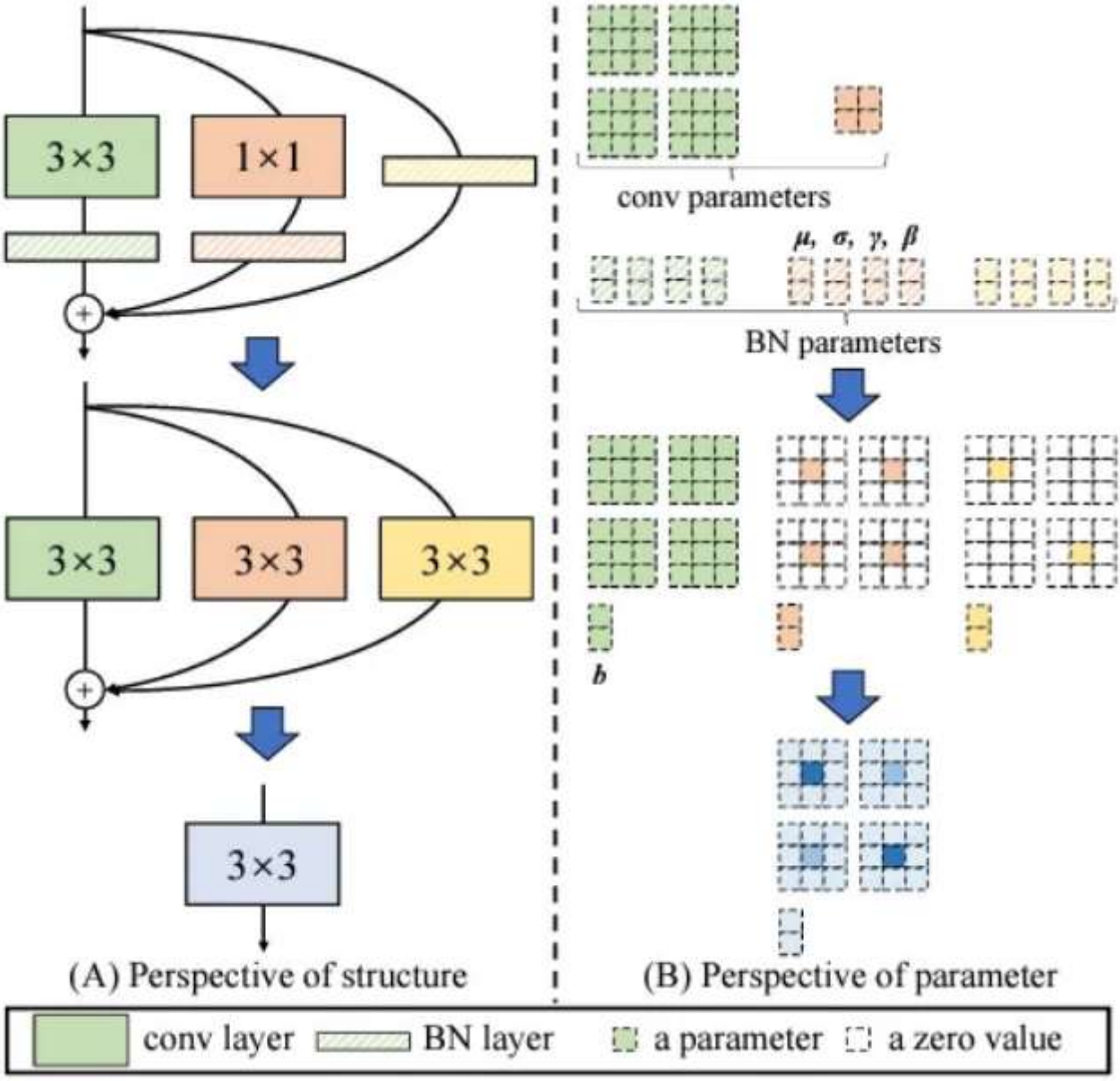
- **3x3 convolution**: 單純把 BN 折進去。
- **1x1 convolution**: 把 BN 折進去後，再 padding 成 3x3 convolution 。
- **identity path**: 把 BN 折進去得到 1x1 的 scales 與 biases ，再 padding 成 3x3 convolution。

接下來就把 kernel weights 與 biases 加總起來，就得到只剩下 convolution 與 activation function 的 plain network。

Mathematical derivation and code implementation of the above three re-parameterization:

https://blog.csdn.net/weixin_41012399/article/details/141642913

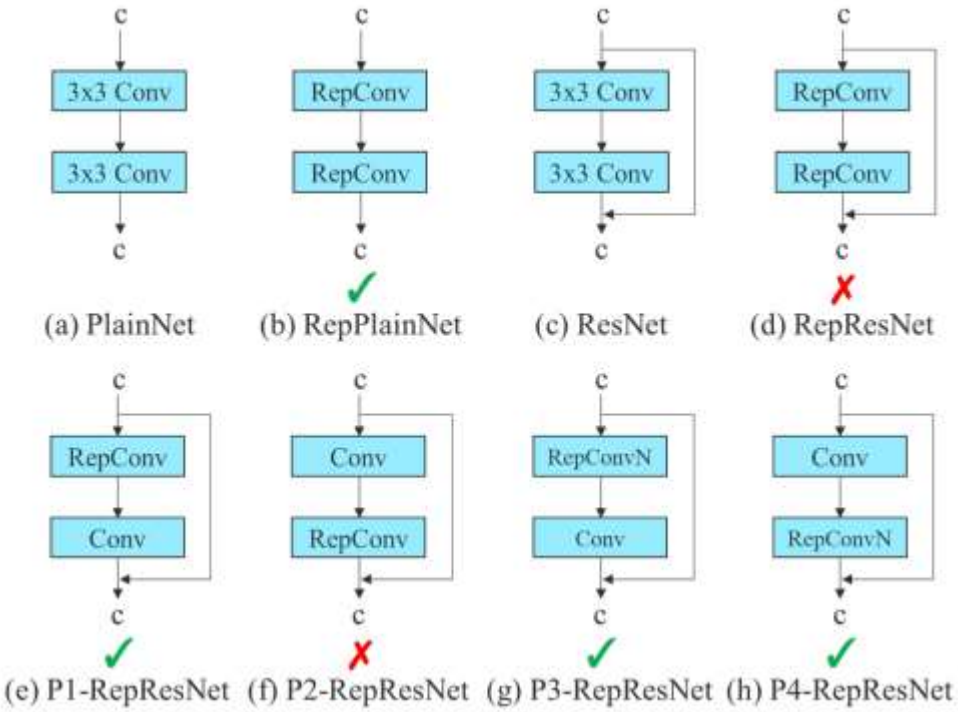
Visualization of the re-parameterization process:



RepVGG 設計的 building block 具體的 reparameterization 過程。(資料來源)

From author of YOLOv7:

Although **RepConv** has achieved excellent performance on the VGG, when we **directly apply** it to ResNet and DenseNet and other architectures, **its accuracy will be significantly reduced**.



經分析後，作者認為 RepConv 中的 identity connection 會破壞 ResNet 中的 residual 及 DenseNet 中的 concatenation 給不同 feature map 的梯度多樣性，由於上述原因，作者使用沒有 identity connection 連接的 RepConv (稱為 RepConvN) 來設計重參數化卷積的架構。

Q: How does identity connection in RepConv harm the diversities of gradients for different feature maps?

Our hypothesis:

During the reparameterization of RepConv block **from training to inference**, the fusion of **BN layer suffers from floating-point errors**, and we believe that practically the resultant 3x3 conv block in inference is **NOT equivalent** to the fusion of three branches (3x3 conv, 1x1 conv, identity connection) in training. **Such errors can be accumulated** from the addition/ concatenation properties of specific blocks (e.g. ResNet, DenseNet).

Note that (d) and (f) 放 RepConv 在最後的計算，導致收尾有誤差，下一層收到這誤差會經過 concat/add 和 RepConv 的 identity connection 繼續累積誤差。

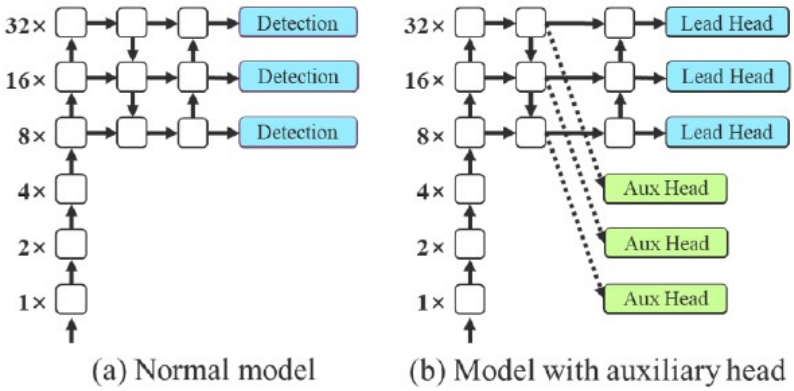
But (e), (g), (h) 放 Conv/ RepConvN 在最後的計算，上一層即使有誤差仍可通過 Conv/ RepConvN 修正 (**沒了 identity connection，這層必須做運算**)，減少誤差。

2. Dynamic Label Assignment: Coarse for auxiliary and fine for lead loss

在**訓練**深層的網路時，經常會在神經網路的中間層加入**分類輔助器** (**auxiliary head**、**auxiliary classifiers**) 來提高穩定性、收斂速度、避免梯度消失問題，也就是使用 auxiliary loss 對淺層的網路

權重進行訓練，這樣的技術稱為 **Deep Supervision**。

下圖 (a)、(b) 為有無使用 Deep Supervision 的 object detector 的比較，其中最後負責輸出的 head 稱為 **lead head**，輔助訓練的 head 稱為 **auxiliary head**。(More on auxiliary classifiers: GoogLeNet)

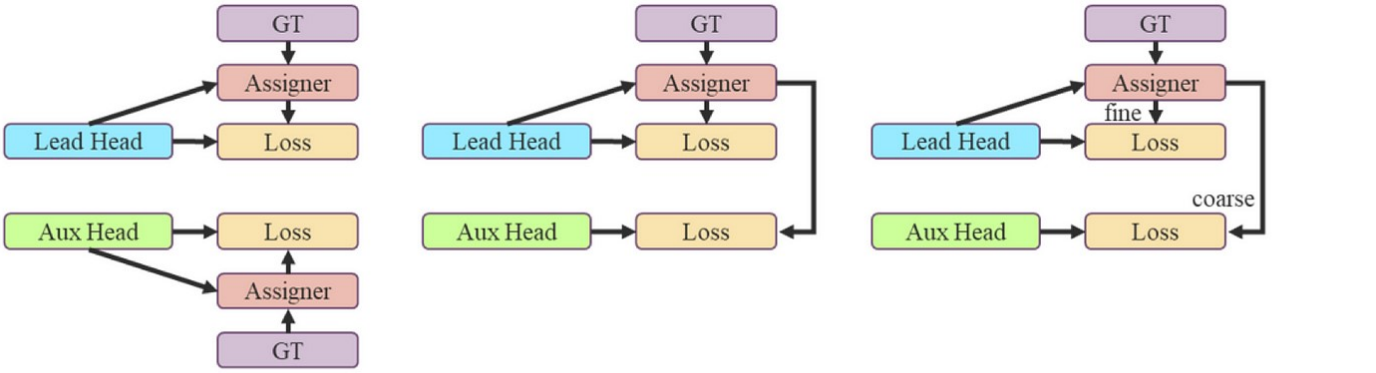


Q: How does adding auxiliary head help train deep network? (i.e. how can auxiliary head achieve 提高穩定性、收斂速度、避免梯度消失問題？)

If aux head **already performs well using the feature map from shallow layers**, it means that the model already learns a good feature expression from the input. When more layers are added to refine the feature extraction, surely the lead head can learn much finer features that are not covered by the aux head, and thus converge faster and speed up training.

過去在深度網路的訓練中，通常會直接以 ground truth 來產生 **hard label** 來訓練網路權重，例如 CNN 分類任務，在近年來，除了 hard label，YOLO 還會利用模型預測的邊界框和 ground truth 的 IoU 來當做 objectness 的 **soft label**，作者把分配 soft label 的機制稱為 **label assigner**。

為了訓練 auxiliary head 和 lead head，**當時最常用的方法就是各自獨立分配**，也就是說使用它們自己的預測結果和 ground truth 來進行 label assigner，如 **圖(c)** 所示，而作者提出了使用 lead head 預測的 soft label 作為指導，來產生 coarse to fine 的階層式 soft label，進而分別用於 auxiliary head 及 lead head 的學習，如 **圖(d)(e)** 所示。



(c) Independent assigner (d) Lead guided assigner (e) Coarse-to-fine lead guided assigner

- **Lead head guided label assigner**

由於 lead head 比 auxiliary head 具有較強的學習力，因此 lead head 的預測結果與 ground truth 進行最佳化運算所得到 soft label 更能表達資料與 ground truth 間的分佈及相關性。

接著將該 soft label 作為 auxiliary head 和 lead head 的 target 進行訓練，讓較淺層的 auxiliary head 直接學習 lead head 已學到的資訊，而 lead head 則是更關注於未學到的 residual information。

- **Coarse-to-fine lead head guided label assigner**

這部分同樣也是採用 lead head 的預測結果與 ground truth 進行最佳化運算所得到 soft label，**差別在於會產生兩組不同的 soft label: coarse label、fine label**，其中 fine label 與 lead head 的 soft label 相同、coarse label 用於 auxiliary head。

那為什麼要分為兩組 soft label 呢？

剛有提到 lead head 比 auxiliary head 具有較強的學習力，換句話說 auxiliary head 的學習效果較差，因此 auxiliary head 可能會遺失掉需要學習的資訊。

而 coarse label 解決上述問題的做法是**藉由放寬 positive sample assignment 的限制**，讓更多 grid 被視為 positive target。這段話的意思是指在 coarse label 中被視為 positive target 的 grid 範圍會比 fine label 來的廣。

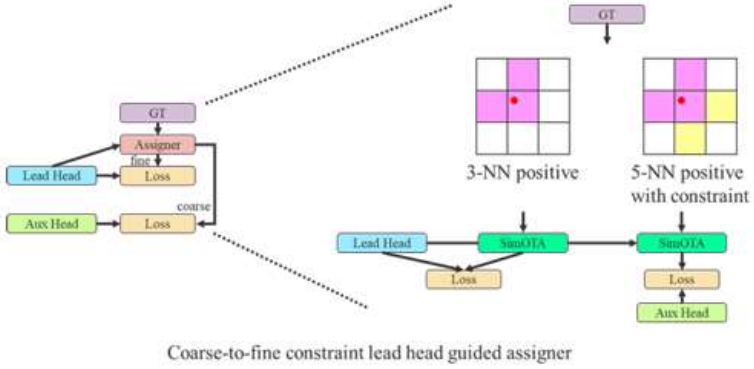


Figure A5. Coarse-to-fine lead head guided label assigner.

Hard negative mining:

這裡是在解決深度學習經典的困難負樣本挖掘(hard negative mining)問題

- Hard Negative
    - 即不容易與正樣本分辨的模糊樣本，在物件檢測任務當中，可以理解為容易被當成有目標物件存在但實際是背景的負樣本、
- Easy Negative
    - 容易被正確判定是沒有目標物件的負樣本(背景)。

物件檢測任務中，模型在隨機產出候選框(anchor-based)或預測的關鍵點(anchor-free)時，特別是對於很小的物體會因背景佔多數而得到大量負樣本，導致正負樣本比例失衡問題，因此，需要特別選擇難以分辨的負樣本(hard negative)加入負樣本集，才能讓模型得到比較好的判別效果。

在 YOLOv7 採用的困難負樣本挖掘方法是，**訓練模型利用中間網路層(deep supervision)產出的特徵圖學習初步分類，放寬對正樣本的限制來增加模型的判斷困難度**，對應前文所說不同階層的粗、細(Coarse-to-fine)概念。其中，輔助頭(分類器)得到的軟標籤(正負樣本判定數值)，是根據導引頭(最後輸出的分類器)的預測結果優化得來(前文所說的 guided label)

YOLOv7 combines the methods of positive sample assignment from YOLOv5 and YOLOX[4] (simOTA). The method of positive sample assignment is still anchor-based.

YOLOv5 label assigner: each GT can be associated to multiple anchor boxes, but **each anchor box must be associated to only one GT**.

[4] YOLOX: Exceeding YOLO Series in 2021. https://arxiv.org/abs/2107.08430

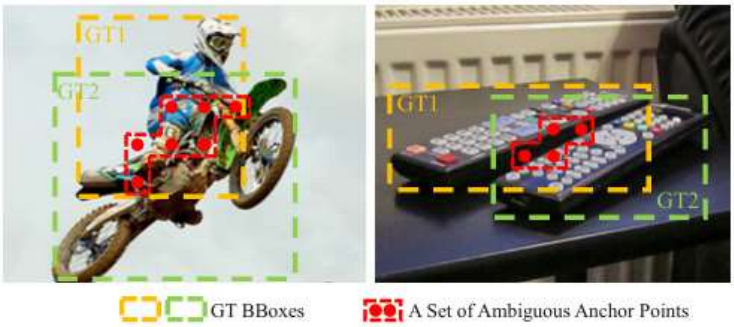
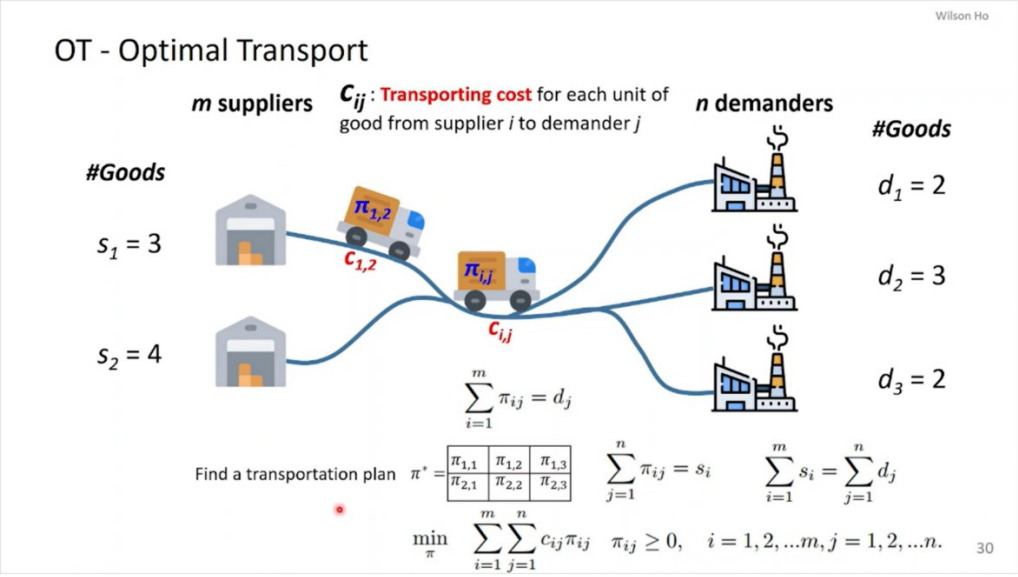


Figure 1. An illustration of ambiguous anchor points in object detection. Red dots show some of the ambiguous anchors in two sample images. Currently, the assignment of these ambiguous anchors is heavily based on hand-crafted rules.

Problem: When two ground truth boxes overlap, some anchors can be qualified as positive samples for multiple GTs simultaneously (ambiguous anchors). assigning ambiguous anchors to any GT (or background) may introduce harmful gradients w.r.t. other GTs. A **better assigning strategy** should get rid of the convention of pursuing optimal assignment for each GT independently and turn to the ideology of global optimum, in other words, **finding the global high confidence assignment for all GTs** in an image.

Somebody[5] formulates such global label assignment problem as an **Optimal Transport (OT) problem**– a special form of Linear Programming (LP) in Optimization Theory. This is related to how Optimal Transport Assignment (OTA) and simOTA are designed.

Brief overview of Optimal Transport (OT) problem:



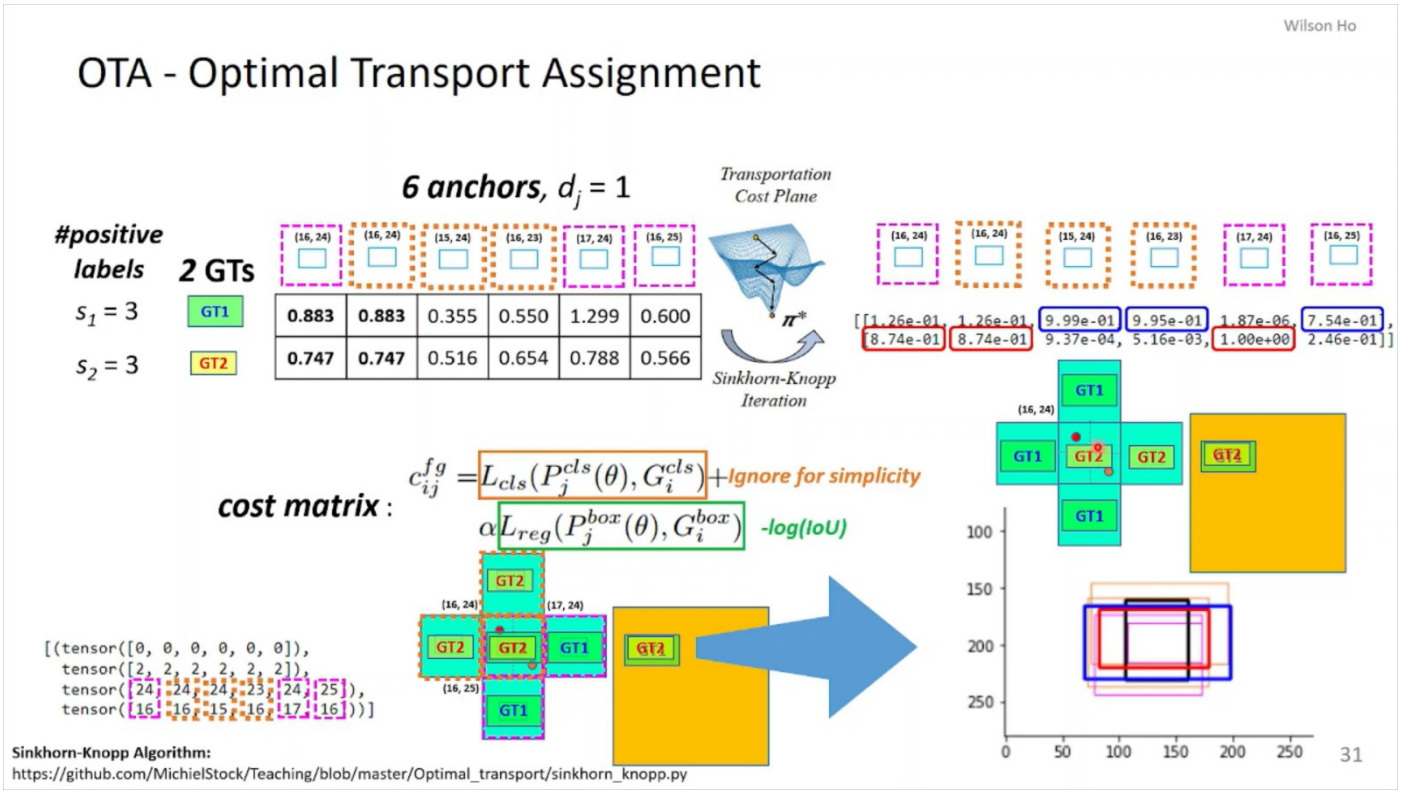
[5] *OTA: Optimal Transport Assignment for Object Detection*. https://openaccess.thecvf.com/content/CVPR2021/papers/Ge_OTA_Optimal_Transport_Assignment_for_Object_Detection_CVPR_2021_paper.pdf

Goal: Find a transportation plan to which all goods from suppliers can be transported to demanders **at a minimal transportation cost**. (We skip the mathematical parts here)

Analogy to Label Assignment in Object Detection:

Optimal Transport Problem	Label Assignment
Supplier	Ground Truth target/ Background target
Demander	Anchor
Goods	Positive labels/ Negative labels
Transportation Cost	Weighted sum of reg loss and cls loss of the label assignment
Optimal Transportation Plan	Optimal Assigning Plan

Note: each anchor is only assigned to one Ground Truth target.



Extracted from 圖解 YOLOv7 loss (2/2) https://www.youtube.com/watch?v=EhXwABGhBrw&t=2735s

The above linear programming problem can be solved by Sinkhorn-Knopp iterative method[6] in polynomial time, and it only **increases the total training time by less than 20%** and is totally cost-free in testing phase when applied in label assignment.

Q: The number of GTs is known, and the number of anchors can be decided by experiences and dataset. To formulate this OTA, we need to decide the number of positive anchors for each GTs. How do we select an

[6] More about this method: https://zhuanlan.zhihu.com/p/10971105566

appropriate number of positive anchors for each GTs?

Dynamic k estimation:

For each GT, we select the top q predictions (usually $q = 10$) according to IoU values. These IoU values are summed up to represent this **GT's estimated number of positive anchors (this value is k)**. Such an estimation method is based on the following intuition: The appropriate number of positive anchors for a certain GT should be positively correlated with the number of anchors that well-regress this GT.

cost	样本点1	样本点2	样本点3	样本点4	样本点5	样本点6	
GT1	0.3	0.6	1.4	4.1	3.5	0.5	
GT2	1.5	3.2	0.3	0.1	0.9	2.4	
GT3	2.5	5.2	0.2	0.2	0.8	1.5	
IoU	样本点1	样本点2	样本点3	样本点4	样本点5	样本点6	求和并向下取整
GT1	0.3	0.6	0.4	0.7	0.5	0.5	3
GT2	0.5	0.2	0.3	0.1	0.9	0.4	2
GT3	0.5	0.2	0.2	0.2	0.8	0.8	2

e.g. for GT2, dynamic k = $\lfloor 0.5 + 0.2 + 0.3 + 0.1 + 0.9 + 0.4 \rfloor = \lfloor 2.4 \rfloor = 2$

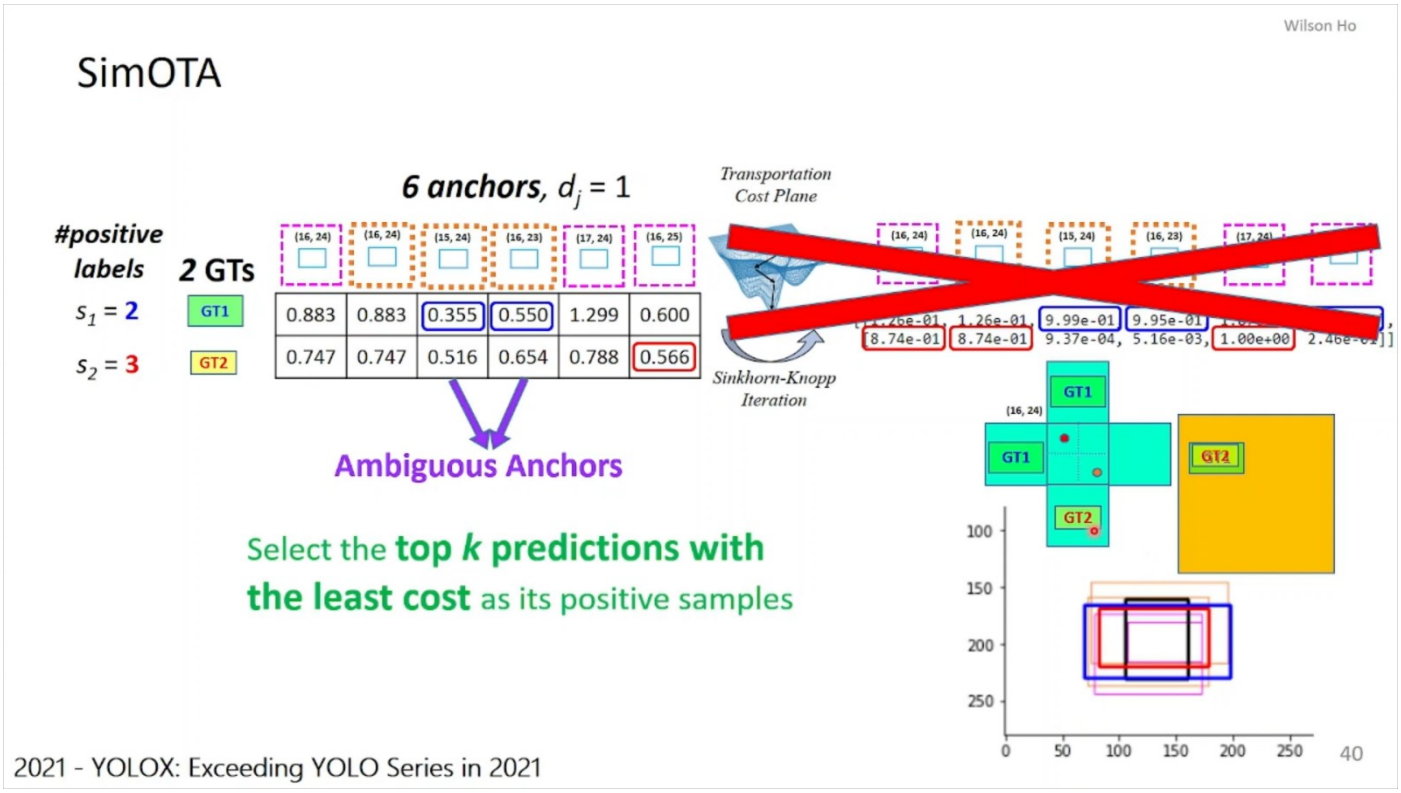
k	AP	AP_{50}	AP_{75}	APs	APm	APl
1	36.5	55.4	38.8	21.4	39.7	46.2
5	39.5	58.1	42.7	23.1	43.0	50.6
8	39.8	58.4	42.9	22.7	43.6	51.5
10	40.3	**58.6**	43.7	23.4	44.2	52.1
12	40.3	**58.6**	43.6	23.2	44.2	51.9
15	40.2	58.4	43.6	23.2	44.1	51.9
20	40.1	58.2	43.6	**23.5**	44.0	52.8
Dyn. k	**40.7**	58.4	**44.3**	23.2	**45.0**	**53.6**

Table 4. Analysis of different values of k and Dynamic k Estimation strategy on the COCO *val* set.

This table is from OTA paper, showing a better performance of using dynamic k estimation strategy.

SimOTA (from YOLOX, also used in YOLOv6):

Main idea: simplified version of OTA by obtaining an approximate solution to the Optimal Transport Assignment with reduction in training time. Simply ignore the iteration steps done in OTA and directly select the top-k predictions with the least cost (cost function: sum of regression loss and classification loss) as its positive sample. We will see in details how SimOTA is applied in YOLOv7.



YOLOv7 positive sample assignment:

①**YOLOv5:使用 yolov5 正负样本分配策略分配正样本**。

②**YOLOX:**计算每个样本对每个 GT 的 Regression + Classification Loss (Loss aware)

③**YOLOX:**使用每个 GT 的预测样本（跟據 IoU 数值）确定它需要分配到的正样本数 (Dynamic k estimation, mentioned in P.24)

④**YOLOX:**为每个 GT 取 Regression + Classification Loss 最小的前 dynamic k 个样本作为正样本

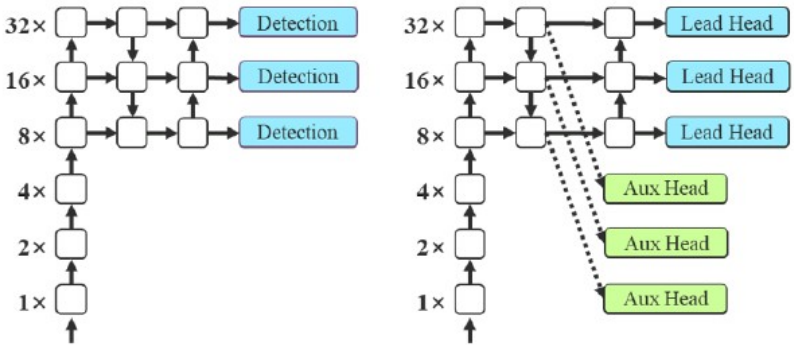
⑤**YOLOX:**人工去掉同一个样本被分配到多个 GT 的正样本的情况 (取 loss 最小的作為正樣本)

就是 SimOTA 中的第一步“使用中心先验” (anchor free) 替换成“yolov5 中的策略” (anchor based)。

通过查看源码，发现 **aux head 的 assigner 和 lead head 的 assigner 仅存在很少的不同**，包括：

①**lead head 中每个 anchor 与 GT 如果匹配上，分配 3 个正样本，而 aux head 分配 5 个。**

②**lead head 中将 top10 个样本 IoU 求和取整，而 aux head 中取 top20。 (top-q)**



Side note: The designed equivalence E-ELAN makes it easier for us to implement **partial auxiliary head**. E-ELAN with normal auxiliary head and partial auxilary head are shown in Figure A4.

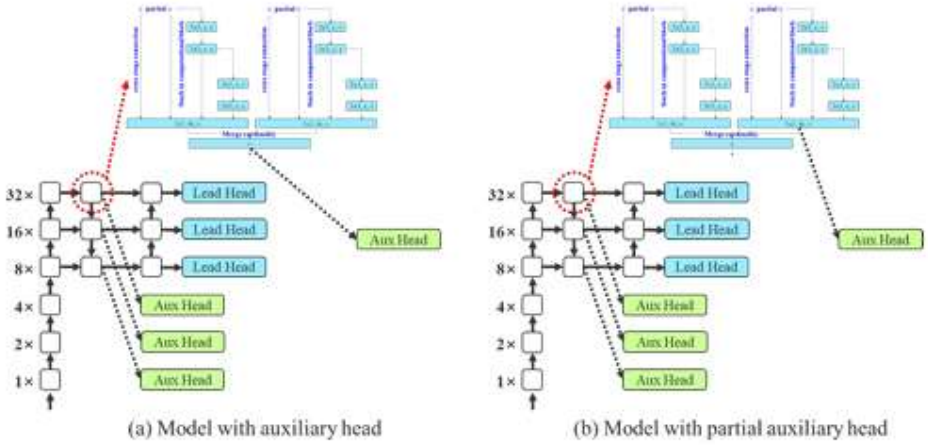


Figure A4. Partial auxilary head on E-ELAN.

Partial auxiliary head:

做法是將 partial auxiliary head 用於 E-ELAN 架構，在 **merging cardinality 操作之前**的 feature map 上進行連接。而這樣的連接操作可以使新的輸出 feature map 的權重不直接被 assistant loss 更新。下表 8 可知使用 partial auxiliary head 有更好的效果。

Table 8: Ablation study on partial auxiliary head.

Model	Size	AP^{val}	AP^{val}_{50}	AP^{val}_{75}
base (v7-E6E)	1280	56.3%	74.0%	61.5%
aux	1280	56.5%	74.0%	61.6%
partial aux	1280	**56.8%**	**74.4%**	**62.1%**
improvement	-	+0.5	+0.4	+0.6

Number of channels of output feature map: 255

Each grid has 3 anchor boxes. Each box has 4 coordinates, 1 objectness score, and 80 classification scores.
Total = 3 x (4 + 1 + 80) = 255

Loss function: Regression loss (CIoU loss) + objectness loss (BCE loss) + classification loss (BCE loss)

A weighting of 0.25 is multiplied to the loss calculated from aux head, and added to the loss calculated from lead head.

```
# Lead head
lbox += (1.0 - iou).mean()
lcls += self.BCEcls(ps[:, 5:], t)

# Aux head
lbox += 0.25 * (1.0 - iou_aux).mean()
lcls += 0.25 * self.BCEcls(ps_aux[:, 5:], t_aux)
```

3. Other trainable bag-of-freebies

3.1 Batch normalization in conv-bn-activation topology

將 batch normalization layer 連接在 CNN layer 後，目的是為了在推理運算時，能夠將 batch normalization layer 的平均數 m 與變異數 v 整合至 CNN layer 的 bias 和 weight 中。

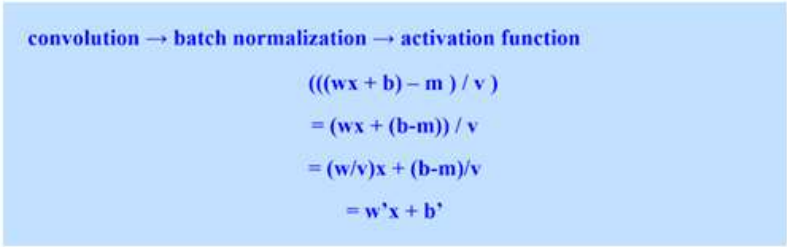
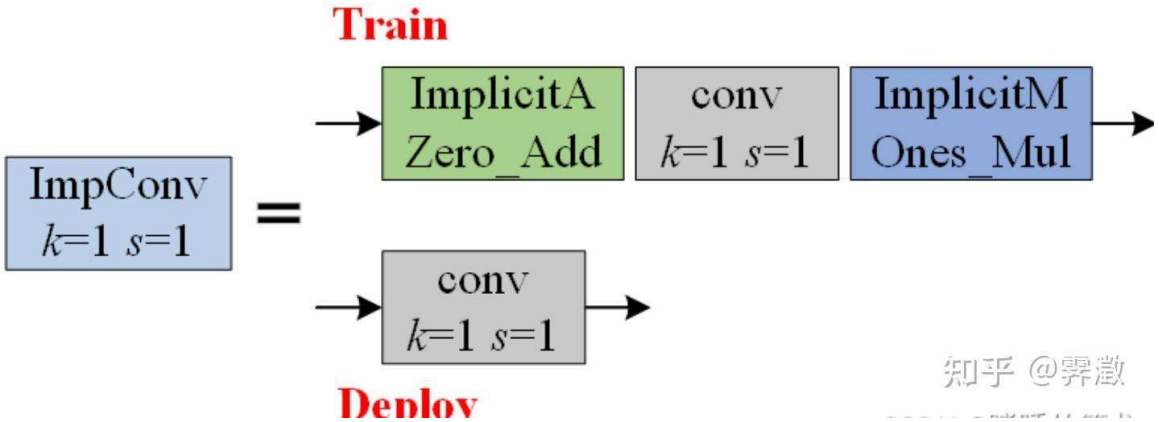


Figure A6. Batch normalization as trainable BoF.

3.2 Implicit knowledge in YOLOR combined with convolution feature map in addition and multiplication manner

在推論階段上，先將 YOLOR 的 Implicit knowledge (information/representation that are hidden in the data) 壓縮為一個 vector，如此一來能以 addition 或 multiplication 的方式跟前、後一個 CNN layer 的 bias 和 weight 結合。

s



This shows how the implicit knowledge vector $g_i(z_i)$ can be fused into the convolution operation.

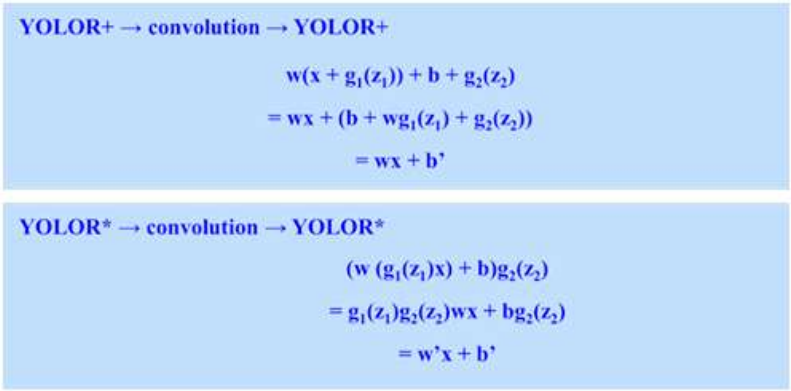


Figure A7. YOLOR implicit knowledge as trainable BoF.

Code implementation is in the class IDetect and IAuxDetect in the official GitHub repo.[7]

3.2.1 Implicit knowledge (YOLOR)[8]

The main component why YOLOv7 can do multiple tasks from the same model.

與前幾代 YOLO 模型不同，YOLOv7 最大特色是一心多用，看一眼就能執行 3 種任務，包括物件偵測任務 (object detection)、實例分割 (Instance segmentation) 以及關節點偵測(Keypoint estimation)。

原本專精單一任務的模型，透過保存起來的特徵 (ImplicitA, ImplicitM)，來學習其他任務，而不必針對其他新任務一一重新訓練。模型能夠學習到一些**通用的 representation**，從而只要對模型進行**微調**就能適應各種不同的任務。

---

[7] https://github.com/WongKinYiu/yolov7/blob/a207844b1ce82d204ab36d87d496728d3d2348e7/models/yolo.py#L97

[8] Explicit knowledge v.s. Implicit knowledge: 廖弘源 (Third author of YOLOv7) 比喻，就像是受過專業訓練的諜報員，**被賦予一個任務**，要找出照片中的車輛，他會專心記住照片中每輛車，這就是**顯性知識（Explicit knowledge）**，但若突然換個題目，改問照片中有幾位行人，他也能憑記憶正確回答，**即便一開始並未被賦予這個任務**；這就是**隱性知識**（Implicit knowledge）。

(c) illustrates the representation of data that can serve different tasks in YOLOR.

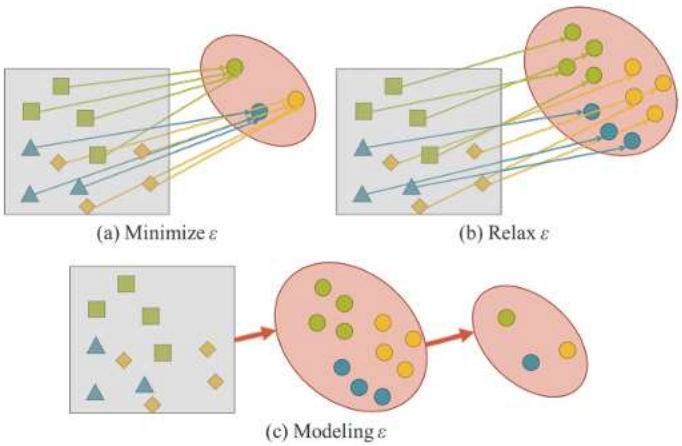


Figure 6: Modeling error term.

B.1.1 YOLOv7-mask

We integrate YOLOv7 with BlendMask [14] to do instance segmentation. We simply fine-tune YOLOv7 object detection model on MS COCO instance segmentation dataset and trained for 30 epochs. It achieves state-of-the-art real-time instance segmentation result. The architecture of YOLOv7-mask and the corresponding results are shown in Figure A10 (a) and Figure A11, respectively.

Figure A11. Sample results after applying YOLOv7-mask.

B.1.2 YOLOv7-pose

We integrate YOLOv7 with YOLO-Pose [15] to do keypoint detection. We follow the same setting as [15] to fine-tune YOLOv7-W6 people detection model on MS COCO keypoint detection dataset. YOLOv7-W6-pose achieves state-of-the-art real-time pose estimation result. The architecture and sample results are shown in Figure A10 (b) and Figure A12, respectively.

Figure A12. Sample results after applying YOLOv7-pose.

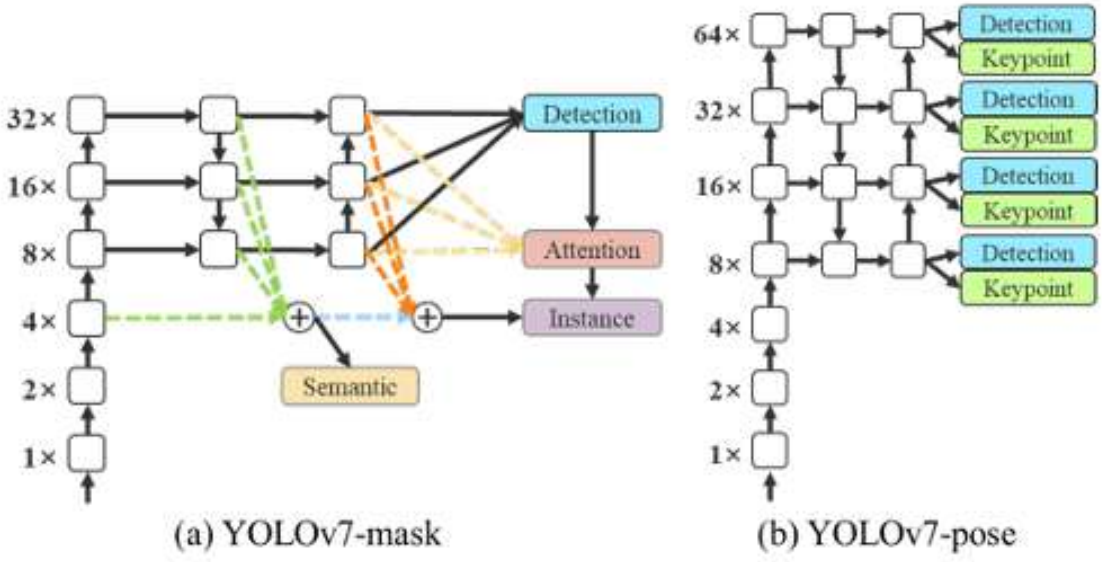
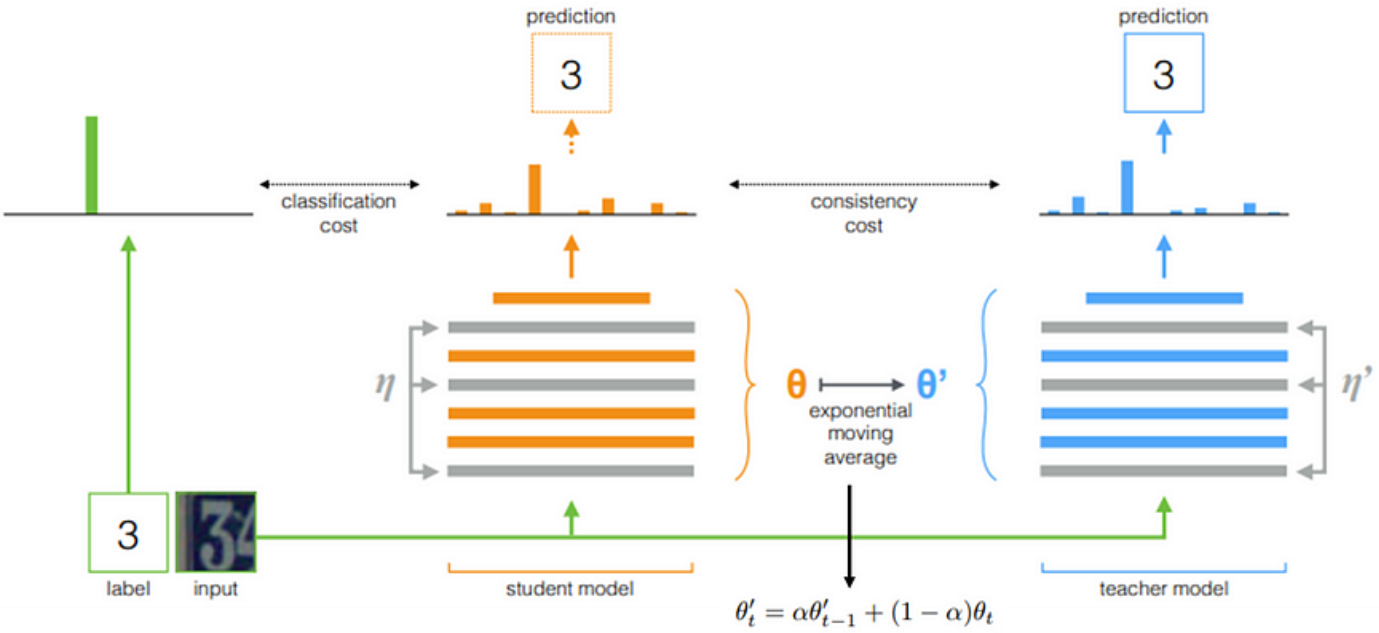


Figure A10. Architectures of YOLOv7-mask and YOLOv7-pose.

3.3 EMA model

EMA 為 Mean teachers are better role models 論文中使用到的技術，作者使用 EMA 模型作為最後的推理運算模型。

Mean teachers 是 Semi-Supervised Learning 的一種方法，其架構包含兩個相同結構的網路 teacher、student，teacher model 的參數是藉由 student model 的參數計算加權平均得到，而 student model 的參數則是透過 loss function gradient descent 更新。



References:

YOLO 演進 — YOLOv7 論文閱讀

https://medium.com/ching-i/yolo%E6%BC%94%E9%80%B2-yolov7-%E8%AB%96%E6%96%87%E9%96%B1%E8%AE%80-97b0e914bdbe

深入淺出之 YOLOv7

https://medium.com/@ericpeng9403/%E6%B7%B1%E5%85%A5%E6%B7%BA%E5%87%BA%E4%B9%8Byolov7-8907f9d05d72

【目标检测】Yolov7 的 ELAN 和 E-ELAN 模块演进（涉及到分组卷积，cardinality，梯度路径）

https://blog.csdn.net/Jiangnan_Cai/article/details/137231142

解读 2-YOLOV7（个人觉得非常精彩）

https://www.cnblogs.com/hansjorn/p/16964478.html

YOLOv7 E-ELAN 学习笔记

https://zhuanlan.zhihu.com/p/586922724

The evolution of the YOLO neural networks family from v1 to v7.

https://medium.com/deelvin-machine-learning/the-evolution-of-the-yolo-neural-networks-family-from-v1-to-v7-4d4fab3c4db7

YOLOv7 拆解

https://hackmd.io/@j3IW-o-4Q_qMBAVzm-ZAQA/BytibsDX3

概述 Model Reparameterization: RepVGG 與後續作 (RepOptimizer, QARepVGG, MobileOne)

https://medium.com/ai-blog-tw/%E6%A6%82%E8%BF%B0model-reparameterization-repvgg-%E8%88%87%E5%BE%8C%E7%BA%8C%E4%BD%9C-repoptimizer-qarepvgg-mobileone-e3b5d952a801

YOLOV7: from zero to hero

https://zhuanlan.zhihu.com/p/670643999

YOLOv7 Paper Appendix

Wang_YOLOv7_Trainable_Bag-of-Freebies_CVPR_2023_supplemental.pdf

圖解 YOLOv7 architecture (1/2)

https://www.youtube.com/watch?v=Ot__47ItjDs

圖解 YOLOv7 loss (2/2)

https://www.youtube.com/watch?v=EhXwABGhBrw