Comparing Scaled-YOLOv4 & YOLOv7

TensorFlow Model Garden

J. Setpal

November 2, 2022

Outline

Outline

General Idea – Scaled-YOLOv4

a. FCOS based model – unlike YOLOv1-3, this does not rely on pre-defined anchor boxes.

General Idea – Scaled-YOLOv4

a. FCOS based model – unlike YOLOv1-3, this does not rely on pre-defined anchor boxes.

b. Places an extended focus on neural scaling to design an efficient system.

General Idea – Scaled-YOLOv4

a. FCOS based model – unlike YOLOv1-3, this does not rely on pre-defined anchor boxes.
b. Places an extended focus on neural scaling to design an efficient system.
c. It **analyzes the gradient path**, enabling the weights of different layers to learn diverse features.

General Idea – Scaled-YOLOv4

a. FCOS based model – unlike YOLOv1-3, this does not rely on pre-defined anchor boxes.

b. Places an extended focus on neural scaling to design an efficient system.

c. It **analyzes the gradient path**, enabling the weights of different layers to learn diverse features. This makes inference faster and more accurate.

General Idea – Scaled-YOLOv4

a. FCOS based model – unlike YOLOv1-3, this does not rely on pre-defined anchor boxes.

b. Places an extended focus on neural scaling to design an efficient system.

c. It **analyzes the gradient path**, enabling the weights of different layers to learn diverse features. This makes inference faster and more accurate.

d. Scaled-YOLOv4 aims to find a method for *synergistic compound scaling* (translation: $\pm$ stages) based on design requirements for object-detection based tasks.

General Idea – YOLOv7

YOLOv7 builds on ELAN. It uses **expand, shuffle and merge cardinalities** to continuously enhance the learning ability of the network <u>without</u> destroying the original gradient path.

It uses the following factors for neural scaling:

- Memory access cost

General Idea – YOLOv7

YOLOv7 builds on ELAN. It uses **expand, shuffle and merge cardinalities** to continuously enhance the learning ability of the network <u>without</u> destroying the original gradient path.

It uses the following factors for neural scaling:

- Memory access cost
- I/O channel ratio

General Idea – YOLOv7

YOLOv7 builds on ELAN. It uses **expand, shuffle and merge cardinalities** to continuously enhance the learning ability of the network <u>without</u> destroying the original gradient path.

It uses the following factors for neural scaling:

- Memory access cost
- I/O channel ratio
- Element-wise operation

General Idea – YOLOv7

YOLOv7 builds on ELAN. It uses **expand, shuffle and merge cardinalities** to continuously enhance the learning ability of the network without destroying the original gradient path.

It uses the following factors for neural scaling:

- Memory access cost
- I/O channel ratio
- Element-wise operation
- Activations

General Idea – YOLOv7

YOLOv7 builds on ELAN. It uses **expand, shuffle and merge cardinalities** to continuously enhance the learning ability of the network without destroying the original gradient path.

It uses the following factors for neural scaling:

- Memory access cost
- I/O channel ratio
- Element-wise operation
- Activations
- Gradient path

What's model scaling?

Traditionally, **model scaling** refers to the idea of modifying the depth of an architecture ($\pm$ hidden layers).

What's model scaling?

Traditionally, **model scaling** refers to the idea of modifying the depth of an architecture ($\pm$ hidden layers).

That definition has since broadened: covering depth, width, bottleneck ratio, slope, quantize, group width, input size as well as the types of layers.

What's model scaling?

Traditionally, **model scaling** refers to the idea of modifying the depth of an architecture ($\pm$ hidden layers).

That definition has since broadened: covering depth, width, bottleneck ratio, slope, quantize, group width, input size as well as the types of layers.

Popular methods for neural scaling like NAS perform parameter-specific scaling. The scaling factors are independent in our case.

What's model scaling?

Traditionally, **model scaling** refers to the idea of modifying the depth of an architecture ($\pm$ hidden layers).

That definition has since broadened: covering depth, width, bottleneck ratio, slope, quantize, group width, input size as well as the types of layers.

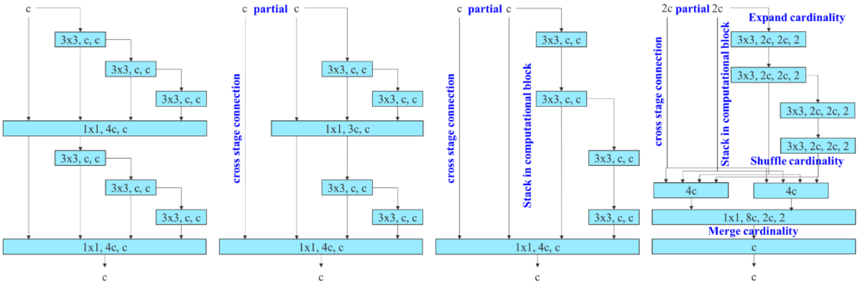
Popular methods for neural scaling like NAS perform parameter-specific scaling. The scaling factors are independent in our case.

Both Scaled-YOLOv4 and YOLOv7 build upon neural scaling as a method for improving inference speed and accuracy.

Outline

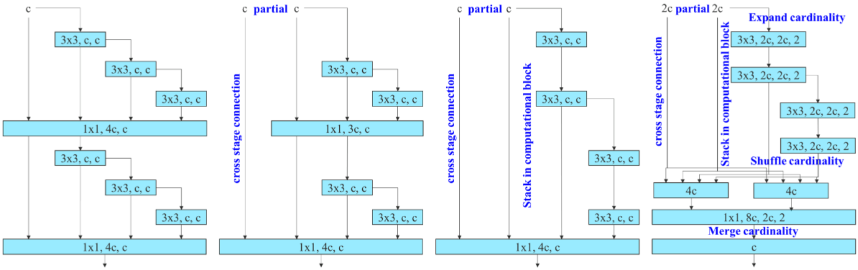
1. Overview
2. Architecture

Comparing Architectures



(a) VoVNet [39] (b) CSPVoVNet [79] (c) ELAN [1] (d) E-ELAN

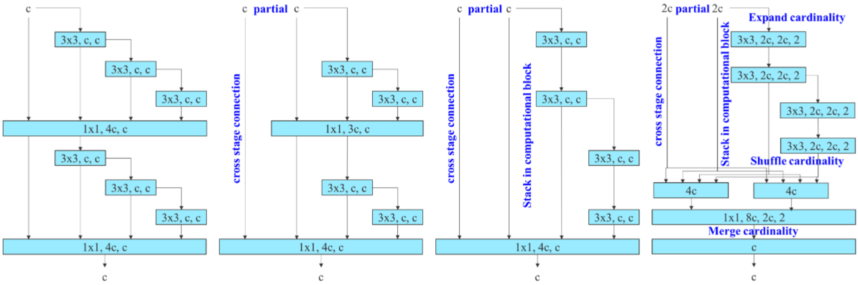
Comparing Architectures



(a) VoVNet [39] (b) CSPVoVNet [79] (c) ELAN [1] (d) E-ELAN

- **VoVNet:** DenseNet with concatenations restricted to the end.

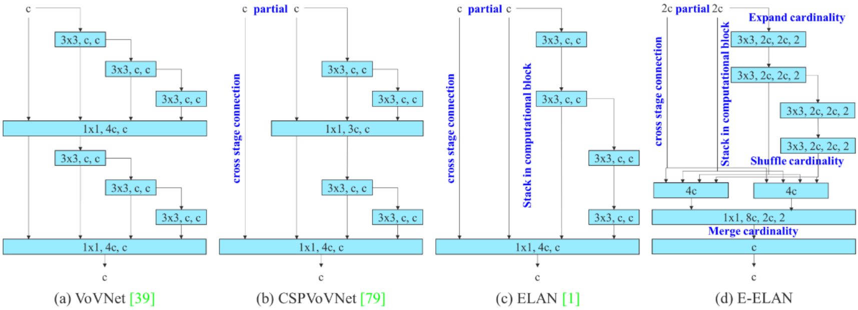
Comparing Architectures



(a) VoVNet [39] (b) CSPVoVNet [79] (c) ELAN [1] (d) E-ELAN

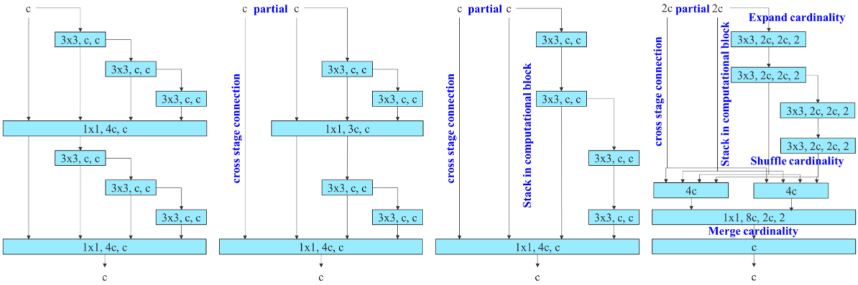
- **VoVNet:** DenseNet with concatenations restricted to the end.
- **CSPVoVNet:** CSP-ized the PAN architecture, reverted the initial DarkNet layer to the original.

Comparing Architectures



- **VoVNet:** DenseNet with concatenations restricted to the end.
- **CSPVoVNet:** CSP-ized the PAN architecture, reverted the initial DarkNet layer to the original.
- **ELAN:** Controlling the longest-shortest gradient path allows a deeper network to effectively converge.

Comparing Architectures



(a) VoVNet [39] (b) CSPVoVNet [79] (c) ELAN [1] (d) E-ELAN

- **VoVNet:** DenseNet with concatenations restricted to the end.
- **CSPVoVNet:** CSP-ized the PAN architecture, reverted the initial DarkNet layer to the original.
- **ELAN:** Controlling the longest-shortest gradient path allows a deeper network to effectively converge.
- **E-ELAN:** Group convolution to increase cardinality.

YOLOv7 – Trainable Bag-of-Freebies

Trainable Bag-of-Freebies are methods that increase performance without increasing training cost.

YOLOv7 – Trainable Bag-of-Freebies

Trainable Bag-of-Freebies are methods that increase performance <u>without</u> increasing training cost.

a. **Planned Re-Parameterized Convolution:**

YOLOv7 – Trainable Bag-of-Freebies

Trainable Bag-of-Freebies are methods that increase performance without increasing training cost.

a. **Planned Re-Parameterized Convolution:**
    - Can be performed at a model level and *module level*.

YOLOv7 – Trainable Bag-of-Freebies

Trainable Bag-of-Freebies are methods that increase performance <u>without</u> increasing training cost.

a. **Planned Re-Parameterized Convolution:**
   - Can be performed at a model level and *module level*.
   - Running a re-parameterization routine individually on modules and using the prepared ensemble as the final model.

YOLOv7 – Trainable Bag-of-Freebies

Trainable Bag-of-Freebies are methods that increase performance <u>without</u> increasing training cost.

a. **Planned Re-Parameterized Convolution:**
   - Can be performed at a model level and *module level*.
   - Running a re-parameterization routine individually on modules and using the prepared ensemble as the final model.

b. **Coarse for Auxilliary and Fine for Lead Loss:**

YOLOv7 – Trainable Bag-of-Freebies

Trainable Bag-of-Freebies are methods that increase performance without increasing training cost.

a. **Planned Re-Parameterized Convolution:**
   - Can be performed at a model level and *module level*.
   - Running a re-parameterization routine individually on modules and using the prepared ensemble as the final model.

b. **Coarse for Auxilliary and Fine for Lead Loss:**
   - Contains a *lead head, auxilliary head* and a *soft label assigner*.

YOLOv7 – Trainable Bag-of-Freebies

Trainable Bag-of-Freebies are methods that increase performance <u>without</u> increasing training cost.

a. **Planned Re-Parameterized Convolution:**
    - Can be performed at a model level and *module level*.
    - Running a re-parameterization routine individually on modules and using the prepared ensemble as the final model.

b. **Coarse for Auxilliary and Fine for Lead Loss:**
    - Contains a *lead head, auxilliary head* and a *soft label assigner*.
    - The lead head predicts the final results, soft labels are generated based on them.

YOLOv7 – Trainable Bag-of-Freebies

Trainable Bag-of-Freebies are methods that increase performance without increasing training cost.

a. **Planned Re-Parameterized Convolution:**
   - Can be performed at a model level and *module level*.
   - Running a re-parameterization routine individually on modules and using the prepared ensemble as the final model.

b. **Coarse for Auxilliary and Fine for Lead Loss:**
   - Contains a *lead head, auxilliary head* and a *soft label assigner*.
   - The lead head predicts the final results, soft labels are generated based on them.
   - Loss is calculated for both the lead head and the auxiliary head based on the same soft labels that are generated.

YOLOv7 – Trainable Bag-of-Freebies

Trainable Bag-of-Freebies are methods that increase performance without increasing training cost.

a. **Planned Re-Parameterized Convolution:**
   - Can be performed at a model level and *module level*.
   - Running a re-parameterization routine individually on modules and using the prepared ensemble as the final model.

b. **Coarse for Auxilliary and Fine for Lead Loss:**
   - Contains a *lead head, auxilliary head* and a *soft label assigner*.
   - The lead head predicts the final results, soft labels are generated based on them.
   - Loss is calculated for both the lead head and the auxiliary head based on the same soft labels that are generated.
   - The lead head has a relatively strong learning capability; the auxilliary head eases it so that it can focus on learning residual information.

Thank you!

That's all I've got; have an awesome rest of your day!