# 使用 General Loss 的 SPDCN

## 出自

## 題目

**SPDCN**

SP ⇒ 尺度優先 (Scale-Prior) : **引用物體尺度的資訊**

D ⇒ **(Deformable Convolution)**：CNN 的 Kernel 不是常規滑過，而是會有**偏移**

Exemplar-Guided ⇒ 會有幾張 refference images

## Introduction

### 背景

### 動機

過去的方法 extracted features 不夠 robust

## 目的

integrating exemplars’ scale information

traditional L2 and generalized loss 無法處理 object scale 變化大的問題 ⇒ 提出 scale-sensitive generalized loss

# 相關研究

GMN : resize refference images 然後抽向量去 query 中找

BMNet : adds a scale embedding

FamNet : 從圖片裡面找 refference images， 並且給予 datasets FSC-147

使用 pretrained 好的 model 並凍結他的參數來抽取 feature， 這個 model 中有一個特色，就是同一層 conv 會用不同的 kernel 來抽取 feature， 以達到 scaling 的效果

L2 loss and Bayesian loss [18] are special
cases of the generalized loss.

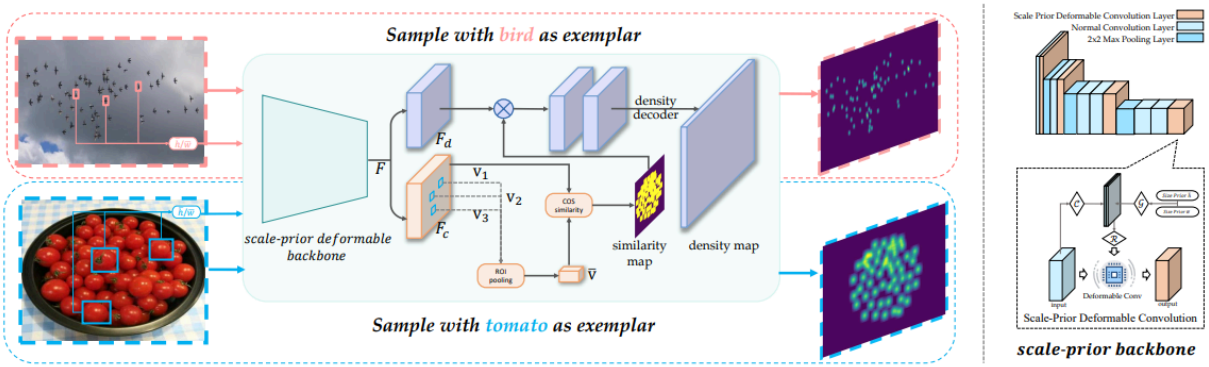
scale-sensitive generalized loss

different object categories have different sizes.

# 方法

SPDCN ⇒ 接收 scale 資訊調整 deform convolution 的 receptive field

refference 與 query image 算 similarity ⇒ generated similarity map ⇒ decoder to estimate the density map.




(1) scale-prior backbone;

VGG-19 的前面十層

某些層被替換為 scale-prior deformable convolutions， 來抽 scale 資訊之後用

用 linear function 把 F 變成 Fc 與 Fd

用 refference image 來學 scale embeddings, 用它來調整 receptive field

一般 CNN 是

For a convolution kernel $\mathbf{w}$ with size of $2r+1$, the output value $\mathbf{y}(p)$ while convolving it with input feature map $\mathbf{x}$ at location $p$ is calculated as:

$$\mathbf{y}(p) = \sum_{j \in \mathcal{R}} \mathbf{w}(j) \cdot \mathbf{x}(p+j), \ \mathcal{R} = \{(-r,-r),(-r,-r+1),\cdots,(r,r-1),(r,r)\}. \quad (2)$$

Deformable CNN 是，多的 delta j 稱為 offset

$$\mathbf{y}(p) = \sum_{j \in \mathcal{R}} \mathbf{w}(j) \cdot \mathbf{x}(p+j+\Delta j).$$

SPDCN train 的 offset 由兩個部分組成 ⇒ the local scale embedding dc and the global scale embedding dg.

dc 求法(這是原本 Defromable CNN 的算法，由原圖 end to end 生成 offset)

$$d_c = \mathcal{C}(\mathbf{x})$$

C , g, R 是一個 conv

dg : 只和 refference image 的大小有關

$$d_g = \mathcal{G}(\bar{h}, \bar{w}), \quad \bar{h} = \sum_{e_i \in E_I} \frac{h_{e_i}}{|E_I|}, \bar{w} = \sum_{e_i \in E_I} \frac{w_{e_i}}{|E_I|}, \tag{4}$$

where $E_I$ is the exemplar set, $h_{e_i}$ and $w_{e_i}$ is the height and width of the $i$-th exemplar $e_i$.

dc 與 dg 接起來再過 R

(2) counted objects segmentation module (orange part);

ROIAlign layer to extract semantic vectors {v1,⋯ , vn} n 是 refference 的數目

並將 這些 v 平均成 v_bar 代表該類別

class-specific representation vector

cosine similarity

$$\tilde{s}_i = \frac{\bar{v}^\top f_i}{\|\bar{v}\|_2 \cdot \|f_i\|_2}, \; f_i \in F_c,$$

(3) class-agnostic density prediction module (blue part).

再將 si~ 依照 fi 的位置放回去，就可以得到一張 similarity 圖，表示哪些地方與 refference image 高相似

Decoder : PSCC ⇒ 用 Pixel Shuffling 來 upsample

H×W×Cin ⇒ H x W x (Cout x r x r) ⇒ (H x r) x (W x r) x Cout

避免棋盤格效應 ⇒ 因為不是將圖片擴大生成，而是使用重組的方法

也被稱為「亞像素卷積 (sub-pixel convolution)」，因為是一個 pixel 的好多 channel 去重組

Fd 與 S~ 做 element-wised 乘積

Generalized loss

measures the distance between
the predicted density map and dot map through an unbalanced optimal transport problem

$$\mathcal{L}_{\mathbf{C}} = \min_{\mathbf{P}} \langle \mathbf{C}, \mathbf{P} \rangle - \varepsilon H(\mathbf{P}) + \tau \|\mathbf{P}\mathbf{1}_m - \mathbf{a}\|_2^2 + \tau \|\mathbf{P}^\top \mathbf{1}_n - \mathbf{b}\|_1$$

試圖找到一個最佳的傳輸計畫 P，將**預測的密度圖（a）**與**目標（b）**之間進行匹配。

<> 表示內積：矩陣對應元素相乘，然後將所有乘積加總。

$$\langle C, P \rangle = \sum_{i=1}^{M} \sum_{j=1}^{N} C_{ij} \cdot P_{ij}$$

矩陣 P ：每個元素 Pij 表示從源點 i 運輸到目標點 j 的**質量數量**

矩陣 C：每個元素 Cij 表示將第 i 個源點的質量運輸到第 j 個目標點的**成本**

$$H(P) = \sum_{i,j} P_{ij} \log(P_{ij})$$

每個元素取 log 後乘以自己再全部相加

其不確定性或混亂程度

避免 P 中的值過於稀疏（即只在少數幾個 P
ij

上有非零值）。

值會是負數或零。

當 P 矩陣非常稀疏（即大部分 P ij

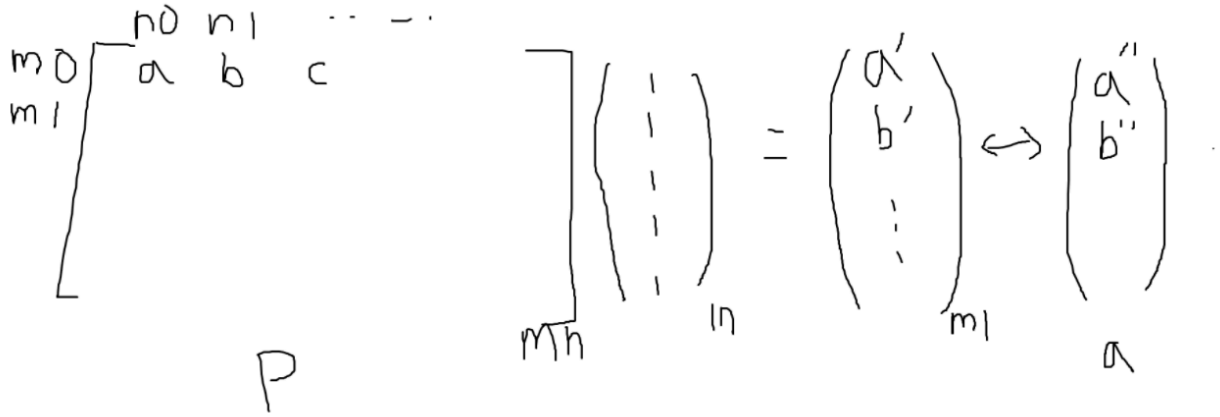
都是 0，只有少數幾個地方有質量傳輸）時，H(P) 的值會接近 0（這是「最大」值，因為負數越接近 0 越大）。

- $\mathbf{1}_m$ 和 $\mathbf{1}_n$：
  - $\mathbf{1}_m$ 是一個**$m$ 維的全一列向量**，即 $\begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}$。
  - $\mathbf{1}_n$ 是一個**$n$ 維的全一列向量**。

P = M * N ， M 是 pred density maps 的像素數量， N 是 target density maps 的數量

P1n ⇒ pred 每個像素運出去的質量

PT1m ⇒ target 每個像素接收的質量



所以 P1n - a 就是希望 P 這個計畫中從 source 搬出去的總質量應該要和 pred 的越近越好，不要移的太多，也不要移的太少

把 pred 中的像素移動到 那 N 點就好 (因為不會移動到空白)

# Experiments

# 可以學習的地方

# 問題

OT(α, β) (Optimal Transport ) = <C * P> (arg min P，其中 P 屬於 PI(alpha, beta)，PI 是所有可能的 P)

alpha 是要搬走的質量， shape 是 N, 1

beta 是要接收的質量， shape 是 M, 1

P 是一個 joint distribution P(alpha, beta)

P 的 margin 分別是 alpha 以及 beta

PIij 指的是 P 上 i row j column 的元素

Hard Constraint :

alpha = A, beta = B