Minimum redundancy Constraint

مقاله

- concate features from different layers along the channel axis.

Feature weigted fusion

$$f = \sum_{i=1}^{K} w_i f_i$$

f_i → feature moule i

w_i → The weight of the ith featre, determined using cross-validation methods

$$\Phi_{j \in \{1,2,3,4\}} = \Phi(x), \quad x \in X_N$$

X_N → normal samples

Minimum Redundancy

قید

معادله ۲ مقاله

$$Rel(f_i) = \frac{1}{n} \left(\sum_{t=1}^{n} \underline{Rel(f_i ; f_t)} \right)$$

$Rel(f_i ; f_t)$ → میزان شباهت f_i و f_t

مراحل ساخت coreset در این روش:

F ← تعداد کل ویژگی‌های مجموعه training و M زیرمجموعه نماینده ویژگی‌ها

۱. یک ویژگی f به صورت رندوم از فضای ویژگی F انتخاب و به عنوان مقدار اول M در نظر گرفته می‌شود - این ویژگی از F حذف می‌شود - میزان redundancy آن (معادله ۲) با همه ویژگی‌های دیگر محاسبه می‌شود و ویژگی با کمترین شباهت به M اضافه می‌شود.

۲. فاصله بین باقی ویژگی‌های F و زیرمجموعه M محاسبه می‌شود (فاصله بین هر ویژگی و زیرمجموعه M را به عنوان مینیمم فاصله بین آن ویژگی و همه ویژگی‌های M تعریف می‌شود

۳- مرحله ۲ تا زمانیکه تعداد ویژگیها به threshold t برسد تکرار می‌شود.

۴- مراحل ۱، ۲، ۳ را چندین بار تکرار می‌کنیم. سپس آنتروپی اطلاعات هر زیرمجموعه را محاسبه می‌کنیم و زیرمجموعه با کوچکترین (کمترین) آنتروپی اطلاعات را به عنوان زیرمجموعه بهینه انتخاب می‌کنیم.

روش محاسبه میزان شباهت ویژگی

$Rel(X,Y) \Rightarrow$ correlation between two vectors

$X = (x_1, x_2, \ldots, x_n)$ و

$Y = (y_1, y_2, \ldots, y_n)$

فاصله کوواریانس

$$R(X,Y) = \begin{cases} \sqrt{\dfrac{V^2(X,Y)}{\sqrt{V^2(X)V^2(Y)}}} & V^2(X)V^2(Y) > 0 \\ 0 & V^2(X)V^2(Y) = 0 \end{cases}$$

$V(X,Y) \Rightarrow$ فاصله کوواریانس

$V(X) \Longrightarrow$ فاصله واریانس

Minimum Redundancy اعمال قید
constraint

MVTec ⟹ carpet, tile, wood

corrugated pipe dataset

pre-trained ⟹ ResNet 50

layer 2 + 3 + 4 or layer 2 + 3
ترکیب بهتری بوده است

Tile ⟹ 97,69 | AUC

Carpet ⟹ 99,17

wood ⟹ 97,17

system windows 11

Nvidia 4070

pytorch

۱ فاصله واریانس $\Rightarrow X = \{x_1, x_2, \dots, x_n\}$

فاصله جفت عناصر $A_{ij} = \| x_i - x_j \|$

ماتریس فاصله مرکزی $\Rightarrow \tilde{A}_{ij} = \underbrace{A_{ij} - \bar{A}_{i.}}_{\text{میانگین ردیف i}} - \underbrace{\bar{A}_{.j}}_{\text{میانگین ستون j}} + \underbrace{\bar{A}_{..}}_{\text{میانگین همه المنت‌های A}}$

$$\Rightarrow V^2(X) = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \tilde{A}_{ij}^2$$

۲ فاصله کوواریانس $\Rightarrow X , Y$

$$A_{ij} = \| x_i - x_j \| , \quad B_{ij} = \| y_i - y_j \|$$

$$V^2(X,Y) = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \tilde{A}_{ij} \tilde{B}_{ij}$$

۳ فاصله کورلیشن $\Rightarrow R(X,Y) = \dfrac{V(X,Y)}{\sqrt{V(X)V(Y)}}$ if $V(X)V(Y) > 0$

$X_N \rightarrow$ Train کل تصاویر نرمال

مقاله Patchcore

$\Phi_{i,j} = \Phi_j(x_i)$ ویژگی نمونه iام در سطح jام شبکه

$j \in [2,3]$

فرض کنید

feature map $\Rightarrow \Phi_{i,j} \in \mathbb{R}^{\overset{\text{عمق}}{c^*} \times \overset{\text{ارتفاع}}{h^*} \times \overset{\text{عرض}}{w^*}}$

$$\Phi_{i,j}(h,w) = \Phi_j(x_i, h, w) \in \mathbb{R}^{c^*}$$

بعد c^* المانی ویژگی در موقعیت (h,w)

به ازای patch size p :

همسایگی $\mathcal{N}_p^{\overset{\text{position}}{(h,w)}} = \{(a,b) \mid a \in [h - \lfloor p/2 \rfloor, \dots, h + \lfloor p/2 \rfloor],\ b \in [w - \lfloor p/2 \rfloor, \dots, w + \lfloor p/2 \rfloor]\}$

locally aware features at position (h,w)

$$\Phi_{i,j}(\mathcal{N}_p^{(h,w)}) = f_{agg}(\{\Phi_{i,j}(a,b) \mid (a,b) \in \mathcal{N}_p^{(h,w)}\})$$

adaptive average pooling

بنابراین patch-feature را می‌توانیم به صورت زیر در نظر بگیریم:

$$P_{s,p}(\Phi_{i,j}) = \{\Phi_{i,j}(\mathcal{N}_p^{(h,w)})|$$

$$h, w \bmod \underbrace{s}_{\text{striding parameter}} = 0$$

$$h < h^*$$

$$w < w^*$$

معمولا پیش‌فرض آن 1 است

همچنین سطح j+1 را نیز مطابقه می‌کنند:

$$P_{s,p}(\Phi_{i,j+1})$$

$$\hookrightarrow |P_{s,p}(\Phi_{i,j+1})| \underset{\text{with}}{\text{match}} |P_{s,p}(\Phi_{i,j})|$$

bilinearly rescaling

$$\text{memory bank } \mathcal{M} = \bigcup_{x_i \in \mathcal{X}_N} P_{s,p}(\Phi_j(x_i))$$

$$\mathcal{M}_C^* = \underset{\mathcal{M}_C \subset \mathcal{M}}{\arg\min} \max_{m \in \mathcal{M}} \min_{n \in \mathcal{M}_C} \|m - n\|_2$$

حل دقیق $\mathcal{M}_C^*$ سخت و NP-Hard است، به جای آن از روش تقریبی استفاده کرده‌اند

patchcore الگوریتم

Pretrained Φ, heirachies j,

X_N, stride s, patch size p

coreset target l, random linear projection ψ

$M \leftarrow \{\}$

for $x_i \in X_N$ do

$M \leftarrow M \cup P_{s,p}(\Phi_j(x_i))$

end

$M_C \leftarrow \{\}$

for $i \in [0, \dots, l-1]$ do

$m_i \leftarrow \underset{m \in M - M_C}{\arg\max} \ \underset{n \in M_C}{\min} \ \| \psi(m) - \psi(n) \|_2$

$M_C \leftarrow M_C \cup \{m_i\}$

end

$M \leftarrow M_C$