Unsupervised Data Augmentation (UDA)

ⓟ Unsupervised Data Augmentation (UDA)

UDA : Target model $p_\theta(y|x)$

$p_L(x)$ // Labeled data dist

$p_U(x)$ // Unlabeled data dist.

Perfect model f^*

Supervised Augmentation : $\hat{x} \sim q(\hat{x}|x)$

UDA : input x : $p_\theta(y|x)$, $p_\theta(y|x,\epsilon)$ → minimize Divergence → $D(p_\theta(y|x) \| p_\theta(y|x,\epsilon))$

ϵ : noise

$\hat{x} = q(x, \epsilon)$ ← quality??

Total Objective

$$\min_\theta J(\theta) = \underbrace{\mathbb{E}_{x_1 \sim p_L(x)} \left[-\log p_\theta(f^*(x_1)|x_1) \right]}_{\text{Supervised}} + \lambda \underbrace{\mathbb{E}_{x_2 \sim p_U(x)} \mathbb{E}_{\hat{x} \sim q(\hat{x}|x_2)} \left[CE\left(p_{\tilde{\theta}}(y|x_2) \| p_\theta(y|\hat{x}) \right) \right]}_{\text{unsupervised}}$$

$\tilde{\theta}$: fixed copy (no grad), θ : updated network

Sharpening Prediction : indicator

$$\frac{1}{|B|} \sum_{x \in B} I\left(\max_{y'} p_{\tilde{\theta}}(y'|x) > \beta \right) CE\left(p_{\tilde{\theta}}^{sharp}(y|x) \| p_\theta(y|\hat{x}) \right)$$

$$p_{\tilde{\theta}}^{sharp}(y|x) = \frac{\exp(z_y/\tau)}{\sum_{y'} \exp(z_{y'}/\tau)}$$

z_y → logit label for y

Theory:

In-domain: $P_U(\hat{x}) > 0$ for $\hat{x} \sim q(\hat{x}|x)$, $x \sim P_U(x)$

Label preserving: $f^*(x) = f^*(\hat{x})$ for $q(\hat{x}|x)$; $x \sim P_U(x)$

Reversible: if $q(\hat{x}|x) > 0$; then $q(x|\hat{x}) > 0$

Theorem: under UDA, $\mathrm{Prr}(A)$: Algo. can't infer the label of new test example | n labeled example from P_L

↳ geometric (succed after n toy)

$$P_r(A) = \sum_i P_i (1 - P_i)^n \quad \text{// Prob bound}$$

$$= \sum_{x \in C_i} P_L(x)$$

// labeled component.

P_i ⇒ observed example fall in i-th component

component number

Further, if $n = O(k/\epsilon)$ ⇒ $\mathrm{Prr}(A) = O(\epsilon)$

Error Rate..

⑤ SAM — SAM

Training Dataset $S = \bigcup_{i=1}^{n} \{(x_i, y_i)\}$ from $\mathcal{D}$ i.i.d

model parameters $w \in W \subseteq \mathbb{R}^d$

Data point loss function $l : W \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}_+$

Training loss: $L_S(w) \triangleq \frac{1}{n} \sum_{i=1}^{n} l(w, x_i, y_i)$

Population loss: $L_{\mathcal{D}}(w) \triangleq \mathbb{E}_{(x,y)\sim\mathcal{D}}[l(w, x, y)]$

Theorem: $\rho > 0$, w.h.p. over training set S from $\mathcal{D}$ (hyper param.)

$$L_{\mathcal{D}}(w) \leq \max_{\|\epsilon\|_2 \leq \rho} L_S(w+\epsilon) + h\left(\|w\|_2^2/\rho^2\right)$$

Regularization $\lambda\|w\|_2$ — can somehow do it !!

h: strictly increasing function $(\mathbb{R}_+ \to \mathbb{R}_+)$

Rewriting conditioned on $L_{\mathcal{D}}$

$$\left[\max_{\|\epsilon\|_2 \leq \rho} L_S(w+\epsilon) - L_S(w)\right] + L_S(w) + h\left(\|w\|_2^2/\rho\right)$$

sharpness — connected to the bound

So SAM optimization problem:

$$\min_w L_S^{SAM}(w) + \lambda\|w\|_2^2 \quad // \quad L_S^{SAM}(w) = \max_{\|\epsilon\|_p \leq \rho} L_S(w+\epsilon)$$

↳ p norm = $[1, \infty]$

finding L_S^{SAM} via optimization

$$\epsilon^*(w) \triangleq \arg\max_{\|\epsilon\|_p \leq \rho} L_S(w+\epsilon) \approx \arg\max_{\|\epsilon\|_p \leq \rho} L_S(w) + \epsilon^T \nabla_w L_S(w)$$

$$= \arg\max_{\|\epsilon\|_p \leq \rho} \epsilon^T \nabla_w L_S(w)$$

Solution revolves math:

Elementwise operation

$$\hat{\epsilon}(w) = \rho \operatorname{sign}\left(\nabla_w L_S(w)\right) \left|\nabla_w L_S(w)\right|^{q-1} \Big/ \left(\|\nabla_w L_S(w)\|_q^q\right)^{1/p}$$

where $\frac{1}{p} + \frac{1}{q} = 1$

norm

ρ as max value

Now,

$$\nabla_w L_S^{SAM}(w) \approx \nabla_w L_S\left(w + \hat{\epsilon}(w)\right)$$

$$= \frac{d\left(w + \hat{\epsilon}(w)\right)}{dw} \nabla_w L_S(w)\Big|_{w+\hat{\epsilon}(w)}$$

at the point $w + \hat{\epsilon}(w)$

$$= \nabla_w L_S(w)\Big|_{w+\hat{\epsilon}(w)} + \frac{\partial \hat{\epsilon}(w)}{\partial w} \nabla_w L_S(w)\Big|_{w+\hat{\epsilon}(w)}$$

Dropped !!

for acceleration : Dropping the second term.

$$\nabla_w L_S^{SAM}(w) \approx \nabla_w L_S(w)\Big|_{w+\hat{\epsilon}(w)}$$

① Meta pseudo labels: Meta pseudo-labels

Teacher, $T \rightarrow \theta_T$ ⊞ Data labeled (x_L, y_L)

Student, $S \rightarrow \theta_S$

Notation Soft prediction $T(x_L; \theta_T)$ ← Network output

↓ Input data, ↘ Net params.

Pseudo label Optimization: (Review)

$$\theta_S^{PL} = \arg\min_{\theta_S} E_{x_u}\left[CE\left(T(x_u; \theta_T), S(x_u; \theta_S)\right)\right]$$

$$:= \mathcal{L}_u(\theta_T, \theta_S)$$

$$E_{x_L, y_L}\left[CE\left(y_L, S(x_L; \theta_S^{PL})\right)\right] := \mathcal{L}_L(\theta_S^{PL})$$ //should be low

↓ is a function of (θ_T)

further,

$$\min_{\theta_T} \mathcal{L}_L\left(\theta_S^{PL}(\theta_T)\right)$$

where $\theta_S^{PL}(\theta_T) = \arg\min_{\theta_S} \mathcal{L}_u(\theta_T, \theta_S)$

Pseudo label adjustment is possible ∵

Now optimization problem until all everything !!

Practical Approximation:

$$\theta_S^{PT}(\theta_T) \approx \theta_S - \eta_S \nabla_{\theta_S} \mathcal{L}_u(\theta_T, \theta_S)$$

$$\min_{\theta_T} \mathcal{L}_l\left(\theta_S - \eta_S \nabla_{\theta_S} \mathcal{L}_u(\theta_T, \theta_S)\right)$$

SGD optimization Objective:

$$\theta_S' = \theta_S - \eta_S \nabla_{\theta_S} \mathcal{L}_u(\theta_T, \theta_S) \quad \text{// 1st}$$

$$\theta_T' = \theta_T - \eta_S \nabla_{\theta_T} \mathcal{L}_l\left(\theta_S - \nabla_{\theta_S} \mathcal{L}_u(\theta_T, \theta_S)\right) \quad \text{// 2nd}$$

labeled data

unlabeled

ⓟ PAWS

PAWS

unlabeled dataset $\mathcal{D} = (x_i)_{i \in [1, N]}$

Support set $\mathcal{S} = \{(x_i, y_i)_{i \in [1, M]}\}$ $M << N$

Leverage both $\mathcal{D}$ & $\mathcal{S}$ — Pretraining $\mathcal{D}$, $\mathcal{S}$ / fine tune with $\mathcal{S}$ — twice??

SwAV, BoYL → positive only

$x_i \in \mathcal{D}$ → Anchor $\hat{x}_i$, positive $\hat{x}_i^+$ } minimize cross entropy between them.

Detailed:

in hard

$x \in \mathbb{R}^{m \times (3 \times H \times W)}$ ← view Anchor

$x_d^+ \in \mathbb{R}^{n \times (3 \times H \times W)}$ ← positive

$x_s \in \mathbb{R}^{m \times (3 \times H \times W)}$

$y_s \in \mathbb{R}^{m \times K}$ } labeled [K total]

→ one hot

encoder: $\mathbb{R}^{3 \times H \times W} \xrightarrow{f_\theta} \mathbb{R}^d$

$z \in \mathbb{R}^{n \times d}$

$z^+ \in \mathbb{R}^{n \times d}$

$z_s \in \mathbb{R}^{S \times d}$ } label matrix. → j-th row

similarity Classifier $\pi_j(z_i, \underline{z_s}) = \sum_{(z_j, y_j) \in \underline{z_s}} \left(\frac{d(z_i, z_j)}{\sum_{z_{s_k} \in \underline{z_s}} d(z_i, z_{s_k})} \right) y_j$

similarity matrices: $d(a,b) = \exp\left(\frac{a^T b}{\tau \|a\| \|b\|}\right)$

$p_i : \pi(z_i; \underline{z}) = \sigma_T(z_i^T \underline{z}_s)\underline{y}_s$ // softmax prob.

sharpening function $[p(p_i)]_k := \frac{[p_i]_k^{1/T}}{\sum_{j=1}^{K}[p_i]_j^{1/T}}$ → temperature.

; $k = 1 \cdots K$

↳ weights sharpening $(p_i)_j$

where, $p_i \in [0,1]^K$

overall objective, for encoder. to minimize

$$\frac{1}{2n}\sum_{i=1}^{n}\left[H(p(p^+), p_i) + H(p(p_i), p^+)\right] - H(\bar{p})$$

cross entropy — tricheck? why? — Entropy

Theoretical Bound:

Assumption: [Balanced class] & [target sharpening & not uniform].

$d(z_i, z_i) = d(z_i, z_j)$. uniform

Prop: Non-collapsing Representation: if rep collapse $z_i = z$ #cies

then $\|\nabla_\theta H(p^+, p)\| > 0$; gradient is positive?).

Proof: if $d(z_i, z) = d(z_j, z)$

$\Rightarrow p_i : \pi(z, S) = \frac{1}{K}\sum_i y_i$ $y_i = \frac{1}{K}$ (yes) // uniform

p^+ not uniform then, $\|\nabla H(p^+, p)\| > 0$

not uniform ↓ ↓ (uniform)

Maximum Entropy IRL

① Maximum entropy IRL

Background: Agent behavior: ζ / trajectory

state: s_i → feature $f_{s_i} \in \mathbb{R}^k$

Action: a_i

Goal: optimizing some function f_{s_i} → Reward value

$f_\zeta = \sum_{s_i \in \zeta} f_{s_i}$ > reward for all the path.

$\text{Reward}(f_\zeta) = \theta^T f_\zeta = \sum_{s_i \in \zeta} \theta^T f_{s_i}$

feature expectation → $\sum_{\text{Path } \zeta_i} P(\zeta_i) f_{\zeta_i} = \bar{f}$ // Probabilistic problem.

Deterministic Path distribution: $P(\zeta_i|\theta) = \frac{1}{Z(\theta)} e^{\theta^T f_{\zeta_i}}$ → Partition function

Distribution

[Plans with higher reward is preferred

Nondeterministic path: $P(\zeta|\theta,T) = \sum_{\sigma \in \zeta} P_T(\sigma) \frac{e^{\theta^T f_\zeta}}{Z(\theta,\sigma)} \mathbb{I}_{\zeta \in \sigma}$ ← Identity

$\approx \frac{e^{\theta^T f_\zeta}}{Z(\theta,T)} \prod_{s_{t+1}, a_t, s_t \in \zeta} P_T(s_{t+1}|a_t, s_t)$ ← given??

Stochastic policies: $P(\text{action}|\theta,T) \propto \sum_{\zeta, a \in \zeta_{t=0}} P(\zeta|\theta,T)$

Learning from Demonstration:

$\theta^* = \arg\max_\theta L(\theta) = \arg\max_\theta \sum_{\text{examples}} \log P(\bar{\zeta}|\theta,T)$

$\nabla L(\theta) = \bar{f} - \sum_\zeta P(\zeta|\theta,T) f_\zeta = \bar{f} - \sum_{s_i} D_s f_{s_i}$ → state visitation freq. (Alg 1)

⑧ NNCLR — NNCLR

infoNCE: $$\ell_i^{\text{infoNCE}} = -\log \frac{\exp(z_i \cdot z_i^+/\tau)}{\exp(z_i \cdot z_i^+/\tau) + \sum_{z^- \in N_i} \exp(z_i, z^-/\tau)}$$

$$\ell_i^{\text{simCLR}} = -\log \frac{\exp(z_i \cdot z_i^+/\tau)}{\sum_{k=1}^{K} \exp(z_i, z_k^+/\tau)}$$

NNCLR: $$\ell_i^{\text{NNCLR}} = -\log \frac{\exp(NN(z_i;\theta) \cdot z_i^+/\tau)}{\sum_{k=1}^{K} \exp(NN(q, z_i) \cdot z_k^+/\tau)}$$

where, $NN(z_i;\theta) = \arg\min_{v \in Q} \|z_i - v\|_2$ key points.

⑧ Perceptual rewards

Perceptual Rewards

IRL

s_t → visual feature activation at time t ⇒ $[s_{1t}, s_{2t}, \cdots]$

τ → $\{s_1, \cdots s_T\}$ // sequence of trajectory.

$$P(\tau) = p(s_1, \cdots, s_T) = \frac{1}{Z} \exp\left(\sum_{t=1}^{T} R_t(s_t)\right)$$; maxEnt model

(R_t: unknown rewards (target))

Boltzmann distribution.

{Dynamic programming ⇐ challenge: How to compute ??}

Now, next state, $s_{t+1} = \begin{cases} f(a_t, s_t) & \text{Deterministic} \\ p(s_{t+1} \mid a_t, s_t) & \text{Probabilistic.} \end{cases}$

Simplifying Assumption,

$$p(\tau) = \prod_{t=1}^{T} \prod_{i=1}^{N} p(s_{it}) = \prod_{t=1}^{T} \prod_{i=1}^{N} \frac{1}{Z_{it}} \exp\left(R_i(s_{it})\right)$$

where ⇒ $R_t(s_t) = \sum_{i=1}^{N} R_i(s_{it})$

Intermediate stage Discovery:

$$p(\tau) = \prod_{t=1}^{T} \prod_{i=1}^{N} \frac{1}{Z_{it}} \exp\left(R_{i g_t}(s_{it})\right)$$

g_t ↓

[index of good step at time t]

Improving MB

(P) Improving MB

given images $x_1 \cdots x_n$ & their encoding $f(x_i) = \phi(x_i, \theta) \in \mathbb{R}^d$ (NN)

$$\mathcal{L}_{CE} = \log \prod_{i=1}^{n} P(i|x_i) = \sum_{i=1}^{n} \log \frac{e^{(f_i^T f_i/\tau)}}{\sum_{j=1}^{n} \exp(f_i^T f_j/\tau)} \quad (I)$$ Normal

mini large batch

K augmentation for each $x_m \rightarrow \{x_m^{(1)}, x_m^{(2)} \cdots x_m^{(k)}\}$

$$\text{modified } \mathcal{L}_{CE} = \sum_{i=1}^{|B|} \sum_{k=1}^{K} \log \frac{\exp(f_i^T f_i^{(k)}/\tau)}{\sum_{j=1}^{n} \exp(\tilde{f}_j f_i^{(k)}/\tau)} \quad (II)$$

(1) two sum; $\tilde{f}_j$: Near embed.

$$\tilde{f}_i = m \tilde{f}_i + (1-m) \sum_{k=1}^{K} \frac{1}{K} f_i^{(k)}$$ // Aggregation with some moment.

Instance consistency (2)

eq (I)

$$\text{Consistency loss, } \mathcal{L}_{cons} = \sum_{k=1}^{K} \sum_{j \neq k} KL\left(P(i|x_i^{(k)}) \,||\, P(i|x_j^{(j)})\right) \quad (III)$$

$x_i^{(k)}$ – $x_j^{(j)}$: diff; $x_i^{(k)}$ – $x_i^{(j)}$: same

the whole thing is contrastive

classified different [Diffent embedding looks different]

Meta learning for Semi-Supervised FSL

① Meta learning for semi-supervised FSL

k-shot N-way episodes. ← total k example from N classes.

i) support (training) set $S = \{(x_1, y_1)\ (x_2, y_2) \cdots (x_{N\times k}, y_{N\times k})\}$

ii) Query/test set $Q = \{(x_1, y_1), (x_2, y_2) \cdots\cdots (x_T, y_T)\}$

Prototype, P_c of class c

$$P_c = \frac{h(x_i)\, z_{ic}}{\sum z_{ic}} \quad ; \quad z_{ic} = \mathbb{1}[y_i = c]$$

$$P(c \mid x^*, \{P_c\}) = \frac{\exp\left(-\|h(x^*) - P_c\|_2^2\right)}{\sum_{c'} \exp\left(-\|h(x^*) - P_{c'}\|_2^2\right)}$$

Loss function: to minimize

$$-\frac{1}{T} \sum_i \log P\left(y_i^* \mid x_i^*, \{P_c\}\right)$$

$\arg\max_{y^*} c$

→ for test data (query)

Prototypical Net with Soft k-means: extra terms [softmax]

$$\hat{P}_c = \frac{\sum_i h(x_i)\, z_{i,c} + \sum_j h(\tilde{x}_j)\, \tilde{z}_{j,c}}{\sum_i z_{i,c} + \sum_j \tilde{z}_{j,c}} \quad ; \quad \tilde{z}_{j,c} = \frac{\exp\left(-\|h(\tilde{x}_j) - P_c\|^2\right)}{\sum_{c'} \exp\left(-\|h(\tilde{x}_j) - P_{c'}\|^2\right)}$$

labeled — unlabeled

Refinement Prototypes: $\tilde{P}_c$

Protop. Net with k-means [+A distractor class]

assumption: $p_c = \begin{cases} \frac{\sum_i h(x_i) z_{ic}}{\sum_i z_{ic}} ; & \text{for } c = 1 \cdots N \\ 0 ; & \text{for } c = N+1 \end{cases}$

$$\tilde{z}_{j,c} = \frac{\exp\left(-\frac{1}{\sigma_c^2}\|\tilde{x}_j - p_c\|_2^2 - A(\sigma_c)\right)}{\sum_{c'} \exp\left(-\frac{1}{\sigma_{c'}^2}\|\tilde{x}_j - p_{c'}\|_2^2 - A(\sigma_{c'})\right)}$$

($\tilde{x}_j = h(x_j)$; σ_c → length scale for distractor class.)

$$; A(x) = \frac{1}{2}\log(2\pi) + \log(\pi)$$

Here this paper, $\pi_{1,\cdots N} = 1$

learn $\pi_{N+1} = ??$

PN soft k-means & masking:

Normalized Distance, $\tilde{d}_{j,c} = \frac{d_{jc}}{\frac{1}{m}\sum_j d_{jc}}$ ($d_{jc} = \|h(x_j) - p_c\|_2^2$)

Soft threshold

$$[\beta_c, \gamma_c] = mLP\left(\left[\min_j(\tilde{d}_{jc}), \max_j(\tilde{d}_{jc}), \text{var}_j(\tilde{d}_{jc}), \text{skew}_j(\tilde{d}_{jc}), \text{kurt}_j(\tilde{d}_{jc})\right]\right)$$

slope

$$\tilde{p}_c = \frac{\sum_i h(x_i) z_{ic} + \sum_j h(\tilde{x}_j)\tilde{z}_{jc} m_{jc}}{\sum_i z_{ic} + \sum_j \tilde{z}_{jc} m_{jc}} ; \quad m_{jc} = \sigma\left(-\gamma_c(\tilde{d}_{jc} - \beta_c)\right)$$

(soft mask; sigmoid; $\sum \tilde{z}_{jc} m_{jc}$ → modified term)

modified cluster.

Ⓟ minimum Entropy regularization

original Data $\mathcal{L}_n = \{\underline{x}_i, y_i\}$ $\underline{x}_i \in X$ Data input

$y_i \in \{w_1 \cdots w_K\}$ → target output

Some of the label is missing !!

Criterion Derivation:

new learning set $\mathcal{L}_n = \{\underline{x}_i, \underline{z}_i\}$

- $z_{ik} = 1$ if $y_i = w_k$ } label is provided
- $z_{ik} = 0$ } $y_i \neq w_k$
- $z_{il} = 1$; $l = 1, \cdots K$
  - when label is not provided.
  - → means x_i is everybody !!
  - → randomly distributed

for an unlabeled case $P(z|x_i, w_k) = P(z|x_i, w_l)$

$\forall (w_k, w_l)$

$\therefore$ Now, $P(w_k|x, z) = \dfrac{z_k P(w_k|x)}{\sum_{l=1}^{K} z_l P(w_l|x)}$

conditional log likelihood

model for $P(w_k|\underline{x})$

↑ θ parameters (concave)

$$L(\theta, \mathcal{L}_n) = \sum_{i=1}^{n} \log \left(\sum_{k=1}^{K} z_{ik} f_k(\underline{x}_i; \theta) \right) + \underbrace{h(\underline{z}_i)}_{\text{independent of } \theta(x,y)}$$

when unlabeled data is informative:

Conditional entropy (unknown about $y \mid x, z$ is known)

$$H(y \mid x, z) = -E_{x,y,z}\left[\log\left[P(y \mid x, z)\right]\right]$$

maximum entropy in reg:

$$E_{\theta,\psi}\left[H(y \mid x, z)\right] = c$$

(θ, ψ: model params.; log triangular)

$$\Rightarrow P(\theta, \psi) \propto \exp\left(-\lambda H(y \mid x, z)\right) \text{ // prior on } \theta$$

$$H_{emp}(y \mid x, z; \mathcal{L}_n) = -\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K} P(\omega_k \mid x_i, z_i)\log P(\omega_k \mid x_i, z_i)$$

Entropy Regularization:

$$g_k(x, z; \theta) = \frac{z_k f_k(x; \theta)}{\sum_{q=1}^{K} z_q f_k(x; \theta)}$$

→ labeled case $g_k(x, z; \theta) = z_k$

→ unlabeled case $g_k(x, z; \theta) = f_k(x; \theta)$ (→ model for $P(\omega_k \mid x, z)$)

So the maximizer $\sim P(\theta, \psi)$ ~~log~~ $P(\omega_k \mid x, z)$ // conditional.

$$c(\theta, \lambda; \mathcal{L}_n) = L(\theta; \mathcal{L}_n) - \lambda H_{emp}(y \mid x, z; \mathcal{L}_n)$$

$$= \sum_{i=1}^{n}\log\left(\sum_{k=1}^{K} z_{ik} f_k(x_i)\right) + \lambda\sum_{i=1}^{n}\sum_{k=1}^{K} g_k(x_i, z_i)\log g_k(x_i, z_i)$$

(first term: labeled data; second term: unlabeled data)

⑧ Neural turing machine — Neural Turing Machine

Reading: memory $M_t \rightarrow (N) \times [M]$ matrix
- N: memory location
- M: vector size at each location
- t: time

Attention weights $0 \le w_t(i) \le 1$; $i \in \{1 \dots N\}$ with constraint $\sum_i w_t(i) = 1$

read memory r_t ($1 \times M$ size) $\leftarrow \sum_i w_t(i) M_t(i)$
- i: ith row
- convex combination, weighted sum of rows
- differentiable

Writing:

$$\tilde{M}_t(i) \leftarrow M_{t-1}(i) \left[1 - w_t(i) e_t \right]$$

- e_t: erase vector ($1 \times M$), $[0 \dots 1]$
- $M_{t-1}(i)$ and $w_t(i) e_t$: pointwise multiply
- both differentiable

After erase add:

$$M_t(i) \leftarrow \tilde{M}_t(i) + w_t(i) a_t$$

Constructing weight vector:

focusing by content

$$w_t^c(i) \leftarrow \frac{\exp\left(\beta_t K\left[k_t, M_t(i)\right]\right)}{\sum_j \exp\left(\beta_t K\left[k_t, M_t(j)\right]\right)}$$

- β_t: key strength
- K: cosine sim
- k_t: key vector ($1 \times M$)

focusing by location:

interpolation gate (0, 1)

$$\underline{w}_t^g \leftarrow g_t \underline{w}_t^c + (1 - g_t)\underline{w}_{t-1}$$

$$\tilde{w}_t(i) \leftarrow \sum_{j=0}^{N-1} w_t^g(j)\, s(i-j)$$

shift weight,

$$w_t(i) \leftarrow \frac{\tilde{w}_t(i)^{\gamma_t}}{\sum_j \tilde{w}_t(j)^{\gamma_t}}$$

→ sharpening & normalization.

ⓟ Prototypical Network ①

- Given N labeled examples $S_N = \{(x_1, y_1) \dots (x_n, y_n)\}$ → label $\in \{0, 1, \dots k\}$

$x_i \in \mathbb{R}^D$

$y_i \in \{1, \dots K\}$

S_k → examples only from k classes out of K classes. (small k, capital K)

Prototype $c_k = \frac{1}{|S_k|} \sum_{(x_i, y_i) \in S_k} f_\phi(x_i)$ // mean vectors.

(uses later)

f_ϕ → Network parameters.

Now, distance function.

$$p_\phi(y = k \mid x) = \frac{\exp\left(-d\left(f_\phi(x), c_k\right)\right)}{\sum_{k'} \exp\left(-d\left(f_\phi(x), c_{k'}\right)\right)}$$ // probabilistic approach

Target to minimize: $J = -\log p_\phi(y = k \mid x_n)$

Prototype as mixture density estimation:

Bregman Divergence: $d_f(z, z') = f(z) - \{f(z') + \langle \nabla f(z'), z - z' \rangle\}$ (Definition)

$\nabla f(z')$ → (gradient) Derivative; $\langle \cdot, \cdot \rangle$ → dot products

f → convex function, $\mathbb{R}^n \to \mathbb{R}$

Exponential family of Distributions

$p_\psi(z \mid \theta) = \exp\left(z^T \theta - \psi(\theta) - g_\psi(z)\right) = \exp\left(-d_\phi(z, \mu(\theta)) - g_\phi(z)\right)$

z → output; θ → Dist params; $g_\psi(z)$ → extra term; $\psi(\theta)$ → cumulant function; $\mu(\theta)$ → Avg !!

d_ϕ → Bregman Divergence

$f_\phi(x) = z$ // network embedding.

Exponential family with mixture model param: $\Gamma = \{\theta_k, \pi_k\}_{k=1}^{K}$

$$p(z|\Gamma) = \sum_{k=1}^{K} \pi_k p_\psi(z|\theta_k)$$

$$= \sum_{k=1}^{K} \pi_k \exp\left(-d_\phi(z, \underbrace{\mu(\theta_k)}_{c_k}) - \underbrace{g_\phi(z)}_{\text{constant term gets canceled in prob equation.}}\right)$$

c_k: prototype (low inductive bias) (just Avg)

$$\Rightarrow p(y=k|z) = \frac{\pi_k \exp\left(-d_f(z, \mu(\theta_k))\right)}{\sum_{k'} \pi_{k'} \exp\left(-d_f(z, \mu(\theta_{k'}))\right)}$$

Connection linear model: for $d_f(z, z') = \|z - z'\|^2$ // Euclidean

$$-\|f_\phi(x) - c_k\|^2 = -f_\phi(x)^T f_\phi(x) + \underbrace{2c_k^T f_\phi(x)}_{w_k^T f_\phi(x)} - \underbrace{c_k^T c_k}_{+ b_k}$$

$\Downarrow$

linear model / term.

①Ⓕ Lifted structure.

Lifted Structure

Loss function:

$$J = \frac{1}{2|\hat{P}|} \sum_{(i,j)\in\hat{P}} \max(0, J_{i,j})^2$$

↳ positive pairs

$$J_{i,j} = \max\left(\max_{(i,k)\in\hat{N}} \alpha - D_{i,k},\ \max_{(j,l)\in\hat{N}} \alpha - D_{j,l} \right) + D_{i,j} + \text{ve}$$

Distance between

↳ Negative pairs.

maximize for what neg pairs.

Embedded feature vector $X \in \mathbb{R}^{m \times c}$ → class no

∴ embedding dimension

squared Norem, $\tilde{x} = \left[\|f(x_1)\|_2^2, \|f(x_2)\|_2^2, \ldots, \|f(x_m)\|_2^2 \right]$

Pairwise density matrix $D^2 = \tilde{x}\mathbf{1}^T + \mathbf{1}\tilde{x}^T - 2XX^T$

/// which leads (Efficient computation)

where $D_{i,j} = \|f(x_i) - f(x_j)\|_2^2$

simple compute.

using upper bound, the loss function.

$$\tilde{J}_{i,j} = \log\left(\sum_{(i,k)\in N} \exp\{\alpha - D_{i,k}\} + \sum_{(j,l)\in N} \exp(\alpha - D_{j,l}) \right) + D_{i,j}$$

should be as close as possible to 0

$$J = \frac{1}{2|P|} \sum_{(i,j)\in P} \max(0, \tilde{J}_{i,j})^2$$

Relation network

Problem definition: Episodic learning:

sample set: $S = \{(x_i, y_i)\}_{i=1}^{m}$ ($m = k \times c$) — k: examples, c: no. of class.

query set: $Q = \{(x_i, y_i)\}_{i=1}^{n}$

one shot learning: Relation Network source.

$k \rightarrow 1$

$$r_{i,j} = g_\phi\left(C\left(f_\varphi(x_i), f_\varphi(x_j)\right)\right); \quad i = 1, 2 \cdots c$$

for each k in k-shot learning.

g_ϕ: Relation Networks. f_φ: Embedding Network

Objective function:

query — support

$$f_\varphi, g_\phi \leftarrow \underset{\varphi, \phi}{\arg\min} \sum_{j=1}^{n} \sum_{i=1}^{m} \left\{ r_{i,j} - \mathbb{1}(y_i == y_j) \right\}^2$$

N-pair loss objective

⑧ N-pair loss objective

Incorporate multiple Negatives $\{x, x^+, \underbrace{x_1, \cdots x_{N-1}}_{\text{negs}}\}$ (x^+: pos, x: query)

$$L(\{x, x^+, \{x_i\}_{i=1}^{N-1}\}; f) = \log\left(1 + \sum_{i=1}^{N-1} \exp\left(f^T f_i - f^T f^+\right)\right)$$

($f^T f_i$: neg, $f^T f^+$: pos)

↑ as neg as possible ! ↑ minimize it

$$= -\log\left[\frac{\exp(f^T f^+)}{\exp(f^T f^+) + \sum_{i=1}^{N-1} \exp(f^T f_i)}\right]$$

!! multi-class logistic loss !!

N-pair loss efficient deep metric learning:

$$L_{\text{N-pair-mc}}\left(\{(x_i, x_i^+)\}_{i=1}^{N}; f\right) = \frac{1}{N} \sum_{i=1}^{N} \log\left(1 + \exp\left(f_i^+ f_j^+ - f_i^T f_i^+\right)\right)$$

⇓ maximizing what we have.

// new Hicks

$$L_{\text{N-pair-ovo}}\left(\{x_i, x_i^+\}_{i=1}^{N}; f\right) = \frac{1}{N} \sum_{j=1}^{N} \sum_{j \neq i} \log\left(1 + \exp\left(f_i^T f_j^+ - f_i^T f_i^+\right)\right)$$

// one-vs-one

Few shot meta-learning

Few shot meta learning?

simple view: Optimal model params

$$\theta^* = \arg\min_{\theta} \underbrace{E_{D \sim P(D)}}_{\text{Dataset}} \left[L_\theta(D) \right]$$

one dataset as one data sample??

$D = \langle S, B \rangle$ → Prediction

[K shot - N-class classification]

↓ learning

Training in the same way as Testing:

$$D = \{(x_i, y_i)\} \in \mathcal{L}^{\text{Label}}$$

classifier f_θ

output probabilities for $y|x \rightarrow P_\theta(y|x)$

Optimal Parameters

$$\theta^* = \arg\max_{\theta} \underbrace{E_{(x,y) \sim D}}_{\text{Expectation}} \left[P_\theta(y|x) \right]$$

// Expectation distribution changes.

$$\theta^* = \arg\max_{\theta} E_{B \subset D} \left[\sum_{x,y \in B} P_\theta(y|x) \right]$$

Few Dataset → small support set → fake → fast learning.

steps

(i) subset of labels $L \subset \mathcal{L}^{label}$. taking few labels. [may be 2 of 6 others]

(ii) sample support $S^L \subset D$, to any batch $B^L \subset D$

$$y \in L, \forall (x, y) \in S^L, B^L$$

(iii) support set → part of model input.

(iv) optimization uses mini batch B_L

(S^L, B^L) → one data point??

Model trained to generalized to other dataset.

$$\theta = \arg\max_{\theta} E_{L \subset \mathcal{L}} \left[E_{S^L \subset D, B^L \subset D} \left[\sum_{(x,y) \in B^L} P_\theta (x, y, S^L) \right] \right]$$

[Meta learning Parts.]

[Optimization: to be Good at many]

Learner Vs meta-learner:

Two stage update:

(i) f_θ → learners model.

(ii) update via support set S; $\theta' = g_\phi(\theta, S)$

$$E_{L \subset \mathcal{L}} \left[E_{S^L \subset D, B^L \subset D} \left[\sum_{(x,y) \in B^L} P_{g_\phi(\theta, S^L)} (y \mid x) \right] \right]$$

Ⓟ Evolving loss ①

Evolving Loss

method : multimodal learning.

$$\mathcal{L} = \sum_m \sum_t \lambda_{m,t} \mathcal{L}_{m,t} + \sum_d \lambda_d \mathcal{L}_d$$

- $\lambda_{m,t}$: weighted $[0,1]$
- $\mathcal{L}_{m,t}$: task → modality
- λ_d: weighted $[0,1]$ constraint
- $\mathcal{L}_d$: distillation

Distillation: $\mathcal{L}_{dl}(L_i, m_i) = \| L_i - m_i \|_2$

- L_i: layer in main network
- m_i: layer in another network.

Evolving an unsupervised loss function?

① constraint

Zipf's distribution matching.

feature $x_{RGB} = E_{rgb}(I) \in \mathbb{R}^D$ → cluster into k.

$$P(x|c_i) = \frac{1}{\sqrt{2\sigma^2\pi}} \exp\left(-\frac{(x-c_i)^2}{2\sigma^2}\right) \quad // \text{centroid } c_i \in \mathbb{R}^D \; \{c_1 \cdots c_k\}$$

$\sqrt{2\sigma^2\pi} \Rightarrow 1$

$$P(c_i|x) = \frac{P(c_i)\,P(x|c_i)}{\sum_j P(x|c_j)\,P(c_j)} = \frac{\exp(-|x-c_i|)^2}{\sum_{j=1}^{k} \exp(-|x-c_j|)^2}$$

prior of $q(c_i)$ = $\frac{1/i^s \rightarrow \text{real constant}}{H_{k,s}}$!! ?

↳ kth harmonic number !! ?

Law of total prob.

$$KL(P\|q) = \sum_i P(c_i) \log \frac{P(c_i)}{q(c_i)} \quad // P(c_i) = \frac{1}{N} \sum_{x \in V} P(c_i|x)$$

What should be contrastive

view Generations:

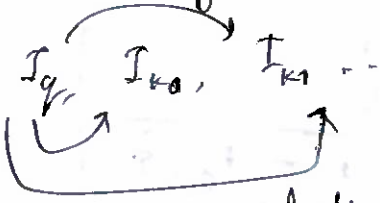
n atomic augmentation.

query view I_q

first key view $I_{k_0} := \tilde{I}_{\{q,k_0\}} = \mathcal{T}\{x_1^{\{q,k_0\}}, x_2^{\{q,k_0\}}, \dots, x_n^{\{q,k_0\}}\}$

n views from reference images I_{k_i}, $\forall i \in \{1, \dots, n\}$

Provides with I_q, I_{k_0}, I_{k_1} ...



Contrastive Embedding Space:

$f(): \mathcal{X} \to v \in \mathbb{P}^d$ $\begin{matrix}1\\2\\\vdots\\n\end{matrix}$ n heads each $\mathbb{P}^d$

NTL setup:

$v^q, v^{k_0} \dots v^{k_n} \to \mathbb{P}^d$

projected into $(n+1)$ normalized embedding.

$z_0, z_1 \dots z_n \in \mathbb{P}^{d'}$ by head $h: v \to z$

$x \to q$, k_0, k_1 : k_n

z_0 ↓ invariant to all (q)

$z_1 \dots z_n$ ↘ dependent on 1st transformation ... so on so forth

(P) what should be — same image (N)

↑ ↗ Different instance

$k_i^{+/-}$ → Augmentation.

embedding → z_i → ~~instance~~ head

which instance

same (+)

diff (-)

image → k_i → z_i → Aug

k_i → Augmentation

three things to keep track

with ~~Augmentation~~ head

$$E_{ij}^{\{+,-\}} = \exp\left(\frac{z_i^q \cdot z_j^{k_j^{\{+,-\}}}}{\tau}\right)$$

overall loss

$$\mathcal{L}_q = -\frac{1}{n+1}\left[\log \frac{E_{0,0}^{+}}{E_{0,0}^{+} + \sum_{k^-} E_{0,0}^{-}}\right]$$

$z_i^{k_j^{+/-}}$

$$-\frac{1}{n+1}\left[\log \frac{E_{i,i}^{+}}{\sum_{j=0}^{n} E_{i,j}^{+} + \sum_{k^-} E_{i,i}^{-}}\right]$$

ⓟ intriguing Properties of CL Loss

Intriguing properties of CL loss

Generalized loss function.

$$\mathcal{L}_{gen\ CL} = \mathcal{L}_{align} + \lambda \mathcal{L}_{distrib}$$

$$\tau \mathcal{L}^{NT\ xent} = -\frac{1}{N} \sum_{i,j} sim(z_i, z_j) + \frac{\tau}{n} \sum_{i} \log \sum_{k=1}^{2n} \mathbb{1}_{[k \neq i]} \exp(sim(z_i, z_k))$$

$$\mathcal{L}^{NT\ xent} = \frac{-1}{n} \sum_{i,j \in [n]} \log \frac{\exp[sim(z_i, z_j)/\tau]}{\sum_{k=1}^{2n} \mathbb{1}_{[k \neq i]} \exp[sim(z_i, z_k)/\tau]}$$

By some calculation, log breaking ↗ log sum exp ↗ normalized.

$$\textcircled{2}\mathcal{L}^{NT\ xent} = -\frac{1}{n} \sum_{i,j} sim[\cancel{\exp}(z_i, z_j)\cancel{/\tau}] + \textcircled{\tau} \frac{1}{n} \sum_{i} \log \sum_{k=1} \mathbb{1}_{k \neq j} \exp\left(sim(z_i, z_k)\cancel{/\tau}\right)$$

remove it

Alignment

Distribution.

[match hidden dist' uniform in hypersphere]

- [Pairwise potential of gaussian kernel.
- [minimized by perfect uniform under]

(R) Intriguing CL Loss. (u)

Sliced Wasserstein Distance (SWD) loss

Activation vectors. $H \in \mathbb{R}^{b \times d}$, a prior distribution S

Draw prior vector $P \in \mathbb{R}^{b \times d}$ using S

generate random orthogonal vector matrix $W \in \mathbb{R}^{d \times d'}$ $d > d'$??

Projection $H^t = HW$; $P^t = PW$ // rotation.

initialize $L = 0$ // SWD

for $j \in \{1, 2, \ldots d'\}$ do → projection for all.

$$L = L + \| \text{sort}(H^t_{:,j}) - \text{sort}(P^t_{:,j}) \|^2$$

↓ column

end for

return $L/(bd')$

---

where it fits ??

Measuring invariance in DL

ⓕ Measuring Invariance in DL

- Measuring Invariance :

firing neuron, $s_i h_i(x) > t_i$

$s_i \in \{-1, 1\}$ // chose s_i to maximize

firing, $f_i(x) = 1_{\{s_i h_i(x) > t_i\}}$

Transformation function: $\tau(x, \gamma)$

Local trajectory $T(x)$ → semantically similar stimuli

$$T(x) = \{ \tau(x, \gamma) \mid \gamma \in \Gamma \}$$

global stimuli $G(i) = \mathbb{E}[f_i(x)]$ // over all possible input.

local firing rate $L(i) = \frac{1}{|Z|} \sum_{z \in Z} \frac{1}{T(z)} \sum_{x \in T(z)} f_i(x)$ // only over semantically similar input

invariance score, $s(i) = \frac{L(i)}{G(i)}$