

Algorithm: Spatial-channel collaborative multi-scale graph interaction deep transfer learning (SCMGITL)

Input:

- Source domains: $S = \{S_1, S_2, \dots, S_M\}$
- Target domain: T
- Batch size: N_m
- Number of categories: K
- Number of scales: H
- Spatial-channel refinement parameters

Output:

- Classification predictions for target domain samples

```

1: // ===== 3.1: Spatial-Channel Collaborative Prototype Extraction =====
2: for each batch ( $X_{src}$ ,  $y_{src}$ ) from  $S$  and ( $X_{tgt}$ ,  $y_{tgt}$ ) from  $T$  do
3:   // Spatial refinement
4:    $X_{out} = \beta + \frac{\psi(X - \mu)}{\sqrt{\sigma^2 + \varepsilon}}$  // Eq. (1)
5:    $\gamma$  = learnable_parameter in  $GN$ 
6:    $W_\gamma = \{w_i\} = \gamma_i / \left( \sum_{j=1}^D \gamma_j \right)$ ,  $i, j = 1, 2, \dots, D$  // Eq. (2)
7:    $W = \text{Gate}(\text{Sigmoid}(W_\gamma))$ 
8:    $W_1, W_2 = \text{threshold\_gate}(W)$ 
10:   $X^{wl} = X_{11}^w \oplus X_{22}^w$  // Eq. (3)
11:   $X^{w2} = X_{21}^w \oplus X_{12}^w$  // Eq. (4)
12:   $X^w = \text{Concat}(X^{wl}, X^{w2})$  // Refined spatial feature
13:  // Channel refinement
14:  ( $X_{up}, X_{low}$ ) =  $\text{Split}(X^w, \text{ratio}=\alpha)$  //  $\alpha$  is splitting ratio
15:   $Y_1 = X_{up} \left( \lambda^{GWC} + \lambda^{PWC_1} \right)$  // Eq. (5)
16:   $Y_2 = \lambda^{PWC_1} X_{low} \cup X_{low}$  // Eq. (6)
17:   $O_1, O_2 = \text{GlobalAveragePool}(Y_1), \text{GlobalAveragePool}(Y_2)$ 
18:   $I_1 = \exp(O_1) / (\exp(O_1) + \exp(O_2))$  // Eq. (7)
19:   $I_2 = \exp(O_2) / (\exp(O_1) + \exp(O_2))$ 
20:   $Y = I_1 * Y_1 + I_2 * Y_2$  // Eq. (8) - channel refined feature
21: // Extract prototypes for source domains
22: for each source domain  $m = 1$  to  $M$  do

```

```

23:   for each category  $k = 1$  to  $K$  do
24:      $\hat{c}_k^{S_m} = \text{mean}(f_S(x_i))$  for all  $x_i$  in category  $k$  // Eq. (9)
25:   end for
26: end for

27: // ===== Temporarily skip the target domain prototype update and proceed with graph construction and propagation =====
28: // Use the current target domain prototype (obtained from the previous batch update or initial value)
29: // If it is the first iteration, initialize to the source domain mean
30: if first_batch then
31:   for  $k = 1$  to  $K$  do
32:      $\hat{c}_k^T = \text{mean}(\hat{c}_k^{S_m} \text{ for } m=1,2,\dots,M)$ 
33:   end for
34: end if

49: // ===== 3.2: Multi-Scale Graph Interaction Transfer Network =====
29: // Step 1: Build multi-domain knowledge graph
30:  $\mathbf{F} = \begin{bmatrix} \underbrace{c_1^{S_1} c_2^{S_1} \cdots c_K^{S_1}}_{\text{prototypes of } S_1} \cdots \underbrace{c_1^{S_M} c_2^{S_M} \cdots c_K^{S_M}}_{\text{prototypes of } S_M} & \underbrace{c_1^T c_2^T \cdots c_K^T}_{\text{prototypes of } T} \end{bmatrix}^T$ 
31: for  $i,j$  in 1 to  $(M+1)*K$  do
32:    $\mathbf{A}_{i,j} = G(\mathbf{F}_i^T, \mathbf{F}_j^T)$  // Eq. (10)
33: end for

34: // Step 2: Build extended graph with query samples
35: for each query sample  $q$  in batch  $N_m$  do
36:    $\bar{\mathbf{F}} = [\mathbf{F}^T f(q_1) f(q_2) \cdots f(q_{|N_m|})]^T$  // Eq. (11)
37:    $\mathbf{S}_{i,j} = G(\mathbf{F}_i^T, f(q_j))$  // Eq. (12)
38:    $\bar{\mathbf{A}} = \begin{bmatrix} \mathbf{A} & \mathbf{S} \\ \mathbf{S}^T & \mathbf{I} \end{bmatrix}$  // Eq. (13)
39: end for

40: // Step 3: Multi-scale graph interaction
41:  $\mathbf{X\_list} = [\mathbf{X}^{(0)}, \mathbf{X}^{(1)}, \mathbf{X}^{(h)}, \dots, \mathbf{X}^{(H)}]$ 
42: for  $h = 1$  to  $H$  do
43:    $\mathbf{AX}^{(h-1)} \rightarrow \mathbf{X}^{(h)}$ ,  $\forall h=1, 2, \dots, H$  // Eq. (14)
44:    $\mathbf{X\_list.append}(\mathbf{X}^{(h)})$ 

```

```

45: end for

46: // Dynamic interaction flow construction
47: for each node  $i = 1$  to  $|\bar{V}|$  do
48:    $\mathbf{Z}_i = \xi(\mathbf{X}_i^{(1)} \| \mathbf{X}_i^{(2)} \| \dots \| \mathbf{X}_i^{(H)})$            // Eq. (15)
49:   for each propagation step  $l$  do
50:      $\mathbf{X}_i^{(l)} = \mathbf{X}_i^{(l)} \| \mathbf{Z}_i$                                 // Eq. (16)
51:      $\tilde{w}_i(l) = \text{Sig}(\mathbf{X}_i^{(l)} \cdot s)$                       // Eq. (17)
52:   end for
53:   // Normalize over scales
54:   for  $l = 1$  to  $H$  do
55:      $w_i(l) = \frac{e^{\tilde{w}_i(l)}}{\sum_{h=0}^H e^{\tilde{w}_i(h)}}$           // Eq. (18)
56:   end for
57: end for

58: // Build weight matrix and fuse features
59: for  $h = 1$  to  $H$  do
60:    $\eta_h[i] = w_i(h) \quad 1 \leq i \leq n \quad \text{for all nodes } i$            // Eq. (19)
61:    $\mathbf{W}_h = \text{Diag}(\eta_h)$                                          // Eq. (20)
62: end for
63:  $\mathbf{U}_x = \sum_{h=0}^H \mathbf{W}_h \mathbf{X}^{(h)}$                            // Eq. (21)

64: // Step 4: Cross-domain diagnosis
65:  $\mathbf{P} = f_H(\bar{\mathbf{F}}, \bar{\mathbf{A}})$                                      // Eq. (22) - classification probabilities

100: // Extract prototypes for target domain using confidence-thresholded pseudo-labeling
101: Now proceed with updating the target domain prototype (based on graph network output)
102: Extracting the classification probability of target samples from  $\mathbf{P}$ 
104: HC = empty_set // High confidence sample set
106: for  $j = 1$  to N_target_samples do
107:   // Node index of the target sample in the extended graph
108:   node_idx =  $|F| + j$ 
109:    $p_t = \mathbf{P}[\text{node\_idx}]$  // The classification probability of the target sample
110:
111: // Calculate entropy value
112: entropy_t =  $-\sum_{k=1}^K p_t[k] * \log(p_t[k])$ 

```

```

113:
114: if entropy_t < threshold_entropy then
115:   pseudo_label = argmax( $p_i$ )
116:   // Store sample features and pseudo labels
117:   HC.add
118: end if
119: end for

120: // Update target domain prototype
121: for each category  $k = 1$  to  $K$  do
122:   // Collect all high confidence sample features for this category
123:   features_k = { $z \mid (z, \text{label})$  in HC and label =  $k$ }
124:
125: if features_k not empty then
126:   // Update the target domain prototype using high confidence samples
127:    $\hat{c}_k^T = \text{mean}(\text{features}_k)$ 
128: else
129:   // If there are no high confidence samples, maintain the original value or use the source domain mean
130:   //  $\hat{c}_k^T$  remain unchanged
131: end if
132: end for

```

66: // ===== 3.3: Category Constraint Loss =====

67: // Global constraint loss

$$68: L_R^g = \frac{1}{(M+1)^4} \sum_{m,n,i,j=1}^{M+1} \| \mathbf{A}_{i,j} - \mathbf{A}_{m,n} \|_F \quad // \text{Eq. (23)}$$

69: // Local constraint loss

$$70: L_R^l = \frac{1}{|N_m|} \sum_{k=1}^K \left(\sum_{(x_i^T, \hat{y}_i^T) \in T_k} \| f(x_i^T) - c_k^T \|_2^2 + \sum_{m=1}^M \sum_{(x_i^{S_m}, y_i^{S_m}) \in \mathcal{S}_m^k} \| f(x_i^{S_m}) - c_k^{S_m} \|_2^2 \right) \quad // \text{Eq. (24)}$$

71: // Classification losses

$$72: L_C^s = \frac{\sum_{m=1}^M (\mathbb{E}_{(x_i^{S_m}, y_i^{S_m}) \in \mathcal{S}_m} L_{ce}(p(x_i^{S_m}), y_i^{S_m}))}{M} \quad // \text{Eq. (25)}$$

$$73: L_C^p = \frac{\sum_{k=1}^K L_{ce}(p(c_k^T), k) + \sum_{m=1}^M \sum_{k=1}^K L_{ce}(p(c_k^{S_m}), k)}{K(M+1)} \quad // \text{Eq. (26)}$$

$$74: L_C^t = -\mathbb{E}_{(x_i^T, \hat{y}_i^T) \in T} \sum_{k=1}^K p(\hat{y}_i^T = k | x_i^T) \log p(\hat{y}_i^T = k | x_i^T) \quad // \text{Eq. (27)}$$

75: // Total loss

$$76: L_{CCL} = L_C + L_R, \quad // \text{Eq. (28)}$$

77: // Optimization

78: Update $f_S, f_H, \lambda, s, \gamma$, etc. via backpropagation

79: // Inference on target domain

80: for each target sample x_t in T do

81: Compute extended graph with x_t

$$82: p_t = f_H(\bar{\mathbf{F}}, \bar{\mathbf{A}})$$

$$83: y_{\text{pred}} = \text{argmax}(p_t)$$

84: end for

85: return y_{pred} for all target samples