Multi-Surrogate-Teacher Assistance for Representation Alignment in Fingerprint-based Indoor Localization
Supplementary Material

Son Minh Nguyen[1], Linh Duy Tran[2], Duc Viet Le[1], Paul J.M Havinga[1]
[1]Department of Computer Science, University of Twente
[2]Viettel AI, Viettel Group
{m.s.nguyen, v.d.le, p.j.m.havinga}@utwente.nl
linhtd15@viettel.com.vn

A. Hyperparemeter Selection

For the angular margin α, we searched within a range from 0 degrees (0 radians) to 30 degrees (0.52 radians) on UIJIndoorLoc, with a step size of 0.1 radians, to determine the optimal angular margin. Our experimentation revealed that an angular margin of 0.2 radians consistently yielded stable results. Higher values of the angular margin α were not recommended, as they led to significant increases in model loss during training, making convergence extremely difficult.

The weighting factors mentioned are also determined through a grid search on UIJIndoorLoc and subsequently applied to other datasets. This process is expedited by leveraging prior knowledge that the primary task of indoor localization (denoted as J_{MAE}) should be accorded greater emphasis, necessitating a larger range and higher amplitude for the weighting factor λ_1. Conversely, the high sensitivity of J_{FI} to the learning ability necessitates a much narrower range for the weighting factor λ_4. To ensure balanced impacts, these weighting factors are normalized together.

The rationale behind conducting grid-search hyperparameter selection exclusively on the UJIIndoorLoc dataset before applying the chosen hyperparameters to other datasets lies in UJIIndoorLoc's notable generalizability. This dataset has a collection period spanning months and encompasses expansive campus coverage across three buildings, totaling 110,000 square meters, effectively capturing the dynamic nature of real-world environments. During the grid search process, we set specific search ranges for each hyperparameter, such as [1-5] for λ_1 with a step size of 0.2, and [0.1-1] for $\lambda_{2,3,4}$ with a step size of 0.1. Through this rigorous exploration of over 7 days, we identified that the set of values [3,0.5,0.5,0.5] yielded the best performance during testing on the UJIIndoorLoc dataset.

B. Incompatibly with other knowledge-transfer in RSS-Fingerprint-based Indoor Localization

Compounded by the nature of radio propagation and multipath effects, the distinctive characteristics of RSS datasets, including variabilities in building structure, occupancy levels, and the arrangement and number of input WiFi anchors, give rise to uncompromising discrepancies in both appearance (input size) and content (locations). Unlike other data types such as images or text that easily achieve a common input size with minimal content alteration using standard interpolation techniques, RSS fingerprints cannot be resized as their arrangements of the disparate number of anchors are just unknown. Even if the input sizes were reluctantly synchronized, the content representing specific locations would undergo significant alterations, leading to deviations from the true data distribution. Consequently, traditional domain adaptation approaches [1, 6], such as meta-learning and adversarial learning, face limitations in their applicability to such datasets.

Meta-learning approaches optimize a common meta-learner for the target task through the learning abilities of its versions trained on sub-tasks. However, implementing this approach to achieve a unified meta-learner for different RSS fingerprint datasets presents challenges. These datasets often vary significantly in the number of anchors, with differences of hundreds observed between datasets. Additionally, standard resizing techniques cannot be applied due to the unique characteristics of RSS fingerprints.

Adversarial domain adaptation offers a potential solution to address differences in input size by employing separate feature generators for different datasets. However, this approach requires substantial modifications to existing architectures to ensure the delivery of homogeneous-sized features to domain discriminators for evaluation. Moreover, blindly learning domain-invariant features solely through artificial coarse-grained domain labels, especially in an adversarial learning framework, is inadequate for capturing fine-grained information particularly essential for precise localization. Additionally, this method is susceptible to instability, including notorious issues of model collapse, where discriminators fail to keep track of distribution changes in generated data.

C. Impacts of Target Relevance

The robustness of the framework is additionally evaluated on the target side where the target distribution is changed with the proportion of training data. Specifically, experiments are first carried out using only 1% and 10% of the training data for UJIIndoorLoc, and then expanded to 10% for the other datasets for general examination. This random partitioning is designed to simulate real-world scenarios for which all the models are subjected to the same conditions, and repeated for ten rounds to achieve statistical results. As demonstrated in Table 1, the framework exhibits its tolerance to target constrictions and consistently empowers state-of-the-art models to achieve strong performance.

D. Specific steps to the final J_{MI} in Eq.4

Cross-Mutual Information Maximization Constraint J_{MI} in Eq.4 can be represented by JS Divergence D_{JS} in retrospect as follows.

$$
\begin{aligned}
&D_{JS} = \mathbb{E}_{Z^E_{G_i}, Z^E_S \sim p\left(Z^E_{G_i}, Z^E_S\right)} \left(\log \frac{p_{Z^E_{G_i} Z^E_S}}{m\left(Z^E_{G_i}, Z^E_S\right)}\right) \\
&+\mathbb{E}_{Z^E_{G_i} \sim p\left(Z^E_{G_i}\right), Z^E_S \sim p\left(Z^E_S\right)} \left(\log \frac{p_{Z^E_{G_i}} p_{Z^E_S}}{m\left(Z^E_{G_i}, Z^E_S\right)}\right) \\
&\text{where } m\left(Z^E_{G_i}, Z^E_S\right) = \tfrac{1}{2} p\left(Z^E_{G_i}\right) p\left(Z^E_S\right) + \tfrac{1}{2} p\left(Z^E_{G_i}, Z^E_S\right) \\
&\Rightarrow 2D_{JS} = \\
&\mathbb{E}_{Z^E_{G_i} \sim p\left(Z^E_S\right)} \left[\begin{array}{l} \mathbb{E}_{Z^E_{G_i} \sim p\left(Z^E_{G_i} \middle| Z^E_S\right)} \left[\begin{array}{l} \log \frac{p\left(Z^E_{G_i} \middle| Z^E_S\right)}{p\left(Z^E_{G_i}\right)} \\ -\log\left(1 + \frac{p\left(Z^E_{G_i} \middle| Z^E_S\right)}{p\left(Z^E_{G_i}\right)}\right) \end{array} \right] \\ +\mathbb{E}_{Z^E_{G_i} \sim p\left(Z^E_{G_i}\right)} \left[-\log\left(1 + \frac{p\left(Z^E_{G_i} \middle| Z^E_S\right)}{p\left(Z^E_{G_i}\right)}\right)\right] \end{array} \right]
\end{aligned}
\tag{1}
$$

In addition, we make use of the mutual information estimator Ψ_θ [3] to estimate the logarithm ratio between $p\left(Z^E_{G_i} \middle| Z^E_S\right)$ and $p\left(Z^E_{G_i}\right)$, represented by $\Psi_\theta\left(Z^E_{G_i}, Z^E_S\right) = \log \frac{p\left(Z^E_{G_i} \middle| Z^E_S\right)}{p\left(Z^E_{G_i}\right)}$. In the reverse direction, it can be interpreted as $\frac{p\left(Z^E_{G_i} \middle| Z^E_S\right)}{p\left(Z^E_{G_i}\right)} = e^{\Psi_\theta\left(Z^E_{G_i}, Z^E_S\right)}$. Put it all together in Eq.1, the JS Divergence is elaborated further, which is exactly the Mutual Information constraint J_{MI} presented in this work:

$$
\begin{aligned}
&D_{JS} = \mathbb{E}_{p\left(Z^E_{G_i}, Z^E_S\right)} \left[\log \frac{e^{\Psi_\theta\left(Z^E_{G_i}, Z^E_S\right)}}{1+e^{\Psi_\theta\left(Z^E_{G_i}, Z^E_S\right)}}\right] \\
&-\mathbb{E}_{p\left(Z^E_{G_i}\right) p\left(Z^E_S\right)} \left[\log\left(1 + e^{\Psi_\theta\left(Z^E_{G_i}, Z^E_S\right)}\right)\right] \\
&= \mathbb{E}_{p\left(Z^E_{G_i}, Z^E_S\right)} \left[\log \frac{1}{e^{-\Psi_\theta\left(Z^E_{G_i}, Z^E_S\right)}+1}\right] \\
&-\mathbb{E}_{p\left(Z^E_{G_i}\right) p\left(Z^E_S\right)} \left[\log\left(1 + e^{\Psi_\theta\left(Z^E_{G_i}, Z^E_S\right)}\right)\right] \\
&= \mathbb{E}_{p\left(Z^E_{G_i}, Z^E_S\right)} \left[-\log\left(1 + e^{-\Psi_\theta\left(Z^E_{G_i}, Z^E_S\right)}\right)\right] \\
&-\mathbb{E}_{p\left(Z^E_{G_i}\right) p\left(Z^E_S\right)} \left[\log\left(1 + e^{\Psi_\theta\left(Z^E_{G_i}, Z^E_S\right)}\right)\right]
\end{aligned}
\tag{2}
$$

E. Specific steps expanded in Eq.9

We elaborate on the expression of the transmitting matrix T_j, which is mentioned in Eq.8 and is reduced to a simple form in Eq.9 as follows:

$$
\begin{aligned}
&T_i \triangleq \left[\left(\widehat{Z}_{j-1}\right)^{\Gamma} \left(\widehat{Z}_j\right)\right]^{\Gamma} \left[\left(\widehat{Z}_{j-1}\right)^{\Gamma} \left(\widehat{Z}_j\right)\right] \\
&= \left(W_j \widehat{Z}_{j-1}\right)^{\Gamma} \left(\widehat{Z}_{j-1}\right) \left(\widehat{Z}_{j-1}\right)^{\Gamma} \left(W_i \widehat{Z}_{j-1}\right) \\
&= \left(W_j \widehat{Z}_{j-1}\right)^{\Gamma} \left(W_j \widehat{Z}_{j-1}\right) \\
&= \widehat{Z}^{\Gamma}_{j-1} \left(W_j{}^{\Gamma} W_j\right) \widehat{Z}_{j-1}
\end{aligned}
\tag{3}
$$

F. Power Iteration Algorithm

The proposed framework estimates the spectral norm of the blocks in neural networks using the Power Iteration Algorithm. This method is chosen for its lightweight computation and continuous differentiability. Here is the pseudocode for the algorithm:

Algorithm 1: Numerical Estimation of Spectral Norm with Tensorflow Pseudocode, called *top_eigenvalue*

Input: Transmitting Matrix T_i, power iteration n
Output: The Largest Eigenvalue $\sigma(\cdot)$

v = tf.random.normal([T_i.shape[0], T_i.shape[1]])
for $i = 0 \to n$ **do**
    m = tf.matmul(T_i, v)
    μ = tf.sqrt(tf.reduce_sum(tf.square(m), axis = 1))
    $v = m/\mu$
v_norm = tf.sqrt(tf.reduce_sum(tf.square(v), axis = 1))
$\sigma(T_i)$ = tf.sqrt(μ/v_norm)
return $\sigma(T_i)$

Table 1. The impact of the target relevance to the source datasets on Expert Distilling phase

Method	UJIIndoorLoc		UTS	Tampere
	1% (199/19937)	10% (1993/19937)	10% (910/9108)	10% (69/697)
	MAE (m)↓	MAE (m)↓	MAE (m)↓	MAE (m) ↓
DNN [2]	160.56±0.24	18.28±0.41	10.65±5.52	25.04±14.99
DNN+++	**36.36**±**2.32**	**17.42**±**0.45**	**8.18**±**0.68**	**24.28**±**10.87**
CNNLoc [8]	22.14±1.09	14.07±0.27	7.34±0.16	15.99±9.58
CNNLoc+++	**21.93**±**1.17**	**13.73**±**0.36**	**6.96**±**0.34**	**12.78**±**1.09**
BayesCNN [7]	26.84±2.21	16.25±0.47	8.53±0.55	15.14±1.25
BayesCNN+++	**25.3**±**1.93**	**15.68**±**0.72**	**8.05**±**0.46**	**15.10**±**0.9**
bAaT [4, 5]	19.56±1.09	13.06±0.26	6.79±0.22	13.30±1.03
bAaT++ (wo J_{FI})	**18.18**±**0.86**	**12.88**±**0.18**	**6.58**±**0.18**	**12.33**±**0.89**

G. Pseudo code of the training pipeline

The framework executes the alignment through two primary phases. The first phase called *Expert Training*, involves modeling the representations established by the specialized networks on their respective source datasets using surrogate teacher networks. In the second phase, *Expert Distilling*, these modeled representations are collectively distilled into essential knowledge for alignment with representations learned on the target dataset.

Algorithm 2: Expert Training Phase

Input: Surrogate teachers $\{G_i\}_{i=1}^{N} = \{F_i^G, E_i^G\}_{i=1}^{N}$,
Critics $\{C_i\}_{i=1}^{N}$, c_step, Source dataset $D_{i=1}^{N}$,
Specialized models $\{S_i\}_{i=1}^{N} = \{F_i^S, E_i^S, R_i^S\}_{i=1}^{N}$,
Epoch E, gradient weight α, loss weights $\{\beta_i\}_{i=1}^{3}$
Output: Pre-trained surrogate teachers $\{G_i\}_{i=1}^{N}$

Initialize T_list
for $i = 1 \rightarrow N$ **do**
  for *epoch* $e = 1 \rightarrow E$ **do**
    Load D_i
    Initialize G_i, C_i, S_i
    for *batch* $b \in D_i$ **do**
      `/* First, training Critics for c steps */`
      for k *to* c_step **do**
        Initialize $Noise$
        $Z_{S_i}^F, Z_{S_i}^E, \hat{Y}_{S_i} = S_i(b)$
        $Z_{G_i}^F, Z_{G_i}^E = G_i(Noise)$
        $r\_logits = C_i(Z_{S_i}^E)$
        $f\_logits = C_i(Z_{G_i}^E)$
        $gp = grad\_penalty(C_i, b, Z_{S_i}^E, Z_{G_i}^E)$
        $L_c = critic\_loss(r\_logits, f\_logits)$
        $L_{tc} = L_c + \alpha * gp$
        $C_i = update(L_{tc}, C_i)$
      `/* Then, training Generators and Specialized model */`
      Initialize $Noise$
      $Z_{S_i}^F, Z_{S_i}^E, \hat{Y}_{S_i} = S_i(b)$
      $Z_{G_i}^F, Z_{G_i}^E = G_i(Noise)$
      $\hat{Y}_{G_i} = R_i^S(Z_{G_i}^E)$
      $L_{S\_MAE} = J_{MAE}(\hat{Y}_{S_i}, Y_{S_i})$
      $L_{G\_MAE} = J_{MAE}(\hat{Y}_{G_i}, Y_{S_i})$
      $L_{Sim} = J_{Sim}(Z_{S_i}^E, Z_{G_i}^E)$
      $L_{tg} = \beta_1 * L_{S\_MAE} + \beta_2 * L_{G\_MAE} + \beta_3 * L_{Sim}$
      $G_i = update(L_{tg}, G_i)$
      $S_i = update(L_{S\_MAE}, S_i)$
  $T\_list.append(G_i)$
return T_list

Algorithm 3: Expert Distilling Phase

Input: Surrogate teachers $\{G_i\}_{i=1}^{N} = \{F_i^G, E_i^G\}_{i=1}^{N}$,
Target dataset D_t
Mutual Information Estimator Ψ_θ,
Specialized models $S = \{F^S, E^S, R^S\}$,
Epoch E, loss weights $\lambda_{i=1}^{4}$
Output: Specialized models $S = \{F^S, E^S, R^S\}$

Load D_t
Initialize Ψ_θ, S
for $e = 1 \rightarrow E$ **do**
  for $b \in D_t$ **do**
    Initialize $Noise$, L_t^{MI}, L_t^{Sim}, L_t^{FI}
    $Z_S^F, Z_S^E, \hat{Y}_S = S(b)$
    $L_{S\_MAE} = J_{MAE}(\hat{Y}_{S_i}, Y_{S_i})$
    `/* Computing Functional Information in the specialized model S for comparison with other surrogate teachors */`
    $TM_S = transmitting\_matrix(Z_S^F, Z_S^E)$
    $FI_S = top\_eigenvalue(TM_S)$
    for $i = 1 \rightarrow N$ **do**
      Load G_i
      $Z_{G_i}^F, Z_{G_i}^E = G_i(Noise)$
      `/* Computing Mutual Information Constraint */`
      $product\_examples = concat(Z_{S_i}^E[1:], Z_{S_i}^E[0])$
      $joint\_stat = \Psi_\theta(Z_{S_i}^E, Z_{G_i}^E)$
      $product\_stat = \Psi_\theta(product\_examples, Z_{G_i}^E)$
      $L_i^{MI} = J_{MI}(joint\_stat, product\_stat)$
      $L_t^{MI} += L_i^{MI}$
      `/* Computing Angular Similarity Constraint */`
      $L_i^{Sim} = J_{Sim}(Z_{S_i}^E, Z_{G_i}^E)$
      $L_t^{Sim} += L_i^{Sim}$
      `/* Computing Functional Information Constraint */`
      $TM_{G_i} = transmitting\_matrix(Z_{G_i}^F, Z_{G_i}^E)$
      $FI_{G_i} = top\_eigenvalue(TM_{G_i})$
      $L_i^{FI} = J_{FI}(FI_S, FI_{G_i})$
      $L_t^{FI} += L_i^{FI}$
    $L_t = J_{overall}(\lambda_{i=1}^{4}, L_{S\_MAE}, L_t^{Sim}, L_t^{MI}, L_t^{FI})$
    $S = update(L_t, S)$
return $S = \{F^S, E^S, R^S\}$

References

[1] Abolfazl Farahani, Sahar Voghoei, Khaled Rasheed, and Hamid R Arabnia. A brief review of domain adaptation. *Advances in data science and information engineering: proceedings from ICDATA 2020 and IKE 2020*, pages 877–894, 2021. 1

[2] Gibrán Félix, Mario Siller, and Ernesto Navarro Alvarez. A fingerprinting indoor localization algorithm based deep learning. In *2016 eighth international conference on ubiquitous and future networks (ICUFN)*, pages 1006–1011. IEEE, 2016. 3

[3] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018. 2

[4] Son Minh Nguyen, Duc Viet Le, and Paul JM Havinga. Learning the world from its words: Anchor-agnostic transformers for fingerprint-based indoor localization. In *2023 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 150–159. IEEE, 2023. 3

[5] Son Minh Nguyen, Duc Viet Le, and Paul JM Havinga. Seeing the world from its words: All-embracing transformers for fingerprint-based indoor localization. *Pervasive and Mobile Computing*, 100:101912, 2024. 3

[6] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009. 1

[7] Shreya Sinha and Duc V Le. Completely automated cnn architecture design based on vgg blocks for fingerprinting localisation. In *2021 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pages 1–8. IEEE, 2021. 3

[8] Xudong Song, Xiaochen Fan, Chaocan Xiang, Qianwen Ye, Leyu Liu, Zumin Wang, Xiangjian He, Ning Yang, and Gengfa Fang. A novel convolutional neural network based indoor localization framework with wifi fingerprinting. *IEEE Access*, 7:110698–110709, 2019. 3