Semi-supervised Deep Transfer for Regression without Domain Alignment

Supplementary Material

A. Derivations

A.1. CRAFT: A MAP Estimation View

In this section, we show that the proposed method can be viewed as regularized training of neural networks. In particular, we show that optimization of the CRAFT objective gives us the maximum a posteriori (MAP) estimate of model parameters. We know the MAP estimate for the parameters θ (given dataset $\mathcal{D}$) is obtained by maximizing the posterior $\log p(\theta|\mathcal{D}) \propto \log p(\mathcal{D}|\theta) + \log p(\theta)$, where, $\log p(\mathcal{D}|\theta)$ is the log-likelihood, and $p(\theta)$ is the prior distribution (regularizer) over the model parameters. The subsections below describe the choices for these components.

A.1.1. Likelihood

We define the log-likelihood for a labeled data as: $\log p(\mathcal{D}|\theta) = \sum_{i=1}^{N} \log p(y_i|\mathbf{x}_i, \theta)$. As this paper deals with unlabeled data, as a design choice, we assume the conditional distribution $p(y|\mathbf{x}, \theta)$ to be constant for unlabeled training data, optimizing the log-likelihood over labeled data only.

Although we only address continuous label prediction (regression – $p(y|\mathbf{x}, \theta) = \mathcal{N}(y; f(\mathbf{x}; \theta), c)$) tasks in this study, the framework can be converted to k-way discrete label prediction (classification) tasks by appropriate assumptions on the conditional distribution – e.g., $p(y|\mathbf{x}, \theta) = \text{Mult}_k(y; f_1(\mathbf{x}; \theta), ..., f_k(\mathbf{x}; \theta))$.

A.1.2. The Prior Distribution

In this subsection, we derive a principled prior distribution that can leverage unlabeled data and/or deal with data scarcity during transfer learning by biasing the selection of parameters that best suit the domain. We achieve this by extending the CUDA [4] framework, originally used for domain adaptation.

We note a few desirable properties for supervised transfer learning algorithms. Firstly, these models are not explicitly trained to match the marginal label distribution, i.e., $p(y) \neq \int_{\mathbf{x}} p(y|\mathbf{x}, \theta) p(\mathbf{x}) d\mathbf{x}$. Hence, the model's performance on unseen data can be compromised if the training and the test label distributions differ, such as due to sampling biases. Hence, $p(y) = p(y|\theta)$ is a property we desire. Secondly, ensuring this property can be particularly useful if we know that the training dataset is small, where sampling biases are likely. Hence, we define a joint distribution using the trained model $p(y|\mathbf{x}, \theta)$ to tackle these challenges:

$$\mathrm{q}(\mathbf{x}, y|\theta) = \left[\frac{p(y|\mathbf{x}, \theta)}{\sum_{i=1}^{N} p(y|\mathbf{x}_i, \theta)} p(y) \right] \tag{7}$$

Now, we incorporate $q(\mathbf{x}, y|\theta)$ as a prior over the model parameters. We use the maximum entropy principle to achieve this. In particular, we wish to maximize the entropy of the model's parameter distribution such that the negative log-probability of the joint distribution q, on average is as small as a constant G, i.e., $\mathbb{E}_{\theta \in \mathbf{\Theta}}[-\log q(\mathbf{x}, y|\theta)] = G$. Solving the Euler-Lagrange equations we get (see Appendix A.2):

$$\log p(\theta) \propto \alpha \log q(\mathbf{x}, y|\theta) \tag{8}$$

where α is the Lagrange multiplier corresponding to G.

A.2. A Maximum Entropy Prior

The optimization objective described above can be expressed as:

$$\theta^* = \underset{\theta \in \Theta}{\text{argmax}} - \int_{\theta} p(\theta) \log p(\theta) \mathrm{d}\theta \tag{9}$$

$$\text{s.t.} \quad \mathbb{E}_{\theta \in \mathbf{\Theta}}[-\log q(\mathbf{x}, y|\theta)] = G, \int_{\theta} p(\theta) \mathrm{d}\theta = 1$$

We ignore the constraint on the density integral, and normalize it at the end. Hence, the Lagrangian can be written as:

$$\mathcal{L}(\theta) = \int_{\theta} \underbrace{p(\theta) \left(-\log p(\theta) + \alpha(\log q(\mathbf{x}, y|\theta) + G)\right)}_{g(\theta,\, p(\theta))} \mathrm{d}\theta$$

Using the Euler-Lagrangian equation, and because g does not depend on the first derivate of p:

$$\frac{\partial g}{\partial p} - \frac{\mathrm{d}}{\mathrm{d}\theta} \frac{\partial g}{\partial p'} = -1 - \log p(\theta) + \alpha(\log q(\mathbf{x}, y|\theta) + G) = 0$$

$$\implies \log p(\theta) = \alpha \log q(\mathbf{x}, y|\theta) + (\alpha G - 1)$$

Dropping the constant terms that are independent of θ:

$$p(\theta) \propto \alpha \log q(\mathbf{x}, y|\theta).$$

A.3. Mixtures of Gaussians and Exponentials

For learning the marginal distribution of labels for CRAFT we fit a mixture of exponential and Gaussian distributions. Given samples $\{\mathbf{x}_i\}_{i=1}^{N}$ from an unknown distribution p_x, s.t., $x_i \in \mathbb{R}_+$. If negative components exist, we can add a constant offset to make the data non-negative. We model

the underlying distribution by k_1 Gaussians and k_2 exponentials. The density of a mixture model p_θ, can be written as;

$$p_\theta(x) = \sum_{i=1}^{k_1+k_2} p_\theta(z)p_\theta(x|z) \tag{10}$$

where z is the latent variable, s.t., $p_\theta(z) = \text{Mult}_{k_1+k_2}(z; \beta_1, \beta_2, ..., \beta_{k_1+k_2})$. The conditionals are defined as:

$$p_\theta(x|z=i) = \begin{cases} \mathcal{N}(x; \mu_i, \sigma_i^2) & i \in \{1, 2, ..., k_1\} \\ \text{Exp}(x; \lambda_i) & i \in \{k_1+1, ..., k_1+k_2\} \end{cases}$$

We estimate the parameters $\theta = \{\{\beta_i\}_{i=1}^{k_1+k_2}, \{\mu_i, \sigma_i^2\}_{i=1}^{k_1}, \{\lambda_i\}_{i=k_1+1}^{k_2}\}$, using a custom designed expectation-maximization algorithm (code available in Appendix E.2).

B. Datasets and Methods

B.1. Gaze prediction from brain signals

Preprocessing. All eye-tracking datasets – Visual Search and Large Grid – were preprocessed similarly. Briefly, blinks were removed using Gaussian filtering, and the temporal locations of saccades were found by detecting very high velocity and acceleration. Then, the eye-tracking data was partitioned into 1s windows so that only one saccade event occurred in that window. The displacement vector of the eye was computed, and its magnitude (saccade amplitude) was designated the target variable $y \in \mathbb{R}^+$. Time-matched to each saccade, a 500Hz EEG signal was extracted and used as the feature vector $\mathbf{x} \in \mathbb{R}^{128\text{x}500}$ (128 electrodes or channels). Table 5 lists the details of the train, cross-validation, and test distributions for these datasets. We followed the leave-participant-out evaluation strategy with an identical train-validation-test split for the Large grid dataset for direct comparison with the benchmark. The Visual Search paradigm has more training data and was used to train the source model, which was transferred to the target datasets - Large Grid (from the benchmark). Before training the model, the target variable (saccade amplitude) was linearly scaled between $[-1, 1]$, and the EEG data were z-scored, by computing a common mean and standard deviation, across time and channels.

Models. The EEG-EyeNet benchmark showed that Pyramidal CNNs [47] performed best at predicting saccade amplitude. However, naive 2D convolution-based neural networks treat the timechannel EEG data as an image, whereas such data typically lack the correlation structure associated with natural images. Moreover, CNN-based models are not ideal for learning temporal autocorrelation structure present in the data.

Table 5. Training, validation and test sets for each dataset. A 60-20-20% leave-participant-out split was used, as in the original benchmark.

Dataset	Train	Validation	Test
Visual Search (Source)	21,191	5,018	5,354
Large Grid (Target)	12,275	2,836	2,719

We address these shortcomings with a novel end-to-end Long-Short-Term-Memory (LSTM) model trained on features extracted by an EEGNet-like [31] architecture. Figure 2B shows the feature extractor, which employs separable convolutions along time and electrode dimensions. The extracted temporal features are then fed to a series of two LSTMs, which predict the saccade amplitude.

In the feature extractor, the EEG data is first convolved along the temporal dimension using 100 ms trainable filters independently for each electrode, as it is the average duration to execute a saccade [10]. The filters are shifted by 10 ms to capture neural dynamics up to $\sim$100 Hz. Finally, depthwise-separable convolution is performed time-point-wise along the electrode dimension to obtain temporal features for the LSTM-based decoder.

B.2. Brain age prediction from structural MRI scans

Models. The Simple Fully Convolutional Network (SFCN, Fig. 2D) [41] feature Extractor contains five blocks of 3D Convolution (3x3x3 filters), Batch-Normalizaton, and 3D-MaxPooling (2x2x2 filters), and ReLU activation, followed by a single block of 3D Convolution with 1x1x1 filters, Batch-Normalizaton, and ReLU activation. The outputs of the feature extractor are then the Global Average Pooled (GAP) and fed as input to a classifier. In the original paper, the classifier predicted the probability of each MRI scan belonging to one of the 40 bins (range: 42-82 years). The expected value of the outputs was reported as the predicted age. In our version of SFCN, we avoid binning by replacing the classifier with a regressor.

B.3. People counting from natural scene images

We addressed the challenge of counting people from photographs of crowds – the "people counting" challenge. **Preprocessing**. High-resolution scene images from two datasets – NWPU [48] and JHU-crowd [44] – were resized to 1152x768. Following this, each image was divided into six non-overlapping 384x384 patches. These patches were fed to a ResNet-based feature extractor, to ultimately predict the number of people in the scene. The target dataset (JHU-crowd) contains images with 0-10,000 people, and thus, we restricted the training samples of the source dataset (NWPU) to those that have less than 10k humans in each

image. Table 6 shows the train-test-validation split of the datasets.

Table 6. Training, validation and test sets for each dataset. The same train-test-validation split was used, as in the original benchmark.

Dataset	Train	Validation	Test
NWPU (Source)	3,100	499	1,500
JHU-crowd (Target)	2,269	500	1,600

Models. Figure 4A shows the model used for people counting. The patches from each image were passed through a ResNet50 [25] model, pretrained on ImageNet [12], to get six 2048-dimensional encoding vectors. These vectors were then combined using an "attention" head to obtain a 64-dimensional embedding for the image. This was then fed to a dense layer to predict the number of people in the scene.

B.4. Tumor size estimation from histopathology images

We also addressed the challenge of estimating tumor sizes from cancer histopathology images.

Preprocessing. We employed the Camelyon-16 [16] and Camelyon-17 [6] high-resolution breast-cancer datasets. The patch level annotations available in these datasets render them particularly suitable for our analysis. We used 256x256 high-resolution patches from the whole-slide images (WSIs) for our analysis. The annotations in these patches were used to compute the fraction of the patch that has tumerous tissue. This fraction was predicted as the "tumor size" at the patch level. Table 7 shows the train-validation-test split for the source and target datasets. To render the transfer more challenging, we used only patches containing 20-80% tumor in the image for training the source model, while the target patches had tumor coverage ranging between 0.1-99.9%.

Table 7. Training, validation and test sets for each dataset. A 64-16-20% stratified split was used for both datasets.

Dataset	Train	Validation	Test
Camelyon-16 (Source)	65,337	16,335	20,418
Camelyon-17 (Target)	7,926	1,982	2,477

Models. The 256x256 patch was passed through a Resnet152v2 [24] model initialized to the ImageNet [12] weights (Fig. 4B). Next, the extracted features were passed through two dense layers to predict the fraction of tissue covered by the tumor.

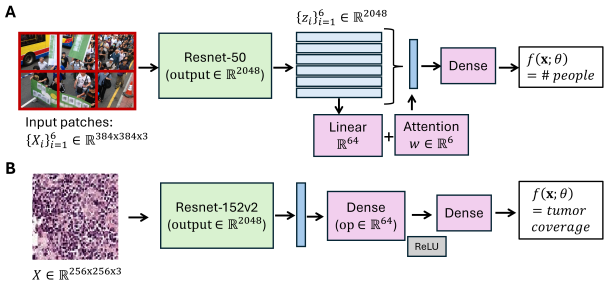

Figure 4. Model Architectures. (**A**) Predicting the number of people in a scene. The Resnet50 model is used to extract features from 6 patches of an image, along with an "attention" head that combines them, followed by a dense layer to make predictions; (**B**) Predicting the fraction of tumor cells in a histopathology image. The Resnet152v2 model is used to extract features, followed by predictions using dense layers.

C. Additional Benchmarks

C.1. People counting

We attempted transfer with models trained on a people counting task. First, we trained the Attn-Resnet model to count people with the NWPU dataset. Source predictions were reasonably effective, with RMSE=459.01 and R=0.74. Next, this pretrained model was transferred to the JHU-crowd dataset (see Appendix B.3). The upper 2 rows of Table 8 show the performance when all the data was used for finetuning. All models were trained with mini-batches of 16 samples, and only the best-performing checkpoint on the held-out validation set was used for evaluating the test set. Adam optimizer with an initial learning rate of 10^{-4} was used. Based on observations in the previous experiments (Section 5) $\alpha = 0.1$ was used as the weight for the unsupervised loss relative to the supervised loss.

To demonstrate the efficacy of CRAFT for SF-SSDA, target labels from the training data of the JHU-crowd dataset were removed randomly, but in a stratified manner. Figure 5A shows how the performance of these models is affected by reducing the fraction of labeled data (n_{ul}/N is the fraction of unlabeled samples). A lower proportion of labeled data worsens the performance of all the models (RMSE increases), but CRAFT outperforms other DA algorithms. Surprisingly, naive transfer learning performed better than SOTA algorithms for this problem. On further investigation, we observed that the model trained on the NWPU dataset readily generalized to the JHU-crowd dataset (RMSE=451.62, R=0.75), already performing close to the ceiling.

Figure 5B shows the prediction of people count in a scene for the best CRAFT-based Attn-Resnet model. Table 8 (lower 6 rows) summarizes the performance of CRAFT when 98% of the data were unlabeled; CRAFT

outperforms SOTA SF-SSDA models by $> 5\%$ in terms of RMSE. But supervised finetuning (TL) beat CRAFT by 17%. The correlation coefficient (R) also showed similar trends (Fig. 6C). Several SF-SSDA methods rely on predicting pseudo-labels from unlabeled target data, and it is possible that subpar pseudo-label prediction exacerbated the poor performance of these algorithms.

Table 8. Source Free Semi-supervised Deep Transfer of the model trained on the NWPU crowd dataset to the JHU dataset. The upper part of the table (rows 1-3) shows the performance ceiling when all labeled data in the target is used for training. The lower part (rows 4-9) shows SF-SSDA results when only 2% of the target samples were labeled (3 seeds).

Method	**R** ↑	**RMSE** (in #people) ↓
Naive Baseline	-	724.17 ± 1.11
Attn-Resnet (TL, 100%)	0.83 ± 0.02	429.25 ± 5.63
SF-SSDA (2% labels)		
Attn-Resnet + TL	$\mathbf{0.73 \pm 0.01}$	$\mathbf{459.05 \pm 4.41}$
Attn-Resnet + Mixup	0.34 ± 0.04	702.05 ± 8.26
Attn-Resnet + BBCN	0.32 ± 0.06	1221.36 ± 237.66
Attn-Resnet + TASFAR	0.27 ± 0.03	620.55 ± 37.28
Attn-Resnet + DataFree	0.67 ± 0.03	582.53 ± 6.63
Attn-Resnet + CRAFT	$\mathit{0.69 \pm 0.02}$	$\mathit{550.77 \pm 9.86}$

C.2. Tumor size estimation

Finally, we attempted transfer with models trained to estimate the fraction of tumor tissue in a high-resolution histopathological slide. The source model (Resnet152v2) was trained to predict what fraction of the patched Camelyon-16 dataset contained a tumor. Source predictions were fairly accurate, with RMSE=0.19 and R=0.46. Next, the model was transferred to make predictions on the Camelyon-17 dataset (see Appendix B.4). All models were trained with mini-batches of 32 samples, and only the best-performing checkpoint on the held-out validation set was used for evaluating the test set. Adam optimizer with an initial learning rate of 10^{-4} was used. As before, $\alpha = 0.01$ was used as the unsupervised loss weightage relative to the supervised loss.

The top of Table 9 (row 2) quantifies the performance ceiling when all the training data are used (3 seeds). In rows 3-8, we compare the performance of all the source-free models when only 1% of the target dataset was labeled. We observe that CRAFT outperformed all SF-SSDA methods by $> 6\%$ on the correlation coefficient (R). Only DataFree performs comparably in terms of RMSE. Figure 5C shows that CRAFT performs better than all competing methods when the fraction of unlabeled data is varied from 90-99%. Similar trends were observed for the correlation coefficient R (Fig. 6D). Figure 5D shows the prediction scatter of the best-performing CRAFT model when 90% of the data was labeled.

Table 9. Source Free Semi-supervised Deep Transfer of the model trained on Camelyon-16 dataset to the Camelyon-17 dataset. The upper part of the table (rows 1-3) shows the performance ceiling when all labeled data in the target is used for training. The lower part (rows 4-9) shows SF-SSDA results when only 1% of the target samples were labeled (3 seeds).

Method	**R** ↑	**RMSE** ↓
Naive Baseline	-	0.313 ± 0.003
Resnet (TL, 100%)	0.773 ± 0.003	0.203 ± 0.003
SF-SSDA (1% labels)		
Resnet + TL	0.577 ± 0.026	0.260 ± 0.005
Resnet + Mixup	0.490 ± 0.009	0.283 ± 0.003
Resnet + BBCN	0.457 ± 0.084	0.307 ± 0.017
Resnet + TASFAR	0.617 ± 0.020	0.290 ± 0.012
Resnet + DataFree	$\mathit{0.630 \pm 0.012}$	$\mathit{0.247 \pm 0.003}$
Resnet + CRAFT	$\mathbf{0.670 \pm 0.009}$	$\mathbf{0.243 \pm 0.003}$

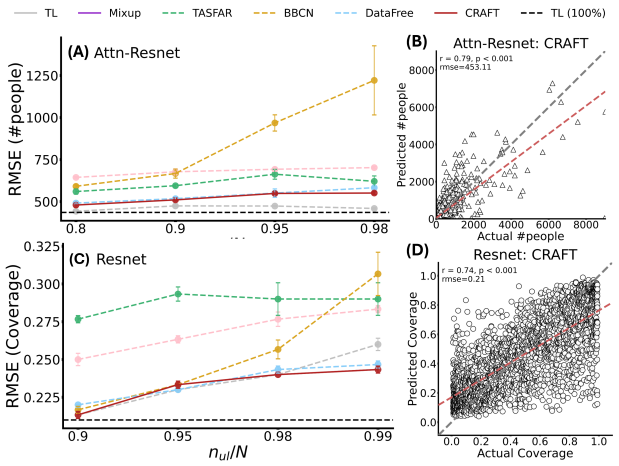

Figure 5. (**A, C**) RMSE (↓) for source-free semi-supervised domain adaptation of people counting (**A**) and tumor size (**C**) prediction tasks, trained with varying proportions of unlabeled target data n_{ul}/N. Dashed black line: Performance ceiling. (**B, D**) Predictions for the best CRAFT model for people counting (**B**) and tumor size prediction (**D**), with 80% and 90% unlabeled data, respectively.

D. Analysis of hyperparameters: bin size, α

Bin Size. We varied the number of bins in the pseudo-label selection step of CRAFT to quantify the effect of bin sizes. We found the algorithm to be fairly robust to the choice of bin sizes for both the neuroscience datasets (Table 10).

Unsupervised loss weight (α). For the Saccade prediction task, a cross-validation set was available and hence, only the best-performing model, based on the corresponding α, is reported. However, a validation set was unavail-

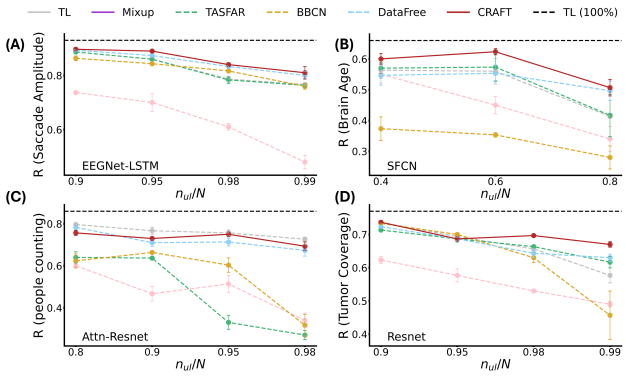

Figure 6. Correlation Coefficient R (↑) for all the competing SF-SSDA models at predicting (**A**) Saccade Amplitude; (**B**) Brain Age; (**C**) Number of People; and (**D**) Tumor Coverage.

Table 10. Effect of bin size on the performance of CRAFT for SF-SSDA for the saccade prediction (5% labeled) and brain-age prediction (60% labeled) tasks.

	EEGNet-LSTM ($P\sim 190K$)		SFCN ($P\sim 2.95M$)	
Bin Size % Range(y)	**R** ↑	**RMSE** ↓	**R** ↑	**RMSE** ↓
0.50	0.88 ± 0.01	67.25 ± 0.46	0.65 ± 0.03	6.53 ± 0.15
0.33	0.89 ± 0.01	65.84 ± 0.98	0.55 ± 0.02	6.91 ± 0.09
0.25	0.89 ± 0.01	67.39 ± 1.39	0.62 ± 0.01	6.60 ± 0.02
0.20	0.89 ± 0.01	65.21 ± 0.89	0.64 ± 0.02	6.36 ± 0.09

able for brain age prediction. Hence, we report the performance across three α values $\{0.01, 0.1, 1.0\}$ in Table 11. While CRAFT outperforms competing models for low values of alpha, large weightages to the unsupervised loss (e.g., $\alpha = 1$) result in comparatively poor performance.

Table 11. Effect of α (unsupervised loss weight) on the performance of CRAFT and other competing methods for SF-SSDA for the brain-age prediction (20% labeled) tasks.

	TASFAR		DataFree		**CRAFT**	
α	**R** ↑	**RMSE** ↓	**R** ↑	**RMSE** ↓	**R** ↑	**RMSE** ↓
0.01	0.40 ± 0.07	7.53 ± 0.18	0.31 ± 0.04	7.82 ± 0.20	**0.51 ± 0.03**	**7.31 ± 0.17**
0.10	0.42 ± 0.07	7.47 ± 0.15	0.50 ± 0.05	7.35 ± 0.15	**0.51 ± 0.03**	**7.14 ± 0.11**
1.00	**0.40 ± 0.06**	**7.47 ± 0.21**	0.07 ± 0.02	8.70 ± 0.06	0.37 ± 0.03	8.64 ± 0.15

E. Further Details

E.1. Compute Resources and Software

All the algorithms were implemented using TensorFlow 2.x, and experiments were run on 24 GB RTX-4090Ti, 32 GB V100, and 40 GB A100 GPUs.

E.2. Code Availability

The implementation of all the methods described in the study is publicly available in the following repository – https://github.com/mainak-biswas1999/CRAFT_ICCV2025.git.