Supplementary Material

Transfer Learning with Affine Model Transformation

A Other Perspectives on Affine Model Transfer

A.1 Transformation Functions for General Loss Functions

Here we discuss the optimal transformation function for general loss functions.

Let $\ell(y, y') \geq 0$ be a convex loss function that returns zero if and only if $y = y'$, and let $g^*(x)$ be the optimal predictor that minimizes the expectation of ℓ with respect to the distribution p_t followed by x and y transformed by ϕ:

$$g^*(x) = \arg\min_g \mathbb{E}_{p_t}\left[\ell(g(x), \phi_{f_s}(y))\right].$$

The function g that minimizes the expected loss

$$\mathbb{E}_{p_t}\left[\ell(g(x), \phi_{f_s}(y))\right] = \iint \ell(g(x), \phi_{f_s}(y)) p_t(x, y) \mathrm{d}x \mathrm{d}y$$

should be a solution to the Euler-Lagrange equation

$$\frac{\partial}{\partial g(x)} \int \ell(g(x), \phi_{f_s}(y)) p_t(x, y) \mathrm{d}y = \int \frac{\partial}{\partial g(x)} \ell(g(x), \phi_{f_s}(y)) p_t(y|x) \mathrm{d}y \, p_t(x) = 0. \tag{S.1}$$

Denote the solution of Eq. (S.1) by $G(x; \phi_{f_s})$. While G depends on the loss ℓ and distribution p_t, we omit those from the argument for notational simplicity. Using this function, the minimizer of the expected loss $\mathbb{E}_{x,y}[\ell(g(x), y)]$ can be expressed as $G(x; \mathrm{id})$, where id represents the identity function.

Here, we consider the following assumption to hold, which generalizes Assumption 2.3 in the main text:

Assumption 2.3(b). *For any distribution on the target domain $p_t(x, y)$ and all $x \in \mathcal{X}$, the following relationship holds:*

$$\psi_{f_s}(g^*(x)) = \arg\min_g \mathbb{E}_{x,y}[\ell(g(x), y)].$$

Equivalently, the transformation functions ϕ_{f_s} and ψ_{f_s} satisfy

$$\psi_{f_s}\big(G(x; \phi_{f_s})\big) = G(x; \mathrm{id}). \tag{S.2}$$

Assumption 2.3(b) states that if the optimal predictor $G(x; \phi_{f_s})$ for the data transformed by ϕ is given to the model transformation function ψ, it is consistent with the overall optimal predictor $G(x; \mathrm{id})$ in the target region in terms of the loss function ℓ. We consider all pairs of ψ and ϕ that satisfy this consistency condition.

Here, let us consider the following proposition:

Proposition A.1. *Under Assumption 2.1, 2.2 and 2.3(b), $\psi_{f_s}^{-1} = \phi_{f_s}$.*

Proof. The proof is analogous to that of Theorem 2.4 in the main text. For any $y_0 \in \mathcal{Y}$, let $p_t(y|x) = \delta_{y_0}$. Combining this with Eq. (S.1) leads to

$$\frac{\partial}{\partial g(x)} \ell(g(x), \phi_{f_s}(y_0)) = 0 \ (\forall y_0 \in \mathcal{Y}).$$

Because $\ell(y, y')$ returns the minimum value zero if and only if $y = y'$, we obtain $G(x; \phi_{f_s}) = \phi_{f_s}(y_0)$. Similarly, we have $G(x; \mathrm{id}) = y_0$. From these two facts and Assumption 2.3(b), we have $\psi_{f_s}(\phi_{f_s}(y_0)) = y_0$, proving that the proposition is true. □

Proposition A.1 indicates that the first statement of Theorem 2.4 holds for general loss functions. However, the second claim of Theorem 2.4 generally depends on the type of loss function. Through the following examples, we describe the optimal class of transformation functions for several loss functions.

Example A.1 (Squared loss)**.** Let $\ell(y, y') = |y - y'|^2$. As a solution of Eq. (S.1), we can see that the optimal predictor is the conditional expectation $\mathbb{E}_{p_t}[\phi_{f_s}(Y)|X = x]$. As discussed in Section 2 in the main paper, the transformation functions ϕ_{f_s} and ψ_{f_s} should be affine transformations.

Example A.2 (Absolute loss)**.** Let $\ell(y, y') = |y - y'|$. Substituting this into Eq. (S.1), we have

$$\begin{aligned} 0 &= \int \frac{\partial}{\partial g(x)} \big|g(x) - \phi_{f_s}(y)\big| p_t(y|x) \mathrm{d}y \\ &= \int \operatorname{sign}\big(g(x) - \phi_{f_s}(y)\big) p_t(y|x) \mathrm{d}y \\ &= \int_{\phi_{f_s}(y) \geq g(x)} p_t(y|x) \mathrm{d}y - \int_{\phi_{f_s}(y) < g(x)} p_t(y|x) \mathrm{d}y. \end{aligned}$$

Assuming that ϕ_{f_s} is monotonically increasing, we have

$$0 = \int_{y \geq \phi_{f_s}^{-1}(g(x))} p_t(y|x) \mathrm{d}y - \int_{y < \phi_{f_s}^{-1}(g(x))} p_t(y|x) \mathrm{d}y.$$

This yields

$$\int_{\phi_{f_s}^{-1}(g(x))}^{\infty} p_t(y|x) \mathrm{d}y = \int_{-\infty}^{\phi_{f_s}^{-1}(g(x))} p_t(y|x) \mathrm{d}y.$$

The same result is obtained even if ϕ_{f_s} is monotonically decreasing. Consequently,

$$\phi_{f_s}^{-1}(g(x)) = \mathrm{Median}[Y|X = x],$$

which results in

$$G(x; \phi_{f_s}) = \phi_{f_s}\big(\mathrm{Median}[Y|X = x]\big).$$

This implies that Eq. (S.2) holds for any ϕ_{f_s} including an affine transformation, and the function form cannot be identified. from this analysis.

Example A.3 (Exponential-squared loss)**.** As an example where the affine transformation is not optimal, consider the loss function $\ell(y, y') = |e^y - e^{y'}|^2$. Substituting this into Eq. (S.1), we have

$$\begin{aligned} 0 &= \int \frac{\partial}{\partial g(x)} \big|\exp(g(x)) - \exp(\phi_{f_s}(y))\big|^2 p_t(y|x) \mathrm{d}y \\ &= 2 \exp(g(x)) \int \big(\exp(g(x)) - \exp(\phi_{f_s}(y))\big) p_t(y|x) \mathrm{d}y. \end{aligned}$$

Therefore,

$$G(x; \phi_{f_s}) = \log \mathbb{E}\big[\exp(\phi_{f_s}(Y))|X = x\big].$$

Consequently, Eq. (S.2) becomes

$$\log \mathbb{E}\big[\exp(\phi_{f_s}(Y))\big] = \phi_{f_s}\big(\log \mathbb{E}\big[\exp(Y)\big]\big).$$

Even if ϕ_{f_s} is an affine transformation, this equation does not generally hold.

A.2 Analysis of the Optimal Function Class Based on the Upper Bound of the Estimation Error

Here, we discuss the optimal class for the transformation function based on the upper bound of the estimation error.

In addition to Assumptions 2.1 and 2.2, we assume the following in place of Assumption 2.3:

Assumption A.2. The transformation functions ϕ and ψ are Lipschitz continuous with respect to the first argument, i.e., there exist constants μ_ϕ and μ_ψ such that,

$$\phi(a, c) - \phi(a', c) \leq \mu_\phi \|a - a'\|_2, \quad \psi(b, c) - \psi(b', c) \leq \mu_\psi \|b - b'\|_2,$$

for any $a, a' \in \mathcal{Y}$ and $b, b' \in \mathbb{R}$ with any given $c \in \mathcal{F}_s$.

Note that each Lipschitz constant is a function of the second argument f_s, i.e., $\mu_\phi = \mu_\phi(f_s)$ and $\mu_\psi = \mu_\psi(f_s)$.

Under Assumptions 2.1, 2.2 and A.2, the estimation error is upper bounded as follows:

$$\begin{aligned}
\mathop{\mathbb{E}}_{x,y}\Big[|f_t(x) - \hat{f}_t(x)|^2\Big] &= \mathop{\mathbb{E}}_{x,y}\Big[\big|\psi(g(x), f_s(x)) - \psi(\hat{g}(x), f_s(x))\big|^2\Big] \\
&\leq \mathop{\mathbb{E}}_{x,y}\Big[\mu_\psi(f_s(x))^2\big|g(x) - \hat{g}(x)\big|^2\Big] \\
&\leq 3\mathop{\mathbb{E}}_{x,y}\Big[\mu_\psi(f_s(x))^2\big(\big|g(x) - \phi(f_t(x), f_s(x))\big|^2 \\
&\qquad\qquad\qquad + \big|\phi(f_t(x), f_s(x)) - \phi(y, f_s(x))\big|^2 \\
&\qquad\qquad\qquad + \big|\phi(y, f_s(x)) - \hat{g}(x)\big|^2\big)\Big] \\
&\leq 3\mathop{\mathbb{E}}_{x,y}\Big[\mu_\psi(f_s(x))^2\big|\psi^{-1}(f_t(x), f_s(x)) - \phi(f_t(x), f_s(x))\big|^2\Big] \\
&\qquad\qquad\qquad + 3\mathop{\mathbb{E}}_{x,y}\Big[\mu_\psi(f_s(x))^2\mu_\phi(f_s(x))^2\big|f_t(x) - y\big|^2\Big] \\
&\qquad\qquad\qquad + 3\mathop{\mathbb{E}}_{x,y}\Big[\mu_\psi(f_s(x))^2\big|z - \hat{g}(x)\big|^2\Big] \\
&= 3\mathop{\mathbb{E}}_{x,y}\Big[\mu_\psi(f_s(x))^2\big|\psi^{-1}(f_t(x), f_s(x)) - \phi(f_t(x), f_s(x))\big|^2\Big] \\
&\qquad\qquad\qquad + 3\sigma^2\mathop{\mathbb{E}}_{x,y}\Big[\mu_\psi(f_s(x))^2\mu_\phi(f_s(x))^2\Big] \\
&\qquad\qquad\qquad + 3\mathop{\mathbb{E}}_{x,y}\Big[\mu_\psi(f_s(x))^2\big|z - \hat{g}(x)\big|^2\Big].
\end{aligned}$$

The derivation of this inequality is based on [1]. We use the Lipschitz property of ψ and ϕ for the first and third inequalities, and the second inequality comes from the inequality $(a-d)^2 \leq 3(a-b)^2 + 3(b-c)^2 + 3(c-d)^2$ for $a, b, c, d \in \mathbb{R}$.

According to this inequality, the upper bound of the estimation error is decomposed into three terms: the discrepancy between the two transformation functions, the variance of the noise, and the estimation error for the transformed data. Although it is intractable to find the optimal solution of $\phi, \psi, \hat{g}$ that minimizes all these terms together, it is possible to find a solution that minimizes the first and second terms expressed as the functions of ϕ and ψ only. Obviously, the first term, which represents the discrepancy between the two transformation functions, reaches its minimum (zero) when $\phi_{f_s} = \psi_{f_s}^{-1}$. The second term, which is related to the variance of the noise, is minimized when the differential coefficient $\frac{\partial}{\partial u}\psi_{f_s}(u)$ is a constant, i.e., when ψ_{f_s} is a linear function. This is verified as follows. From $\phi_{f_s} = \psi_{f_s}^{-1}$ and the continuity of ψ_{f_s}, it follows that

$$\mu_\psi = \max\Big|\frac{\partial}{\partial u}\psi_{f_s}(u)\Big|, \quad \mu_\phi = \max\Big|\frac{\partial}{\partial u}\psi_{f_s}^{-1}(u)\Big| = \frac{1}{\min|\frac{\partial}{\partial u}\psi_{f_s}(u)|},$$

and thus the product $\mu_\phi\mu_\psi$ takes the minimum value (one) when the maximum and minimum of the differential coefficient are the same. Therefore, we can write

$$\phi(y, f_s) = \frac{y - g_1(f_s)}{g_2(f_s)}, \quad \psi(g(x), f_s) = g_1(f_s) + g_2(f_s)g(x),$$

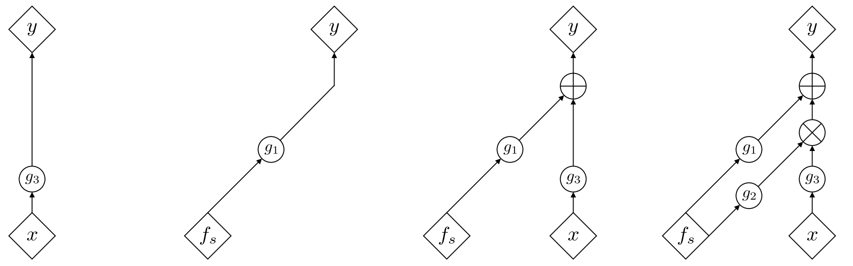
where $g_1, g_2 : \mathcal{F}_s \to \mathbb{R}$ are arbitrarily functions. Thus, the minimization of the third term in the upper bound of the estimation error can be expressed as

$$\min_{g_1, g_2, g} \mathop{\mathbb{E}}_{x,y}|y - g_1(f_s) + g_2(f_s)g(x)|^2.$$

As a result, the suboptimal function class for the upper bound of the estimated function is given as

$$\mathcal{H} = \big\{x \mapsto g_1(f_s) + g_2(f_s)\cdot g_3(x) \mid g_1 \in \mathcal{G}_1, g_2 \in \mathcal{G}_2, g_3 \in \mathcal{G}_3\big\}.$$

This is the same function class derived in Section 2 in the main paper.



(a) Direct learning (b) Feature extraction (c) HTL-offset (d) Affine model transfer

Figure S.1: Model architectures for the affine model transfer and related procedures. (a) Direct learning predicts outputs using only the original inputs x, while (b) feature extraction-based neural transfer predicts outputs using only the source features f_s. (c) The HTL procedure proposed in [2] (HTL-offset) constructs the predictor as the sum of $g_1(f_s)$ and $g_3(x)$. (d) The affine model transfer encompasses these procedures, computing g_1 and g_2 as functions of the source features and constructing the predictor as an affine combination with g_3.

B Additional Experiments

B.1 Eigenvalue Decay of the Hadamard Product of Two Gram Matrices

We experimentally investigated how the decay rate s_2 in Theorem 4.4 is related to the overlap degree in the spaces spanned by the original input x and source features f_s.

For the original input $x \in \mathbb{R}^{100}$, we randomly constructed a set of 10 orthonormal bases, and then generated 100 samples from their spanning space. For the source features $f_s \in \mathbb{R}^{100}$, we selected d bases randomly from the 10 orthonormal bases selected for x and the remaining $10 - d$ bases from their orthogonal complement space. We then generated 100 samples of f_s from the space spanned by these 10 bases. The overlap number d can be regarded as the degree of overlap of two spaces spanned by the samples of x and f_s. We generated the 100 different sample sets of x and f_s.

We calculated the Hadamard product of the Gram matrices K_2 and K_3 using the samples of x and f_s, respectively. For the computation of K_2 and K_3, all combinations of the following five kernels were tested:

$$\textbf{Linear kernel}\quad k(x, x') = \frac{x^\top x}{2\gamma^2} + 1,$$

$$\textbf{Matérn kernel}\quad k(x,x') = \frac{2^{1-\nu}}{\Gamma(\nu)}\left(\frac{\sqrt{2\nu}\|x-x'\|_2}{\gamma}\right)^{\nu} K_\nu\left(\frac{\sqrt{2\nu}\|x-x'\|_2}{\gamma}\right) \text{ for } \nu = \frac{1}{2}, \frac{3}{2}, \frac{5}{2}, \infty,$$

where $K_\nu(\cdot)$ is a modified Bessel function and $\Gamma(\cdot)$ is the gamma function. Note that for $\nu = \infty$, the Matérn kernel is equivalent to the Gaussian RBF kernel. The scale parameter γ of both kernels was set to $\gamma = \sqrt{\dim(x)} = \sqrt{10}$. For a given matrix K, the decay rate of the eigenvalues was estimated as the smallest value of s that satisfies $\lambda_i \leq \|K\|_F^2 \cdot i^{-\frac{1}{s}}$ where $\|\cdot\|_F$ denotes the Frobenius norm. Note that this inequality holds for any matrices K with $s = 1$ [3].

Figure S.2 shows the change of the decay rates with respect to varying d for various combinations of the kernels. In all cases, the decay rate of $K_2 \circ K_3$ showed a clear trend of monotonically decreasing as the degree of overlap d increases. In other words, the greater the overlap between the spaces spanned by x and f_s, the smaller the decay rate, and the smaller the complexity of the RKHS $\mathcal{H}_2 \otimes \mathcal{H}_3$.

B.2 Lattice Thermal Conductivity of Inorganic Crystals

Here, we describe the relationship between the qualitative differences in source features and the learning behavior of the affine model transfer, in contrast to ordinary feature extraction using neural networks.

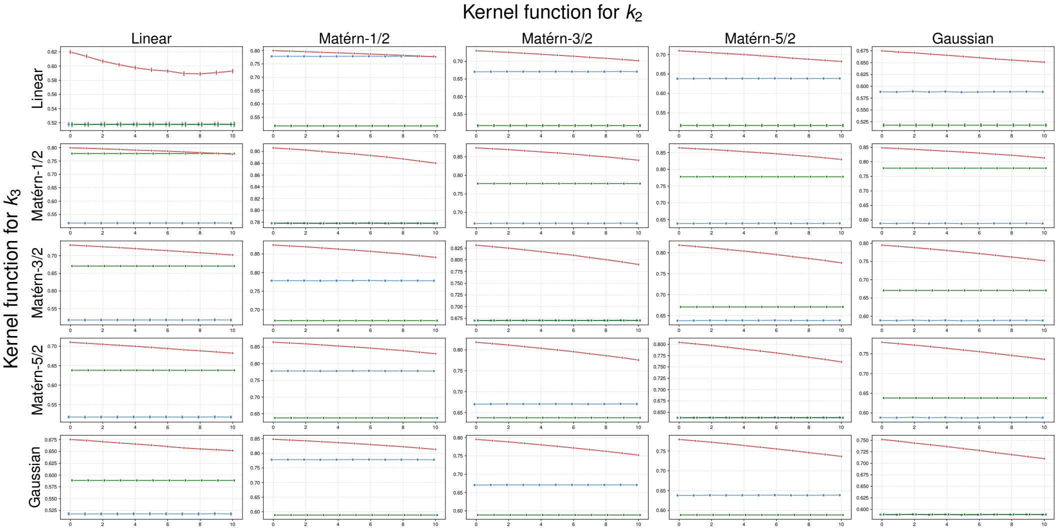


Figure S.2: Decay rates of eigenvalues of K_2 (blue lines), K_3 (green lines) and $K_2 \circ K_3$ (red lines) for all combinations of the five different kernels. The vertical axis represents the decay rate, and the horizontal axis represents the overlap dimension d in the space where x and f_s are distributed.

The target task is to predict the lattice thermal conductivity (LTC) of inorganic crystalline materials, where the LTC is the amount of vibrational energy propagated by phonons in a crystal. In general, LTC can be calculated ab initio by performing many-body electronic structure calculations based on quantum mechanics. However, it is quite time-consuming to perform the first-principles calculations for thousands of crystals, which will be used as a training sample set to create a surrogate statistical model. Therefore, we perform TL with the source task of predicting an alternative, computationally tractable physical property called scattering phase space (SPS), which is known to be physically related to LTC.

B.2.1 Data

We used the dataset from [4] that records SPS and LTC for 320 and 45 inorganic compounds, respectively. The input compounds were translated to 290-dimensional compositional descriptors using XenonPy [5][1].

B.2.2 Model Definition and Hyperparameter Search

Fully connected neural networks were used for both the source and target models, with a LeakyReLU activation function with $\alpha = 0.01$. The model training was conducted using the Adam optimizer [6]. Hyperparameters such as the width of the hidden layer, learning rate, number of epochs, and regularization parameters were adjusted with 5-fold cross-validation. For more details on the experimental conditions and procedure, refer to the provided Python code.

Source Model For the preliminary step, neural networks with three hidden layers that predict SPS were trained using 80% of the 320 samples. 100 models with different numbers of neurons were randomly generated and the top 10 source models that showed the highest generalization performance in the source domain were selected. The hidden layer width L was randomly chosen from the range $[50, 100]$, and we trained a neural network with a structure of (input)-L-L-L-1. Each of the three hidden layers of the source model was used as an input to the transfer models, and we examined the difference in prediction performance for the three layers.

Target Model In the target task, an intermediate layer of a source model was used as the feature extractor. A model was trained using 40 randomly chosen samples of LTC, and its performance was evaluated with the remaining 5 samples. For each of the 10 source models, we performed the

---

[1] `https://github.com/yoshida-lab/XenonPy`

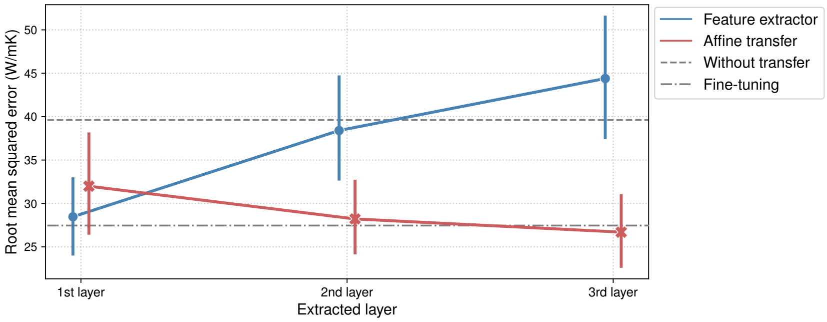


Figure S.3: Change of RMSE values between the affine transfer model and the ordinary feature extractor when using different levels of intermediate layers as the source features. The line plot shows the mean and 95% confidence interval. As a baseline, RMSE values for direct learning without transfer and fine-tuned neural networks are shown as dotted and dashed lines, respectively.

training and testing 10 times with different sample partitions and compared the mean values of RMSE among four different methods: (i) the affine model transfer using neural networks to model the three functions g_1, g_2 and g_3, (ii) a neural network using the XenonPy compositional descriptors as input without transfer, (iii) a neural network using the source features as input, and (iv) fine-tuning of the pre-trained neural networks. The width of the layers of each neural network, the number of training epochs, and the dropout rate were optimized during 5-fold cross-validation looped within each training set. For the affine model transfer, the functions g_1, g_2, and g_3 were modeled by neural networks. We used neural networks with one hidden layer for g_1, g_2 and g_3.

B.2.3 Results

Figure S.3 shows the change in prediction performance of TL models using source features obtained from different intermediate layers from the first to the third layers. The affine transfer model and the ordinary feature extractor showed opposite patterns. The performance of the feature extractor improved when the first intermediate layer closest to the input layer was used as the source features and gradually degraded when layers closer to the output were used. When the third intermediate layer was used, a negative transfer occurred in the feature extractor as its performance became worse than that of the direct learning. In contrast, the affine transfer model performs better as the second and third intermediate layers closer to the output were used. The affine transfer model using the third intermediate layer reached a level of accuracy slightly better than fine-tuning, which intuitively uses more information to transfer than the extracted features.

In general, the features encoded in an intermediate layer of a neural network are more task-independent as the layer is closer to the input, and the features are more task-specific as the layer is closer to the output [7]. Because the first layer does not differ much from the original input, using both x and f_s in the affine model transfer does not contribute much to performance improvement. However, when using the second and third layers as the feature extractors, the use of both x and f_s contributes to improving the expressive power of the model, because the feature extractors have acquired different representational capabilities from the original input. In contrast, a model based only on f_s from a source task-specific feature extractor could not account for data in the target domain, so its performance would become worse than direct learning without transfer, i.e., a negative transfer would occur.

B.3 Heat Capacity of Organic Polymers

We highlight the benefits of separately modeling and estimating domain-specific factors through a case study in polymer chemistry. The objective is to predict the specific heat capacity at constant pressure C_{P} of any given organic polymer with its chemical structure in the polymer's repeating unit. Specifically, we conduct TL to bridge the gap between experimental values and physical properties calculated from molecular dynamics (MD) simulations.

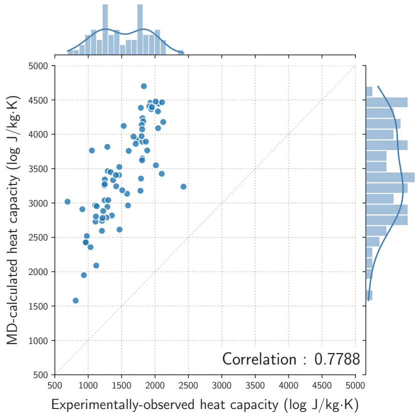


Figure S.4: MD-calculated (vertical axis) and experimental values (horizontal axis) of the specific heat capacity at constant pressure for various amorphous polymers.

Table S.1: Force field parameters that form the General AMBER force field [8] version 2 (GAFF2), and their detailed descriptions.

Parameter	Description
mass	Atomic mass
σ	Equilibrium radius of van der Waals (vdW) interactions
ϵ	Depth of the potential well of vdW interactions
charge	Atomic charge of Gasteiger model
r_0	Equilibrium length of chemical bonds
K_{bond}	Force constant of bond stretching
polar	Bond polarization defined by the absolute value of charge difference between atoms in a bond
θ_0	Equilibrium angle of bond angles
K_{angle}	Force constant of bond bending
K_{dih}	Rotation barrier height of dihedral angles

As shown in Figure S.4, there was a large systematic bias between experimental and calculated values; the MD-calculated properties $C_{\mathrm{P}}^{\mathrm{MD}}$ exhibited an evident overestimation with respect to their experimental values. As discussed in [9], this bias is inevitable because classical MD calculations do not reflect the presence of quantum effects in the real system. According to Einstein's theory for the specific heat in physical chemistry, the logarithmic ratio between $C_{\mathrm{P}}^{\exp}$ and $C_{\mathrm{P}}^{\mathrm{MD}}$ can be calibrated by the following equation:

$$\log C_{\mathrm{P}}^{\exp} = \log C_{\mathrm{P}}^{\mathrm{MD}} + 2\log\left(\frac{\hbar\omega}{k_B T}\right) + \log\frac{\exp\left(\frac{\hbar\omega}{k_B T}\right)}{\left[\exp\left(\frac{\hbar\omega}{k_B T}\right) - 1\right]^2}, \tag{S.3}$$

where k_B is the Boltzmann constant, $\hbar$ is the Planck constant, ω is the frequency of molecular vibrations, and T is the temperature. The bias is a monotonically decreasing function of frequency ω, which is described as a black-box function of polymers with their molecular features. Hereafter, we consider the calibration of this systematic bias using the affine transfer model.

B.3.1 Data

Experimental values of the specific heat capacity of the 70 polymers were collected from PoLyInfo [10]. The MD simulation was also applied to calculate their heat capacities. For models to predict the log-transformed heat capacity, a given polymer with its chemical structure was translated into the 190-dimensional force field descriptors, using RadonPy [9][2].

The force field descriptor represents the distribution of the ten different force field parameters ($t \in \mathcal{T} = \{\text{mass}, \sigma, \epsilon, \text{charge}, r_0, K_{\text{bond}}, \text{polar}, \theta_0, K_{\text{angle}}, K_{\text{dih}}\}$ that make up the empirical potential (i.e., the General AMBER force field [8] version 2 (GAFF2)) of the classical MD simulation. The

---

[2] `https://github.com/RadonPy/RadonPy`

Algorithm S.1 Block relaxation algorithm for the model in Eq. (S.4).

```
**Initialize:** $\alpha_0 \leftarrow \hat{\alpha}_{0,\mathrm{olr}}$, $\alpha_1 \leftarrow \hat{\alpha}_{1,\mathrm{olr}}$, $\beta \leftarrow 0$, $\gamma \leftarrow \hat{\gamma}_{\mathrm{diff}}$
**repeat**
    $\alpha \leftarrow \arg\min_{\alpha} F_{\alpha,\beta,\gamma}$
    $\beta \leftarrow \arg\min_{\beta} F_{\alpha,\beta,\gamma}$
    $\gamma \leftarrow \arg\min_{\gamma} F_{\alpha,\beta,\gamma}$
**until** convegence
```

detailed descriptions for each parameter are listed in Table S.1. For each t, pre-defined values are assigned to their constituent elements in a polymer, such as individual atoms (mass, charge, σ, and ϵ), bonds (r_0, K_{bond}, and polar), angles (θ_0 and K_{angle}), or dihedral angles (K_{dih}), respectively. The probability density function of the assigned values of t is then estimated and discretized into 10 points corresponding to 10 different element species such as hydrogen and carbon for mass, and 20 equally spaced grid points for the other parameters.

The source feature f_s was given as the log-transformed value of $C_{\mathrm{P}}^{\mathrm{MD}}$. Therefore, f_s is no longer a function of x; this modeling was intended for calibrating the MD-calculated properties.

We randomly sampled 60 training polymers and tested the prediction performance of a trained model on the remaining 10 polymers 20 times. The PoLyInfo sample identifiers for the selected polymers are listed in the code.

B.3.2 Model Definition and Hyperparameter Search

As described above, the 190-dimensional force field descriptor consists of ten blocks corresponding to different types of features. The J_t features that make up block t represent discretized values of the density function of the force field parameters assigned to the atoms, bonds, or dihedral angles that constitute the given polymer. Therefore, the regression coefficients of the features within a block should be estimated smoothly. To this end, we imposed fused regularization on the parameters as

$$\lambda_1 \|\gamma\|_2^2 + \lambda_2 \sum_{t \in \mathcal{T}} \sum_{j=2}^{J_t} \big(\gamma_{t,j} - \gamma_{t,j-1}\big)^2,$$

where $\mathcal{T} = \{\mathrm{mass}, \mathrm{charge}, \epsilon, \sigma, K_{\mathrm{bond}}, r_0, K_{\mathrm{angle}}, \theta, K_{\mathrm{dih}}\}$, and $J_t = 10$ for $t = \mathrm{mass}$ and $J_t = 20$ otherwise. The regression coefficient $\gamma_{t,j}$ corresponds to the j-th feature of block t.

Ordinary Linear Regression The experimental heat capacity $y = \log C_{\mathrm{P}}^{\mathrm{exp}}$ was regressed on the MD-calculated property, without regularization, as $\hat{y} = \alpha_0 + \alpha_1 f_s$ where $\hat{y}$ denotes the conditional expectation and $f_s = \log C_{\mathrm{P}}^{\mathrm{MD}}$.

Learning the Log-Difference We calculated the log-difference $\log C_{\mathrm{P}}^{\mathrm{exp}} - \log C_{\mathrm{P}}^{\mathrm{MD}}$ and trained the linear model with the ridge penalty. The hyperparameters λ_1 and λ_2 for the scale- and smoothness-regularizers were determined based on 5-fold cross validation across 25 equally space grids in the interval $[10^{-2}, 10^{2}]$ for λ_1 and across the set $\{50, 100, 150\}$ for λ_2.

Affine Transfer The log-transformed value of $C_{\mathrm{p}}^{\mathrm{exp}}$ is modeled as

$$y := \log C_{\mathrm{P}}^{\mathrm{exp}} = \underbrace{\alpha_0 + \alpha_1 f_s}_{g_1} - \underbrace{(\beta f_s + 1)}_{g_2} \cdot \underbrace{x^{\top}\gamma}_{g_3} + \epsilon_{\sigma}, \tag{S.4}$$

where ϵ_σ represents observation noise, and $\alpha_0, \alpha_1, \beta$ and γ are unknown parameters to be estimated. When $\alpha_1 = 1$ and $\beta = 0$, Eq. (S.4) is consistent with the theoretical equation in Eq. (S.3) in which the quantum effect is linearly modeled as $\alpha_0 + x^{\top}\gamma$.

Table S.2: Mean and standard deviation of RMSE of three prediction models.

Model	RMSE (log J/kg·K)
$y = \alpha_0 + \alpha_1 f_s + \epsilon_\sigma$	0.1403 ± 0.0461
$y = f_s + x^\top \gamma + \epsilon_\sigma$	0.1368 ± 0.04265
$y = \alpha_0 + \alpha_1 f_s - (\beta f_s + 1) x^\top \gamma + \epsilon_\sigma$	**0.1357 ± 0.04173**

In the model training, the objective function was given as follows:

$$F_{\alpha,\beta,\gamma} = \frac{1}{n} \sum_{i=1}^{n} \left\{ y_i - (\alpha_0 + \alpha_1 f_{s,i} - (\beta f_{s,i} + 1) x^\top \gamma) \right\}^2$$

$$+ \lambda_\beta \beta^2 + \lambda_{\gamma,1} \|\gamma\|_2^2 + \lambda_{\gamma,2} \sum_{t \in T} \sum_{j=2}^{J_t} \left(\gamma_{t,j} - \gamma_{t,j-1} \right)^2,$$

where $\alpha = [\alpha_0 \ \alpha_1]^\top$. With a fixed $\lambda_\beta = 1$, the remaining hyperparameters $\lambda_{\gamma,1}$ and $\lambda_{\gamma,2}$ were optimized through 5-fold cross validation over 25 equally space grids in the interval $[10^{-2}, 10^2]$ for $\lambda_{\gamma,1}$ and across the set $\{50, 100, 150\}$ for $\lambda_{\gamma,2}$.

The algorithm to estimate the parameters α, β and γ is described in Algorithm S.1, where $\alpha_{0,\text{olr}}$ and $\alpha_{1,\text{olr}}$ are the estimated parameters of the ordinary linear regression model, and $\hat{\gamma}_{\text{diff}}$ is the estimated parameter of the log-difference model. For each step, the full conditional minimization of $F_{\alpha,\beta,\gamma}$ with respect to each parameter can be made analytically as

$$\begin{aligned}
\arg\min_{\alpha} F_{\alpha,\beta,\gamma} &= (F_s^\top F_s)^{-1} y_s^\top (y + (\beta f_{s,1:n} + 1) \circ (X\gamma)), \\
\arg\min_{\beta} F_{\alpha,\beta,\gamma} &= -(f_{s,1:n}^\top \text{diag}(X\gamma)^2 f_{s,1:n} + n\lambda_2)^{-1} f_{s,1:n}^\top \text{diag}(X\gamma)(y - F_s\alpha + X\gamma), \\
\arg\min_{\gamma} F_{\alpha,\beta,\gamma} &= -(X^\top \text{diag}(\beta f_{s,1:n} + 1)^2 X + \Lambda)^{-1} X^\top \text{diag}(\beta f_{s,1:n} + 1)(y - F_s\alpha),
\end{aligned}$$

where X denote the matrix in which the i-th row is x_i, $y = [y_1 \cdots y_n]^\top$, $f_{s,1:n} = [f_{s,1} \cdots f_{s,n}]^\top$, $F_s = [f_{s,1:n} \ \mathbf{1}]$, and $d = 190$. Λ is a matrix including the two regularization parameters $\lambda_{\gamma,1}$ and $\lambda_{\gamma,2}$ as

$$\Lambda = D^\top D, \text{ where } D = \begin{bmatrix} \lambda_{\gamma,1} I_d \\ \lambda_{\gamma,2} M \end{bmatrix}, \ M = \begin{bmatrix} -1 & 1 & 0 & \cdots & 0 & 0 \\ 0 & -1 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -1 & 1 \end{bmatrix} \begin{array}{l} \\ \\ \\ \leftarrow m\text{-th rows} \\ \\ \\ \end{array},$$

where $m \in \{10, 30, 50, 70, 90, 110, 130, 150, 170\}$. Note that the matrix M is the same as the matrix $[\mathbf{0} \ I_{189}] - [I_{189} \ \mathbf{0}]$ except that the m-th row is all zeros. Note also that $M \in \mathbb{R}^{189 \times 190}$, and therefore $D \in \mathbb{R}^{279 \times 190}$ and $\Lambda \in \mathbb{R}^{190 \times 190}$.

The stopping criterion of the algorithm was set as

$$\max_{\theta \in \{a,b,c\}} \frac{\max_i \left| \theta_i^{(\text{new})} - \theta_i^{(\text{old})} \right|}{\max_i \left| \theta_i^{(\text{old})} \right|} < 10^{-4}, \tag{S.5}$$

where θ_i denotes the i-th element of the parameter θ. This convergence criterion is employed in several existing machine learning libraries, e.g., scikit-learn [3].

B.3.3 Results

Table S.2 summarizes the prediction performance (RMSE) of the three models. The ordinary linear model $y = \alpha_0 + \alpha_1 f_s + \epsilon_\sigma$, which ignores the force field descriptors, exhibited the lowest prediction

---

[3] `https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Lasso.html`

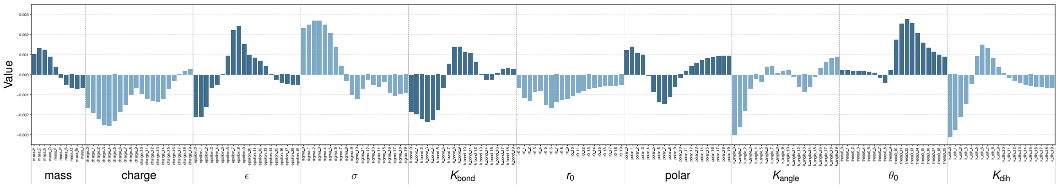


Figure S.5: Bar plot of regression coefficients γ of linear calibrator filling the discrepancy between experimental and MD-calculated specific heat capacity of amorphous polymers.

performance. The other two calibration models $y = f_s + x^\top \gamma + \epsilon_\sigma$ and the full model in Eq. (S.4) reached almost the same accuracy, but the latter had achieved slightly better prediction accuracy. The estimated parameters of the full model were $\alpha_1 \approx 0.889$ and $\beta \approx -0.004$. The model form is highly consistent with the theoretical equation in Eq. (S.3) as well as the restricted model ($\alpha_1 = 1, \beta = 0$). This supports the validity of the theoretical model in [9].

It is expected that physicochemical insights can be obtained by examining the estimated coefficient γ, which would capture the contribution of the force field parameters to the quantum effects. The magnitude of the quantum effect is a monotonically increasing function of the frequency ω, and is known to be highly related to the descriptors ϵ, K_{dih}, K_{bond}, K_{angle} and mass. According to physicochemical intuition, it is considered that as ϵ, K_{bond}, K_{angle}, and K_{dih} decrease, their potential energy surface becomes shallow, which leads to the decrease of ω, and in turn the decrease of quantum effects. Furthermore, because the molecular vibration of light-weight atoms is faster than that of heavy atoms, ω and quantum effects should theoretically increase with decreasing mass.

Figure S.5 shows the mean values of the estimated parameter γ for the full calibration model. The physical relationships described above can be captured consistently with the estimated coefficients. The coefficients in lower regions of ϵ, K_{bond}, K_{angle} and K_{dih} showed large negative values, indicating that polymers containing more atoms, bonds, angles, and dihedral angles with lower values will have smaller quantum effects. Conversely, the coefficients in lower regions of mass showed positive large values, meaning that polymers containing more atoms with smaller masses will have larger quantum effects. As illustrated in this example, separate inclusion of the domain-common and domain-specific factors in the affine transfer model enables us to infer the features relevant to the cross-domain differences.

C Experimental Details

Instructions for obtaining the datasets used in the experiments are described in the code.

C.1 Kinematics of the Robot Arm

C.1.1 Data

We used the SARCOS dataset in [11]. The task is to predict the feed-forward torque required to follow the desired trajectory in the seven joints of the SARCOS anthropomorphic robot arm. The twenty one features representing the joints' position, velocity, and acceleration were used as x. The observed values of six torques other than the torque at the joint in the target domain were given to the source features f_s. The dataset includes 44,484 training samples and 4,449 test samples. We selected $\{5, 10, 15, 20, 30, 40, 50\}$ samples randomly from the training set. The prediction performances of the trained models were evaluated using the 4,449 test samples. Repeated experiments were conducted 20 times with different independently sampled datasets.

C.1.2 Model Definition and Hyperparameter Search

Source model For each target task, a multi-task neural network was trained to predict the torque values of the remaining six source tasks. The source model shares four layers (256-128-64-32) up to the final layer, and only the output layer is task-specific. We used all training data and Adagrad [12] with learning rate of 0.01.

Algorithm S.2 Block relaxation algorithm for **AffineTL-full**.

```
Initialize: $a_0 \leftarrow (K_1 + \lambda_1 I_n)^{-1} y$, $b_0 \sim \mathcal{N}(\mathbf{0}, I_n)$, $c_0 \sim \mathcal{N}(\mathbf{0}, I_n)$, $d_0 \leftarrow 0.5$
repeat
  $a \leftarrow (K_1 + \lambda_1 I_n)^{-1}(y - (K_2 b + \mathbf{1}) \circ (K_3 c) - d\mathbf{1})$
  $b \leftarrow (\mathrm{Diag}(K_3 c)^2 K_2 + \lambda_2 I_n)^{-1}((K_3 c) \circ (y - K_1 a - K_3 c - d\mathbf{1}))$
  $c \leftarrow (\mathrm{Diag}(K_2 b + \mathbf{1})^2 K_3 + \lambda_3 I_n)^{-1}((K_2 b + \mathbf{1}) \circ (y - K_1 a - d\mathbf{1}))$
  $d \leftarrow \langle y - K_1 a - (K_2 b + \mathbf{1}) \circ (K_3 c), \mathbf{1} \rangle / n$
until convergence
```

Direct, Only source, Augmented, HTL-offset, HTL-scale For each procedure, we used kernel ridge regression with the RBF kernel $k(x, x') = \exp(-\frac{1}{2\ell^2}\|x - x'\|_2^2)$. The scale parameter ℓ was set to the square root of the input dimension as $\ell = \sqrt{21}$ for **Direct**, **HTL-offset** and **HTL-scale**, $\ell = \sqrt{6}$ for **Only source** and $\ell = \sqrt{27}$ for **Augmented**. The regularization parameter λ was selected in 5-fold cross-validation in which the grid search was performed over 50 grid points in the interval $[10^{-4}, 10^2]$.

AffineTL-full, AffineTL-const We considered the following kernels:

$$k_1(f_s(x), f_s(x')) = \exp\Big(-\frac{1}{2\ell^2}\|f_s(x) - f_s(x')\|_2^2\Big) \ (\ell = \sqrt{6}),$$
$$k_2(f_s(x), f_s(x')) = \exp\Big(-\frac{1}{2\ell^2}\|f_s(x) - f_s(x')\|_2^2\Big) \ (\ell = \sqrt{6}),$$
$$k_3(x, x') = \exp\Big(-\frac{1}{2\ell^2}\|x - x'\|_2^2\Big) \ (\ell = \sqrt{27}),$$

for g_1, g_2 and g_3 in the affine transfer model, respectively.

Hyperparameters to be optimized are the three regularization parameters λ_1, λ_2 and λ_3. We performed 5-fold cross-validation to identify the best hyperparameter set from the candidate points; $\{10^{-3}, 10^{-2}, 10^{-1}, 1\}$ for λ_1 and $\{10^{-2}, 10^{-1}, 1, 10\}$ for each of λ_2 and λ_3.

To learn the **AffineTL-full** and **AffineTL-const**, we used the following objective functions:

AffineTL-full $\|y - (K_1 a + (K_2 b + \mathbf{1}) \circ (K_3 c) + d)\|_2^2 + \lambda_1 a^\top K_1 a + \lambda_2 b^\top K_2 b + \lambda_3 c^\top K_3 c,$

AffineTL-const $\frac{1}{n}\|y - (K_1 a + K_3 c + d)\|_2^2 + \lambda_1 a^\top K_1 a + \lambda_3 c^\top K_3 c.$

Algorithm S.2 summarizes the block relaxation algorithm for **AffineTL-full**. For **AffineTL-const**, we found the optimal parameters as follows:

$$\begin{bmatrix} \hat{a} \\ \hat{c} \\ \hat{d} \end{bmatrix} = \left(\begin{bmatrix} K_1 \\ K_3 \\ \mathbf{1}^\top \end{bmatrix} \begin{bmatrix} K_1 & K_3 & \mathbf{1} \end{bmatrix} + \begin{bmatrix} \lambda_1 K_1 & & \\ & \lambda_3 K_3 & \\ & & 0 \end{bmatrix} \right)^{-1} \begin{bmatrix} K_1 \\ K_3 \\ \mathbf{1}^\top \end{bmatrix} y$$

The stopping criterion for the algorithm was the same as Eq. (S.5).

Fine-tuning The target network was constructed by adding a one-dimensional output layer to the shared layers of the source network. As initial values for the training, we used the weights of the source neural network for the shared layer and the average of the multidimensional output layer of the source network for the output layer. Adagrad [12] was used for the optimization. The learning rate was fixed at 0.01 and the number of training epochs was selected from $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 20, 50, 100\}$ through 5-fold cross-validation.

MAML A fully connected neural network with 256-64-32-16-1 layer width was used, and the initial values were searched through MAML [13] using the six source tasks. The obtained base model was fine-tuned with the target samples. Adam [14] with a fixed learning rate of 0.01 was used for the optimization. The number of training epochs was selected from $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 20, 50, 100\}$ through 5-fold cross-validation.

L^2-SP L^2-SP is a regularization method proposed by [15] in which the following regularization term is added so that the weights of the target network are estimated in the neighborhood of the weights of the source network:

$$\Omega(w) = \frac{\alpha}{2}\|w - w_s\|_2^2, \tag{S.6}$$

where w and w_s are the weights of the target and source model, respectively, and $\alpha > 0$ is a hyperparameter. We used the weights of the source network as the initial point for the training of the target model, and added a regularization parameter as in (S.6). Adagrad [12] were used for the optimizer, and the regularization parameters and learning rate were fixed at 0.01 and 0.001, respectively. The number of training epochs was selected from $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 20, 50, 100\}$ through 5-fold cross-validation.

PAC-Net PAC-Net, proposed in [16], is a TL method that leverages pruning of the weights of the source network. Its training strategy consists of three steps: identifying the important weights in the source model, fine-tuning them using the source samples, and updating the remaining weights using the target samples.

Firstly, we pruned the bottom 10% of weights, based on absolute value, from the pre-trained source network. Following this, the remaining weights were retrained using the stochastic gradient descent (SGD). Finally, the pruned weights were retrained using target samples. For the final training phase, SGD with learning rate 0.01, was employed, and the number of training epochs was selected from $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 20, 50, 100\}$ through 5-fold cross-validation.

Table S.3: Performance on predicting the torque values at the first and seventh joints of the SARCOS robot arm. Definitions of an asterisk and bold type are the same as in Table 1 in the main paper.

Target	Model	Number of training samples						
		$n < d$			$n \approx d$	$n > d$		
		5	10	15	20	30	40	50
Torque 1	Direct	21.3 ± 2.04	18.9 ± 2.11	**17.4 ± 1.79**	15.8 ± 1.70	13.7 ± 1.26	12.2 ± 1.61	10.8 ± 1.23
	Only source	24.0 ± 6.37	22.3 ± 3.10	21.0 ± 2.49	19.7 ± 1.34	18.5 ± 1.92	17.6 ± 1.59	17.3 ± 1.31
	Augmented	21.8 ± 2.88	19.2 ± 1.37	17.8 ± 2.30	**15.7 ± 1.53**	**13.3 ± 1.19**	**11.9 ± 1.37**	**10.7 ± 0.954**
	HTL-offset	23.7 ± 6.50	21.2 ± 3.85	19.8 ± 3.23	17.8 ± 2.35	16.2 ± 3.31	15.0 ± 3.16	15.1 ± 2.76
	HTL-scale	23.3 ± 4.47	22.1 ± 5.31	20.4 ± 3.84	18.5 ± 2.72	17.6 ± 2.41	16.9 ± 2.10	16.7 ± 1.74
	AffineTL-full	**21.2 ± 2.23**	**18.8 ± 1.31**	18.6 ± 2.83	15.9 ± 1.65	13.7 ± 1.53	12.3 ± 1.45	11.1 ± 1.12
	AffineTL-const	21.2 ± 2.21	18.8 ± 1.44	17.7 ± 2.44	15.9 ± 1.58	13.4 ± 1.15	12.2 ± 1.54	10.9 ± 1.02
	Fine-tune	25.0 ± 7.11	20.5 ± 3.33	18.6 ± 2.10	17.6 ± 2.55	14.1 ± 1.39	12.6 ± 1.13	11.1 ± 1.03
	MAML	29.8 ± 12.3	22.5 ± 3.21	20.8 ± 2.12	20.3 ± 3.14	16.7 ± 3.00	14.4 ± 1.85	13.4 ± 1.19
	L^2-SP	24.9 ± 7.09	20.5 ± 3.30	18.8 ± 2.04	18.0 ± 2.45	14.5 ± 1.36	13.0 ± 1.13	11.6 ± 0.983
	PAC-Net	25.2 ± 8.68	22.7 ± 5.60	20.7 ± 2.65	20.1 ± 2.16	18.5 ± 2.77	17.6 ± 1.85	17.1 ± 1.38
Torque 2	Direct	15.8 ± 2.37	13.0 ± 1.41	11.5 ± 0.985	10.4 ± 0.845	9.20 ± 0.827	8.35 ± 0.802	7.78 ± 0.780
	Only source	14.9 ± 1.77	13.6 ± 2.51	12.3 ± 1.77	11.2 ± 1.16	10.6 ± 1.22	9.74 ± 0.920	9.06 ± 0.785
	Augmented	15.2 ± 1.95	**12.3 ± 0.923**	**11.4 ± 1.48**	**10.2 ± 0.813**	**9.07 ± 0.983**	**8.06 ± 0.862**	**7.23 ± 0.629**
	HTL-offset	14.8 ± 1.71	13.4 ± 2.41	12.2 ± 1.81	10.9 ± 1.29	10.4 ± 1.37	9.32 ± 1.11	8.78 ± 0.829
	HTL-scale	14.8 ± 1.71	13.4 ± 2.47	12.2 ± 1.82	11.0 ± 1.32	10.5 ± 1.28	9.39 ± 1.01	8.91 ± 0.946
	AffineTL-full	14.7 ± 1.83	13.0 ± 1.34	11.9 ± 1.22	11.3 ± 1.39	9.38 ± 0.842	8.25 ± 0.932	7.34 ± 0.605
	AffineTL-const	**14.6 ± 1.47**	12.6 ± 1.09	11.5 ± 0.807	10.5 ± 1.19	9.28 ± 0.828	8.35 ± 1.06	7.33 ± 0.57
	Fine-tune	24.4 ± 5.87	15.0 ± 2.01	13.6 ± 2.31	11.9 ± 1.21	10.7 ± 0.897	9.52 ± 0.774	8.43 ± 0.907
	MAML	21.8 ± 7.33	14.8 ± 4.51	13.1 ± 2.69	11.5 ± 2.24	9.77 ± 1.24	8.90 ± 1.10	7.89 ± 0.713
	L^2-SP	24.4 ± 5.87	15.1 ± 2.02	13.6 ± 2.29	12.0 ± 1.22	10.8 ± 0.886	9.70 ± 0.78	8.68 ± 0.868
	PAC-Net	24.0 ± 6.94	16.7 ± 4.14	13.7 ± 2.36	13.2 ± 2.49	12.4 ± 2.05	11.6 ± 0.844	11.2 ± 0.706
Torque 3	Direct	9.91 ± 1.65	8.15 ± 1.01	7.39 ± 1.21	6.84 ± 0.878	5.90 ± 0.850	5.26 ± 0.774	4.66 ± 0.523
	Only source	9.00 ± 1.44	7.51 ± 1.05	6.90 ± 1.15	6.51 ± 0.930	5.67 ± 0.890	5.29 ± 0.840	4.89 ± 0.604
	Augmented	9.47 ± 1.35	7.72 ± 1.05	6.99 ± 1.25	6.29 ± 0.967	5.42 ± 0.938	**4.76 ± 0.826**	**4.32 ± 0.592**
	HTL-offset	**8.96 ± 1.42**	7.47 ± 1.06	6.88 ± 1.15	6.39 ± 0.952	5.58 ± 0.856	5.18 ± 0.821	4.83 ± 0.603
	HTL-scale	9.05 ± 1.40	7.49 ± 1.08	6.89 ± 1.18	6.63 ± 1.03	5.60 ± 0.955	5.21 ± 0.836	4.86 ± 0.503
	AffineTL-full	9.24 ± 1.46	**7.45 ± 1.25**	6.85 ± 1.23	6.28 ± 0.930	5.54 ± 1.15	4.89 ± 0.907	4.46 ± 0.733
	AffineTL-const	9.08 ± 1.21	7.55 ± 0.974	**6.67 ± 1.00**	**6.17 ± 0.916**	**5.42 ± 0.971**	4.85 ± 0.752	4.42 ± 0.614
	Fine-tune	9.00 ± 2.14	7.38 ± 1.09	6.72 ± 1.01	*5.91 ± 0.734	*5.26 ± 0.541	4.86 ± 0.488	4.41 ± 0.325
	MAML	9.50 ± 4.94	*7.11 ± 0.966	*6.44 ± 1.01	*5.92 ± 0.793	*5.22 ± 0.626	4.87 ± 0.539	4.79 ± 0.525
	L^2-SP	9.00 ± 2.14	7.39 ± 1.08	6.73 ± 1.02	*5.91 ± 0.73	5.39 ± 0.633	4.89 ± 0.493	4.46 ± 0.319
	PAC-Net	9.14 ± 2.11	*7.31 ± 1.03	*6.33 ± 0.841	*5.96 ± 0.926	5.34 ± 0.633	5.17 ± 0.474	5.05 ± 0.371
Torque 4	Direct	14.2 ± 2.30	11.1 ± 2.28	9.49 ± 2.19	7.78 ± 1.02	6.86 ± 0.768	6.13 ± 0.714	5.48 ± 0.592
	Only source	13.1 ± 3.36	9.62 ± 2.05	8.38 ± 2.06	7.06 ± 1.32	6.36 ± 1.24	5.79 ± 0.768	5.37 ± 0.897
	Augmented	13.5 ± 2.83	9.69 ± 1.89	8.51 ± 1.84	*6.96 ± 1.03	*6.09 ± 0.931	***5.39 ± 0.685**	***4.87 ± 0.618**
	HTL-offset	**13 ± 3.38**	9.62 ± 2.05	8.34 ± 2.00	7.02 ± 1.24	6.26 ± 1.17	5.76 ± 0.764	5.36 ± 0.897
	HTL-scale	13.0 ± 3.35	9.63 ± 2.07	**8.30 ± 1.95**	7.01 ± 1.16	6.30 ± 1.17	5.77 ± 0.758	5.37 ± 0.902
	AffineTL-full	13.0 ± 2.69	9.48 ± 2.10	8.38 ± 1.85	7.14 ± 1.62	*5.91 ± 0.838	*5.45 ± 0.777	*4.94 ± 0.603
	AffineTL-const	13.2 ± 3.16	***9.32 ± 1.99**	8.39 ± 1.84	***6.88 ± 1.00**	***5.85 ± 0.710**	*5.55 ± 0.679	*4.94 ± 0.581
	Fine-tune	*11.7 ± 2.70	*8.24 ± 1.31	*6.71 ± 1.02	*5.90 ± 0.971	*5.17 ± 0.785	*4.59 ± 0.442	*4.21 ± 0.376
	MAML	14.3 ± 7.75	10.9 ± 3.44	9.55 ± 1.99	9.41 ± 2.33	7.98 ± 2.36	6.70 ± 1.25	6.18 ± 1.35
	L^2-SP	*11.7 ± 2.70	*8.24 ± 1.31	*6.73 ± 1.01	*5.92 ± 0.959	*5.22 ± 0.765	*4.67 ± 0.45	*4.28 ± 0.363
	PAC-Net	11.2 ± 5.24	*8.84 ± 2.75	*7.64 ± 1.17	7.34 ± 1.56	6.77 ± 0.966	6.29 ± 0.536	6.02 ± 0.446
Torque 5	Direct	1.07 ± 0.157	0.993 ± 0.0903	0.910 ± 0.119	**0.847 ± 0.129**	**0.744 ± 0.113**	**0.686 ± 0.0996**	**0.623 ± 0.0944**
	Only source	1.15 ± 0.214	1.04 ± 0.0775	0.998 ± 0.145	0.975 ± 0.133	0.863 ± 0.111	0.826 ± 0.155	0.775 ± 0.106
	Augmented	**1.04 ± 0.113**	0.987 ± 0.109	0.907 ± 0.120	0.874 ± 0.136	0.755 ± 0.130	0.710 ± 0.110	0.637 ± 0.0893
	HTL-offset	1.14 ± 0.221	1.02 ± 0.0864	0.965 ± 0.157	0.925 ± 0.141	0.837 ± 0.104	0.800 ± 0.156	0.738 ± 0.101
	HTL-scale	1.13 ± 0.194	1.01 ± 0.0786	0.980 ± 0.177	0.914 ± 0.132	0.830 ± 0.114	0.844 ± 0.171	0.785 ± 0.123
	AffineTL-full	1.04 ± 0.121	0.989 ± 0.175	0.907 ± 0.162	0.860 ± 0.170	0.747 ± 0.117	0.691 ± 0.0924	0.654 ± 0.0716
	AffineTL-const	1.05 ± 0.106	**0.974 ± 0.102**	**0.899 ± 0.123**	0.854 ± 0.121	0.756 ± 0.106	0.700 ± 0.0869	0.638 ± 0.0796
	Fine-tune	1.22 ± 0.356	1.04 ± 0.105	0.976 ± 0.0878	0.913 ± 0.137	0.749 ± 0.111	0.688 ± 0.103	0.598 ± 0.0697
	MAML	1.45 ± 0.479	1.18 ± 0.183	1.07 ± 0.208	0.999 ± 0.193	0.816 ± 0.211	0.703 ± 0.124	0.613 ± 0.0634
	L^2-SP	1.22 ± 0.355	1.04 ± 0.105	0.973 ± 0.0873	0.917 ± 0.133	0.756 ± 0.109	0.699 ± 0.113	0.606 ± 0.0644
	PAC-Net	1.27 ± 0.319	1.10 ± 0.115	1.03 ± 0.108	0.985 ± 0.145	0.881 ± 0.151	0.806 ± 0.118	0.781 ± 0.124
Torque 6	Direct	1.86 ± 0.248	1.67 ± 0.192	**1.50 ± 0.162**	**1.36 ± 0.159**	**1.22 ± 0.163**	1.12 ± 0.102	1.05 ± 0.0916
	Only source	1.91 ± 0.230	1.87 ± 0.357	1.76 ± 0.179	1.64 ± 0.190	1.53 ± 0.255	1.36 ± 0.153	1.26 ± 0.0883
	Augmented	1.86 ± 0.180	**1.66 ± 0.17**	1.55 ± 0.219	1.45 ± 0.231	1.27 ± 0.265	**1.11 ± 0.130**	**1.01 ± 0.0944**
	HTL-offset	1.88 ± 0.214	1.81 ± 0.369	1.67 ± 0.246	1.54 ± 0.239	1.46 ± 0.267	1.33 ± 0.142	1.21 ± 0.133
	HTL-scale	2.05 ± 0.649	1.88 ± 0.389	1.71 ± 0.241	1.60 ± 0.308	1.62 ± 0.474	1.37 ± 0.158	1.25 ± 0.0922
	AffineTL-full	**1.82 ± 0.232**	1.74 ± 0.209	1.55 ± 0.242	1.43 ± 0.231	1.27 ± 0.230	1.13 ± 0.122	1.06 ± 0.191
	AffineTL-const	1.83 ± 0.173	1.68 ± 0.207	1.55 ± 0.236	1.40 ± 0.227	1.23 ± 0.198	1.12 ± 0.113	1.02 ± 0.0953
	Fine-tune	2.41 ± 0.375	2.03 ± 0.387	1.71 ± 0.463	1.49 ± 0.297	1.27 ± 0.257	1.15 ± 0.110	1.07 ± 0.0817
	MAML	2.69 ± 0.676	2.17 ± 0.511	1.96 ± 0.526	1.68 ± 0.373	1.42 ± 0.365	1.23 ± 0.111	1.16 ± 0.0829
	L^2-SP	2.41 ± 0.375	2.03 ± 0.371	1.72 ± 0.455	1.49 ± 0.299	1.28 ± 0.254	1.16 ± 0.108	1.09 ± 0.0777
	PAC-Net	2.47 ± 0.385	2.22 ± 0.550	2.22 ± 0.599	1.99 ± 0.372	1.91 ± 0.355	1.74 ± 0.144	1.69 ± 0.0595
Torque 7	Direct	2.66 ± 0.307	2.13 ± 0.420	1.85 ± 0.418	1.54 ± 0.353	1.32 ± 0.200	1.18 ± 0.138	1.05 ± 0.111
	Only source	2.31 ± 0.618	*1.73 ± 0.560	*1.49 ± 0.513	*1.22 ± 0.269	*1.09 ± 0.232	*0.969 ± 0.144	*0.927 ± 0.170
	Augmented	2.47 ± 0.406	1.90 ± 0.515	1.67 ± 0.552	*1.31 ± 0.214	1.16 ± 0.225	*0.984 ± 0.149	*0.897 ± 0.138
	HTL-offset	2.29 ± 0.621	***1.69 ± 0.507**	*1.49 ± 0.513	*1.22 ± 0.269	*1.09 ± 0.233	*0.969 ± 0.144	*0.925 ± 0.171
	HTL-scale	2.32 ± 0.599	*1.71 ± 0.516	1.51 ± 0.513	*1.24 ± 0.271	*1.12 ± 0.234	*0.999 ± 0.175	0.948 ± 0.172
	AffineTL-full	***2.23 ± 0.554**	*1.71 ± 0.501	***1.45 ± 0.458**	*1.21 ± 0.256	*1.06 ± 0.219	*0.974 ± 0.164	***0.870 ± 0.121**
	AffineTL-const	*2.30 ± 0.565	*1.73 ± 0.420	*1.48 ± 0.527	***1.20 ± 0.243**	***1.04 ± 0.217**	***0.963 ± 0.161**	*0.884 ± 0.136
	Fine-tune	*2.33 ± 0.511	*1.62 ± 0.347	*1.35 ± 0.340	*1.12 ± 0.165	*0.959 ± 0.12	*0.848 ± 0.0824	*0.790 ± 0.0547
	MAML	2.54 ± 1.29	1.90 ± 0.507	1.67 ± 0.313	1.63 ± 0.282	1.28 ± 0.272	1.20 ± 0.199	1.06 ± 0.111
	L^2-SP	*2.33 ± 0.509	*1.65 ± 0.378	*1.35 ± 0.340	*1.12 ± 0.165	*0.968 ± 0.114	*0.858 ± 0.0818	*0.802 ± 0.0535
	PAC-Net	2.24 ± 0.706	*1.61 ± 0.394	*1.43 ± 0.389	*1.24 ± 0.177	*1.18 ± 0.100	1.13 ± 0.0726	1.100 ± 0.0589

C.2 Evaluation of Scientific Papers

C.2.1 Data

We used SciRepEval benchmark dataset for scientific documents, proposed in [11]. This dataset comprises abstracts, review scores, and decision statuses of papers submitted to various machine learning conferences. We conducted two primary tasks: predicting the average review score (a regression task) and determining the acceptance or rejection status of each paper (a binary classification task). The original input x, was represented as a two-gram bag-of-words vector derived from the abstract. As for source features f_s, we employed text embeddings from the abstracts, which were generated by various pre-trained language models, including BERT [17], SciBERT [18], T5 [19], and GPT-3 [20]. When building the vocabulary for the bag-of-words, we ignore phrases with document frequencies strictly higher than 0.9 or strictly lower than 0.01. Additionally, we eliminated certain stop-words using the default settings in scikit-learn [21]. The sentences containing URLs were removed from the abstracts because accepted papers tend to include GitHub links in their abstracts after acceptance, which may cause leakage of information to the prediction. The models were trained on a dataset comprising 8,166 instances, and their performance were subsequently evaluated on a test dataset of 2,043 instances.

C.2.2 Model Definition and Hyperparamter Search

For both the affine model transfer and feature extraction, we employed neural networks with ReLU activation and dropout layer with 0.1 dropout rate. The parameters were optimized using Adagrad [12] with 0.01 learning rate.

Affine Model Transfer For functions g_1 and g_2 in the affine model transfer, we used a neural network composed of layers with widths 128, 64, 32, and 16, wherein the source features f_s were used as inputs. The number of layers of each width was determined based on Bayesian optimization. Sigmoid activation was employed to the output of g_2 in order to facilitate the interpretation of g_3. For g_3, we employed a linear model with the input x. To prevent overfitting and promote model simplicity, we applied ℓ_1 regularization to the parameters of g_3 with a regularization parameter of 0.01. The final output of the model was computed as $g_1 + g_2 \cdot g_3$. In the case of the binary classification task, we applied sigmoid activation function to this final output.

Feature Extraction As in the affine model transfer, we used a neural network composed of layers with widths 128, 64, 32, and 16, wherein the source features f_s were used as inputs. The number of layers of each width was determined based on Bayesian optimization. In the case of the binary classification task, we applied sigmoid activation function to the final output.

D Proofs

D.1 Proof of Theorem 2.4

Proof. According to Assumption 2.3, it holds that for any $p_t(y|x)$,

$$\psi_{f_s}\left(\int \phi_{f_s}(y)p_t(y|x)\mathrm{d}y\right) = \int y p_t(y|x)\mathrm{d}y. \tag{S.7}$$

(i) Let δ_{y_0} be the Dirac delta function supported on y_0. Substituting $p_t(y|x) = \delta_{y_0}$ into Eq. (S.7), we have

$$\psi_{f_s}(\phi_{f_s}(y_0)) = y_0 \ (\forall y_0 \in \mathcal{Y}).$$

Under Assumption 2.2, this implies the property (i).

(ii) For simplicity, we assume the inputs x are fixed and $p_t(y|x) > 0$. Applying the property (i) to Eq. (S.7) yields

$$\int \phi_{f_s}(y)p_t(y|x)\mathrm{d}y = \phi_{f_s}\left(\int y p_t(y|x)\mathrm{d}y\right).$$

We consider a two-component mixture $p_t(y|x) = (1-\epsilon)q(y|x) + \epsilon h(y|x)$ with a mixing rate $\epsilon \in [0,1]$, where q and h denote arbitrary probability density functions. Then, we have

$$\int \phi_{f_s}(y)\big\{(1-\epsilon)q(y|x) + \epsilon h(y|x)\big\}\mathrm{d}y = \phi_{f_s}\bigg(\int y\big\{(1-\epsilon)q(y|x) + \epsilon h(y|x)\big\}\mathrm{d}y\bigg).$$

Taking the derivative at $\epsilon = 0$, we have

$$\int \phi_{f_s}(y)\big\{h(y|x) - q(y|x)\big\}\mathrm{d}y = \phi'_{f_s}\bigg(\int yq(y|x)\mathrm{d}y\bigg)\bigg(\int y\big\{h(y|x) - q(y|x)\big\}\mathrm{d}y\bigg),$$

which yields

$$\int \big\{h(y|x) - q(y|x)\big\}\big\{\phi_{f_s}(y) - \phi'_{f_s}\big(\mathbb{E}_q[Y|X]\big)y\big\}\mathrm{d}y = 0. \tag{S.8}$$

For Eq. (S.8) to hold for any q and h, $\phi_{f_s}(y) - \phi'_{f_s}\big(\mathbb{E}_q[Y|X=x]\big)y$ must be independent of y. Thus, the function ϕ_{f_s} and its inverse $\psi_{f_s} = \phi_{f_s}^{-1}$ are limited to affine transformations with respect to y. Since ϕ depends on y and $f_s(x)$, it takes the form $\phi(y, f_s(x)) = g_1(f_s(x)) + g_2(f_s(x))y$. ☐

D.2 Proof of Theorem 4.1

To bound the generalization error, we use the empirical and population Rademacher complexity $\hat{\mathfrak{R}}_S(\mathcal{F})$ and $\mathfrak{R}(\mathcal{F})$ of hypothesis class $\mathcal{F}$, defined as:

$$\hat{\mathfrak{R}}_S(\mathcal{F}) = \mathbb{E}_\sigma \sup_{f\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^n \sigma_i f(x_i), \qquad \mathfrak{R}(\mathcal{F}) = \mathbb{E}_S\hat{\mathfrak{R}}_S(\mathcal{F}),$$

where $\{\sigma_i\}_{i=1}^n$ is a set of Rademacher variables that are independently distributed and each take one of the values in $\{-1,+1\}$ with equal probability, and S denotes a set of samples. The following proof is based on the one of Theorem 11 shown in [22].

Proof of Theorem 4.1. For any hypothesis class $\mathcal{F}$ with feature map Φ where $\|\Phi\|^2 \le 1$, the following inequality holds:

$$\mathbb{E}_\sigma \sup_{\|\theta\|^2\le\Lambda} \frac{1}{n}\sum_{i=1}^n \sigma_i\langle\theta, \Phi(x_i)\rangle \le \sqrt{\frac{\Lambda}{n}}.$$

The proof is given, for example, in Theorem 6.12 of [23]. Thus, the empirical Rademacher complexity of $\tilde{\mathcal{H}}$ is bounded as

$$\begin{aligned}
\hat{\mathfrak{R}}_S(\tilde{\mathcal{H}}) &= \mathbb{E}_\sigma \sup_{\substack{\|\alpha\|^2_{\mathcal{H}_1} \le \lambda_\alpha^{-1}\hat{R}_s,\\ \|\beta\|^2_{\mathcal{H}_2} \le \lambda_\beta^{-1}\hat{R}_s,\\ \|\gamma\|^2_{\mathcal{H}_3} \le \lambda_\gamma^{-1}\hat{R}_s}} \frac{1}{n}\sum_{i=1}^n \sigma_i\Big\{\langle\alpha, \Phi_1(f_s(x_i))\rangle_{\mathcal{H}_1} + \langle\beta, \Phi_2(f_s(x_i))\rangle_{\mathcal{H}_2}\langle\gamma, \Phi(x_i)\rangle_{\mathcal{H}_3}\Big\} \\
&\le \mathbb{E}_\sigma \sup_{\|\alpha\|^2_{\mathcal{H}_1}\le\lambda_\alpha^{-1}\hat{R}_s} \frac{1}{n}\sum_{i=1}^n \sigma_i\langle\alpha, \Phi_1(f_s(x_i))\rangle_{\mathcal{H}_1} \\
&\qquad\qquad + \sup_{\substack{\|\beta\|^2_{\mathcal{H}_2} \le \lambda_\beta^{-1}\hat{R}_s,\\ \|\gamma\|^2_{\mathcal{H}_3} \le \lambda_\gamma^{-1}\hat{R}_s}} \frac{1}{n}\sum_{i=1}^n \sigma_i\langle\beta\otimes\gamma, \Phi_2(f_s(x_i))\otimes\Phi(x_i)\rangle_{\mathcal{H}_2\otimes\mathcal{H}_3} \\
&\le \mathbb{E}_\sigma \sup_{\|\alpha\|^2_{\mathcal{H}_1}\le\lambda_\alpha^{-1}\hat{R}_s} \frac{1}{n}\sum_{i=1}^n \sigma_i\langle\alpha, \Phi_1(f_s(x_i))\rangle_{\mathcal{H}_1} \\
&\qquad\qquad + \sup_{\|\beta\otimes\gamma\|^2_{\mathcal{H}_2\otimes\mathcal{H}_3}\le\lambda_\beta^{-1}\lambda_\gamma^{-1}\hat{R}_s^2} \frac{1}{n}\sum_{i=1}^n \sigma_i\langle\beta\otimes\gamma, \Phi_2(f_s(x_i))\otimes\Phi(x_i)\rangle_{\mathcal{H}_2\otimes\mathcal{H}_3} \\
&\le \sqrt{\frac{\hat{R}_s}{\lambda_\alpha n}} + \sqrt{\frac{\hat{R}_s^2}{\lambda_\beta\lambda_\gamma n}} \qquad\qquad (\mathrm{S}.9) \\
&\le \sqrt{\frac{\hat{R}_s}{n}}\bigg\{\sqrt{\frac{1}{\lambda_\alpha}} + \sqrt{\frac{L}{\lambda_\beta\lambda_\gamma}}\bigg\}.
\end{aligned}$$

The first inequality follows from the subadditivity of supremum. The last inequality follows from the fact that $\hat{R}_s \leq P_n\ell(y, \langle 0, \Phi_1\rangle) + \lambda_\alpha\|0\|^2 \leq L$.

Let $C = \sqrt{\frac{1}{\lambda_\alpha}} + \sqrt{\frac{L}{\lambda_\beta\lambda_\gamma}}$, and applying Talagrand's lemma [23] and Jensen's inequality, we obtain

$$\mathfrak{R}(\mathcal{L}) = \mathbb{E}\hat{\mathfrak{R}}_S(\mathcal{L}) \leq \mu_\ell \mathbb{E}\hat{\mathfrak{R}}_S(\tilde{\mathcal{H}}) \leq C\mu_\ell \mathbb{E}\sqrt{\frac{\hat{R}_s}{n}} \leq C\mu_\ell\sqrt{\frac{\mathbb{E}\hat{R}_s}{n}}.$$

To apply Corollary 3.5 of [24], we should solve the equation

$$r = C\mu_\ell\sqrt{\frac{r}{n}}, \tag{S.10}$$

and obtain $r^* = \frac{\mu_\ell^2 C^2}{n}$. Thus, for any $\eta > 0$, with probability at least $1 - e^{-\eta}$, there exists a constant $C' > 0$ that satisfies

$$P_n\ell(y,\, h) \leq C'\left(\mathbb{E}\hat{R}_s + \frac{\mu_\ell^2 C^2}{n} + \frac{\eta}{n}\right) \leq C'\left(R_s + \frac{\mu_\ell^2 C^2}{n} + \frac{\eta}{n}\right). \tag{S.11}$$

Note that, for the last inequality, because $\hat{R}_s \leq P_n\ell(y, \langle \alpha, \Phi_1\rangle) + \lambda_\alpha\|\alpha\|^2$ for any α, taking the expectation of both sides yields $\mathbb{E}\hat{R}_s \leq P\ell(y, \langle \alpha, \Phi_1\rangle) + \lambda_\alpha\|\alpha\|^2$, and this gives $\mathbb{E}\hat{R}_s \leq \inf_\alpha\{P\ell(y, \langle\alpha, \Phi_1\rangle) + \lambda_\alpha\|\alpha\|^2\} = R_s$. Consequently, applying Theorem 1 of [25], we have

$$P\ell(y,\, h(x)) \leq P_n\ell(y,\, h(x)) + \tilde{O}\left(\left(\sqrt{\frac{R_s}{n}} + \frac{\mu_\ell C + \sqrt{\eta}}{n}\right)\left(\sqrt{L}C + \sqrt{L\eta}\right) + \frac{C^2 L + L\eta}{n}\right). \tag{S.12}$$

Here, we use $\hat{\mathfrak{R}}_S(\tilde{\mathcal{H}}) \leq C\sqrt{\frac{\hat{R}_s}{n}} \leq C\sqrt{\frac{L}{n}}$. □

Remark D.1. As in [22], without the estimation of the parameters α and β, the right-hand side of Eq. (S.9) becomes $\frac{1}{\sqrt{n}}\left(c_1 + c_2\sqrt{\hat{R}_s}\right)$ with some constant $c_1 > 0$ and $c_2 > 0$, and Eq. (S.10) becomes

$$r = \frac{1}{\sqrt{n}}(c_1 + c_2\sqrt{r}).$$

This yields the solution

$$r^* = \left(\frac{c_2}{2\sqrt{n}} + \sqrt{\left(\frac{c_2}{2\sqrt{n}}\right)^2 + \frac{c_1}{\sqrt{n}}}\right)^2 \leq \frac{c_2^2}{n} + \frac{c_1}{\sqrt{n}},$$

where we use the inequality $\sqrt{x} + \sqrt{x+y} \leq \sqrt{4x + 2y}$. Thus, Eq. (S.11) becomes

$$P_n\ell(y,\, h) \leq C'\left(R_s + \frac{c_2^2}{n} + \frac{c_1}{\sqrt{n}} + \frac{\eta}{n}\right).$$

Consequently, we have the following result:

$$\begin{aligned} &P\ell(y,\, h(x)) \leq P_n\ell(y,\, h(x)) \\ &\quad + \tilde{O}\left(\left(\sqrt{\frac{R_s}{n}} + \frac{\sqrt{c_1}}{n^{3/4}} + \frac{c_2 + \sqrt{\eta}}{n}\right)\left(c_1 + c_2\sqrt{L} + \sqrt{L\eta}\right) + \frac{(c_1 + c_2\sqrt{L})^2 + L\eta}{n}\right). \end{aligned}$$

This means that even if $R_s = \tilde{O}(n^{-1})$, the resulting rate only improves to $\tilde{O}(n^{-3/4})$.

D.3 Proof of Theorem 4.4

Recall that loss function $\ell(\cdot, \cdot)$ is assumed to be μ_ℓ-Lipschitz for the first argument. In addition, we impose the following assumption.

Assumption D.2. There exists a constant $B \geq 1$ such that for every $h \in \mathcal{H}$, $P(h - h^*) \leq BP(\ell(y, h) - \ell(y, h^*))$.

Because we consider $\ell(y, y') = (y - y')^2$ in Theorem 4.4, Assumption D.2 holds as stated in [24].

First, we show the following corollary, which is a slight modification of Theorem 5.4 of [24].

Corollary D.3. *Let $\hat{h}$ be any element of $\mathcal{H}$ satisfying $P_n\ell(y, \hat{h}) = \inf_{h \in \mathcal{H}} P_n\ell(y, h)$, and let $\hat{h}^{(m)}$ be any element of $\mathcal{H}^{(m)}$ satisfying $P_n\ell(y, \hat{h}^{(m)}) = \inf_{h \in \mathcal{H}^{(m)}} P_n\ell(y, h)$. Define*

$$\hat{\psi}(r) = c_1 \hat{\mathfrak{R}}_S\{h \in \mathcal{H} : \max_{m \in \{1,2\}} P_n(h^{(m)} - \hat{h}^{(m)})^2 \le c_3 r\} + \frac{c_2 \eta}{n},$$

where c_1, c_2 and c_3 are constants depending only on B and μ_ℓ. Then, for any $\eta > 0$, with probability at least $1 - 5e^{-\eta}$,

$$P\ell(y, \hat{h}) - P\ell(y, h^*) \le \frac{705}{B}\hat{r}^* + \frac{(11\mu_\ell + 27B)\eta}{n},$$

where $\hat{r}^*$ is the fixed point of $\hat{\psi}$.

Proof. Define the function ψ as

$$\psi(r) = \frac{c_1}{2}\mathfrak{R}\{h \in \mathcal{H} : \mu_\ell^2 \max P(h^{(m)} - h^{(m)*})^2 \le r\} + \frac{(c_2 - c_1)\eta}{n}.$$

Because $\mathcal{H}, \mathcal{H}^{(1)}$ and $\mathcal{H}^{(2)}$ are all convex and thus star-shaped around each of its points, Lemma 3.4 of [24] implies that ψ is a sub-root. Also, define the sub-root function ψ_m as

$$\psi_m(r) = \frac{c_1^{(m)}}{2}\mathfrak{R}\{h^{(m)} \in \mathcal{H}^{(m)} : \mu_\ell^2 P(h^{(m)} - h^{(m)*})^2 \le r\} + \frac{(c_2 - c_1)\eta}{n}.$$

Let r_m^* be the fixed point of $\psi_m(r_m)$. Now, for $r_m \ge \psi_m(r_m)$, Corollary 5.3 of [24] and the condition on the loss function imply that, with probability at least $1 - e^{-\eta}$,

$$\mu_\ell^2 P(\hat{h}^{(m)} - h^{(m)*})^2 \le B\mu_\ell^2 P(\ell(y, \hat{h}^{(m)}) - \ell(y, \hat{h}^{(m)*})) \le 705\mu_\ell^2 r_m + \frac{(11\mu_\ell + 27B)B\mu_\ell^2\eta}{n}.$$

Denote the right-hand side by s_m, and define $r = \max r_m$ and $s = \max s_m$. Because $s \ge s_m \ge r_m \ge r_m^*$, we obtain $s \ge \psi_m(s)$ according to Lemma 3.2 of [24], and thus,

$$s \ge 10\mu_\ell^2\mathfrak{R}\{h^{(m)} \in \mathcal{H}^{(m)} : \mu_\ell^2 P(h^{(m)} - h^{(m)*})^2 \le s\} + \frac{11\mu_\ell^2\eta}{n}.$$

Therefore, applying Corollary 2.2 of [24] to the class $\mu_\ell\mathcal{H}^{(m)}$, it follows that with probability at least $1 - e^{-\eta}$,

$$\{h^{(m)} \in \mathcal{H}^{(m)} : \mu_\ell^2 P(h^{(m)} - h^{(m)*})^2 \le s\} \subseteq \{h^{(m)} \in \mathcal{H}^{(m)} : \mu_\ell^2 P_n(h^{(m)} - h^{(m)*})^2 \le 2s\}.$$

This implies that with probability at least $1 - 2e^{-\eta}$,

$$\begin{aligned} P_n(\hat{h}^{(m)} - h^{(m)*})^2 &\le 2\left(705r + \frac{(11\mu_\ell + 27B)B\eta}{n}\right) \\ &\le 2\left(705 + \frac{(11\mu_\ell + 27B)B}{n}\right)r, \end{aligned}$$

where the second inequality follows from $r \ge \psi(r) \ge \frac{c_2\eta}{n}$. Define $2\left(705 + \frac{(11\mu_\ell+27B)B}{n}\right) = c'$. According to the triangle inequality in $L_2(P_n)$, it holds that

$$\begin{aligned} P_n(h^{(m)} - \hat{h}^{(m)})^2 &\le \left(\sqrt{P_n(h^{(m)} - h^{(m)*})^2} + \sqrt{P_n(h^{(m)*} - \hat{h}^{(m)})^2}\right)^2 \\ &\le \left(\sqrt{P_n(h^{(m)} - h^{(m)*})^2} + \sqrt{c'r}\right)^2. \end{aligned}$$

Again, applying Corollary 2.2 of [24] to $\mu_\ell \mathcal{H}^{(m)}$ as before, but now for $r \geq \psi_m(r)$, it follows that with probability at least $1 - 4e^{-\eta}$,

$$
\begin{aligned}
&\{h \in \mathcal{H} : \mu_\ell^2 \max P(h^{(m)} - h^{(m)*})^2 \leq r\} \\
&\qquad = \bigcap_{m=1}^{2} \{h^{(m)} \in \mathcal{H}^{(m)} : \mu_\ell^2 P(h^{(m)} - h^{(m)*})^2 \leq r\} \\
&\qquad \subseteq \bigcap_{m=1}^{2} \{h^{(m)} \in \mathcal{H}^{(m)} : \mu_\ell^2 P_n(h^{(m)} - h^{(m)*})^2 \leq 2r\} \\
&\qquad \subseteq \bigcap_{m=1}^{2} \{h^{(m)} \in \mathcal{H}^{(m)} : \mu_\ell^2 P_n(h^{(m)} - \hat{h}^{(m)})^2 \leq (\sqrt{2r} + \sqrt{c'r})^2\} \\
&\qquad = \{h \in \mathcal{H} : \mu_\ell^2 \max P_n(h^{(m)} - \hat{h}^{(m)})^2 \leq c_3 r\},
\end{aligned}
$$

where $c_3 = (\sqrt{2} + \sqrt{c'})^2$. Combining this with Lemma A.4 of [24] leads to the following inequality: with probability at least $1 - 5e^{-x}$

$$
\begin{aligned}
\psi(r) &= \frac{c_1}{2} \mathfrak{R}\{h \in \mathcal{H} : \mu_\ell^2 \max P(h^{(m)} - h^{(m)*})^2 \leq r\} + \frac{(c_2 - c_1)\eta}{n} \\
&\leq c_1 \hat{\mathfrak{R}}_S\{h \in \mathcal{H} : \mu_\ell^2 \max P(h^{(m)} - h^{(m)*})^2 \leq r\} + \frac{c_2\eta}{n} \\
&\leq c_1 \hat{\mathfrak{R}}_S\{h \in \mathcal{H} : \mu_\ell^2 \max P_n(h^{(m)} - \hat{h}^{(m)})^2 \leq c_3 r\} + \frac{c_2\eta}{n} \\
&= \hat{\psi}(r).
\end{aligned}
$$

Letting $r = r^*$ and using Lemma 4.3 of [24], we obtain $r^* \leq \hat{r}^*$, thus proving the statement. □

Under Assumption 4.2, we obtain the following excess risk bound for the proposed model class using Corollary D.3. The proof is based on [24].

Theorem D.4. *Let $\hat{h}$ be any element of $\mathcal{H}$ satisfying $P_n\ell(y, \hat{h}(x)) = \inf_{h\in\mathcal{H}} P_n\ell(y, h(x))$. Suppose that Assumption 4.2 is satisfied, then there exists a constant c depending only on μ_ℓ such that for any $\eta > 0$, with probability at least $1 - 5e^{-\eta}$,*

$$
\begin{aligned}
&P(y - \hat{h}(x))^2 - P(y - h^*(x))^2 \\
&\qquad \leq c \left(\min_{0 \leq \kappa_1, \kappa_2 \leq n} \left\{ \frac{\kappa_1 + \kappa_2}{n} + \left(\frac{1}{n} \sum_{j=\kappa_1+1}^{n} \hat{\lambda}_j^{(1)} + \sum_{j=\kappa_2+1}^{n} \hat{\lambda}_j^{(2)} \right)^{\frac{1}{2}} \right\} + \frac{\eta}{n} \right).
\end{aligned}
$$

Theorem D.4 is a multiple-kernel version of Corollary 6.7 of [24], and a data-dependent version of Theorem 2 of [26] which considers the eigenvalues of the Hilbert-Schmidt operators on $\mathcal{H}$ and $\mathcal{H}^{(m)}$. Theorem D.4 concerns the eigenvalues of the Gram matrices $K^{(m)}$ computed from the data.

Proof of Theorem D.4. Define $R = \max_m \sup_{h\in\mathcal{H}^{(m)}} P_n(y - h(x))^2$. For any $h \in \mathcal{H}^{(m)}$, we obtain

$$
P_n(h^{(m)}(x) - \hat{h}^{(m)}(x))^2 \leq 2P_n(y - h^{(m)}(x))^2 + 2P_n(y - \hat{h}^{(m)}(x))^2 \leq 4 \sup_{h\in\mathcal{H}^{(m)}} P_n(y - h(x))^2 \leq 4R.
$$

From the symmetry of the σ_i and the fact that $\mathcal{H}^{(m)}$ is convex and symmetric, we obtain the following:

$$
\begin{aligned}
&\hat{\mathfrak{R}}_S\{h \in \mathcal{H} : \max P_n(h^{(m)} - \hat{h}^{(m)})^2 \leq 4R\} \\
&\qquad = \mathbb{E}_\sigma \sup_{\substack{h^{(m)} \in \mathcal{H}^{(m)} \\ P_n(h^{(m)} - \hat{h}^{(m)})^2 \leq 4R}} \frac{1}{n}\sum_{i=1}^{n} \sigma_i \sum_{m=1}^{2} h^{(m)}(x_i) \\
&\qquad = \mathbb{E}_\sigma \sup_{\substack{h^{(m)} \in \mathcal{H}^{(m)} \\ P_n(h^{(m)} - \hat{h}^{(m)})^2 \leq 4R}} \frac{1}{n}\sum_{i=1}^{n} \sigma_i \sum_{m=1}^{2} (h^{(m)}(x_i) - \hat{h}^{(m)}(x_i)) \\
&\qquad \leq \mathbb{E}_\sigma \sup_{\substack{h^{(m)}, g^{(m)} \in \mathcal{H}^{(m)} \\ P_n(h^{(m)} - g^{(m)})^2 \leq 4R}} \frac{1}{n}\sum_{i=1}^{n} \sigma_i \sum_{m=1}^{2} (h^{(m)}(x_i) - g^{(m)}(x_i)) \\
&\qquad = 2\mathbb{E}_\sigma \sup_{\substack{h^{(m)} \in \mathcal{H}^{(m)} \\ P_n h^{(m)2} \leq R}} \frac{1}{n}\sum_{i=1}^{n} \sigma_i \sum_{m=1}^{2} h^{(m)}(x_i) \\
&\qquad \leq 2\sum_{m=1}^{2} \mathbb{E}_\sigma \sup_{\substack{h^{(m)} \in \mathcal{H}^{(m)} \\ P_n h^{(m)2} \leq R}} \frac{1}{n}\sum_{i=1}^{n} \sigma_i h^{(m)}(x_i) \\
&\qquad \leq 2\sum_{m=1}^{2} \left\{ \frac{2}{n} \sum_{j=1}^{n} \min\{R, \hat{\lambda}_j^{(m)}\} \right\}^{\frac{1}{2}} \\
&\qquad \leq \left\{ \frac{16}{n} \sum_{m=1}^{2} \sum_{j=1}^{n} \min\{R, \hat{\lambda}_j^{(m)}\} \right\}^{\frac{1}{2}}.
\end{aligned}
$$

The second inequality comes from the subadditivity of supremum and the third inequality follows from Theorem 6.6 of [24]. To obtain the last inequality, we use $\sqrt{x} + \sqrt{y} \leq \sqrt{2(x+y)}$. Thus, we have

$$
\begin{aligned}
&2c_1\hat{\mathfrak{R}}_S\{h \in \mathcal{H} : \max P_n(h^{(m)} - \hat{h}^{(m)})^2 \leq 4R\} + \frac{(c_2+2)\eta}{n} \\
&\qquad \leq 4c_1 \left\{ \frac{16}{n} \sum_{m=1}^{2} \sum_{j=1}^{n} \min\left\{R, \hat{\lambda}_j^{(m)}\right\} \right\}^{\frac{1}{2}} + \frac{(c_2+2)\eta}{n},
\end{aligned}
$$

for some constants c_1 and c_2. To apply Corollary D.3, we should solve the following inequality for r

$$
r \leq 4c_1 \left\{ \frac{16}{n} \sum_{m=1}^{2} \sum_{j=1}^{n} \min\left\{r, \hat{\lambda}_j^{(m)}\right\} \right\}^{\frac{1}{2}}.
$$

For any integers $\kappa_m \in [0, n]$, the right-hand side is bounded as

$$
\begin{aligned}
4c_1 \left\{ \frac{16}{n} \sum_{m=1}^{2} \sum_{j=1}^{n} \min\left\{r, \hat{\lambda}_j^{(m)}\right\} \right\}^{\frac{1}{2}} &\leq 4c_1 \left\{ \frac{16}{n} \sum_{m=1}^{2} \left(\sum_{j=1}^{\kappa_m} r + \sum_{j=\kappa_m+1}^{n} \hat{\lambda}_j^{(m)} \right) \right\}^{\frac{1}{2}} \\
&= \left\{ \left(\frac{256c_1^2}{n} \sum_{m=1}^{2} \kappa_m \right) r + \frac{256c_1^2}{n} \sum_{m=1}^{2} \sum_{j=\kappa_m+1}^{n} \hat{\lambda}_j^{(m)} \right\}^{\frac{1}{2}},
\end{aligned}
$$

and we obtain the solution r^* as

$$\begin{aligned} r^* &\leq \frac{128c_1^2}{n}\sum_{m=1}^{2}\kappa_m + \left(\left\{\frac{128c_1^2}{n}\sum_{m=1}^{2}\kappa_m\right\}^2 + \frac{256c_1^2}{n}\sum_{m=1}^{2}\sum_{j=\kappa_m+1}^{n}\hat{\lambda}_j^{(m)}\right)^{\frac{1}{2}} \\ &\leq \frac{256c_1^2}{n}\sum_{m=1}^{2}\kappa_m + \left(\frac{256c_1^2}{n}\sum_{m=1}^{2}\sum_{j=\kappa_m+1}^{n}\hat{\lambda}_j^{(m)}\right)^{\frac{1}{2}}. \end{aligned}$$

Optimizing the right-hand side with respect to κ_1 and κ_2, we obtain the solution as

$$r^* \leq \min_{0\leq\kappa_1,\kappa_2\leq n}\left\{\frac{256c_1^2}{n}\sum_{m=1}^{2}\kappa_m + \left(\frac{256c_1^2}{n}\sum_{m=1}^{2}\sum_{j=\kappa_m+1}^{n}\hat{\lambda}_j^{(m)}\right)^{\frac{1}{2}}\right\}.$$

Furthermore, according to Corollary D.3, there exists a constant c such that with probability at least $1-5e^{-\eta}$,

$$\begin{aligned} &P(y-\hat{h}(x))^2 - P(y-h^*(x))^2 \\ &\qquad \leq c\left(\min_{0\leq\kappa_1,\kappa_2\leq n}\left\{\frac{1}{n}\sum_{m=1}^{2}\kappa_m + \left(\frac{1}{n}\sum_{m=1}^{2}\sum_{j=\kappa_m+1}^{n}\hat{\lambda}_j^{(m)}\right)^{\frac{1}{2}}\right\} + \frac{\eta}{n}\right). \end{aligned}$$

□

With Theorem D.4 and Assumption 4.3, we prove Theorem 4.4 as follows.

Proof of Theorem 4.4. Using the inequality $\sqrt{x+y} \leq \sqrt{x}+\sqrt{y}$ for $x \geq 0, y \geq 0$, we have

$$\begin{aligned} &P(y-\hat{h}(x))^2 - P(y-h^*(x))^2 \\ &\quad = O\left(\min_{0\leq\kappa_1,\kappa_2\leq n}\left\{\frac{\kappa_1+\kappa_2}{n} + \left(\frac{1}{n}\sum_{j=\kappa_1+1}^{n}\hat{\lambda}_j^{(1)} + \frac{1}{n}\sum_{j=\kappa_2+1}^{n}\hat{\lambda}_j^{(2)}\right)^{\frac{1}{2}}\right\} + \frac{\eta}{n}\right) \\ &\quad \leq O\left(\min_{0\leq\kappa_1,\kappa_2\leq n}\left\{\frac{\kappa_1+\kappa_2}{n} + \left(\frac{1}{n}\sum_{j=\kappa_1+1}^{n}j^{-\frac{1}{s_1}} + \frac{1}{n}\sum_{j=\kappa_2+1}^{n}j^{-\frac{1}{s_2}}\right)^{\frac{1}{2}}\right\} + \frac{\eta}{n}\right) \\ &\quad \leq O\left(\min_{0\leq\kappa_1,\kappa_2\leq n}\left\{\frac{\kappa_1+\kappa_2}{n} + \left(\frac{1}{n}\sum_{j=\kappa_1+1}^{n}j^{-\frac{1}{s_1}}\right)^{\frac{1}{2}} + \left(\frac{1}{n}\sum_{j=\kappa_2+1}^{n}j^{-\frac{1}{s_2}}\right)^{\frac{1}{2}}\right\} + \frac{\eta}{n}\right). \end{aligned}$$

Because it holds that

$$\sum_{j=\kappa_m+1}^{n}j^{-\frac{1}{s_m}} < \int_{\kappa_m}^{\infty}x^{-\frac{1}{s_m}}\mathrm{d}x < \left[\frac{1}{1-\frac{1}{s_m}}x^{1-\frac{1}{s_m}}\right]_{\kappa_m}^{\infty} = \frac{s_m}{1-s_m}\kappa_m^{1-\frac{1}{s_m}},$$

for $m=1,2$, we should solve the following minimization problem:

$$\min_{0\leq\kappa_1,\kappa_2\leq n}\left\{\frac{\kappa_1+\kappa_2}{n} + \left(\frac{1}{n}\frac{s_1}{1-s_1}\kappa_1^{1-\frac{1}{s_1}}\right)^{\frac{1}{2}} + \left(\frac{1}{n}\frac{s_1}{1-s_1}\kappa_2^{1-\frac{1}{s_1}}\right)^{\frac{1}{2}}\right\} \equiv g(\kappa).$$

Taking the derivative, we have

$$\frac{\partial g(\kappa)}{\partial\kappa_1} = \frac{1}{n} + \frac{1}{2}\left(\frac{1}{n}\frac{s_1}{1-s_1}\kappa_1^{1-\frac{1}{s_1}}\right)^{-\frac{1}{2}}\left(-\frac{\kappa_1^{-\frac{1}{s_1}}}{n}\right).$$

Setting this to zero, we find the optimal κ_1 as

$$\kappa_1 = \left(\frac{s_1}{1-s_1}\frac{4}{n}\right)^{\frac{s_1}{1+s_1}}.$$

Similarly, we have

$$\kappa_2 = \left(\frac{s_2}{1-s_2}\frac{4}{n}\right)^{\frac{s_2}{1+s_2}},$$

and

$$\begin{aligned}
&P(y-\hat{h}(x))^2 - P(y-h^*(x))^2\\
&\le O\bigg(\frac{1}{n}\left(\frac{s_1}{1-s_1}\frac{4}{n}\right)^{\frac{s_1}{1+s_1}} + \frac{1}{n}\left(\frac{s_2}{1-s_2}\frac{4}{n}\right)^{\frac{s_2}{1+s_2}}\\
&\qquad\qquad + 2^{\frac{1-s_1}{1+s_1}}\left(\frac{s_1}{1-s_1}\frac{1}{n}\right)^{\frac{1}{1+s_1}} + 2^{\frac{1-s_2}{1+s_2}}\left(\frac{s_2}{1-s_2}\frac{1}{n}\right)^{\frac{1}{1+s_2}} + \frac{\eta}{n}\bigg)\\
&= O\Big(n^{-\frac{1}{1+s_1}} + n^{-\frac{1}{1+s_2}}\Big)\\
&= O\Big(n^{-\frac{1}{1+\max\{s_1,s_2\}}}\Big).
\end{aligned}$$

$\square$

References

[1] S. S. Du, J. Koushik, A. Singh, and B. Póczos, "Hypothesis transfer learning via transformation functions," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[2] I. Kuzborskij and F. Orabona, "Stability and hypothesis transfer learning," *International Conference on Machine Learning*, pp. 942–950, 2013.

[3] R. Vershynin, *High-dimensional probability: An introduction with applications in data science*, vol. 47. Cambridge university press, 2018.

[4] S. Ju, R. Yoshida, C. Liu, K. Hongo, T. Tadano, and J. Shiomi, "Exploring diamond-like lattice thermal conductivity crystals via feature-based transfer learning," *Physical Review Materials*, vol. 5, no. 5, p. 053801, 2021.

[5] C. Liu, E. Fujita, Y. Katsura, Y. Inada, A. Ishikawa, R. Tamura, K. Kimura, and R. Yoshida, "Machine learning to predict quasicrystals from chemical compositions," *Advanced Materials*, vol. 33, no. 36, p. 2102507, 2021.

[6] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *International Conference for Learning Representations*, 2015.

[7] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?," *Advances in Neural Information Processing Systems*, vol. 27, 2014.

[8] J. Wang, R. M. Wolf, J. W. Caldwell, P. A. Kollman, and D. A. Case, "Development and testing of a general amber force field," *Journal of Computational Chemistry*, vol. 25, no. 9, pp. 1157–1174, 2004.

[9] Y. Hayashi, J. Shiomi, J. Morikawa, and R. Yoshida, "Radonpy: automated physical property calculation using all-atom classical molecular dynamics simulations for polymer informatics," *npj Computational Materials*, vol. 8, no. 222, 2022.

[10] S. Otsuka, I. Kuwajima, J. Hosoya, Y. Xu, and M. Yamazaki, "Polyinfo: Polymer database for polymeric materials design," *2011 International Conference on Emerging Intelligent Data and Web Technologies*, pp. 22–29, 2011.

[11] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning*, vol. 2. MIT press Cambridge, MA, 2006.

[12] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization.," *Journal of machine learning research*, vol. 12, no. 7, 2011.

[13] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," *International Conference on Machine Learning*, 2017.

[14] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *International Conference for Learning Representations*, 2015.

[15] L. Xuhong, Y. Grandvalet, and F. Davoine, "Explicit inductive bias for transfer learning with convolutional networks," *International Conference on Machine Learning*, pp. 2825–2834, 2018.

[16] S. Myung, I. Huh, W. Jang, J. M. Choe, J. Ryu, D. Kim, K.-E. Kim, and C. Jeong, "PAC-Net: A model pruning approach to inductive transfer learning," *International Conference on Machine Learning*, pp. 16240–16252, 2022.

[17] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[18] I. Beltagy, K. Lo, and A. Cohan, "SciBERT: A pretrained language model for scientific text," in *Conference on Empirical Methods in Natural Language Processing*, 2019.

[19] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of Machine Learning Research*, vol. 21, no. 140, pp. 1–67, 2020.

[20] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[21] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[22] I. Kuzborskij and F. Orabona, "Fast rates by transferring from auxiliary hypotheses," *Machine Learning*, vol. 106, no. 2, pp. 171–195, 2017.

[23] M. Mohri, A. Rostamizadeh, and A. S. Talwalkar, *Foundations of Machine Learning*. MIT press, 2018.

[24] P. L. Bartlett, O. Bousquet, and S. Mendelson, "Local Rademacher complexities," *Annals of Statistics*, vol. 33, pp. 1497–1537, 2005.

[25] N. Srebro, K. Sridharan, and A. Tewari, "Smoothness, low noise and fast rates," *Advances in Neural Information Processing Systems*, vol. 23, 2010.

[26] M. Kloft and G. Blanchard, "The local Rademacher complexity of lp-norm multiple kernel learning," *Advances in Neural Information Processing Systems*, vol. 24, 2011.