LEVERAGING ENSEMBLE DIVERSITY FOR ROBUST SELF-TRAINING IN THE PRESENCE OF SAMPLE SELECTION BIAS

Ambroise Odonnat, Vasilii Feofanov, Ievgen Redko
Huawei Paris Noah's Ark Lab

ABSTRACT

Self-training is a well-known approach for semi-supervised learning. It consists of iteratively assigning pseudo-labels to unlabeled data for which the model is confident and treating them as labeled examples. For neural networks, `softmax` prediction probabilities are often used as a confidence measure, despite the fact that they are known to be overconfident, even for wrong predictions. This phenomenon is particularly intensified in the presence of sample selection bias, i.e., when data labeling is subject to some constraint. To address this issue, we propose a novel confidence measure, called $\mathcal{T}$-similarity, built upon the prediction diversity of an ensemble of linear classifiers. We provide the theoretical analysis of our approach by studying stationary points and describing the relationship between the diversity of the individual members and their performance. We empirically demonstrate the benefit of our confidence measure for three different pseudo-labeling policies on classification datasets of various data modalities.

1 Introduction

Deep learning has been remarkably successful in the past decade when large amounts of labeled data became available (He et al., 2016; Goodfellow et al., 2014; Dosovitskiy et al., 2021). However, in many real-world applications data annotation is costly and time-consuming (Imran et al., 2020), while data acquisition is cheaper and may result in an abundance of unlabeled examples (Fergus et al., 2009). In this context, semi-supervised learning (Chapelle et al., 2006, denoted by `SSL`) has emerged as a powerful approach to exploit both labeled and unlabeled data (van Engelen and Hoos, 2020). Among existing `SSL` techniques, self-training (Amini et al., 2023) received a lot of interest in recent years (Lee, 2013; Sohn et al., 2020; Zhang et al., 2021; Chen et al., 2022). The main idea behind the self-training approach is to use the predictions of a classifier learned on available labeled data to pseudo-label unlabeled data and progressively include them in the labeled set during the training. Traditionally, at each iteration of self-training, we select for pseudo-labeling the unlabeled examples that have a prediction confidence above a certain threshold. The latter can be fixed (Yarowsky, 1995), dynamic along the training (Cascante-Bonilla et al., 2021), or optimized (Feofanov et al., 2019). When it comes to analyzing the performance of self-training algorithms, two fundamental questions primarily arise: **(1)** How to rank unlabeled data to reflect their difficulty for classification? **(2)** How to select the threshold for pseudo-labeling at each iteration? While the second question is related to the choice of a pseudo-labeling policy and is already well addressed in the literature (Feofanov et al., 2019; Cascante-Bonilla et al., 2021; Zhang et al., 2021), the first question is related to confidence estimation and remains an open problem as the most conventional choice for such a ranking – the `softmax` prediction probability – is known to suffer from overconfidence (Wei et al., 2022).

In this work, we propose a reliable ranking measure for pseudo-labeling and place ourselves in a challenging, yet realistic, scenario of `SSL` with a distribution shift. For the latter, we consider the sample selection bias (`SSB`) setup (Heckman, 1974) where the annotation of training data is subject to certain constraints. Selection bias is known to occur in survey design (Quionero-Candela et al., 2009), in medical research studies during the creation of cohorts and control groups (Alves, 2006; Ahern, 2018; Arias et al., 2023), or in industry due to privacy or security reasons. We denote this challenging scenario by `SSL + SSB`.

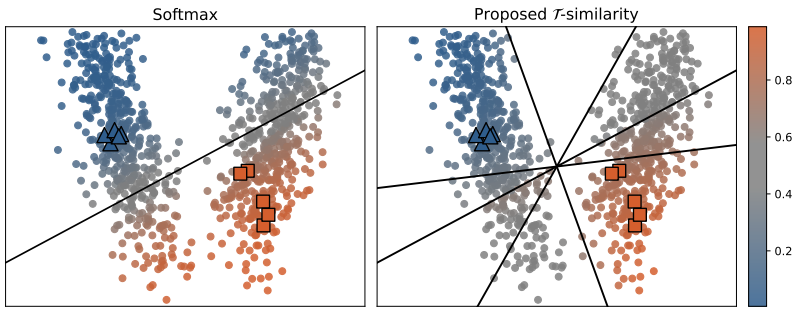


Figure 1: Unlabeled data (circles) colored by the confidence value of being from the orange class (right point cloud), from blue to orange as it increases. **Left**: Given a model trained on few labeled examples (▲■), `softmax` may provide wrong confidence estimates for unlabeled data. **Right**: Our method averages confidence estimates of a diverse set of classifiers leading to a well-calibrated model robust to distribution shift.

Summary of contributions. We summarize our contributions as follows:

1. We propose a novel confidence measure, illustrated in Figure 1, that builds upon the diversity of an ensemble of classifiers. Such a measure is easy to implement into any popular SSL method using neural networks as a backbone;
2. We provide a thorough theoretical analysis of our method by studying stationary points and showing the connection between diversity and the performance of the individual classifiers;
3. We experimentally demonstrate the superiority of our approach for self-training on various SSL datasets under SSB. Additionally, we show that SSB degrades the performance of other popular methods when not dealt with properly.

2 Related Work

Self-training. In SSL, a classical strategy to incorporate unlabeled data into the learning process is to use the predictions of a supervised model on it, either for regularization (Grandvalet and Bengio, 2004; Feofanov et al., 2023) or for self-training by iteratively including most confident pseudo-labeled data to the labeled set. The latter approach is widespread in computer vision, where self-training is often combined with consistency regularization to encourage similar predictions for different augmentations of the same unlabeled image (Sohn et al., 2020; Zhang et al., 2021; Chen et al., 2022). Correctly choosing the confidence threshold for unlabeled data is key to the success of self-training. Instead of using a fixed threshold, several works propose to select the threshold at each iteration via curriculum learning to control the number of pseudo-labeled examples (Cascante-Bonilla et al., 2021; Zhang et al., 2021), while Feofanov et al. (2019) finds the threshold as a balance between the upper-bounded transductive error on the pseudo-labeled examples and their number. All of these methods strongly depend on the confidence measure of the base classifier, and thus are not well suited when the base classifier is biased towards the labeled set, or when labeled and unlabeled data follow different probability distributions. In this work, we aim to fill this gap and propose a model- and application-agnostic confidence estimation approach that is robust to such distribution mismatch.

Sample selection bias. SSB describes the situation where the distribution mismatch between labeled and unlabeled data is due to some unknown sample selection process, i.e., when data labeling is subject to some constraints. Formalized by Heckman (1974), this framework received a lot of attention in the 1980s in the case of linear regression from the econometrics community (Lee, 1982; Heckman, 1990). In the context of classification, SSB was properly defined by Zadrozny (2004), and most of the methods address it via importance sampling by estimating biased densities or selection probabilities (Zadrozny, 2004; Dudík et al., 2005; Shimodaira, 2000) or by using prior knowledge about the class distributions (Lin et al., 2002). Alternatively, the resampling weights can be inferred by kernel mean matching (Huang et al., 2006). All these methods heavily rely on density estimation and thus are not well suited in SSL where labeled data is scarce. In our work, we propose to turn this curse of scarcity of labeled data into a blessing by exploiting the diversity of a set of classifiers that can be fit to a handful of available labeled points.

Ensemble diversity. It is well known that an ensemble of learners (Hansen and Salamon, 1990) is efficient when its members are *"diverse"* in a certain sense (Dietterich, 2000; Kuncheva, 2004; Lu et al., 2010). Over the last decades,

generating diversity has been done in many ways, including bagging (Breiman, 2001), boosting (Freund and Schapire, 1997; Friedman, 2001), and random subspace training (Ho, 1998). These methods, however, are based on implicit diversity criteria, calling for new approaches where the ensemble diversity can be defined explicitly. To this end, Liu and Yao (1999) introduce a mixture of experts that are diversified through the negative correlation loss that forces a trade-off between specialization and cooperation of the experts. Buschjäger et al. (2020) derive a bias-variance decomposition that encompasses many existing diversity methods, particularly showing that the negative correlation loss is linked to the prediction variance of the ensemble members. Ortega et al. (2022) derive an upper bound over the generalization error of the majority vote classifier showing that the performance of the ensemble depends on the error variance of the individual classifiers. Some recent works rely on ensemble diversity to estimate accuracy on a given test set, namely, Jiang et al. (2022) use the disagreement rate of two independently trained neural networks, while Chen et al. (2021) evaluate the disagreement between a deep ensemble and a given pre-trained model. The closest method to our work is that of Zhang and Zhou (2013) which learns an ensemble classifier by imposing diversity in predictions on unlabeled data. In this paper, we extend their binary setting to multi-class classification and push their idea further by showing the benefits of using diversity for calibration and confidence estimation in self-training under distribution shift. In addition, we provide a theoretical explanation of why the diversity imposed in such a way works in practice.

3 Our Contributions

Notations. Scalar values are denoted by regular letters (e.g. parameter λ), vectors are represented in bold lowercase letters (e.g. vector $\mathbf{x}$) and matrices are represented by bold capital letters (e.g., matrix $\mathbf{A}$). The i-th row of the matrix $\mathbf{A}$ is denoted by $\mathbf{A}_i$ and its j-th column is denoted by $\mathbf{A}_{\cdot,j}$. The trace of a matrix $\mathbf{A}$ is denoted $\mathrm{Tr}(\mathbf{A})$ and its transpose by $\mathbf{A}^\top$. The identity matrix of size n is denoted by $\mathbf{I}_n \in \mathbb{R}^{n\times n}$. The unit vector of size n with each entry equal to 1 is denoted by $\mathbb{1}_n$. The unit matrix of size n with each entry equal to 1 is denoted by $\mathbf{U}^{[n]} \in \mathbb{R}^{n\times n}$. We have $\mathbf{U}^{[n]} = \mathbb{1}_n \mathbb{1}_n^\top$. We denote $\lambda_{\min}(\mathbf{A})$ and $\lambda_{\max}(\mathbf{A})$ the minimum and maximum eigenvalues of a matrix $\mathbf{A}$, respectively.

3.1 Problem setup

Semi-supervised learning. Consider the classification problem with input space $\mathcal{X}$ and label space $\mathcal{Y}=\{1,\dots,C\}$. Let $(\mathbf{X}_\ell, \mathbf{y}_\ell) = (\mathbf{x}_i, y_i)_{i=1}^{n_\ell} \in (\mathcal{X}\times\mathcal{Y})^{n_\ell}$ be the set of labeled training examples. Let $\mathbf{X}_u = (\mathbf{x}_i)_{i=n_\ell+1}^{n_\ell+n_u} \in \mathcal{X}^{n_u}$ be the set of unlabeled training examples. The hypothesis space is denoted $\mathcal{H} = \{h\colon \mathcal{X} \to \Delta_C\}$, where $\Delta_C = \{p \in [0,1]^C | \sum_{c=1}^C p_c = 1\}$ is the probability simplex. For an input $\mathbf{x}\in\mathcal{X}$, a learning model $h\in\mathcal{H}$, $h(\mathbf{x})$ is a probability measure on $\mathcal{Y}$, and the predicted label is defined as $\hat{y} = \arg\max h(\mathbf{x})$.

Sample selection bias. We model SSB for labeled data following Quionero-Candela et al. (2009, chap. 3) and introduce a random binary selection variable s, where $s=1$ means that the training point is labeled, while $s=0$ implies that it remains unlabeled. Assuming the true stationary distribution of data P on $\mathcal{X}\times\mathcal{Y}$, we consider that labeled training examples are i.i.d. drawn from P_{lab}, while unlabeled training and test examples are from P_{unlab} and P_{test} respectively, with P_{lab}, P_{unlab} and P_{test} defined as follows:

$$P_{\text{lab}}(\mathbf{x},y) = P(\mathbf{x},y|s=1) = \frac{P(s=1|\mathbf{x},y)}{P(s=1)}P(\mathbf{x},y),$$
$$P_{\text{unlab}}(\mathbf{x},y) = P_{\text{test}}(\mathbf{x},y) = P(\mathbf{x},y).$$

Self-training. Most commonly, a self-training algorithm is first initialized by the base classifier trained using only labeled data $(\mathbf{X}_l, \mathbf{y}_\ell)$. Then, at each iteration i, the algorithm measures the prediction confidence for unlabeled points from $\mathbf{X}_u$, typically, through prediction probabilities like `softmax`. Based on these confidence estimates, a pseudo-labeling policy determines the unlabeled examples that are pseudo-labeled by the corresponding model's predictions. These pseudo-labeled data are moved from $\mathbf{X}_u$ to $(\mathbf{X}_\ell, \mathbf{y}_\ell)$ and the classifier is re-trained. The same procedure is repeated for several iterations until stop criteria are satisfied. Algorithm 1, given in Appendix A.1, outlines the pseudo-code of this learning method.

3.2 Proposed approach

Similarity as a surrogate of confidence. Our main idea is to use the similarity between ensemble predictions on a given unlabeled example to estimate the prediction's confidence, instead of using the usual `softmax` prediction

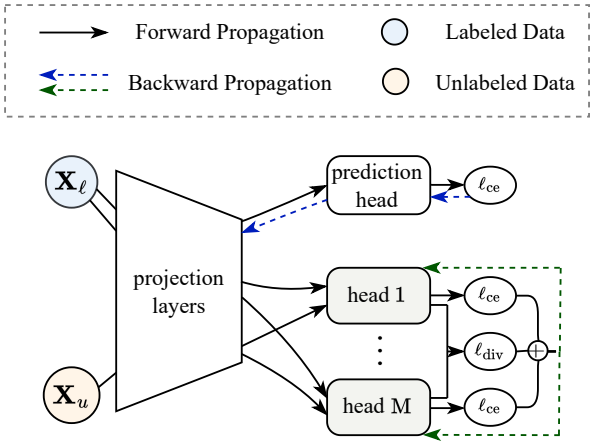


Figure 2: Architecture of the model.

probability. The underlying intuition is to say that if the individual, but diverse, classifiers agree on a point, then the associated prediction can be trusted with high confidence. Conversely, if we find many ways to disagree on a given point then it is likely a difficult point for our model, so a low confidence is attributed. Below, we formalize the proposed confidence measure:

Definition 3.1 ($\mathcal{T}$-similarity)**.** *Consider an unlabeled data point $\mathbf{x} \in \mathbf{X}_u$ and an ensemble of classifiers $\mathcal{T} = \{h_m \in \mathcal{H} | 1 \leq m \leq M\}$. The $\mathcal{T}$-similarity of $\mathbf{x}$ is defined by:*

$$s_{\mathcal{T}}(\mathbf{x}) = \frac{1}{M(M-1)} \sum_{m \neq k} h_m(\mathbf{x})^\top h_k(\mathbf{x}).$$

We now present a simple property of the proposed confidence measure $s_{\mathcal{T}}$ highlighting that it can be used as an alternative to `softmax` prediction probabilities when the output of the individual classifiers lies in the probability simplex Δ_C. The proof is given in Appendix E.1.

Proposition 3.2 (Property of $s_{\mathcal{T}}$)**.** *Let $\mathcal{T}$ be an ensemble of probabilistic classifiers. Then, for any input $x \in \mathcal{X}$, we have:*

$$0 \leq s_{\mathcal{T}}(\mathbf{x}) \leq 1.$$

Learning with the $\mathcal{T}$-similarity. Following Zhang and Zhou (2013), we train an ensemble to fit well the labeled set while having the most diverse possible individual predictions on the unlabeled set. We achieve this by minimizing the following loss function:

$$\mathcal{L}_{\text{conf}}(\mathcal{T}, (\mathbf{X}_\ell, \mathbf{y}_\ell), \mathbf{X}_u) = \underbrace{\frac{1}{M} \sum_{m=1}^{M} \ell_{\text{sup}}(h_m, \mathbf{X}_\ell, \mathbf{y}_\ell)}_{\text{label fidelity term}} - \underbrace{\gamma\, \ell_{\text{div}}(\mathcal{T}, \mathbf{X}_u)}_{\text{diversity term}}, \tag{1}$$

where ℓ_{sup} is a supervised loss evaluated on the labeled examples, typically the cross-entropy loss ℓ_{ce}, ℓ_{div} corresponds to the diversity loss of the ensemble $\mathcal{T}$, and $\gamma \geq 0$ is a hyperparameter that controls the strength of the imposed diversity. As maximizing the diversity amounts to minimizing the similarity, we consider

$$\ell_{\text{div}}(\mathcal{T}, \mathbf{X}_u) = -\frac{1}{n_u} \sum_{i=n_\ell+1}^{n_\ell+n_u} s_{\mathcal{T}}(\mathbf{x}). \tag{2}$$

Practical implementation. To combine confidence estimation and prediction, we introduce the neural network described in Figure 2. First, input data is projected on a high-dimensional feature space. The projection layers are

learned together with a classification head that is also used for predicting pseudo-labels. This prediction head, denoted by h_{pred}, is updated via backpropagation of the supervised loss $\mathcal{L}_{\text{sup}}\left(h_{\text{pred}}, (\mathbf{X}_\ell, \mathbf{y}_\ell)\right) = \ell_{\text{sup}}(h_{\text{pred}}, (\mathbf{X}_\ell, \mathbf{y}_\ell))$. Another part of the network is responsible for confidence estimation. After projecting inputs to the hidden space, an ensemble of M heads follows and is optimized using (1). The important thing is that backpropagation of (1) only influences the ensemble heads, not the projection layers, since we want to estimate prediction confidence given a fixed representation. Then, the total loss for the network is:

$$\mathcal{L}_{\text{sup}}\left(h_{\text{pred}}, (\mathbf{X}_\ell, \mathbf{y}_\ell)\right) + \mathcal{L}_{\text{conf}}\left(\mathcal{T}, (\mathbf{X}_\ell, \mathbf{y}_\ell), \mathbf{X}_u\right).$$

In our experiments, we consider $M = 5$ linear heads. It results in fast training and no significant computational burden. The ease of implementation enables the combination of the proposed $\mathcal{T}$-similarity with any `SSL` method that uses a neural network as a backbone.

3.3 Theoretical analysis

We now provide the theoretical guarantees of the proposed learning framework. For the sake of clarity, we formulate the problem in the binary classification case and consider only the confidence estimation part (the ensemble of heads) for a fixed representation space. We show that, under a mild assumption, the solution of the optimization problem (minimization of (1)) is unique. In addition, we establish a lower bound on the diversity of the optimal ensemble that gives theoretical insights into the relationship between diversity and the performance of the individual classifiers from the considered ensemble.

Problem formulation. Assuming $\mathcal{Y} = \{-1, +1\}$ and centered training data, we parameterize a linear head h_m, $m \in [\![1, M]\!]$, by a separating hyperplane $\boldsymbol{\omega}_m \in \mathbb{R}^d$ and $h_m(\mathbf{x}) = \boldsymbol{\omega}_m^\top \mathbf{x} \in \mathbb{R}$ so that Proposition 3.2 no longer holds. For a given example $\mathbf{x} \in \mathbb{R}^d$, the classifier h_m predicts the label by $\text{sign}(h_m(\mathbf{x}))$. We denote $\mathbf{W} \in \mathbb{R}^{d \times M}$ the matrix whose columns are the separating hyperplanes $\boldsymbol{\omega}_m$, i.e., $\forall m \in [\![1, M]\!]$, $\mathbf{W}_{\cdot, m} = \boldsymbol{\omega}_m \in \mathbb{R}^d$. In the rest of this section, we refer to $\mathbf{W}$ as the ensemble of classifiers instead of using the notation $\mathcal{T}$. For practical considerations from the theoretical point of view, we consider ridge (also known as LS-SVM (Suykens and Vandewalle, 1999)) classifiers that minimize the least-square loss with Tikhonov regularization. Then, following this setup, we can re-write (1), and formulate the optimization problem as:

$$\begin{aligned}\underset{\mathbf{W} \in \mathbb{R}^{d \times M}}{\arg\min}\ \mathcal{L}(\mathbf{W}) \coloneqq & \underbrace{\frac{1}{M n_\ell} \sum_{m=1}^{M} \sum_{i=1}^{n_\ell} \left(y_i - \boldsymbol{\omega}_m^\top \mathbf{x}_i\right)^2}_{\text{label fidelity term}} + \underbrace{\frac{1}{M} \sum_{m=1}^{M} \lambda_m \|\boldsymbol{\omega}_m\|_2^2}_{\text{regularization}} \\ & + \underbrace{\frac{\gamma}{n_u M (M-1)} \sum_{m \neq k} \sum_{i=n_\ell+1}^{n_\ell+n_u} \boldsymbol{\omega}_m^\top \mathbf{x}_i \boldsymbol{\omega}_k^\top \mathbf{x}_i}_{\text{unlabeled diversity term}},\end{aligned} \tag{P}$$

where the last term is the diversity loss ℓ_{div} from (2) with linear classifiers. We note that in the binary framework, the similarity measure does not lie in the $[0, 1]$ interval, but can take any real value. However, with a reasonable choice of γ, the binarized objective should lead to similar diverse ensembles. In the next paragraph, we show that, under a mild assumption on the λ_m, the problem (**P**) can be solved efficiently.

Convergence to a stationary point. In practice, as $\mathcal{L}$ is differentiable, the learning problem is solved via gradient descent, which aims at finding local minimizers of $\mathcal{L}$. Such minimizers are stationary points, i.e., solutions of the Euler equation

$$\nabla \mathcal{L}(\mathbf{W}) = 0. \tag{3}$$

In our setting, (3) reduces to a linear problem in $\mathbf{W}$. More details can be found in Appendix E.2. We now make the following assumption on the parameters λ_m:

$$\forall m \in [\![1, M]\!], \lambda_m > \frac{\gamma(M+1)}{n_u(M-1)} \lambda_{\max}(\mathbf{X}_u^\top \mathbf{X}_u). \tag{A}$$

Assumption **A** ensures that $\lambda_m \mathbf{I}_d - \frac{\gamma(M+1)}{n_u(M-1)} \mathbf{X}_u^\top \mathbf{X}_u$ is positive definite, and thereby that $\lambda_m > 0$, for all m. The next proposition establishes the convergence of (**P**) towards a unique solution. The detailed proof is given in Appendix E.3.

Proposition 3.3 (Convergence of (P)). *Under Assumption A, $\mathcal{L}$ is strictly convex and coercive on $\mathbb{R}^{d\times M}$. Hence, the optimization problem (P) admits a unique solution $\mathbf{W}^*$ that verifies (3).*

Proposition 3.3 highlights the fact that, under assumption **A**, solving (**P**) amounts to solving (3). In the rest of this section, we study a measure of diversity for the stationary points of $\mathcal{L}$, i.e., solutions of (3).

Relationship between diversity and individual performance. We proceed with the analysis of the diversity loss ℓ_{div} on the unlabeled set re-written as:

$$\ell_{\text{div}}(\mathbf{W}, \mathbf{X}_u) = -\frac{1}{n_u M(M-1)} \sum_{m\neq k} \boldsymbol{\omega}_m^\top \mathbf{X}_u^\top \mathbf{X}_u \boldsymbol{\omega}_k. \tag{4}$$

We now want to develop our intuition about when ℓ_{div} achieves its maximum value. We note that $\mathbf{X}_u^\top \mathbf{X}_u$ is positive semi-definite then $\boldsymbol{\omega}_m^\top \mathbf{X}_u^\top \mathbf{X}_u$ remains in the same half-space as $\boldsymbol{\omega}_m$. The positive value of ℓ_{div} is then achieved when the angle between $\boldsymbol{\omega}_m$ and $\boldsymbol{\omega}_k$ is between 90 and 180 degrees. The next theorem characterizes the diversity of stationary points of $\mathcal{L}$, i.e., solutions of (3). The detailed proof is given in Appendix E.4.

Theorem 3.4 (A lower bound on the diversity). *Let $\tilde{\mathbf{W}}$ be a stationary point of $\mathcal{L}$, i.e., solution of (3). We denote $\tilde{\boldsymbol{\omega}}_m$ its m-th column and assume that $\frac{1}{M}\sum_{m=1}^{M} \lambda_m \|\tilde{\boldsymbol{\omega}}_m\|_2^2 \geq 1$. Then, we have*

$$\gamma \ell_{div}(\tilde{\mathbf{W}}, \mathbf{X}_u) \geq \frac{1}{2n_\ell M} \sum_{m=1}^{M} \|\mathbf{y}_\ell - \mathbf{X}_\ell \tilde{\boldsymbol{\omega}}_m\|_2^2 + \frac{1}{2M} \sum_{m=1}^{M} \tilde{\boldsymbol{\omega}}_m^\top \left(\lambda_m \mathbf{I}_d + \frac{\mathbf{X}_\ell^\top \mathbf{X}_\ell}{n_\ell}\right) \tilde{\boldsymbol{\omega}}_m.$$

From Theorem 3.4, we obtain that the diversity loss of stationary points of $\mathcal{L}$ is non-negative, although, as discussed above, $\ell_{\text{div}}(\mathbf{W}, \mathbf{X}_u)$ may be negative. This implies that the diversity term encourages opposite predictions between classifiers, while the labeled loss term minimized explicitly by our approach likely prevents completely colinear solutions that would degrade the supervised loss too much. We also note that the second term, which we do not optimize explicitly, can be decomposed into a weighted sum of the norms of the individual classifiers $\frac{1}{2M}\sum_{m=1}^{M} \lambda_m ||\tilde{\boldsymbol{\omega}}_m||_2^2$ and a margin term $\frac{1}{2Mn_\ell}\sum_{m=1}^{M} \left(\tilde{\boldsymbol{\omega}}_m^\top \mathbf{X}_\ell^\top \mathbf{X}_\ell \tilde{\boldsymbol{\omega}}_m\right)$. Assuming for simplicity that the $\tilde{\boldsymbol{\omega}}_m$ are orthogonal, it implies that high diversity is achieved by finding predictors of the largest possible margin so that they also span the M directions of the largest variance of the labeled data. This insight is quite important as we do not explicitly consider the spectral properties of the labeled data in our approach. Yet, we implicitly exploit them by using a very simple and lightweight approach.

The role of representation learning. So far, we assumed that the labeled data representation is fixed. In this case, we showed that the diversity is high when the individual predictors cover the directions of large variance in the data. Below, we show that the direction of the smallest variance in $\mathbf{X}_\ell$ is also important for diversity suggesting that labeled data covering the input space evenly is beneficial to our approach. The proof is given in Appendix E.5.

Corollary 3.5 (Connection to contrastive learning). *Let $\tilde{\mathbf{W}}$ be a stationary point of $\mathcal{L}$, i.e., solution of (3). We denote $\tilde{\boldsymbol{\omega}}_m$ its m-th column. Assuming that the condition of Theorem 3.4 holds and that all λ_m are equal to some λ, we have:*

$$\gamma \ell_{div}(\tilde{\mathbf{W}}, \mathbf{X}_u) \geq \frac{1}{2M}\left(\lambda + \frac{1}{n_\ell}\lambda_{\min}\left(\mathbf{X}_\ell^\top \mathbf{X}_\ell\right)\right) \|\tilde{\mathbf{W}}\|_{\text{F}}^2.$$

As $\lambda_{\min}(\mathbf{X}_\ell^\top \mathbf{X}_\ell)$ represents the magnitude of the spread in the direction of less variance, we want to push $\lambda_{\min}(\mathbf{X}_\ell^\top \mathbf{X}_\ell)$ away from 0 to span the whole space as evenly as possible. When $\mathbf{X}_\ell$ is the output of an embedding layer of a neural network, this idea is reminiscent of contrastive learning methods that learn an embedding space having a uniform distribution on the sphere (Wang and Isola, 2020). We leave the idea of coupling diversity maximization with contrastive learning suggested by Corollary 3.5 for future research.

4 Experiments

In this section, we first showcase the failure of self-training in the SSB setting when the `softmax` is used as a confidence measure. Then, we empirically demonstrate the effectiveness of the $\mathcal{T}$-similarity for confidence estimation and for self-

training on common classification datasets with different data modalities. All the implementation details are given in Appendix B.2.

Datasets. We consider 13 publicly available SSL datasets with various data modalities: biological data for Cod-RNA (Chang and Lin, 2011), DNA (Chang and Lin, 2011), Protein (Dua and Graff, 2017), Splice (Dua and Graff, 2017); images for COIL-20 (Nene et al., 1996), Digits (Pedregosa et al., 2011), Mnist (Lecun et al., 1998); tabular data for DryBean (Dua and Graff, 2017), Mushrooms (Dua and Graff, 2017), Phishing (Chang and Lin, 2011), Rice (Dua and Graff, 2017), Svmguide1 (Chang and Lin, 2011); time series for HAR (Dua and Graff, 2017). More details can be found in Appendix B.1. All experimental results reported below are obtained over 9 different seeds.

Baselines. We conduct experiments with various pseudo-labeling policies: ERM, the supervised baseline that does not pseudo-label and is trained using labeled data only; $\text{PL}_{\theta=0.8}$ with the fixed threshold $\theta = 0.8$ (Lee, 2013); $\text{CSTA}_{\Delta=0.4}$ with the curriculum step $\Delta = 0.4$ (Cascante-Bonilla et al., 2021) and MSTA (Feofanov et al., 2019). More details about the policies and their implementation can be found in Appendix A.2 and B.3 respectively. Each baseline is evaluated with both the usual softmax prediction probability and our proposed $\mathcal{T}$-similarity. For the sake of simplicity, we denote softmax and $\mathcal{T}$-similarity in the result tables to distinguish those methods.

Labeling procedure. To generate SSL data, we consider the two following labeling strategies and compare the studied baselines in both cases (see more details in Appendix B.4):

1. IID: this is the case usually considered in classification tasks that verifies the i.i.d. assumption. The selection variable s is completely random, i.e., independent of $\mathbf{x}$ and y. In this case, we have $P(s = 1|\mathbf{x}, y) = P(s = 1)$ and thus $P_{\text{lab}}(\mathbf{x}, y) = P(\mathbf{x}, y)$.

2. SSB: in this case, s is dependent of $\mathbf{x}$ and y. For each class c and each data $\mathbf{x}$ with label $y = c$, we impose:

$$P(s = 1|\mathbf{x}, y = c) = \frac{1}{\beta} \exp(r \times |\text{proj}_1(\mathbf{x})|),$$

where $r > 0$ is a hyperparameter, $\text{proj}_1(\mathbf{x})$ is the projection value of $\mathbf{x}$ on the first principal component of the training data in class c and $\beta = \sum_{\mathbf{x}} \exp(r \times |\text{proj}_1(\mathbf{x})|)$ is a normalizing constant to ensure that $P(s = 1|\mathbf{x} \in \mathbf{X}_\ell, y = c) = 1$. The impact of the strength of the bias is studied in Appendix C.5.

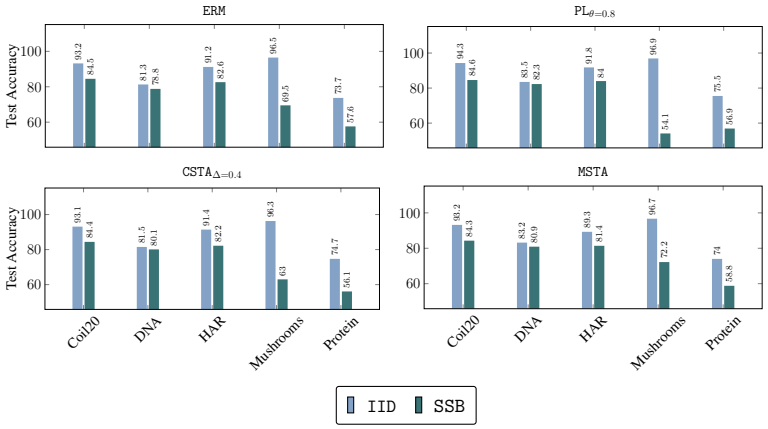


Figure 3: Test accuracies of the different baselines on 5 datasets. Full results are in Appendix C.1.

4.1 Failure of self-training with the softmax

We start by empirically illustrating that the performance of self-training is heavily dependent on the initial performance of the base classifier when the softmax prediction probabilities are used as a confidence measure. In Figure 3, we compare the performance of the base classifier ERM and the self-training methods under the two considered labeling procedures. One can see that all self-training methods together with their base classifier exhibit a drop in performance (in some cases, up to 30%) indicating that softmax predictions are not robust to the distribution shift. Of particular interest here is Mushrooms data set, where unlabeled data degrades the performance even further by 6.5% for $\text{CSTA}_{\Delta=0.4}$ and by 15.4% for $\text{PL}_{\theta=0.8}$ when we compare them to ERM. We also show a similar failure on model selection task in Appendix C.2. In what follows, we show that our proposal tackles these drawbacks inherent to other baselines.

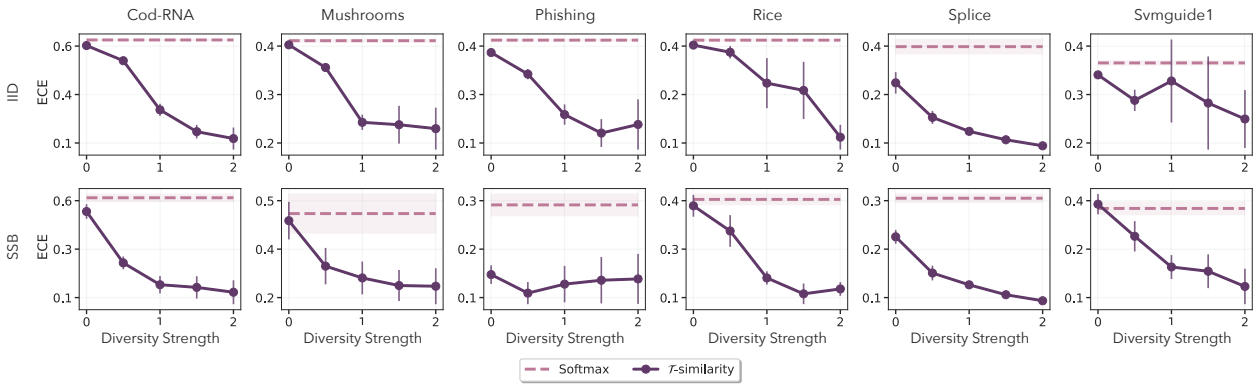
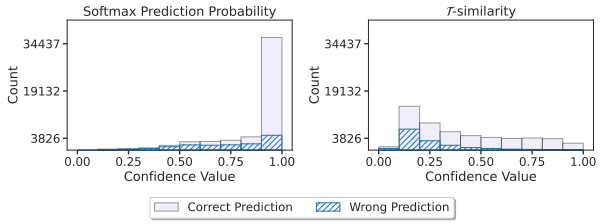
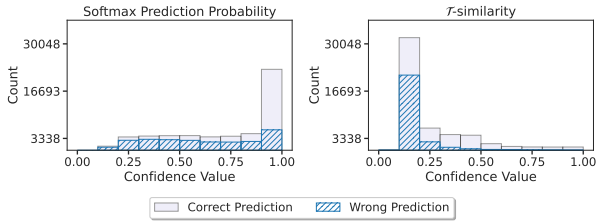


Figure 4: Increasing the diversity leads to a better-calibrated classifier in both IID and SSB settings.



(a) IID



(b) SSB

Figure 5: Softmax classifier is overconfident for both correctly and wrongly predicted samples. $s_{\mathcal{T}}$ assigns lower confidence to samples that are likely to be misclassified both in IID and SSB settings.

4.2 Ensemble diversity provides a calibrated confidence measure

In this section, we train the model on the labeled set and then compute $\mathcal{T}$-similarity on the unlabeled data to verify its advertised behavior both in confidence estimation and in calibration of the final model.

Correcting the overconfidence of the `softmax`. We plot the distributions of the confidence values for both `softmax` and $\mathcal{T}$-similarity for both accurate (Correct Prediction) and incorrect predictions (Wrong Prediction). We display the plots obtained on `Mnist` in Figure 5 (results for other datasets given in Appendix C.3). Our conclusion here is two-fold: **(1)** `softmax` is overconfident both in IID and SSB settings (high confidence leads to the highest error rate), while $\mathcal{T}$-similarity confidence is low when the model makes most of its mistakes and is confident when it reduces them to 0; **(2)** SSB setting degrades the confidence estimation with `softmax` degrades even further while $\mathcal{T}$-similarity remains robust to such a distribution shift. This vividly highlights that $\mathcal{T}$-similarity possesses the desired properties and behaves as expected.

Higher diversity improves calibration. In Figure 4, we compare the Expected Calibration Error (Naeini et al., 2015) obtained on the unlabeled set with the `softmax` and $\mathcal{T}$-similarity as a function of a varying regularization strength γ. We can see that when no diversity is imposed, i.e., $\gamma = 0$, the ECE obtained with both confidence measures is comparable. However, for any positive value of γ, the calibration error becomes smaller and smaller in most of the

Table 1: Classification performance of the different baselines on the datasets described in Table 2 when labeling is done with SSB. We display the average and the standard deviation of the test accuracy (both in %). The softmax corresponds to the usual self-training which uses the softmax prediction probability as a confidence estimate while the $\mathcal{T}$-similarity corresponds to our proposed method in Algorithm 1. For each baseline, the best result between softmax and $\mathcal{T}$-similarity is in **bold**.

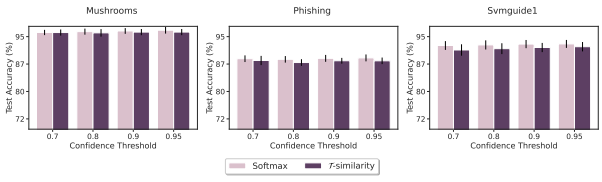
Dataset	ERM	$\text{PL}_{\theta=0.8}$		$\text{CSTA}_{\Delta=0.4}$		MSTA	
		softmax	$\mathcal{T}$-similarity	softmax	$\mathcal{T}$-similarity	softmax	$\mathcal{T}$-similarity
Cod-RNA	74.51 ± 8.86	74.75 ± 8.14	$\mathbf{80.06 \pm 3.55}$	73.39 ± 7.36	$\mathbf{78.39 \pm 4.66}$	75.28 ± 8.79	$\mathbf{76.88 \pm 7.67}$
COIL-20	84.54 ± 2.19	$\mathbf{84.69 \pm 3.56}$	84.57 ± 2.85	84.38 ± 3.05	$\mathbf{84.57 \pm 3.16}$	$\mathbf{84.32 \pm 2.34}$	84.07 ± 2.85
Digits	75.68 ± 4.59	$\mathbf{80.47 \pm 3.8}$	78.2 ± 3.34	78.4 ± 3.28	$\mathbf{79.14 \pm 3.5}$	78.02 ± 5.15	$\mathbf{79.8 \pm 5.92}$
DNA	78.82 ± 2.31	$\mathbf{80.29 \pm 2.24}$	79.06 ± 2.31	80.12 ± 2.08	$\mathbf{80.76 \pm 2.24}$	80.89 ± 2.64	$\mathbf{84.09 \pm 1.7}$
DryBean	64.6 ± 3.89	$\mathbf{65.6 \pm 4.18}$	61.55 ± 4.91	$\mathbf{64.91 \pm 3.72}$	64.6 ± 3.53	66.24 ± 4.31	$\mathbf{67.0 \pm 3.96}$
HAR	82.57 ± 1.96	82.87 ± 3.02	$\mathbf{83.12 \pm 2.27}$	82.19 ± 2.61	$\mathbf{83.53 \pm 3.77}$	$\mathbf{81.35 \pm 2.54}$	81.16 ± 1.63
Mnist	50.74 ± 2.25	51.08 ± 2.55	$\mathbf{52.69 \pm 2.42}$	51.7 ± 3.52	$\mathbf{54.26 \pm 1.82}$	51.6 ± 2.58	$\mathbf{54.18 \pm 2.34}$
Mushrooms	69.45 ± 7.29	59.53 ± 10.46	$\mathbf{71.36 \pm 6.63}$	62.98 ± 7.25	$\mathbf{77.55 \pm 7.65}$	72.16 ± 7.59	$\mathbf{76.16 \pm 13.04}$
Phishing	67.42 ± 3.55	66.08 ± 5.66	$\mathbf{77.41 \pm 3.93}$	66.88 ± 5.64	$\mathbf{76.17 \pm 8.58}$	69.48 ± 4.37	$\mathbf{75.83 \pm 7.52}$
Protein	57.57 ± 6.33	57.45 ± 6.36	$\mathbf{57.61 \pm 6.23}$	56.09 ± 5.61	$\mathbf{57.74 \pm 7.8}$	58.81 ± 6.54	$\mathbf{59.88 \pm 6.29}$
Rice	79.19 ± 5.12	80.54 ± 4.31	$\mathbf{81.1 \pm 4.28}$	79.88 ± 4.48	$\mathbf{81.56 \pm 3.61}$	80.35 ± 4.89	$\mathbf{82.63 \pm 5.63}$
Splice	66.13 ± 4.47	67.14 ± 2.62	$\mathbf{67.45 \pm 2.53}$	67.28 ± 2.07	$\mathbf{68.05 \pm 2.17}$	66.08 ± 4.98	$\mathbf{66.32 \pm 4.73}$
Svmguide1	70.89 ± 10.98	70.35 ± 11.74	$\mathbf{81.07 \pm 5.39}$	69.84 ± 11.06	$\mathbf{74.46 \pm 7.23}$	71.04 ± 11.11	$\mathbf{73.13 \pm 8.82}$

cases considered. This is equally true for both IID and SSB settings and backs up our claim about the robustness of $\mathcal{T}$-similarity to the distribution shift.

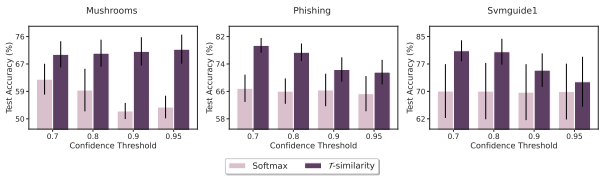
We now move to the evaluation of $\mathcal{T}$-similarity when used in self-training within the SSL+SSB paradigm.

4.3 Robust self-training with the $\mathcal{T}$-similarity

Improved performance under SSB. We now compare the softmax prediction probability and our $\mathcal{T}$-similarity under SSB. For each dataset, we display the average and the standard deviation of the test accuracy (both in %) in Table 1. For $\text{CSTA}_{\Delta=0.4}$ and MSTA, the $\mathcal{T}$-similarity leads to a substantial improvement on 11 of the 13 datasets. For $\text{PL}_{\theta=0.8}$, it improves the baseline on 8 of the 13 datasets. The obtained improvement is significant on Mushrooms and Phishing, where the softmax performs worse than ERM, the supervised baseline. For a fair evaluation of our method, we perform the same experiment as above when the labeling is done with IID in Appendix C.4. The obtained performance of our method is close to that of softmax suggesting that diversity can be safely promoted without any assumptions on the statistical relationship between the labeled and unlabeled data.



(a) IID.



(b) SSB.

Figure 6: Ablation study on the confidence threshold on Mushrooms and Phishing and Svmguide1. We display the average test accuracy (%) with a 95% confidence interval.

Sensitivity analysis. In Table 1 and Table 4, we display the results of the pseudo-labeling policy PL_θ with $\theta = 0.8$. To show that our obtained improvements are robust to the choice of confidence threshold θ, we study the performance

of self-training with the `softmax` and the $\mathcal{T}$-similarity on `Mushrooms`, `Phishing` and `Svmguide1`, under both the `IID` and `SSB` settings, when θ varies in $\{0.7, 0.8, 0.9, 0.95\}$. We present the results in Figure 6 and observe that for `IID` setting, the choice of the confidence level is not very important for both baselines considered. However, in `SSB` setting, it appears safer to choose a lower confidence level for pseudo-labeling as in most cases it leads to the best performance. Finally, and in accordance with Table 1, we note that our approach behaves much better under distribution shift compared to `softmax`.

Additional results. Due to the space constraints, we provide a study on the impact of the number of labeled examples per class n_ℓ in Appendix D.1. We also highlight the sensitivity of our method as a function of γ with respect to other pseudo-labeling strategies, and the number of classifiers M in Appendix D.2.

5 Conclusion

In this paper, we studied the effect of the sample selection bias on the performance of self-training methods with the semi-supervised learning framework. We showed that the conventional choice of `softmax` as a confidence measure degrades their performance regardless of the choice of the pseudo-labeling policy. To overcome this problem, we proposed a new $\mathcal{T}$-similarity measure that assigns high confidence to those unlabeled examples for which an ensemble of diverse linear classifiers $\mathcal{T}$ agrees in its predictions. Firstly, we empirically showed that the proposed confidence measure improves all the considered self-training methods in the case of biased labeling procedure. Secondly, we performed a theoretical analysis of the proposed similarity and found that the representation space plays an important role in the utility of the ensemble diversity. This suggests another direction of future work, where the representation could be jointly learned with the diverse ensemble as a part of a batch self-training architecture (Chen et al., 2022).

References

Ahern, T. P. (2018). Chapter Five - Pharmacoepidemiology in Pharmacogenetics. In Brøsen, K. and Damkier, P., editors, *Pharmacogenetics*, volume 83 of *Advances in Pharmacology*, pages 109–130. Academic Press.

Alves, W. M. (2006). CHAPTER 10 - Biostatistical Issues in Neuroemergency Clinical Trials. In Alves, W. M. and Skolnick, B. E., editors, *Handbook of Neuroemergency Clinical Trials*, pages 205–227. Academic Press, Burlington.

Amini, M.-R., Feofanov, V., Pauletto, L., Devijver, E., and Maximov, Y. (2023). Self-Training: A Survey.

Amini, M. R., Usunier, N., and Laviolette, F. (2008). A transductive bound for the voted classifier with an application to semi-supervised learning. *Advances in Neural Information Processing Systems*, 21.

Arias, F. D., Navarro, M., Elfanagely, Y., and Elfanagely, O. (2023). Chapter 31 - Biases in research studies. In Eltorai, A. E., Bakal, J. A., Newell, P. C., and Osband, A. J., editors, *Translational Surgery*, Handbook for Designing and Conducting Clinical and Translational Research, pages 191–194. Academic Press.

Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press.

Breiman, L. (2001). Random Forests. *Machine Learning*, 45(1):5–32.

Buschjäger, S., Pfahler, L., and Morik, K. (2020). Generalized negative correlation learning for deep ensembling.

Cascante-Bonilla, P., Tan, F., Qi, Y., and Ordonez, V. (2021). Curriculum labeling: Revisiting pseudo-labeling for semi-supervised learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 6912–6920.

Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):1–27.

Chapelle, O., Schölkopf, B., and Zien, A., editors (2006). *Semi-Supervised Learning*. The MIT Press.

Chen, B., Jiang, J., Wang, X., Wan, P., Wang, J., and Long, M. (2022). Debiased Self-Training for Semi-Supervised Learning. In *Advances in Neural Information Processing Systems*.

Chen, J., Liu, F., Avci, B., Wu, X., Liang, Y., and Jha, S. (2021). Detecting Errors and Estimating Accuracy on Unlabeled Data with Self-training Ensembles. *Advances in Neural Information Processing Systems*, 34.

de Mathelin, A., Deheeger, F., Richard, G., Mougeot, M., and Vayatis, N. (2021). Adapt: Correcting sample bias with transfer learning. https://adapt-python.github.io/adapt/examples/Sample_bias_example.html.

Dietterich, T. G. (2000). Ensemble methods in machine learning. In *Proceedings of the International Workshop on Multiple Classifier Systems*, MCS 2000, pages 1–15, Berlin, Heidelberg. Springer Berlin Heidelberg.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. (2021). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations*.

Dua, D. and Graff, C. (2017). UCI Machine Learning Repository.

Dudík, M., Phillips, S., and Schapire, R. E. (2005). Correcting sample selection bias in maximum entropy density estimation. In Weiss, Y., Schölkopf, B., and Platt, J., editors, *Advances in Neural Information Processing Systems*, volume 18. MIT Press.

Feofanov, V., Devijver, E., and Amini, M.-R. (2019). Transductive Bounds for the Multi-Class Majority Vote Classifier. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'19/IAAI'19/EAAI'19. AAAI Press.

Feofanov, V., Devijver, E., and Amini, M.-R. (2021). Multi-class probabilistic bounds for self-learning. *arXiv preprint arXiv:2109.14422*.

Feofanov, V., Tiomoko, M., and Virmaux, A. (2023). Random matrix analysis to balance between supervised and unsupervised learning under the low density separation assumption. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J., editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 10008–10033. PMLR.

Fergus, R., Weiss, Y., and Torralba, A. (2009). Semi-supervised learning in gigantic image collections. In Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C., and Culotta, A., editors, *Advances in Neural Information Processing Systems*, volume 22. Curran Associates, Inc.

Freund, Y. and Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680.

Grandvalet, Y. and Bengio, Y. (2004). Semi-supervised Learning by Entropy Minimization. In Saul, L., Weiss, Y., and Bottou, L., editors, *Advances in Neural Information Processing Systems*, volume 17. MIT Press.

Hansen, L. and Salamon, P. (1990). Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(10):993–1001.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep Residual Learning for Image Recognition. In *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '16, pages 770–778. IEEE.

Heckman, J. (1974). Shadow Prices, Market Wages, and Labor Supply. *Econometrica*, 42(4):679–694.

Heckman, J. (1990). Varieties of selection bias. *The American Economic Review*, 80(2):313–318.

Ho, T. K. (1998). The random subspace method for constructing decision forests. *IEEE transactions on pattern analysis and machine intelligence*, 20(8):832–844.

Huang, J., Gretton, A., Borgwardt, K., Schölkopf, B., and Smola, A. (2006). Correcting Sample Selection Bias by Unlabeled Data. In Schölkopf, B., Platt, J., and Hoffman, T., editors, *Advances in Neural Information Processing Systems*, volume 19. MIT Press.

Imran, A.-A.-Z., Huang, C., Tang, H., Fan, W., Xiao, Y., Hao, D., Qian, Z., and Terzopoulos, D. (2020). Partly supervised multi-task learning. In *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 769–774.

Jiang, Y., Nagarajan, V., Baek, C., and Kolter, J. Z. (2022). Assessing Generalization of SGD via Disagreement. In *International Conference on Learning Representations*.

Kingma, D. and Ba, J. (2015). Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations (ICLR)*, San Diega, CA, USA.

Kuncheva, L. I. (2004). *Combining Pattern Classifiers: Methods and Algorithms*. Wiley.

Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Lee, D.-H. (2013). Pseudo-Label : The Simple and Efficient Semi-Supervised Learning Method for Deep Neural Networks. *ICML 2013 Workshop : Challenges in Representation Learning (WREPL)*.

Lee, L.-F. (1982). Some approaches to the correction of selectivity bias. *The Review of Economic Studies*, 49(3):355–372.

Lin, Y., Lee, Y., and Wahba, G. (2002). Support Vector Machines for Classification in Nonstandard Situations. *Machine Learning*, 46(1):191–202.

Liu, Y. and Yao, X. (1999). Ensemble learning via negative correlation. *Neural Networks*, 12(10):1399–1404.

Lu, Z., Wu, X., Zhu, X., and Bongard, J. (2010). Ensemble Pruning via Individual Contribution Ordering. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, page 871–880, New York, NY, USA. Association for Computing Machinery.

Naeini, M. P., Cooper, G. F., and Hauskrecht, M. (2015). Obtaining well calibrated probabilities using bayesian binning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, page 2901–2907.

Nene, S. A., Nayar, S. K., and Murase, H. (1996). Columbia object image library (coil-20). Technical Report CUCS-005-96, Department of Computer Science, Columbia University.

Ortega, L. A., Cabañas, R., and Masegosa, A. (2022). Diversity and Generalization in Neural Network Ensembles . In Camps-Valls, G., Ruiz, F. J. R., and Valera, I., editors, *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pages 11720–11743. PMLR.

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

Quionero-Candela, J., Sugiyama, M., Schwaighofer, A., and Lawrence, N. D. (2009). *Dataset Shift in Machine Learning*. The MIT Press.

Shimodaira, H. (2000). Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90(2):227–244.

Sohn, K., Berthelot, D., Li, C.-L., Zhang, Z., Carlini, N., Cubuk, E. D., Kurakin, A., Zhang, H., and Raffel, C. (2020). Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Suykens, J. A. and Vandewalle, J. (1999). Least squares support vector machine classifiers. *Neural processing letters*, 9:293–300.

van Engelen, J. E. and Hoos, H. H. (2020). A survey on semi-supervised learning. *Machine Learning*, 109(2):373–440.

Wang, T. and Isola, P. (2020). Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *Proceedings of the 37th International Conference on Machine Learning*, pages 9929–9939.

Wei, H., Xie, R., Cheng, H., Feng, L., An, B., and Li, Y. (2022). Mitigating neural network overconfidence with logit normalization. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S., editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 23631–23644. PMLR.

Yarowsky, D. (1995). Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In *Proceedings of the 33rd Annual Meeting on Association for Computational Linguistics*, ACL '95, page 189–196, USA. Association for Computational Linguistics.

Zadrozny, B. (2004). Learning and Evaluating Classifiers under Sample Selection Bias. In *Proceedings of the Twenty-First International Conference on Machine Learning*, ICML '04, page 114, New York, NY, USA. Association for Computing Machinery.

Zhang, B., Wang, Y., Hou, W., Wu, H., Wang, J., Okumura, M., and Shinozaki, T. (2021). FlexMatch: Boosting Semi-Supervised Learning with Curriculum Pseudo Labeling. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems*.

Zhang, M.-L. and Zhou, Z.-H. (2013). Exploiting unlabeled data to enhance ensemble diversity. *Data Mining and Knowledge Discovery*, 26(1):98–129.

A Related work

A.1 Wrapper self-training

In Algorithm 1, we present a general pseudo-code for wrapper self-training algorithms that are based on different confidence estimators ϕ_h and pseudo-labeling policies ψ. For neural networks, the conventional choice of ϕ_h is the `softmax` function, so $\hat{\mathbf{P}}_u \in (\Delta_C)^{n_j}$.

Algorithm 1: Wrapper Self-Training

Input: Labeled training set $(\mathbf{X}_l, \mathbf{y}_\ell)$, unlabeled training set $\mathbf{X}_u$,
Base classifier h
Parameters: Pseudo-labeling policy ψ,
Confidence estimator ϕ_h based on h
Maximum number of iterations N
Initialization: Iteration $t = 1$
while $t \leq N$ *and* $\mathbf{X}_u \neq \emptyset$ **do**
1. Training
(Re-)Train the base classifier h on $(\mathbf{X}_\ell, \mathbf{y}_\ell)$
2. Confidence estimation
$\hat{\mathbf{P}}_u = \{\phi_h(\mathbf{x})\}_{\mathbf{x}\in\mathbf{X}_u}$
3. Pseudo-labeling
Determine subset $\mathbf{X}_{pl} \subset \mathbf{X}_u$ for pseudo-labeling $\mathbf{X}_{pl} = \psi(\mathbf{X}_u, \hat{\mathbf{P}}_u, t)$
Compute pseudo-labels $\hat{\mathbf{y}}_{pl} = \{\arg\max h(\mathbf{x})\}_{\mathbf{x}\in\mathbf{X}_{pl}}$
4. Update of training sets
$(\mathbf{X}_\ell, \mathbf{y}_\ell) \leftarrow (\mathbf{X}_\ell, \mathbf{y}_\ell) \cup (\mathbf{X}_{pl}, \hat{\mathbf{y}}_{pl})$
$\mathbf{X}_u \leftarrow \mathbf{X}_u \setminus \mathbf{X}_{pl}$
$t \leftarrow t + 1$
end
Output Final classifier h

A.2 Self-training policies

Fixed Threshold $\texttt{PL}_\theta$**.** It is the most standard policy (Yarowsky, 1995; Lee, 2013; Sohn et al., 2020) that fixes the threshold to a certain value θ. In the case of $\phi_h \equiv$ `softmax`, we have $[\hat{\mathbf{P}}_u]_j \in \Delta_C$, and then the pseudo-labeling policy outputs:

$$\psi(\mathbf{X}_u, \hat{\mathbf{P}}_u, t) = \{\mathbf{x}_j | \max_c [\hat{\mathbf{P}}_u]_{j,c} > \theta\}_{j=1}^{n_u}. \tag{5}$$

Curriculum $\texttt{CSTA}_\Delta$**.** We follow the implementation of Cascante-Bonilla et al. (2021) that finds a new threshold $\theta^{(t)}$ at every iteration t as the $(1 - t \cdot \Delta)$-th quantile of the distribution of the prediction confidence $\{\max_c [\hat{\mathbf{P}}_u]_{j,c}\}_{j=1}^{n_u}$, that is assumed to follow a Pareto distribution. Then, the final policy is Eq. (5), where θ is replaced by $\theta^{(t)}$.

Transductive policy `MSTA` This policy is based on the upper-bound of the transductive error on the unlabeled examples that have a confidence score larger than a threshold $\boldsymbol{\theta}$, denoted by $R_{u,\geq\boldsymbol{\theta}}$ (Amini et al., 2008). We use the multi-class implementation of Feofanov et al. (2019) that employs a threshold vector $\boldsymbol{\theta}^{(t)} = \left(\theta_c^{(t)}\right)_{c=1}^{C}$, where $\theta_c^{(t)}$ is a threshold for class c at iteration t:

$$\psi(\mathbf{X}_u, \hat{\mathbf{P}}_u, t) = \left\{\mathbf{x}_j | [\hat{\mathbf{P}}_u]_{j,\hat{y}_j} > \theta_{\hat{y}_j}^{(t)}\right\}_{j=1}^{n_u},$$

where $\hat{y}_j = \arg\max_c [\hat{\mathbf{P}}_u]_{j,c}$ is the prediction for $\mathbf{x}_j$. The threshold is found by solving the following minimization problem:

$$\boldsymbol{\theta}^{(t)} = \underset{\boldsymbol{\theta}\in[0,1]^C}{\arg\min} \frac{R_{u,\geq\boldsymbol{\theta}}}{(1/n_u)\sum_{j=1}^{n_u} \mathbb{I}([\hat{\mathbf{P}}_u]_{j,\hat{y}_j} > \theta_{\hat{y}_j})},$$

where $\mathbb{I}$ denotes the indicator function. Thus, the threshold at each iteration is chosen by minimizing the ratio between the upper bound on the error and the number of examples to be pseudo-labeled.

B Experimental Setup

B.1 Datasets

In all experiments, the only pre-processing step is to standardize the features. Table 2 sums up the characteristics of the datasets used in our experiments and the corresponding values of hyperparameter r used in the SSB labeling procedure (Algorithm 2). We considered 13 publicly available SSL datasets with various data modalities:

- Biological data for Cod-RNA (Chang and Lin, 2011), DNA (Chang and Lin, 2011), Protein (Dua and Graff, 2017), Splice (Dua and Graff, 2017)
- Images for COIL-20 (Nene et al., 1996), Digits (Pedregosa et al., 2011), Mnist (Lecun et al., 1998)
- Tabular data for DryBean (Dua and Graff, 2017), Mushrooms (Dua and Graff, 2017), Phishing (Chang and Lin, 2011), Rice (Dua and Graff, 2017), Svmguide1 (Chang and Lin, 2011)
- Time series for HAR (Dua and Graff, 2017)

Table 2: Characteristics of the datasets and corresponding values of hyperparameter r.

Dataset	Size	# of lab. examples n_ℓ	Dimension d	# classes C	SSB hyperparameter r
Cod-RNA	59535	99	8	2	2
COIL-20	1440	200	1024	20	0.33
Digits	1797	99	64	10	0.5
DNA	3186	149	180	6	25
DryBean	13543	104	16	7	2
HAR	10299	299	561	3	0.33
Mnist	70000	100	784	10	0.33
Mushrooms	8124	79	112	2	2
Phishing	11055	99	68	2	2
Protein	1080	80	77	8	0.6
Rice	3810	29	7	2	2
Splice	3175	39	20	2	2
Svgmguide1	3089	39	4	2	2

B.2 Architecture and training parameters

For all of our experiments, we train a 3-layer MLP, with the Adam optimizer (Kingma and Ba, 2015) and a learning rate of 0.001. Training is performed during 5 epochs with 100 training iterations per epoch. We evaluate the model on a test set of size 25% of the dataset. For each dataset, we perform the experiment with 9 different seeds and display the average and the standard deviation of the test accuracy (both in $\%$) over the 9 trials.

B.3 Baselines

We performed experiments with three different pseudo-labeling policies: PL, CSTA, and MSTA. For a fair comparison of the methods, we tested manually different values of hyperparameters and kept those that gave good results on average, namely, $\theta = 0.8$ for PL and $\Delta = 0.4$ for CSTA. The maximum number of self-training iterations N is set to 5.

Implementation of MSTA In the original implementation of (Feofanov et al., 2019), the authors estimate the posterior probability $P(y|\mathbf{x})$ by the votes of the majority vote classifier. To apply the policy MSTA in Algorithm 1, the posterior probabilities in the upper-bound expression has to be estimated, and they have empirically shown that the prediction probabilities given by the supervised baseline allows to have a good proxy for the bound (Feofanov et al., 2021). We have tested several ways to estimate these probabilities for our network (majority vote of the ensemble, using $\mathcal{T}$-similarities), and the use of predicted probabilities by the prediction head gives the most stable results.

B.4 More details on the labeling procedure

Labeling procedure. A classical strategy in SSL benchmarks is to use an i.i.d. sampling to select the same number of labeled data in each class. It can be achieved by applying i.i.d. sampling in a class-wise manner. In our work, we

refer to this labeling procedure as IID. Inspired by Huang et al. (2006); Zadrozny (2004), we consider another strategy that simulates sample selection bias to select training examples in each class. We make sure that the original class proportion is preserved. In our work, we refer to this labeling procedure as SSB. The pseudo-code of the SSB labeling procedure is outlined in Algorithm 2 (values of r are given in the last column of Table 2).

Visualization. As a picture is worth a thousand words, we illustrate the sample selection bias on the Mushrooms dataset (Dua and Graff, 2017). We select 80 labeled examples out of the 6093 training examples with the IID and the SSB procedures. Inspired by (de Mathelin et al., 2021), we plot in the first row of Figure 7 the distribution of the projection values on the first principal component (PC1) of the labeled training set (blue) and of the whole training set (orange). Contrary to the IID sampling, we can see that the SSB sampling injects a clear bias between distributions. We also visualize in the second row of Figure 7 the projection values on the first two principal components (PC1 and PC2) of the labeled training examples (blue) and of all the training samples (orange). We can see that the IID procedure samples in all regions of the space while the SSB concentrates in specific areas.

Algorithm 2: SSB Labeling Procedure

Input: Training examples $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_m)\}$, class proportions $\{p_1, \ldots, p_C\}$
Parameters: number of training examples to label n_ℓ, hyperparameter $r > 0$.
Initialize labeled training set $(\mathbf{X}_\ell, \mathbf{y}_\ell) = \emptyset$
for $c \in [C]$ **do**
  $\mathcal{T}_c = \{(\mathbf{x}_i, y_i) | y_i = c\}$ set of training examples of class c
  Compute projection values
  Apply PCA on features of $\mathcal{T}_c$
  Recover for each $\mathbf{x}_i$ the projection value on the first principal component $\text{proj}_1(\mathbf{x}_i)$
  Compute $\beta = \sum_{\mathbf{x}_i} \exp(r \times |\text{proj}_1(\mathbf{x}_i)|)$
  Draw without replacement in $\mathcal{T}_k$
  while $|\mathbf{X}_\ell| < p_k n_\ell$ **do**
    Draw $\mathbf{x}_i$ with probability $P(s = 1 | \mathbf{x}_i, y_i = c) = \frac{1}{\beta} \exp(r \times |\text{proj}_1(\mathbf{x}_i)|)$
    $\mathbf{X}_\ell \leftarrow \mathbf{X}_\ell \cup \{(\mathbf{x}_i, y_i)\}$
  end
end
Create the unlabeled training set
$\mathbf{X}_u = \{\mathbf{x}_j | \mathbf{x}_j \notin \mathbf{X}_\ell\}$
Output labeled training set $(\mathbf{X}_\ell, \mathbf{y}_\ell)$, unlabeled training set $\mathbf{X}_u$

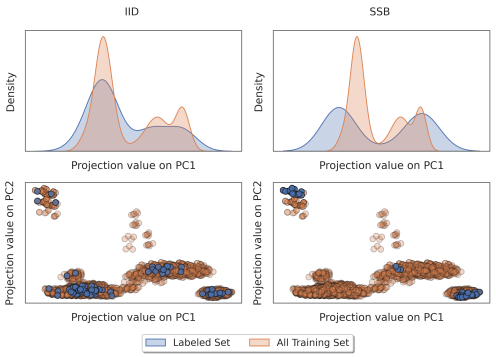


Figure 7: Visualization of sample selection bias on Mushrooms. **First row:** Distribution of the projection values on the first principal component (PC1). **Second row:** Visualization of the projection values on the PC1 and the PC2.

C Additional Experiments

In this section, we provide additional experiments and sensitivity analysis.

C.1 Failure cases of self-training

The classification performance of the self-training algorithms is shown in Table 3. For each method, on all datasets, we observe a huge drop in performance when sample selection bias is applied (labeling with SSB) compared to when it is not (labeling with IID). Self-training even turns out to be harmful in some situations, for instance on Mushrooms, Phishing, or Protein. These results confirm the fact that a biased confidence measure, due to the sample selection bias, diminishes the quality of the pseudo-labeling and in some cases, the benefit of using unlabeled data becomes a disadvantage. It should be noted that the performance of the supervised baselines ERM also declines when sample selection bias is applied.

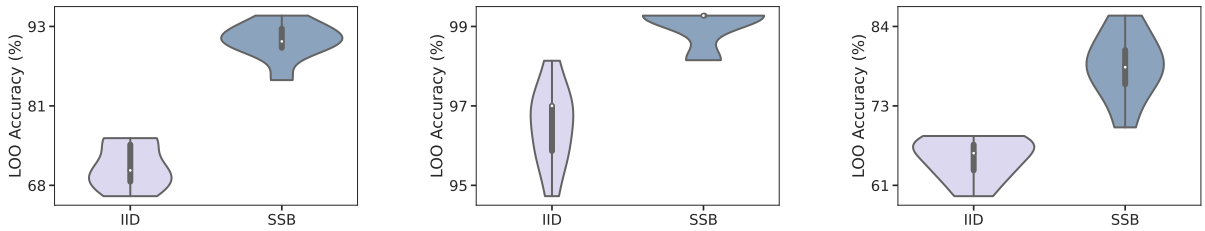
C.2 Unreliable model selection

To further motivate the importance of this framework, we consider a model selection task. A traditional way to find a good set of hyperparameters is to do cross-validation, that is to evaluate the model on a separate validation set. In semi-supervised learning, as labeled data are scarce, one can do leave-one-out instead: each labeled example is used as a test set while training is performed on the remaining samples. We will refer to the corresponding average test accuracy as the leave-one-out accuracy. In the context of model selection, the chosen set of hyperparameters would be the one giving the best leave-one-out accuracy. We propose to compare the leave-one-out accuracy when sample selection bias is applied (labeling with SSB) and when it is not (labeling with IID). We perform this experiment on Mushrooms, Protein and Mnist using the same architectures and training parameters as before. On each dataset, we

Table 3: Classification performance of the different baselines on the datasets described in Table 2. We display the average and the standard deviation of the test accuracy (both in %) over the 9 trials. For each baseline, the best result between IID and SSB is in **bold**.

Dataset	Bias	Baselines			
		ERM	$\text{PL}_{\theta=0.8}$	$\text{CSTA}_{\Delta=0.4}$	MSTA
Cod-RNA	IID	89.28 ± 2.13	89.91 ± 2.03	89.09 ± 2.37	89.89 ± 1.89
	SSB	74.51 ± 8.86	74.21 ± 7.76	73.39 ± 7.36	75.28 ± 8.79
COIL-20	IID	93.18 ± 1.5	94.32 ± 1.13	93.09 ± 1.73	93.21 ± 1.57
	SSB	84.54 ± 2.19	84.6 ± 3.86	84.38 ± 3.05	84.32 ± 2.34
Digits	IID	81.38 ± 2.45	84.27 ± 2.98	81.78 ± 2.51	83.04 ± 2.13
	SSB	75.68 ± 4.59	80.86 ± 4.11	78.4 ± 3.28	78.02 ± 5.15
DNA	IID	81.28 ± 2.27	83.45 ± 2.01	81.54 ± 2.44	83.19 ± 1.96
	SSB	78.82 ± 2.31	82.28 ± 2.5	80.12 ± 2.08	80.89 ± 2.64
DryBean	IID	86.85 ± 1.68	88.02 ± 1.49	86.98 ± 1.61	87.72 ± 1.54
	SSB	64.6 ± 3.89	66.12 ± 4.35	64.91 ± 3.72	66.24 ± 4.31
HAR	IID	91.16 ± 0.54	91.82 ± 0.4	91.36 ± 0.24	89.29 ± 1.24
	SSB	82.57 ± 1.96	84.02 ± 2.61	82.19 ± 2.61	81.35 ± 2.54
Mnist	IID	73.98 ± 1.46	75.37 ± 1.57	75.24 ± 1.48	74.6 ± 1.76
	SSB	50.74 ± 2.25	51.07 ± 2.2	51.7 ± 3.52	51.6 ± 2.58
Mushrooms	IID	96.48 ± 1.57	96.94 ± 1.4	96.3 ± 1.32	96.68 ± 1.31
	SSB	69.45 ± 7.29	54.08 ± 5.56	62.98 ± 7.25	72.16 ± 7.59
Phishing	IID	88.51 ± 1.51	89.41 ± 1.44	88.96 ± 1.37	88.82 ± 1.7
	SSB	67.42 ± 3.55	65.34 ± 7.86	66.88 ± 5.64	69.48 ± 4.37
Protein	IID	73.74 ± 4.78	75.51 ± 2.83	74.73 ± 3.01	73.99 ± 5.6
	SSB	57.57 ± 6.33	56.87 ± 5.79	56.09 ± 5.61	58.81 ± 6.54
Rice	IID	88.24 ± 3.63	88.5 ± 3.39	88.15 ± 3.46	88.69 ± 3.49
	SSB	79.19 ± 5.12	80.87 ± 4.43	79.88 ± 4.48	80.35 ± 4.89
Splice	IID	69.09 ± 4.09	70.64 ± 4.52	70.46 ± 4.32	69.65 ± 4.27
	SSB	66.13 ± 4.47	67.02 ± 2.11	67.28 ± 2.07	66.08 ± 4.98
Svmguide1	IID	93.01 ± 1.63	93.22 ± 1.66	92.77 ± 1.77	93.4 ± 1.21
	SSB	70.89 ± 10.98	70.22 ± 11.64	69.84 ± 11.06	71.04 ± 11.11

repeat the experiment with 9 different seeds and display the distribution of the leave-one-out accuracy over the 9 trials in Figure 8. We can see that on all datasets, the leave-one-out accuracy under SSB is always better than with IID. The analysis of these results is two-fold: **(1)** it empirically demonstrates the fact that the failure of self-training methods is due to a biased confidence measure and not to a potentially uninformative labeled set. **(2)** it highlights the risk of performing cross-validation in semi-supervised learning: the chosen set of hyperparameters corresponds to the highest leave-one-out accuracy, but under sample selection bias, the real generalization performance of the model is low as shown in Table 3.



(a) Mnist. (b) Mushrooms. (c) Protein.

Figure 8: Leave-one-out on the labeled set of Mnist, Mushrooms and Protein. We display the distribution of the leave-one-out accuracy (%) over the 9 trials.

C.3 Correcting the `softmax` overconfidence

As in the main paper, for this experiment, we artificially use the true labels on $\mathbf{X}_u$ to compute the distribution of $s_{\mathcal{T}}$ on the examples for which the prediction of the model is accurate (Correct Prediction) and on the examples for which the prediction is incorrect (Wrong Prediction). We compute the `softmax` prediction probabilities and the $\mathcal{T}$-similarity values on the unlabeled examples. Again, we artificially use the true labels to plot the distribution of the confidence value (`softmax` or $\mathcal{T}$-similarity) on the examples for which the prediction is accurate (Correct Prediction) and on the examples for which the prediction is incorrect (Wrong Prediction). We display the plots obtained in Figure 9 for various datasets. We can see that the obtained results are very similar to those presented in the main paper and our conclusions about them hold.

C.4 The $\mathcal{T}$-similarity is comparable to the `softmax` when there is no sample selection bias

We perform the same experiment as in Table 1 but when the labeling is done with `IID`. The obtained results are in Table 4. It should be noted that in this setting, even if the `softmax` is subject to overconfidence, the pseudo-labeling can still be of good quality as the distributions of labeled and unlabeled samples do not differ. We can observe that the $\mathcal{T}$-similarity induces a slight decrease in performance on most datasets. However, it manages to remain competitive, notably for $\texttt{CSTA}_{\Delta=0.4}$ and `MSTA`, where it even manages to be better or similar on 4 datasets out of 13.

Table 4: Classification performance of the different baselines on the datasets described in Table 2 when labeling is done with `IID`. We display the average and the standard deviation of the test accuracy (both in %) over the 9 trials. The `softmax` corresponds to the usual self-training which uses the `softmax` prediction probability as a confidence estimate while the $\mathcal{T}$-similarity corresponds to our proposed method in Algorithm 1. For each baseline, the best result between `softmax` and $\mathcal{T}$-similarity is in **bold**.

Dataset	ERM	$\texttt{PL}_{\theta=0.8}$		$\texttt{CSTA}_{\Delta=0.4}$		`MSTA`	
		`softmax`	$\mathcal{T}$-similarity	`softmax`	$\mathcal{T}$-similarity	`softmax`	$\mathcal{T}$-similarity
`Cod-RNA`	89.28 ± 2.13	$\mathbf{89.34 \pm 2.44}$	84.87 ± 3.54	$\mathbf{89.09 \pm 2.37}$	87.85 ± 2.24	$\mathbf{89.89 \pm 1.89}$	89.65 ± 2.19
`COIL-20`	93.18 ± 1.5	$\mathbf{93.77 \pm 1.19}$	93.49 ± 1.97	93.09 ± 1.73	$\mathbf{93.3 \pm 1.77}$	93.21 ± 1.57	$\mathbf{94.17 \pm 1.99}$
`Digits`	81.38 ± 2.45	$\mathbf{83.58 \pm 3.26}$	81.36 ± 2.75	81.78 ± 2.51	$\mathbf{81.88 \pm 3.02}$	$\mathbf{83.04 \pm 2.13}$	82.62 ± 3.03
`DNA`	81.28 ± 2.27	$\mathbf{81.25 \pm 2.5}$	79.49 ± 2.53	81.54 ± 2.44	$\mathbf{81.64 \pm 2.3}$	83.19 ± 1.96	$\mathbf{84.72 \pm 2.3}$
`DryBean`	86.85 ± 1.68	$\mathbf{87.59 \pm 1.6}$	86.6 ± 1.85	$\mathbf{86.98 \pm 1.61}$	86.74 ± 1.75	$\mathbf{87.72 \pm 1.54}$	87.35 ± 2.19
`HAR`	91.16 ± 0.54	91.24 ± 1.03	$\mathbf{91.24 \pm 0.38}$	$\mathbf{91.36 \pm 0.24}$	91.29 ± 0.39	$\mathbf{89.29 \pm 1.24}$	89.25 ± 1.26
`Mnist`	73.98 ± 1.46	$\mathbf{74.61 \pm 1.85}$	72.16 ± 3.1	$\mathbf{75.24 \pm 1.48}$	73.42 ± 1.6	$\mathbf{74.6 \pm 1.76}$	73.36 ± 1.33
`Mushrooms`	96.48 ± 1.57	$\mathbf{96.56 \pm 1.26}$	96.23 ± 1.57	$\mathbf{96.3 \pm 1.32}$	96.25 ± 1.38	$\mathbf{96.68 \pm 1.31}$	96.44 ± 1.33
`Phishing`	88.51 ± 1.51	$\mathbf{89.02 \pm 1.37}$	88.15 ± 1.4	$\mathbf{88.96 \pm 1.37}$	88.82 ± 1.06	88.82 ± 1.7	$\mathbf{88.94 \pm 1.85}$
`Protein`	73.74 ± 4.78	74.86 ± 3.48	$\mathbf{75.43 \pm 2.65}$	74.73 ± 3.01	$\mathbf{75.68 \pm 2.93}$	73.99 ± 5.6	$\mathbf{75.1 \pm 4.98}$
`Rice`	88.24 ± 3.63	$\mathbf{88.34 \pm 3.18}$	88.32 ± 3.29	88.15 ± 3.46	$\mathbf{88.7 \pm 2.92}$	88.69 ± 3.49	$\mathbf{89.75 \pm 2.1}$
`Splice`	69.09 ± 4.09	$\mathbf{70.26 \pm 4.08}$	69.65 ± 3.94	$\mathbf{70.46 \pm 4.32}$	70.32 ± 4.09	69.65 ± 4.27	$\mathbf{70.51 \pm 4.26}$
`Svmguide1`	93.01 ± 1.63	$\mathbf{92.94 \pm 1.91}$	91.92 ± 2.26	$\mathbf{92.77 \pm 1.77}$	92.31 ± 2.04	$\mathbf{93.4 \pm 1.21}$	92.83 ± 1.48

C.5 Robustness to the strength of the bias

In this experiment, we study the robustness of our method to the strength of the bias. More specifically, we make the labeling procedure gradually evolve from `IID` to `SSB`. We consider for each class c and for each data $\mathbf{x}$ with label $y = c$

$$P_\alpha(s = 1|\mathbf{x}, y = c) = (1 - \alpha) \cdot \underbrace{\frac{1}{\text{card}(\mathcal{T}_c)}}_{\texttt{IID}} + \alpha \cdot \underbrace{\frac{1}{\beta} \exp(r \times |\text{proj}_1(\mathbf{x})|)}_{\texttt{SSB}},$$

where $\alpha \in [0, 1]$ is the interpolation coefficient and $\mathcal{T}_c = \{(\mathbf{x}, y)|y = c\}$ is the set of training data of class c. Increasing α corresponds to imposing more sample selection bias in the labeling procedure. In Figure 10, we display the test accuracy on `Mushrooms` when the labeling procedure is done with P_α, where α varies in $[0, 1]$. We note that, up to some point, increasing α does not lead to a strong distribution mismatch, and both the `softmax` prediction probabilities and the proposed $\mathcal{T}$-similarity give similar results. However, when the value of α exceeds 0.8, test accuracy suffers from a sharp drop for both methods, with a more drastic decrease in the case of `softmax`. We empirically demonstrate that our proposed $\mathcal{T}$-similarity is more robust to distribution shift than the `softmax`.

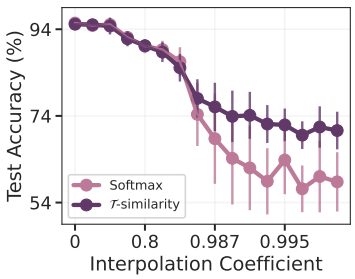
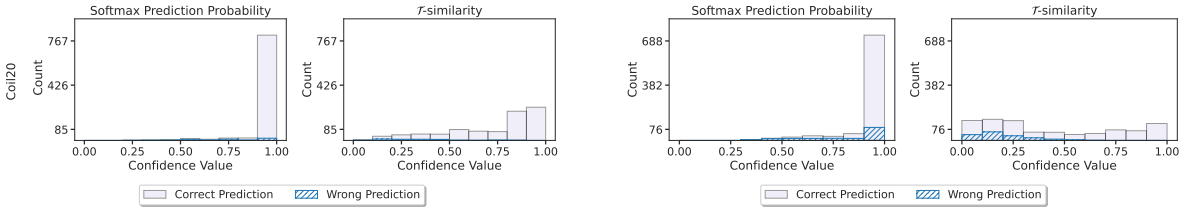
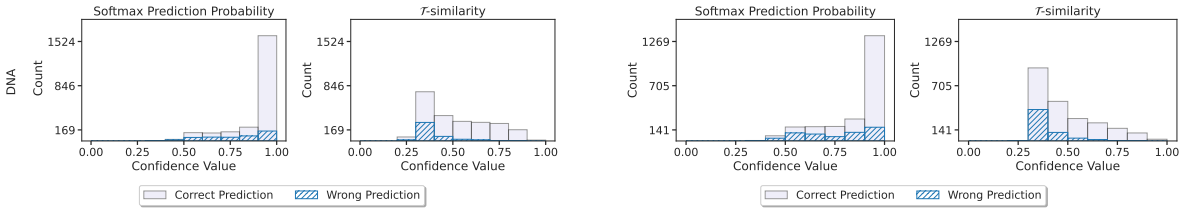


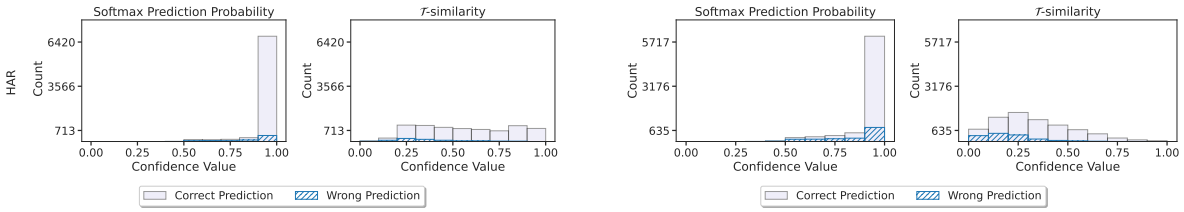
Figure 10: The $\mathcal{T}$-similarity is more robust to distribution shift than `softmax`.



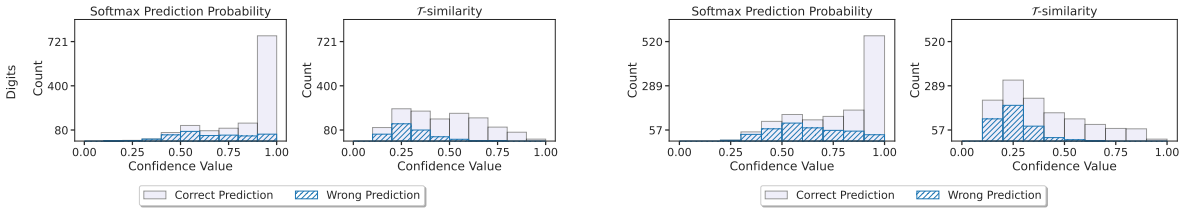
(a) IID (b) SSB



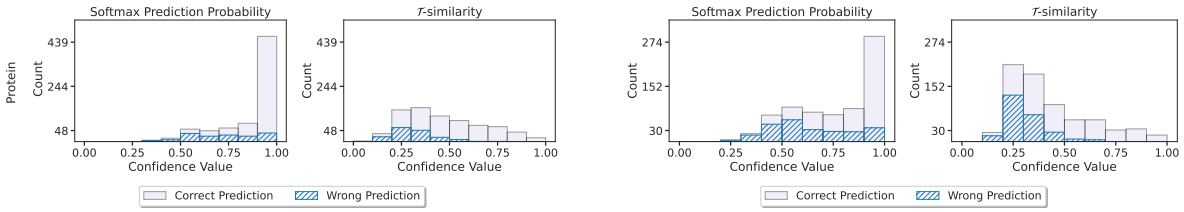
(c) IID (d) SSB



(e) IID (f) SSB



(g) IID (h) SSB



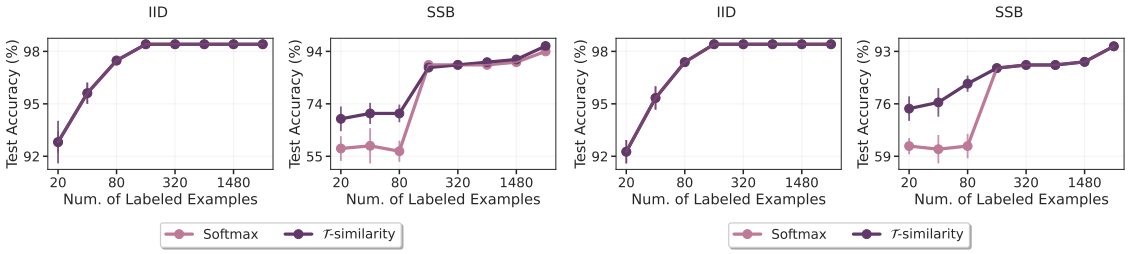
(i) IID (j) SSB

Figure 9: We display the distribution over the `softmax` and the $\mathcal{T}$-similarity on the unlabeled examples correctly classified by the base classifier and on the unlabeled examples mis-classified by the base classifier.

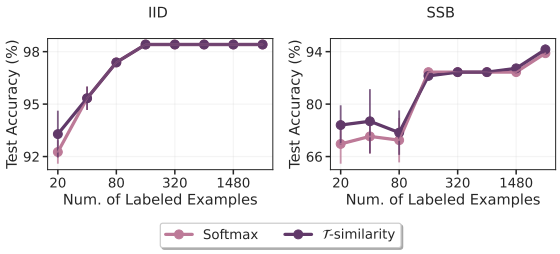
D Ablation study and sensitivity analysis

D.1 Ablation study with different number of labeled examples

Intuitively, when the number of labeled examples increases, the impact of the SSB labeling procedure should decrease as the base classifier has more labeled data to learn from and hence relies less on the unlabeled data. To study the robustness of our method when the impact of sample selection bias decreases, we observe the performance of each pseudo-labeling policy with the `softmax` and the $\mathcal{T}$-similarity when the number of labeled data per class n_ℓ increases from 20 to 2000. We display the results in Figure 11. We observe that in IID setting, the two methods behave similarly and the performance gradually increases with the number of available labeled points. However, in SSB setting, we see that $\mathcal{T}$-similarity leads to much better results for realistic scenarios commonly considered in SSL when the number of samples is small. This improvement is consistent across all pseudo-labeling policies considered.



(a) $\text{PL}_{\theta=0.8}$ (b) $\text{CSTA}_{\Delta=0.4}$

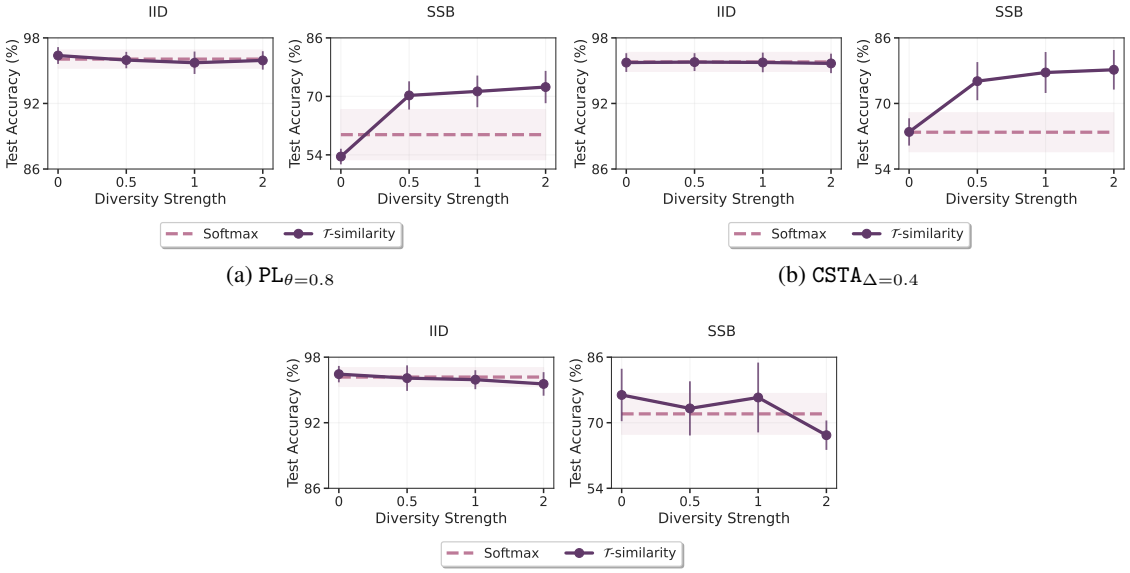


(c) MSTA

Figure 11: Ablation study on the number of labeled examples per class n_ℓ on `Mushrooms` for the three pseudo-labeling policies.

D.2 Sensitivity to hyperparameters

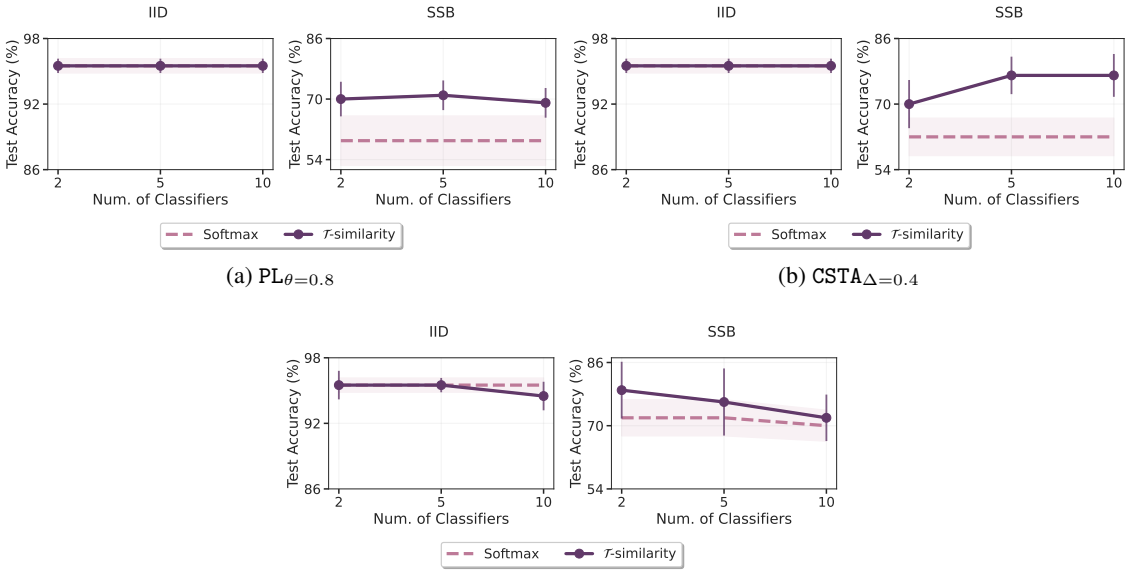
We conduct a sensitivity analysis on `Mushrooms` to see how self-training with the proposed $\mathcal{T}$-similarity behaves under different choices of hyperparameters. For each pseudo-labeling policy, we make the diversity strength γ vary in $\{0, 0.5, 1, 1.5, 2\}$ and display the results in Figure 12. We observe that in SSB setting all pseudo-labeling policies benefit from the diversity almost for all positive values of γ. Similarly, we see that in IID, the diversity does not hurt the performance. In the same fashion, we make the number of classifiers M vary in $\{2, 5, 10\}$ and display the results Figure 13. The behavior is similar to the previous experiment and is consistent with our observations made so far.



(a) $\text{PL}_{\theta=0.8}$ (b) $\texttt{CSTA}_{\Delta=0.4}$

(c) MSTA

Figure 12: Sensitivity analysis of the diversity strength parameter γ on `Mushrooms` for the three pseudo-labeling policies.



(a) $\text{PL}_{\theta=0.8}$ (b) $\texttt{CSTA}_{\Delta=0.4}$

(c) MSTA

Figure 13: Sensitivity analysis of the number of classifiers M on `Mushrooms` for the three pseudo-labeling policy. We display the average and standard deviation of the test accuracy over 9 seeds.

E Proofs

In this section, we detail the proofs of our theoretical results. We use the notations introduced in Section 3 and, for ease of notation, we define the following quantities:

$$\mathbf{S}_u = \frac{\gamma(M+1)}{n_u(M-1)}\mathbf{X}_u^\top \mathbf{X}_u \text{ and } \alpha_u = \frac{\gamma}{2n_u(M-1)}. \tag{6}$$

E.1 Proof of Proposition 3.2

The proof of Proposition 3.2 is detailed below.

Proof. For ease of notation, we will denote h_m^c the c-th entry of $h_m(\mathbf{x})$. Using the fact that each $h_m(\mathbf{x})$ is in the simplex Δ_C, we have that

$$0 \leq s_{\mathcal{T}}(\mathbf{x}) = \frac{1}{M(M-1)}\sum_{m\neq k} h_m(\mathbf{x})^\top h_k(\mathbf{x}) = \frac{1}{M(M-1)}\sum_{m\neq k}\sum_{c=1}^{C} h_m^c \underbrace{h_k^c}_{\leq 1} \leq \frac{1}{M(M-1)}\sum_{m\neq k}\sum_{c=1}^{C} h_m^c = 1.$$

□

E.2 Stationary points of $\mathcal{L}$ are solutions of a linear problem

In this section, we show that solving (3) amounts to solving the following linear problem in $\mathbf{W}$

$$\left(\Lambda + \frac{\mathbf{X}_\ell^\top \mathbf{X}_\ell}{n_\ell}\right)\mathbf{W} + \frac{\gamma}{n_u(M-1)}\mathbf{X}_u^\top \mathbf{X}_u \mathbf{W}\left(\mathbf{U}^{[M]} - \mathbf{I}_M\right) = \frac{\mathbf{X}_\ell^\top \mathbf{Y}}{n_\ell}, \tag{7}$$

where $\mathbf{Y} \in \mathbb{R}^{n_\ell \times M}$ is the matrix with the vector $\mathbf{y}_\ell$ repeated M times as columns and $\mathbf{\Lambda} \in \mathbb{R}^{M\times M}$ is a diagonal matrix with entries λ_m. Moreover, denoting $\boldsymbol{\omega}_m$ the columns of $\mathbf{W}$, (7) is equivalent to

$$\forall m \in [\![1, M]\!], \quad \left(\lambda_m \mathbf{I}_d + \frac{\mathbf{X}_\ell^\top \mathbf{X}_\ell}{n_\ell}\right)\boldsymbol{\omega}_m = \frac{\mathbf{X}_\ell^\top \mathbf{y}_\ell}{n_\ell} - \frac{\gamma}{n_u(M-1)}\mathbf{X}_u^\top \mathbf{X}_u \sum_{k=1|k\neq m}^{M} \boldsymbol{\omega}_k. \tag{8}$$

The proof is detailed below.

Proof. We first recall that stationary points of $\mathcal{L}$ verify the Euler equation

$$\nabla\mathcal{L}(\boldsymbol{W}) = 0. \tag{9}$$

To compute $\nabla\mathcal{L}\colon \mathbb{R}^{d\times M} \to \mathbb{R}^{d\times M}$, we will rewrite the loss function $\mathcal{L}$ using the Frobenius inner product, that is the usual inner product on matrix spaces. We will use this formulation to write its Taylor expansion of order 1 and identify $\nabla\mathcal{L}$. Using the formulation of (**P**) and the notations introduced in (6), we have:

$$\begin{aligned}
\mathcal{L}(\mathbf{W}) &= \frac{1}{M}\sum_{m=1}^{M}\frac{1}{n_\ell}\sum_{i=1}^{n_\ell}\left(y_i - \boldsymbol{\omega}_m^\top \mathbf{x}_i\right)^2 + \frac{1}{M}\sum_{m=1}^{M}\lambda_m\|\boldsymbol{\omega}_m\|_2^2 + \frac{2\alpha_u}{M}\sum_{m\neq k}\sum_{i=n_\ell+1}^{n_\ell+n_u} w_k^\top \mathbf{x}_i w_\ell^\top \mathbf{x}_i \\
&= \frac{1}{n_\ell M}\sum_{m=1}^{M}\|\mathbf{y}_\ell - \mathbf{X}_\ell\boldsymbol{\omega}_m\|_2^2 + \frac{1}{M}\sum_{m=1}^{M}\lambda_m\|\boldsymbol{\omega}_m\|_2^2 + \frac{2\alpha_u}{M}\sum_{m\neq k}\left(\mathbf{X}_u w_m\right)^\top\left(\mathbf{X}_u w_k\right) \\
&= \frac{1}{n_\ell M}\sum_{m=1}^{M}\left(\mathbf{y}_\ell - \mathbf{X}_\ell\boldsymbol{\omega}_m\right)^\top\left(\mathbf{y}_\ell - \mathbf{X}_\ell\boldsymbol{\omega}_m\|\right) + \frac{1}{M}\sum_{m=1}^{M}\left(\sqrt{\lambda_m}\boldsymbol{\omega}_m\right)^\top\left(\sqrt{\lambda_m}\boldsymbol{\omega}_m\right) \\
&\quad + \frac{2\alpha_u}{M}\sum_{m=1}^{M}\sum_{k=1}^{M}\left(\mathbf{X}_u w_m\right)^\top\left(\mathbf{X}_u w_k\right) - 2\alpha_u\sum_{m=1}^{M}\left(\mathbf{X}_u w_m\right)^\top\left(\mathbf{X}_u w_m\right) \\
&= \frac{1}{n_\ell M}\|\mathbf{Y} - \mathbf{X}_\ell\mathbf{W}\|_{\mathrm{F}}^2 + \frac{1}{M}\|\mathbf{\Lambda}^{1/2}\mathbf{W}\|_{\mathrm{F}}^2 + \frac{2\alpha_u}{M}\mathbb{1}_M^\top\left(\mathbf{X}_u\mathbf{W}\right)^\top\left(\mathbf{X}_u\mathbf{W}\right)\mathbb{1}_M - \frac{2\alpha_u}{M}\|\mathbf{X}_u\mathbf{W}\|_{\mathrm{F}}^2,
\end{aligned}$$

where $\mathbf{Y} \in \mathbb{R}^{n_\ell \times M}$ is the matrix with the vector $\mathbf{y}_\ell$ repeated M times as columns and $\mathbf{\Lambda} \in \mathbb{R}^{M \times M}$ is a diagonal matrix with entries λ_m. For the last equality, we used the fact that for any matrices $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{d \times M}$ with columns $\mathbf{A}_{\cdot,m}, \mathbf{B}_{\cdot,m} \in \mathbb{R}^d$, the following property of the Frobenius inner product holds:

$$\langle \mathbf{A}, \mathbf{B} \rangle_{\mathrm{F}} = \mathrm{Tr}(\mathbf{A}^\top \mathbf{B}) = \sum_{m=1}^{M} \sum_{k=1}^{d} \mathbf{A}_{km} \mathbf{B}_{km} = \sum_{m=1}^{M} \mathbf{A}_{\cdot,m}^\top \mathbf{B}_{\cdot,m}. \tag{10}$$

Then, we write the first order Taylor expansion $\mathcal{L}$ near $\mathbf{W}$ by considering a small displacement $\mathbf{H} \in \mathbb{R}^{d \times M}$ and obtain that

$$\begin{aligned}
\mathcal{L}(\mathbf{W}+\mathbf{H}) &= \frac{1}{n_\ell M} \|\mathbf{Y} - \mathbf{X}_\ell (\mathbf{W}+\mathbf{H})\|_{\mathrm{F}}^2 + \frac{1}{M} \|\mathbf{\Lambda}^{1/2}(\mathbf{W}+\mathbf{H})\|_{\mathrm{F}}^2 + \frac{2\alpha_u}{M} \mathbb{1}_M^\top \left(\mathbf{X}_u (\mathbf{W}+\mathbf{H})\right)^\top \left(\mathbf{X}_u (\mathbf{W}+\mathbf{H})\right) \mathbb{1}_M \\
&\quad - \frac{2\alpha_u}{M} \|\mathbf{X}_u(\mathbf{W}+\mathbf{H})\|_{\mathrm{F}}^2 \\
&= \frac{1}{n_\ell M} \|\mathbf{Y} - \mathbf{X}_\ell \mathbf{W}\|_{\mathrm{F}}^2 + \frac{1}{M} \|\mathbf{\Lambda}^{1/2}\mathbf{W}\|_{\mathrm{F}}^2 + \frac{2\alpha_u}{M} \mathbb{1}_M^\top \left(\mathbf{X}_u \mathbf{W}\right)^\top \left(\mathbf{X}_u \mathbf{W}\right) \mathbb{1}_M \\
&\quad - \frac{2\alpha_u}{M} \|\mathbf{X}_u \mathbf{W}\|_{\mathrm{F}}^2 - \langle \frac{2}{n_l M} \left(\mathbf{Y} - \mathbf{X}_\ell \mathbf{W}\right), \mathbf{X}_\ell \mathbf{H} \rangle_{\mathrm{F}} \\
&\quad + \langle \frac{2}{M} \mathbf{\Lambda}^{1/2}\mathbf{W}, \mathbf{\Lambda}^{1/2}\mathbf{H} \rangle_{\mathrm{F}} + \frac{4\alpha_u}{M} \mathbb{1}_M^\top \left(\mathbf{X}_u \mathbf{W}\right)^\top \left(\mathbf{X}_u \mathbf{H}\right) \mathbb{1}_M \\
&\quad - \frac{4\alpha_u}{M} \langle \mathbf{X}_u \mathbf{W}, \mathbf{X}_u \mathbf{H} \rangle_{\mathrm{F}} \\
&\quad + \underbrace{\frac{1}{n_\ell M} \|\mathbf{X}_\ell \mathbf{H}\|_{\mathrm{F}}^2 + \frac{1}{M} \|\mathbf{\Lambda}^{1/2}\mathbf{H}\|_{\mathrm{F}}^2 + \frac{2\alpha_u}{M} \mathbb{1}_M^\top \left(\mathbf{X}_u \mathbf{H}\right)^\top \left(\mathbf{X}_u \mathbf{H}\right) \mathbb{1}_M - \frac{2\alpha_u}{M} \|\mathbf{X}_u \mathbf{H}\|_{\mathrm{F}}^2}_{o(\|\mathbf{H}\|_{\mathrm{F}})} \\
&= \mathcal{L}(\mathbf{W}) + \frac{2}{M} \langle \mathbf{\Lambda}\mathbf{W} - \frac{1}{n_\ell} \mathbf{X}_\ell^\top \left(\mathbf{Y} - \mathbf{X}_\ell \mathbf{W}\right) - 2\alpha_u \mathbf{X}_u^\top \mathbf{X}_u \mathbf{W}, \mathbf{H} \rangle_{\mathrm{F}} \\
&\quad + \frac{4\alpha_u}{M} \underbrace{\mathbb{1}_M^\top \left(\mathbf{X}_u \mathbf{W}\right)^\top \left(\mathbf{X}_u \mathbf{H}\right) \mathbb{1}_M}_{d(\mathbf{H})} + o(\|\mathbf{H}\|_{\mathrm{F}}).
\end{aligned} \tag{11}$$

Moreover, we have that

$$\begin{aligned}
d(\mathbf{H}) = \mathbb{1}_M^\top \left(\mathbf{X}_u \mathbf{W}\right)^\top \left(\mathbf{X}_u \mathbf{H}\right) \mathbb{1}_M &= \sum_{m=1}^{M} \sum_{k=1}^{M} \mathbf{W}_{\cdot,m}^\top \mathbf{X}_u^\top \mathbf{X}_u \mathbf{H}_{\cdot,k} \\
&= \sum_{k=1}^{M} \left(\mathbf{X}_u^\top \mathbf{X}_u \sum_{m=1}^{M} \mathbf{W}_{\cdot,m} \right)^\top \mathbf{H}_{\cdot,k} \\
&= \sum_{k=1}^{M} \mathbf{L}_{\cdot,k}^\top \mathbf{H}_{\cdot,k} \\
&= \langle \mathbf{L}, \mathbf{H} \rangle_{\mathrm{F}},
\end{aligned}$$

where $\mathbf{L} \in \mathbb{R}^{d \times d}$ is the matrix with the vector $\mathbf{X}_u^\top \mathbf{X}_u \sum_{m=1}^{M} \mathbf{W}_{\cdot,m}$ repeated d times as columns. Another way to write columns of $\mathbf{L}$ is the following:

$$\mathbf{L}_{\cdot,m} = \mathbf{X}_u^\top \mathbf{X}_u \sum_{k=1}^{M} \mathbf{W}_{\cdot,k} = \mathbf{X}_u^\top \mathbf{X}_u \mathbf{W} \mathbb{1}_M.$$

Hence, by introducing $\mathbf{U}^{[M]}$, the unit matrix of size M, we obtain:

$$\mathbf{L} = \mathbf{X}_u^\top \mathbf{X}_u \mathbf{W} \mathbb{1}_M \mathbb{1}_M^\top = \mathbf{X}_u^\top \mathbf{X}_u \mathbf{W} \mathbf{U}^{[M]}.$$

It leads to:

$$d(\mathbf{H}) = \langle \mathbf{X}_u^\top \mathbf{X}_u \mathbf{W} \mathbf{U}^{[M]}, \mathbf{H} \rangle_{\mathrm{F}}.$$

By injecting $d(\mathbf{H})$ into (11), we obtain that

$$\mathcal{L}(\mathbf{W}+\mathbf{H}) = \mathcal{L}(\mathbf{W}) + \langle \underbrace{\frac{2}{M}\left(\mathbf{\Lambda}\mathbf{W} - \frac{1}{n_\ell}\mathbf{X}_\ell^\top(\mathbf{Y}-\mathbf{X}_\ell\mathbf{W}) - 2\alpha_u\mathbf{X}_u^\top\mathbf{X}_u\mathbf{W} + 2\alpha_u\mathbf{X}_u^\top\mathbf{X}_u\mathbf{W}\mathbf{U}^{[M]}\right)}_{\nabla\mathcal{L}(\mathbf{W})}, \mathbf{H}\rangle_{\mathrm{F}} + o(\|\mathbf{H}\|_{\mathrm{F}}).$$

Finally, the gradient of $\mathcal{L}$ writes:

$$\nabla\mathcal{L}(\mathbf{W}) = \frac{2}{M}\left[\left(\Lambda + \frac{\mathbf{X}_\ell^\top\mathbf{X}_\ell}{n_\ell}\right)\mathbf{W} + 2\alpha_u\mathbf{X}_u^\top\mathbf{X}_u\mathbf{W}\left(\mathbf{U}^{[M]} - \mathbf{I}_M\right) - \frac{\mathbf{X}_\ell^\top\mathbf{Y}}{n_\ell}\right]. \tag{12}$$

By combining (9) and (12), recalling that $\alpha_u = \frac{\gamma}{2n_u(M-1)}$, we deduce that any stationary point $\mathbf{W}$ is characterized by the following linear problem:

$$\left(\Lambda + \frac{\mathbf{X}_\ell^\top\mathbf{X}_\ell}{n_\ell}\right)\mathbf{W} + \frac{\gamma}{n_u(M-1)}\mathbf{X}_u^\top\mathbf{X}_u\mathbf{W}\left(\mathbf{U}^{[M]} - \mathbf{I}_M\right) = \frac{\mathbf{X}_\ell^\top\mathbf{Y}}{n_\ell}.$$

Moreover, this matrix equality holds if and only if it holds at the column level. Hence, we have

$$\begin{aligned}
&\left(\Lambda + \frac{\mathbf{X}_\ell^\top\mathbf{X}_\ell}{n_\ell}\right)\mathbf{W} + \frac{\gamma}{n_u(M-1)}\mathbf{X}_u^\top\mathbf{X}_u\mathbf{W}\left(\mathbf{U}^{[M]} - \mathbf{I}_M\right) = \frac{\mathbf{X}_\ell^\top\mathbf{Y}}{n_\ell} \\
\iff &\forall m \in [\![1, M]\!], \quad \left(\lambda_m\mathbf{I}_d + \frac{\mathbf{X}_\ell^\top\mathbf{X}_\ell}{n_\ell}\right)\boldsymbol{\omega}_m + \frac{\gamma}{n_u(M-1)}\mathbf{X}_u^\top\mathbf{X}_u\left(\sum_{k=1}^{k}\boldsymbol{\omega}_k - \boldsymbol{\omega}_m\right) = \frac{\mathbf{X}_\ell^\top\mathbf{y}_\ell}{n_\ell} \\
\iff &\forall m \in [\![1, M]\!], \quad \left(\lambda_m\mathbf{I}_d + \frac{\mathbf{X}_\ell^\top\mathbf{X}_\ell}{n_\ell}\right)\boldsymbol{\omega}_m = \frac{\mathbf{X}_\ell^\top\mathbf{y}_\ell}{n_\ell} - \frac{\gamma}{n_u(M-1)}\mathbf{X}_u^\top\mathbf{X}_u\sum_{k=1|k\neq m}^{M}\boldsymbol{\omega}_k.
\end{aligned}$$

□

E.3 Proof of Proposition 3.3

We start by recalling the definition of coercivity.

Definition E.1. *A bilinear form $a\colon \mathbf{H}\times\mathbf{H}\mapsto\mathbb{R}$, where $\mathbf{H}$ is a Hilbert space, is called coercive if there exists $\gamma > 0$ such that:*

$$\forall \mathbf{x}\in\mathbf{H}, a(\mathbf{x},\mathbf{x}) \geq \gamma\|\mathbf{x}\|^2. \tag{13}$$

Then, we prove the following technical lemmas.

Lemma E.2. *(see Boyd and Vandenberghe, 2004, chap. 3, p. 74). Let $\mathbf{E}$ be a vector space. A function $f\colon \mathbf{E}\to\mathbb{R}$ is convex if and only if for all $\mathbf{x}\in\mathrm{dom}(f)$ and all $\mathbf{v}\in E$, the function $g\colon\mathbb{R}\to\mathbb{R}, t\mapsto f(\mathbf{x}+t\mathbf{v})$ is convex on its domain $\{t\in\mathbb{R}|\mathbf{x}+t\mathbf{v}\in\mathrm{dom}(f)\}$.*

Remark E.1. *By construction,* $\mathrm{dom(f)}$ *is a convex set if and only if all the* $\mathrm{dom(g)}$ *are convex sets.*

Proof. The implication part is straightforward as composing by an affine function preserves convexity and that the convexity of $\mathrm{dom}(f)$ induces the convexity of all the $\mathrm{dom}(g)$. We will now prove the converse. We assume that for all $\mathbf{x}\in\mathrm{dom}(f)$ and all $\mathbf{v}\in E$, the function $g\colon t\mapsto f(\mathbf{x}+t\mathbf{v})$ is convex on its domain i.e. for all $\mathbf{x}\in\mathrm{dom}(f), v\in\mathbf{E}$, we have for all $\alpha\in[0,1]$ and for all $t,t'\in\mathrm{dom}(g)$:

$$\begin{aligned}
&g(\alpha t + (1-\alpha)t') \leq \alpha g(t) + (1-\alpha)g(t') \\
\iff\ &f(\mathbf{x}+(\alpha t+(1-\alpha)t')\mathbf{v}) \leq \alpha f(\mathbf{x}+tv) + (1-\alpha)f(\mathbf{x}+t'\mathbf{v}).
\end{aligned} \tag{14}$$

Let $\theta\in[0,1]$ and $\mathbf{u},\mathbf{y}\in\mathrm{dom}(f)$. By assumption, all the $\mathrm{dom}(g)$ are convex sets, so $\mathrm{dom}(f)$ is a convex set. We can apply (14) with $\mathbf{x}=\mathbf{u}\in\mathrm{dom}(f), \mathbf{v}=\mathbf{y}-\mathbf{u}\in\mathbf{E}, \alpha=\theta\in[0,1], t=0, t'=1$. Indeed, $\mathbf{x}+t\mathbf{v}=\mathbf{u}\in\mathrm{dom}(f), \mathbf{x}+t'\mathbf{v}=y\in\mathrm{dom}(f)$, ensuring that $t,t'\in\mathrm{dom}(g)$. We obtain:

$$\begin{aligned}
&f(\mathbf{x}+(\alpha t+(1-\alpha)t')\mathbf{v}) \leq \alpha f(\mathbf{x}+t\mathbf{v}) + (1-\alpha)f(\mathbf{x}+t'\mathbf{v}) \\
\iff\ &f(\theta\mathbf{u}+(1-\theta)\mathbf{y}) \leq \theta f(\mathbf{x}) + (1-\theta)f(\mathbf{y}).
\end{aligned}$$

Hence, f is convex. □

Lemma E.3. *For any positive semi-definite matrix* $\mathbf{S} \in \mathbb{R}^{d\times d}$ *with minimum and maximum eigenvalues* $\lambda_{\min}, \lambda_{\max}$ *respectively, we have:*

$$\forall \boldsymbol{\omega} \in \mathbb{R}^d, \lambda_{\min}\|\boldsymbol{\omega}\|_2^2 \leq \boldsymbol{\omega}^\top \mathbf{S}\boldsymbol{\omega} \leq \lambda_{\max}\|\boldsymbol{\omega}\|_2^2. \tag{15}$$

Proof. Let $\mathbf{S} \in \mathbb{R}^{d\times d}$ be a positive semi-definite matrix. Let $\lambda_{\max} = \lambda_1 \geq \cdots \geq \lambda_d = \lambda_{\min} \geq 0$ be the eigenvalues sorted in decreasing order. The spectral theorem ensures the existence of $\mathbf{U} \in \mathbb{R}^{d\times d}$ orthogonal and $\mathbf{D} \in \mathbb{R}^{d\times d}$ diagonal with entries $\mathbf{D}_{ii} = \lambda_i$ such that:

$$\mathbf{S} = \mathbf{U}^\top \mathbf{D}\mathbf{U}.$$

Let $\boldsymbol{\omega} \in \mathbb{R}^d$. We have:

$$\begin{aligned}\boldsymbol{\omega}^\top \mathbf{S}\boldsymbol{\omega} &= \boldsymbol{\omega}^\top \mathbf{U}^\top \mathbf{D}\mathbf{U}\boldsymbol{\omega} \\ &= (\mathbf{U}\boldsymbol{\omega})^\top \mathbf{D}\mathbf{U}\boldsymbol{\omega} \\ &= \sum_{k=1}^{d} \underbrace{\lambda_k}_{\geq 0} \underbrace{(\mathbf{U}\boldsymbol{\omega})_k^2}_{\geq 0}.\end{aligned}$$

Hence, we deduce:

$$\lambda_{\min}\|\mathbf{U}\boldsymbol{\omega}\|_2^2 = \lambda_{\min}\sum_{k=1}^{d}(\mathbf{U}\boldsymbol{\omega})_k^2 \leq \boldsymbol{\omega}^\top \mathbf{S}\boldsymbol{\omega} \leq \lambda_{\max}\sum_{k=1}^{d}(\mathbf{U}\boldsymbol{\omega})_k^2 = \lambda_{\max}\|\mathbf{U}\boldsymbol{\omega}\|_2^2. \tag{16}$$

Using the fact that $\mathbf{U}$ is orthogonal, we know that:

$$\|\mathbf{U}\boldsymbol{\omega}\|_2^2 = \boldsymbol{\omega}^\top \underbrace{\mathbf{U}^\top \mathbf{U}}_{=\mathbf{I}_d}\boldsymbol{\omega} = \boldsymbol{\omega}^\top \boldsymbol{\omega} = \|\boldsymbol{\omega}\|_2^2. \tag{17}$$

Finally, we combine (16) and (17) to obtain the desired inequalities:

$$\lambda_{\min}\|\boldsymbol{\omega}\|_2^2 \leq \boldsymbol{\omega}^\top \mathbf{S}\boldsymbol{\omega} \leq \lambda_{\max}\|\boldsymbol{\omega}\|_2^2.$$

□

Then, we show that the loss function in (**P**) can be reformulated as

$$\begin{aligned}\mathcal{L}(\mathbf{W}) = \frac{1}{M}\sum_{m=1}^{M}\left[\frac{1}{n_\ell}\|\mathbf{y}_\ell - \mathbf{X}_\ell\boldsymbol{\omega}_m\|_2^2 + \boldsymbol{\omega}_m^\top\left(\lambda_m\mathbf{I}_d - \frac{\gamma(M+1)}{n_u(M-1)}\mathbf{X}_u^\top\mathbf{X}_u\right)\boldsymbol{\omega}_m\right] \\ + \frac{\gamma}{2n_uM(M-1)}\sum_{m=1}^{M}\sum_{k=1}^{M}(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k)^\top \mathbf{X}_u^\top\mathbf{X}_u(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k).\end{aligned} \tag{18}$$

Proof. We have that

$$\begin{aligned}\mathcal{L}(\mathbf{W}) &= \frac{1}{M}\sum_{m=1}^{M}\frac{1}{n_\ell}\sum_{i=1}^{n_\ell}\left(y_i - \boldsymbol{\omega}_m^\top\mathbf{x}_i\right)^2 + \frac{1}{M}\sum_{m=1}^{M}\lambda_m\|\boldsymbol{\omega}_m\|_2^2 \\ &\quad + \frac{\gamma}{n_uM(M-1)}\sum_{m\neq k}\sum_{i=n_\ell+1}^{n_\ell+n_u}\boldsymbol{\omega}_m^\top\mathbf{x}_i\boldsymbol{\omega}_k^\top\mathbf{x}_i \\ &= \frac{1}{M}\sum_{m=1}^{M}\frac{1}{n_\ell}\|\mathbf{y}_\ell - \mathbf{X}_\ell\boldsymbol{\omega}_m\|_2^2 + \frac{1}{M}\sum_{m=1}^{M}\lambda_m\boldsymbol{\omega}_m^\top\boldsymbol{\omega}_m \\ &\quad + \frac{\gamma}{n_uM(M-1)}\sum_{m\neq k}(\mathbf{X}_u\boldsymbol{\omega}_m)^\top(\mathbf{X}_u\boldsymbol{\omega}_k) \\ &= \frac{1}{M}\sum_{m=1}^{M}\frac{1}{n_\ell}\|\mathbf{y}_\ell - \mathbf{X}_\ell\boldsymbol{\omega}_m\|_2^2 + \frac{1}{M}\sum_{m=1}^{M}\lambda_m\boldsymbol{\omega}_m^\top\boldsymbol{\omega}_m \\ &\quad + \frac{\gamma}{n_uM(M-1)}\sum_{m\neq k}\boldsymbol{\omega}_m^\top\mathbf{X}_u^\top\mathbf{X}_u\boldsymbol{\omega}_k.\end{aligned} \tag{19}$$

Using the fact that $\boldsymbol{\omega}_m^\top \mathbf{X}_u^\top \mathbf{X}_u \boldsymbol{\omega}_k$ is a real number, it is equal to its transpose term. We deduce

$$\begin{aligned}\boldsymbol{\omega}_m^\top \mathbf{X}_u^\top \mathbf{X}_u \boldsymbol{\omega}_k &= \frac{1}{2}\left[\boldsymbol{\omega}_m^\top \mathbf{X}_u^\top \mathbf{X}_u \boldsymbol{\omega}_k + \boldsymbol{\omega}_k^\top \mathbf{X}_u^\top \mathbf{X}_u \boldsymbol{\omega}_m\right] \\ &= \frac{1}{2}\left[\left(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k\right)^\top \mathbf{X}_u^\top \mathbf{X}_u \left(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k\right) - \boldsymbol{\omega}_m^\top \mathbf{X}_u^\top \mathbf{X}_u \boldsymbol{\omega}_m - \boldsymbol{\omega}_k^\top \mathbf{X}_u^\top \mathbf{X}_u \boldsymbol{\omega}_k\right].\end{aligned}$$

Thus, by summing over $\{m \neq k\}$, we obtain:

$$\begin{aligned}\sum_{m\neq k} \boldsymbol{\omega}_m^\top \mathbf{X}_u^\top \mathbf{X}_u \boldsymbol{\omega}_k &= \sum_{m\neq k} \frac{1}{2}\left[\left(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k\right)^\top \mathbf{X}_u^\top \mathbf{X}_u \left(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k\right) - \boldsymbol{\omega}_m^\top \mathbf{X}_u^\top \mathbf{X}_u \boldsymbol{\omega}_m - \boldsymbol{\omega}_k^\top \mathbf{X}_u^\top \mathbf{X}_u \boldsymbol{\omega}_k\right] \\ &= \frac{1}{2}\sum_{m\neq k} \left(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k\right)^\top \mathbf{X}_u^\top \mathbf{X}_u \left(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k\right) - (M-1)\sum_{m=1}^M \boldsymbol{\omega}_m^\top \mathbf{X}_u^\top \mathbf{X}_u \boldsymbol{\omega}_m \\ &= \frac{1}{2}\sum_{m,k} \left(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k\right)^\top \mathbf{X}_u^\top \mathbf{X}_u \left(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k\right) \\ &\quad - \frac{1}{2}\sum_{m=1}^M (2\boldsymbol{\omega}_m)^\top \mathbf{X}_u^\top \mathbf{X}_u (2\boldsymbol{\omega}_m) - (M-1)\sum_{m=1}^M \boldsymbol{\omega}_m^\top \mathbf{X}_u^\top \mathbf{X}_u \boldsymbol{\omega}_m \\ &= \frac{1}{2}\sum_{m,k} \left(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k\right)^\top \mathbf{X}_u^\top \mathbf{X}_u \left(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k\right) - (M+1)\sum_{m=1}^M \boldsymbol{\omega}_m^\top \mathbf{X}_u^\top \mathbf{X}_u \boldsymbol{\omega}_m.\end{aligned}$$

We now inject this term in (19) and gather quadratic and cross terms to obtain

$$\begin{aligned}\mathcal{L}(\mathbf{W}) &= \frac{1}{M}\sum_{m=1}^M \frac{1}{n_\ell}\|\mathbf{y}_\ell - \mathbf{X}_\ell \boldsymbol{\omega}_m\|_2^2 + \frac{1}{M}\sum_{m=1}^M \lambda_m \boldsymbol{\omega}_m^\top \boldsymbol{\omega}_m \\ &\quad + \frac{\gamma}{n_u M(M-1)}\left[\frac{1}{2}\sum_{m,k}\left(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k\right)^\top \mathbf{X}_u^\top \mathbf{X}_u \left(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k\right) - (M+1)\sum_{m=1}^M \boldsymbol{\omega}_m^\top \mathbf{X}_u^\top \mathbf{X}_u \boldsymbol{\omega}_m\right] \\ &= \frac{1}{M}\sum_{m=1}^M \frac{1}{n_\ell}\|\mathbf{y}_\ell - \mathbf{X}_\ell \boldsymbol{\omega}_m\|_2^2 \\ &\quad + \frac{1}{M}\sum_{m=1}^M \boldsymbol{\omega}_m^\top \left(\lambda_m \mathbf{I}_d - \frac{\gamma(M+1)}{n_u(M-1)}\mathbf{X}_u^\top \mathbf{X}_u\right)\boldsymbol{\omega}_m \\ &\quad + \frac{\gamma}{2n_u M(M-1)}\sum_{m=1}^M\sum_{k=1}^M \left(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k\right)^\top \mathbf{X}_u^\top \mathbf{X}_u \left(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k\right).\end{aligned}$$

□

We now proceed to the proof of Proposition 3.3.

Proof. We assume that assumption **A** holds. Using the formulation of the loss in (18) and the quantities $\mathbf{S}_u, \alpha_u$ introduced in (6), we can decompose it as follows:

$$\begin{aligned}\mathcal{L}(\mathbf{W}) &= \underbrace{\frac{1}{M}\sum_{m=1}^M \frac{1}{n_\ell}\|\mathbf{y}_\ell - \mathbf{X}_\ell \boldsymbol{\omega}_m\|_2^2}_{\ell_1(\mathbf{W})} + \underbrace{\frac{1}{M}\sum_{m=1}^M \boldsymbol{\omega}_m^\top \left(\lambda_m \mathbf{I}_d - \mathbf{S}_u\right)\boldsymbol{\omega}_m}_{\ell_2(\mathbf{W})} \\ &\quad + \underbrace{\frac{\alpha_u}{M}\sum_{m=1}^M\sum_{k=1}^M \left(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k\right)^\top \mathbf{X}_u^\top \mathbf{X}_u \left(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k\right)}_{\ell_3(\mathbf{W})}.\end{aligned}$$

Continuity of $\mathcal{L}$. The losses ℓ_1, ℓ_2 and ℓ_3 are differentiable w.r.t $\mathbf{W}$, leading to $\mathcal{L}$ being differentiable. Thus, $\mathcal{L}$ is continuous on $\mathbb{R}^{d\times M}$.

Strict Convexity of $\mathcal{L}$. We will show that ℓ_1, ℓ_2, ℓ_3 are convex and that, in addition, ℓ_2 is strictly convex. This will lead to the strict convexity of $\mathcal{L}$ on the convex set $\mathbb{R}^{d\times M}$.

- $\|\cdot\|_2^2 \colon \mathbb{R}^d \to \mathbb{R}$ is convex as it is a norm function. For all $m \in [\![1, M]\!]$, we define the affine function

$$\begin{aligned} a^m \colon \mathbb{R}^{d\times M} &\to \mathbb{R}^d \\ \mathbf{W} &\mapsto \mathbf{y}_\ell - \mathbf{X}_\ell \boldsymbol{\omega}_m. \end{aligned}$$

  As composing by an affine function preserves the convexity, we deduce that for all $m \in [\![1, M]\!]$,

$$\begin{aligned} \ell_1^m \colon \mathbb{R}^{d\times M} &\to \mathbb{R} \\ \mathbf{W} &\mapsto \|\mathbf{y}_\ell - \mathbf{X}_\ell \boldsymbol{\omega}_m\|_2^2. \end{aligned}$$

  is convex. By non-negative weighted summation, we obtain that $\ell_1 \colon \mathbb{R}^{d\times M} \to \mathbb{R}$ is convex.

- For a given matrix $\mathbf{P} \in \mathbb{R}^{d\times d}$, we define

$$\begin{aligned} \|\cdot\|_{\mathbf{P}}^2 \colon \mathbb{R}^d &\to \mathbb{R} \\ \boldsymbol{\omega} &\mapsto \boldsymbol{\omega}^\top \mathbf{P} \boldsymbol{\omega}. \end{aligned}$$

  Such a function is twice differentiable with Hessian $\mathbf{P}$. Using the property of differentiable convex functions, $\|\cdot\|_{\mathbf{P}}^2$ is convex (respectively strictly convex) if and only if $\mathbf{P}$ is positive semi-definite (respectively positive definite). Using Assumption (A), we know that for all $m \in [\![1, M]\!]$, $\|\cdot\|^2_{\lambda_m \mathbf{I}_d - \mathbf{S}_u}$ is strictly convex. As before, for all $m \in [\![1, M]\!]$, we can define the linear function

$$\begin{aligned} e^m \colon \mathbb{R}^{d\times M} &\to \mathbb{R}^d \\ \mathbf{W} &\mapsto \boldsymbol{\omega}_m. \end{aligned}$$

  As composing by a linear function preserves the strict convexity (provided that this linear function is not identically equal to zero), we deduce that for all $m \in [\![1, M]\!]$,

$$\begin{aligned} \ell_2^m \colon \mathbb{R}^{d\times M} &\to \mathbb{R} \\ \mathbf{W} &\mapsto \boldsymbol{\omega}_m^\top \left(\lambda_m \mathbf{I}_d - \mathbf{S}_u\right) \boldsymbol{\omega}_m. \end{aligned}$$

  is strictly convex. By non-negative weighted summation of strictly convex functions, we obtain that $\ell_2 \colon \mathbb{R}^{d\times M} \to \mathbb{R}$ is strictly convex.

- Let $\mathbf{W}, \mathbf{H} \in \mathbb{R}^{d\times M}$. We consider

$$\begin{aligned} f \colon \mathbb{R} &\to \mathbb{R} \\ t &\mapsto \ell_3(\mathbf{W} + t\mathbf{H}). \end{aligned}$$

  We will show that f is convex. Following Lemma E.2, as $\mathbf{W}$ and $\mathbf{H}$ are taken arbitrary in $\mathbb{R}^{d\times M}$, it will induce the convexity of ℓ_3. Let $t \in \mathbb{R}$. We have:

$$\begin{aligned} f(t) &= \ell_3(\mathbf{W} + t\mathbf{H}) \\ &= \frac{\alpha_u}{M} \sum_{m=1}^{M} \sum_{k=1}^{M} \left(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k + t(\mathbf{h}_m + \mathbf{h}_k)\right)^\top \mathbf{X}_u^\top \mathbf{X}_u \left(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k + t(\mathbf{h}_m + \mathbf{h}_k)\right) \\ &= t^2 \times \frac{\alpha_u}{M} \sum_{m=1}^{M} \sum_{k=1}^{M} \left(\mathbf{h}_m + \mathbf{h}_k\right)^\top \mathbf{X}_u^\top \mathbf{X}_u \left(\mathbf{h}_m + \mathbf{h}_k\right) \\ &\quad + 2t \times \frac{\alpha_u}{M} \sum_{m=1}^{M} \sum_{k=1}^{M} \left(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k\right)^\top \mathbf{X}_u^\top \mathbf{X}_u \left(\mathbf{h}_m + \mathbf{h}_k\right) \\ &\quad + \frac{\alpha_u}{M} \sum_{m=1}^{M} \sum_{k=1}^{M} \left(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k\right)^\top \mathbf{X}_u^\top \mathbf{X}_u \left(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k\right) \\ &= at^2 + bt + c. \end{aligned}$$

where

$$\begin{cases} a = \frac{\alpha_u}{M} \sum_{m=1}^{M} \sum_{k=1}^{M} \left(\mathbf{h}_m + \mathbf{h}_k\right)^\top \mathbf{X}_u^\top \mathbf{X}_u \left(\mathbf{h}_m + \mathbf{h}_k\right) \\ b = \frac{2\alpha_u}{M} \sum_{m=1}^{M} \sum_{k=1}^{M} \left(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k\right)^\top \mathbf{X}_u^\top \mathbf{X}_u \left(\mathbf{h}_m + \mathbf{h}_k\right) \\ c = \frac{\alpha_u}{M} \sum_{m=1}^{M} \sum_{k=1}^{M} \left(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k\right)^\top \mathbf{X}_u^\top \mathbf{X}_u \left(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k\right). \end{cases}$$

We see that f is a second-order polynomial function of the real line. It is a convex function if and only if $a \geq 0$. It should be noted that $\mathbf{X}_u^\top \mathbf{X}_u$ is symmetric positive semi-definite by construction. Indeed, we have:

$$\forall \boldsymbol{\omega} \in \mathbb{R}^d, \boldsymbol{\omega}^\top \mathbf{X}_u^\top \mathbf{X}_u \boldsymbol{\omega} = \|\mathbf{X}_u \boldsymbol{\omega}\|_2^2 \geq 0.$$

Hence, we deduce:

$$a = \sum_{m=1}^{M} \sum_{k=1}^{M} \underbrace{\left(\mathbf{h}_m + \mathbf{h}_k\right)^\top \mathbf{X}_u^\top \mathbf{X}_u \left(\mathbf{h}_m + \mathbf{h}_k\right)}_{\geq 0} \geq 0.$$

We have shown that f is a convex function of the real line. As $\mathbf{W}$ and $\mathbf{H}$ were taken arbitrarily, we obtain that $\ell_3 \colon \mathbb{R}^{d \times M} \to \mathbb{R}$ is convex.

By summation of convex functions and a strictly convex function, we finally obtain that $\mathcal{L} \colon \mathbb{R}^{d \times M} \to \mathbb{R}$ is strictly convex.

Coercivity of $\mathcal{L}$. We will show that under assumptions A, $\mathcal{L}$ is lower-bounded by a coercive function, implying the coercivity of $\mathcal{L}$ i.e

$$\lim_{\|\mathbf{W}\|_{\mathrm{F}} \to +\infty} \mathcal{L}(\mathbf{W}) = +\infty.$$

where $\|\cdot\|_{\mathrm{F}}$ is the norm associated to the Frobenius inner product

$$\begin{aligned} \langle \cdot, \cdot \rangle_{\mathrm{F}} \colon \mathbb{R}^{d \times M} \times \mathbb{R}^{d \times M} &\to \mathbb{R} \\ (\mathbf{A}, \mathbf{B}) &\mapsto \mathrm{Tr}(\mathbf{A}^\top \mathbf{B}). \end{aligned}$$

- The function ℓ_1 is non-negative as the sum of non-negative functions.
- By construction, $\mathbf{X}_u^\top \mathbf{X}_u$ is symmetric positive semi-definite. Hence, we have:

  $$\forall m, k \in [\![1, M]\!], \quad \left(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k\right)^\top \mathbf{X}_u^\top \mathbf{X}_u \left(\boldsymbol{\omega}_m + \boldsymbol{\omega}_k\right) \geq 0.$$

  It leads to ℓ_3 being non-negative as the sum of non-negative functions.
- By lower bounding, we obtain:

$$\mathcal{L}(\mathbf{W}) = \underbrace{\ell_1(\mathbf{W})}_{\geq 0} + \ell_2(\mathbf{W}) + \underbrace{\ell_3(\mathbf{W})}_{\geq 0} \geq \ell_2(\mathbf{W}) = \frac{1}{M} \sum_{m=1}^{M} \boldsymbol{\omega}_m^\top \left(\lambda_m \mathbf{I}_d - \mathbf{S}_u\right) \boldsymbol{\omega}_m. \tag{20}$$

Under Assumption A, we know that for all $m \in [\![1, M]\!]$, $\lambda_m \mathbf{I}_d - \mathbf{S}_u$ is positive definite. It ensures that $\lambda_{\min}(\lambda_m \mathbf{I}_d - \mathbf{S}_u)$, the minimum eigenvalue of $\lambda_m \mathbf{I}_d - \mathbf{S}_u$, is positive. Hence, we have:

$$\begin{aligned} \ell_2(\mathbf{W}) &= \frac{1}{M} \sum_{m=1}^{M} \underbrace{\boldsymbol{\omega}_m^\top \left(\lambda_m \mathbf{I}_d - \mathbf{S}_u\right) \boldsymbol{\omega}_m}_{\geq \lambda_{\min}(\lambda_m \mathbf{I}_d - \mathbf{S}_u) \boldsymbol{\omega}_m^\top \boldsymbol{\omega}_m} && \text{(from Lemma E.3)} \\ &\geq \frac{1}{M} \sum_{m=1}^{M} \underbrace{\lambda_{\min}(\lambda_m \mathbf{I}_d - \mathbf{S}_u)}_{>0} \underbrace{\boldsymbol{\omega}_m^\top \boldsymbol{\omega}_m}_{\geq 0} \\ &\geq \underbrace{\frac{1}{M} \min_{m \in [\![1, M]\!]} \{\lambda_{\min}(\lambda_m \mathbf{I}_d - \mathbf{S}_u)\}}_{>0} \sum_{m=1}^{M} \boldsymbol{\omega}_m^\top \boldsymbol{\omega}_m \\ &\geq \gamma \sum_{m=1}^{M} \boldsymbol{\omega}_m^\top \boldsymbol{\omega}_m \text{ where } \gamma = \frac{1}{M} \min_{m \in [\![1, M]\!]} \{\lambda_{\min}(\lambda_m \mathbf{I}_d - \mathbf{S}_u)\} > 0. \end{aligned}$$

As we know that the columns of $\mathbf{W} \in \mathbb{R}^{d\times M}$ are the $\boldsymbol{\omega}_m \in \mathbb{R}^d$, we can use (10) and have that

$$\sum_{m=1}^{M} \boldsymbol{\omega}_m^\top \boldsymbol{\omega}_m = \langle \mathbf{W}, \mathbf{W} \rangle_{\mathrm{F}}.$$

By using the previous lower bound on ℓ_2 in (20), we obtain:

$$\mathcal{L}(\mathbf{W}) \geq \gamma \langle \mathbf{W}, \mathbf{W} \rangle_{\mathrm{F}} \coloneqq a(\mathbf{W}, \mathbf{W}) \text{ where } a(\mathbf{A}, \mathbf{B}) = \gamma \langle \mathbf{A}, \mathbf{B} \rangle_{\mathrm{F}}.$$

Following Definition E.1, it is straightforward that a is a coercive bilinear form on the Hilbert $\mathbb{R}^{d\times M}$:

$$a(\mathbf{W}, \mathbf{W}) = \gamma \langle \mathbf{W}, \mathbf{W} \rangle_{\mathrm{F}} = \gamma \|\mathbf{W}\|_{\mathrm{F}}^2 \to +\infty.$$

Hence by lower bounding, we obtain:

$$\lim_{\|\mathbf{W}\|_{\mathrm{F}} \to +\infty} \mathcal{L}(\mathbf{W}) = +\infty.$$

We have proved that under Assumption **A**, the loss function $\mathcal{L}$ is strictly convex and coercive w.r.t $\mathbf{W}$.

Convergence As a continuous, strictly convex, and coercive function on the convex $\mathbb{R}^{d\times M}$, $\mathcal{L}$ admits a unique global minimizer. As $\mathcal{L}$ is differentiable, this minimizer is a stationary point of $\mathcal{L}$, i.e., must verify the Euler equation (3). Hence, (**P**) converges towards the unique stationary point of $\mathcal{L}$. □

E.4 Proof of Theorem 3.4

The proof of Theorem 3.4 is detailed below.

Proof. Let $\tilde{\mathbf{W}}$ be a stationary point of $\mathcal{L}$, i.e., $\tilde{\mathbf{W}}$ is solution of (3). From Appendix E.2, the columns $\tilde{\boldsymbol{\omega}}_m$ of $\tilde{\mathbf{W}}$, verify (8) and we have for all $m \in [\![1, M]\!]$:

$$\left(\lambda_m \mathbf{I}_d + \frac{\mathbf{X}_\ell^\top \mathbf{X}_\ell}{n_\ell}\right) \tilde{\boldsymbol{\omega}}_m = \frac{\mathbf{X}_\ell^\top \mathbf{y}_\ell}{n_\ell} - \frac{\gamma}{n_u(M-1)} \mathbf{X}_u^\top \mathbf{X}_u \sum_{k=1|k\neq m}^{M} \tilde{\boldsymbol{\omega}}_k$$

$$\iff \frac{\gamma}{n_u(M-1)} \mathbf{X}_u^\top \mathbf{X}_u \sum_{k=1|k\neq m}^{M} \tilde{\boldsymbol{\omega}}_k = \frac{\mathbf{X}_\ell^\top \mathbf{y}_\ell}{n_\ell} - \left(\lambda_m \mathbf{I}_d + \frac{\mathbf{X}_\ell^\top \mathbf{X}_\ell}{n_\ell}\right) \tilde{\boldsymbol{\omega}}_m$$

$$\iff \frac{\gamma}{n_u M(M-1)} \mathbf{X}_u^\top \mathbf{X}_u \sum_{k=1|k\neq m}^{M} \tilde{\boldsymbol{\omega}}_k = \frac{1}{M}\left[\frac{\mathbf{X}_\ell^\top \mathbf{y}_\ell}{n_\ell} - \left(\lambda_m \mathbf{I}_d + \frac{\mathbf{X}_\ell^\top \mathbf{X}_\ell}{n_\ell}\right) \tilde{\boldsymbol{\omega}}_m\right].$$

By taking the left inner product with $\tilde{\boldsymbol{\omega}}_m$ and summing over all $m \in [\![1, M]\!]$, we obtain:

$$\underbrace{\frac{\gamma}{n_u M(M-1)} \sum_{m=1}^{M} \tilde{\boldsymbol{\omega}}_m \mathbf{X}_u^\top \mathbf{X}_u \sum_{k=1|k\neq m}^{M} \tilde{\boldsymbol{\omega}}_k}_{\text{(LHS)}} = \underbrace{\frac{1}{M} \sum_{m=1}^{M} \tilde{\boldsymbol{\omega}}_m^\top \left[\frac{\mathbf{X}_\ell^\top \mathbf{y}_\ell}{n_\ell} - \left(\lambda_m \mathbf{I}_d + \frac{\mathbf{X}_\ell^\top \mathbf{X}_\ell}{n_\ell}\right) \tilde{\boldsymbol{\omega}}_m\right]}_{\text{(RHS)}}.$$

The left-hand side (LHS) term can be rewritten as

$$\begin{aligned}
\text{(LHS)} &= \frac{\gamma}{n_u M(M-1)} \sum_{m=1}^{M} \tilde{\boldsymbol{\omega}}_m \mathbf{X}_u^\top \mathbf{X}_u \sum_{k=1|k\neq m}^{M} \tilde{\boldsymbol{\omega}}_k \\
&= \frac{1}{n_u M(M-1)} \sum_{m=1}^{M} \sum_{k=1|k\neq m}^{M} \tilde{\boldsymbol{\omega}}_m \mathbf{X}_u^\top \mathbf{X}_u \tilde{\boldsymbol{\omega}}_k \\
&= \frac{\gamma}{n_u M(M-1)} \sum_{m\neq k} \tilde{\boldsymbol{\omega}}_m \mathbf{X}_u^\top \mathbf{X}_u \tilde{\boldsymbol{\omega}}_k \\
&= \frac{\gamma}{n_u M(M-1)} \sum_{m\neq k} \sum_{i=n_\ell+1}^{n_\ell+n_u} w_m^\top \mathbf{x}_i w_k^\top \mathbf{x}_i \\
&= -\gamma \ell_{\text{div}}(\tilde{\mathbf{W}}, \mathbf{X}_u),
\end{aligned}$$

where ℓ_{div} is the diversity introduced in (4). Moreover, using the decomposition of the Euclidean norm, we have:

$$\|\mathbf{y}_\ell - \mathbf{X}_\ell\tilde{\boldsymbol{\omega}}_m\|_2^2 = \mathbf{y}_\ell^\top\mathbf{y}_\ell - 2\mathbf{y}_\ell^\top\mathbf{X}_\ell\tilde{\boldsymbol{\omega}}_m + \tilde{\boldsymbol{\omega}}_m^\top\mathbf{X}_\ell^\top\mathbf{X}_\ell\tilde{\boldsymbol{\omega}}_m$$
$$\iff \mathbf{y}_\ell^\top\mathbf{X}_\ell\tilde{\boldsymbol{\omega}}_m = \frac{1}{2}\left[\mathbf{y}_\ell^\top\mathbf{y}_\ell + \tilde{\boldsymbol{\omega}}_m^\top\mathbf{X}_\ell^\top\mathbf{X}_\ell\tilde{\boldsymbol{\omega}}_m - \|\mathbf{y}_\ell - \mathbf{X}_\ell\tilde{\boldsymbol{\omega}}_m\|_2^2\right].$$

As we are in binary classification, we know that each entry of $\mathbf{y}_\ell$ is in $\{-1, +1\}$. Thus, we have:

$$\mathbf{y}_\ell^\top\mathbf{y}_\ell = \|\mathbf{y}_\ell\|_2^2 = \sum_{i=1}^{n_\ell}|y_{\ell,i}|^2 = n_\ell.$$

We inject the new formula for $\mathbf{y}_\ell^\top\mathbf{X}_\ell\tilde{\boldsymbol{\omega}}_m$ in the right-hand side (RHS) part of (E.4) and obtain:

$$\begin{aligned}
(\text{RHS}) &= \frac{1}{M}\sum_{m=1}^{M}\tilde{\boldsymbol{\omega}}_m^\top\left[\frac{\mathbf{X}_\ell^\top\mathbf{y}_\ell}{n_\ell} - \left(\lambda_m\mathbf{I}_d + \frac{\mathbf{X}_\ell^\top\mathbf{X}_\ell}{n_\ell}\right)\tilde{\boldsymbol{\omega}}_m\right] \\
&= \frac{1}{M}\left[\frac{1}{n_l}\sum_{m=1}^{M}\mathbf{y}_\ell^\top\mathbf{X}_\ell\tilde{\boldsymbol{\omega}}_m - \sum_{m=1}^{M}\tilde{\boldsymbol{\omega}}_m^\top\left(\lambda_m\mathbf{I}_d + \frac{\mathbf{X}_\ell^\top\mathbf{X}_\ell}{n_\ell}\right)\tilde{\boldsymbol{\omega}}_m\right] \\
&= \frac{1}{M}\left(\frac{1}{2n_l}\sum_{m=1}^{M}\left[n_\ell + \tilde{\boldsymbol{\omega}}_m^\top\mathbf{X}_\ell^\top\mathbf{X}_\ell\tilde{\boldsymbol{\omega}}_m - \|\mathbf{y}_\ell - \mathbf{X}_\ell\tilde{\boldsymbol{\omega}}_m\|_2^2\right]\right) \\
&\quad - \frac{1}{M}\sum_{m=1}^{M}\tilde{\boldsymbol{\omega}}_m^\top\left(\lambda_m\mathbf{I}_d + \frac{\mathbf{X}_\ell^\top\mathbf{X}_\ell}{n_\ell}\right)\tilde{\boldsymbol{\omega}}_m \\
&= \frac{1}{M}\left(\frac{M}{2} - \frac{1}{2n_\ell}\sum_{m=1}^{M}\|\mathbf{y}_\ell - \mathbf{X}_\ell\tilde{\boldsymbol{\omega}}_m\|_2^2\right) \\
&\quad + \frac{1}{M}\left(\sum_{m=1}^{M}\tilde{\boldsymbol{\omega}}_m^\top\frac{\mathbf{X}_\ell^\top\mathbf{X}_\ell}{2n_\ell}\tilde{\boldsymbol{\omega}}_m - \tilde{\boldsymbol{\omega}}_m^\top\left(\lambda_m\mathbf{I}_d + \frac{\mathbf{X}_\ell^\top\mathbf{X}_\ell}{n_\ell}\right)\tilde{\boldsymbol{\omega}}_m\right) \\
&= \frac{1}{2}\left[1 - \frac{1}{n_\ell M}\sum_{m=1}^{M}\|\mathbf{y}_\ell - \mathbf{X}_\ell\tilde{\boldsymbol{\omega}}_m\|_2^2 - \frac{2}{M}\sum_{m=1}^{M}\tilde{\boldsymbol{\omega}}_m^\top\left(\lambda_m\mathbf{I}_d + \frac{\mathbf{X}_\ell^\top\mathbf{X}_\ell}{2n_\ell}\right)\tilde{\boldsymbol{\omega}}_m\right] \\
&= \frac{1}{2}\left[1 - \frac{1}{n_\ell M}\sum_{m=1}^{M}\|\mathbf{y}_\ell - \mathbf{X}_\ell\tilde{\boldsymbol{\omega}}_m\|_2^2 - \frac{1}{M}\sum_{m=1}^{M}\tilde{\boldsymbol{\omega}}_m^\top\left(2\lambda_m\mathbf{I}_d + \frac{\mathbf{X}_\ell^\top\mathbf{X}_\ell}{n_\ell}\right)\tilde{\boldsymbol{\omega}}_m\right] \\
&= \frac{1}{2} - \frac{1}{2n_\ell M}\sum_{m=1}^{M}\|\mathbf{y}_\ell - \mathbf{X}_\ell\tilde{\boldsymbol{\omega}}_m\|_2^2 - \frac{1}{2M}\sum_{m=1}^{M}\lambda_m\tilde{\boldsymbol{\omega}}_m^\top\tilde{\boldsymbol{\omega}}_m \\
&\quad - \frac{1}{2M}\sum_{m=1}^{M}\tilde{\boldsymbol{\omega}}_m^\top\left(\lambda_m\mathbf{I}_d + \frac{\mathbf{X}_\ell^\top\mathbf{X}_\ell}{n_\ell}\right)\tilde{\boldsymbol{\omega}}_m \\
&= \frac{1}{2} - \frac{1}{2n_\ell M}\sum_{m=1}^{M}\|\mathbf{y}_\ell - \mathbf{X}_\ell\tilde{\boldsymbol{\omega}}_m\|_2^2 - \frac{1}{2M}\sum_{m=1}^{M}\lambda_m\|\tilde{\boldsymbol{\omega}}_m\|_2^2 \\
&\quad - \frac{1}{2M}\sum_{m=1}^{M}\tilde{\boldsymbol{\omega}}_m^\top\left(\lambda_m\mathbf{I}_d + \frac{\mathbf{X}_\ell^\top\mathbf{X}_\ell}{n_\ell}\right)\tilde{\boldsymbol{\omega}}_m.
\end{aligned}$$

We finally obtain from (LHS) = (RHS), after multiplying by -1, that

$$\gamma\ell_{\text{div}}(\tilde{\mathbf{W}}, \mathbf{X}_u) = \frac{1}{2n_\ell M}\sum_{m=1}^{M}\|\mathbf{y}_\ell - \mathbf{X}_\ell\tilde{\boldsymbol{\omega}}_m\|_2^2 + \frac{1}{2M}\sum_{m=1}^{M}\tilde{\boldsymbol{\omega}}_m^\top\left(\lambda_m\mathbf{I}_d + \frac{\mathbf{X}_\ell^\top\mathbf{X}_\ell}{n_\ell}\right)\tilde{\boldsymbol{\omega}}_m + \frac{1}{2M}\sum_{m=1}^{M}\lambda_m\|\tilde{\boldsymbol{\omega}}_m\|_2^2 - \frac{1}{2}.$$

Under the hypothesis that $\frac{1}{M}\sum_{m=1}^{M}\lambda_m\|\tilde{\boldsymbol{\omega}_m}\|_2^2 \geq 1$, we have:

$$\frac{1}{2M}\sum_{m=1}^{M}\lambda_m\|\tilde{\boldsymbol{\omega}}_m\|_2^2 - \frac{1}{2} = \frac{1}{2}\left[\frac{1}{M}\sum_{m=1}^{M}\lambda_m\|\tilde{\boldsymbol{\omega}}_m\|_2^2 - 1\right] \geq 0.$$

Taking this fact into account, we obtain:

$$\gamma\ell_{\text{div}}(\tilde{\mathbf{W}}, \mathbf{X}_u) = \frac{1}{2n_\ell M}\sum_{m=1}^{M}\|\mathbf{y}_\ell - \mathbf{X}_\ell\tilde{\boldsymbol{\omega}}_m\|_2^2 + \frac{1}{2M}\sum_{m=1}^{M}\tilde{\boldsymbol{\omega}}_m^\top\left(\lambda_m\mathbf{I}_d + \frac{\mathbf{X}_\ell^\top\mathbf{X}_\ell}{n_\ell}\right)\tilde{\boldsymbol{\omega}}_m + \underbrace{\frac{1}{2M}\sum_{m=1}^{M}\lambda_m\|\tilde{\boldsymbol{\omega}}_m\|_2^2}_{\geq 0} - \frac{1}{2}$$

$$\geq \frac{1}{2n_\ell M}\sum_{m=1}^{M}\|\mathbf{y}_\ell - \mathbf{X}_\ell\tilde{\boldsymbol{\omega}}_m\|_2^2 + \frac{1}{2M}\sum_{m=1}^{M}\tilde{\boldsymbol{\omega}}_m^\top\left(\lambda_m\mathbf{I}_d + \frac{\mathbf{X}_\ell^\top\mathbf{X}_\ell}{n_\ell}\right)\tilde{\boldsymbol{\omega}}_m.$$

□

E.5 Proof of Corollary 3.5

The proof of Corollary 3.5 is detailed below.

Proof. By hypothesis, $\frac{1}{M}\sum_{m=1}^{M}\lambda_m\|\tilde{\boldsymbol{\omega}}_m\|_2^2 \geq 1$, thus Theorem 3.4 holds. Using the fact that all λ_m are equal to some λ, it leads to:

$$\gamma\ell_{\text{div}}(\tilde{\mathbf{W}}, \mathbf{X}_u) \geq \underbrace{\frac{1}{2n_\ell M}\sum_{m=1}^{M}\|\mathbf{y}_\ell - \mathbf{X}_\ell\tilde{\boldsymbol{\omega}}_m\|_2^2}_{\geq 0} + \frac{1}{2M}\sum_{m=1}^{M}\tilde{\boldsymbol{\omega}}_m^\top\left(\lambda\mathbf{I}_d + \frac{\mathbf{X}_\ell^\top\mathbf{X}_\ell}{n_\ell}\right)\tilde{\boldsymbol{\omega}}_m$$

$$\geq \frac{1}{2M}\left(\lambda\sum_{m=1}^{M}\|\tilde{\boldsymbol{\omega}}_m\|_2^2 + \frac{1}{n_\ell}\sum_{m=1}^{M}\tilde{\boldsymbol{\omega}}_m^\top\mathbf{X}_\ell^\top\mathbf{X}_\ell\tilde{\boldsymbol{\omega}}_m\right)$$

$$\geq \frac{1}{2M}\left(\lambda + \frac{1}{n_\ell}\lambda_{min}\left(\mathbf{X}_\ell^\top\mathbf{X}_\ell\right)\right)\sum_{m=1}^{M}\|\tilde{\boldsymbol{\omega}}_m\|_2^2 \qquad \text{(from Lemma E.3)}$$

$$= \frac{1}{2M}\left(\lambda + \frac{1}{n_\ell}\lambda_{min}\left(\mathbf{X}_\ell^\top\mathbf{X}_\ell\right)\right)\|\tilde{\mathbf{W}}\|_{\text{F}}^2. \qquad \text{(from (10))}$$

□