Supervised Distributional Reduction via Optimal Transport and Dependence Maximization

Sai-Aakash Ramesh[1*] **Archit Sood**[1] **Andrew Corbett**[1] **Tim Dodwell**[1,2]

[1]digiLab, UK [2]University of Bristol, UK

Abstract

Learning representations that capture both intrinsic data geometry and target-relevant structure remains a fundamental challenge, particularly in settings where data reduction must balance compression with predictive fidelity. While distributional reduction—encompassing joint clustering and dimensionality reduction—offers a principled way to summarize data, its supervised variants remain relatively under-explored, despite the importance of retaining task-relevant signal for downstream prediction and decision-making. We propose Supervised Distributional Reduction (SDR), an algorithm for learning target-aware representations by combining optimal transport with explicit dependence maximization. SDR builds on the Fused Gromov–Wasserstein (FGW) objective to align the relational structure of the input distribution with a set of representative points, while augmenting it with a direct dependence term that encourages the learned embeddings to capture predictive signal more explicitly. This results in compact representations that reflect both geometric structure and supervision. Beyond representation learning, SDR naturally induces a data-dependent, non-stationary geometry that can be leveraged for settings such as Gaussian Process (GP) modelling. By redefining distances through target-aware distributional alignment, SDR enables the construction of adaptive kernels that respond to local variations in both data geometry and supervision, offering an optimal transport-based perspective on non-stationary kernel design.

1 Introduction

Optimal Transport (OT) provides a principled framework for comparing probability distributions, with Gromov–Wasserstein (GW) [Peyré et al., 2016] enabling alignment across heterogeneous spaces by matching relational structure. Building on this, Distributional Reduction (DistR) [Assel et al., 2025] learns a low-dimensional set of representative points that preserves the geometry of a dataset, providing a framework for unifying clustering and dimensionality reduction. However, existing distributional reduction methods are largely unsupervised, which can lead to representations that fail to retain task-relevant information and lack a clear mechanism for out-of-sample generalization. This prevents these methods from being used for downstream prediction tasks.

We address these limitations by proposing Supervised Distributional Reduction (SDR), a framework for learning target-aware latent representations by combining optimal transport with explicit dependence maximization. SDR builds on the Fused Gromov–Wasserstein (FGW) objective [Vayer et al., 2018] to align the relational structure of the input distribution with representative points, while incorporating supervision. Crucially, rather than relying solely on the coupling matrix—where supervision is mediated indirectly and may be diluted by structural constraints—we augment the objective with a direct representation-level dependence term based on Centered Kernel Alignment

*Correspondence to Sai-Aakash Ramesh: <sai-aakash.ramesh@digilab.ai>

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

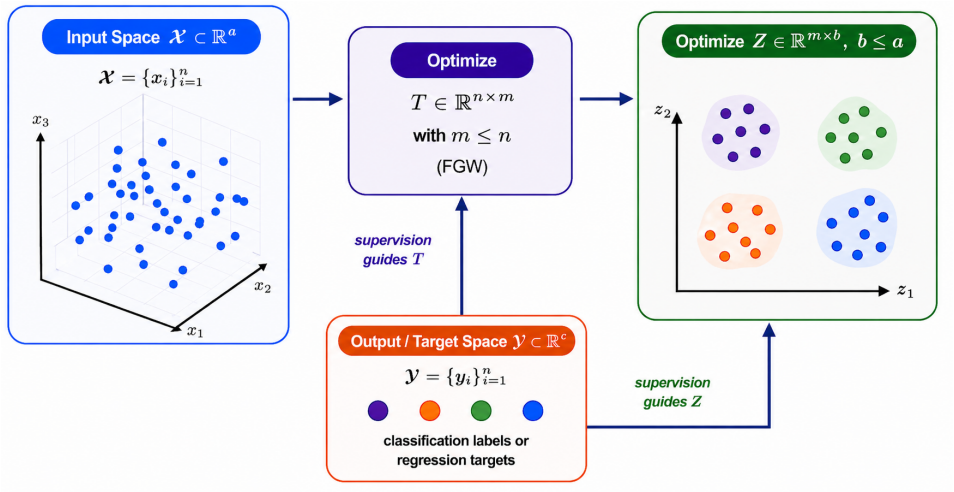


Figure 1: Supervised Distributional Reduction (SDR) with Fused Gromov-Wasserstein OT formulation and explicit dependence maximization for target-aware distributional reduction.

(CKA). This enables explicit alignment between embeddings and targets while preserving intrinsic data geometry.

Beyond representation learning, SDR induces a data-dependent, non-stationary geometry that can be exploited for kernel learning. By constraining embeddings to admit an approximate kernel-based out-of-sample extension, we obtain a stable projection mechanism for unseen data. This allows SDR to define adaptive covariance functions for Gaussian Processes (GPs), where the learned geometry translates into non-stationary kernel behaviour that responds to both structure and supervision in the data.

1.1 Contributions

We make the following contributions in this work:

1. **Supervised distributional reduction via OT and dependence maximization:** We propose SDR, an algorithm that combines OT-based alignment with explicit dependence maximization to learn compact, target-aware representations that preserve both geometry and predictive signal.
2. **Out-of-sample extension via RKHS projection with applications to GPs:** We formulate the mapping from inputs to embeddings as a regularized kernel ridge regression problem, yielding an out-of-sample transformation map that allows SDR to be used as part of predictive pipelines and for non-stationary kernel construction in GPs.

2 Related Work

2.1 Supervised Dimensionality Reduction and Clustering

Supervised dimensionality reduction (DR) methods incorporate label information to learn representations that are more discriminative for downstream tasks. Classical approaches such as Linear Discriminant Analysis (LDA) [Fisher, 1936] maximize class separability through linear projections, while metric learning methods, including Neighbourhood Components Analysis (NCA) [Goldberger et al., 2004] and Large Margin Nearest Neighbour (LMNN) [Weinberger et al., 2005], shape the embedding space using supervised distance objectives. Extensions of unsupervised DR techniques have also been proposed in supervised settings; for instance, Supervised UMAP [McInnes et al., 2020] integrates label information to guide the preservation of local structure in a target-aware manner. In parallel, dependence-based approaches maximize statistical dependence between representations and targets. For example, using the Hilbert Schmidt Independence Criterion (HSIC) [Gretton et al.,

2007] in conjunction with PCA gives rise to a version of Supervised PCA [Barshan et al., 2011]. On the other hand, supervised and semi-supervised clustering methods introduce constraints such as must-link and cannot-link pairs to guide cluster formation [Wagstaff et al., 2001]. Xing et al. [2002] suggest providing supervision in the form of distances (similarities) for metric-informed clustering. Despite these advances, most approaches either focus solely on embedding or clustering, or rely on heuristic combinations of the two, lacking a unified formulation that captures both aspects at the distributional level.

Alongside the rise of deep learning came the paradigm of deep metric learning that used neural networks to learn similarity measures between inputs. Deep metric learning introduced target dependent objectives like contrastive, triplet and proxy-based losses([Schroff et al., 2015], [Khosla et al., 2020], [Kim et al., 2020]) that produced models which shape embedding geometry for discriminability alone. Prototype-based methods such as Prototypical Networks ([Snell et al., 2017]) and Learning Vector Quantization (LVQ) methods like General LVQ ([Sato and Yamada, 1995]) were introduced for classification problems and typically represent each class using a single prototype, thereby collapsing all within-class variation. SDR instead retains input-space structure through an explicit structure-preserving term. Its prototypes are free variables inferred jointly with the embedding, rather than class means fixed in number and identity by the labels, and carry a mass-conserving transport coupling in place of a softmax over distances. Treating labels as features further admits continuous targets, which these classification methods do not.

2.2 Optimal Transport, GW/FGW, and Distributional Reduction

Optimal transport (OT) provides a principled framework for comparing probability distributions by minimizing the cost of transporting probability mass between them [Monge, 1781, Kantorovitch, 1958]. Beyond classical Wasserstein distances, Gromov–Wasserstein (GW) extends OT to settings where only relational information, such as pairwise distances, is available, enabling comparisons across heterogeneous spaces [Peyré et al., 2016]. The Fused Gromov–Wasserstein (FGW) formulation further combines feature-level and structural alignment by integrating Wasserstein and GW objectives into a single framework [Vayer et al., 2018]. These formulations provide a natural mechanism for geometry-aware representation learning. Building on these ideas, OT-based approaches have been proposed for distributional summarization, including Wasserstein barycenters [Cuturi and Doucet, 2014] and reduced-support approximations [Forrow et al., 2019]. Methods such as DistR [Assel et al., 2025] leverage GW to jointly perform clustering and dimensionality reduction by learning a reduced set of representative points that preserve dataset geometry. However, these approaches are largely unsupervised and do not explicitly incorporate target information during the reduction process.

SDR builds on this line of work by integrating supervision directly into the distributional reduction objective, enabling the learning of compact representations that are both geometry-preserving and target-aware.

3 Method

3.1 Distributional Reduction with OT

Distributional Reduction (DistR) [Assel et al., 2025] formulates joint clustering and DR as an optimal transport problem between the original dataset and a reduced set of representative embeddings. Given input data $X \in \mathbb{R}^{n\times d}$, DistR seeks a reduced representation $Z \in \mathbb{R}^{m\times p}$ with $m \leq n$ and $p \leq d$ by solving a Gromov–Wasserstein (GW) OT problem between their respective metric structures.

Let C and $\bar{C}$ denote pairwise distance matrices in the input and embedding spaces respectively and $\mathbf{h}_X$ denote the weight vector typically assumed to be uniform, that is $\mathbf{h}_X = \frac{1}{n}\mathbf{1}_n$. DistR solves:

$$\min_{\substack{\mathbf{Z}\in\mathbb{R}^{m\times p}\\ \mathbf{T}\in\mathcal{U}(\mathbf{h}_X)\\ \mathbf{h}_Z\in\Sigma_m}} \sum_{ijkl} L_u(C_{ij},\bar{C}_{kl})T_{ik}T_{jl} \tag{DistR}$$

where T is a transport plan in $\mathcal{U}(\mathbf{h}_X) = \{T \in \mathbb{R}_+^{n\times m} : T\mathbf{1}_m = \mathbf{h}_X\}$ between the input and reduced representations, and L_u is an unsupervised, structural loss function comparing pairwise distances. The problem is optimized via alternating minimization over T and Z, where T is computed using standard GW solvers and Z is updated using gradient-based methods. When $m < n$, this yields

a reduced set of representative points weighted non-uniformly with h_Z, unifying clustering and dimensionality reduction within a single framework.

3.2 Supervised Distributional Reduction

In this section, we extend the Distributional Reduction Algorithm to the supervised setting. SDR optimizes a single unified objective over (Z, T), where alternating minimization is used to handle the non-convex coupling between the variables. The SDR objective solves the following two subproblems in the T and Z update steps:

- T-step: Solving a Fused-Gromov Wasserstein (FGW) alignment subproblem for T
- Z-step: Optimizing a supervised representation subproblem for Z

3.2.1 Optimal Transport with Supervised FGW

The Fused Gromov-Wasserstein (FGW) [Vayer et al., 2018] objective is formulated as follows:

$$FGW_\alpha(C, \bar{C}, Y, h, \bar{h}; T) = \min_{T \in \mathcal{U}(h,\bar{h})} (1-\alpha) \sum_{ij} L_s(y_i, g_j) T_{ij} + \alpha \sum_{ijkl} L_u(C_{ij}, \bar{C}_{kl}) T_{ik} T_{jl} \quad (1)$$

where, y_i and g_j are features of a point x_i in the input space and a point z_j in the embedding space respectively, L_s is a feature-level loss function, and α is a hyperparameter that controls the magnitudes of both these losses. In this work, we specifically choose the target values — regression targets or class labels present in a dataset as the features. This objective is intractable in conventional ML settings because prototype targets g_j are unavailable (in graph settings, by contrast, both graphs in question can have observed node-level labels or features). That makes $\mathbf{G} = (g_1, g_2, ..., g_m)$, additional parameters to be optimized as part of the objective. For notational convenience, we make the summation variables explicit, abbreviate the GW loss and rewrite the objective in Eqn.(1) as follows:

$$\mathcal{J}(T, Z, G) = (1-\alpha) \sum_{i=1}^{n} \sum_{j=1}^{m} L_s(y_i, g_j) T_{ij} + \alpha GW(Z; T) \quad \text{(S-FGW)}$$

The key insight is that we can analytically eliminate the prototype targets g_j from the S-FGW objective by solving for a partially minimized objective as described in Remark 3.1. Proposition 3.2 shows how we can eliminate prototype targets in closed form by leveraging the Bregman barycentric structure, yielding a tractable objective that depends only on (T, Z) for common loss functions.

Remark 3.1 (Elimination of Prototype Targets by Partial Minimization)**.**
The supervised FGW objective is separable in the prototype targets $G = \{g\}_{j=1}^m$. For fixed T and Z, each g_j can therefore be optimized independently as

$$g_j^*(T) = \underset{g \in \mathcal{G}}{arg\,min} \sum_{i=1}^{n} L_s(y_i, g) T_{ij} \quad (2)$$

Substituting these minimizers yield an equivalent reduced objective depending only on (T, Z),

$$\min_{T,Z,G} \mathcal{J}(T, Z, G) = \min_{T,Z} \mathcal{J}(T, Z, G^*(T)) \quad (3)$$

Proposition 3.2 (Optimal Prototypes for Bregman Losses; adapted from Banerjee et al. [2005] and Nielsen and Nock [2009])**.**
Let L_s be a Bregman divergence generated by a differentiable, strictly convex function ϕ on a convex domain $\mathcal{G} \subseteq \mathbb{R}^c$ and let $\pi_j := \Sigma_{i=1}^n T_{ij} > 0$. Then the optimal prototype target satisfies:

$$g_j^*(T) = \begin{cases} (\nabla\phi)^{-1}\left(\frac{1}{\pi_j}\sum_{i=1}^n T_{ij} \nabla\phi(y_i)\right), & \text{(g_j is first argument in L_s)} \\ \frac{1}{\pi_j}\sum_{i=1}^n T_{ij} y_i, & \text{(g_j is second argument in L_s)} \end{cases} \quad (4)$$

For standard supervised losses used in practice particularly the L2-loss for regression and the cross-entropy loss for classification, the optimal prototype targets reduce to:

$$g_j^*(T) = \frac{\sum_{i=1}^n T_{ij} y_i}{\sum_{i=1}^n T_{ij}} = \frac{1}{\pi_j} \sum_{i=1}^{n} T_{ij} y_i \quad (5)$$

The proof for Proposition 3.2 is provided in Appendix A. Substituting $g_j^*(T)$ yields a reduced objective depending only on (T, Z), which we refer to as the barycentric supervised FGW (BS-FGW) objective.

$$\mathcal{J}(T, Z) = (1-\alpha)\sum_{i=1}^{n}\sum_{j=1}^{m} L_s(y_i, g_j^*(T))T_{ij} + \alpha GW(Z; T) \tag{BSFGW}$$

The BSFGW problem is posed over $\mathcal{U}(\mathbf{h}_X, \mathbf{h}_Z)$ which fixes both marginals. In our setting, $\mathbf{h}_Z$ is an unknown quantity. We therefore optimize over the semi-relaxed set $\mathcal{U}(\mathbf{h}_X)$ which constrains only the first marginal as introduced in [Vincent-Cuaz et al., 2022]. The second marginal is left free and recovered from the solution as $\mathbf{h}_Z = \mathbf{T}^\top \mathbf{1}_n$ mirroring the DistR approach. The $\mathbf{T}$-subproblem is now defined as:

$$\mathbf{T}^* = \underset{\mathbf{T}\in\mathcal{U}(\mathbf{h}_X)}{\arg\min}(1-\alpha)\sum_{i=1}^{n}\sum_{j=1}^{m} L_s(y_i, g_j^*(T))T_{ij} + \alpha GW(Z; T) \tag{srBSFGW}$$

3.2.2 Supervised Representation Optimization

The DistR objective is elegant in the sense that it uses the same GW loss to optimize for both Z and T. However, this is not as straightforward in the supervised setting. The BSFGW objective introduces supervision indirectly via T, and the gradient of the supervision term with respect to Z is inert almost everywhere as described in the following remark.

Remark 3.3. *The partially minimized BSFGW objective exhibits two key limitations:*

(i) ***No direct supervision in representation updates -*** *For fixed T, the supervised term does not depend on Z, and the representation update is governed solely by the geometric term.*

(ii) ***Gradient-inertness of supervision for the representation variable holding a.e for the joint objective and for any first-order method -*** *Let $F(Z) = \min_T J(Z, T)$, then the gradient $\nabla F(Z)$ is practically inert wherever T^* is unique and F is differentiable. Supervision influences Z only through discrete switches on the coupling polytope which is not visible to a first-order optimization method.*

In regimes where $T^*(Z)$ is dominated by structural constraints or is locally insensitive to Z, the supervised gradient on Z is attenuated leading to weak alignment between Z and Y. Theoretical justification for Remark 3.3 is provided in Appendix C.1.

To address this, we introduce a direct dependence-maximization term, which complements FGW by providing strong gradients on Z. We inject an explicit supervision term using a normalized version of the HSIC score called the **Centred Kernel Alignment (CKA)** [Kornblith et al., 2019] which is defined over two variables Z and Y as:

$$\text{CKA}(Z, Y) = \frac{\langle K_Z, K_Y\rangle_{\mathcal{F}}}{||K_Z||_{\mathcal{F}}||K_Y||_{\mathcal{F}}} \tag{CKA}$$

where, K_Z and K_Y are kernel matrices over Z and Y respectively and $||.||_{\mathcal{F}}$ denotes the Frobenius norm. We use CKA as a dependence maximization objective to directly align representations with targets, building on prior work that employs similar dependence-based criteria (e.g., HSIC) for supervised dimensionality reduction [Barshan et al., 2011, Gretton et al., 2007, Zhang et al., 2017]. Unlike pointwise losses, CKA operates at the representation level and captures global dependence consistent with the global, structure-preserving perspective of OT. The normalized range of CKA makes it interpretable and offers easier tuning for the hyperparameter η when combined with the GW loss in the SDR objective. While a GW-style residual on $K_Z, K_{\tilde{Y}}$ seems like a natural option here, this can be counterproductive. For example, with discrete labels, K_Y is block constant and minimizing $||K_Z - K_{\tilde{Y}}||^2$ pushes within-class distances toward a constant resulting in class collapse. This destroys the intra-class geometry the GW term preserves. On the other hand, CKA maximizes dependence rather than kernel equality making it invariant to isotropic scaling and orthogonal transforms. This aligns Z with Y while leaving degrees of freedom for structure preservation.

We fix K_Z to be the PSD kernel whose decay profile matches that of the embedding affinity $\bar{C}$ (for example $\bar{C} \to$ Student-t $\implies K_Z \to$ RQ, $\bar{C} \to$ Gaussian $\implies K_Z \to$ RBF). We chose this

convention so CKA and GW gradients act on a consistent geometry making the method transferable to similar affinity-distance pairs. For $m < n$, we compute CKA using a coupling-induced projection $K_{\tilde{Y}\tilde{Y}} = T^\top K_{YY} T$, which defines target similarities in the reduced space. This can be viewed as a transport-weighted kernel compression technique that allows CKA computation in the prototype space. In Appendix C.2 we provide some empirical evidence on the effect of the additional CKA term compared to the vanilla BSFGW objective. Combining the CKA term with the BSFGW objective, the joint optimization objective for $\mathbf{Z}$ and $\mathbf{T}$ is given by:

$$J_{\text{SDR}}(Z, T, h_Z) = (1-\alpha) \sum_{i=1}^{n} \sum_{j=1}^{m} L_s(y_i, g_j^*(T)) T_{ij} + \alpha \, GW(Z; T) - \eta \, \text{CKA}(Z, \tilde{Y}) \quad \text{(SDR)}$$

By minimizing the negative of the CKA term we maximize the correlation between the variables Z and $\tilde{Y}$ with η controlling the relative contribution of the CKA term to the objective. We minimize J_{SDR} to obtain the embeddings Z, coupling matrix T and the weighting h_Z of the embeddings:

$$\mathbf{Z}^*, \mathbf{T}^*, \mathbf{h}_Z^* = \underset{\substack{\mathbf{Z} \in \mathbb{R}^{m \times p} \\ \mathbf{T} \in \mathcal{U}(\mathbf{h}_X) \\ \mathbf{h}_Z \in \Sigma_m}}{\arg\min} \; J_{\text{SDR}}(Z, T, h_Z) \tag{6}$$

We adopt an inexact alternating scheme w.r.t T to solve for the SDR objective. In particular, the T-step optimizes the srBSFGW objective ignoring the CKA term. This design is intentional for two reasons: (i) including the CKA term in the transport subproblem would break the structure required for efficient OT solvers and introduce high-order, normalized dependencies on T; and (ii) CKA is designed to provide direct supervision on the representation Z, addressing the gradient inertness of the supervision term without requiring gradients through the transport solver. We show that the worst-case mismatch of a T-update is bounded by the user-chosen parameter η in Proposition 3.4.

Proposition 3.4 (Upper bound on worst-case mismatch of T-update)**.** *Optimizing the srBSFGW objective instead of the complete SDR objective results in a worst-case mismatch for the T-update at most by a value of η.*

$$J(T^*) \leq J(T) + \eta \tag{7}$$

Proof. Let,

$$J(T) = F(T) - \eta C(T) \tag{8}$$

where F is the srBSFGW objective and $C(T) = \text{CKA}(Z, Y(T)) \in [0, 1]$. The T-step returns $T^* \in \underset{T}{\arg\min} \, F$, hence for any feasible T:

$$\begin{aligned} J(T^*) &= F(T^*) - \eta C(T^*) && (9)\\ &\leq F(T) - \eta C(T^*) && (10)\\ &= J(T) + \eta(C(T) - C(T^*)) && (11)\\ &\leq J(T) + \eta && (12) \end{aligned}$$

□

Each T-step thus decreases the complete SDR objective up to an additive error uniformly bounded by the user-chosen hyperparameter η, and the Z-step is standard descent on the full objective. The srBSFGW objective is solved using a CG solver (Jaggi [2013], Vincent-Cuaz et al. [2022]) to optimize for T which will consequently recover $\mathbf{h}_Z = T^\top \mathbf{1}_n$ as the second marginal. The Z-step optimizes the sum of the GW and CKA terms using SGD optimizers like Adam [Kingma and Ba, 2017]. We also implemented a safeguarded variant of this algorithm with an accept-reject step gated on the SDR objective which accepts a T-update only if it does not increase the full SDR objective. The results for this variant are shown in Appendix E. Despite being more stable, such a variant tends to perform empirically worse than the inexact alternating scheme since, a reject-only variant can be stuck in a rejection loop. It is worth noting that SDR inherits the same runtime complexity as DistR which is $\mathcal{O}(nm^2 + mn^2)$ according to Proposition 1 in [Peyré et al., 2016].

4 Mapping Estimation for SDR

While SDR learns embeddings that are strongly aligned with targets, these representations are not guaranteed to admit a functional mapping from the input space. This lack of an explicit mapping prevents SDR from being used in downstream predictive settings, where embeddings for unseen inputs must be computed. To enable out-of-sample generalization, we therefore seek embeddings Z that are not only target-aware but also approximately representable as a kernel map from the inputs. Concretely, we enforce that Z lies close to the image of a function in the RKHS induced by a kernel K, leading to the approximation $Z \approx KL$, where L parametrizes the mapping. This perspective allows us to learn embeddings that retain the advantages of SDR while ensuring they can be consistently extended to unseen data. Consequently, this allows SDR to be used as part of a predictive pipeline.

We take inspiration from Perrot et al. [2016] to propose a modified objective which augments the original SDR objective with a projection-consistency term as follows

$$J_{\text{SDR-OOS}}(Z, T, L) = J_{\text{SDR}}(Z, T) + \frac{\mu}{2}||TZ - KL||_F^2 + \frac{\lambda_L}{2}\text{Tr}(L^\top KL) \tag{SDR-OOS}$$

where, $K \in \mathbb{R}^{n\times n}$ is the covariance matrix of the training points X according to a kernel function (typically RBF), $L \in \mathbb{R}^{n\times p}$ is the transformation map, and λ_L is a regularization parameter for the transformation map L with the last term defining the RKHS norm of L. μ is a regularization parameter for Z that forces Z to lie close to the RKHS representable set of the kernel function used to compute K. The solution for the L-subproblem can be obtained in closed form and is given by Proposition 4.1.

Proposition 4.1 (Closed form solution for L-subproblem)**.** *The unique minimizer of the L-subproblem in $J_{\text{SDR-OOS}}(\cdot, \cdot, L)$ is given by*

$$L^* = (\mu K + \lambda_L I)^{-1}\mu TZ \tag{13}$$

The proof is provided in Appendix B and is a simple least-squares derivation. The SDR-OOS objective can be similarly solved with BCD by having three alternating updates for T, Z and L. After updating L, we perform a relaxation step that moves Z towards the RKHS-representable set defined by the current projection KL^*, while also retaining flexibility introduced by the SDR objective. This update can be written down as follows:

$$Z' = (1 - \beta)Z + \beta KL^* \tag{14}$$

where β controls the trade-off between preserving target-aligned structure in the learned embeddings and enforcing consistency with the RKHS projection for improved generalization. The soft update can be extended to the $m \neq n$ (for SDR typically $m < n$) case analogously as,

$$Z' = (I + \beta T^{\text{T}}T)^{-1}(Z + \beta T^{\text{T}}KL^*) \tag{15}$$

4.1 SDR for Non-Stationary Kernel Construction in GPs

Gaussian Processes (GPs) with stationary kernels are limited in their ability to capture input-dependent variations in function behaviour. A common approach to address this is Deep Kernel Learning (DKL) [Wilson et al., 2016], where a neural network is jointly trained with the GP to learn a feature representation that induces non-stationarity. While effective, this approach relies on parametric feature extractors and joint optimization of the network parameters and the GP hyperparameters, which can be sensitive to initialization and challenging to train in low-data regimes. Deep Gaussian Processes (DGPs) [Damianou and Lawrence, 2013] provide another way to introduce non-stationarity by modelling each intermediate representation in a deep belief network with a data-driven GP prior but are expensive to train and converge much slower in practice.

SDR provides an alternative mechanism for constructing non-stationary kernels by learning target-aware representations through optimal transport and dependence maximization in the pure DR setting ($p < d, m = n$). When a stationary kernel (e.g., RBF) is applied in the SDR embedding space, the induced kernel in the original input space becomes data-dependent and non-stationary. The GP prior with an SDR induced kernel can be written as,

$$f \sim \mathcal{GP}(m(\cdot), \psi(z(\cdot), z(\cdot))) \tag{SDR-GP}$$

where, m is a mean function, ψ is a stationary kernel like the RBF. The mapping $z(\cdot)$ uses the projection map L optimized by the SDR-OOS objective to project an unseen point x^* and is given by,

$$z(x^*) = K(x^*, X)L \tag{16}$$

where K is also a stationary kernel like RBF acting in the input space.

Unlike DKL, this approach decouples representation learning from GP training and leverages a target-aware objective that explicitly aligns representations with targets. The SDR-OOS objective equipped with a projection map allows SDR to serve as a plug-in feature extractor for GPs, providing a non-parametric route to non-stationary kernel design grounded in distributional alignment.

5 Experiments

5.1 Distributional Reduction Benchmarks

To assess the effect of incorporating supervision, we follow the evaluation protocol used in prior joint clustering and dimensionality reduction methods, particularly DistR. We use three classification datasets namely: COIL-20 [Nene et al., 1996], Fashion-MNIST [Xiao et al., 2017] and SNAREseq [Chen et al., 2019] for the experiments in this subsection. Further details of the datasets are provided in Appendix D. Since label information Y is explicitly used during optimization, these metrics do not reflect unsupervised clustering performance, but rather the extent to which the learned representation captures target-relevant structure. Accordingly, we report homogeneity score, k-means normalized mutual information (NMI), and silhouette score, interpreting them as measures of label consistency and semantic coherence of the embeddings.

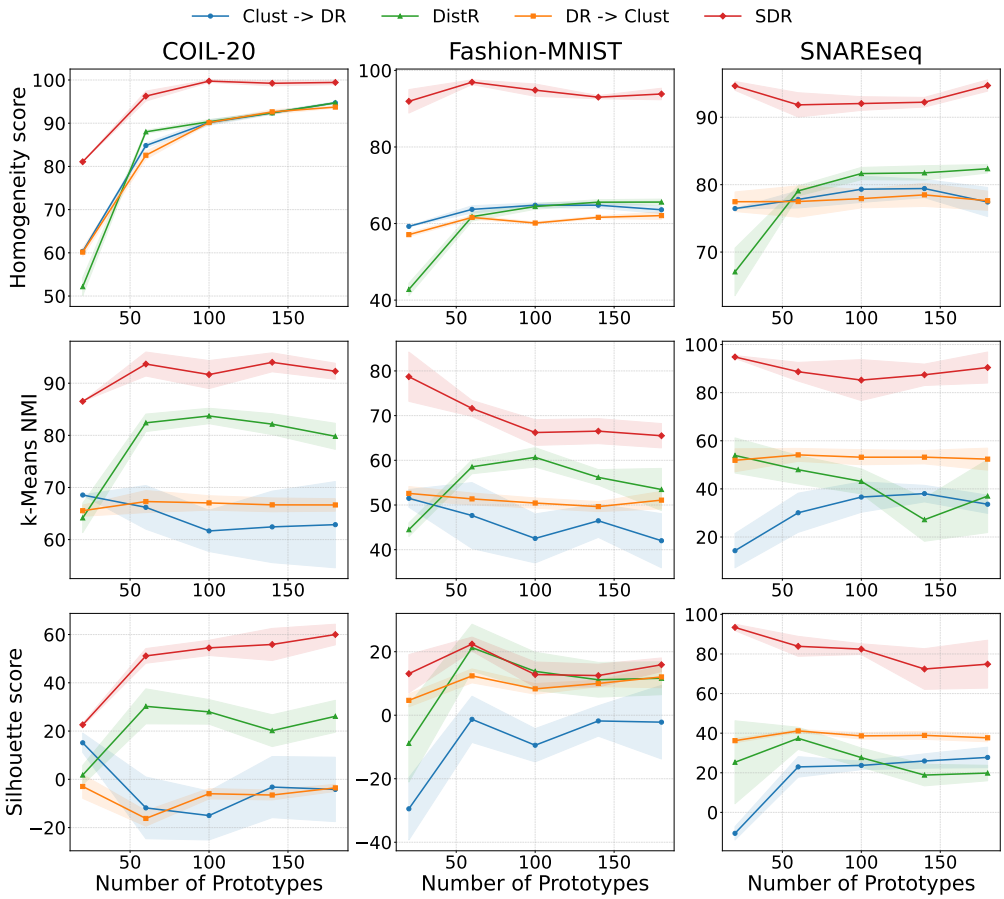


Figure 2: Scores (x100) across varying number of prototypes in $\mathbb{R}^2$ for all 4 methods.

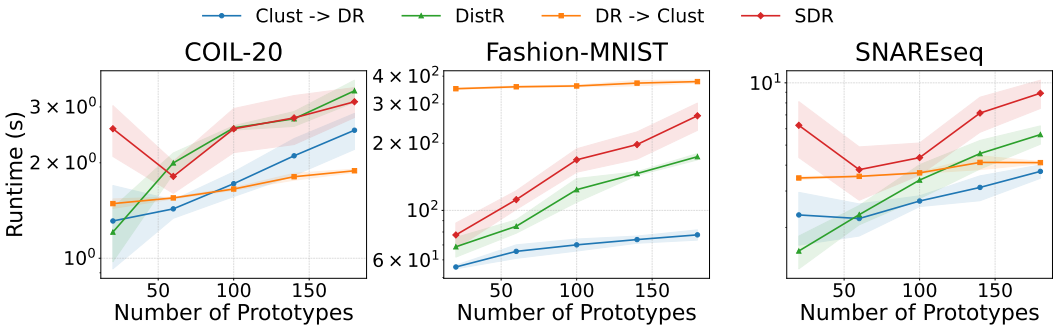


Figure 3: Runtimes across varying number of prototypes in $\mathbb{R}^2$ for all 4 methods.

We compare SDR against DistR, Cluster-then-DR, and DR-then-Cluster to isolate the effect of incorporating supervision within a distributional reduction pipeline. The results of this benchmark are shown in Figure 2, with runtimes plotted in Figure 3. SDR achieves comparable runtime to DistR with a modest computational overhead. Additional experimental details are provided in Appendix D. An ablation study on the η hyperparameter has been performed and the results are reported in Appendix G.2. Since SDR jointly reduces the number of samples, predictive evaluation on held-out data is not directly applicable. Instead, we focus on representation-level properties in these experiments, with predictive performance evaluated separately in Section 5.2 under the context of kernel learning when SDR reduces to a pure DR method.

5.2 Kernel Learning Benchmarks

In this set of experiments, we try to evaluate the effectiveness of SDR as a representation learning method in the context of non-stationary kernel design for GPs. We evaluate on small to medium-sized datasets ($n \leq 2000$) where training flexible, non-stationary GP models can be challenging from an optimization perspective. We focus on this regime where GPs remain practical models, since predictive posterior computation and marginal likelihood evaluation scales as $\mathcal{O}(N^3)$. SDR exhibits the same $\mathcal{O}(N^3)$ scaling in the pure DR setting inheriting this computational complexity directly from the DistR framework.

We test SDR-GP against supervised pure DR methods like neighbour Components Analysis [Goldberger et al., 2004], Supervised UMAP [McInnes et al., 2020] and Kernel Supervised PCA [Barshan et al., 2011] (dubbed as NCA-GP, KSPCA-GP and UMAP-GP respectively). We also compare SDR-GP with DKL and DGP models that are typically considered for non-stationary kernel construction in the GP literature. We evaluate these models across three standard UCI benchmark datasets for regression: Boston Housing, Energy Efficiency and Concrete Compressive Strength [Dua and Graff, 2017] along with two image classification benchmark datasets: MNIST [LeCun and Cortes, 2010] and COIL-20 [Nene et al., 1996]. The Mean Log Likelihood (MLL) and the Mean Squared Error (MSE) for the regression tasks are presented in Table 1. The Mean Log Probability (MLP) for the correct class and Accuracy (ACC) for the classification tasks are reported in Table 2. We also test on a variant of DKL that uses pre-trained feature extractor networks in Appendix K.

	Boston		Energy		Concrete	
	MLL (↑)	MSE (↓)	MLL (↑)	MSE (↓)	MLL (↑)	MSE (↓)
NCA-GP	-0.76 ± 0.31	0.25 ± 0.11	-0.81 ± 0.38	0.29 ± 0.38	-1.01 ± 0.16	0.45 ± 0.12
KSPCA-GP	-0.99 ± 0.27	0.38 ± 0.13	-0.67 ± 0.05	0.22 ± 0.03	-1.28 ± 0.16	0.79 ± 0.23
UMAP-GP	-1.32 ± 0.43	0.39 ± 0.11	-0.53 ± 0.08	0.13 ± 0.06	-1.38 ± 0.19	0.67 ± 0.18
DGP	-1.18 ± 0.09	0.46 ± 0.18	-0.88 ± 0.04	0.12 ± 0.01	-1.10 ± 0.05	0.33 ± 0.06
DKL	-0.42 ± 0.21	0.13 ± 0.07	-0.45 ± 0.03	0.09 ± 0.02	-0.65 ± 0.11	0.21 ± 0.05
SDR-GP (ours)	-0.32 ± 0.18	0.14 ± 0.08	**-0.37 ± 0.02**	**0.05 ± 0.01**	**-0.40 ± 0.07**	**0.13 ± 0.02**

Table 1: Test results on UCI regression datasets based on five random seeds

	MNIST-2k		COIL-20	
	MLP (↑)	ACC (↑)	MLP (↑)	ACC (↑)
NCA-GP	-0.31 ± 0.04	0.91 ± 0.02	-0.15 ± 0.02	0.98 ± 0.01
KSPCA-GP	-0.46 ± 0.05	0.88 ± 0.02	-0.13 ± 0.02	0.97 ± 0.01
UMAP-GP	-1.05 ± 0.06	0.82 ± 0.00	-0.90 ± 0.22	0.83 ± 0.04
DKL	-0.30 ± 0.05	0.90 ± 0.02	-0.09 ± 0.02	0.99 ± 0.00
SDR-GP (ours)	**-0.23 ± 0.02**	**0.95 ± 0.00**	**-0.07 ± 0.02**	0.99 ± 0.00

Table 2: Test results on image datasets based on five random seeds.

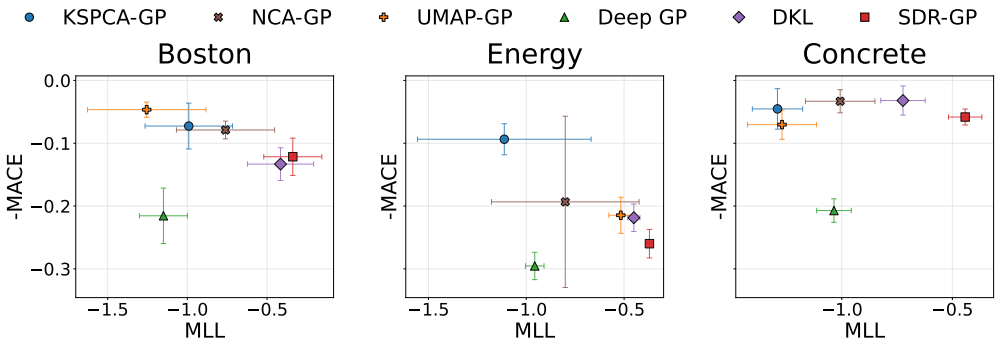


Figure 4: Visualization of different methods depicting the tradeoff between predictive performance in terms of Mean Log Likelihood (MLL) and uncertainty calibration in terms of Mean Absolute Calibration Error (MACE) for the three regression datasets. An ideal model would typically exist in the top-right of this plot which signifies good predictive performance and good uncertainty calibration.

We evaluate the models not only in terms of predictive performance but also in terms of uncertainty calibration, a key property that makes GPs attractive. Figure 4 shows how the various models behave in terms of predictive performance and uncertainty calibration. While the SDR objective is aimed at improving predictive performance, a GP trained on SDR representations also provides reasonably calibrated uncertainties (more details in Appendix J). The details of the datasets, models and hyperparameters used in these experiments are presented in Appendix I. An ablation study on the β hyperparameter was performed and the results are reported in Appendix G.1.

6 Conclusion and Future Work

We introduced Supervised Distributional Reduction (SDR), a framework that integrates optimal transport with dependence maximization to learn compact, target-aware representations. By identifying a gradient inertness problem in the supervision term of the FGW formulation and addressing it through a representation-level objective, SDR provides a mechanism for stronger alignment between embeddings and targets while preserving intrinsic data geometry. We further proposed an out-of-sample extension via RKHS projection that provides an explicit transformation map to project unseen points to the embedding space. This allows SDR to be used in predictive pipelines while also regularizing the optimization of the embeddings. Empirical results demonstrate that SDR improves representation quality and yields strong downstream performance, highlighting the effectiveness of combining transport-based structure with explicit supervision.

Despite these advancements, we identify a few key areas for improvement. The current version of the algorithm does not provide any convergence guarantees from an optimization theory perspective. Future work can explore optimization alternatives like KL-Mirror Descent for SDR optimization and establish convergence guarantees for the algorithm. Another useful avenue could be to employ low-rank factorizations [Scetbon et al., 2023, Assel et al., 2025] to scale SDR to larger datasets. Exploring alternative dependence measures [Zhang et al., 2017] and developing more expressive projection operators for the out-of-sample extension can improve predictive alignment further.

References

Hugues Van Assel, Titouan Vayer, Rémi Flamary, and Nicolas Courty. Snekhorn: Dimension reduction with symmetric entropic affinities, 2023. URL https://arxiv.org/abs/2305.13797.

Hugues Van Assel, Cédric Vincent-Cuaz, Nicolas Courty, Rémi Flamary, Pascal Frossard, and Titouan Vayer. Distributional reduction: Unifying dimensionality reduction and clustering with gromov-wasserstein, 2025. URL https://arxiv.org/abs/2402.02239.

Arindam Banerjee, Srujana Merugu, Inderjit S. Dhillon, and Joydeep Ghosh. Clustering with bregman divergences. *J. Mach. Learn. Res.*, 6:1705–1749, December 2005. ISSN 1532-4435.

Elnaz Barshan, Ali Ghodsi, Zohreh Azimifar, and Mansoor Zolghadri Jahromi. Supervised principal component analysis: Visualization, classification and regression on subspaces and submanifolds. *Pattern Recognition*, 44(7):1357–1371, 2011. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2010.12.015. URL https://www.sciencedirect.com/science/article/pii/S0031320310005819.

Edwin V. Bonilla, Kian Ming A. Chai, and Christopher K. I. Williams. Multi-task gaussian process prediction. In *Proceedings of the 21st International Conference on Neural Information Processing Systems*, NIPS'07, page 153–160, Red Hook, NY, USA, 2007. Curran Associates Inc. ISBN 9781605603520.

Song Chen, Blue Lake, and Kun Zhang. High-throughput sequencing of the transcriptome and chromatin accessibility in the same cell. *Nature Biotechnology*, 37, 12 2019. doi: 10.1038/s41587-019-0290-0.

Youngseog Chung, Ian Char, Han Guo, Jeff Schneider, and Willie Neiswanger. Uncertainty toolbox: an open-source library for assessing, visualizing, and improving uncertainty quantification, 2021. URL https://arxiv.org/abs/2109.10254.

Marco Cuturi and Arnaud Doucet. Fast computation of wasserstein barycenters. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 685–693, Bejing, China, 22–24 Jun 2014. PMLR. URL https://proceedings.mlr.press/v32/cuturi14.html.

Andreas Damianou and Neil D. Lawrence. Deep Gaussian processes. In Carlos M. Carvalho and Pradeep Ravikumar, editors, *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*, volume 31 of *Proceedings of Machine Learning Research*, pages 207–215, Scottsdale, Arizona, USA, 29 Apr–01 May 2013. PMLR. URL https://proceedings.mlr.press/v31/damianou13a.html.

Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2):179–188, 1936. doi: https://doi.org/10.1111/j.1469-1809.1936.tb02137.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1469-1809.1936.tb02137.x.

Aden Forrow, Jan-Christian Hütter, Mor Nitzan, Philippe Rigollet, Geoffrey Schiebinger, and Jonathan Weed. Statistical optimal transport via factored couplings. In Kamalika Chaudhuri and Masashi Sugiyama, editors, *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pages 2454–2465. PMLR, 16–18 Apr 2019. URL https://proceedings.mlr.press/v89/forrow19a.html.

Jerome H. Friedman. Multivariate Adaptive Regression Splines. *The Annals of Statistics*, 19(1):1 – 67, 1991. doi: 10.1214/aos/1176347963. URL https://doi.org/10.1214/aos/1176347963.

Jacob R. Gardner, Geoff Pleiss, David Bindel, Kilian Q. Weinberger, and Andrew Gordon Wilson. Gpytorch: blackbox matrix-matrix gaussian process inference with gpu acceleration. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 7587–7597, Red Hook, NY, USA, 2018. Curran Associates Inc.

Jacob Goldberger, Geoffrey E Hinton, Sam Roweis, and Russ R Salakhutdinov. Neighbourhood components analysis. In L. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 17. MIT Press, 2004. URL https://proceedings.neurips.cc/paper_files/paper/2004/file/42fe880812925e520249e808937738d2-Paper.pdf.

Arthur Gretton, Kenji Fukumizu, Choon Teo, Le Song, Bernhard Schölkopf, and Alex Smola. A kernel statistical test of independence. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2007. URL https://proceedings.neurips.cc/paper_files/paper/2007/file/d5cfead94f5350c12c322b5b664544c1-Paper.pdf.

Tom Heskes. Bias-variance decompositions: the exclusive privilege of bregman divergences. *ArXiv*, abs/2501.18581, 2025. URL https://api.semanticscholar.org/CorpusID:275993629.

Martin Jaggi. Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 427–435, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR. URL https://proceedings.mlr.press/v28/jaggi13.html.

L. Kantorovitch. On the translocation of masses. *Management Science*, 5(1):1–4, 1958. doi: 10.1287/mnsc.5.1.1. URL https://doi.org/10.1287/mnsc.5.1.1.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Sungyeon Kim, Dongwon Kim, Minsu Cho, and Suha Kwak. Proxy anchor loss for deep metric learning, 2020. URL https://arxiv.org/abs/2003.13911.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL https://arxiv.org/abs/1412.6980.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited, 2019. URL https://arxiv.org/abs/1905.00414.

Volodymyr Kuleshov, Nathan Fenner, and Stefano Ermon. Accurate uncertainties for deep learning using calibrated regression. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2796–2804. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/kuleshov18a.html.

Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010. URL http://yann.lecun.com/exdb/mnist/.

Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction, 2020. URL https://arxiv.org/abs/1802.03426.

Dimitrios Milios, Raffaello Camoriano, Pietro Michiardi, Lorenzo Rosasco, and Maurizio Filippone. Dirichlet-based gaussian processes for large-scale calibrated classification, 2018. URL https://arxiv.org/abs/1805.10915.

Gaspard Monge. *Mémoire sur la théorie des déblais et des remblais*. De l'Imprimerie Royale, 1781.

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

Frank Nielsen and Richard Nock. Sided and symmetrized bregman centroids. *IEEE Transactions on Information Theory*, 55(6):2882–2904, 2009. doi: 10.1109/TIT.2009.2018176.

Sebastian W. Ober, Carl E. Rasmussen, and Mark van der Wilk. The promises and pitfalls of deep kernel learning, 2021. URL https://arxiv.org/abs/2102.12108.

Michaël Perrot, Nicolas Courty, Rémi Flamary, and Amaury Habrard. Mapping estimation for discrete optimal transport. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/file/26f5bd4aa64fdadf96152ca6e6408068-Paper.pdf.

Gabriel Peyré, Marco Cuturi, and Justin Solomon. Gromov-wasserstein averaging of kernel and distance matrices. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 2664–2672. JMLR.org, 2016.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 11 2005. ISBN 9780262256834. doi: 10.7551/mitpress/3206.001.0001. URL https://doi.org/10.7551/mitpress/3206.001.0001.

Hugh Salimbeni and Marc Peter Deisenroth. Doubly stochastic variational inference for deep gaussian processes. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 4591–4602, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

Atsushi Sato and Keiji Yamada. Generalized learning vector quantization. In D. Touretzky, M.C. Mozer, and M. Hasselmo, editors, *Advances in Neural Information Processing Systems*, volume 8. MIT Press, 1995. URL https://proceedings.neurips.cc/paper_files/paper/1995/file/9c3b1830513cc3b8fc4b76635d32e692-Paper.pdf.

Meyer Scetbon, Gabriel Peyré, and Marco Cuturi. Linear-time gromov wasserstein distances using low rank couplings and costs, 2023. URL https://arxiv.org/abs/2106.01128.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 4080–4090, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

Kevin Tran, Willie Neiswanger, Junwoong Yoon, Qingyang Zhang, Eric Xing, and Zachary W. Ulissi. Methods for comparing uncertainty quantifications for material property predictions, 2020. URL https://arxiv.org/abs/1912.10066.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008. URL http://jmlr.org/papers/v9/vandermaaten08a.html.

Titouan Vayer. On sparsity, extremal structure, and monotonicity properties of wasserstein and gromov-wasserstein optimal transport plans, 2026. URL https://arxiv.org/abs/2602.16265.

Titouan Vayer, Nicolas Courty, Romain Tavenard, Laetitia Chapel, and Rémi Flamary. Optimal transport for structured data with application on graphs. In *International Conference on Machine Learning*, 2018. URL https://api.semanticscholar.org/CorpusID:59523642.

Cédric Vincent-Cuaz, Rémi Flamary, Marco Corneli, Titouan Vayer, and Nicolas Courty. Semi-relaxed gromov-wasserstein divergence with applications on graphs, 2022. URL https://arxiv.org/abs/2110.02753.

Kiri Wagstaff, Claire Cardie, Seth Rogers, and Stefan Schrödl. Constrained k-means clustering with background knowledge. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, page 577–584, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. ISBN 1558607781.

Kilian Q Weinberger, John Blitzer, and Lawrence Saul. Distance metric learning for large margin nearest neighbor classification. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems*, volume 18. MIT Press, 2005. URL https://proceedings.neurips.cc/paper_files/paper/2005/file/a7f592cef8b130a6967a90617db5681b-Paper.pdf.

Andrew Gordon Wilson, Zhiting Hu, Ruslan Salakhutdinov, and Eric P. Xing. Deep kernel learning. In Arthur Gretton and Christian C. Robert, editors, *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, volume 51 of *Proceedings of Machine Learning Research*, pages 370–378, Cadiz, Spain, 09–11 May 2016. PMLR. URL https://proceedings.mlr.press/v51/wilson16.html.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017. URL https://arxiv.org/abs/1708.07747.

Eric P. Xing, Andrew Y. Ng, Michael I. Jordan, and Stuart Russell. Distance metric learning, with application to clustering with side-information. In *Proceedings of the 16th International Conference on Neural Information Processing Systems*, NIPS'02, page 521–528, Cambridge, MA, USA, 2002. MIT Press.

Qinyi Zhang, Sarah Filippi, Arthur Gretton, and Dino Sejdinovic. Large-scale kernel methods for independence testing. *Statistics and Computing*, 28(1):113–130, January 2017. ISSN 1573-1375. doi: 10.1007/s11222-016-9721-7. URL http://dx.doi.org/10.1007/s11222-016-9721-7.

A Proof for Proposition 2 - Optimal Prototype Targets for Bregman Losses

We restate this classical result from Theorem 3.2 in Nielsen and Nock [2009] for the left-sided Bregman centroid (dual form) and Proposition 1 in Banerjee et al. [2005] for the right-sided Bregman centroids (primal form). We provide a self-contained proof in our notation, since the specific form involving the transport coupling T is needed for the elimination argument in Remark 3.1 and the BSFGW objective.

Case 1 (g_j as first argument - dual form): The definition of a Bregman divergence is given by,

$$D_\phi(g, y_i) = \phi(g) - \phi(y_i) - \langle \nabla\phi(y_i), g - y_i \rangle \tag{17}$$

Consider the supervision loss term,

$$F(g) := \sum_{i=1}^{n} w_i D_\phi(g, y_i) \tag{18}$$

Expanding the loss term we get,

$$F(g) = \sum_{i=1}^{n} w_i \phi(g) - \sum_{i=1}^{n} w_i \phi(y_i) - \sum_{i=1}^{n} w_i \langle \nabla\phi(y_i), g - y_i \rangle \tag{19}$$

Setting $g = g_j$, $w_i = T_{ij}$, $\sum_{i=1}^{n} T_{ij} = \pi_j$ we have,

$$F(g_j) = \pi_j \phi(g_j) - \sum_{i=1}^{n} T_{ij} \phi(y_i) - \langle \sum_{i=1}^{n} T_{ij} \nabla\phi(y_i), g_j \rangle + \sum_{i=1}^{n} T_{ij} \langle \nabla\phi(y_i), y_i \rangle \tag{20}$$

Grouping the terms that only depend on g_j we get,

$$F(g_j) = \pi_j \phi(g_j) - \langle \sum_{i=1}^{n} T_{ij} \nabla\phi(y_i), g_j \rangle + C \tag{21}$$

where C is constant in g_j. Hence minimizing $F(g)$ is equivalent to minimizing

$$G(g_j) = \pi_j \phi(g_j) - \langle \sum_{i=1}^{n} T_{ij} \nabla\phi(y_i), g_j \rangle \tag{22}$$

Since ϕ is differentiable and strictly convex, G is also differentiable and strictly convex and its unique minimizer is characterized by the first-order condition

$$\nabla G(g_j) = \pi_j \nabla\phi(g_j) - \sum_{i=1}^{n} T_{ij} \nabla\phi(y_i) = 0 \tag{23}$$

Therefore,

$$\nabla\phi(g_j^*) = \frac{1}{\pi_j} \sum_{i=1}^{n} T_{ij} \nabla\phi(y_i) \tag{24}$$

Applying $(\nabla\phi)^{-1}$ both sides yields,

$$g_j^* = (\nabla\phi)^{-1} \left(\frac{1}{\pi_j} \sum_{i=1}^{n} T_{ij} \nabla\phi(y_i) \right) \tag{25}$$

Case 2 (g_j as second argument - primal form): Consider the objective

$$F(g) := \sum_{i=1}^{n} w_i D_\phi(y_i, g) \tag{26}$$

Expanding the objective we get,

$$F(g) = \sum_{i=1}^{n} w_i \phi(y_i) - \sum_{i=1}^{n} w_i \phi(g) - \sum_{i=1}^{n} w_i \langle \nabla \phi(g), y_i - g \rangle \tag{27}$$

Setting $g = g_j$, $w_i = T_{ij}$, $\sum_{i=1}^{n} T_{ij} = \pi_j$ we have,

$$F(g_j) = \sum_{i=1}^{n} T_{ij} \phi(y_i) - \pi_j \phi(g_j) - \langle \nabla \phi(g_j), \sum_{i=1}^{n} T_{ij} y_i - \pi_j g_j \rangle \tag{28}$$

Using the definition of $\bar{y}$ (weighted average of the targets), given by $\Sigma_{i=1}^{n} T_{ij} y_i = \pi_j \bar{y}$, we get

$$F(g_j) = \sum_{i=1}^{n} T_{ij} \phi(y_i) - \pi_j \phi(g_j) - \pi_j \langle \nabla \phi(g_j), \bar{y} - g \rangle \tag{29}$$

$$= \sum_{i=1}^{n} T_{ij} \phi(y_i) + \pi_j (-\phi(g_j) - \langle \nabla \phi(g_j), \bar{y} - g_j \rangle) \tag{30}$$

$$= \sum_{i=1}^{n} T_{ij} \phi(y_i) - \pi_j \phi(\bar{y}) + \pi_j (\phi(\bar{y}) - \phi(g_j) - \langle \nabla \phi(g_j), \bar{y} - g_j \rangle) \tag{31}$$

Recognizing the Bregman Divergence in the last term

$$F(g_j) = \sum_{i=1}^{n} T_{ij} \phi(y_i) - \pi_j \phi(\bar{y}) + \pi_j D_\phi(\bar{y}, g_j) \tag{32}$$

The first two terms do not depend on g_j. Therefore minimizing $F(g_j)$ over g_j is equivalent to minimizing $D_\phi(\bar{y}, g_j)$. Since $D_\phi(\bar{y}, g_j) \geq 0$ for all $g \in \mathcal{G}$, with equality iff $g = \bar{y}$ (by strict convexity of ϕ), the unique minimizer is

$$g_j^* = \bar{y} \tag{33}$$

Hence, the optimal prototype targets are given by the expression

$$g_j^* = \frac{1}{\pi_j} \sum_{i=1}^{n} T_{ij} y_i \tag{34}$$

Now, we derive the closed form expression of the optimal prototype targets for two commonly used loss functions namely: the L2-loss and the Cross-Entropy loss.

A.1 $L_s \rightarrow$ L2-loss

We can make use of Lemma 2 to derive the optimal prototype targets when L_s is the L2-loss. The L2-loss is symmetric w.r.t to the targets and prototype targets and so it does not really make a difference on which case of Lemma 2 we employ to arrive at the expression for g_j^*. In this case, we will just use Case 1 of Lemma 2 assuming g_j is the first argument of the L_s loss function. For L2-loss we have

$$\phi(y) = \frac{1}{2} \|y\|_2^2 \tag{35}$$

The gradient of ϕ is then given by,

$$\nabla \phi(y) = y \tag{36}$$

$$\Rightarrow (\nabla \phi)^{-1}(y) = y \tag{37}$$

Lemma 2 then gives,

$$g_j^*(T) = (\nabla \phi)^{-1} \left(\frac{1}{\pi_j} \sum_{i=1}^{n} T_{ij} \nabla \phi(y_i) \right) \tag{38}$$

$$= \frac{1}{\pi_j} \sum_{i=1}^{n} T_{ij} y_i \tag{39}$$

This just means that the prototype targets are nothing but a weighted sum of all targets weighted according to the coupling matrix T.

A.2 $L_s \rightarrow$ Cross-Entropy Loss

The modified Cross-Entropy (CE) loss as described in Heskes [2025] is given by

$$L_s(y, g) = \sum_{k=1}^{c} y_k \log\left(\frac{y_k}{g_k}\right) + \sum_{k=1}^{c} g_k - \sum_{k=1}^{c} y_k \qquad \text{(Modified CE-Loss)}$$

The modified CE loss is a valid Bregman divergence with the following generator function

$$\phi(u) = \sum_{k=1}^{c} u_k \log u_k - \sum_{k=1}^{c} u_k \tag{40}$$

The modified CE loss takes g as the second argument and thus Case 2 of Lemma 2 applies. As a direct consequence, the optimal prototype targets are given by

$$g_j^* = \frac{1}{\pi_j} \sum_{i=1}^{n} T_{ij} y_i \tag{41}$$

This means that the prototypes in the embedding space carry soft class labels unlike the points in the input space which might have hard labels when one-hot encoded.

Note: The result established above is general and applies when the domain of g is extended from the probability simplex Δ^{c-1} to the unit cube $[0, 1]^c$ i.e even when we ignore the simplex normalization constraint $\Sigma_{k=1}^{c} g_k = 1$. In the case where y is a (one-hot encoded) label vector and g satisfies the simplex normalization constraint, the modified CE loss collapses to the classic CE loss.

B Closed form solution for the L-subproblem

The L-subproblem is given by,

$$J_{\text{SDR-OOS}}(\cdot, \cdot, L) = \psi(L) = \frac{\mu}{2} ||TZ - KL||_F^2 + \frac{\lambda_L}{2} \text{Tr}(L^\top K L) \tag{42}$$

Note that the L-subproblem is a quadratic function in L. The L-subproblem therefore is the sum of a least-squares term with an RKHS norm-regularizer. Since $\lambda_L > 0$, the Hessian with respect to L is positive definite. Hence, the L-subproblem is strongly convex and admits a unique minimizer. Differentiating $\psi(L)$ w.r.t L and using the first-order optimality condition we have,

$$\nabla_L \psi(L) = \mu K(KL - TZ) + \lambda_L KL = 0 \tag{43}$$

With some basic algebraic manipulations, the minimizer is given by,

$$L^* = (\mu K + \lambda_L I)^{-1} \mu TZ \tag{44}$$

Estimating the OOS map reduces to vector-valued kernel ridge regression from the input kernel K to the SDR representation, providing a stable and well-posed projection operator for unseen points. Without loss of generality, we can set $\mu = 1$ since it can be absorbed into the regularization parameter λ_L. In the case that the optimal coupling matrix is a permutation matrix under uniform marginals, the expression for L^* simplifies to the usual kernel ridge form,

$$L^* = (K + \lambda_L I)^{-1} Z \tag{45}$$

C FGW Gradient Inertness

C.1 Mathematical intuition for the gradient inertness in FGW

In this section, we provide some mathematical intuition into the observation made in Remark 3.3. Recall the partially minimized BS-FGW objective,

$$\mathcal{J}(T, Z) = (1 - \alpha) S(T) + \alpha GW(T; Z) \tag{46}$$

where $S(T)$ is the supervision loss term defined as,

$$S(T) = \sum_{i=1}^{n} \sum_{j=1}^{m} L_s(y_i, g_j^*(T)) T_{ij} \tag{47}$$

For part (i), observe that once the prototype targets $g_j^*(T)$ are replaced with the closed forms from Proposition 3.2, the supervised term $S(T)$ depends only on the coupling matrix T and not explicitly on Z. Therefore, for fixed T,

$$\nabla_Z S(T) = 0 \tag{48}$$

The Z-update is governed only by the geometric term $\alpha GW(Z;T)$. This establishes that supervision does not directly act on the representation updates.

For Part (ii), let $T^*(Z)$ denote the optimal coupling as a function of Z. The joint minimization of $J(Z,T)$ over both variables is equivalent to minimizing the value function

$$F(Z) = \min_T J(Z,T) \tag{49}$$

over Z alone. To compute the gradient of $F(Z)$ through the minimizer, we can invoke the Danskin theorem to show,

$$\nabla F(Z) = \nabla_Z J(Z, T^*(Z)) \tag{50}$$

wherever the minimiser T^* is unique and F is differentiable. This implies, the gradient of the joint objective is equal to the partial gradient with T frozen at its optimal value. From Part (i) we already know the supervised term is purely a function of T. Combining this with the fact that $\nabla F(Z)$ does not have a $\nabla_T J$ component because T^* is already optimal, we have $\nabla_Z S(T) = 0$. Consequently, we can conclude that the supervision term therefore does not contribute to $\nabla F(Z)$ wherever F is differentiable.

For quadratic OT problems like GW, the solution T^* can sit at extreme points of the coupling polytope ([Vayer, 2026]) and hence, $T^*(Z)$ is piecewise constant in Z. This implies F is piecewise-smooth and the supervision gradient is inert on each piece. Therefore, the labels influence Z solely through discrete switches (or flips) of the optimal coupling. This is a non-differentiable gradient channel that cannot be detected by any first-order method like BCD or Gradient Descent. Hence, we conclude that gradient saturation exists almost everywhere except for the points where the optimal coupling on the coupling polytope flips.

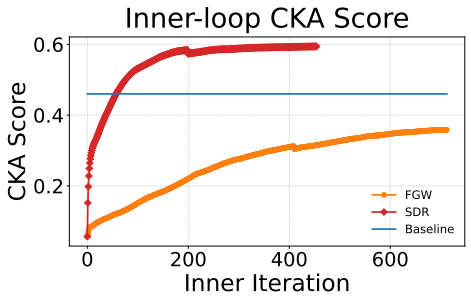
C.2 Empirical Evidence of the Gradient Inertness and the Effect of Explicit Dependence Maximization in SDR

In this study, we provide empirical evidence of the gradient inertness and why an explicit dependence maximization term is required for Supervised Distributional Reduction to work. The FGW-only version is a variant which uses the BSFGW objective for optimizing both **T** and **Z** with the BCD algorithm. We compare this approach with the SDR objective that includes an explicit dependence maximization term in the form of CKA. We also include DistR and a variant of DistR where we combine the DistR objective with the CKA term to assess what the FGW formulation provides. We report the Homogeneity scores and the Silhouette scores for the DistR, DistR + CKA, FGW-only and SDR variants in Table 3 for the COIL-20 and SNAREseq datasets.

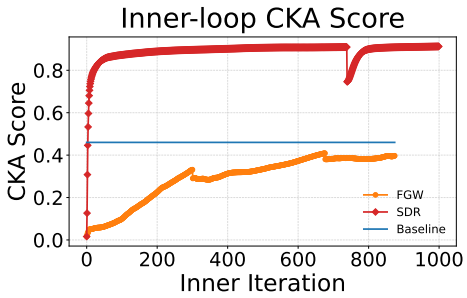
	COIL-20 ($m = 100$)		SNAREseq ($m = 100$)	
	Homogeneity (↑)	Silhouette Score (↑)	Homogeneity (↑)	Silhouette Score (↑)
DistR	90.33 ± 0.50	22.96 ± 9.35	81.62 ± 0.63	30.15 ± 3.24
DistR + CKA	90.35 ± 0.51	51.63 ± 1.54	81.79 ± 0.36	74.22 ± 9.67
FGW-only	99.71 ± 0.32	32.04 ± 2.26	96.01 ± 1.52	13.43 ± 1.63
SDR	99.27 ± 0.61	55.65 ± 2.15	92.46 ± 0.70	80.05 ± 8.17

Table 3: Homogeneity and Silhouette Score comparison for different objectives across 5 seeds

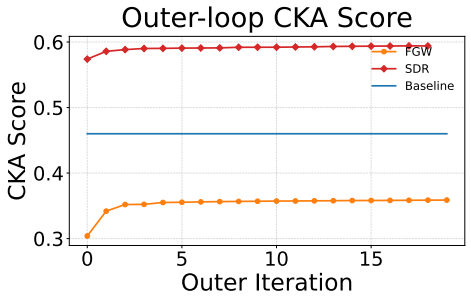
We can observe from Table 3 that the DistR + CKA variant improves on the Silhouette score but not on the Homogeneity score as the optimization of T is unsupervised and this does not improve cluster



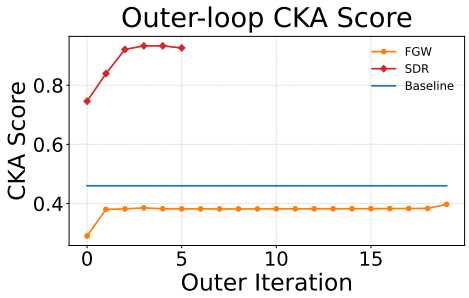
(a) Typical first inner loop for COIL-20



(b) Typical first inner loop for SNARESeq



(c) Typical Outer Loops for COIL-20



(d) Typical Outer Loops for SNARESeq

Figure 5: Variation of the CKA score across inner and outer loops of the optimization process for COIL20 and SNARESeq datasets.

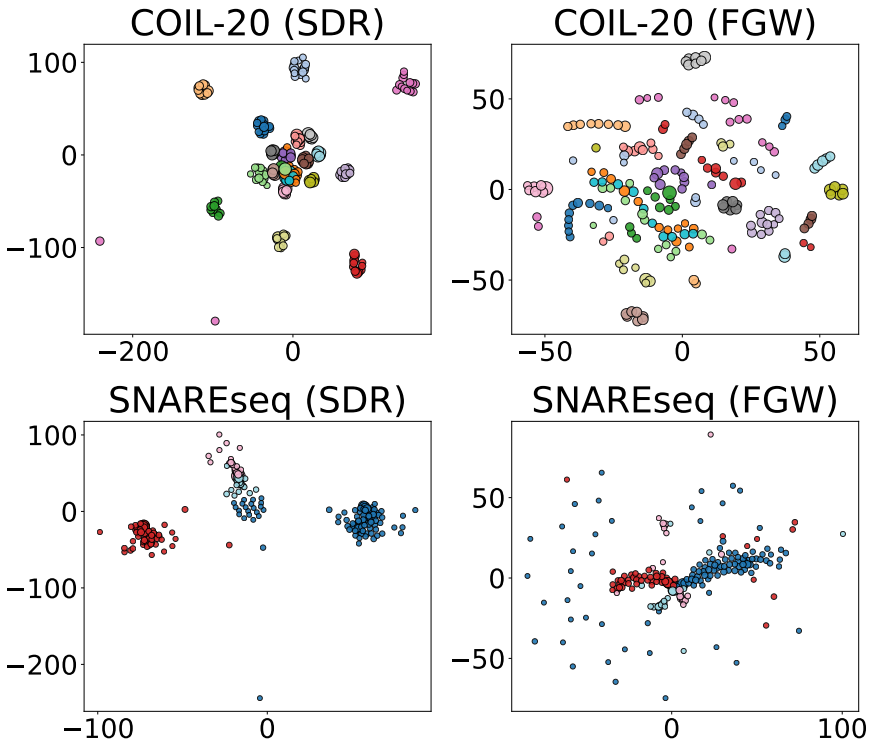


Figure 6: Example embeddings for the COIL20 and SNAREseq datasets for the SDR and FGW-only objectives for visual comparison.

purity. On the other hand, both FGW-only and the SDR formulations that use FGW for the T-step improve on the Homogeneity scores. On the other axis of the analysis, the FGW-only formulation does not improve over DistR when it comes to the silhouette score because of the gradient inertness discussed in Remark 3.3. Since, SDR incorporates target information through both channels - the FGW supervision term and through the CKA term, it improves on both metrics compared to its DistR ad FGW counterparts.

Plots (a) and (b) in Figure 5 show how the CKA score of the embeddings Z with respect to the targets evolve in the first inner loop of the SDR objective. This inner loop corresponds to the optimization of Z using an Adam optimizer and runs at most for 1000 steps. Plots (c) and (d) show how the CKA score progresses with the outer loop which consists of two steps: an inner loop for optimizing Z and a solve of the srBSFGW objective to update the coupling matrix T. Clearly, the FGW objective plateaus and produces a worse CKA score than the actual data, whereas the SDR objective produces a high CKA score indicating the retention of predictive signal leading to target-aware embeddings.

Figure 6 shows example embeddings generated by SDR and the FGW-only variant for the COIL-20 and SNAREseq datasets. SDR produces reasonably well separated clusters of the different classes in the dataset whereas the FGW-only variant while managing to bring prototypes from the same class together, does not form well separated clusters. These results indicate that the FGW extension is useful but the dependence maximization term in the SDR objective makes it much more effective in capturing the nuances of the dataset.

D Distributional Reduction Experimental Details

D.1 Datasets

The experiments for distributional reduction (joint clustering and DR) used three datasets with varying properties. We use the COIL-20 [Nene et al., 1996], Fashion MNIST [Xiao et al., 2017] and SNAREseq [Chen et al., 2019] datasets for all distributional reduction experiments. The details of these datasets are provided in Table 4.

Dataset	Number of Points	Input Features	Number of Classes
COIL-20	1440	16384	20
Fashion-MNIST	10000	784	10
SNAREseq (Chromatin)	1047	19	4

Table 4: Details of Datasets used in Distributional Reduction experiments

COIL-20 and the Fashion-MNIST datasets were pre-processed with PCA to reduce the number of features to 50 as done in practice (Assel et al. [2025], van der Maaten and Hinton [2008]) and the inputs were normalized for all three datasets.

D.2 Experiments

The distributional reduction experiments were performed on 3 different datasets as mentioned in the previous sub-section. The methods tested were DistR, Cluster-then-DR and DR-then-Cluster following the same experimental setup in Assel et al. [2025]. The experiments were run for different number of prototypes {20, 60, 100, 140, 180} with 5 random seeds. For evaluation metrics, we consider the Homogeneity score, k-means Normalized Mutual Information and the Silhouette Score as defined in Assel et al. [2025]. All experiments were run on a remote server equipped with an NVIDIA-L4 GPU.

D.3 Hyperparameter settings

All steps involving clustering both for the initialization of the coupling matrices and for methods that explicitly perform clustering, the Spectral Clustering implementation from scikit-learn was used. For all experiments, DistR and SDR used the Symmetric Entropic Affinity [Assel et al., 2023] for $\mathbf{X}$ with the perplexity hyperparameter tuned between {20, 30, 50, 100, 200} and the learning rate tuned between {0.01, 0.1, 1.0}. The learning rates were chosen as the highest value in the set that did not

break the optimization routine of the SEA transport problem. We report these tuned values for the various datasets in Table 5.

Dataset	Perplexity	Learning Rate
COIL-20	20	0.1
Fashion-MNIST	50	0.01
SNAREseq (Chromatin)	30	0.01

Table 5: Hyperparameter details of SEA for various datasets

For the embeddings $\mathbf{Z}$, the normalized Student-t affinity [van der Maaten and Hinton, 2008] was used. For the CKA computation, we require positive semi-definite kernels and so affinities cannot be directly used. Therefore, we choose the kernel that is closest in behaviour to the normalized Student-t affinity which is the Rational Quadratic (RQ) kernel [Rasmussen and Williams, 2005] given by

$$k_{RQ}(x, x') = \left(1 + \frac{||x - x'||^2}{2\alpha l^2}\right)^{-\alpha} \tag{51}$$

This kernel is equivalent to adding several SE kernels with different lengthscale parameters. The parameter α determines how large scale variations are weighted relative to small scale variations. The RQ kernel behaves like a heavy-tailed similarity function since it decays polynomially. As all the datasets used in these experiments are classification datasets, we use the delta kernel for the outputs as described in Barshan et al. [2011]. For all experiments, α in the srBSFGW objective was set to 0.2 and η in the SDR objective was set to 1000. The Adam optimizer was used for updating the embeddings Z with the default learning rate of 0.001.

E Accept-Reject step for SDR

In this section, we show some results on the safeguarded variant of SDR. This modification of SDR introduces an accept-reject step: a T-update is accepted only if it does not increase the full objective J, otherwise the previous coupling matrix is retained. It should be noted that, accepted steps are non-increasing and guarantee monotonic descent on the T-block. We compare this variant with the original algorithm in Table 6 where the experiments were performed on the COIL-20 and SNAREseq datasets for 3 random seeds and the results averaged across $m \in 20, 60, 100, 140, 180$.

Method	COIL-20			SNAREseq		
	Homogeneity (↑)	k-Means NMI (↑)	Silhouette Score (↑)	Homogeneity (↑)	k-Means NMI (↑)	Silhouette Score (↑)
SDR	94.95 ± 7.93	91.69 ± 3.01	48.00 ± 17.01	93.35 ± 1.53	90.69 ± 2.53	82.66 ± 7.28
SDR gated on J	94.95 ± 7.93	91.69 ± 3.01	48.00 ± 17.01	90.62 ± 2.22	87.16 ± 4.77	80.36 ± 8.41

Table 6: Comparison of vanilla SDR and SDR with an accept-reject gate on J.

The safeguarded variant is indistinguishable from our original scheme on COIL-20 — the FGW T-update acts as a descent step for J in practice. On SNAREseq it performs slightly worse, with the gap growing with m as shown in Table 7.

m	SDR			SDR gated on J		
	Homogeneity (↑)	k-Means NMI (↑)	Silhouette Score (↑)	Homogeneity (↑)	k-Means NMI (↑)	Silhouette Score (↑)
20	93.84 ± 3.90	94.05 ± 3.70	93.19 ± 3.85	93.84 ± 3.90	94.05 ± 3.70	93.19 ± 3.85
60	91.27 ± 0.44	91.44 ± 0.32	85.52 ± 1.51	90.48 ± 0.79	90.07 ± 1.06	83.95 ± 1.35
100	92.90 ± 0.05	89.09 ± 5.12	79.79 ± 11.54	91.43 ± 0.63	84.54 ± 5.02	75.31 ± 13.70
140	93.23 ± 0.13	91.44 ± 1.36	81.19 ± 1.51	87.95 ± 7.40	84.76 ± 10.18	77.38 ± 4.95
180	95.50 ± 1.66	87.43 ± 8.51	73.58 ± 13.79	89.37 ± 8.05	82.40 ± 11.20	71.94 ± 12.64

Table 7: Comparison of vanilla SDR with the safeguarded variant gated on the objective J.

The accept-reject mechanism has only one action - to reject when the objective increases from a T-update. This can force the optimization routine to be stuck in a loop. The occasional non-monotone

T-steps whose magnitude is η-bounded, controls and helps the scheme escape poor local minima, which is very common in non-convex alternating minimization. An alternative to this would be to use line search/backtracking methods instead of having a simple accept-reject gate. We leave this improvements and convergence guarantees as future work.

F Comparison with Sequential Baselines

The core SDR method brings two data reduction paradigms together namely: Clustering and Dimensionality Reduction (DR). SDR produces prototypes by jointly performing clustering and DR. In these ablations, we test how the joint reduction procedure compares with sequential baselines, where clustering and DR are performed in a sequence, one after another. We tested two protocols:

- **Protocol A** - Clustering and DR both steps are supervised. Both stages run SDR, clustering step runs with $m < n$ and the DR step runs with $m = n$.
- **Protocol B** - Clustering step is unsupervised and uses spectral clustering, but the DR step is supervised and runs SDR with $m = n$.

Protocol A is, in essence closer to the SDR algorithm as both stages are supervised, whereas Protocol B is closer to the DistR evaluation framework. Since both variants use SDR as part of the clustering and DR steps, we use this ablation to understand which mode of usage of SDR works better for the datasets we used in the experiments. The results for $m = 20, 100, 200$ over 5 random seeds are provided in Tables 8, 9 and 10 respectively for the COIL-20 and SNAREseq datasets.

Method	COIL-20 (m=20)				SNAREseq (m = 20)			
	Homogeneity (↑)	k-Means NMI (↑)	Silhouette Score (↑)	Runtime (s) (↓)	Homogeneity (↑)	k-Means NMI (↑)	Silhouette Score (↑)	Runtime (s) (↓)
Spec-C→DR	60.33 ± 0.09	68.55 ± 0.06	21.70 ± 2.97	3.03 ± 0.15	76.45 ± 0.00	66.46 ± 10.09	53.94 ± 7.84	1.77 ± 0.15
DR→Spec-C	89.13 ± 4.89	91.57 ± 3.24	6.03 ± 2.93	209.24 ± 87.32	81.38 ± 2.02	77.91 ± 4.74	65.69 ± 2.39	113.07 ± 26.64
C→DR	81.07 ± 0.00	86.52 ± 0.00	24.26 ± 0.44	3.39 ± 0.15	100.00 ± 0.00	95.99 ± 8.96	87.95 ± 12.73	1.98 ± 0.06
DR→C	88.85 ± 6.18	91.81 ± 3.96	5.65 ± 5.10	214.51 ± 86.21	73.79 ± 19.27	72.54 ± 20.51	60.18 ± 23.76	115.88 ± 26.20
Joint SDR	81.07 ± 0.00	86.52 ± 0.00	19.42 ± 3.33	3.47 ± 0.11	95.16 ± 3.40	95.31 ± 3.23	92.24 ± 4.50	2.49 ± 0.83

Table 8: Comparison with sequential baselines for $m = 20$.

Method	COIL-20 (m=100)				SNAREseq (m = 100)			
	Homogeneity (↑)	k-Means NMI (↑)	Silhouette Score (↑)	Runtime (s) (↓)	Homogeneity (↑)	k-Means NMI (↑)	Silhouette Score (↑)	Runtime (s) (↓)
Spec-C→DR	90.16 ± 0.63	78.52 ± 2.62	37.68 ± 6.99	3.31 ± 0.16	79.32 ± 1.95	68.93 ± 4.74	56.63 ± 7.15	2.25 ± 0.38
DR→Spec-C	87.72 ± 4.30	84.93 ± 3.72	51.44 ± 7.19	206.27 ± 85.57	70.50 ± 4.25	67.22 ± 5.56	52.00 ± 10.88	112.84 ± 26.23
C→DR	98.78 ± 0.31	93.02 ± 2.40	68.79 ± 3.56	4.39 ± 0.83	93.06 ± 0.84	93.06 ± 0.76	89.45 ± 1.55	3.69 ± 0.66
DR→C	92.26 ± 7.68	90.20 ± 5.73	32.62 ± 37.72	213.44 ± 86.62	79.88 ± 5.79	30.50 ± 42.39	34.37 ± 35.39	118.12 ± 22.71
Joint SDR	99.27 ± 0.61	94.10 ± 1.54	55.65 ± 2.15	4.36 ± 0.28	92.46 ± 0.70	89.42 ± 3.69	80.05 ± 8.17	3.44 ± 0.39

Table 9: Comparison with sequential baselines for $m = 100$.

Method	COIL-20 (m=200)				SNAREseq (m = 200)			
	Homogeneity (↑)	k-Means NMI (↑)	Silhouette Score (↑)	Runtime (s) (↓)	Homogeneity (↑)	k-Means NMI (↑)	Silhouette Score (↑)	Runtime (s) (↓)
Spec-C→DR	95.70 ± 0.42	83.88 ± 2.42	50.97 ± 4.65	5.04 ± 0.81	76.59 ± 1.87	71.80 ± 3.07	67.84 ± 4.90	3.64 ± 0.85
DR→Spec-C	84.40 ± 3.02	81.87 ± 5.01	45.47 ± 4.77	211.10 ± 85.72	65.99 ± 3.01	64.80 ± 2.71	40.88 ± 5.40	113.38 ± 26.64
C→DR	96.82 ± 0.46	93.94 ± 1.59	81.24 ± 3.46	7.33 ± 1.54	94.60 ± 0.80	94.67 ± 0.78	90.36 ± 2.33	8.69 ± 1.04
DR→C	62.07 ± 50.62	57.17 ± 52.24	37.50 ± 39.59	217.32 ± 85.55	63.72 ± 20.64	32.89 ± 45.16	24.67 ± 46.56	122.66 ± 24.35
Joint SDR	98.86 ± 0.74	91.30 ± 1.78	58.93 ± 2.70	6.61 ± 0.49	94.71 ± 1.16	79.35 ± 3.45	64.13 ± 9.72	7.40 ± 0.70

Table 10: Comparison with sequential baselines for $m = 200$.

Under Protocol A, joint SDR is competitive and is better than most baselines except C→DR. We hypothesize this is because purity metrics are saturated for any pipeline that clusters first, since supervised clustering in input space can be near-perfect/trivial. DR→C incurs 30–60× the runtime with prototype collapse and severe seed instability (e.g., silhouette score 34.37 ± 35.39 on SNAREseq for $m = 100$).

Under Protocol B, the setting matching the original DistR paper, joint SDR wins Homogeneity and k-Means NMI in every configuration and silhouette in the large majority. The sequential failure modes are complementary: Spec-C→DR fixes the partition from geometry alone before supervision

enters, so target-mixing prototypes can never be repaired downstream; DR→Spec-C computes the partition without labels on a 2-D projection at 30–60× runtime. Joint optimization is thus the only mechanism transporting label information into both subproblems efficiently.

G Hyperparameter Sensitivity - Ablation studies for α, η and β

G.1 Ablation study (β): Effect of Projection Regularization

The hyperparameter β used in the SDR-OOS objective for the intermediate projection of the embeddings Z plays a key role in defining the predictive performance of the downstream model. β handles the trade-off of the embeddings being representable by a projection map against the retention of predictive signal in the optimized embeddings. In this experiment, we ran a sweep over the range of β values where, $\beta \in [0, 1]$. $\beta = 0$ corresponds to no regularization of Z with respect to L and $\beta = 1$ tries to replace the optimized Z in an iteration with its projection onto the RKHS representable set of K. We sample β at uniform intervals in this range which corresponds to $\{0.0, 0.25, 0.5, 0.75, 1.0\}$ and run the SDR-GP method on all three regression datasets. From the plots it is evident that a high β value improves the test error by pulling the optimized Z back onto the RKHS representable set of K. However, this can potentially misalign Z with targets Y to some degree and this manifests as increasing training error. On the other hand, a low β value improves the training error but increases the test error since the optimization of Z is now less regularized. To maintain a balance between these competing factors, a value close to the mean/median of this range which is 0.5 seems to be a sensible choice for this parameter and consequently we use this value across all other kernel learning experiments. We ran this experiment across 3 different random seeds and the mean is reported in the various plots of Figure 7.

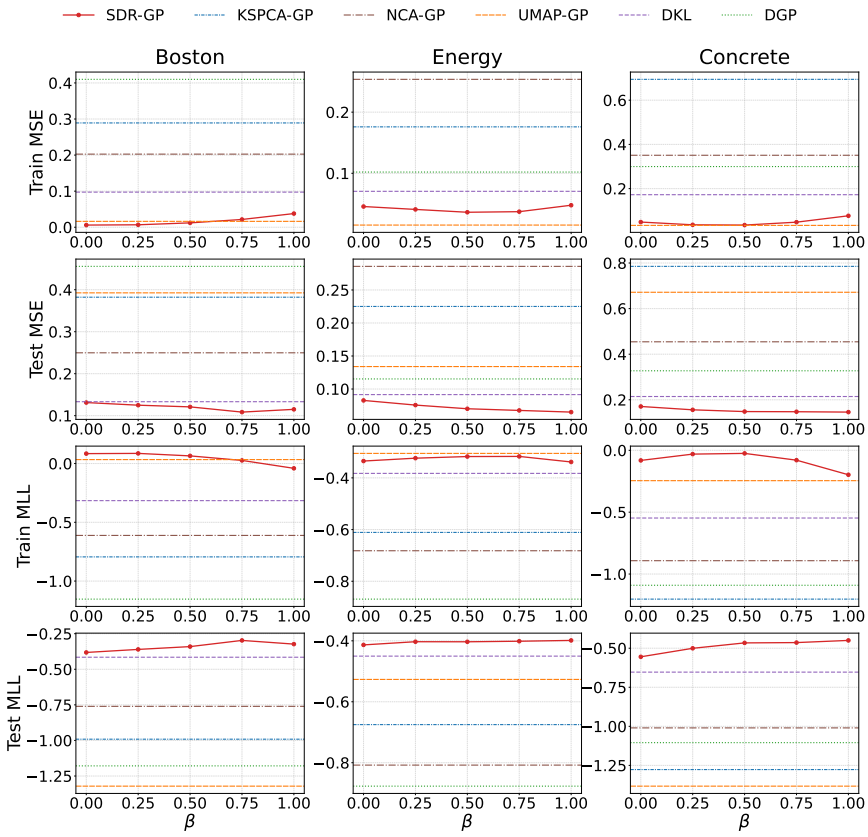


Figure 7: Ablation on the β parameter for the regression datasets. A very high β value (close to 1) increases training error but improves the test error. A low β value (close to 0) decreases the training error but increases the test error. The dashed lines corresponds to the baseline performance of other methods.

Additionally, the behavior observed across the β sweep highlights the role of the SDR regularization term as an implicit complexity control mechanism on the learned embeddings. The relatively stable performance observed around intermediate β values further suggests that SDR-GP is not overly sensitive to precise tuning of this parameter, indicating robustness of the proposed objective across a reasonably broad operating regime. This behavior is consistent across all evaluated datasets and supports the interpretation of β as governing a bias-variance trade-off between representability and predictive alignment of the optimized embeddings.

G.2 Ablation study (η): Effect of Dependence Strength

η is a very important hyperparameter that determines the influence of the targets in the optimization process of the embeddings Z according to the SDR objective. In this section, we present some results on the ablation of the η parameter. Since the CKA term is a normalized version of the HSIC loss it always lies in $[0, 1]$. Therefore, depending on the magnitude of the GW loss η can be suitably tuned. We sweep across η in the log space and for three different number of prototypes {20, 100, 180}. The results for the COIL-20 and SNAREseq datasets are presented in Figures 8 and 9 respectively. For most of the cases in Figure 8 and Figure 9, it is evident that a higher η value improves the quality of the prototypes produced by SDR. However, we observe diminishing gains in the quality of the embeddings in many of these plots after a value of 10^3 for η. Also, for the DKL experiments there was a trade-off that we observed between how well the OOS map L can approximate a very complex map produced by SDR through OT and maximizing predictive signal in the prototypes. Therefore, a reasonably high value of η produces high-quality prototypes and η was set to 10^3 for all our experiments.

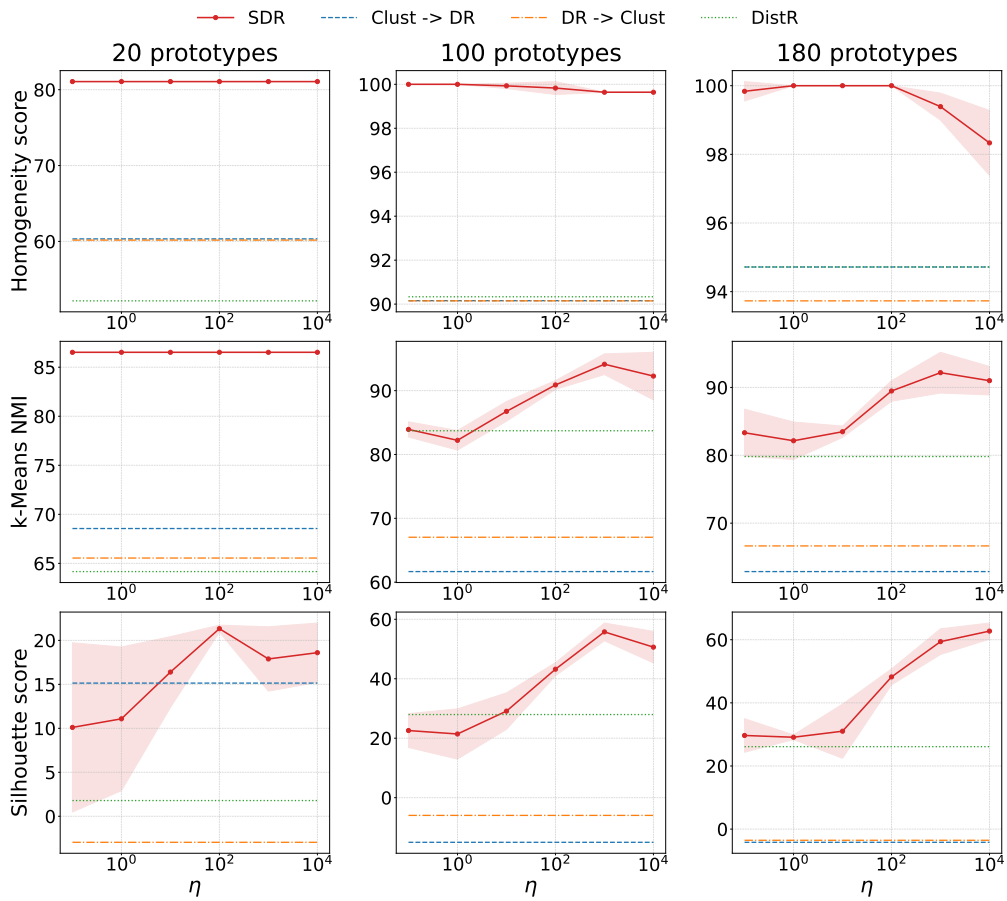


Figure 8: Scores (x100) across varying values of η for the COIL-20 dataset averaged over three random seeds. The flat lines show the average performance of the other methods as a baseline for comparison for a given number of prototypes.

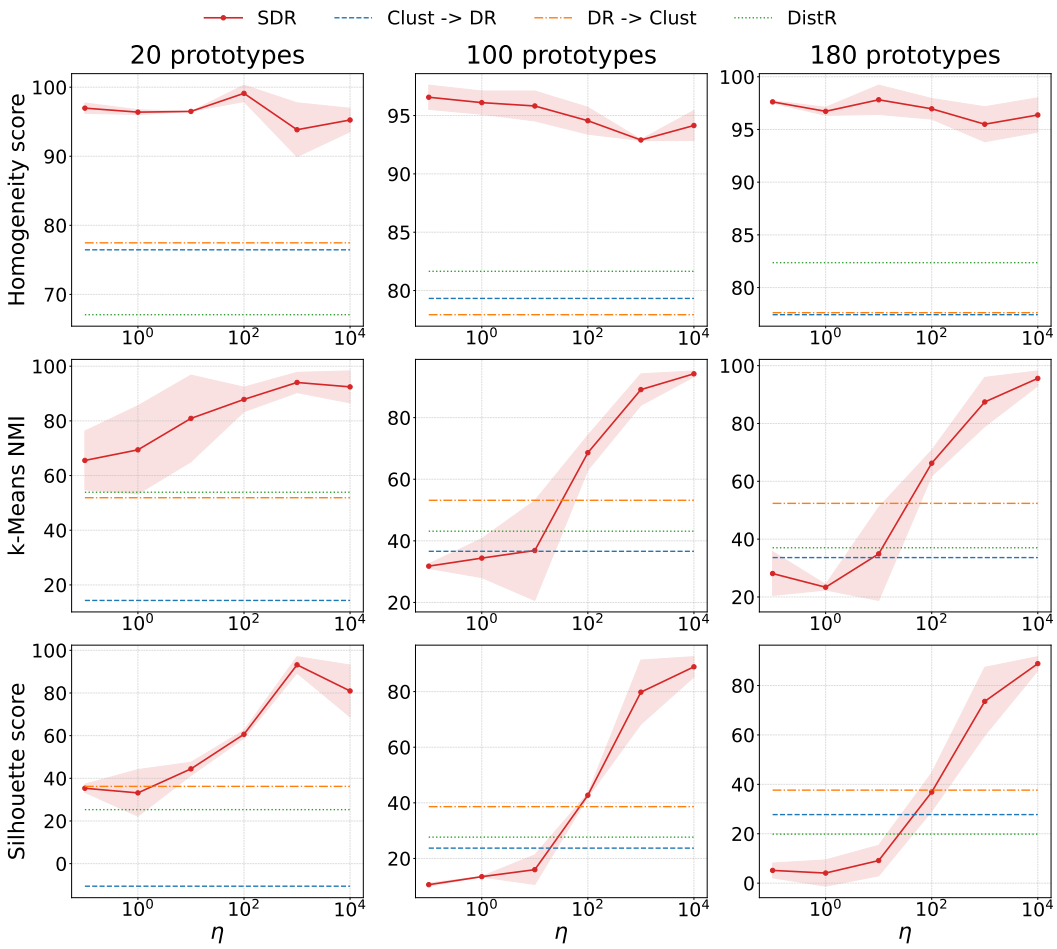


Figure 9: Scores (x100) across varying values of η for the SNAREseq dataset averaged over three random seeds. The flat lines show the average performance of the other methods as a baseline for comparison for a given number of prototypes.

G.3 Ablation study (α): Effect of Supervision on T

As stated in Vayer et al. [2018], the interpretation of α is to interpolate between prioritising feature similarity (output similarity for SDR) and relational structure. In this work, we consider metrics that look at purity of the formed clusters and retention of predictive fidelity since that is the focus of this work. Under these metrics, the ablation over α shows an expected trend - worst predictive alignment for $\alpha \approx 1$ and best predictive alignment for $\alpha \approx 0$. We ran SDR with $\alpha \in \{0.10, 0.25, 0.50, 0.75, 0.90\}$ for the COIL-20 and SNAREseq datasets with $m = 100$. We report the results for the two datasets in Tables 11 and 12.

α	Homogeneity (↑)	k-Means NMI (↑)	Silhouette Score (↑)	CKA (↑)	Runtime (s) (↓)
0.10	100.00 ± 0.00	93.17 ± 0.32	55.73 ± 3.68	0.617 ± 0.008	4.98
0.25	96.61 ± 0.06	89.81 ± 4.84	49.14 ± 6.57	0.601 ± 0.013	4.05
0.50	91.12 ± 0.83	90.98 ± 1.47	50.24 ± 6.04	0.610 ± 0.008	5.11
0.75	90.46 ± 0.57	88.20 ± 1.74	43.66 ± 3.22	0.584 ± 0.007	5.10
0.90	90.34 ± 0.49	87.55 ± 2.17	46.60 ± 4.01	0.589 ± 0.026	5.58

Table 11: Ablation of the SDR interpolation parameter α on COIL-20. Mean $\pm$ std over seeds 0–2 (perplexity 20, learning rate 0.1).

α	Homogeneity (↑)	k-Means NMI (↑)	Silhouette Score (↑)	CKA (↑)	Runtime (s) (↓)
0.10	98.92 ± 0.16	88.22 ± 10.19	81.96 ± 13.15	0.916 ± 0.019	3.84
0.25	89.68 ± 0.34	88.87 ± 0.54	81.57 ± 6.14	0.925 ± 0.003	3.44
0.50	86.44 ± 0.73	78.03 ± 4.23	64.49 ± 4.95	0.889 ± 0.022	3.34
0.75	84.51 ± 0.78	72.40 ± 3.50	60.65 ± 1.78	0.909 ± 0.008	3.19
0.90	82.93 ± 1.04	72.90 ± 4.05	60.06 ± 2.18	0.892 ± 0.013	3.22

Table 12: Ablation of the SDR interpolation parameter α on SNAREseq. Mean $\pm$ std over seeds 0–2 (perplexity 30, learning rate 0.01).

Depending on the goal of the reduction task, α can be varied between 0 and 1 to preserve different properties in the embeddings. Since, the goal of this work is retain predictive fidelity, small values of α are preferred to retain maximum predictive fidelity. It is worth noting that GW is still performing its role of structure preservation, but we do not measure metrics related to geometric preservation here and thus the trend is seemingly monotonous. The discrepancies between $\alpha = 0.25$ and $\alpha = 0.5$ for the COIL-20 dataset highlight that the trend is non-monotonous, showing that the GW term is battling with target alignment in that range.

H Additional DR Visualizations and Downstream Evaluation

SDR just like DistR inherits the same runtime complexity $\mathcal{O}(n^2m + nm^2)$ where n is the number of source samples and m is the number of target samples. For the pure DR case ($p < d, n = m$), SDR scales as $\mathcal{O}(n^3)$. This is one of the core reasons why SDR is feasible as a distributional reduction method but not very scalable as a DR method. However, SDR might still be a very useful DR method in the low-data regime. While this is evident from the experiments in Section 5.2, we show visualizations of two toy datasets namely: (i) S-curve and (ii) Swiss Roll datasets to show how SDR behaves as a DR method when compared with other supervised DR methods in Figure 10. We chose these datasets, since they are the most basic datasets tested for every DR method and is easier to visualize the embeddings along with the original data since the original data lives in 3D. We compare the embeddings generated from Supervised-UMAP (S-UMAP) and Kernel Supervised PCA (KS-PCA) visually. For the S-curve dataset, we set the number of neighbours to 50 and for the swiss roll dataset we set this parameter to 20 to generate the supervised UMAP visualizations.

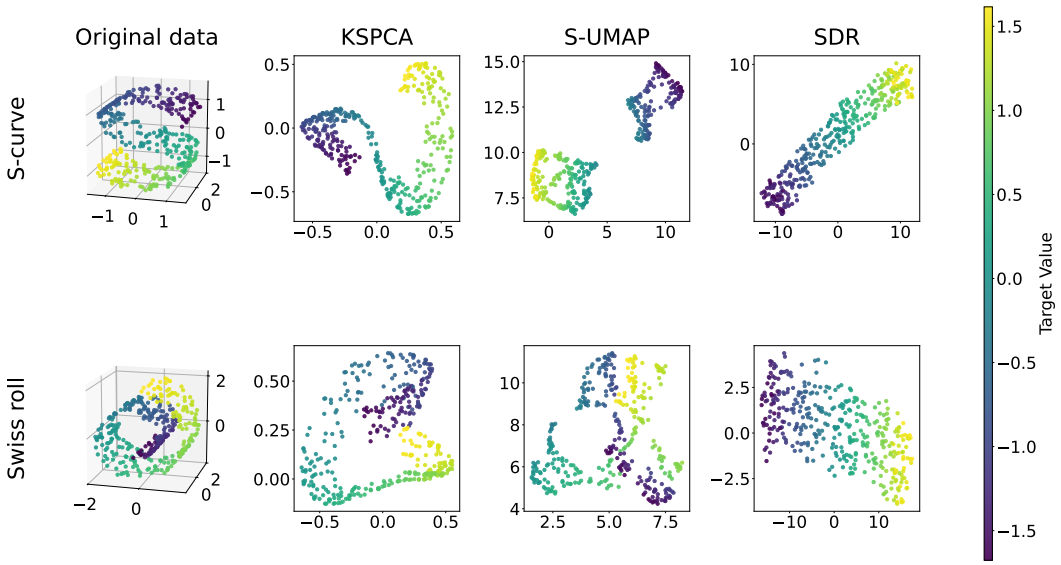


Figure 10: Visualizing embeddings in the pure DR setting for KSPCA, (S-UMAP) and SDR on toy 3D datasets.

For SDR, we choose the exact same settings used in the kernel learning experiments. The expected behaviour for any DR method on these two datasets is to unroll the curves onto flat rectangular strips in the 2D space. We can see that SDR achieves this quite cleanly compared to S-UMAP or KS-PCA. Along with this we also use the Friedman dataset [Friedman, 1991] for testing models on a downstream predictive task. We test the predictive performance of a k-Nearest Neighbours (k-NN) model and a Kernel Ridge Regression (KRR) model to show that the SDR embeddings work well in a model-agnostic manner. We report the R^2 scores for this experiment in Tables 13 and 14 and the metrics reported are across 5 different random seeds.

	S-curve (100 points)		Swiss Roll (100 points)		Friedman (500 points)	
	Train	Test	Train	Test	Train	Test
Supervised UMAP	$\mathbf{0.99 \pm 0.00}$	0.88 ± 0.08	0.99 ± 0.00	0.47 ± 0.49	$\mathbf{0.99 \pm 0.00}$	0.51 ± 0.14
KS-PCA	0.92 ± 0.06	0.93 ± 0.05	0.89 ± 0.10	0.85 ± 0.10	0.81 ± 0.02	0.71 ± 0.10
SDR	0.97 ± 0.02	$\mathbf{0.94 \pm 0.06}$	0.99 ± 0.01	$\mathbf{0.97 \pm 0.03}$	0.98 ± 0.00	$\mathbf{0.91 \pm 0.03}$

Table 13: R^2 scores for k-NN Regressor with $k = 10$.

	S-curve (100 points)		Swiss Roll (100 points)		Friedman (500 points)	
	Train	Test	Train	Test	Train	Test
Supervised UMAP	0.97 ± 0.01	0.82 ± 0.13	0.98 ± 0.01	0.53 ± 0.34	0.98 ± 0.00	0.55 ± 0.12
KS-PCA	0.90 ± 0.06	0.92 ± 0.06	0.86 ± 0.12	0.82 ± 0.12	0.80 ± 0.03	0.72 ± 0.08
SDR	$\mathbf{0.98 \pm 0.01}$	$\mathbf{0.96 \pm 0.03}$	$\mathbf{0.99 \pm 0.01}$	$\mathbf{0.96 \pm 0.05}$	0.98 ± 0.00	$\mathbf{0.91 \pm 0.02}$

Table 14: R^2 scores for KRR Regressor with $\alpha = 1$ (lengthscale of RBF kernel).

In several cases, SDR is better than Supervised UMAP on the test R^2 scores and is comparable on the train R^2 scores. This also shows that the mechanism of optimizing for L not only provides a mechanism to project OOS points onto the embedding space but also regularizes the optimized embeddings Z resulting in better test performance for a downstream predictor.

I Kernel Learning Experimental Details

I.1 Datasets

The experiments comparing SDR-GP with all the other methods included 5 datasets where 3 of them are regression datasets sourced from the UCI repository and 2 of them are classification datasets. The details of these datasets are provided in Tables 15 and 16.

Dataset	**Number of Points**	**Input Features**	**Output Features**
Boston Housing	506	13	1
Energy Efficiency	768	8	2
Concrete Compressive Strength	1030	8	1

Table 15: Details of Regression Datasets used in Kernel Learning experiments

Dataset	**Number of Points**	**Input Features**	**Number of Classes**
MNIST	2000	784 (28×28)	10
COIL-20	1440	16384 (128×128)	20

Table 16: Details of Classification (Image) Datasets used in Kernel Learning experiments

I.2 Models

We compare 5 different models in this experiment. We compare SDR-GP against Deep Kernel Learning [Wilson et al., 2016] and Deep Gaussian Processes (Damianou and Lawrence [2013], Salimbeni and Deisenroth [2017]) which are the most natural ways to add non-stationarity to GPs for modelling non-smooth functions. Since SDR also acts as a pure DR method in this application, we compare against two other supervised DR methods namely Kernel Supervised PCA [Barshan et al., 2011] and Supervised UMAP [McInnes et al., 2020] that act as embedding models/feature extractors for the GP. We dub these models KSPCA-GP and UMAP-GP respectively. For KSPCA, we use RBF kernels for the inputs, RBF and Delta kernels on the targets for regression and classification datasets respectively. For supervised UMAP, we use the official UMAP[4] library. We set the number of neighbours to 50, the effective minimum distance between embedded points to 0.1 and the distance metric to 'L2'.

I.3 NN architectures and GP settings

For regression tasks, we use the same NN Feature Extractor which is a Multilayer-Perceptron network with ReLU activations used in Wilson et al. [2016]. The network is a fully connected DNN with [d-1000-500-50-2] architecture. The GP for the Boston and the Concrete datasets are exact GPs that use an RBF kernel with Automatic Relevance Determination (ARD) [Rasmussen and Williams, 2005] and the standard Gaussian likelihood. The GP for the Energy dataset is a multi-task GP [Bonilla et al., 2007] that models both the covariances between the inputs and the output tasks with an RBF kernel (equipped with ARD) and a Gaussian likelihood.

The Deep GP model was constructed with one hidden layer and an output layer. The latent dimensions of the hidden layer were set to match the latent dimensions of all the other models. This model is trained variationally using the doubly stochastic variational inference technique described by Salimbeni and Deisenroth [2017]. The number of inducing points is set to 128 and the number of likelihood samples is set to 16.

For the classification tasks, we again use the same CNN feature extractor described in [Wilson et al., 2016] (Appendix A.1 in the paper). However, we use this feature extractor for classification instead of orientation extraction as described in the DKL paper. We use an exact GP with a Dirichlet likelihood for multi-class classification inspired by the work of Milios et al. [2018]. We set α_ϵ (concentration parameter of the Dirichlet likelihood) to 0.01 and the number of samples from the GP posterior to 256 since the posterior predictions cannot be obtained in closed form and needs to be MC-sampled.

I.4 SDR settings

For SDR, we use the Symmetric Entropic Affinity(SEA) from Assel et al. [2023] on the inputs $\mathbf{X}$ for all experiments in this section with the perplexity and the learning rate tuned for each dataset. The perplexity value was set to 20 for all experiments. The learning rate parameter was set to 0.01 for the regression datasets and 0.1 for the image datasets. The normalized Student-t affinity was used to compute affinities for the embeddings $\mathbf{Z}$. For the HSIC computation, we use the Rational Quadratic (RQ) kernel for the embedding space and the RBF/Delta kernel [Barshan et al., 2011] for the regression/classification outputs. The kernel $\mathbf{K}$ in the SDR-OOS objective is computed with RBF for all the experiments. The α parameter in the srBSFGW objective is set to 0.2, η is set to 1000, β is set to 0.5 and λ_L was set to $1e-2$ for all experiments. The coupling matrix $\mathbf{T}$ is initialized using spectral clustering for regression tasks. $\mathbf{T}$ is initialized randomly for classification tasks since any clustering technique can run into the issue of producing degenerate clusters when the number of clusters required is much higher than the number of actual clusters present in the data which is typically the case with classification and the pure DR setting.

I.5 Experimental Details

For the regression datasets, the number of embedding dimensions was set to 2 and for the classification datasets the number of embedding dimensions was set to 10 following the practice in Ober et al. [2021]. All the GP models were trained for 20 epochs for the regression benchmarks and were

[4] https://umap-learn.readthedocs.io/en/latest/index.html

trained for 50 epochs for the classification benchmarks. The learning rate was set to 0.1 for all the models. The test data size was set to 0.2 for all experiments. We were unable to obtain stable DGP results within our compute budget for the classification setting due to memory pressure when scaling the number of latent dimensions to 10. This experiment was run on a remote server equipped with an NVIDIA A100-80GB GPU so that the Dirichlet GP fits in memory for the image classification datasets.

I.6 Model training times

We report the training times for the various models that were considered for the kernel learning benchmarks (Section 5.2) in Table 17. The training times reported for KSPCA-GP, UMAP-GP and SDR-GP includes the time taken to fit the DR transform and the time taken to train the GP since the fitting of the DR transform is decoupled from GP training in these models.

	Boston	Energy	Concrete	COIL-20	MNIST-2k
NCA-GP	0.45 ± 0.07	1.34 ± 0.30	3.00 ± 1.20	1.76 ± 0.13	2.62 ± 0.24
KSPCA-GP	0.15 ± 0.08	0.61 ± 0.01	0.49 ± 0.05	1.27 ± 0.04	1.09 ± 0.03
UMAP-GP	8.01 ± 16.42	1.70 ± 0.05	1.93 ± 0.15	16.59 ± 12.10	6.80 ± 0.10
DGP	0.44 ± 0.07	0.80 ± 0.01	0.41 ± 0.00	-	-
DKL	0.53 ± 0.89	0.56 ± 0.02	0.39 ± 0.03	34.63 ± 0.31	2.40 ± 0.06
SDR-GP (ours)	7.07 ± 2.21	5.14 ± 0.60	8.42 ± 2.44	8.00 ± 0.57	17.47 ± 2.55

Table 17: Runtime (in seconds) for all datasets based on five random seeds

J Uncertainty Calibration Diagnostics

When assessing probabilistic ML models, uncertainty calibration plays a major role in analyzing the predictive uncertainties produced by a model. For this, we use standard calibration curves and Mean Absolute Calibration Error as the metric. Figure 11 shows typical calibration curves of all the models in our experimental suite for the UCI regression datasets.

A calibration curve fundamentally validates the fact: **"An $\alpha\%$ prediction interval should contain $\alpha\%$" of the true values"**. For a scalar metric, we compute MACE following the uncertainty-toolbox convention of Chung et al. [2021], based on the regression calibration framework of Kuleshov et al. [2018]. For Gaussian predictive uncertainties, this corresponds to comparing nominal Gaussian interval/quantile probabilities with empirical coverage, as in Tran et al. [2020]. In this work, we stick to the symmetric interval calibration mode for generating the calibration curves. The number of levels across which we assess the calibration is set to $n_{bins} = 99$. For $n_{bins} = 99$, we consider uniformly spaced confidence intervals $\alpha_k \in (0, 1)$. The "Interval" mode constructs symmetric prediction bands as follows:

$$[\hat{y} - z_{\alpha_k}\hat{\sigma}, \hat{y} + z_{\alpha_k}\hat{\sigma}],\ z_{\alpha_k} = \Phi^{-1}\left(\frac{1+\alpha_k}{2}\right) \tag{52}$$

where $\hat{y}$ is the predicted mean, $\hat{\sigma}$ is the predicted standard deviation and Φ^{-1} is the inverse CDF of a Gaussian distribution.

Figure 11 reports observed against expected coverage for each embedding–GP pipeline. SDR-GP attains the highest mean log-likelihood on all three datasets, improving on the strongest competing pipeline, DKL, while its miscalibration area remains comparable to the other deep baselines and uniformly below that of Deep GP. Its coverage curves lie above the diagonal throughout, so the residual miscalibration is under-confidence: the predictive intervals are conservative rather than optimistic, which is the preferable failure mode in a downstream decision setting, and SDR-GP never exhibits the over-confident regions visible for KSPCA-GP and UMAP-GP on Boston and Concrete. We note that MACE assesses marginal coverage only and is insensitive to sharpness; KSPCA-GP illustrates the distinction, attaining the lowest MACE on every dataset while yielding the least informative predictive densities. Energy is the hardest case for all methods, with every pipeline markedly under-confident.

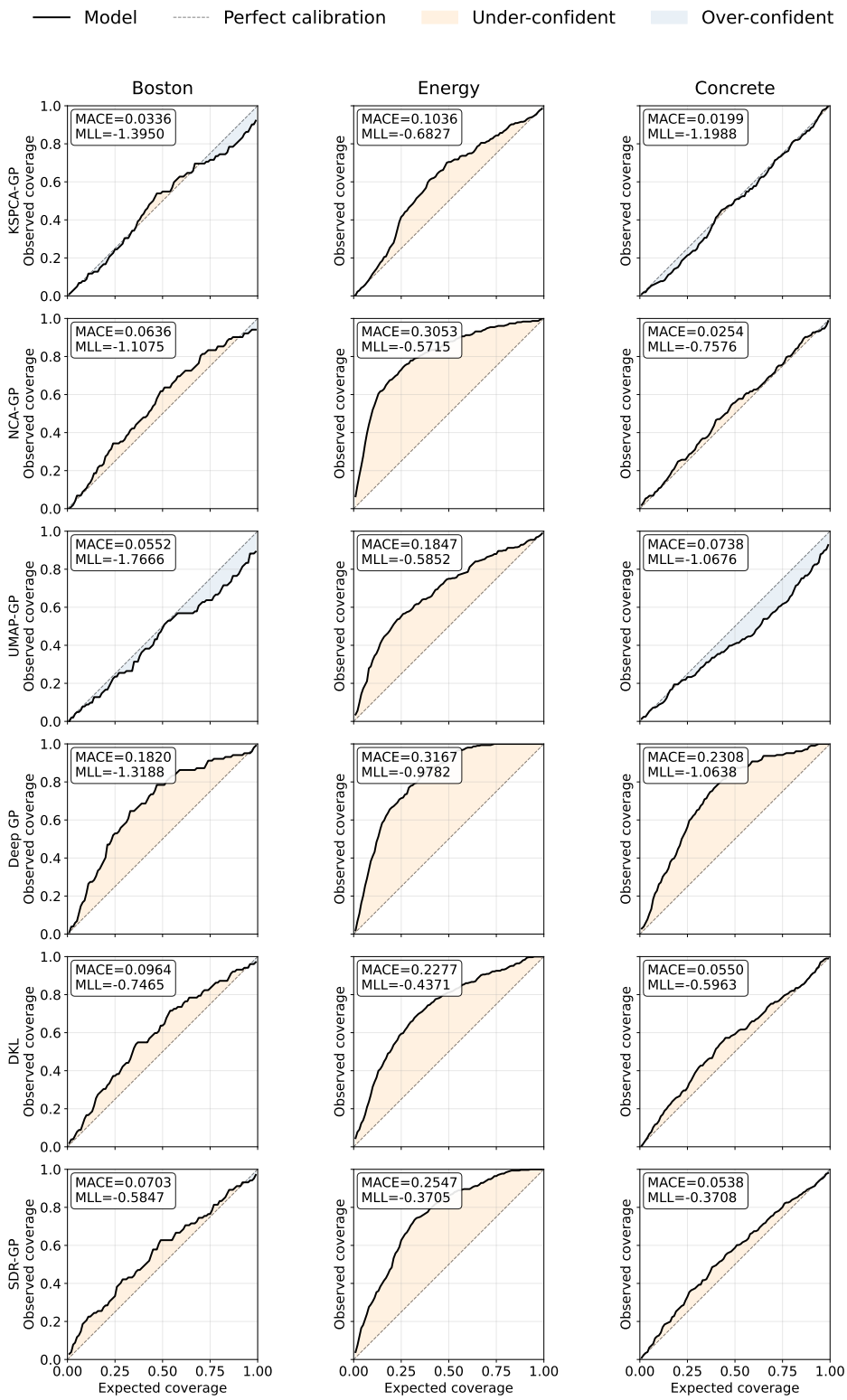


Figure 11: Calibration curves for the UCI regression datasets. The dotted diagonal line is the ideal calibration curve. Anything above the diagonal means the model is under-confident and anything below the diagonal means the model is over-confident.

K Comparison with pre-trained feature extractors for DKL

A widely used variant of DKL is where the feature extractor networks are pre-trained and then fine-tuned jointly along with the GP hyperparameters instead of training them from scratch. We use the same networks used for the experiments in Section 5.2 but pre-train them using squared loss and cross-entropy loss for the regression and classification benchmarks respectively. We train them during the pre-training phase for 75 epochs and with a learning rate of $1e-3$ followed by a training phase of 20 epochs for the regression datasets and 50 epochs for the classification datasets with a learning rate of 0.01. The results are show in Tables 18 and 19 where DKL-joint corresponds to the model where the NN feature extractor is trained from scratch jointly with the GP hyperparameters and DKL-pretrained corresponds to the model where the NN feature extractor is pre-trained and then fine-tuned along with the GP hyperparameters in the joint-training phase.

	Boston		Energy		Concrete	
	MLL (↑)	MSE (↓)	MLL (↑)	MSE (↓)	MLL (↑)	MSE (↓)
DKL-joint	-0.42 ± 0.21	0.13 ± 0.07	-0.45 ± 0.03	0.09 ± 0.02	-0.65 ± 0.11	0.21 ± 0.05
DKL-pretrained	-0.76 ± 0.05	0.11 ± 0.07	-0.71 ± 0.10	0.08 ± 0.02	-0.82 ± 0.01	0.17 ± 0.01
SDR-GP (ours)	-0.32 ± 0.18	0.14 ± 0.08	-0.37 ± 0.02	0.05 ± 0.01	-0.40 ± 0.07	0.13 ± 0.02

Table 18: Test results for DKL variants on UCI regression datasets based on five random seeds

	MNIST-2k		COIL-20	
	MLP (↑)	ACC (↑)	MLP (↑)	ACC (↑)
DKL-joint	-0.30 ± 0.05	0.90 ± 0.02	-0.09 ± 0.02	0.99 ± 0.00
DKL-pretrained	-0.25 ± 0.10	0.94 ± 0.03	-0.13 ± 0.01	0.97 ± 0.03
SDR-GP (ours)	-0.23 ± 0.02	0.95 ± 0.00	-0.07 ± 0.02	0.99 ± 0.00

Table 19: Test results for DKL variants on image classification datasets based on five random seeds.

These results emphasise that the performance of improvements of SDR are not solely attributable to the joint feature learning regime of DKL. Even in this controlled setting where both SDR and DKL operate on comparable pretrained features, SDR consistently achieves comparable or better predictive performances. These results suggest that the observed gains arise from the proposed SDR formulation itself rather than differences in representation learning or end-to-end optimization dynamics.

L Code/Implementation details

The implementation of SDR was built on top of the DistR[1] implementation from Assel et al. [2025]. The benchmarking scripts for kernel learning use GP models constructed with GPyTorch[2][Gardner et al., 2018]. For solving the srBSFGW objective, we use the FGW/semi-relaxed FGW solvers from the Python Optimal Transport (POT)[3] library. All experiments were implemented in Python using PyTorch for automatic differentiation and GPU acceleration. Experiments were run on NVIDIA GPUs; the distributional reduction benchmarks used an NVIDIA L4 GPU, while the kernel learning benchmarks used an NVIDIA A100-80GB GPU. All reported means and standard deviations (1-sigma error bars) are computed over multiple random seeds, with the number of seeds specified in the corresponding experimental details.

The code for the SDR implementation and benchmarking scripts is available at: https://github.com/SaiAakash/supervised-distributional-reduction.

[1] https://github.com/huguesva/Distributional-Reduction

[2] https://docs.gpytorch.ai/en/stable/

[3] https://pythonot.github.io

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction describe the SDR method, and an out-of-sample extension for SDR with application to GPs. Sections 3 and 4 talk about these key contributions in detail. Experiments in Section 5 show results on these main claims with the proof for Proposition 3.2 in Appendix A.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 5.2 and Appendix H talk about the scalability of SDR as a pure DR method. Section 6 also proposes as future work, potential avenues to reduce runtime complexity and provide convergence guarantees for this framework.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The paper contains the proof for Proposition 3.2 in Appendix A. All the underlying assumptions for this Proposition are clearly specified in the corresponding statement.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Appendix L provides basic details of the code implementation for the method and benchmarks. Appendix D and I provide detailed information of the datasets used and all the settings required to reproduce the results of the Distributional Reduction and Kernel Learning benchmarks shown in the paper.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: A link to the code repository has been provided in Appendix L. The repository contains all the datasets used for the experiments with clear instructions to run the benchmarking scripts.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: While Section 5.1 and Section 5.2 explain the experimental setting and the evaluation metrics briefly, Appendix D and I explain in detail the experimental setting, evaluation metrics and the complete settings for the SDR algorithm, GP models and the baseline methods to understand the results in the main paper.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

Justification: The paper reports mean and standard deviations (1-sigma error bars) for all the evaluation metrics in the various experiments across different random seeds. We also plot the uncertainty bands in the various graphs to indicate the same.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix L talks about the compute resources used to run the experiments (these are also briefly mentioned in Appendix D and I).

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research involves algorithmic development and evaluation on standard regression and classification benchmarks. It does not involve human subjects or obviously ethically sensitive applications, and we assume it conforms to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.

- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: This work focuses on a theoretical and methodological contribution to distributional reduction, without a specific application domain or deployment context. As such, we do not identify direct or immediate positive or negative societal impacts; any downstream effects would depend on how the method is applied in practice, which is beyond the scope of this paper.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not involve any foundational or pre-trained models that are more widely usable and hence, safeguards are not applicable for this paper.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: Appendix L cites the previous code base on top of which the implementation of the current work was built. Appendix D and I cite the corresponding work for all of the datasets used in this work. The license and terms of these datasets are attached in the supplementary code as well.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [N/A]

Justification: There are no new assets introduced in this paper and therefore this is not applicable for this work.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A].

Justification: The paper does not involve any experiments requiring human subjects and so this requirement is not applicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.

- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A].

    Justification: The research does not involve human subjects, therefore IRB approval is not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: Apart from using LLMs for editing and formatting parts of the paper, they were also used to create some visualizations and facilitate the running of experiments.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.