The Balanced-Pairwise-Affinities Feature Transform

Daniel Shalam [1] **Simon Korman** [1]

Abstract

The Balanced-Pairwise-Affinities (BPA) feature transform is designed to upgrade the features of a set of input items to facilitate downstream matching or grouping related tasks. The transformed set encodes a rich representation of high order relations between the input features. A particular min-cost-max-flow fractional matching problem, whose entropy regularized version can be approximated by an optimal transport (OT) optimization, leads to a transform which is efficient, differentiable, equivariant, parameterless and probabilistically interpretable. While the Sinkhorn OT solver has been adapted extensively in many contexts, we use it differently by minimizing the cost between a set of features to *itself* and using the transport plan's *rows* as the new representation. Empirically, the transform is highly effective and flexible in its use and consistently improves networks it is inserted into, in a variety of tasks and training schemes. We demonstrate state-of-the-art results in few-shot classification, unsupervised image clustering and person re-identification. Code is available at `github.com/DanielShalam/BPA`.

1. Introduction

In this work, we reassess the functionality of features in *set-input* problems, in which a task is defined over a *set* of items. Prominent examples of this setting are few-shot classification (Ravi & Larochelle, 2017), clustering (Van Gansbeke et al., 2020), feature matching (Korman & Avidan, 2015) and person re-identification (Ye et al., 2021), to name but a few. In such tasks, features computed at test time are mainly compared relative to one another, and less so to the features seen at training time. For such tasks, the practice of learning a generic feature extractor during training and applying it at test time is sub-optimal.

[1]Department of Computer Science, University of Haifa, Israel. Correspondence to: Daniel Shalam <dani360@gmail.com>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

In set-input problems, such as few-shot classification, an *instance* of the task is in the form of a set of n items (e.g. images) $\{x_i\}_{i=1}^n$. A generic neural-network pipeline (Fig. 1 Left) typically uses a feature embedding (extractor) F, that is applied independently to each input item, to obtain a set of features $V=\{v_i\}_{i=1}^n=\{F(x_i)\}_{i=1}^n$, prior to downstream task-specific processing G (e.g. a clustering head or classifier). The features V can be of high quality (concise, unique, descriptive), but are limited in representation since they are extracted based on knowledge acquired for similar examples at train time, with no context of the test time instance they are part of, which is critical in set-input tasks.

We rather consider the more general framework (Fig. 1 Right), in which the per-item independently extracted feature collection V is passed to an attention-mechanism type computation, in which some *transform* jointly processes the entire set of instance features, re-embedding each feature in light of the joint statistics of the entire instance.

The main idea of BPA is very intuitive and is demonstrated on a toy example in Fig. 2. The embedding of each feature will encode the *distribution* of its affinities to the rest of the set items. Specifically, items in the embedded space will be close if and only if they share a similar such distribution, i.e. 'agree' on the way they 'see' the entire set. In fact, the transform largely discards the item-specific feature information, resulting in a purely relative normalized representation that results in a highly efficient embedding with many attractive properties.

The proposed transformation can be computed very efficiently, with negligible runtime within the hosting network, and can be easily used in different contexts, as can be seen in the pseudo-code snippets we provide in Sections A and C of the Appendix. The embedding itself is given by rows of an optimal-transport (OT) plan matrix, which is the solution to a regularized min-cost-max-flow fractional matching problem that is defined over the pairwise (self)-affinities matrix of the features in the set.

Technically, it involves the computation of pairwise distances and several normalization iterations of a Sinkhorn (Cuturi, 2013) algorithm, baring apparent similarities to many related methods based on either Spectral Clustering (Ng et al., 2001) that normalize the same affinity matrix), attention-mechanisms (Vaswani et al., 2017) that learn fea-

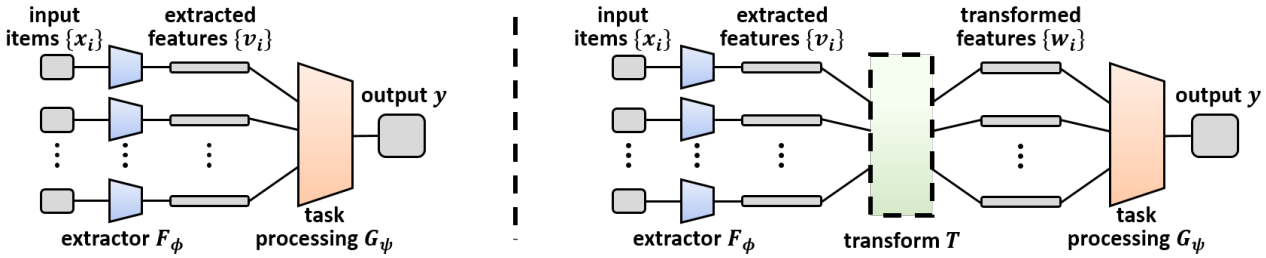

Figure 1. **Generic designs of networks that act on *sets* of items.** These cover relevant architectures, e.g. for few-shot-classification and clustering. **Left:** A generic network for processing a set of input items typically follows the depicted structure: (i) Each item *separately* goes through a common feature extractor F. (ii) The set of extracted features is the input to a downstream task processing module G. ; **Right:** A more general structure in which the extracted features undergo a *joint* processing by a transform T. Our BPA transform (as well as other attention mechanisms) is of this type and its high-level design (within the 'green' module) is detailed in Fig. 2.

tures based on a self-affinities matrix perhaps even normalized (Sander et al., 2022) and other matching (Sarlin et al., 2020) or classification (Hu et al., 2020) algorithms where optimal-transport plans are computed between source items and target items or class centers. However, the most important difference and our main novel observation is that the self fractional matching itself (which can be viewed as a balanced affinity matrix) can serve as a powerful embedding, since the distances in this space (between assignment vectors) have explicit interpretations that we explore, which are highly beneficial to general grouping based algorithms that are applied to such set-input tasks.

Contribution

We propose a parameter-less optimal-transport based feature transform, termed BPA, which can be used as a drop-in addition that converts a generic feature extraction scheme to one that is well suited to set-input tasks (e.g. from Figure 1 Left to Right). It is analyzed and shown to have the following attractive set of qualities. (i) *efficiency*: having real-time inference; (ii) *differentiability*: allowing end-to-end training of the entire 'embedding-transform-inference' pipeline of Fig. 1 Right; (iii) *equivariance*: ensuring that the embedding works coherently under any order of the input items; (iv) *probabilistic interpretation*: each embedded feature will encode its distribution of affinities to all other features, by conforming to a doubly-stochastic constraint; (iv) *valuable metrics for the item set*: Distances between embedded vectors will include both direct and indirect (third-party) similarity information between input features.

Empirically, we show BPA's flexibility and ease of application to a wide variety of tasks, by incorporating it in leading methods of each type. We test different configurations, such as whether the hosting network is pre-trained or re-trained with BPA inside, across different backbones, whether transductive or inductive. Few-shot-classification is our main application with extensive experimentation on standard benchmarks, testing on unsupervised-image-clustering shows the potential of BPA in the unsupervised domain and the person-re-identification experiments show how BPA deals with non-curated large-scale tasks. In all three applications, over the different setups and datasets, BPA consistently improves its hosting methods, achieving new state-of-the-art results.

2. Relation to Prior Work

2.1. Related Techniques

Set-to-Set (or Set-to-Feature) Functions have been developed to act jointly on a set of items (typically features) and output an updated set (or a single feature), which are used for downstream inference tasks. Deep-Sets (Zaheer et al., 2017) formalized fundamental requirements from architectures that process sets. Point-Net (Qi et al., 2017) presented an influential design for learning local and global features on 3D point-clouds, while Maron et al. (2020) study the design of equi/in-variant layers. Unlike BPA, the joint processing in these methods is limited, amounting to weight-sharing between separate processes and joint aggregations.

Attention Mechanisms. The introduction of Relational Networks (Santoro et al., 2017) and Transformers (Vaswani et al., 2017) with their initial applications in vision models (Ramachandran et al., 2019) have lead to the huge impact of Vision Transformers (ViTs) (Dosovitskiy et al., 2020) in many vision tasks (Khan et al., 2021). While BPA can be seen as a self-attention module, it is very different, first, since it is *parameterless*, and hence can work at test-time on a pre-trained network. In addition, is can provide an explicit probabilistic global interpretation of the instance data.

Spectral Methods have been widely used as simple transforms applied on data that needs to undergo grouping or search based operations, jointly processing the set of features, resulting in a compact and perhaps discriminative representation. PCA (Pearson, 1901) provides a joint dimension reduction, which maximally preserves data variance, but does not necessarily improve feature affinities for downstream tasks. Spectral Clustering (SC) (Shi & Malik, 2000; Ng et al., 2001) is the leading non-learnable

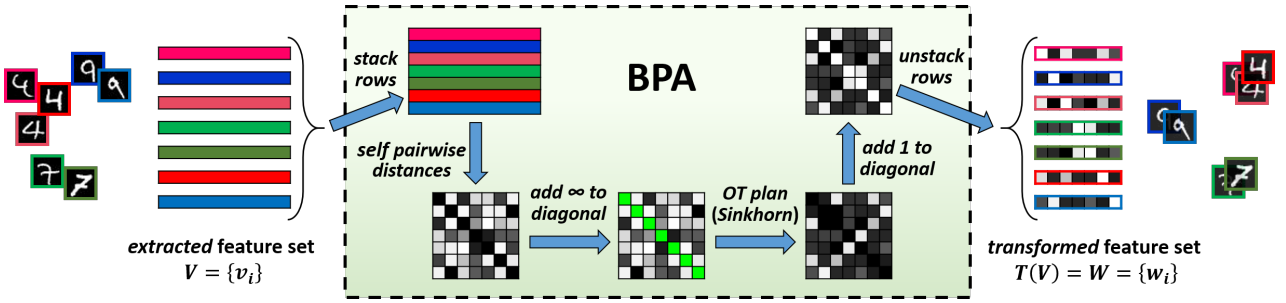

Figure 2. **The BPA transform:** illustrated on a toy 7 image 3-class MNIST example.

clustering method in use in the field. If we ignore its final clustering stage, SC consists of forming a pairwise affinity matrix which is normalized (Zass & Shashua, 2006) before extracting its leading eigenvectors, which form the final embedding. BPA is also based on normalizing an affinity matrix, but uses this matrix's rows as embedded features and avoids any further spectral decompositions, which are costly and difficult to differentiate through.

Optimal Transport (OT) problems are directly related to measuring distances between distributions or sets of features. Cuturi (2013) popularized the Sinkhorn algorithm which is a simple, differentiable and fast approximation of entropy-regularized OT, which has since been used extensively, for clustering (Lee et al., 2019; Asano et al., 2020), few-show-classification (Huang et al., 2019; Ziko et al., 2020; Hu et al., 2020; Zhang et al., 2021; Chen & Wang, 2021; Zhu & Koniusz, 2022), matching (Wang et al., 2019; Fey et al., 2020; Sarlin et al., 2020), representation learning (Caron et al., 2020; Asano et al., 2020), retrieval (Xie et al., 2020), person re-identification (Wang et al., 2022), style-transfer (Kolkin et al., 2019) and attention (Sander et al., 2022).

Our approach also builds on some attractive properties of the Sinkhorn solver. While our usage of Sinkhorn is extremely simple (see Algorithm 1), it is fundamentally different from all other OT usages we are aware of, since: (i) We compute the transport-plan between a set of features and *itself* - not between feature-sets and label/class-prototypes (Huang et al., 2019; Ziko et al., 2020; Hu et al., 2020; Zhang et al., 2021; Chen & Wang, 2021; Zhu & Koniusz, 2022; Lee et al., 2019; Asano et al., 2020; Xie et al., 2020; Wang et al., 2022; Kolkin et al., 2019), or between two different feature-sets (Wang et al., 2019; Fey et al., 2020; Sarlin et al., 2020; Sander et al., 2022); (ii) While others use the transport-plan to obtain distances or associations between features and features/classes, we use *its own rows* as new feature vectors for downstream tasks.

2.2. Instance-Specific Applications

Few-Shot Classification (FSC) is a branch of few-shot learning in which a classifier learns to recognize previously unseen classes given a limited number of labeled examples. In the *meta-learning* approach, the training data is split into tasks (or episodes) mimicking the test time tasks to which the learner is required to generalize. MAML (Finn et al., 2017) "learns to fine-tune" by learning a network initialization from which it can quickly adapt to novel classes. In ProtoNet (Snell et al., 2017), a learner is meta-trained to predict query feature classes, based on distances from support class-prototypes in the embedding space. The trainable version of BPA can be viewed as a meta-learning algorithm.

Subsequent works (Chen et al., 2018; Dhillon et al., 2020) advocate using larger and more expressive backbones, employing *transductive* inference, which fully exploits the data at inference, including unlabeled images. BPA is transductive, but does not make assumptions on (nor needs to know) the number of classes (ways) or items per class (shots), as it executes a general probabilistic grouping action.

Recently, *attention* mechanisms were shown to be effective for FSC (Kang et al., 2021; Zhang et al., 2020; Ye et al., 2020) and a number of works (Ziko et al., 2020; Huang et al., 2019; Hu et al., 2020; Zhang et al., 2021; Chen & Wang, 2021) have adopted Sinkhorn (Cuturi, 2013) as a parameterless unsupervised classifier that computes matchings between query embeddings and class centers. Sill-Net (Zhang et al., 2021) that augments training samples with illumination features and PTMap-SF (Chen & Wang, 2021) that proposes DCT-based feature embedding, are both based on PTMap (Hu et al., 2020). The state-of-the-art PMF (Hu et al., 2022), proposed a 3 stage pipeline of pre-training on external data, meta-training with labelled tasks, and fine-tuning on unseen tasks. BPA can be incorporated into these methods, immediately after their feature extraction stage.

Unsupervised Image Clustering (UIC) is the task of grouping related images, without any label information, into representative clusters. Naturally, the ability to measure the similarities among samples is a crucial aspect of UIC.

Recent methods have achieved tremendous progress in this task, towards closing the gap with supervised counterparts. The leading approaches directly learn to map images to labels, by constraining the training of an unsupervised classification model with different types of indirect loss functions. Prominent works in this area include DAC (Chang et al.,

2017), which recasts the clustering problem into a binary pairwise-classification framework and SCAN (Van Gansbeke et al., 2020) which builds on a pre-trained encoder that provides nearest-neighbor based constraints for training a classifier. The recent state-of-the-art SPICE (Niu et al., 2022), is a pseudo-labeling based method, which divides the clustering network into a feature model for measuring the instance-level similarity and a clustering head for identifying the cluster-level discrepancy.

Person Re-Identification (Re-ID) is the task identifying a certain person (identity) between multiple detected pedestrian images, from different non-overlapping cameras. It is challenging due to the scale of the problem and large variation in pose, background and illumination.

See Ye et al. (2021) for an excellent comprehensive survey on the topic. Among the most popular methods are OSNet (Zhou et al., 2019) that developed an efficient small-scale network with high performance and DropBlock (Top-DB-Net) (Quispe & Pedrini, 2020) which achieved state-of-the-art results by dropping a region block in the feature map for attentive learning. The Re-ID task is typically larger scale - querying thousands of identities against a target of tens of thousands. Also, the data is much more real-world compared to the carefully curated FSC sets.

3. The BPA Transform

3.1. Derivation

Assume we are given a task instance which consists of an inference problem over a set of n items $\{x_i\}_{i=1}^n$, where each of the items belongs to a space of input items $\Omega \subseteq \mathbb{R}^D$. The inference task can be modeled as $f_\theta(\{x_i\}_{i=1}^n)$, using a learned function f_θ, which acts on the set of input items and is parameterized by a set of parameters θ. Typically, such functions combine an initial feature extraction stage that is applied independently to each input item, with a subsequent stage of (separate or joint) processing of the feature vectors (see Fig. 1 Left or Right, respectively).

That is, the function f_θ takes the form $f_\theta(\{x_i\}_{i=1}^n) = G_\psi(\{F_\phi(x_i)\}_{i=1}^n)$, where F_ϕ is the feature extractor (or embedding network) and G_ψ is the task inference function, parameterized by ϕ and ψ respectively, where $\theta = \phi \cup \psi$.

The feature embedding $F : \mathbb{R}^D \to \mathbb{R}^d$, usually in the form of a neural-network (with $d \ll D$), could be either pre-trained, or trained in the context of the task function f, along with the inference function G.

For an input $\{x_i\}_{i=1}^n$, let us define the set of embedded features $\{v_i\}_{i=1}^n = \{F(x_i)\}_{i=1}^n$. In the following, we consider these sets of input vectors and features as real-valued row-stacked matrices $\mathcal{X} \in \mathbb{R}^{n\times D}$ and $\mathcal{V} \in \mathbb{R}^{n\times d}$.

We suggest a novel re-embedding of the feature set $\mathcal{V}$, using a transform, that we denote by T, in order to obtain a new set of features $\mathcal{W} = T(\mathcal{V})$, where $\mathcal{W} \in \mathbb{R}^{n\times n}$. The new feature set $\mathcal{W}$ has an explicit probabilistic interpretation, which is specifically suited for tasks related to classification, matching or grouping of items in the input set $\mathcal{X}$. In particular, $\mathcal{W}$ will be a symmetric, doubly-stochastic matrix (non-negative, with rows and columns that sum to 1), where the entry w_{ij} (for $i \neq j$) encodes the belief that items x_i and x_j belong to the same class or cluster.

The proposed transform $T : \mathbb{R}^{n\times d} \to \mathbb{R}^{n\times n}$ (see Fig. 2) acts on the original feature set $\mathcal{V}$ as follows. It begins by computing the squared Euclidean pairwise distances matrix $\mathcal{D}$, namely, $d_{ij} = ||v_i - v_j||^2$, which can be computed efficiently as $d_{ij} = 2(1 - cos(v_i, v_j)) = 2(1 - v_i \cdot v_j^T)$, when the rows of $\mathcal{V}$ are unit normalized. Or in a compact form, $\mathcal{D} = 2(\mathbf{1} - \mathcal{S})$, where $\mathbf{1}$ is the all ones $n \times n$ matrix and $\mathcal{S} = \mathcal{V} \cdot \mathcal{V}^T$ is the cosine affinity matrix of $\mathcal{V}$.

$\mathcal{W}$ will be computed as the optimal transport (OT) plan matrix between the n-dimensional all-ones vector $\mathbf{1}_n$ and itself, under the self cost matrix $\mathcal{D}_\infty$, which is the distance matrix $\mathcal{D}$ with a very (infinitely) large scalar replacing each of the entries on its diagonal (which were all zero), that enforces the affinities of each feature to distribute among the others. Explicitly, let $\mathcal{D}_\infty = \mathcal{D} + \alpha I$, where α is a very (infinitely) large constant and I is the $n \times n$ identity matrix.

$\mathcal{W}$ is defined to be the doubly-stochastic matrix that is the minimizer of the functional

$$\mathcal{W} = \underset{\mathcal{W}\in B_n}{\arg\min} \langle \mathcal{D}_\infty, \mathcal{W} \rangle \tag{1}$$

where B_n is the set (known as the Birkhoff polytope) of $n \times n$ doubly-stochastic matrices and $\langle \cdot, \cdot \rangle$ stands for the Frobenius (standard) dot-product.

This objective can be minimized using simplex or interior point methods with complexity $\Theta(n^3 \log n)$. In practice, we use the highly efficient Sinkhorn-Knopp method (Cuturi, 2013), which is an iterative scheme that optimizes an entropy-regularized version of the problem, where each iteration takes $\Theta(n^2)$. Namely:

$$\mathcal{W} = \underset{\mathcal{W}\in B_n}{\arg\min} \langle \mathcal{D}_\infty, \mathcal{W} \rangle - \frac{1}{\lambda} h(\mathcal{W}) \tag{2}$$

where $h(\mathcal{W}) = -\sum_{i,j} w_{ij} \log(w_{ij})$ is the Shannon entropy of $\mathcal{W}$ and λ is the entropy regularization parameter.

The *transport-plan* matrix $\mathcal{W}$ that is the minimizer of Equation (2) will become the result of our transform, after 'restoring' perfect affinities on the diagonal (replacing the diagonal entries from 0s to 1s) by $\mathcal{W} = \mathcal{W} + I$, where I is the $n \times n$ identity matrix. Our final set of features is $T(\mathcal{V}) = \mathcal{W}$ and each of its rows is the re-embedding of each of the corresponding features (rows) in $\mathcal{V}$. The BPA transform is given

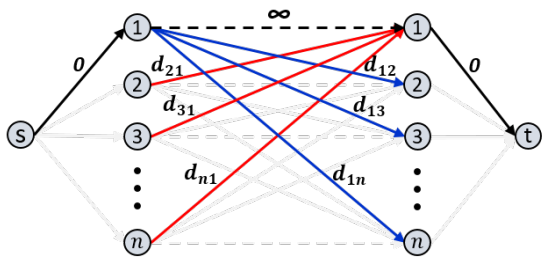

Figure 3. **The min-cost max-flow perspective**: Costs are shown.

in Algorithm 1 in the appendix, in PyTorch-style pseudo-code. Note that $\mathcal{W}$ is symmetric as a result of the symmetry of $\mathcal{D}$ and its own double-stochasticity. We next explain its probabilistic interpretation.

3.2. Probabilistic interpretation

The optimization problem in Equation (1) can be written more explicitly as follows:

$$\min_{\mathcal{W}} \langle \mathcal{D}_\infty, \mathcal{W} \rangle \qquad \text{s.t.} \qquad \mathcal{W} \cdot \mathbf{1}_n = \mathcal{W}^T \cdot \mathbf{1}_n = \mathbf{1}_n \tag{3}$$

which can be seen to be the same as:

$$\min_{\mathcal{W}} \langle \mathcal{D}, \mathcal{W} \rangle \qquad \text{s.t.} \qquad \begin{aligned} &\mathcal{W} \cdot \mathbf{1}_n = \mathcal{W}^T \cdot \mathbf{1}_n = \mathbf{1}_n \\ &w_{ii} = 0 \quad \text{for} \quad i = 1, \dots n \end{aligned} \tag{4}$$

since the use of the infinite weights on the diagonal of $\mathcal{D}_\infty$ is equivalent to using the original $\mathcal{D}$ with a constraint of zeros along the diagonal of $\mathcal{W}$.

The optimization problem in Equation (4) is in fact a fractional matching instance between the set of n original features and itself. It can be posed as a bipartite-graph min-cost max-flow instance (The problem of finding a min cost flow out of all max-flow solutions), as depicted in Fig. 3. The graph has n nodes on each side, representing the original features $\{v_i\}_{i=1}^n$ (the rows of $\mathcal{V}$). Across the two sides, the cost of the edge (v_i, v_j) is the distance d_{ij} and the edges of the type (v_i, v_i) have a cost of infinity (or can simply be removed). Each 'left' node is connected to a 'source' node S by an edge of cost 0 and similarly each 'right' node is connected to a 'target' (sink) node T. All edges in the graph have a capacity of 1 and the goal is to find an optimal fractional self matching, by finding a min-cost max-flow from source to sink. Note that the max-flow can easily be seen to be n, but a min-cost flow is sought among max-flows.

In this set-to-itself matching view, each vector v_i is fractionally matched to the set of all other vectors $\mathcal{V} - \{v_i\}$ based on the pairwise distances, but importantly taking into account the fractional matches of the rest of the vectors in order to satisfy the double-stochasticity constraint. The construction constrains the max flow to have a total outgoing flow of 1 from each 'left' node and a total incoming flow of 1 to each 'right' node. Therefore, the ith transformed feature

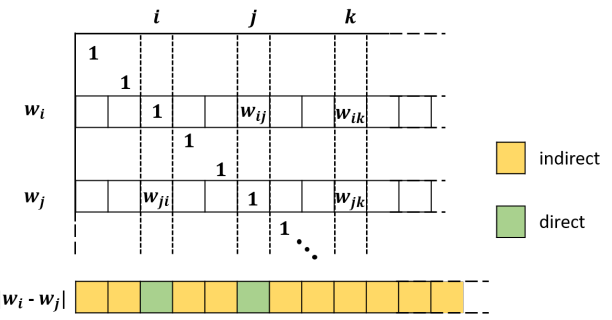

Figure 4. **The (symmetric) embedding matrix** $\mathcal{W}$ and the absolute difference between its ith and jth rows.

w_i (ith row of $\mathcal{W}$) is a *distribution* (non-negative entries, summing to 1), where $w_{ii} = 0$ and w_{ij} is the relative belief that features i and j belong to the same 'class'.

3.3. Properties

We can now point out some important properties of the proposed embedding, given by the rows of the matrix $\mathcal{W}$. Some of these properties can be observed in the toy 3-class MNIST digit example, illustrated in Fig. 2.

Interpretability of distances in the embedded space: An important property of our embedding is that each embedded feature encodes its distribution of affinities to all other features. In particular, the comparison of embedded vectors w_i and w_j (of items i and j in a set) includes both *direct* and *indirect* information about the similarity between the features. Refer to Figure 4 for a detailed explanation of this property. If we look at the different coordinates k of the absolute difference vector $a = |w_i - w_j|$, BPA captures (i) *direct affinity*: For k which is either i or j, it holds that $a_k = 1 - w_{ij} = 1 - w_{ji}$ [1]. This amount measures how high (close to 1) is the mutual belief of features i and j about one another. (ii) *indirect (3rd-party) affinity*: For $k \notin \{i, j\}$, we have $a_k = |w_{ik} - w_{jk}|$, which is a comparison of the beliefs of features i and j regarding the (third-party) feature k. The double-stochasticity of the transformed feature-set ensures that the compared vectors are similarly scaled (as distributions, plus 1 on the diagonal) and the symmetry further enforces the equal relative affinity between pairs.

As an example, observe the output features 4 and 5 in Fig. 2, that re-embed the 'green' features of the digit '7' images. As desired, these embedding are close in the target 7D space. The closeness is driven by both their closeness in the original space (coordinates 4 and 5) as well as the agreement on specific large differences from other images. This property is responsible for better separation between classes in the target domain, which leads to improved performance on tasks like classification, clustering or retrieval.

[1] Note: (i) $w_{ii} = w_{jj} = 1$; (ii) $w_{ij} = w_{ji}$ from symmetry of $\mathcal{W}$; (iii) all elements of $\mathcal{W}$ are ≤ 1 hence the $|\cdot|$ can be dropped ;

Parameterless-ness, **Differentiability** and **Equivariance**: These three properties are inherited from the Sinkhorn OT solver. The transform is parameterless, giving it the flexibility to be used in other pipelines, directly over different kinds of embeddings, without the harsh requirement of re-training the entire pipeline. Retraining is certainly possible, and beneficial in many situations, but not mandatory, as our experiments work quite well without it. Also, due to the differentiability of the Sinkhorn algorithm (Cuturi, 2013), back-propagating through BPA can be done naturally, hence it is possible to (re-)train the hosting network to adapt to BPA, if desired. The embedding works coherently with respect to any change of order of the input items (features). This can be shown by construction, since min-cost max-flow solvers as well as the Sinkhorn OT solver are equivariant with respect to permutations of their inputs.

Usage flexibility: Recall that BPA is applied on sets of features, typically computed by some embedding network and its output features are passed to downstream network components. Since BPA is *parameterless*, it can be simply inserted to any trained hosting network and since it is *differentiable*, it is possible to train the hosting network with BPA inside it. We therefore denote by $\mathbf{BPA}_{p}$ the basic drop-in usage of BPA, inserted into a *pretrained* network. This is the easiest and most flexible way to use BPA, nevertheless showing consistent benefits in the different tested applications. We denote by $\mathbf{BPA}_{t}$ the usage where the hosting network is *trained* with BPA within. It allows to adapt the hosting network's parameters to the presence of the transform, with the potential of further improving performance.

Transductive or Inductive: Note that BPA is a *transductive* method in the sense that it needs to jointly process the data, but in doing so, unlike many transductive methods, it does not make any limiting assumptions about the input structure, such as knowing the number of classes, or items per class. In any case, we consider the $\mathbf{BPA}_{p}$ and $\mathbf{BPA}_{t}$ variants to be transductive, regardless of the nature of the hosting network. Nevertheless, being transductive is possibly restrictive for certain tasks, for which test-time inputs might be received one-by-one. Therefore, we suggest a third usage type, $\mathbf{BPA}_{i}$, where the hosting network is trained with BPA inside (just like in $\mathbf{BPA}_{t}$), but BPA is not applied at inference (simply not inserted), hence the hosting network remains *inductive* if it was so in the first place.

Dimensionality: BPA has the unique property that the dimension of its embedded feature depends on (equals) the number of features in the set. Given a batch of n d-dimensional features $V \in \mathbb{R}^{n\times d}$, it outputs a batch of n n-dimensional features $W = \mathbf{BPA}(V) \in \mathbb{R}^{n\times n}$. On one hand, this is a desired property, since it is natural that the feature dimensionality (and capacity) depends on the complexity of the task, which typically grows with the number of features (Think of the inter-relations which are more complex to model). On the other hand, it might impose a problem in situations at which the downstream calculation that follows expects a specific feature dimension, for example with a pre-trained non-convolutional layer.

Table 1. **Feature-dimension control strategies**: Accuracy on ***5*-way *1*-shot** *MiniImagenet*. * marks the dimension of original 640d pre-trained resnet-12 features. # marks the size of a batch that includes a single 5-way 1-shot 15-query task ($80 = 5 \cdot (1 + 15)$), which is the output dimension of vanilla **BPA**. Best and second best results, per dimension, are in **Bold** and ***italics***.

input to ProtoNet / dim.	5	10	20	40	$80^{\#}$	640^{*}
V (orignal)	-	-	-	-	-	64.6
PCA(V)	66.2	65.7	64.4	64.1	64.3	-
SC(V)	66.8	58.2	46.2	38.3	25.5	-
$\mathbf{BPA}_{p}(V)$	-	-	-	-	***71.2***	-
$\mathbf{BPA}_{t}(V)$	-	-	-	-	**72.1**	-
$\mathbf{BPA}_{p}$**_Attn**(V)	-	-	-	-	-	***69.1***
$\mathbf{BPA}_{t}$**_Attn**(V)	-	-	-	-	-	**70.0**
$\mathbf{BPA}_{p}$**_Attn**(SC(V))	**69.1**	**69.1**	**68.1**	**68.5**	69.2	-
$\mathbf{BPA}_{p}$**_Attn**(PCA(V))	***67.1***	***67.8***	***67.5***	***67.6***	67.8	-

In order to make BPA usable in such cases, we propose an attention-like variant, **BPA_Attn**, in which the normalized BPA matrix is used to balance the input features without changing their dimension, by simple multiplication, i.e. $\mathbf{BPA\_Attn}(V) = \mathbf{BPA}(V) \cdot V$. This variant allows to maintain the original feature dimension d, or even a smaller dimension if desired, by applying dimension reduction on the original set of features prior to applying **BPA_Attn**.

In Table 1, we examine few-shot classification accuracy on *MiniImagenet* (Vinyals et al., 2016) with downstream classification by ProtoNet (Snell et al., 2017). Each classification instance consists of 80 images, encoded to 640-dimensional features by a pre-trained resnet-12 network. ProtoNet works on either: (i) the original feature set V (ii) its dimension reduced versions, calculated by either PCA or Spectral-Clustering (SC) (iii) vanilla **BPA** (iv) **BPA_Attn** on original or reduced features. As can be observed, the best accuracies are achieved by vanilla **BPA**, but the attention provided by BPA is able to stabilize performance across the entire range of dimensions.

Hyper-parameters and ablations: BPA has two hyper-parameters that were chosen through cross-validation and kept fixed for each application over all datasets. The number of Sinkhorn iterations for computing the optimal transport plan was fixed to 5 and entropy regularization parameter λ (Eq. (3.1)) was set to 0.1 for UIC and FSC and to 0.25 for ReID. In Appendix B we ablate these hyper-parameters as well as the scalability of BPA in terms of set-input size (Fig. 5) on few-shot-classification, and in Appendix D, we study its robustness to noise and feature dimensionality (Fig. 10) by a controlled synthetic clustering experiment.

Table 2. **Few-Shot Classification (FSC)** accuracy on *MiniImagenet*. Results are ordered by backbone (resnet-12, wrn-28-10, ViT small/base), each listing baseline methods and BPA variants. BPA improvements (colored percentages) are in comparison with each respective baseline hosting method (obtained by division). **Bold** and ***italics*** highlight best and second best results per backbone. ***T/I*** denotes transductive/inductive methods. (&) from Ziko et al. (2020); ($) from original paper; (#) our implementation;

method	*T/I*	network	*5*-way *1*-shot	*5*-way *5*-shot
ProtoNet(#)	*I*	ResNet	62.39	80.33
DeepEMD($)	*I*	ResNet	65.91	82.41
FEAT($)	*I*	ResNet	66.78	82.05
RENet($)	*I*	ResNet	67.60	82.58
ProtoNet-**BPA**$_p$	*T*	ResNet	67.34 (+7.9%)	81.84 (+1.6%)
ProtoNet-**BPA**$_i$	*I*	ResNet	64.36 (+3.1%)	81.82 (+1.8%)
ProtoNet-**BPA**$_t$	*T*	ResNet	67.90 (+8.8%)	83.09 (+3.2%)
ProtoNet(&)	*I*	WRN	62.60	79.97
PTMap($)	*T*	WRN	82.92	88.80
SillNet($)	*T*	WRN	82.99	89.14
PTMap-SF($)	*T*	WRN	***84.81***	***90.62***
PTMap-**BPA**$_p$	*T*	WRN	83.19 (+0.3%)	89.56 (+0.9%)
PTMap-**BPA**$_t$	*T*	WRN	84.18 (+1.5%)	90.51 (+1.9%)
SillNet-**BPA**$_p$	*T*	WRN	83.35 (+0.4%)	89.65 (+0.6%)
PTMap-SF-**BPA**$_p$	*T*	WRN	**85.59** (+0.9%)	**91.34** (+0.8%)
PMF($)	*I*	ViT-s	*93.10*	**98.00**
PMF-**BPA**$_p$	*T*	ViT-s	94.49 (+1.4%)	97.68 (-0.3%)
PMF-**BPA**$_i$	*I*	ViT-s	92.70 (-0.4%)	**98.00** (+0.0%)
PMF-**BPA**$_p$	*T*	ViT-s	**95.30** (+2.3%)	***97.90*** (-0.1%)
PMF($)	*I*	ViT-b	***95.30***	98.40
PMF-**BPA**$_p$	*T*	ViT-b	95.90 (+0.6%)	98.30 (-0.1%)
PMF-**BPA**$_i$	*I*	ViT-b	95.20 (-0.1%)	**98.70** (+0.3%)
PMF-**BPA**$_t$	*T*	ViT-b	**96.3** (+1.0%)	***98.5*** (+0.1%)

Table 3. **Few-Shot Classification (FSC)** accuracy on *CIFAR-FS*.

method	*T/I*	network	*5*-way *1*-shot	*5*-way *5*-shot
PTMap($)	*T*	WRN	87.69	90.68
SillNet($)	*T*	WRN	87.73	91.09
PTMap-SF($)	*T*	WRN	*89.39*	*92.08*
PTMap-**BPA**$_p$	*T*	WRN	87.37 (-0.4%)	91.12 (+0.5%)
SillNet-**BPA**$_p$	*T*	WRN	87.30 (-0.5%)	91.40 (+0.3%)
PTMap-SF-**BPA**$_p$	*T*	WRN	**89.94** (+0.6%)	**92.83** (+0.8%)
PMF($)	*I*	ViT-s	81.1	92.5
PMF-**BPA**$_p$	*T*	ViT-s	84.7 (+4.4%)	92.8 (+0.3%)
PMF-**BPA**$_i$	*I*	ViT-s	***84.80*** (+4.5%)	***93.40*** (+0.9%)
PMF-**BPA**$_t$	*T*	ViT-s	**88.90** (+9.6%)	**93.80** (+1.4%)
PMF($)	*I*	ViT-b	84.30	92.20
PMF-**BPA**$_p$	*T*	ViT-b	88.2 (+4.6%)	94 (+1.9%)
PMF-**BPA**$_i$	*I*	ViT-b	***87.10*** (+3.3%)	***94.70*** (+2.7%)
PMF-**BPA**$_t$	*T*	ViT-b	**91.00** (+7.9%)	**95.00** (+3.0%)

4. Results

In this section, we experiment with BPA on three applications: Few-Shot Classification (Sec. 4.1), Unsupervised Image Clustering (Sec. 4.2) and Person Re-Identification (Sec. 4.3). In each, we achieve state-of-the-art results, by merely using current state-of-the-art methods as hosting networks of the BPA transform. Perhaps more importantly, we demonstrate the flexibility and simplicity of applying BPA in these setups, with improvements in the entire range of testing, including different hosting methods, different feature embeddings of different complexity backbones and whether retraining the hosting network or just dropping-in BPA and performing standard inference. To show the simplicity of inserting BPA into hosting algorithms, we provide pseudocodes for each of the experiments in Appendix C.

4.1. Few-Shot Classification (FSC)

Our main experiment is a comprehensive evaluation on the standard few-shot classification benchmarks *MiniImagenet* (Vinyals et al., 2016) and *CIFAR-FS* (Bertinetto et al., 2019), with detailed results in Tables 2 and 3 respectively. We evaluate the performance of the proposed BPA, applying it to a variety of FSC methods including the recent state-of-the-art (PTMap (Hu et al., 2020), SillNet (Zhang et al., 2021), PTMap-SF (Chen & Wang, 2021) and PMF (Hu et al., 2022)) as well as to conventional methods like the popular ProtoNet (Snell et al., 2017). While in the *MiniImagenet* evaluation we include a wide range of methods and backbones, in the *CIFAR-FS* evaluation we focus on the state-of-the-art methods and configurations.

For each evaluated 'hosting' method, we incorporate BPA into the pipeline as follows. Given an FSC instance, we transform the entire set of method-specific feature representations using BPA, in order to better capture relative information. The rest of the pipeline is resumed, allowing for both inference and training. Note that BPA flexibly fits into the FSC task, with no required knowledge or assumptions regarding the setting (# of ways, shots or queries).

The basic 'drop-in' **BPA**$_p$ consistently, and in many cases also significantly, improves the hosting method performance, including state-of-the-art, across all benchmarks and backbones with accuracy improvement of around 3.5% and 1.5% on the 1- and 5- shot tasks. This improvement without retraining the embedding backbone shows BPA's effectiveness in capturing meaningful relationships between features in a very general sense. When re-training the hosting network with BPA inside, in an end-to-end fashion, **BPA**$_t$ provides further improvements, in almost every method, with averages of 5% and 3% on the 1- and 5- shot tasks.

While most of the leading methods are transductive, our inductive version, **BPA**$_i$, can be seen to steadily improve on inductive methods like ProtoNet and PMF, without introducing transductive inference. This further emphasizes the generality and applicability of our method.

4.2. Unsupervised Image Clustering (UIC)

Next, we evaluate BPA in the unsupervised domain, using the unsupervised image clustering task, with the additional challenge of capturing the relation between features that were learned without labels. To do so, we adopt SPICE (Niu et al., 2022), a recent method that has shown phenomenal success in the field. In SPICE, training is divided into 3 phases: (i) unsupervised representation learning (using

Table 4. **Unsupervised Image Clustering (UIC)** results on *STL-10* (Coates et al., 2011), *CIFAR-100-20* (Krizhevsky & Hinton, 2009) and *CIFAR-100-20* (Krizhevsky & Hinton, 2009).

benchmark	*STL-10*			*CIFAR-10*			*CIFAR-100-20*		
network	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI
k-means	0.192	0.125	0.061	0.229	0.087	0.049	0.130	0.084	0.028
DAC	0.470	0.366	0.257	0.522	0.396	0.306	0.238	0.185	0.088
DSEC	0.482	0.403	0.286	0.478	0.438	0.340	0.255	0.212	0.110
IDFD	0.756	0.643	0.575	0.815	0.711	0.663	0.425	0.426	0.264
SPICE$_s$	0.908	0.817	0.812	0.838	0.734	0.705	0.468	0.448	0.294
SPICE	***0.938***	***0.872***	***0.870***	***0.926***	***0.865***	***0.852***	***0.538***	0.567	***0.387***
SPICE$_s$-**BPA**$_t$	0.912	0.823	0.821	0.880	0.784	0.769	0.494	0.477	0.334
SPICE-**BPA**$_t$	**0.943**	**0.880**	**0.879**	**0.933**	**0.870**	**0.866**	**0.550**	***0.560***	**0.402**

Table 5. **Image Re-Identification (Re-ID)** results on *CUHK03* (Li et al., 2014) and *Market-1501* (Zheng et al., 2015).

benchmark	*CUHK03-det*		*CUHK03-lab*		*Market-1501*	
network	mAP	Rank-1	mAP	Rank-1	mAP	Rank-1
MHN	65.4	71.7	72.4	77.2	85.0	95.1
SONA	76.3	79.1	79.2	81.8	88.6	95.6
OSNet	67.8	72.3	–	–	84.9	94.8
Pyramid	74.8	78.9	76.9	81.8	88.2	**95.7**
TDB	72.9	75.7	75.6	77.7	85.7	94.3
TDB$_{RK}$	***87.1***	***87.1***	***89.1***	***89.0***	**94.0**	***95.3***
TDB-**BPA**$_p$	77.9	80.4	80.4	82.6	*88.1*	94.4
TDB$_{RK}$-**BPA**$_p$	**87.9**	**88.0**	**89.5**	**89.8**	**94.0**	95.0

MoCo (He et al., 2019) over a resnet-34 backbone); (ii) clustering-head training, with result termed SPICE$_s$; and (iii) a joint training phase (using FixMatch (Sohn et al., 2020) over a wrn backbone), result termed SPICE.

We insert BPA into phase (ii), clustering-head training, as follows. Given a batch of representations, SPICE assigns class pseudo-labels to the nearest neighbors of the most probable samples (k samples with the highest probability per class). In the original work, SPICE uses the dot-product of the MoCo features to find the neighbors. Instead, we transform each batch of MoCo features using BPA and use the same dot-product on the resulting informative BPA features to find a more reliable set of neighbors. We experiment on 3 standard datasets, *STL-10* (Coates et al., 2011), *CIFAR-10* and *CIFAR-100-20* (Krizhevsky & Hinton, 2009), while keeping all original SPICE implementation hyper-parameters unchanged. We report both SPICE$_s$ and SPICE results, as in the original work (Niu et al., 2022).

Table 4 summarizes the experiment, in terms of clustering Accuracy (ACC), Normalized Mutual Information (NMI), and Adjusted Rand Index (ARI). It is done for the two stages of SPICE, with and without BPA, along with several other baselines. The results show a significant improvement of SPICE$_s$-**BPA**$_t$ over SPICE$_s$ (just by applying BPA to the learned features), with an average increase of 5% in NMI and 8% in ARI. The advantage brought by the insertion of BPA carries on to the joint-processing stage (**BPA**$_t$ over SPICE$_s$), though with a smaller average increase of 0.1% in NMI and 2.2% in ARI, leading to new state-of-the-art results on these datasets. These results demonstrate the relevance of BPA to unsupervised feature learning setups and its possible potential to other applications in this area.

4.3. Person Re-Identification (Re-ID)

We explore the application of BPA to large-scale instances and datasets by considering the person re-identification task (Ye et al., 2021). Given a set of *query* images and a large set of *gallery* images, the task is to rank the similarities of each query against the entire gallery. This is typically done by learning specialized image features that are compared by Euclidean distances. BPA is used to replace such pre-computed image features, by a well balanced representation with strong relative information, that is jointly computed over the union of query and gallery features. BPA is applied on pre-trained TopDBNet (Quispe & Pedrini, 2020) resnet-50 features and tested on the large-scale ReID benchmarks *CUHK03* (Li et al., 2014) (both 'detected' and 'labeled') as well as the *Market-1501* (Zheng et al., 2015) set, reporting mAP (mean Average Precision) and Rank-1 metrics.

In Table 5, TDB and TDB$_{RK}$ are shorthands for using TopDBNet features, before and after re-ranking (Zhong et al., 2017). There is a consistent benefit in applying BPA to these state-of-the-art features, prior to the distance computations, with a significant average increase of over 5% in mAP and 4% in Rank-1 prior to re-ranking and a modest increase of 0.5% in both measures after ranking. These results demonstrate that BPA can handle large-scale instances (with thousands of features) and successfully improve performance measures in such retrieval oriented tasks.

5. Conclusions, Limitations and Future Work

We presented a novel feature-embedding approach for set-input grouping-related tasks such as clustering, classification and retrieval. The proposed BPA feature-set transform is non-parametric, differentiable, efficient, easy to use and is shown to capture complex relations between the set-input items. Applying BPA to the tasks of few-shot-classification, unsupervised-image-clustering and person-re-identification, whether by insertion into a pre-trained network or by re-training the hosting network, has shown across-the-board improvements, setting new state-of-the-art results.

In future work, we plan to address current limitations and explore potential extensions. BPA is currently limited to producing features that represent *relative* information, within the set-items. It could possibly be applied to tokens (e.g. patches) of a single item (e.g. image), similar to transformers, perhaps dropping the equivariance property and utilizing spatial encoding, to improve non-relative representations. In addition, it could be useful for guiding contrastive self-supervised learning, where embeddings are trained by relative information of augmented views.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Asano, Y., Rupprecht, C., and Vedaldi, A. Self-labelling via simultaneous clustering and representation learning. In *International Conference on Learning Representations (ICLR)*, 2020.

Bertinetto, L., Henriques, J. F., Torr, P., and Vedaldi, A. Meta-learning with differentiable closed-form solvers. In *International Conference on Learning Representations (ICLR)*, 2019.

Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., and Joulin, A. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in neural information processing systems (NeurIPS)*, 2020.

Chang, J., Wang, L., Meng, G., Xiang, S., and Pan, C. Deep adaptive image clustering. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.

Chen, W.-Y., Liu, Y.-C., Kira, Z., Wang, Y.-C. F., and Huang, J.-B. A closer look at few-shot classification. In *International Conference on Learning Representations (ICLR)*, 2018.

Chen, X. and Wang, G. Few-shot learning by integrating spatial and frequency representation. *arXiv:2105.05348*, 2021.

Coates, A., Ng, A., and Lee, H. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 2011.

Cuturi, M. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2013.

Dhillon, G. S., Chaudhari, P., Ravichandran, A., and Soatto, S. A baseline for few-shot image classification. In *International Conference on Learning Representations (ICLR)*, 2020.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv:2010.11929*, 2020.

Fey, M., Lenssen, J. E., Morris, C., Masci, J., and Kriege, N. M. Deep graph matching consensus. *arXiv:2001.09621*, 2020.

Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning (ICML)*, 2017.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. Momentum contrast for unsupervised visual representation learning. *arXiv:1911.05722*, 2019.

Hu, S. X., Li, D., Stühmer, J., Kim, M., and Hospedales, T. M. Pushing the limits of simple pipelines for few-shot learning: External data and fine-tuning make a difference. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, 2022.

Hu, Y., Gripon, V., and Pateux, S. Leveraging the feature distribution in transfer-based few-shot learning. In *arXiv:2006.03806*, 2020.

Huang, G., Larochelle, H., and Lacoste-Julien, S. Are few-shot learning benchmarks too simple? solving them without task supervision at test-time. *arXiv:1902.08605*, 2019.

Kang, D., Kwon, H., Min, J., and Cho, M. Relational embedding for few-shot classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

Khan, S., Naseer, M., Hayat, M., Zamir, S. W., Khan, F. S., and Shah, M. Transformers in vision: A survey. *arXiv:2101.01169*, 2021.

Kolkin, N., Salavon, J., and Shakhnarovich, G. Style transfer by relaxed optimal transport and self-similarity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Korman, S. and Avidan, S. Coherency sensitive hashing. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2015.

Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images. 2009.

Kuhn, H. W. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2, 1955.

Lee, J., Lee, Y., Kim, J., Kosiorek, A., Choi, S., and Teh, Y. W. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International Conference on Machine Learning (ICML)*, 2019.

Li, W., Zhao, R., Xiao, T., and Wang, X. Deepreid: Deep filter pairing neural network for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

Maron, H., Litany, O., Chechik, G., and Fetaya, E. On learning sets of symmetric elements. In *International Conference on Machine Learning (ICML)*, 2020.

Ng, A., Jordan, M., and Weiss, Y. On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems (NeurIPS)*, 2001.

Niu, C., Shan, H., and Wang, G. Spice: Semantic pseudo-labeling for image clustering. *IEEE Transactions on Image Processing (TIP)*, 2022.

Pearson, K. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, 1901.

Qi, C. R., Su, H., Mo, K., and Guibas, L. J. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Quispe, R. and Pedrini, H. Top-db-net: Top dropblock for activation enhancement in person re-identification. *International Conference on Pattern Recognition (ICPR)*, 2020.

Ramachandran, P., Parmar, N., Vaswani, A., Bello, I., Levskaya, A., and Shlens, J. Stand-alone self-attention in vision models. *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Ravi, S. and Larochelle, H. Optimization as a model for few-shot learning. In *International Conference on Learning Representations (ICLR)*, 2017.

Sander, M. E., Ablin, P., Blondel, M., and Peyré, G. Sinkformers: Transformers with doubly stochastic attention. In *International conference on artificial intelligence and statistics (AISTATS)*. PMLR, 2022.

Santoro, A., Raposo, D., Barrett, D. G., Malinowski, M., Pascanu, R., Battaglia, P., and Lillicrap, T. A simple neural network module for relational reasoning. *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Sarlin, P.-E., DeTone, D., Malisiewicz, T., and Rabinovich, A. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Shi, J. and Malik, J. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence (PAMI)*, 2000.

Snell, J., Swersky, K., and Zemel, R. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Sohn, K., Berthelot, D., Li, C.-L., Zhang, Z., Carlini, N., Cubuk, E. D., Kurakin, A., Zhang, H., and Raffel, C. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *arXiv:2001.07685*, 2020.

Van Gansbeke, W., Vandenhende, S., Georgoulis, S., Proesmans, M., and Van Gool, L. Scan: Learning to classify images without labels. In *European Conference on Computer Vision (ECCV)*, 2020.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., and Wierstra, D. Matching networks for one shot learning. In *Proceedings of the 30th International Conference on Neural Information Processing Systems (NeurIPS)*, 2016.

Wang, J., Zhang, Z., Chen, M., Zhang, Y., Wang, C., Sheng, B., Qu, Y., and Xie, Y. Optimal transport for label-efficient visible-infrared person re-identification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022.

Wang, R., Yan, J., and Yang, X. Learning combinatorial embedding networks for deep graph matching. In *Proceedings of the IEEE/CVF international conference on computer vision (ICCV)*, 2019.

Xie, Y., Dai, H., Chen, M., Dai, B., Zhao, T., Zha, H., Wei, W., and Pfister, T. Differentiable top-k with optimal transport. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Ye, H.-J., Hu, H., Zhan, D.-C., and Sha, F. Few-shot learning via embedding adaptation with set-to-set functions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Ye, M., Shen, J., Lin, G., Xiang, T., Shao, L., and Hoi, S. C. Deep learning for person re-identification: A survey and outlook. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2021.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R. R., and Smola, A. J. Deep sets. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Zass, R. and Shashua, A. Doubly stochastic normalization for spectral clustering. *Advances in neural information processing systems (NeurIPS)*, 2006.

Zhang, C., Cai, Y., Lin, G., and Shen, C. Deepemd: Few-shot image classification with differentiable earth mover's

distance and structured classifiers. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Zhang, H., Cao, Z., Yan, Z., and Zhang, C. Sill-net: Feature augmentation with separated illumination representation. *arXiv:2102.03539*, 2021.

Zheng, L., Shen, L., Tian, L., Wang, S., Wang, J., and Tian, Q. Scalable person re-identification: A benchmark. In *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015.

Zhong, Z., Zheng, L., Cao, D., and Li, S. Re-ranking person re-identification with k-reciprocal encoding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Zhou, K., Yang, Y., Cavallaro, A., and Xiang, T. Omni-scale feature learning for person re-identification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

Zhu, H. and Koniusz, P. Ease: Unsupervised discriminant subspace learning for transductive few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Ziko, I. M., Dolz, J., Granger, E., and Ayed, I. B. Laplacian regularized few-shot learning. In *International Conference on Machine Learning (ICML)*, 2020.

Appendix

The Appendix includes the following sections:

A. PyTorch-style BPA Implementation

We provide in Algorithm 1 a PyTorch Style implementation that fully aligns with the description in the paper as well as with our actual implementation that was used to execute all of the experiments. In Appendix C we further demonstrate the "insertions" of BPA into hosting methods, for each of our three main applications.

Notice mainly that: (i) The transform can easily be dropped-in, using the simple one-line call: X = **BPA**(X). (ii) It is fully differentiable (as Sinkhorn and the other basic operations are). (iii) The transform does not need to know (or even assume) anything about the number of features, their dimension, or distribution statistics among classes (e.g. whether balanced or not).

It follows the simple steps of: (i) Computing Euclidean self pairwise distances (using cosine similarities between unit normalized input features); (ii) Avoiding self-matching by placing infinity values on the distances matrix diagonal; (iii) Applying a standard Sinkhorn procedure, given the distance matrix and the only 2 (hyper-) parameters with their fixed values: entropy regularization parameter λ and the number of row/col iterative normalization steps. Note that Sinkhorn defaultly maps between source and target vectors of ones; (iv) Restoring the perfect self-matching probabilities of one, along the diagonal.

Algorithm 1 **BPA** transform on a set of n features.
input: $n \times d$ matrix V **output**: $n \times n$ matrix W

```
def BPA(V):
  # compute self pairwise-distances
  D = 1 - pwise_cosine_sim(V/V.norm())
  # infinity self-distances on diagonal
  D_inf = D.fill_diagonal(10e9)
  # compute optimal transport plan
  W = Sinkhorn(D_inf,lambda=.1,iters=5)
  # stretch affinities to [0,1]
  W = W/W.max()
  # self-affinity on diagonal to 1
  return W.fill_diagonal(1)
```

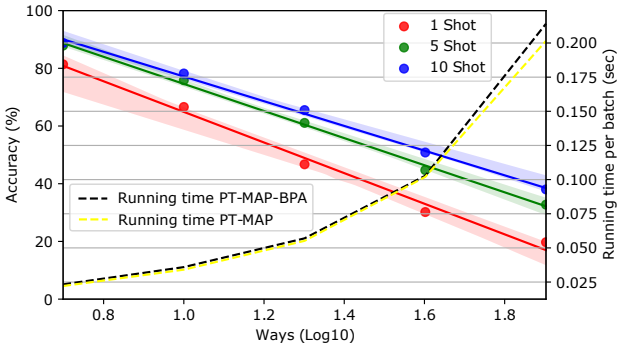

Figure 5. **BPA scaling in terms of accuracy and efficiency**.

B. Ablation Studies

B.1. Scalability (accuracy, runtime vs. input size)

Being a transductive module, the accuracy and efficiency of the BPA transform depend on the number of inputs that are processed as a batch. Recall that BPA is a drop-in addition that usually follows feature extraction and precedes further computation - e.g. k-means for clustering, or (often transductive) layers in FSC and ReID.

The ReID experiment is a good stress-test for BPA, since we achieve excellent results for batch sizes of up to ∼15K image descriptors. In terms of runtime, although BPA's complexity is quadratic in sample size, its own (self) runtime is empirically negligible compared to that of the processing that follows, in all applications tested.

Typical FSC tasks sizes ($(shots+queries)\cdot ways$) are small: $100 = (5 + 15) \cdot 5$ at the largest. To concretely address this matter, we test a resnet-12 PTMap-**BPA**$_p$ on large-scale FSC, following (Dhillon et al., 2020), on the Tiered-Imagenet dataset and report accuracy for 1/5/10-shot (15-query) tasks for an increasing range of ways. The results, shown in Fig. 5, show that: (i) Total runtime, where BPA is only a small contributor (compare black vs. yellow dashed line), increases gracefully (notice log10 x-axis) even for extremely large FSC tasks of $4000 = (10 + 15) \cdot 160$ images; (ii) Our accuracy scales as expected - following the observation in (Dhillon et al., 2020) that it changes logarithmically with ways (straight line in log-scale).

B.2. Sinkhorn Iterations

In Table 6 we ablate the number of normalization iterations in the Sinkhorn-Knopp (SK) (Cuturi, 2013) algorithm at test-time. We measured accuracy on the validation set of *MiniImagenet* (Vinyals et al., 2016), using ProtoNet-**BPA**$_p$ (which is the non-fine-tuned drop-in version of BPA within ProtoNet (Snell et al., 2017)). As was reported in prior works following (Cuturi, 2013), we empirically observe that a very small number of iterations provide rapid convergence, with diminishing return for higher numbers of iterations.

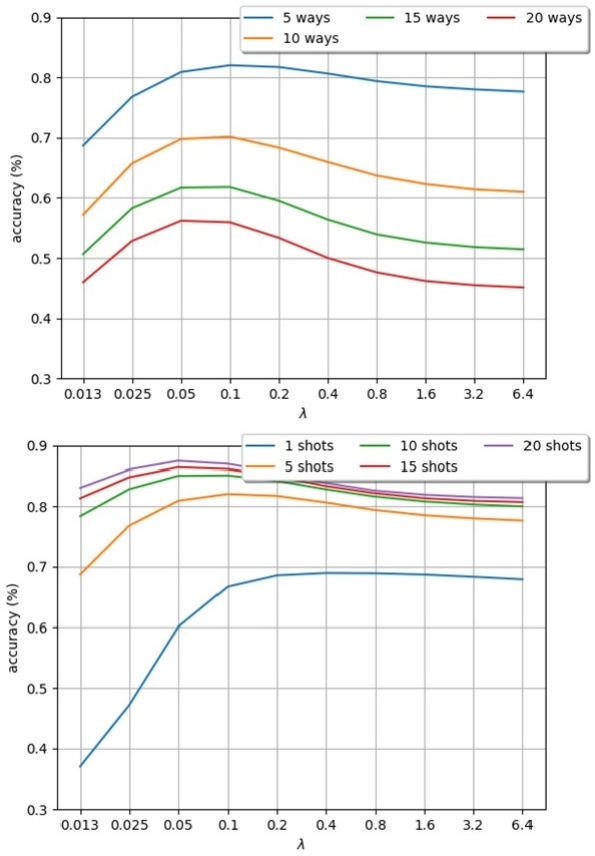

Figure 6. **Ablation of entropy regularization parameter λ using the *Few-Shot-Classification* (FSC) task**: Considering different 'ways' (**top**), and different 'shots' (**bottom**). See text for details.

We observed similar behavior for other hosting methods, and therefore chose to use a fixed number of 5 iterations throughout the experiments.

Table 6. **Sinkhorn iterations ablation study:** See text for details.

method	iters	*5*-way *1*-shot	*5*-way *5*-shot
ProtoNet-**BPA**$_p$	1	70.71	83.79
ProtoNet-**BPA**$_p$	2	71.10	84.01
ProtoNet-**BPA**$_p$	4	71.18	84.08
ProtoNet-**BPA**$_p$	8	71.20	84.10
ProtoNet-**BPA**$_p$	16	71.20	84.10

B.3. Sinkhorn Entropy Regularization λ

We measured the impact of using different values of the optimal-transport entropy regularization parameter λ (the main parameter of the Sinkhorn algorithm) on a variety of configurations (ways and shots) in Few-Shot-Classification (FSC) on *MiniImagenet* (Vinyals et al., 2016) in Fig. 6 as well as on the Person-Re-Identification (RE-ID) experiment on Market-1501 (Zheng et al., 2015) in Fig. 7. In both cases, the ablation was executed on the validation set.

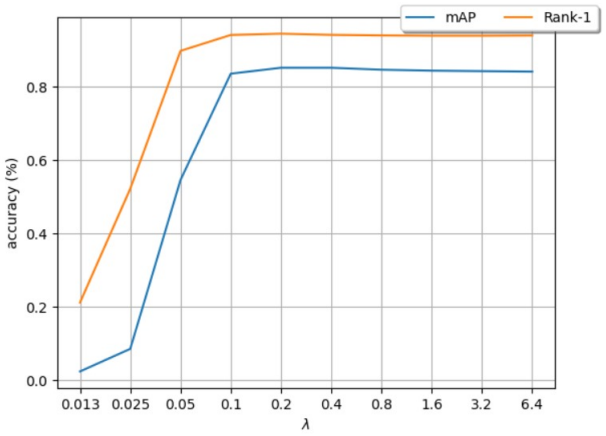

Figure 7. **Ablation of entropy regularization parameter λ using the *Person-Re-Identification* (Re-ID) task**. Accuracy vs. λ, using the validation set of Market-1501 (Zheng et al., 2015) and considering both mAP and Rank-1 measures. See text for details.

For FSC, in Fig. 6, the **top** plot shows that the effect of the choice of λ is similar across tasks with a varying number of ways. The **bottom** plot shows the behavior as a function of λ across multiple shot-values, where the optimal value of λ can be seen to have a certain dependence on the number of shots. Recall that we chose to use a fixed value of $\lambda = 0.1$, which gives an overall good accuracy trade-off. Note that a further improvement could be achieved by picking the best values for the particular cases. Notice also the log-scale of the x-axes to see that performance is rather stable around the chosen value.

For Re-ID, in Fig. 7, we experiment with a range of λ values on the validation set of the Market-1501 dataset. The results (shown both for mAP and rank-1 measures) reveal a strong resemblance to those of the FSC experiment in Fig. 6, however, the optimal choices for λ are slightly higher, which is consistent with the dependence on the shots number, since the re-ID tasks are typically large ones. We found that a value of $\lambda = 0.25$ gives good results across both datasets.

B.4. BPA vs. Naive Baselines

In Fig. 8, we ablate different simple alternatives to BPA, with the PTMap (Hu et al., 2020) few-shot-classifier as the 'hosting' method, using *MiniImagenet* (Vinyals et al., 2016). Each result is the average of 100 few-shot episodes, using a WRN-28-10 backbone feature encoder. In blue is the baseline of applying no transform at all, using the original features. In orange - using BPA. In gray and yellow, respectively, are other naive ways of transforming the features, where the affinity matrix is only row-normalized ('softmax') or not normalized at all ('cosine') before taking its rows as the output features. It is empirically evident that only BPA outperforms the baseline consistently, which is due to the properties that we had proved regarding the transform.

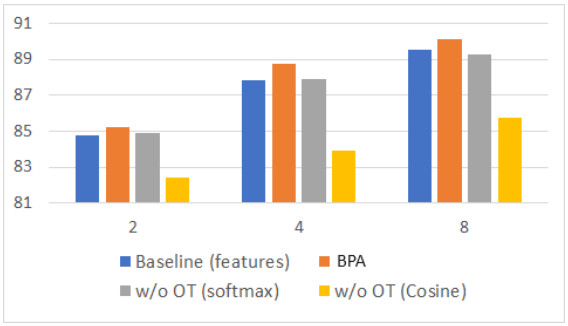

Figure 8. **Comparison of BPA to different baselines over different configurations** in few-shot learning tasks over *MiniImagenet* (Vinyals et al., 2016). Created by measuring accuracy (y-axis) over a varying number of shots (x-axis), with fixed 5-ways and 15-queries. See text for details.

C. BPA Insertion into Hosting Algorithms

C.1. PTMap (Hu et al., 2020) (Few-Shot Classification)

We present the pseudo-code for utilizing BPA within the PTMap pipeline, as outlined in Alg. C.1. The only alteration from the original implementation pertains to row 5, wherein the support and query sets are concatenated and transformed using BPA. This approach can be extended to a wide range of distance-based methodologies, thus providing a simple and versatile solution to a variety of applications.

Algorithm 2 PTMap *training* **and** *inference*

inputs: $\mathbf{x_s}, \mathbf{x_q}$ # support, query images
$\ell_\mathbf{s}, (\ell_\mathbf{q})$ # support, (query) labels
f_ϕ # pre-trained embedding network

$\mathbf{f_s} = f_\phi(\mathbf{x_s})$, $\mathbf{f_q} = f_\phi(\mathbf{x_q})$ # extract features
$(\mathbf{f_s} \cup \mathbf{f_q}) = \texttt{BPA}(\mathbf{f_s} \cup \mathbf{f_q})$ # BPA transformed features
$\mathbf{c}_j = \frac{1}{s} \cdot \sum_{\mathbf{f} \in \mathbf{f_s}, \ell_\mathbf{s}(\mathbf{f})=j} \mathbf{f}, \forall j$ # init class centers
repeat:
· $\mathbf{L}_{ij} = \|\mathbf{f}_i - \mathbf{c}_j\|^2, \ \forall i, \mathbf{f}_i \in \mathbf{f}_q$ # feature-center dists
· $\mathbf{M} = Sinkhorn(\mathbf{L}, \lambda)$ # S-horn soft assignments
· $\mathbf{c}_j \leftarrow \mathbf{c}_j + \alpha(g(\mathbf{M}, j) - \mathbf{c}_j), \ \forall j$ # update centers
$\hat{\ell}_\mathbf{q}(\mathbf{f}_i) = \arg\max_j (\mathbf{M}[i, j])$ # prediction per $\mathbf{f}_i \in \mathbf{f}_q$
if *inference*:
return $\hat{\ell}_\mathbf{q}$ # query predictions
else (*training*):
update f_ϕ by ∇_ϕC-Entropy($\mathbf{M}$, $\ell_\mathbf{q}$) # grad-desc.

C.2. SPICE (Niu et al., 2022) (Unsupervised Clustering)

In our implementation of SPICE, as detailed in the paper, we utilize BPA during phase 2 of the algorithm (clustering-head training). Specifically, as depicted in Alg. C.2, we transform the features using BPA, batch-wise, before conducting a nearest-neighbor search. Afterwards, we retrieve the pseudo-labels and resume with the original features, as in the original implementation.

Algorithm 3 SPICE *training*

Phase (i): pre-train embedding network f_ϕ
Phase (ii): train clustering network c_θ
repeat per batch $\mathbf{x}$:
· $\mathbf{f} = f_\phi(\mathbf{x})$ # extract features
· $\mathbf{f}^{\text{BPA}} = \texttt{BPA}(\mathbf{f})$ # BPA transformed features
· Find 3 most confident samples per cluster (use $\mathbf{f}$)
· Compute cluster centers as their means (use $\mathbf{f}^{\text{BPA}}$)
· Find nearest-neighbors of each center (use $\mathbf{f}^{\text{BPA}}$)
· Assign them to the cluster (as pseudo-labels)
· Use pseudo-labels to train (update) c_θ
Phase (iii): jointly fine-tune f_ϕ **and** c_θ

C.3. TopDBNet (Quispe & Pedrini, 2020) (Person ReID)

Finally, Alg. C.3 illustrates the application of BPA during inference in the context of Person ReID. Typically, the query identity search within the gallery involves identifying the nearest sample to each query. In our implementation, we adopt the same methodology, with the additional step of transforming the concatenated set of query and gallery features, using the BPA transform prior to the search.

Algorithm 4 TopDBNet *inference*

inputs: $\mathbf{x_g}, \mathbf{x_q}$ # gallery images, query images
f_ϕ # pre-trained embedding network

extract features
$\mathbf{f_g} = f_\phi(\mathbf{x_g})$, $\mathbf{f_q} = f_\phi(\mathbf{x_q})$
transform them with BPA
$(\mathbf{f_g} \cup \mathbf{f_q}) = \texttt{BPA}(\mathbf{f_g} \cup \mathbf{f_q})$
return gallery image with closest feature
return $\underset{\{j:\mathbf{f}_j \in \mathbf{f_g}\}}{\text{argmin}} \ \|\mathbf{f}_i - \mathbf{f}_j\|$ for every $\{i : \mathbf{f}_i \in \mathbf{f_q}\}$

D. Clustering on the Sphere - a Case Study

We demonstrate the effectiveness of BPA using a controlled synthetically generated clustering experiment, with $k = 10$ cluster centers that are distributed uniformly at random on a d-dimensional unit-sphere, and 20 points per cluster (200 in total) that are perturbed around the cluster centers by Gaussian noise of increasing standard deviation, of up to 0.75, followed by a re-projection back to the sphere by dividing each vector by its L_2 magnitude. See Fig. 9 for a visualization of the $3D$ case, for several noise STDs. Following the random data generation, we also apply dimen-

sionality reduction with PCA to $d = 50$, if $d > 50$.

We performed the experiment over a logarithmic 2D grid of combinations of data dimensionalities d in the range $[10, 1234]$ and Gaussian in-cluster noise STD in the range $[0.1, 0.75]$. Each point is represented by its d-dimensional coordinates vector, where the baseline clustering is obtained by running k-means on these location features. In addition, we run k-means on the set of features that has undergone BPA. Hence, the benefits of the transform (embedding) are measured indirectly through the accuracy[2] achieved by running k-means on the embedded vs. original vectors.

Evaluation results, in terms of Normalized Mutual Information (NMI) and Adjusted Rand Index (ARI), are reported in Fig. 10, averaged over 10 runs, as a function of either dimensionality (for different noise STDs) or noise STDs (for different dimensionalities). The results show (i) general gains and robustness to wide ranges of data dimensionality (ii) the ability of BPA to find meaningful representations that enable clustering quality to degrade gracefully with the increase in cluster noise level. Note that the levels of noise are rather high, as they are relative to a unit radius sphere.

[2]Accuracy is measured by comparison with the optimal permutation of the predicted labels, found by the Hungarian Algorithm (Kuhn, 1955).

Figure 9. **Clustering on the sphere: Data Generation.** 10 Random cluster centers on the unit sphere, perturbed by increasing noise STD.

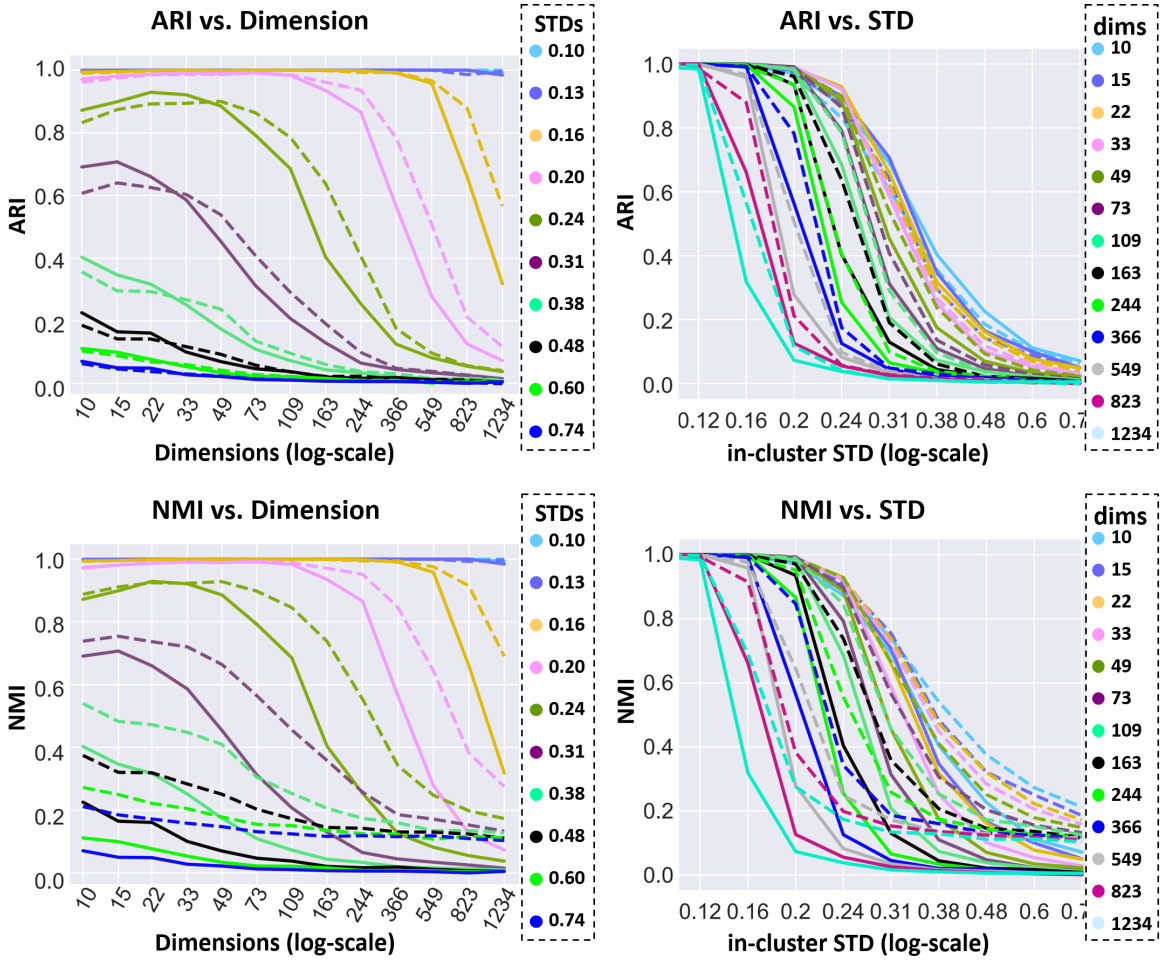

Figure 10. **Clustering on the sphere: Detailed Results.** Clustering measures (top: ARI, bottom: NMI) of k-means, using BPA features (*dashed* lines) vs. original features (*solid* lines). For both measures - the higher the better. Shown over different configurations of feature dimensions d (left) and noise levels σ (right).