entropy

MDPI

Article

Learning Ordinal Embedding from Sets

Aïssatou Diallo [1] and Johannes Fürnkranz [2,*]

[1] Research Training Group AIPHES, Technische Universität Darmstadt, 64289 Darmstadt, Germany; diallo@ke.tu-darmstadt.de
[2] Computational Data Analytics, FAW, Johannes Kepler University Linz, 4040 Linz, Austria
* Correspondence: juffi@faw.jku.at

Abstract: Ordinal embedding is the task of computing a meaningful multidimensional representation of objects, for which only qualitative constraints on their distance functions are known. In particular, we consider comparisons of the form "Which object from the pair (j,k) is more similar to object i?". In this paper, we generalize this framework to the case where the ordinal constraints are not given at the level of individual points, but at the level of sets, and propose a distributional triplet embedding approach in a scalable learning framework. We show that the query complexity of our approach is on par with the single-item approach. Without having access to features of the items to be embedded, we show the applicability of our model on toy datasets for the task of reconstruction and demonstrate the validity of the obtained embeddings in experiments on synthetic and real-world datasets.

Keywords: ordinal embedding; sets; representation learning

Citation: Diallo, A.; Fürnkranz, J. Learning Ordinal Embedding from Sets. *Entropy* **2021**, *23*, 964. https://doi.org/10.3390/e23080964

Academic Editors: Fabio Aiolli and Mirko Polato

Received: 21 June 2021
Accepted: 19 July 2021
Published: 27 July 2021

1. Introduction

The objective of an ordinal embedding algorithm is to find a low-dimensional Euclidean representation of a number of abstract items, for which no feature representation or numerical distance information is available. Instead, the learner has access to a set of comparisons where for a quadruple of points i, j, l, and k from an abstract space $\mathcal{X}$, it is specified whether the pair (i,j) is closer to each other than the pair (l,k), i.e., whether $\delta(i,j) < \delta(l,k)$ for some latent distance function δ [1,2]. A special case of this problem results when points i and l coincide, i.e., when the learner has access to triplet comparisons, which specify for three objects i, j, and k whether i is closer to j or to k.

Several tasks in Machine Learning (ML) and Information Retrieval (IR) depend on some underlying notion of similarity between objects. Supervised learning attempts to assign labels to objects based on the notion of feature similarity, while unsupervised learning attempts to discover hidden patterns of similarity between objects. Prior work in this area has focused on various aspects of the ordinal embedding problem. Agarwal et al. [3] provided a flexible and modular algorithm with proven convergence guarantees. Later work focused on explaining disagreement among human assessors, modeled by noisy triplets, and more tailored to crowd-sourcing [4–6]. Terada and von Luxburg [7] promised embeddings that recovered the exact point position with application for density estimation. Other works focused on theoretical aspects of the ordinal embedding problem. For example, Kleindessner and von Luxburg [8] proved that under reasonable distributional assumptions, it is possible to recover an embedding that places all objects within a reasonable error range of their correct position, and Jamieson and Nowak [9] showed that the triplet selection phase is as critical as the algorithm itself and derived a lower bound on the number of triples necessary for recovering an ordinal embedding. In our own prior work [10], we proposed a method for finding distributional ordinal embeddings, i.e., embeddings that can also model and explicitly represent the uncertainty of the location of a point recovered from noisy comparisons.

A common application for ordinal embedding methods is crowd-sourcing. In practice, the triplet comparisons are often obtained by combining the answers from multiple human

assessors that are asked to give subjective feedback. Moreover, it has been shown that eliciting such ordinal feedback is more reliable than the feedback based on the question "how close is item i to item j" [11]. The unequivocal advantage is that this method is a solution for the issue of comparing subjective scales across different crowd-workers.

In this work, we extend this concept to the case when the learner is not presented with triplets of abstract items, but rather, sets of abstract items. We are now given (unordered) sets of items and training triples that provide the form "the set of items J is closer to the sets of items I than the set of items K", where $I, J, K \subset \mathcal{X}$. Note that the sets may have overlaps, so that each element $x \in \mathcal{X}$ may occur in multiple sets. The task is now to output a meaningful representation of all items $x \in \mathcal{X}$ in a low-dimensional space that respects the observed constraints. Obviously, this problem is a generalization of the classical ordinal embedding problem, where each set only consists of a single element. The set-based formulation is particularly useful when the number of abstract items is very large and/or the number of times the oracle that yields the training information can be interrogated is limited. How can we build a model that deals with sets of abstract values and outputs sets of low-dimensional representation while satisfying the triplet constraints? This paper aims at answering this question. We summarize our main contributions as follows:

- A Set-valued Ordinal Embedding (SetOE) is proposed to embed data points in a low-dimensional space. We reformulate the classical ordinal embedding problem based on single data points into a generalization based on sets while assuring the permutation invariance necessary when dealing with the set. We develop an architecture to allow for conditioning with possibly different sizes of sets and adapt the margin-based loss for set-valued input;
- We propose a distributional approach that does not rely on the features of the individual data points for the ordinal embedding problem with sets. We motivate the advantages of such a setting and explain the properties we use to enable this;
- Experiments on both artificial and reals datasets demonstrate the validity of our approach for embedding datasets of considerable size in a significantly low-dimensional space (e.g., two). We evaluate our approach on several datasets. First, we present a proof-of-concept with different synthetic datasets. Then, we escalate the complexity of the tasks to the MNIST dataset, poker, and more real-word datasets such as Reuters.

The remainder of the paper is organized as follows: In Section 2, we formally introduce the ordinal embedding problem and lay down the mathematical preliminaries, as well as the notation used throughout the paper. In Section 3, we present our approach for a set-valued ordinal embedding, which we evaluate in Section 4 with empirical studies in a variety of datasets. Finally, Section 5 collects related work on ordinal embedding, set-valued input, and representation learning, before we draw some conclusions in Section 6.

2. Ordinal Embedding

In this section, we formally state the ordinal embedding problem and establish the notation, for which we follow [12]. $\|\cdot\|$ denotes the ℓ_2 norm. $\mathcal{S}_+^d$ is the set of all positive-definite matrices. In the scope of this work, we only focus on Gaussian distributions, which belong to the family of parametrized probability distributions $z_{h,\mathbf{a},\mathbf{A}}$ having a location vector $\mathbf{a} \in \mathbb{R}^d$, which represents the shift of the distribution, a scale parameter $\mathbf{A} \in \mathcal{S}_+^d$, which represents the statistical dispersion of the distribution, and a characteristic generator function h. Specifically, for Gaussian distributions, the scale parameter coincides with the covariance matrix $var(z_{h,\mathbf{a},\mathbf{A}}) = \mathbf{A}$. From now on, we denote Gaussian distributions (or embeddings) as $z_{(h,\mathbf{a},\mathbf{A})} = \mathcal{N}(\mathbf{a}, \mathbf{A})$.

2.1. Classical Ordinal Embedding

The ordinal embedding problem, also called nonmetric multidimensional scaling [1,2], aims at obtaining the corresponding embeddings in a low-dimensional space. Consider n items in an abstract space $\mathcal{X}$, which, without loss of generality, we represent by their indices $[n] = 1, \ldots, n$. It is worth mentioning that no explicit representation of the items is

available, so it is not possible to analytically express the dissimilarity between the items. We thus assume a latent underlying dissimilarity (or similarity) function $\delta : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$, which cannot be directly observed, but information on δ is indirectly available via a set of training triples.

Let $\mathcal{T} := \{\langle i, j, k\rangle : 1 \leq i \neq j \neq k \leq n\}$ be a set of unique triplets of elements in $\mathcal{X}$. We further assume an oracle $\mathcal{O}$, which provides a binary label $+1$ or -1 for each triplet $\langle i, j, k\rangle \in \mathcal{T}$, indicating whether $\delta(i, j) < \delta(i, k)$ holds or not:

$$\mathcal{O}(\langle i, j, k\rangle) = \begin{cases} +1 & \text{if } \delta(i, j) < \delta(i, k) \\ -1 & \text{if } \delta(i, j) > \delta(i, k) \end{cases} \tag{1}$$

Note that at this stage, we do not require δ to be a metric. Together, $\mathcal{T}$ and $\mathcal{O}$ represent the observed ordinal constraints on distances.

The learning problem problem can now be formally defined as follows:

Definition 1 (Ordinal embedding)**.** *Given n points $\{1, \ldots, n\}$ in an abstract space $\mathcal{X}$, a set of triplets $\mathcal{T} \subset \mathcal{X}^3$, and an oracle $\mathcal{O} : \mathcal{X}^3 \to \{-1, 1\}$, which provides information about a latent similarity function δ as specified in (1), the* ordinal embedding problem *consists of finding a suitable embedding function $\phi : \mathcal{X} \to \mathbb{R}^d$, such that:*

$$\operatorname{sgn}(\|\phi(i) - \phi(k)\| - \|\phi(i) - \phi(j)\|) = \mathcal{O}(\langle i, j, k\rangle) \tag{2}$$

2.2. Distributional Ordinal Embedding

An extension to the classical ordinal embedding problem is to learn probabilistic embeddings in lieu of the conventional Euclidean embeddings, taking advantage of the fact that vectors can be considered as an extreme case of probability measures, namely Dirac [12]. For this purpose, we focus on the family of elliptical distributions, more precisely Gaussian distributions, which enjoy many advantages. The main results were extracted from our previous work [10], which extended the ordinal embedding problem defined in Section 2.1 from Euclidean embeddings to Gaussian embeddings.

Hence, the considered problem becomes:

Definition 2 (Probabilistic ordinal embedding)**.** *Suppose $\mathcal{T} \subset \mathcal{X}^3$ is a set of triplets over $\mathcal{X}$ and $\mathcal{O} : \mathcal{X}^3 \to \{-1, 1\}$ is an oracle as defined in (1). Let $\mathbf{Z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_n\}$ be the desired probabilistic embedding, where each of the original points $\mathbf{x}_i$ is mapped to the probability distribution parameterized by $\mathbf{z}_i$ and $\Delta(.,.)$, a distance function between distributions.* Probabilistic ordinal embedding *is the problem of finding a function $\psi : \mathcal{X} \to \mathbf{Z}$ that maps each point $i \in \mathcal{X}$ to a probability distribution $\mathbf{z}_i = \psi(i)$, such that:*

$$\operatorname{sgn}(\Delta(\mathbf{z}_i, \mathbf{z}_j) - \Delta(\mathbf{z}_i, \mathbf{z}_k)) = O(\langle i, j, k\rangle), \tag{3}$$

for $\langle i, j, k\rangle \in \mathcal{T}$.

This definition requires a distance measure Δ between distributions. For this purpose, we selected the *Wasserstein distance* [13], also known as the *Earth mover's distance*, which has been previously used as a loss function for supervised learning [14] and in several applications.

2.2.1. The Two-Wasserstein Distance

In Optimal Transport (OT) theory, the Wasserstein or Kantorovich–Rubinstein metric is a distance function defined between probability distributions (measures) on a given metric space M. The squared Wasserstein metric for two arbitrary probability measures $\mu, \nu \in \mathcal{P}(\mathbb{R}^d)$ is defined as:

$$W_2^2(\mu, \nu) \stackrel{\text{def}}{=} \inf_{X \sim \mu, Y \sim \nu} \mathbb{E}_{\|X - Y\|^2}$$

In the general case, it is difficult to find analytical solutions for the Wasserstein distance. However, a closed-form solution exists in the case of Gaussian distributions. Let $\alpha \stackrel{\text{def}}{=} \mathcal{N}(\mathbf{a}, \mathbf{A})$ and $\beta \stackrel{\text{def}}{=} \mathcal{N}(\mathbf{b}, \mathbf{B})$, where $\mathbf{a}, \mathbf{b} \in \mathbb{R}^d$ and $\mathbf{A}, \mathbf{B} \in \mathcal{S}_+^d$ are positive semi-definite. Hence:

$$W_2^2(\alpha, \beta) = \|\mathbf{a} - \mathbf{b}\|^2 + \mathfrak{B}^2(\mathbf{A}, \mathbf{B}) \tag{4}$$

where $\mathfrak{B}^2$ is the *squared Bures metric* [15], defined as:

$$\mathfrak{B}^2(\mathbf{A}, \mathbf{B}) \stackrel{\text{def}}{=} \operatorname{Tr}(\mathbf{A} + \mathbf{B} - 2(\mathbf{A}^{\frac{1}{2}}\mathbf{B}\mathbf{A}^{\frac{1}{2}})^{\frac{1}{2}}) \tag{5}$$

When $\mathbf{A} = \operatorname{diag} \mathbf{d_A}$ and $\mathbf{B} = \operatorname{diag} \mathbf{d_B}$ are diagonal, W_2^2 simplifies to the sum of two terms:

$$W_2^2(\alpha, \beta) = \|\mathbf{a} - \mathbf{b}\|^2 + \mathfrak{h}^2(\mathbf{d_A}, \mathbf{d_B}) \tag{6}$$

where $\mathfrak{h}^2(\mathbf{d_A}, \mathbf{d_B}) \stackrel{\text{def}}{=} \|\sqrt{\mathbf{d_A}} - \sqrt{\mathbf{d_B}}\|^2$ is the *squared Hellinger distance* [16] between the diagonal $\mathbf{d_A}$ and $\mathbf{d_B}$.

2.2.2. Learning Gaussian Embeddings

As mentioned in Definition 2, our goal is to learn a function that maps each item i to a parametrized probability distribution $\mathbf{z}_i$, such that the two-Wasserstein distances between the embeddings satisfy as many triplets as possible. In our case, we use a d-dimensional Gaussian embedding, so that $\mathbf{z}_i = (\mu_i, \Sigma_i)$. Let E_{ij} be the energy function between two items (i, j) [17], which characterizes our energy-based learning approach. In particular, we set $E_{ij} = W_2^2(\mathbf{z}_i, \mathbf{z}_j)$. Finally, the corresponding optimization problem is the following:

$$\max_{\mathbf{z}_1, \ldots \mathbf{z}_n \in \mathbb{R}^d} \sum_{t=(i,j,k) \in \mathcal{T}} \mathcal{O}(t) \cdot \operatorname{sgn}(E_{ij} - E_{ik}) \tag{7}$$

which is discrete, nonconvex, and NP-hard. For these reasons, a relaxation of this optimization problem is needed. We make the choice of using the hinge loss $\mathcal{L}((t = \langle i, j, k \rangle, \mathcal{O}(t))$, a well-established loss function in contrastive metric learning, as a convex surrogate:

$$\mathcal{L} = \sum_{t=\langle i,j,k \rangle \in \mathcal{T}} \max(1 - \mathcal{O}(t) \cdot (E_{ij} - E_{ik}), 0) \tag{8}$$

The empirical performance of embedding methods is evaluated by the *empirical error*, also called the *triplet error*.

$$Err = \frac{1}{|T'|} \sum_{\langle i,j,k \rangle \in \mathcal{T}} \mathbb{1}[(y \cdot \operatorname{sgn}(E_{ij} - E_{ik})) = 1] \tag{9}$$

3. Ordinal Embedding for Sets

This section contains our primary contribution: an approach for encoding sets of abstract items into sets of low-dimensional vectors. As previously established for the ordinal embedding problem, the only supervision given is in the form of triplet comparisons by an oracle $\mathcal{O}(t)$ that takes as the input the triplet of sets $t = \langle I, J, K \rangle$ and returns a value $\{-1, +1\}$.

3.1. Problem Statement

As in conventional ordinal embedding, we consider n items in the abstract space $\mathcal{X}$, which we represent without loss of generality by their indices $[n] = 1, \ldots, n$. Without loss of generality, we represent a set X as an unordered collection of indices of size k_X, i.e., $X = \{x_1, \ldots, x_{k_X}\} \subset \mathcal{X}$.

The input to our model is a triplet of sets $t = \langle I, J, K \rangle$. The sets can be overlapping, so that each item $i \in \mathcal{X}$ may occur in arbitrarily many sets. Furthermore, each set X can have different cardinalities k_X, so a constant set size is not a prerequisite of our approach.

However, in the following, if the context is clear, we omit the set index and denote the cardinality of each set as k. Analogous to (1), we assume that each triplet in the set of training triplets $\mathcal{T}$ has been labeled by an oracle $\mathcal{O}$ as:

$$\mathcal{O}(\langle I, J, K \rangle) = \begin{cases} +1 & \text{if } \delta(I,J) < \delta(I,K) \\ -1 & \text{if } \delta(I,J) > \delta(I,K) \end{cases} \tag{10}$$

where $\delta(.,.)$ is a latent, unspecified similarity function between sets.

The goal is to learn a mapping function that takes as the input a set of indices $\mathcal{X}$ and a training set of labeled triplets $\mathcal{T}$ defined over elements in $\mathcal{X}$ and outputs a set of vectors in $\mathbb{R}^d$ corresponding to the embedding of the individual items that compose $\mathcal{X}$. Formally, we can define the problem as follows:

Definition 3 (Set-based ordinal embedding)**.** *Let $\mathcal{X}$ be an abstract space of items, which we denote with $\{1, \ldots, n\}$, and $\mathcal{P}_\mathcal{X} = 2^\mathcal{X} \times 2^\mathcal{X} \times 2^\mathcal{X}$ the space of all triples of subsets of $\mathcal{X}$. Given $\mathcal{T} \subset \mathcal{P}_\mathcal{X}$ and an oracle $\mathcal{O} : \mathcal{P}_\mathcal{X} \to \{-1, 1\}$, which provides information about a latent similarity function δ as specified in (10), the* set-based ordinal embedding problem *consists of finding a suitable embedding function $\phi : \mathcal{X} \to \mathbb{R}^d$, such that:*

$$\operatorname{sgn}(\| \operatorname{agg}(\Phi(I)) - \operatorname{agg}(\Phi(K)) \| - \| \operatorname{agg}(\Phi(I)) - \operatorname{agg}(\Phi(J)) \|) = \mathcal{O}(\langle I, J, K \rangle) \tag{11}$$

where $\Phi(.)$ denotes the elementwise extension of $\phi(.)$ to sets and $\operatorname{agg}(.) : \mathbb{R}^{d \times k} \to \mathbb{R}^d$ is an aggregation operator defined over elements in $\mathbb{R}^d$.

3.2. Set Encoding

Clearly, the ordinal embedding for sets problem is a natural generalization of the classical ordinal embedding problem as defined in Definition 1: The generic input for our model is not a single item, but a set of items $X = \{x_1, \ldots, x_k\}$ of size k, where each x_i is an indexed item of $\mathcal{X}$. The output of the model is a set of feature vectors of dimensionality d represented by the matrix $\mathbf{Y} = \Phi(X) \in \mathbb{R}^{d \times k}$ with the column elements $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_k]$, where Φ is the learned mapping function.

In order to properly deal with sets, all the operations that compose Φ need to have the properties of permutation equivariance and permutation invariance. That is to say that Φ should not rely on the arbitrary order of the elements of the input set. The approach proposed in this paper relies on set encoders. A *set encoder* is a model that encodes a set of elements into feature vectors in a latent space. They are built as a composition of permutation-equivariant operations with a permutation-invariant operation at the end. Specifically, we assume that the function Φ can be decomposed into an elementwise function $\phi(.)$, which can be independently applied to each element, i.e.,

$$\Phi(X) = [\phi(x_1), \ldots, \phi(x_k)] = [\mathbf{y}_1, \ldots, \mathbf{y}_k]. \tag{12}$$

Essentially, the function $\phi(.)$ corresponds to the elementwise function of Definition 1. The resulting $\Phi(.)$ is permutation equivariant because its defining function $\phi(.)$ is applied to every element individually. Hence, it does not rely on the arbitrary order of the input set, and the the order of the output will adapt to any change in the order of the input.

Since the supervision is available only for the set and not for the single components, intuitively, the vectors $\{\mathbf{y}_i\}_{i=1}^k$ need to be aggregated into a single vector using an aggregation function $\operatorname{agg}(.)$. Obviously, $\operatorname{agg}(.)$ also needs to respect the property of permutation invariance. Multiple operations abide by this rule, for example the sum, average, min, or max. This allows us to obtain a representation of the set and its elements regardless of the order in which of the set elements are presented. In our experiments, we focus on the *centroid*, the center of mass $\bar{\mathbf{y}} \in \mathbb{R}^d$, i.e.,

$$\operatorname{agg}(\mathbf{Y}) \stackrel{\text{def}}{=} \bar{\mathbf{y}} = \frac{1}{k} \sum_k \mathbf{y}_k. \tag{13}$$

3.3. Distributional Embeddings for Sets

Learning ordinal embeddings from triplet comparisons based on sets is a problem that can lead to substantial approximations and imprecisions. One of the main advantages of the distributional approach outlined in Section 2.2 is that it is possible to represent and address severe perturbations in the data. As stated earlier, the representation through probability measures naturally allows encapsulating the uncertainty about the representation. Hence, following this approach, we further generalize our approach to a distributional embedding, where each set in the target space corresponds to a probability distribution.

To that end, we chose the Gaussian distribution $\mathcal{N}(\mu, \Sigma)$ characterized by a location vector μ and a covariance matrix Σ, i.e., we associated each set X with a distribution $\mathbf{z}_X = (\mu_X, \Sigma_X)$, where $\mu_X = \bar{\mathbf{y}}$, i.e., the centroid (13) of the set, and, following what is detailed in Section 2.2, a diagonal covariance matrix Σ_X. Overall, such a distributional embedding for sets allows obtaining a meaningful representation for each set, while being bounded by the components of the sets themselves.

The updates performed on a single training triplet are illustrated in Algorithm 1. Essentially, it takes a set triplet $\langle I, J, K \rangle$ and updates the item embeddings for all the elements in these sets, so that the hinge loss (8), which is based on the Wasserstein distances between the distribution of the item embeddings of the elements in each of the three sets, is reduced.

Algorithm 1 Distributional ordinal embeddings from set constraints.

Require: set of items $\mathcal{X}$, set of training triplets $\mathcal{T}$, oracle $\mathcal{O}$
Ensure: set embeddings $\mathbf{Y}$ of points in $\mathcal{X}$
1: initialize $\mathbf{Y}$ randomly
2: **for all** $(t = \langle I, J, K \rangle) \in \mathcal{T}$ **do**
3: **for all** $(S \in \{I, J, K\}$ **do**
4: $Y_S = \{\mathbf{y}_1, \ldots, \mathbf{y}_{|S|}\}$ column vectors of $\mathbf{Y}$ corresponding to S
5: $\mathbf{z}_S \leftarrow \mathcal{N}(\boldsymbol{\mu}_S, \Sigma_S)$ {mean and variance of Y_S}
6: **end for**
7: $l \leftarrow \max(1 - \mathcal{O}(t)(W(\mathbf{z}_I, \mathbf{z}_J) - W(\mathbf{z}_I, \mathbf{z}_K)), 0)$ {compute hinge loss}
8: $\mathbf{Y}_\cdot \leftarrow \mathbf{Y}_\cdot - \eta \frac{\partial l}{\partial \mathbf{Y}}$ {gradient descent step}
9: **end for**

3.4. Deep Set Encoder

We represent the elementwise embedding function $\phi(.)$ as a deep neural network, as illustrated in Figure 1. While our work relates to numerous architectures proposed in metric learning such as that in [18], our deep neural encoder is fundamentally different because of the nature of the problem. The most distinctive point is that we do not have access to the features of the items we aim to embed. In fact, our model learns a representation of the items based on a random input to the encoder. In particular, we chose as inputs random vectors on input dimension $h = 64$ sampled from $\mathcal{N}(0, I_h)$. A first deep encoder, namely two-layer MLP with ReLU, $\phi_\theta(\cdot)$ maps these random inputs into d-dimensional outputs. These are then aggregated to $\mu_\theta(\cdot)$ and fed to produce the variance $\Sigma_\theta(\cdot)$, a function, which is again represented with a deep forward network.

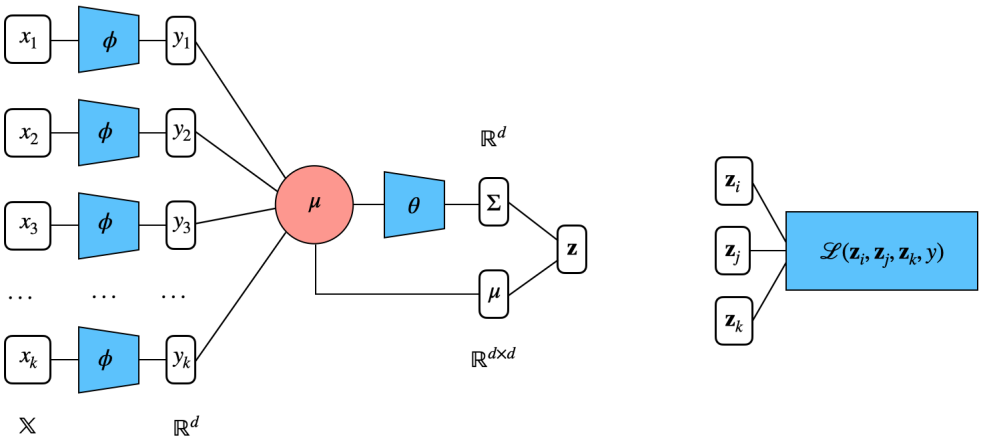


Figure 1. Distributional ordinal embeddings for sets. Each element of the input set i is fed to identical 2-layer MLP and outputs y_i. These feature representations are averaged into μ, the centroid of the set. Finally, θ takes as input μ and outputs the covariance matrix Σ (in our case, it is reduced to the diagonal of the covariance matrix). Coupled with the previously obtained μ, these vectors constitute the distributional embedding of a given input set. The Wasserstein-based hinge loss allows the optimization for learning the low-dimensional representation of the elements of a triplet of sets.

3.5. Complexity

The training complexity is linear in the size of $\mathcal{T}$, which is the set of all triplets, and bounded by $\mathcal{O}(n^3)$. However, a well-chosen sampling strategy may decrease this bound. It has been shown by Jamieson and Nowak [9] that the minimum number of triplets to recover an ordinal embedding is $\Omega(nd \log n)$ in $\mathbb{R}^d$. We adapted this result to the setting in which the parameters to be learned are a mean vector in $\mathbb{R}^d$ and a covariance matrix $\mathcal{S}_+^d$. Hence, the dimensionality can be considered to be $d' = d + d^2$ and $\mathcal{O}(d^2)$. Thus, the newly recovered lower bound for the triplets becomes $\Omega(d^2 n \log n)$, which is still polynomial in d and $\mathcal{O}(n \log n)$. Since ordinal embeddings typically map into a low-dimensional space, this is not a drastic loss in efficiency. Moreover, it is worth mentioning that a low number of epochs was needed for convergence for all experiments. Finally, the computational bottleneck when dealing with Wasserstein distance in its closed-form is computing the matrix square roots of the scale parameters. However, as we opted to learn diagonal covariances, hence this problem is not present in our approach.

3.6. Practical Tricks

Besides relying on the optimization of the energy-based max-margin loss (8), we applied some regularization to the learning process. We observed that no regularization is needed for learning the location vectors. However, the covariance matrix needs to be bounded, since the main goal of our approach is to obtain perceptual embeddings. Hence, we constrained the covariance matrix to lie within the hypercube $[0, C]^d$, C being a chosen constant. We chose to focus on diagonal covariance because we argue the rotation angle is not easily interpretable to appreciate the similarity between items and that the principal axes are sufficient to appreciate the uncertainty of the representation. Thus, the regularization is achieved by bounding each element of the covariance matrix, $\Sigma_{ii} = \max(\Sigma_{ii}, C)$. Moreover, we adapted our approach to be able to handle sets of variable size. First, we padded all sets in a batch to allow for efficient computation. We then provided an additional mask feature m_i for each set P_i that indicates whether it is a regular element of the set $(m_i(k) = 1)$ or padding element $(m_i(k) = 0)$. This mask is useful for computing the centroid of the set through the weighted average and the covariance of the set.

4. Experiments

Our main objective was to investigate the effectiveness of our approach for the ordinal embedding problem. With this objective in mind, we evaluated our model in two different settings. First, we performed experiments with reconstruction tasks on synthetic datasets with particular shapes. These sets of experiments were particularly useful as a controlled environment because the ground truth was available. Then, we applied our approach to real-world datasets for more complex data in order to assess the performance of our model in real cases.

In all experiments, we used the following hyperparameters: $l_r = 1e-3$ as the learning rate, batch size 512, $h = 64$ as the hidden size of the input layer, and $d = 2$, the output size of the embeddings. We compared our proposed model to a model that learns to embed a set simply as its centroid. In order to evaluate our approach, we used as the metric the Procrustes distance, as well as the triplet error (9) between the ground truth and learned embeddings.

4.1. Synthetic Datasets

Data. In this section, we present a series of experiments that use reconstruction tasks in order to illustrate the capabilities of our approach. We followed the experimental setting of [19]. More specifically, we used 4 2-dimensional synthetic datasets generated with the `scikit-learn` package in Python. The datasets were:

(i) Gaussian isotropic blobs;
(ii) A large circle containing a smaller circle in 2D;
(iii) Two interwoven spirals;
(iv) Two interleaving half circles;

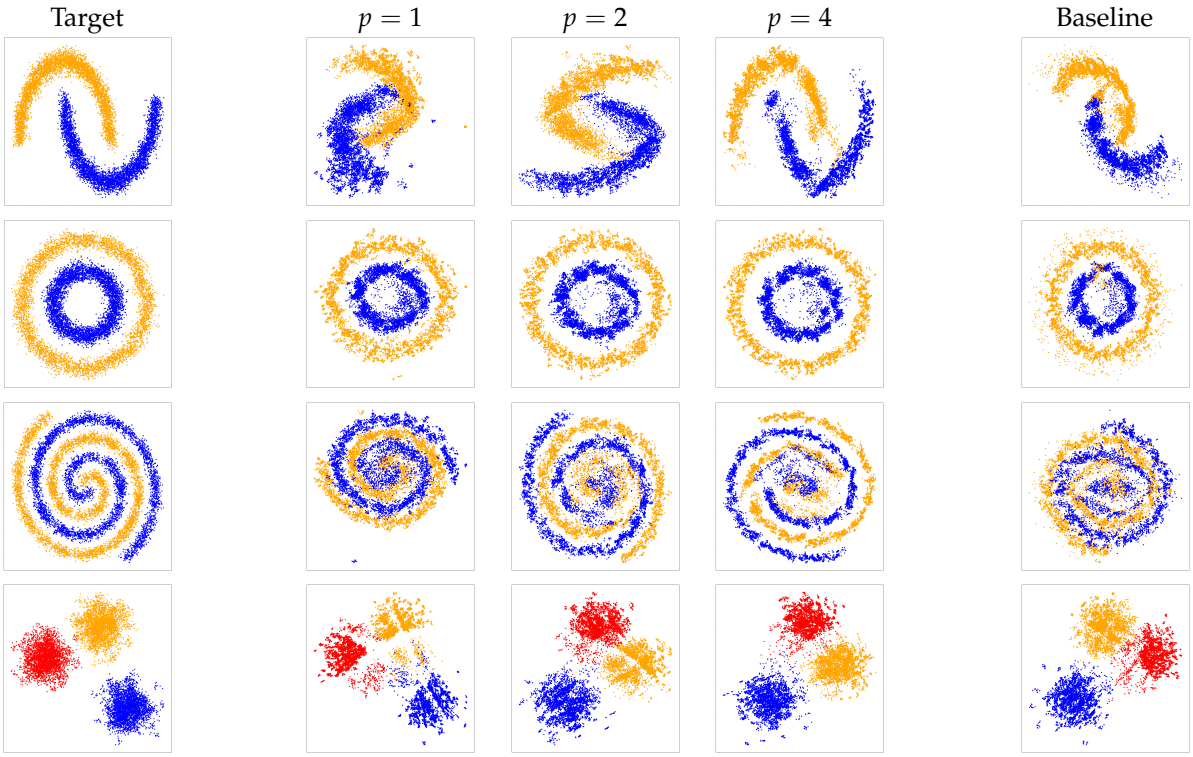
as illustrated in Figure 2a. For each of these datasets, we proceeded as follows:

- Fix n and k_i with $i = 1, \ldots, n$, respectively the number of sets of items, and the size of each set;
- Generate n points c_i that follow the pattern of the chosen dataset. These points are the centroids of the n sets;
- Given c_i, draw k_i random points from a normal distribution parameterized as $\mathcal{N}(c_i, \epsilon)$. We chose ϵ, the spread of the set, between 0 and 0.5;
- We divided the obtained n cloud of points in overlapping sets.

Following the approach described, we generated $|T|$ triplets sampled from a uniform distribution. In order to simulate the ordinal feedback from the oracle, we computed the difference of the squared l^2 norm between the centroids of the sets for a given triplet. The total number $|T|$ of sets was set to be $pnd \log n$ with $p = 1, 2, 4$.

Results. This series of experiments on synthetic datasets illustrates the performance for reconstruction and density estimation of our approach and, in particular, the influence of the number of triplets on the reconstruction ability. The first column of Figure 2 depicts the ground truth. Then, proceeding left to right, the embeddings werelearned for different values of T. The number of triplets increases with $T = pnd \log n$, where $p = \{1, 2, 4\}$.

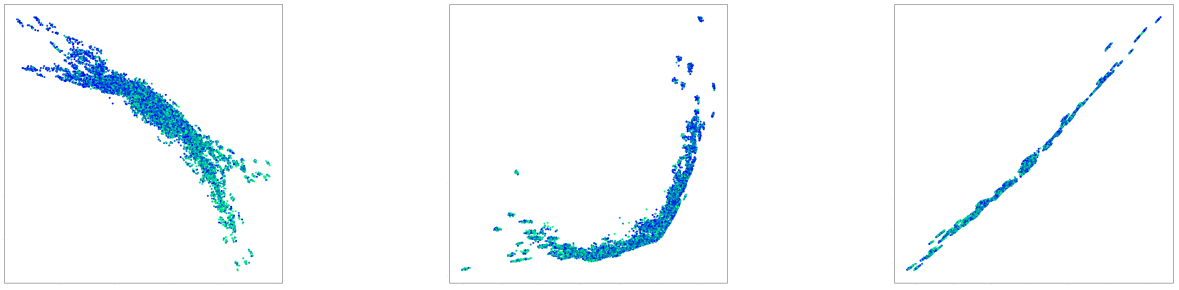
For all datasets, we observed that the quality of the reconstruction with respect to the location of the point vector improves when the number of triplets increases. We recall that an ordinal embedding is not unique, but the distances can be recovered up to an orthogonal transformation (translation, rotation, and reflection).



(**a**) Ground Truth (**b**) $p = 1, 2, 4$ (**c**) $p = 4$

Figure 2. SteOE embeddings for the synthetic experiments. The first column (**a**) shows the ground truth, (**b**) the progression of learned embeddings from increasing number for triplets $pnd \log n$ with $p = \{1, 2, 4\}$, and (**c**) the learned embedding from the baseline model with $p = 4$. The colors are merely used for better visibility of the different groups; they were not used for training.

The last column in Figure 3c is the visualization of the embeddings obtained by the baseline model, which is visibly less accurate than the proposed model. In order to obtain a quantitative, objective evaluation of the difference, Table 1 shows the Procrustes distance between the ground truth and the resulting embedding for both our proposed model and the baseline approach. We notice that our embeddings are consistently more precise, and this suggests that our distributional approach for set embedding leads to a better and more accurate representation.



(**a**) $k = 10$ (**b**) $k = 25$ (**c**) $k = 50$

Figure 3. Ordinal embeddings of MNIST digits. The supervision information is the sum of the digits in the sets. Colors represent the label of single digits, used only for visualization purposes.

Table 1. Procrustes distance from the ground truth and learned embeddings. Smaller (bold numbers) is better.

	Ours	Baseline
Circles	**0.12**	0.18
Moons	**0.09**	0.33
Spirals	**0.25**	0.91
Blobs	**0.03**	0.04

4.2. Sum of MNIST Digits

Data. Next, we applied our approach to more complex distributions than the synthetic datasets previously illustrated. We adapted the MNIST dataset for this task. MNIST contains 60,000 instances of 28×28 grey-scale stamps of digits in the range $0, \ldots, 9$. We randomly sampled N = 100,000 for training and 1000 for testing with a maximum size of k = 10, 25, 50 images. The supervision information provided by the oracle (10) is based on the difference of the sum of the digits in a set, i.e.,

$$\delta(I, J) = \left| \sum_{i \in I} \lambda_i - \sum_{j \in J} \lambda_j \right| \tag{14}$$

where λ_i is the label of image i, i.e., the one-digit number displayed by it. Thus, we cannot directly observe the label of the image, but we can only observe whether it tends to appear in sets with larger or smaller sums. The desired outcome is that the embedding of the individual images is able to capture the hidden label information.

Results. Figure 3 illustrates the obtained results for three different maximal set sizes. Each point represents a single image of the MNIST dataset. The color indicates the label of each single digit. We can clearly recognize the linear order in the learned embeddings. Low digits are separated from high digits, and the gradient is clearly noticeable.

Moreover, the smaller the sets used for learning, the more the order is distinguishable. In fact, although it is clear that the model was able to capture the linear order from the feedback of each triplet comparison, the results of Figure 3c are not as clear as those of Figure 3a. This can be expected, because in larger sets, the contribution of each individual number of the sum is lower than in smaller sets. This is an important factor from which we conclude that there exists a trade-off between the reconstruction ability and precision of the obtained representations. When density estimation is the priority, a bigger set size is advantageous because it necessitates fewer comparisons. However, when the focus is on the preciseness of the location of individual items in the space, a smaller set size should be preferred. Finally, we train a simple classifier on the learned embeddings to whose objective is to predict the label of the single MNIST digits. The results are reported in Table 2 in terms of mean accuracy. We can notice that our embeddings perform better than the ones learned through the baseline model, which proves the validity of our approach.

Table 2. Mean accuracy obtained for the classification of single MNIST digit embeddings. Best results are in bold.

	Baseline	Ours
$k = 10$	0.76	**0.78**
$k = 25$	0.76	**0.82**
$k = 50$	0.77	**0.79**

4.3. Poker Hands

Data. Poker is one of the best-known card games. The players bet whether the value of the hand they hold will beat all others according to a predefined ranking of hands. The complexity of the ranking system, where each card can be a part of a winning hand

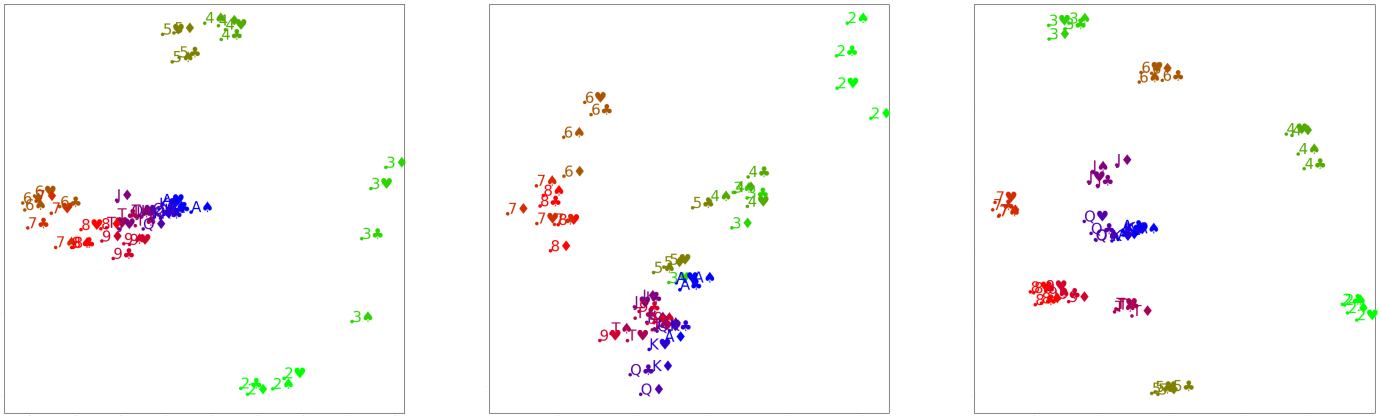
depending on the other cards in the hand, provides an interesting use case for assessing the embedding abilities of our approach. Variants largely differ on how cards are dealt and the methods by which players can improve a hand. In our experiments, we modeled a setting that was motivated by the Texas hold'em variant. We assumed two players J and K, each holding p cards, and a set of c community cards I.

The supervision information we used was based on which of the two players could obtain the best hand of five cards by combining his/her own cards with the community cards. The hand strength computations were based on Cactus Kev's algorithm.

Results. Figure 4 illustrates the results. In all cases, the number of triplets is $2nd \log n$, with $d = 2$ and $n = 52$. The colors show the rank of the cards, but this information was not available during the training. We illustrate three different variants:

Figure 4a shows the results of the setting with five community cards and each of the two players having two cards. This corresponds to the Texas Hold'em game.

The remaining figures show variants with differing numbers of board and community cards, which do not correspond to actual game settings, but which we studied to gain more insight into the obtained embeddings.



(**a**) $p = 2, c = 5$ (**b**) $p = 5, c = 2$ (**c**) $p = 1, c = 4$

Figure 4. Ordinal embeddings from poker for different numbers of player cards p and community cards c.

The results for the classical variant (Figure 4a) show that even if the supervision comes from a highly nonlinear source, our approach is still able to learn ordinal embeddings and output latent representations for the game that are fair and interpretable. First, we notice that unlike in the previous experiment, where a clear linear order of the embedded MNIST images was obtained, the structure obtained here is more complex. Nevertheless, we see that cards with similar values tend to form clusters because they go well together, forming pairs, triples, or even pokers, which have a high evaluation in the game. We can also see that clusters of cards with a low rank are pushed afar, whereas cards with higher ranks tend to be closer to each other. This, again, makes sense from the perspective of the game, because two high cards (regardless of whether they match or not) are a good combination. In particular, aces, being the cards with the highest rank in the deck, remain in the center of the plot. This finding is in accord with the rules of the game, since the probability of having a good hand is higher if it includes aces. It is possible to notice how the clusters are arranged in a spiral-like shape with the aces being in the center and moving farther away, and we can find the other values in decreasing order.

We then chose to evaluate the opposite setting of the one just described, in which the board has less cards than each player hands, more specifically two cards for the board and five for each player. This corresponds to Figure 4b. Even though this setting does not correspond to any game configuration, we assumed that investigating it could be of interest. Once again, we reached the same conclusions: cards with similar ranks are close to each other, also arranged in a spiral-like shape emanating from the center of cluster of aces.

However, contrary to the canonical setting, the cluster of aces is further from the middle of the point, and this is probably due to the higher combinatorial nature of the evaluation that perturbed the original order.

In the next setting, we tried to remove the combinatorial factor of the excess community cards. For this, we used the configuration in which each player had one card (a singleton) and the board had four cards. The results, shown in Figure 4c, exhibit the best separation between the different clusters of ranks and the spiral shape of their arrangement. Low ranks are far apart from high-ranked cards, and the higher ones are closer to the center, with the cluster of aces being the most centered one.

Overall, the found embeddings appear to be quite reasonable in all three cases. They seem to capture the expected property that cards that go well together in a poker hand have a low pairwise distance, whereas pairs of hands that do not go well together are further apart. For that reason, the low cards are rather far from all other cards except for their own kind, whereas the higher cards tend to be closer to each other. In order to test this, we computed a correlation coefficient between the distance of a pair of cards and the pairs' Chen score (the Chen score is a formula proposed by Bill Chen for assessing the strength of a pair of starting cards in Texas hold'em [20]) for all pairs of cards. As expected, the results, in Table 3, show a reasonably high positive correlation.

Table 3. Pearson correlation coefficient between l_2 of the different embedding vectors and the Chen score for all possible pairs of poker cards.

Figure	Pearson Coeff.
Figure 4a	0.58
Figure 4b	0.68
Figure 4b	0.65

4.4. Reuters

Data. For this experiment, we used data from the Reuters-21578 benchmark corpus [21]. This dataset contains n =10,788 Reuters Newswire articles. Each article is represented as a set of paragraphs with size $k \in \{2, 15\}$, with 60,222 paragraphs in total. The goal was to cluster these documents according to the categories of the newspaper. The documents belong to 90 different categories. The supervision used was the same as the one used in the MNIST sums of digits experiments.

Results. The results are shown in Figure 5. We embedded each document as a set of paragraphs in a two-dimensional space. As we can notice from the colors that indicate the label of the documents, the clusters are clearly distinguishable. Moreover, we conducted a quantitative evaluation on the obtained embeddings. For this, we computed the centroid of each set from the learned feature vector. Then, we trained an MLP classifier to predict the label associated with the n documents. We compared the mean accuracy of our approach to the baseline. We obtained an improvement of 5% in the mean accuracy, which proves that our obtained embedding are better suited for downstream tasks.

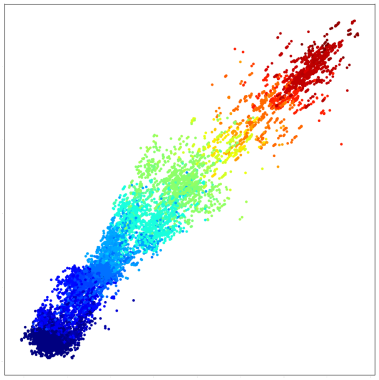


Figure 5. Ordinal embeddings for the Reuters dataset.

5. Related Work

In this section, we briefly summarize work that is related to our approach, both in the realm of ordinal embeddings, as well as in finding set representations.

5.1. Ordinal Embeddings

In recent years, ordinal data have received a growing interest in machine learning. The ordinal embedding problem has been studied from different points of view, for example: the question of finding the minimum number of triplets necessary to determine an ordinal embedding in the Euclidean space was tackled in [9] and further extended and generalized in [22]. Multiple methods have been designed to deal with triplet similarity. They typically produce representations of data points as low-dimensional Euclidean vectors. In particular, Generalized Nonmetric Multidimensional Scaling (GNMDS) [3] relies on a max-margin approach to minimize a hinge loss. Stochastic Triplet Embedding (STE) [6], on the other hand, assumes a Gaussian noise model and minimizes a logistic noise. The crowd kernel model [5] makes the assumption that triplets have been generated by an explicit noise model. It is worth mentioning that these models adopt a classification scheme to solve the problem by predicting the label of the relative comparisons. Another notable work is [7], which solved the ordinal embedding problem via a reduction to the problem of embedding nearest-neighbor graphs. Moreover, these methods rely on expensive gradient projections and are unsuitable for large datasets. The main purpose of those methods is to facilitate data visualization of similarity inferred from human assessments. However, other tasks employing similarity triplets have been studied, such as medoid estimation [23], density estimation [24], or clustering [25]. Closely related to our approach is [19], which employed deep learning to scale the ordinal problem to large datasets.

5.2. Sets' Representation

Machine learning on sets includes different subgroups depending on the nature of the input and output (e.g., vector-to-set, set-to-set, set-to-sequence). To the best of our knowledge, we are the first to propose an approach for set-valued input that does not rely on features. However, there are different works in the literature that are related to ours, more specifically in the set-to-set domain, where both the input and output are structured as sets. Notable examples are [26], which offered a permutation-invariant function for inference over sets by relying on the summation of all element representations prior to further nonlinear transformations. Other less recent works in the set-to-set domain are [27–29]. A more complete comparison of set encoders can be found in the next section. It is important to clearly differentiate our work, which falls into the set-to-set mappings, from some related works on vector-to-set mappings [30,31]. In fact, our work relates more to [26]. The main difference is that the input to our model is necessarily a set of items, albeit without features. Notable works in the vector-to-set literature are suited to tasks such as object detection, taking as input a feature representation of images and producing a set of coordinates for the bounding boxes. Loosely related are also methods such as in [32–34]

that learn a permutation matrix for sets of items. Once the permutation matrix is learned, it is applied to the input set, hence turning the set into an (ordered) sequence. Once again, our method differs because, there, the output set is not ordered, hence not a sequence.

5.3. Sets Encoder Models

The goal of a set encoder is to encode an input set into an embedding vector as an output. Several studies have proposed different approaches for performing this task. In this section, we list the most relevant to our work and provide a more thorough comparison in order to deepen the comparison from the previous section. As stated earlier, in this work, we propose a deep neural architecture for encoding sets without features. Moreover, our approach belongs to the set-to-set category. However, since the ordinal feedback available is only at the set level, the optimization is performed on the embedding of the set rather than the encoding of the single items that compose the set. The most natural comparison to our work is *Deep Sets* [26], which provides a robust mathematical analysis for designing permutation-invariant and permutation-equivariant deep learning models. This framework offers a simplified procedure by relying on the summation of all elements' representation for obtaining the set feature vector, which is consequently transformed into the desired output (e.g., the class probability for classification or a single number for set regression). Our approach is a generalization of the Deep Sets framework in which we do not require the input feature of the elements of the sets. Additional main differences from our work are that our set representation is a probabilistic measure rather than a point vector and the permutation-equivariant function is the mean rather than the sum.

Other works have tackled the task of finding robust set encoders. They largely differ from our proposition, but we discuss them for the sake of completion. The *Pointer Network* [28] is an encoder-decoder architecture that provides a modified attention mechanism, and its main goal is to learn the target reordering of the input elements. An important characteristic of Pointer Networks is that they do not treat set-valued input in the strict sense. In fact, the input is treated through sequential recurrent neural networks; hence, the obtained representation is not permutation-equivariant. The primary applications of Pointer Networks are tasks where the target output is a reordering or permutation of the elements of the initial input. This reordering is based on pointers to indices of the original input sequence.

Another important set encoder architecture is represented by the *Read-Process-and-Write-Model*. This is a neural network architecture made of different blocks, which aims to obtain a permutation-invariant representation of the input set and learn a mapping to arbitrary target outputs. It relies on an attention mechanism to satisfy the property of permutation-invariance and can be seen as a special case of *Memory Networks* [35]. In fact, it is a recurrent neural network model that creates a memory representation of each element in the input sequence and accesses the representation via the attention mechanism.

Among the most complex methods designed to handle set input problems, there is the *Set Transformer* [36]. The Set Transformer consists of stacked multi-head self-attention layers for both the internal encoder and decoder, as seen in the classic Transformer [37]. One main difference from the previously described set encoding methods is that instead of using a fixed pooling operation such as sum() or average() to ensure permutation-invariance. It employs a parameterized pooling function that is learned and therefore results in being much more adaptive to the particular task at hand. The Set Transformer [36] is designed to model higher-order interactions among elements and their subsets within the input set. Its key advantage is that it concurrently encodes the entire input set through a sequence of permutation-equivariant *Set Attention Blocks* (SABs). By comparison, the previously discussed Deep Sets and our proposed approach method obtained element features independently of other input set elements. The main limitation of the Set Transformer is its computational cost. In fact, the SABs require quadratic complexity $\mathcal{O}(n^2)$ with n being the cardinality of the input set. A lower projection was proposed by the authors to try to

overcome this limit bringing the overall complexity to $\mathcal{O}(mn)$ with m being the chosen number of inducing points for the low-rank projection.

Reference [30] proposed a permutation-invariant approach that derives from the the naive method of sorting all the elements of the input set by a chosen feature. However, when the output is a set, this approach leads to discontinuities, which the authors described as the *responsibility problem*. In a nutshell, these discontinuities arise whenever two elements are swapped in the input and the output. To avoid this difficulty, the authors developed a novel pooling method, which sorts each feature across the elements of the input set and then performs a weighted sum. This allows the model to remember the permutation applied through the feature-wise sorting and apply its inverse in the decoder. This process restores the original, arbitrary order of the input elements making the encoding a permutation-equivariant operation, preventing the discontinuity in the outputs of the model.

Another interesting approach to encode sets based on the reordering of the input is the Janossy pooling approach by [38]: the symmetric (permutation-invariant) encoding function is expressed as the average of a mixture of permutation-sensitive functions applied to all re-orderings of the original input. Generating all permutations of a set results in $n!$ intermediate inputs, all of which would then require the application of the chosen permutation-sensitive function. However, this approach is not tractable. To mitigate this, the authors proposed a number of strategies, among them the use of a smaller number of selected canonical orderings that are presumed to carry relevant information.

PointNet by [27] is a neural architecture for encoding 3D point clouds. An additional constraint for this architecture is that the output should be independent of the translation or rotation of the input point cloud. Loosely speaking, PointNet first obtains an embedding of each of the input points through stacked, fully connected layers in the form of an MLP, such that each element is identically and independently transformed. This permutation-equivariant representation is then pooled via the max() operator (per dimension) and further transformed through an additional fully connected layer. Finally, the obtained point cloud encoding is concatenated with the embedding of each point. This combination of local and global features is shown to be crucial for point segmentation tasks.

The *AttSets* model, proposed by [39], uses weighted attention to obtain a permutation-invariant representation of the input set. This model was originally meant to be applied to a multi-view 3D reconstruction task, where a set of images of the same object from different angles is used to estimate its true 3D shape. In order to achieve this, each element of the set is individually and independently transformed via a learned attention function, which can take the form of an MLP or a multidimensional CNN, according to the form of the input. The output of this function is normalized via softmax() and then used as an attention mask over the original input elements. This allows the model to learn to pay a varying degree of attention to individual dimensions of the input elements' representations. Finally, the original input elements are multiplied by the attention mask and summed together to a fixed length set encoding.

Finally, the *RepSet* [40] model consists of stacked feed-forward, fully connected layers, as in the Deep Sets method [28], followed by a custom permutation invariant layer replacing the sum() operator. This layer was inspired by the concepts from the field of bipartite graph matching. The permutation-invariance is achieved through a configurable number of hidden sets (potentially of different sizes), whose elements correspond to columns of trainable weight matrices. These are then compared with the elements of the actual input set to create matrices that are fed the Hungarian algorithm. The resulting values can be further transformed through standard neural network layers according to the problem at hand. One limitation of this approach is the computational complexity of $\mathcal{O}(mn + n^2 \log n)$, where n is the cardinality of the input set and m is the chosen number of hidden sets.

6. Conclusions

We proposed an approach to solve the ordinal embedding problem when the input is under the form of sets of items and the feedback is available only for triplets of sets.

Our approach maps the objects in a low-dimensional space endowed with the Wasserstein distance. This is based on learning a representation for sets and taking advantage of the common statistic for sets, which is the centroid. Each set is described by a location parameter, its centroid, and a scale parameter, which represents the spread of the set. We argue that reformulating the problem under this point of view allows prompting fewer triplets comparisons for a greater number of learned items. Our algorithm is suitable when the input sets have variable size. Moreover, a trade-off between the precision of the individual embeddings and the accuracy of the overall density estimation has to be taken into account when choosing the size of the input sets. In a number of experiments on different datasets, we demonstrated the validity of our approach. We showed that the proposed framework is robust and beneficial when the triplet comparisons are limited. Overall, with our proposed approach, we were able to obtain valid embeddings that can be used for downstream tasks. In conclusion, we think that our main idea should be readily extensible to a similar domain, even including features, such as set-to-sequence, set-to-graphs, or set-to-set. Future directions of improvement involve extending the model to consider pairwise or more complex interactions among the elements of a given set. In fact, so far, the encoding step focuses on one element at time. Moreover, an additional way of improvement might be to relax the link between the cardinality of the input set and the precision of the output embedding. Finally, a next improvement we aim to make would be to improve the aggregation function by using a parameterized and learned function, which could improve the performance according to the task at hand.

Author Contributions: Conceptualization, A.D. and J.F.; Formal analysis, A.D. and J.F.; Investigation, A.D.; Software, A.D.; Supervision, J.F.; Validation, J.F.; Writing—original draft preparation, A.D.; Writing—review and editing, J.F. and A.D.; Funding acquisition, J.F. Both authors have read and agreed to the published version of the manuscript.

Funding: This work has been supported by the German Research Foundation as part of the Research Training Group Adaptive Preparation of Information from Heterogeneous Sources (AIPHES) under grant No. GRK 1994/1.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Shepard, R.N. The analysis of proximities: Multidimensional scaling with an unknown distance function. I. *Psychometrika* **1962**, *27*, 125–140. [CrossRef]
2. Shepard, R.N. Metric structures in ordinal data. *J. Math. Psychol.* **1966**, *3*, 287–315. [CrossRef]
3. Agarwal, S.; Wills, J.; Cayton, L.; Lanckriet, G.; Kriegman, D.; Belongie, S. Generalized non-metric multidimensional scaling. In Proceedings of the 11th International Conference on Artificial Intelligence and Statistics (AISTATS), San Juan, PR, USA, 21–24 March 2007; pp. 11–18.
4. McFee, B. More Like This: Machine Learning Approaches to Music Similarity. Ph.D. Thesis, University of California, San Diego, CA, USA, 2012.
5. Tamuz, O.; Liu, C.; Belongie, S.; Shamir, O.; Kalai, A. Adaptively Learning the Crowd Kernel. In Proceedings of the 28th International Conference on Machine Learning (ICML), Bellevue, WA, USA, 28 June–2 July 2011; pp. 673–680.
6. Van Der Maaten, L.; Weinberger, K. Stochastic triplet embedding. In Proceedings of the IEEE International Workshop on Machine Learning for Signal Processing, Santander, Spain, 23–26 September 2012; pp. 1–6.
7. Terada, Y.; von Luxburg, U. Local ordinal embedding. In Proceedings of the 31st International Conference on Machine Learning (ICML), Beijing, China, 21–26 June 2014; pp. 847–855.
8. Kleindessner, M.; von Luxburg, U. Kernel functions based on triplet comparisons. In Proceedings of the Advances in Neural Information Processing Systems 30, Long Beach, CA, USA, 4–9 December 2017.

9. Jamieson, K, G.; Nowak, R, D. Low-dimensional embedding using adaptively selected ordinal data. In Proceedings of the 49th Annual Allerton Conference on Communication, Control, and Computing, Monticello, IL, USA, 28–30 August 2011.
10. Diallo, A.; Fürnkranz, J. Elliptical Ordinal Embedding. *arXiv* **2021**, arXiv:2105.10457.
11. Joachims, T.; Granka, L.; Pan, B.; Hembrooke, H.; Gay, G. Accurately interpreting clickthrough data as implicit feedback. *ACM SIGIR Forum* **2017**, *51*, 4–11. [CrossRef]
12. Muzellec, B.; Cuturi, M. Generalizing point embeddings using the Wasserstein space of elliptical distributions. In Proceedings of the Advances in Neural Information Processing System 31, Montréal, QC, Canada, 3–8 December 2018; pp. 10258–10269.
13. Olkin, I.; Pukelsheim, F. The distance between two random vectors with given dispersion matrices. *Linear Algebra Appl.* **1982**, *48*, 257–263. [CrossRef]
14. Frogner, C.; Zhang, C.; Mobahi, H.; Araya, M.; Poggio, T.A. Learning with a Wasserstein loss. In Proceedings of the Advances in Neural Information Processing Systems 28, Montréal, QB, Canada, 2–5 December 2015.
15. Dittmann, J. Explicit formulae for the Bures metric. *J. Phys. Math. Gen.* **1999**, *32*, 2663–2670. [CrossRef]
16. Beran, R. Minimum Hellinger distance estimates for parametric models. *Ann. Stat.* **1977**, *5*, 445–463. [CrossRef]
17. LeCun, Y.; Chopra, S.; Hadsell, R.; Ranzato, M.; Huang, F.J. A Tutorial on Energy-Based Learning. In *"Predicting Structured Data*; MIT Press: Cambridge, MA, USA, 2006; Volume 1.
18. Hoffer, E.; Ailon, N. Deep metric learning using triplet network. In Proceedings of the 3rd International Conference on Similarity-Based Pattern Recognition (ICLR Workshop Track), San Diego, CA, USA, 7–9 May 2015; pp. 84–92.
19. Haghiri, S.; Vankadara, L.C.; von Luxburg, U. Large scale representation learning from triplet comparisons. *arXiv* **2019**, arXiv:1912.01666.
20. Chen, B.; Ankenman, J. *The Mathematics of Poker*; ConJelCo LLC., Jerrod: Pittsburgh, PA, USA, 2006.
21. Xue, N.; Bird, E.; Natural language processing with python. *Nat. Lang. Eng.* **2011**, *17*, 419. [CrossRef]
22. Jain, L.; Jamieson, K.G.; Nowak, R. Finite Sample Prediction and Recovery Bounds for Ordinal Embedding. In Proceedings of the Advances in Neural Information Processing Systems 29, Barcelona, Spain, 5–10 December 2016; pp. 2711–2719.
23. Heikinheimo, H.; Ukkonen, A. The crowd-median algorithm. In Proceedings of the 1st AAAI Conference on Human Computation and Crowdsourcing (HCOMP), Palm Springs, CA, USA, 7–9 November 2013.
24. Ukkonen, A.; Derakhshan, B.; Heikinheimo, H. Crowdsourced nonparametric density estimation using relative distances. In Proceedings of the 3rd AAAI Conference on Human Computation and (HCOMP), San Diego, CA, USA, 8–11 November 2015.
25. Ukkonen, A. Crowdsourced correlation clustering with relative distance comparisons. In Proceedings of the IEEE International Conference on Data Mining (ICDM), New Orleans, LA, USA, 18–21 November 2017; pp. 1117–1122.
26. Zaheer, M.; Kottur, S.; Ravanbakhsh, S.; Poczos, B.; Salakhutdinov, R.R.; Smola, A.J. Deep Sets. In *Advances in Neural Information Processing Systems 30*; Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2017.
27. Qi, C.R.; Su, H.; Mo, K.; 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, Honolulu, HI, USA, 21–26 July 2017; pp. 652–660.
28. Vinyals, O.; Fortunato, M.; Jaitly, N. Pointer Networks. In *Advances in Neural Information Processing Systems 28*; Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2015.
29. Vinyals, O.; Bengio, S.; Kudlur, M. Order Matters: Sequence to sequence for sets. In Proceedings of the 4th International Conference on Learning Representations (ICLR), San Juan, PR, USA, 2–4 May 2016.
30. Zhang, Y.; Hare, J.; Prügel-Bennett, A. FSPool: Learning Set Representations with Featurewise Sort Pooling. In Proceedings of the International Conference on Learning Representations (ICLR), Addis Ababa, Ethiopia, 26 April–1 May 2020.
31. Zhang, Y.; Hare, J.; Prügel-Bennett, A. Deep Set Prediction Networks. In Proceedings of the Advances in Neural Information Processing Systems 32, Vancouver, BC, Canada, 8–14 December 2019.
32. Zhang, Y.; Hare, J.; Prügel-Bennett, A. Learning Representations of Sets through Optimized Permutations. In Proceedings of the International Conference on Learning Representations (ICLR), New Orleans, LA, USA, 6–9 May 2019.
33. Diallo, A.; Zopf, M.; Fürnkranz, J. Permutation Learning via Lehmer Codes. In Proceedings of the 24th European Conference on Artificial Intelligence (ECAI), Santiago de Compostela, Spain, 31 August–2 September 2020.
34. Mena, G.; Belanger, D.; Linderman, S.; Snoek, J. Learning Latent Permutations with Gumbel-Sinkhorn Networks. In Proceedings of the International Conference on Learning Representations (ICLR), Vancouver, BC, Canada, 30 April–3 May 2018.
35. Weston, J.; Chopra, S.; Bordes, A. Memory networks. *arXiv* **2014**, arXiv:1410.3916.
36. Lee, J.; Lee, Y.; Kim, J.; Kosiorek, A.; Choi, S.; Teh, Y.W. Set transformer: A framework for attention-based permutation-invariant neural networks. In Proceedings of the International Conference on Machine Learning, PMLR, Long Beach, CA, USA, 10–15 June 2019; pp. 3744–3753.
37. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
38. Murphy, R.L.; Srinivasan, B.; Rao, V.; Ribeiro, B. Janossy Pooling: Learning Deep Permutation-Invariant Functions for Variable-Size Inputs. In Proceedings of the International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May 2019.

39. Yang, B.; Wang, S.; Markham, A.; Trigoni, N. Robust attentional aggregation of deep feature sets for multi-view 3D reconstruction. *Int. J. Comput. Vis.* **2020**, *128*, 53–73. [CrossRef]
40. Skianis, K.; Nikolentzos, G.; Limnios, S.; Vazirgiannis, M. Rep the set: Neural networks for learning set representations. In Proceedings of the International Conference on Artificial Intelligence and Statistics. PMLR, Palermo, Italy, 26–28 August 2020; pp. 1410–1420.