Set2Box: Similarity Preserving Representation Learning for Sets

Geon Lee
KAIST AI
geonlee0325@kaist.ac.kr

Chanyoung Park
KAIST ISysE & AI
cy.park@kaist.ac.kr

Kijung Shin
KAIST AI & EE
kijungs@kaist.ac.kr

***Abstract*—Sets have been used for modeling various types of objects, and measuring similarity between them has been a key building block of a wide range of applications. However, as sets have grown in numbers and sizes, the computational cost and storage required for set similarity computation have become substantial. In this work, we propose SET2BOX, which represents sets as boxes to precisely capture overlaps of sets and thus accurately estimate various similarity measures. Additionally, based on the proposed box quantization scheme, we design SET2BOX$^+$, which yields more concise but more accurate box representations of sets. Through extensive experiments on 8 real-world datasets, we show that, compared to baseline approaches, SET2BOX$^+$ is *(a) Accurate:* achieving up to *40.8× smaller estimation error* while requiring 60% fewer bits to encode sets, *(b) Concise:* yielding up to *96.8× more concise* representations with similar estimation error, and *(c) Versatile:* enabling the estimation of four set-similarity measures from a single representation of each set. For reproducibility, the source code and datasets used in the paper are available at https://github.com/geon0325/Set2Box.**

I. INTRODUCTION

Sets are ubiquitous, modeling various types of objects in many domains, including texts, purchase records, social circles, and online discussions. Moreover, a number of set similarity measures (e.g., Jaccard Index), most of which are based on the overlaps between sets, have been developed.

As a result of the omnipresence of sets, measuring their similarity has been employed as a fundamental building block of a wide range of applications, including **(a) plagiarism detection:** a document is modeled as a "bag of words," and documents whose set representations are highly similar are suspected of plagiarism [1], **(b) gene expression mining:** the functionality of a set of genes is estimated by comparing the set with other sets with known functionality [2], **(c) recommendation:** users who purchased similar sets of items are identified for collaborative filtering [3], and more (e.g., graph compression [4] and medical image analysis [5]).

As sets grow in numbers and sizes, computation of set similarity requires substantial computational cost and storage. For example, similarities between tens of thousands of movies, which are represented as sets of up to hundreds of thousands of users who have rated them, were measured for movie recommendation [3]. In order to reduce the space and computation required for set-similarity computation, a number of approaches based on hashing and sketching [6], [7] have been developed. While their simplicity and theoretical guarantees are tempting, significant gains are expected if patterns in a given collection of sets can be learned and exploited.

In this paper, we propose SET2BOX, a learning-based approach for compressed representations of sets from which various similarity measures can be estimated accurately in constant time. The key idea of SET2BOX is to represent sets as boxes, which share primary characteristics of sets, to capture the overlaps between sets and thus their similarity based on them. In addition, we propose SET2BOX$^+$, which yields even more concise but more accurate boxes based on the proposed box quantization scheme. Our contributions are as follows:

- **Accurate & Versatile Algorithm:** We propose SET2BOX, a set representation learning method that accurately preserves similarity between sets in terms of four measures.
- **Concise & Effective Algorithm:** Based on an end-to-end box quantization scheme, we devise SET2BOX$^+$ which yields more accurate and concise similarity estimation.
- **Extensive Experiments:** Using 8 real-world datasets, we validate the advantages of SET2BOX$^+$ over its competitors and the effectiveness of each of its components.

In Section II, we review related work. In Section III, we provide preliminaries. In Section IV, we present SET2BOX and SET2BOX$^+$. In Section V, we provide experimental results. Lastly, we offer conclusions in Section VI.

II. RELATED WORK

Similarity-Preserving Embedding: Representation learning for preserving similarities between instances has been studied for graphs [8], [9], images [10], and texts [11]. These methods aim to yield high-quality embeddings by minimizing the information loss of the original data. However, most of them are designed to preserve the predetermined similarity matrix, which are not extensible to new measures [9].

Box Embedding: Thanks to the powerful expressiveness of boxes [12], they have been used in diverse applications including knowledge bases [13]–[15], word embedding [16], image embedding [17], and recommender systems [18], [19]. In an algorithmic aspect, methods for improving the optimization of learning boxes (e.g., Gaussian convolutions [20] and Gumbel random variables [21]) have been presented.

Differentiable Product Quantization: Product quantization [22], [23] is an effective strategy for vector compression, and recently, deep learning methods for learning discrete codes in an end-to-end manner have been proposed [24], [25].

III. PRELIMINARIES

In this section, we introduce notations and define the problem. Then, we review some intuitive methods for the problem.

Notations: Consider a set $\mathcal{S} = \{s_1, \cdots, s_{|\mathcal{S}|}\}$ of *sets* and a set $\mathcal{E} = \{e_1, \cdots, e_{|\mathcal{E}|}\}$ of *entities*. Each set $s \in \mathcal{S}$ is a non-empty subset of $\mathcal{E}$ and its size (i.e., cardinality) is denoted by $|s|$. A representation of the set s is denoted by z_s and its encoding cost (the number of bits to encode z_s) in bits is denoted by $Cost(\mathrm{z}_s)$. Refer to Table I for frequently-used notations.

Problem Definition: The problem of learning similarity-preserving set representations is formulated as:

Problem 1 (Similarity-Preserving Set Embedding)**.**
1) **Given:** *(1) a set $\mathcal{S}$ of sets and (2) a budget b*
2) **Find:** *a latent representation z_s of each set $s \in \mathcal{S}$*
3) **to Minimize:** *the difference between (1) the similarity between s and s', and (2) the similarity between z_s and $\mathrm{z}_{s'}$ for all $s \neq s' \in \mathcal{S}$*
4) **Subject to:** *the total encoding cost $Cost(\{\mathrm{z}_s : s \in \mathcal{S}\}) \leq b$.*

In this paper, we consider four set-similarity measures and use the mean squared error (MSE)[1] to measure the differences, while our proposed methods are not specialized to the choices.

Desirable Properties: We expect set embeddings for Problem 1 to be: **(a) accurate:** similarities approximated using learned representations are close to those between sets, **(b) concise:** less amount of memory is desirably used to store embeddings while keeping them informative, **(c) generalizable:** embeddings of unseen sets can be obtained, **(d) versatile:** representations can be used to approximated diverse similarity measures, and **(e) fast:** set similarities are rapidly estimated.

Intuitive Methods: We discuss simple and intuitive set-embedding methods for similarity preservation.

- **Random Hashing [6]:** Each set s is encoded as a binary vector z_s by mapping each entity into one of the d different values using a hash function $h(\cdot) : \mathcal{E} \to \{1, \cdots, d\}$. Specifically, the representation $\mathrm{z}_s \in \{0, 1\}^d$ is derived by $\mathrm{z}_s[i]$ is 1 if $\exists e \in s$ s.t. $h(e) = i$ and 0 otherwise. The size of the set s is estimated from the L1 norm of z_s, i.e., $|s| \approx \|\mathrm{z}_s\|_1$. In addition, sizes of the intersection and the union of sets s and s' are estimated from $|s \cap s'| \approx \|\mathrm{z}_s$ **AND** $\mathrm{z}_{s'}\|_1$ and $|s \cup s'| \approx \|\mathrm{z}_s$ **OR** $\mathrm{z}_{s'}\|_1$, respectively, where **AND** and **OR** are dimension-wise operations. Based on these approximations, any set similarities can be estimated.
- **Vector Embedding:** Another popular approach is to represent sets as vectors and compute the inner products between them to estimate a predefined set similarity. More precisely, given two sets s and s' and their vector representations z_s and $\mathrm{z}_{s'}$, it aims to approximate predefined $\mathrm{sim}(s, s')$ by the inner product of z_s and $\mathrm{z}_{s'}$, i.e., $\langle \mathrm{z}_s, \mathrm{z}_{s'} \rangle \approx \mathrm{sim}(s, s')$.

However, random hashing cannot accurately represent sets whose sizes are larger than d, and vector embedding shows weakness in its versatility, i.e., it can only preserve a predefined similarity measure. The proposed end-to-end methods, SET2BOX and SET2BOX+, effectively address these issues.

[1] $\sum_{s \neq s' \in \mathcal{S}} |\mathrm{sim}(s, s') - \widehat{\mathrm{sim}}(\mathrm{z}_s, \mathrm{z}_{s'})|^2$ $\mathrm{sim}(\cdot, \cdot)$ and $\widehat{\mathrm{sim}}(\cdot, \cdot)$ are similarity between sets and that between latent representations, respectively.

TABLE I: Frequently-used symbols.

Notation	Definition
$\mathcal{S} = \{s_1, ..., s_{\lvert\mathcal{S}\rvert}\}$ $\mathcal{E} = \{e_1, ..., e_{\lvert\mathcal{E}\rvert}\}$	set of sets set of entities
$\mathrm{B} = (\mathrm{c}, \mathrm{f})$ $\mathbb{V}(\mathrm{B})$	a box with center c and offset f volume of box B
$\mathcal{T}^+$ and $\mathcal{T}^-$	a set of positive & negative samples
$\mathbf{Q}^{\mathrm{c}} \in \mathbb{R}^{\lvert\mathcal{E}\rvert \times d}$ $\mathbf{Q}^{\mathrm{f}} \in \mathbb{R}_+^{\lvert\mathcal{E}\rvert \times d}$	center embedding matrix of entities offset embedding matrix of entities
D K	number of subspaces number of key boxes in each subspace

IV. PROPOSED METHOD

In this section, we first present SET2BOX, a novel algorithm for learning similarity-preserving set representations using boxes. Then we propose SET2BOX+, an advanced version of SET2BOX, which derives better conciseness and accuracy.

A. SET2BOX: *Preliminary Version*

We first present SET2BOX, a preliminary set representation method that effectively learns the set itself and the structural relations with other sets for similarity preserving.

Concepts: A *box* is a d-dimensional hyper-rectangle whose representation consists of its center and offset [12]. The center describes the location of the box in the latent space and the offset is the length of each edge of the box. Formally, given a box $\mathrm{B} = (\mathrm{c}, \mathrm{f})$ whose center $\mathrm{c} \in \mathbb{R}^d$ and offset $\mathrm{f} \in \mathbb{R}_+^d$ are in the same latent space, the box is defined as a bounded region:

$$\mathrm{B} \equiv \{\mathrm{p} \in \mathbb{R}^d : \mathrm{c} - \mathrm{f} \preceq \mathrm{p} \preceq \mathrm{c} + \mathrm{f}\}.$$

Let $\mathrm{m} \in \mathbb{R}^d$ and $\mathrm{M} \in \mathbb{R}^d$ be the vectors of the minimum and the maximum at each dimension, respectively, i.e., $\mathrm{m} = \mathrm{c} - \mathrm{f}$ and $\mathrm{M} = \mathrm{c} + \mathrm{f}$. The intersection of two boxes $\mathrm{B}_X = (\mathrm{c}_X, \mathrm{f}_X)$ and $\mathrm{B}_Y = (\mathrm{c}_Y, \mathrm{f}_Y)$ is also a box, represented as:

$$\mathrm{B}_X \cap \mathrm{B}_Y \equiv \{\mathrm{p} \in \mathbb{R}^d : \mathbf{max}(\mathrm{m}_X, \mathrm{m}_Y) \preceq \mathrm{p} \preceq \mathbf{min}(\mathrm{M}_X, \mathrm{M}_Y)\}.$$

The *volume* $\mathbb{V}(\mathrm{B})$ of the box B is computed by the product of the length of an edge in each dimension, i.e., $\mathbb{V}(\mathrm{B}) = \prod_{i=1}^{d} (\mathrm{M}[i] - \mathrm{m}[i])$. The volume of the union of the two boxes is simply computed by $\mathbb{V}(\mathrm{B}_X) + \mathbb{V}(\mathrm{B}_Y) - \mathbb{V}(\mathrm{B}_X \cap \mathrm{B}_Y)$.

Representation: The core idea of SET2BOX is to model each set s as a box $\mathrm{B}_s = (\mathrm{c}_s, \mathrm{f}_s)$ so that the relations with other sets are properly preserved in the latent space. To this end, SET2BOX approximates the volumes of the boxes to the relative sizes of the sets, i.e., $\mathbb{V}(\mathrm{B}_s) \propto |s|$. In addition, it aims to preserve the relations between different sets by approximating the volumes of the intersection of the boxes to the intersection sizes of the sets, i.e., $\mathbb{V}(\mathrm{B}_{s_i} \cap \mathrm{B}_{s_j}) \propto |s_i \cap s_j|$.

Objective: Recall that our goal is to derive *accurate* and *versatile* representations of sets, and towards the first goal, we take relations beyond pairwise into consideration. To this end, we design an objective function that aims to preserve elemental relations among triple of sets. Specifically, given a triple $\{s_i, s_j, s_k\}$ of sets, we consider seven cardinalities from different levels of subsets: **(1)** $|s_i|$, $|s_j|$, $|s_k|$, **(2)** $|s_i \cap s_j|$, $|s_j \cap s_k|$, $|s_k \cap s_i|$, and **(3)** $|s_i \cap s_j \cap s_k|$ which contain single,

pair, and triple-wise information, respectively, and we denote them from $c_1(s_i, s_j, s_k)$ to $c_7(s_i, s_j, s_k)$. These elements fully-describe the relations among the three sets, and any similarity measures are computable using them. In this regard, we aim to preserve the ratios of the seven cardinalities by the volumes of the boxes B_{s_i}, B_{s_j}, and B_{s_k} by minimizing the objective:

$$\mathcal{J}(s_i, s_j, s_k, \mathrm{B}_{s_i}, \mathrm{B}_{s_j}, \mathrm{B}_{s_k}) = \sum\nolimits_{\ell=1}^{7} \left(p_\ell(s_i, s_j, s_k) - \hat{p}_\ell(\mathrm{B}_{s_i}, \mathrm{B}_{s_j}, \mathrm{B}_{s_k})\right)^2,$$

where p_ℓ is the ratio of the ℓ^{th} cardinality among the three sets (i.e., $p_\ell = c_\ell / \sum_{\ell'} c_{\ell'}$) and $\hat{p}_\ell$ is the corresponding ratio estimated by the boxes. We sample a set $\mathcal{T}^+$ of *positive* triples (three overlapping sets) and a set $\mathcal{T}^-$ of *negative* triples (three uniform random sets) and, using $\mathcal{T} = \mathcal{T}^+ \cup \mathcal{T}^-$, minimize

$$\mathcal{L} = \sum\nolimits_{\{s_i, s_j, s_k\} \in \mathcal{T}} \mathcal{J}(s_i, s_j, s_k, \mathrm{B}_{s_i}, \mathrm{B}_{s_j}, \mathrm{B}_{s_k}). \tag{1}$$

Notably, the proposed objective function aims to capture not only the pairwise interactions between sets, but also the triple-wise relations to capture high-order overlapping patterns of the sets. In addition, it does not rely on any predefined similarity measure. It is a general objective for learning key structural patterns of sets and their neighbors, and thus it enables the model to yield accurate estimates to diverse metrics.

Box Embedding: Then, how can we derive the box B_s of a set s, i.e., its center c_s and offset f_s? To make the method generalizable to unseen sets, we introduce a pair of learnable matrices $\mathbf{Q}^{\mathrm{c}} \in \mathbb{R}^{|\mathcal{E}| \times d}$ and $\mathbf{Q}^{\mathrm{f}} \in \mathbb{R}_+^{|\mathcal{E}| \times d}$ of entities, where $\mathbf{Q}^{\mathrm{c}}_i \in \mathbb{R}^d$ and $\mathbf{Q}^{\mathrm{f}}_i \in \mathbb{R}^d_+$ represent the center and offset of an entity e_i, respectively.

Then, the embeddings of the entities in the set s are aggregated to obtain c_s and r_s.

Here, we use attentions to highlight the entities that are important to obtain the center or the offset of the box. To this end, we define a pooling function that takes the context of each set into account, termed set-context pooling (**SCP**). Specifically, given a set s and an item embedding matrix $\mathbf{Q}$ (which can be either $\mathbf{Q}^{\mathrm{c}}$ or $\mathbf{Q}^{\mathrm{f}}$), it first obtains the set-specific context vector b_s:

$$\mathrm{b}_s = \sum_{e_i \in s} \alpha_i \mathbf{Q}_i \quad \text{where} \quad \alpha_i = \frac{\exp(\mathrm{a}^\intercal \mathbf{Q}_i)}{\sum_{e_j \in s} \exp(\mathrm{a}^\intercal \mathbf{Q}_j)}$$

where a is a global context vector shared by all sets. Then using the context vector b_s, which specifically contains the information on set s, it obtains the output embedding from:

$$\textbf{SCP}(s, \mathbf{Q}) = \sum_{e_i \in s} \omega_i \mathbf{Q}_i \;\text{ where }\; \omega_i = \frac{\exp(\mathrm{b}_s^\intercal \mathbf{Q}_i)}{\sum_{e_j \in s} \exp(\mathrm{b}_s^\intercal \mathbf{Q}_j)}.$$

To be precise, $\mathrm{c}_s = \textbf{SCP}(s, \mathbf{Q}^{\mathrm{c}})$ and $\mathrm{f}_s = |s|^{\frac{1}{d}} \textbf{SCP}(s, \mathbf{Q}^{\mathrm{f}})$. For the offset f_s, we multiply an additional regularizer $|s|^{\frac{1}{d}}$ without which a natural condition for boxes (spec., $\min_{e_i \in s} \mathbf{Q}^{\mathrm{f}}_i \preceq \mathrm{f}_s \preceq \max_{e_i \in s} \mathbf{Q}^{\mathrm{f}}_i$) does not hold (see [26] for details).

Smoothing Boxes: By definition, a box $\mathrm{B} = (\mathrm{c}, \mathrm{f})$ is a bounded region with *hard edges* whose volume is $\mathbb{V}(\mathrm{B}) = \prod_{i=1}^d \mathrm{ReLU}(\mathrm{M}[i] - \mathrm{m}[i])$ where $\mathrm{m} = \mathrm{c} - \mathrm{f}$ and $\mathrm{M} = \mathrm{c} + \mathrm{f}$. This, however, disables gradient-based optimization when boxes are disjoint [20], and thus we *smooth* the boxes by $\mathbb{V}(\mathrm{B}) = \prod_{i=1}^d \mathrm{Softplus}(\mathrm{M}[i] - \mathrm{m}[i])$ where $\mathrm{Softplus}(\mathrm{x}) = \frac{1}{\beta}\log(1 + \exp(\beta \mathrm{x}))$ is an approximation to $\mathrm{ReLU}(\mathrm{x})$, and it becomes closer to ReLU as β increases. In this way, any pairs of boxes overlap each other, and thus non-zero gradients are computed for optimization.

Encoding Cost: Each box consists of two vectors, a center and an offset, and it requires $2 \cdot 32d = 64d$ bits to encode them. [2] Thus, $64|\mathcal{S}|d$ bits are required for $|S|$ sets.

B. SET2BOX$^+$*: Advanced Version*

We describe SET2BOX$^+$, which enhances SET2BOX in terms of conciseness and accuracy, based on an end-to-end *box quantization* scheme. Specifically, SET2BOX$^+$ compresses the the box embeddings into a compact set of key boxes and a set of discrete codes to reconstruct the original boxes.

Box Quantization: We propose box quantization, a novel scheme for compressing boxes by using substantially smaller number of bits. Note that conventional product quantization methods [24], which are for vector compression, are straightforwardly applicable, by independently reducing the center and the offset of the box. However, it hardly makes use of geometric properties of boxes, and thus it does not properly reflect the complex relations between them. The proposed box quantization scheme effectively addresses this issue through two steps: **(1)** box discretization and **(2)** box reconstruction.

∘ **Box Discretization.** Given a box $\mathrm{B}_s = (\mathrm{c}_s, \mathrm{f}_s)$ of set s, we discretize the box as a K-way D-dimensional discrete code $\mathrm{C}_s \in \{1, \cdots, K\}^D$ which is more compact and requires much less number of bits to encode than real numbers. To this end, we divide the d-dimensional latent space into D subspaces ($\mathbb{R}^{d/D}$) and, for each subspace, learn K *key boxes*. Specifically, in the i^{th} subspace, the j^{th} key box is denoted by $\mathrm{K}^{(i)}_j = (\mathrm{c}^{(i)}_j, \mathrm{f}^{(i)}_j)$ where $\mathrm{c}^{(i)}_j \in \mathbb{R}^{d/D}$ and $\mathrm{f}^{(i)}_j \in \mathbb{R}^{d/D}_+$ are the center and offset of the key box, respectively. The original box B_s is also partitioned into D sub-boxes $\mathrm{B}^{(1)}_s, \cdots, \mathrm{B}^{(D)}_s$ and the i^{th} code of C_s is decided by:

$$\mathrm{C}_s[i] = \arg\max_j \textbf{sim}\left(\mathrm{B}^{(i)}_s,\ \mathrm{K}^{(i)}_j\right)$$

where $\textbf{sim}(\cdot, \cdot)$ measures the similarity between two boxes, and we can flexibly select the criterion. In this paper, we specify the **sim** function, using softmax, as:

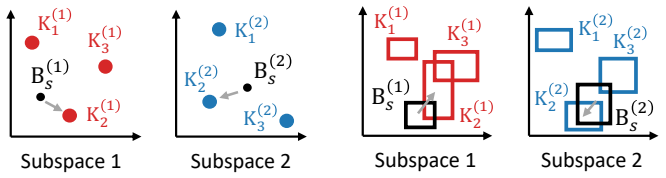
$$\mathrm{C}_s[i] = \arg\max_j \frac{\exp\left(\textbf{BOR}\left(\mathrm{B}^{(i)}_s, \mathrm{K}^{(i)}_j\right)\right)}{\sum_{j'} \exp\left(\textbf{BOR}\left(\mathrm{B}^{(i)}_s, \mathrm{K}^{(i)}_{j'}\right)\right)} \tag{2}$$

where **BOR** (Box Overlap Ratio) is defined to measure how much a box B_X and a box B_Y overlap:

$$\textbf{BOR}(\mathrm{B}_X, \mathrm{B}_Y) = \frac{1}{2}\left(\frac{\mathbb{V}(\mathrm{B}_X \cap \mathrm{B}_Y)}{\mathbb{V}(\mathrm{B}_X)} + \frac{\mathbb{V}(\mathrm{B}_X \cap \mathrm{B}_Y)}{\mathbb{V}(\mathrm{B}_Y)}\right).$$

As shown in Figure 1, the proposed box quantization scheme incorporates the geometric relations between boxes, differently

[2] We assume that we are using float-32 to represent each real number.



(a) Product Quantization (b) Box Quantization

Fig. 1: An example of (a) product quantization and (b) box quantization when D (number of subspaces) $= 2$ and K (number of key boxes) $= 3$. The proposed box quantization incorporates geometric relations between boxes.

from conventional product quantization methods on vectors. To sum up, for each i^{th} subspace, we search for the key sub-box closest to the sub-box $\mathrm{B}_s^{(i)}$ and assign its index as the i^{th} dimension's value of its discrete code.

∘ **Box Reconstruction.** Once the discrete code C_s of set s is generated, in this step, we reconstruct the original box based on it. To be specific, we obtain the reconstructed box $\widehat{\mathrm{B}}_s = (\widehat{c}_s, \widehat{f}_s)$ by concatenating D key boxes from each subspace encoded in C_s, i.e., $\widehat{\mathrm{B}}_s = \|_{i=1}^{D} \mathrm{K}^{(i)}_{\mathrm{C}_s[i]}$. That is, $\widehat{\mathrm{B}}_s$ is reconstructed by concatenating the centers and the offsets of the D key boxes respectively. Since C_s encodes key boxes that largely overlap with the box B_s (i.e., high BOR), if properly encoded, we can expect the reconstructed box $\widehat{\mathrm{B}}_s$ to be geometrically similar to the original box B_s.

Differentiable Optimization: Recall that SET2BOX$^+$ is an end-to-end differentiable algorithm. However, the $\arg\max$ operation in Eq. (2) is non-differentiable, and to this end, we utilize the softmax with the temperature τ:

$$\widetilde{\mathrm{C}}_s^{(i)}[j] = \frac{\exp\left(\textbf{BOR}\left(\mathrm{B}_s^{(i)}, \mathrm{K}_j^{(i)}\right)/\tau\right)}{\sum_{j'} \exp\left(\textbf{BOR}\left(\mathrm{B}_s^{(i)}, \mathrm{K}_{j'}^{(i)}\right)/\tau\right)}. \tag{3}$$

$\widetilde{\mathrm{C}}_s^{(i)}$ is a K-dimensional probabilistic vector whose j^{th} element indicates the probability for $\mathrm{K}_j^{(i)}$ being assigned as the closest key box, i.e., the probability of $\mathrm{C}_s[i] = j$. Then, the key box $\widetilde{\mathrm{K}}_s^{(i)} = (\widetilde{\mathrm{c}}_s^{(i)}, \widetilde{\mathrm{f}}_s^{(i)})$ in the i^{th} subspace is the weighted sum of the K key boxes, i.e., $\widetilde{\mathrm{K}}_s^{(i)} = \sum_{j=1}^{K} \widetilde{C}_s^{(i)}[j] \cdot \mathrm{K}_j^{(i)}$. If $\tau = 0$, Eq. (3) is equivalent to the $\arg\max$ function, i.e., a one-hot vector where $\mathrm{C}_s[i]^{\text{th}}$ dimension is 1 and others are 0. In this case, $\widetilde{\mathrm{K}}_s^{(i)}$ becomes equivalent to $\mathrm{K}^{(i)}_{\mathrm{C}_s[i]}$, which is the exact reconstruction derivable from the discrete code C_s. However, since this *hard selection* is non-differentiable and thus prevents an end-to-end optimization, we resort to the approximation by using the softmax with $\tau \neq 0$ which is fully differentiable. Specifically, we use different τs' in forward ($\tau = 0$) and backward ($\tau = 1$) passes, which effectively enables differentiable optimization.

Joint Training: For further improvement, we introduce a joint learning scheme in the box quantization scheme. Given a triple $\{s_i, s_j, s_k\}$ of sets from the training data $\mathcal{T}$, we obtain their boxes B_{s_i}, B_{s_j}, and B_{s_k} and their reconstructed ones $\widehat{\mathrm{B}}_{s_i}$, $\widehat{\mathrm{B}}_{s_j}$, and $\widehat{\mathrm{B}}_{s_k}$ using the box quantization. While the basic version of SET2BOX$^+$ optimizes $\mathcal{J}(s_i, s_j, s_j, \widehat{\mathrm{B}}_{s_i}, \widehat{\mathrm{B}}_{s_j}, \widehat{\mathrm{B}}_{s_k})$, we jointly train the original boxes together with the reconstructed ones so that both types of boxes can achieve high accuracy. To this end, we consider the following eight losses:

$$\begin{aligned}
&\mathcal{J}(s_i, s_j, s_j, \mathrm{B}_{s_i}, \mathrm{B}_{s_j}, \mathrm{B}_{s_k}), \quad &&\mathcal{J}(s_i, s_j, s_j, \widehat{\mathrm{B}}_{s_i}, \mathrm{B}_{s_j}, \mathrm{B}_{s_k}),\\
&\mathcal{J}(s_i, s_j, s_j, \mathrm{B}_{s_i}, \widehat{\mathrm{B}}_{s_j}, \mathrm{B}_{s_k}), \quad &&\mathcal{J}(s_i, s_j, s_j, \mathrm{B}_{s_i}, \mathrm{B}_{s_j}, \widehat{\mathrm{B}}_{s_k}),\\
&\mathcal{J}(s_i, s_j, s_j, \widehat{\mathrm{B}}_{s_i}, \widehat{\mathrm{B}}_{s_j}, \mathrm{B}_{s_k}), \quad &&\mathcal{J}(s_i, s_j, s_j, \widehat{\mathrm{B}}_{s_i}, \mathrm{B}_{s_j}, \widehat{\mathrm{B}}_{s_k}),\\
&\mathcal{J}(s_i, s_j, s_j, \mathrm{B}_{s_i}, \widehat{\mathrm{B}}_{s_j}, \widehat{\mathrm{B}}_{s_k}), \quad &&\mathcal{J}(s_i, s_j, s_j, \widehat{\mathrm{B}}_{s_i}, \widehat{\mathrm{B}}_{s_j}, \widehat{\mathrm{B}}_{s_k}),
\end{aligned}$$

where we denote them by $\mathcal{J}_1$ to $\mathcal{J}_8$, for the sake of brevity. Notably, $\mathcal{J}_1$, which utilizes only the original boxes, is an objective used for SET2BOX, and $\mathcal{J}_8$ considers only the reconstructed boxes. Based on these joint views from different types of boxes, the final loss function we aim to minimize is:

$$\mathcal{L} = \sum_{\{s_i, s_j, s_k\} \in \mathcal{T}} \lambda\left(\mathcal{J}_1 + \mathcal{J}_2 + \mathcal{J}_3 + \mathcal{J}_4 + \mathcal{J}_5 + \mathcal{J}_6 + \mathcal{J}_7\right) + \mathcal{J}_8, \tag{4}$$

where λ is the coefficient for balancing the losses between the joint views and the loss from the reconstructed boxes. In this way, both original and reconstructed ones are trained together in the latent space. Note that even though both types of boxes are jointly trained to achieve high accuracy, only the reconstructed boxes are used for inference.

Encoding Cost: To encode the box for each set, SET2BOX$^+$ requires **(1)** key boxes and **(2)** discrete codes to encode each set, which requires $64Kd$ bits and $|D \log_2 K$ bits, respectively. Thus, it requires $64Kd + |\mathcal{S}|D \log_2 K$ bits to encode $|S|$ sets. Notably, if $K \ll |\mathcal{S}|$, then $64Kd$ bits are negligible, and typically, $D \log_2 K \ll 64d$ holds. Thus, the encoding cost of SET2BOX$^+$ is considerably smaller than that of SET2BOX.

Similarity Computation: Once we obtain set representations, it is desirable to rapidly compute the estimated similarities in the latent space. Boxes, which SET2BOX and SET2BOX$^+$ derive, require constant time ($O(d)$) to compute a pairwise similarity between two sets (formalized in [26]).

V. Experimental Results

We review our experiments designed for answering Q1-Q3.

- **Q1. Accuracy & Conciseness:** Does SET2BOX$^+$ derive concise and accurate set representations than its competitors?
- **Q2. Effectiveness:** How does SET2BOX$^+$ yield concise and accurate representations? Are all its components useful?
- **Q3. Effects of Parameters:** How do the parameters of SET2BOX$^+$ affect the quality of set representations?

A. Experimental Settings

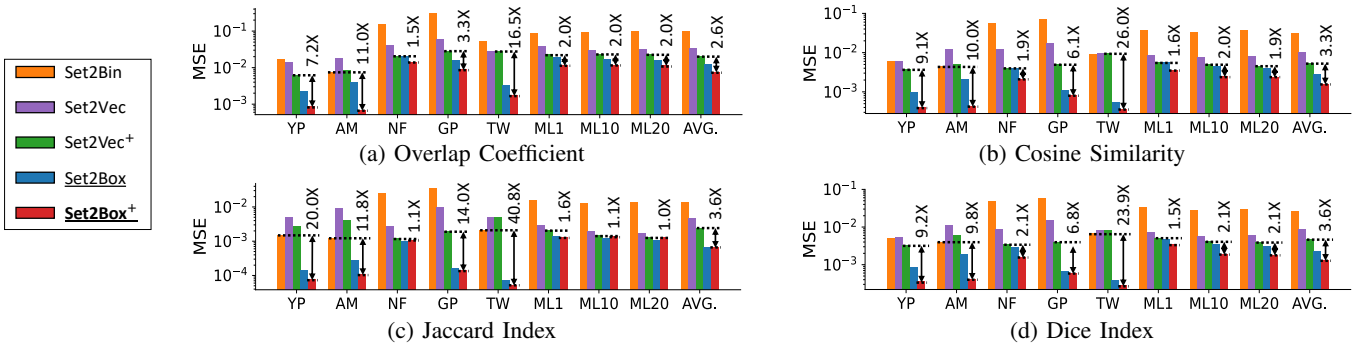
Below, we briefly describe settings (see [26] for details).

Machines & Implementations: All experiments were conducted on a Linux server with RTX 3090Ti GPUs.

Datasets: Yelp (YP), Amazon (AM), Netflix (NF), and MovieLens (ML) are review datasets where each sets is a group of items that a user rated. Gplus (GP) and Twitter (TW) are social networks where each set is a group of neighbors of each node.

Baselines: We compare SET2BOX and SET2BOX$^+$ with:

- **SET2BIN** encodes each set s as a binary vector $\mathrm{z}_s \in \{0, 1\}^d$ using a random hash function. See Section III for details.



(a) Overlap Coefficient (b) Cosine Similarity

(c) Jaccard Index (d) Dice Index

Fig. 2: SET2BOX+ preserves set similarities more accurately than SET2BIN, SET2VEC, SET2VEC+, and SET2BOX. Note that in SET2BOX+, only $0.31 - 0.40$ of the bits costed by the competitors are used to embed sets. Moreover, while SET2VEC and SET2VEC+ need to be trained for specifically each similarity metric, SET2BOX and SET2BOX+ do not separate training.

- **SET2VEC** embeds each set s as a vector $z_s \in \mathbb{R}^d$ which is obtained by pooling learnable entity embeddings using SCP.
- **SET2VEC+** incorporates entity features $\mathbf{X} \in \mathbb{R}^{|\mathcal{E}| \times d}$ into the set representation. Features are projected using a fully-connected layer and then pooled to a set embedding.

Recall that vector-based methods, SET2VEC and SET2VEC+, are not versatile, and thus they need to be trained specifically to each metric, while the proposed methods SET2BOX and SET2BOX+ do not separate training. The encoding cost of each method is analyzed in [26].

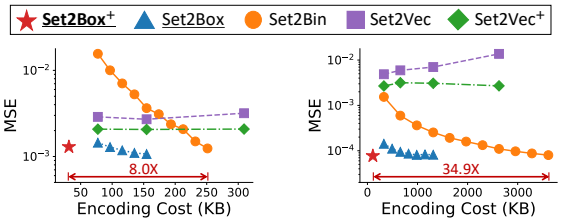
Evaluation: We used 20% of the sets for training (5% in Netflix), and the remaining sets are split into two halves for validation and test. We sampled 100,000 pairs uniformly at random for and measured the Mean Squared Error (MSE) to evaluate the accuracy of the set similarity approximation. We consider four representative set-similarity measures for evaluation: Overlap Coefficient (OC), Cosine Similarity (CS), Jaccard Index (JI), and Dice Index (DI).

B. Q1. Accuracy & Conciseness

We compare the MSE of the set similarity estimation derived by SET2BOX+ and its competitors. We set dimensions to 256 for SET2BIN, 8 for vector based methods (SET2VEC and SET2VEC+), and 4 for SET2BOX, so that they use the same number of bits to encode sets. For SET2BOX+, we set $(d, D, K) = (32, 16, 30)$, which results in only $31-40\%$ of the encoding cost of the other methods, unless otherwise stated.

Accuracy: As seen in Figure 2, SET2BOX+ yields the most accurate set representations while using a smaller number of bits to encode them. For example, in the Twitter, SET2BOX+ gives $40.8\times$ smaller MSE for the Jaccard Index compared to SET2BIN. In both cases, SET2BOX+ uses about 31% of the encoding costs used by the competitors.

Conciseness: To verify the conciseness of SET2BOX+, we measure the accuracy of competitors across various encoding costs. As seen in Figure 3, SET2BOX+ yields compact representations of sets while keeping them informative. Vector-based methods are prone to the curse of dimensionality and hardly benefit from high dimensionality. While the MSE of SET2BIN decreases with respect to its dimension, it still requires larger space to achieve the MSE of SET2BOX+. For example, SET2BIN requires $34.9\times$ more bits to achieve the same accuracy of SET2BOX+ in Yelp. This is more noticeable in larger datasets, where SET2BIN requires up to $96.8\times$ of the encoding cost of SET2BOX+, as shown in Figure 3c.



(a) MovieLens 1M (b) Yelp

Metric	ML1	ML10	ML20	YP	AM	GP	TW	NF
JI	8.0	11.1	12.9	34.9	33.6	76.2	41.2	16.2
DI	8.0	15.9	17.7	27.3	27.2	63.5	31.7	22.7
OC	8.0	12.7	16.1	34.9	28.8	96.8	60.3	19.5
CS	8.0	15.9	16.1	28.8	28.8	68.2	38.0	22.7

(c) The number of times of the encoding cost that SET2BIN requires to catch up the accuracy of SET2BOX+.

Fig. 3: SET2BOX+ yields concise set representations.

C. Q2. Effectiveness

To verify the effectiveness of each component of SET2BOX+, we conduct ablation studies by comparing it with its variants. We first consider the following variants:

- **SET2BOX-PQ:** For a box $\mathrm{B} = (\mathrm{c}, \mathrm{f})$, we apply an end-to-end differentiable product quantization (PQ) [24] to the center c and the offset f independently. It yields two independent discrete codes for the center and the offset, and thus its encoding cost is about twice that of SET2BOX+.
- **SET2BOX-BQ:** SET2BOX+ with $\lambda = 0$, where the proposed box quantization is applied but joint training is not.

We set (d, D, K) to $(32, 8, 30)$ for SET2BOX-PQ and $(32, 16, 30)$ for SET2BOX-BQ and SET2BOX+ so they all require the the same amount of storage.

Effects of Box Quantization: We examine the effectiveness of the proposed box quantization scheme by comparing SET2BOX-BQ with SET2BOX-PQ. As shown in Table II, on average, SET2BOX-BQ yields up to 26% smaller MSE than SET2BOX-PQ while using about the same number of bits. While SET2BOX-PQ discretizes the center and the offset of the boxes independently, without considering their geometric

TABLE II: The proposed schemes: box quantization and joint training in SET2BOX$^+$ incrementally improves the accuracy (in terms of MSE) averaged over all considered datasets.

Method	OC	CS	JI	DI
SET2BOX-PQ	0.0129	0.0028	0.0012	0.0023
SET2BOX-BQ	0.0106 (-17%)	0.0023 (-17%)	0.0009 (-26%)	0.0019 (-17%)
SET2BOX$^+$	**0.0077** (**-40%**)	**0.0016** (**-44%**)	**0.0007** (**-41%**)	**0.0013** (**-42%**)

properties, the proposed box quantization scheme effectively takes the geometric relations between boxes into account and thus yields high-quality compression.

Effects of Joint Training: We analyze the effects of the joint training scheme of SET2BOX$^+$ by comparing SET2BOX-BQ ($\lambda = 0$) and SET2BOX$^+$ ($\lambda \geq 0$). As summarized in Table II, joint training reduces the average MSEs on the considered datasets, by up to 44%, together with the box quantization scheme. These results imply that learning quantized boxes simultaneously with the original boxes improves the quality of the quantization and thus its effectiveness. We also observe that joint training helps not only stabilize but also facilitate the training optimization [26].

Effects of Boxes: To confirm the effectiveness of using boxes for representing sets, we consider SET2BOX-ORDER, which is also a region-based geometric embedding method:

- **SET2BOX-ORDER [27]:** A set s is represented as a d-dimensional vector $\mathrm{z}_s \in \mathbb{R}_+^d$ whose volume is computed as $\mathbb{V}(\mathrm{z}_s) = \exp(-\sum_i \mathrm{z}_s[i])$. It is equivalent to restrict boxes to be located in positive latent space.

We set the dimensions for SET2BOX-ORDER and SET2BOX to 8 and 4, respectively, so that their encoding costs are the same. In Table III, we compare SET2BOX with SET2BOX-ORDER in terms of the average MSE on the considered datasets for each measure. SET2BOX yields more accurate representations than SET2BOX-ORDER, implying the effectiveness of boxes to represent sets for similarity preservation.

D. Q3. Effects of Parameters

We analyze how parameters in SET2BOX$^+$ affect the embedding quality of the set representations. In summary, the estimation error decreases as the number of subspaces (D) and the number of key boxes in each subspace (K) increase, and especially it is affected more heavily by D than by K. Detailed experimental results are available in [26].

VI. CONCLUSIONS

In this work, we propose SET2BOX and SET2BOX$^+$, space-efficient similarity-preserving embedding methods for sets. Compared to the competitors, by representing sets as boxes with novel quantization and training schemes, SET2BOX$^+$ is (a) **Accurate:** with $40.8\times$ smaller estimation error (when encoding costs are similar or smaller) (b) **Concise:** with $96.8\times$ smaller encoding costs (when accuracies are similar), and (c) **Versatile:** accurate in terms of various similarity measures.

Acknowledgements: This work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2022-0-00157, Robust, Fair, Extensible Data-Centric Continual Learning) (No. 2019-0-00075, Artificial Intelligence Graduate School Program (KAIST)).

TABLE III: SET2BOX yields smaller MSE on average in the considered datasets than SET2BOX-ORDER.

Method	OC	CS	JI	DI
SET2BOX-ORDER	0.0320	0.0033	0.0008	0.0027
SET2BOX	**0.0121** (**-62%**)	**0.0028** (**-14%**)	**0.0006** (**-22%**)	**0.0022** (**-17%**)

REFERENCES

[1] L. Moussiades and A. Vakali, "Pdetect: A clustering approach for detecting plagiarism in source code datasets," *The Computer Journal*, vol. 48, no. 6, pp. 651–661, 2005.
[2] N. A. Yousri and D. M. Elkaffash, "Associating gene functional groups with multiple clinical conditions using jaccard similarity," in *BIBMW*, 2011.
[3] Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," in *KDD*, 2008.
[4] K. Shin, A. Ghoting, M. Kim, and H. Raghavan, "Sweg: Lossless and lossy summarization of web-scale graphs," in *WWW*, 2019.
[5] L. Chauvin, K. Kumar, C. Desrosiers, W. Wells III, and M. Toews, "Efficient pairwise neuroimage analysis using the soft jaccard index and 3d keypoint sets," *arXiv:2103.06966*, 2021.
[6] R. Guerraoui, A.-M. Kermarrec, O. Ruas, and F. Taïani, "Smaller, faster & lighter knn graph constructions," in *WWW*, 2020.
[7] B. Cui, H. T. Shen, J. Shen, and K. L. Tan, "Exploring bit-difference for approximate knn search in high-dimensional databases," in *AusDM*, 2005.
[8] Y. Xie, M. Gong, S. Wang, W. Liu, and B. Yu, "Sim2vec: Node similarity preserving network embedding," *Information Sciences*, vol. 495, pp. 37–51, 2019.
[9] A. Tsitsulin, D. Mottin, P. Karras, and E. Müller, "Verse: Versatile graph embeddings from similarity measures," in *WWW*, 2018.
[10] Q. Li, Z. Sun, R. He, and T. Tan, "Deep supervised discrete hashing," in *NeurIPS*, 2017.
[11] Y. Li, Y. Sun, and N. Zhu, "Berttocnn: Similarity-preserving enhanced knowledge distillation for stance detection," *Plos one*, vol. 16, no. 9, p. e0257130, 2021.
[12] L. Vilnis, X. Li, S. Murty, and A. McCallum, "Probabilistic embedding of knowledge graphs with box lattice measures," in *ACL*, 2018.
[13] R. Abboud, I. Ceylan, T. Lukasiewicz, and T. Salvatori, "Boxe: A box embedding model for knowledge base completion," in *NeurIPS*, 2020.
[14] X. Chen, M. Boratko, M. Chen, S. S. Dasgupta, X. L. Li, and A. McCallum, "Probabilistic box embeddings for uncertain knowledge graph reasoning," in *ACL*, 2021.
[15] H. Ren, W. Hu, and J. Leskovec, "Query2box: Reasoning over knowledge graphs in vector space using box embeddings," in *ICLR*, 2019.
[16] S. S. Dasgupta, M. Boratko, S. Atmakuri, X. L. Li, D. Patel, and A. McCallum, "Word2box: Learning word representation using box embeddings," *arXiv:2106.14361*, 2021.
[17] A. Rau, G. Garcia-Hernando, D. Stoyanov, G. J. Brostow, and D. Turmukhambetov, "Predicting visual overlap of images through interpretable non-metric box embeddings," in *ECCV*, 2020.
[18] S. Zhang, H. Liu, A. Zhang, Y. Hu, C. Zhang, Y. Li, T. Zhu, S. He, and W. Ou, "Learning user representations with hypercuboids for recommender systems," in *WSDM*, 2021.
[19] K. Deng, J. Huang, and J. Qin, "Box4rec: Box embedding for sequential recommendation," in *PAKDD*, 2021.
[20] X. Li, L. Vilnis, D. Zhang, M. Boratko, and A. McCallum, "Smoothing the geometry of probabilistic box embeddings," in *ICLR*, 2018.
[21] S. Dasgupta, M. Boratko, D. Zhang, L. Vilnis, X. Li, and A. McCallum, "Improving local identifiability in probabilistic box embeddings," in *NeurIPS*, 2020.
[22] H. Jegou, M. Douze, and C. Schmid, "Product quantization for nearest neighbor search," *IEEE TPAMI*, vol. 33, no. 1, pp. 117–128, 2010.
[23] T. Ge, K. He, Q. Ke, and J. Sun, "Optimized product quantization for approximate nearest neighbor search," in *CVPR*, 2013.
[24] T. Chen, L. Li, and Y. Sun, "Differentiable product quantization for end-to-end embedding compression," in *ICML*, 2020.
[25] T. Chen, M. R. Min, and Y. Sun, "Learning k-way d-dimensional discrete codes for compact embedding representations," in *ICML*, 2018.
[26] G. Lee, C. Park, and K. Shin, "Set2box: Similarity preserving representation learning for sets (extended version)," *arXiv:2210.03282*, 2022.
[27] A. Lai and J. Hockenmaier, "Learning to predict denotational probabilities for modeling entailment," in *ACL*, 2017.