Transport of Algebraic Structure to Latent Embeddings

Samuel Pfrommer [1] **Brendon G. Anderson** [1] **Somayeh Sojoudi** [1]

Abstract

Machine learning often aims to produce latent embeddings of inputs which lie in a larger, abstract mathematical space. For example, in the field of 3D modeling, subsets of Euclidean space can be embedded as vectors using implicit neural representations. Such subsets also have a natural algebraic structure including operations (e.g., union) and corresponding laws (e.g., associativity). How can we learn to "union" two sets using only their latent embeddings while respecting associativity? We propose a general procedure for parameterizing latent space operations that are provably consistent with the laws on the input space. This is achieved by learning a bijection from the latent space to a carefully designed *mirrored algebra* which is constructed on Euclidean space in accordance with desired laws. We evaluate these *structural transport nets* for a range of mirrored algebras against baselines that operate directly on the latent space. Our experiments provide strong evidence that respecting the underlying algebraic structure of the input space is key for learning accurate and self-consistent operations.

1. Introduction

Algebraic structure underpins a wide range of interesting mathematical objects such as sets, functions, distributions, and symbolic strings. In machine learning (ML), these objects are often learned and subsequently embedded into Euclidean space for downstream tasks: consider embeddings of implicit neural representations (INRs) for sets (De Luigi et al., 2023), hypernetworks for functions (Ha et al., 2017), conditional embeddings of generative architectures for probability distributions (Sohn et al., 2015; Winkler et al., 2019; Nichol et al., 2021), and text embeddings for strings (Wang et al., 2022; Devlin et al., 2018). Our goal is to enable mathematical operations from the underlying algebraic structure (e.g., set union when the underlying objects are sets) to be applied directly to latent embeddings in a way that respects axiomatic laws.

The importance of respecting mathematical structure has motivated machine learning developments of immense importance. Indeed, much of geometric deep learning is directly driven by symmetries in underlying objects (Bronstein et al., 2021). Graph neural networks learn functions that provably respect equivariance or invariance properties under node-relabeling graph isomorphisms (Maron et al., 2018; Azizian & Lelarge, 2020). The seminal DeepSet architecture enforces permutation invariance, reflecting the unordered nature of its finite set inputs (Zaheer et al., 2017). Convolutional filters are also known to be approximately equivariant to translations in input images—a structure which naturally mirrors that of the underlying image manifold (Cohen & Welling, 2016; Cohen et al., 2019; Kondor & Trivedi, 2018).

This work is a first attempt to transport general algebraic structures from input data onto learned latent embeddings. We outline a general procedure for defining algebraic *operations* on the latent space that respect *laws* on the *source space* (input space). Defining operations directly on latent space embeddings, rather than using the original source objects, is crucial for computational efficiency and compatibility with larger ML workflows. There has been some interest in algebraic and category theoretic approaches to the study of specific computational architectures and automatic differentiation (Martin-Maroto & de Polavieja, 2018; Shiebler et al., 2021; Sennesh et al., 2023), as well as in the application of ML to computational problems arising in algebra (He & Kim, 2023). However, to the best of our knowledge, our work provides the first general method to transport algebraic structures to learned embeddings.

We discuss our ideas using the language of *universal algebra*, which studies algebraic structures as general pairings of a set with a collection of operations (Burris & Sankappanavar, 1981). We note that universal algebra is subsumed within category theory. As the universal algebraic perspective is sufficient here, we avoid generalizing to more complex category-theoretic frameworks.

[1]University of California, Berkeley. Correspondence to: Samuel Pfrommer <sam.pfrommer@berkeley.edu>.

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

Source code for our experiments is available on GitHub.

As our transport of algebraic structures relies on the construction of a bijection map, we leverage architectures from the invertible neural network literature. Our model of choice is the seminal NICE architecture, which uses coupling layers to enable easily-computable forward and inverse methods (Dinh et al., 2015). These coupling layers have been shown to be universal diffeomorphism approximators (Teshima et al., 2020), and are best known for their usefulness in constructing normalizing flows (Papamakarios et al., 2021; Kobyzev et al., 2020). Since our application requires differentiation through the function inverse, other architectures which rely on solving fixed-point iterations to compute inverses are not considered (Behrmann et al., 2019).

We focus on embeddings of positive-volume subsets of $\mathbb{R}^d$ as a working example. This is distinct from methods that consider finite sets, such as DeepSets (Zaheer et al., 2017). Our setting is motivated by the practical application of learning shapes for 3D modeling and graphics (Park et al., 2019). Typical approaches parameterize a signed distance function or simply regress on a shape indicator function (Park et al., 2019; Mescheder et al., 2019; Chen & Zhang, 2019). As the object surface is implicitly defined as a level set of the resulting network, this is termed an *Implicit Neural Representation* (INR). A subsequent innovation that we adopt improves representation quality by introducing sinusoidal activations (Sitzmann et al., 2020). While implicit representations of shapes achieve strong performance for a variety of objects, the significant storage requirements of the corresponding networks are impractical for larger workflows. Recent research has addressed this by directly compressing INR weights into latent embeddings (De Luigi et al., 2023), enabling a variety of downstream tasks such as shape generation.

1.1. Contributions

Our work establishes the following contributions.

1. We develop a general procedure for transporting algebraic structure from the source data to the latent embedding space. This is accomplished via a learned bijection to a carefully designed mirrored algebra.
2. We illustrate the subtleties that arise with this procedure by considering algebras of sets as a case study. Namely, we mathematically prove that transporting all three basic set operations (union, intersection, and complementation) is infeasible and subsequently drop complementation, yielding a distributive lattice structure on the source space which is transportable.
3. We experimentally validate Hypothesis 1 on this distributive lattice of sets, showing that adherence to source algebra laws is crucial for strong learned operation performance.

Hypothesis 1. *Learned latent space operations will achieve higher performance if they are constructed to satisfy the laws of the underlying source algebra.*

2. Universal algebra primer

In this section, we briefly recall the pertinent definitions and notations used throughout this paper. We refer the reader to Burris & Sankappanavar (1981) and Wechler (2012) for detailed texts concerning universal algebra.

Algebras and isomorphisms. Let A be a nonempty set and n a nonnegative integer. If $n = 0$, we define $A^n = \{\emptyset\}$. A function $f\colon A^n \to A$ is called an *n-ary operation on A*, and n is called the *arity of f*. If the arity of f is 1, then f is called a *unary operation*, and if the arity of f is 2, then f is called a *binary operation*. If the arity of f is 0, then f is called a *nullary operation*, which may be identified with an element of A. We will commonly denote nullary operations, unary operations, and binary operations by $f = f(\emptyset)$, $fa = f(a)$, and $afb = f(a, b)$, respectively.

A *type* is a set $\mathcal{F}$, whose elements are called *operation symbols*, together with a function $\mathrm{ar}\colon \mathcal{F} \to \mathbb{N} \cup \{0\}$. If $f \in \mathcal{F}$ and $\mathrm{ar}(f) = n$, then an n-ary operation $f^{\mathcal{A}}\colon A^n \to A$ is called a *realization of f on A*.

An *algebra of type $\mathcal{F}$* is an ordered pair $\mathcal{A} = (A, \mathcal{F}^{\mathcal{A}})$ with A being a nonempty set and $\mathcal{F}^{\mathcal{A}} = \{f^{\mathcal{A}} : f \in \mathcal{F}\}$ being a family of realizations $f^{\mathcal{A}}$ of operation symbols f on A, and with $\mathcal{F}^{\mathcal{A}}$ in one-to-one correspondence with $\mathcal{F}$.

One of the most fundamental algebras is a *group*, which is an algebra $(A, \bullet, {}^{-1}, e)$ whose operations satisfy

$$e \bullet a = a, \tag{G1}$$

$$(a^{-1}) \bullet a = e, \tag{G2}$$

$$(a \bullet b) \bullet c = a \bullet (b \bullet c), \tag{G3}$$

for all $a, b, c \in A$. Here, $\bullet$ is a binary operation, ${}^{-1}$ is a unary operation, and e is a nullary operation. The equations (G1), (G2), and (G3) are the group's underlying *laws*, which we will define shortly. We use the term *algebraic structure* to refer to a combination of a type and a collection of laws.

Consider two algebras $\mathcal{A} = (A, \mathcal{F}^{\mathcal{A}})$ and $\mathcal{B} = (B, \mathcal{F}^{\mathcal{B}})$ of type $\mathcal{F}$. A function $\varphi\colon A \to B$ is called an *homomorphism from $\mathcal{A}$ to $\mathcal{B}$* if it satisfies

$$\varphi(f^{\mathcal{A}}(a_1, \ldots, a_n)) = f^{\mathcal{B}}(\varphi(a_1), \ldots, \varphi(a_n))$$

for all $f \in \mathcal{F}$ and all $a_1, \ldots, a_n \in A$, where of course $n = \mathrm{ar}(f)$. If, additionally, φ is bijective, then it is called an *isomorphism from $\mathcal{A}$ to $\mathcal{B}$*. If $\mathcal{A}$ is isomorphic to $\mathcal{B}$ (meaning there is an isomorphism φ from $\mathcal{A}$ to $\mathcal{B}$), then we

write $\mathcal{A} \cong \mathcal{B}$. Isomorphic algebras satisfy the same laws, and hence can be viewed as the same algebraic structures.

Two algebras may be of the same type yet not be isomorphic, and thus have fundamentally different structures. For example, rings and lattices are distinct algebraic structures of common type $\mathcal{F} = \{f_1, f_2\}$ with $\mathrm{ar}(f_1) = \mathrm{ar}(f_2) = 2$.

Terms and laws. For a set of *variables* X and a type $\mathcal{F}$, the set $T_{\mathcal{F}}(X)$ is the set of *terms of type $\mathcal{F}$ over X* and consists of all strings of variables in X and nullary operations in $\mathcal{F}$, connected by n-ary operations. For example, consider a type $\mathcal{F}$ with one binary operation $\bullet$ and a nullary operation e. If $X = \{x, y\}$, then x, y, e, $x \bullet y$, $x \bullet (y \bullet e)$, and $x \bullet (x \bullet y)$ are all examples of terms in $T_{\mathcal{F}}(X)$.

Note that a term $p(x_1, \ldots, x_n) \in T_{\mathcal{F}}(X)$ is defined independently of any specific algebra of type $\mathcal{F}$. Making the term concrete for a particular algebra $\mathcal{A} = (A, \mathcal{F}^{\mathcal{A}})$ of type $\mathcal{F}$ yields a *term function* $p^{\mathcal{A}} \colon A^n \to A$. Namely, $p^{\mathcal{A}}(a_1, \ldots, a_n)$ substitutes $a_i \in A$ for x_i in the term $p(x_1, \ldots, x_n)$, and recursively evaluates using the realized operations from $\mathcal{A}$. Continuing the previous example, let $\mathcal{A}$ be the group of the real numbers equipped with the standard addition operation. The term $p(x, y) = x \bullet (y \bullet e)$ would yield the term function given by $p^{\mathcal{A}}(a, b) = a + (b + 0)$.

We call two terms $p(x_1, \ldots, x_n), q(x_1, \ldots, x_n) \in T_{\mathcal{F}}(X)$ *equivalent* with respect to an algebra $\mathcal{A}$ if, for all $a_i \in A$, it holds that $p^{\mathcal{A}}(a_1, \ldots, a_n) = q^{\mathcal{A}}(a_1, \ldots, a_n)$.

A *law* R for a type $\mathcal{F}$ is now defined as the equality of two terms $p(x_1, \ldots, x_n), q(x_1, \ldots, x_n) \in T_{\mathcal{F}}(X)$:

$$R : p(x_1, \ldots, x_n) = q(x_1, \ldots, x_n).$$

We use R instead of the more common letter L, which we reserve for referring to latent spaces. For our running example, the commutative law for the underlying type $\mathcal{F} = \{\bullet, {}^{-1}, e\}$ over a set of variables $X = \{x, y\}$ is given by

$$x \bullet y = y \bullet x.$$

Finally, we say that an algebra $\mathcal{A}$ of type $\mathcal{F}$ *satisfies*, or *respects*, a law $R : p(x_1, \ldots, x_n) = q(x_1, \ldots, x_n)$ if the law holds for realizations of the terms as term functions:

$$R^{\mathcal{A}} : p^{\mathcal{A}}(a_1, \ldots, a_n) = q^{\mathcal{A}}(a_1, \ldots, a_n) \text{ for all } a_i \in A.$$

It is clear that the group of reals under addition satisfies the commutative law, since $a + b = b + a$ for all $a, b \in \mathbb{R}$.

3. Method

With the framework of universal algebra now developed, we may formally describe the goal of this paper. Consider a machine learning task in which input data is drawn from a *source algebra* $\mathcal{S} = (S, \mathcal{F}^{\mathcal{S}})$ of type $\mathcal{F}$. The canonical example we consider is that where input data takes the form of a set, and hence has associated operations of intersection, union, and complementation. The typical ML pipeline embeds source data from the source space S into a Euclidean latent space $L = \mathbb{R}^l$. However, such latent space embeddings do not respect the algebraic structures encoded in $\mathcal{S}$; they are only endowed with the unrelated vector space structure of $\mathbb{R}^l$. Thus, the goal of this paper is as follows:

Transport the algebraic structure $\mathcal{S}$ of the source space S onto the latent space L.

Specifically, we seek to transport both the operations and laws of $\mathcal{S}$ onto L. We emphasize that our goal of structural transport is distinct from constructing an isomorphism (or even a nontrivial homomorphism) $S \to L$; this is not generally possible, since S is problem-determined and our setting assumes a pretrained encoder-decoder architecture which fixes L.

Algorithm 1 Transport of algebraic structure from $\mathcal{S}$ to L

In: Source alg. $\mathcal{S}$, latent space L, encoder E, decoder D
Out: Latent algebra $\mathcal{L}$

Fix mirrored space $M = \mathbb{R}^l$
Select mirrored algebra $\mathcal{M}$ `# Same type as` $\mathcal{S}$
Parameterize bijection φ
Define induced latent algebra $\mathcal{L}$ `# Via` (1)
Learn parameters of φ `# Via` (2)

Description of the method. The general steps of our method are described in Algorithm 1, with a corresponding visualization in Figure 1. We assume that there is a fixed encoder $E \colon S \to L$ mapping source data to latent embeddings and a corresponding decoder D (e.g., a pretrained autoencoder-style network). To transport the algebraic structure from the source algebra $\mathcal{S}$ to the latent space L, we propose to learn a bijective map φ from L to another space $M = \mathbb{R}^l$ of the same dimension. We may consider M as an "alternative latent space," albeit one in which we have complete design authority to impose operations that turn M into an algebra $\mathcal{M} = (M, \mathcal{F}^{\mathcal{M}})$ of the same type $\mathcal{F}$ as $\mathcal{S}$. Although we focus on the pretrained encoder-decoder setting for maximum flexibility, it is certainly possible to jointly learn φ together with the E and D in practice.

Concretely, we endow our *mirrored space* M with an n-ary operation $f^{\mathcal{M}}$ for each n-ary operation $f^{\mathcal{S}}$ from the source algebra. For an exemplar $\mathcal{S}$ with group structure, we would define one binary operation $\bullet^{\mathcal{M}} : \mathbb{R}^l \times \mathbb{R}^l \to \mathbb{R}^l$, one unary operation $({}^{-1})^{\mathcal{M}} : \mathbb{R}^l \to \mathbb{R}^l$, and one nullary operation identified with some element $e^{\mathcal{M}} \in \mathbb{R}^l$.

We refer to the constructed $\mathcal{M}$ as the *mirrored algebra*. Al-

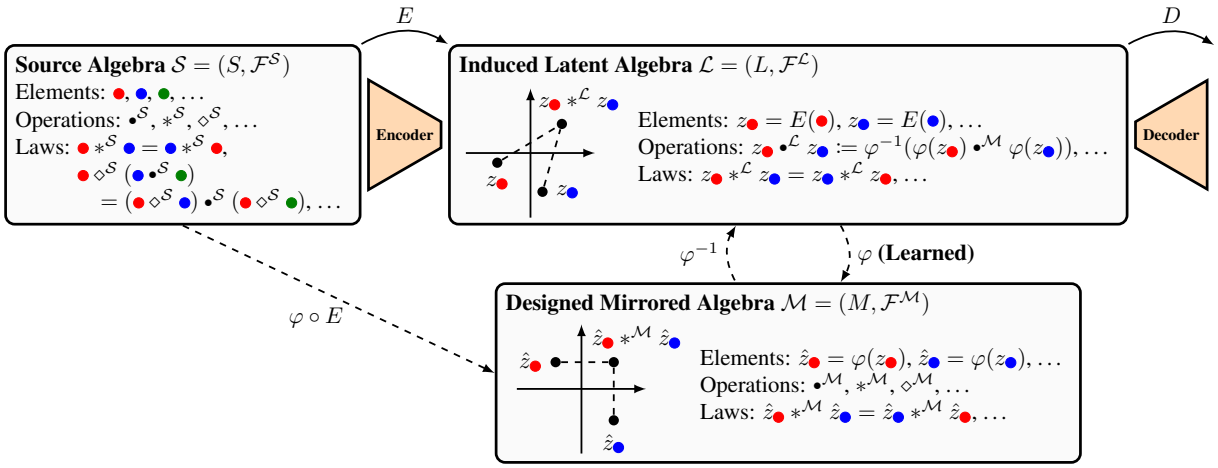

Figure 1. The proposed method for transporting algebraic structure from $\mathcal{S}$ onto the latent space L. The bijection φ is learned (hence the dashed arrows) in such a way as to best "align" the latent structure $\mathcal{L}$, induced from $\mathcal{M}$, with the given source structure $\mathcal{S}$. All other components are either fixed (e.g., the encoder and decoder) or designed *a priori* (e.g., the mirrored algebra).

though it is always possible to endow M with an algebra of the same type as $\mathcal{S}$, it is generally not possible to ensure that the resulting algebra $\mathcal{M}$ is isomorphic to $\mathcal{S}$. This may either be due to the fact that S has cardinality strictly greater than M (due to the embedding process E), or due to inherent incompatibilities between the laws of $\mathcal{S}$ and the natural Euclidean structure on M. Such incompatibilities are discussed in further detail with our case study in Section 4. We note that the term "mirrored algebra" is our own and should not be conflated with other concepts in the literature.

We now transport the structure of our designed mirrored algebra $\mathcal{M}$ to the latent space L via a learned bijection $\varphi : L \to M$. Bijectivity is ensured by parameterizing φ as an invertible neural network using the architecture proposed in Dinh et al. (2015). This automatically induces an algebraic structure from $\mathcal{M}$ onto L. Namely, for every n-ary operation $f^{\mathcal{M}} \in \mathcal{F}^{\mathcal{M}}$, we define the realization $f^{\mathcal{L}} : L^n \to L$ of the corresponding operation symbol f by

$$f^{\mathcal{L}}(z_1, \ldots, z_n) := \varphi^{-1}\big(f^{\mathcal{M}}(\varphi(z_1), \ldots, \varphi(z_n))\big), \quad (1)$$

for $z_1, \ldots, z_n \in L$ and $\mathrm{ar}(f) = n$. Intuitively, the operation $f^{\mathcal{L}}$ is implemented by mapping latent embeddings into the mirrored space M, performing the corresponding operation $f^{\mathcal{M}}$ on these mirrored embeddings, and then pulling the result back to the latent space L. Of course, if $f^{\mathcal{M}}$ is a nullary operation M, then we define the corresponding operation $f^{\mathcal{L}}$ to be the nullary operation on L given by $f^{\mathcal{L}}(\emptyset) = \varphi^{-1}(f^{\mathcal{M}}(\emptyset))$.

Learning φ. We briefly describe the process of learning φ to "align" the induced latent algebra $\mathcal{L}$ with the source algebra $\mathcal{S}$. Aligning $\mathcal{L}$ to $\mathcal{S}$ may be viewed as learning φ so that the laws of $\mathcal{S}$ are also satisfied by $\mathcal{L}$. To achieve this alignment, it suffices to align individual terms realized by $\mathcal{S}$ and $\mathcal{L}$, as laws are just equalities between terms. We propose the following procedure, which is illustrated in Figure 2.

Let $p_i(x_1, \ldots, x_{n_i}) \in T_{\mathcal{F}}(X_i)$ be a "sampled" term of type $\mathcal{F}$ over a variable set X_i. The manner in which this term is sampled is task-dependent, but it suffices to identify this term as a random string involving operation symbols from $\mathcal{F}$ and variables from X_i—see Section 5 for concrete examples. Next, consider data $s_1, \ldots, s_{n_i} \in S$ sampled from the source space. The term is first realized on this source data by computing $p_i^{\mathcal{S}}(s_1, \ldots, s_{n_i})$. The term is then also realized by the induced latent algebra as $D(p_i^{\mathcal{L}}(z_1, \ldots, z_{n_i}))$, with $z_j = E(s_j)$. The loss between this prediction and the ground truth, as a function of the bijection φ, is given by

$$\mathfrak{L}_i(\varphi) := \mathrm{Loss}\left(D(p_i^{\mathcal{L}}(z_1, \ldots, z_{n_i})), p_i^{\mathcal{S}}(s_1, \ldots, s_{n_i})\right),$$

for some appropriately chosen loss function Loss.

For example, if S is the power set of $\mathbb{R}^d$ equipped with intersection and union, the true sampled term might be realized as $p_i^{\mathcal{S}}(s_1, s_2, s_3) = s_1 \cap^{\mathcal{S}} (s_2 \cup^{\mathcal{S}} s_3)$ for some subset data $s_1, s_2, s_3 \subseteq \mathbb{R}^d$, where $\cap^{\mathcal{S}}$ and $\cup^{\mathcal{S}}$ are actual set intersection and union operations, and the corresponding predicted term would be given by $D(E(s_1) \cap^{\mathcal{L}} (E(s_2) \cup^{\mathcal{L}} E(s_3)))$, where $\cap^{\mathcal{L}}$ and $\cup^{\mathcal{L}}$ are the intersection and union realized in Euclidean space by efficient arithmetic operations.

The final learning problem then amounts to solving

$$\inf_{\varphi \in \Phi} \frac{1}{N} \sum_{i=1}^{N} \mathfrak{L}_i(\varphi), \quad (2)$$

for some parameterized class Φ of bijections.

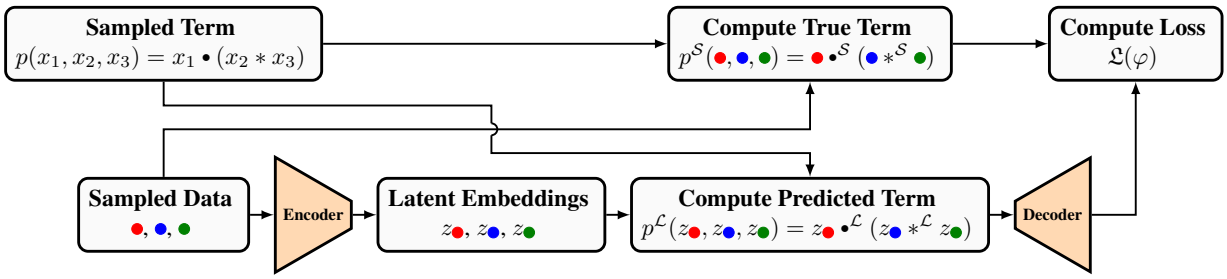

Figure 2. The bijection φ is learned to align true sampled terms $p_i^{\mathcal{S}}(s_1, \ldots, s_{n_i})$ with predicted terms $D(p_i^{\mathcal{L}}(E(s_1), \ldots, E(s_{n_i})))$.

Theoretical developments. Our method comes equipped with theoretical guarantees that the induced latent algebra respects the underlying source algebra. First, we show that the induced algebra is *always* isomorphic to the mirrored algebra by construction. Full proofs for all results are provided in Appendix A.

Proposition 3.1. *Suppose that $L, M = \mathbb{R}^l$ and that $\varphi\colon L \to M$ is a bijection. Let $\mathcal{M} = (M, \mathcal{F}^{\mathcal{M}})$ be an algebra of type $\mathcal{F}$ and define the family $\mathcal{F}^{\mathcal{L}} := \{f^{\mathcal{L}} : f \in \mathcal{F}\}$ of n-ary operations on L by (1). Then, φ is an isomorphism from the induced algebra $\mathcal{L} = (L, \mathcal{F}^{\mathcal{L}})$ to $\mathcal{M}$.*

As a consequence of Proposition 3.1, a well-constructed mirrored space induces an algebra $\mathcal{L}$ such that laws on the source space are satisfied.

Theorem 3.2. *Consider a source algebra $\mathcal{S} = (S, \mathcal{F}^{\mathcal{S}})$ of type $\mathcal{F}$, and let $\mathcal{M} = (M, \mathcal{F}^{\mathcal{M}})$ be a mirrored space such that every law R satisfied by $\mathcal{S}$ is also satisfied by $\mathcal{M}$. Then, the induced latent algebra $\mathcal{L}$, defined by (1), also satisfies every such law R, for any bijection $\varphi\colon L \to M$.*

Proof sketch. For a law $p(x_1, \ldots, x_n) = q(x_1, \ldots, x_n)$ which is satisfied by $\mathcal{M}$, we want to show that $p^{\mathcal{L}}(z_1, \ldots, z_n) = q^{\mathcal{L}}(z_1, \ldots, z_n)$ for all $z_i \in L$. Proposition 3.1 implies that

$$\varphi(p^{\mathcal{L}}(z_1, \ldots, z_n)) = p^{\mathcal{M}}(\varphi(z_1), \ldots, \varphi(z_n)).$$

After applying a similar procedure to q, we can use the fact that R is satisfied by $\mathcal{M}$ to conclude that

$$\varphi(p^{\mathcal{L}}(z_1, \ldots, z_n)) = \varphi(q^{\mathcal{L}}(z_1, \ldots, z_n)).$$

Inverting by φ concludes the proof. □

Unfortunately, there is no general guarantee that an isomorphism, or even a nontrivial homomorphism, exists from the source algebra $\mathcal{S}$ to the induced algebra on L, even when the mirrored algebra satisfies the same laws as $\mathcal{S}$.

Proposition 3.3. *There exists a source algebra $\mathcal{S} = (S, \mathcal{F}^{\mathcal{S}})$ and a mirrored algebra $\mathcal{M} = (M, \mathcal{F}^{\mathcal{M}})$ with $M = \mathbb{R}^l$, both of the same type $\mathcal{F}$, such that $\mathcal{M}$ satisfies every law R that $\mathcal{S}$ satisfies, and, for all bijections $\varphi\colon L \to M$, there is no nontrivial homomorphism $\chi\colon S \to L$ when $L = \mathbb{R}^l$ is equipped with the algebra induced by $\mathcal{M}$ via (1).*

On the other hand, under strong assumptions on the encoder and the expressibility of the source data within Euclidean space, we can guarantee the existence of a bijection φ that recovers an isomorphism $\mathcal{S} \cong \mathcal{M} \cong \mathcal{L}$, despite the fact that the encoder E is fixed.

Proposition 3.4. *Consider a source algebra $\mathcal{S} = (S, \mathcal{F}^{\mathcal{S}})$ of type $\mathcal{F}$, the latent space $L = \mathbb{R}^l$, and an arbitrary encoder $E\colon S \to L$. If E is bijective and there exists a mirrored algebra $\mathcal{M} = (M, \mathcal{F}^{\mathcal{M}})$ with $M = \mathbb{R}^l$ and an isomorphism $\psi\colon S \to M$, then there exists a bijection $\varphi\colon L \to M$ such that $\varphi \circ E$ equals the isomorphism ψ.*

Limitations. There is a major challenge in transporting structure from $\mathcal{S}$ to L: the mirrored space structure may not be amenable to the structure that we want. We will demonstrate this in Section 4, providing a general impossibility result as well as a specific corollary for the Boolean lattice setting. Section 5 experimentally explores this challenge and shows that even satisfying a subset of source algebra laws can still yield substantial benefits. At this point, it is also worth mentioning that our method requires the mirrored space to have the same dimension as the latent space, since our transport of structure depends on the invertibility of φ. Generalizing past this restriction poses an interesting direction for future work.

4. Case study: transporting algebras of sets

We apply our framework to learning the algebra of subsets of Euclidean space. This would empower neural networks to operate directly on *subsets* of $\mathbb{R}^d$ (De Luigi et al., 2023). Conventional networks generally only operate *pointwise*, producing a single output for a single input in $\mathbb{R}^d$. Allowing

for sets to be tractably encoded and operated on unlocks new approaches for a variety of downstream tasks, such as prediction with set-valued uncertainties (Mahjourian et al., 2022), reachable set computation (Meng et al., 2022), safety-constrained trajectory optimization (Michaux et al., 2023), bin packing (Pan et al., 2023), object pile manipulation (Wang et al., 2023), and swept volume approximation in robotics (Chiang et al., 2021).

The purpose of our work is to illustrate the general principles behind structural transport nets and to experimentally test Hypothesis 1. We thus do not specialize to any particular downstream application. Instead, this section explores the procedure for constructing a mirrored algebra via a concrete example, and Section 5 provides controlled synthetic experiments which support Hypothesis 1.

4.1. Lattices of sets

We introduce here the algebraic structures that are considered in this section. A *Boolean lattice* is an algebra $(A, \wedge, \vee, \neg, 0, 1)$ such that the operations $\wedge$, $\vee$, and $\neg$ satisfy the laws listed in Table 1. In a Boolean lattice, the binary operations $\wedge$ and $\vee$ are read "meet" and "join," respectively, and the unary operation $\neg$ is read "not" or "complement." Since 0 and 1 are nullary operations, the "0" and "1" in the listed laws are to be interpreted as these operations' images $0(\emptyset)$ and $1(\emptyset)$ as elements in A. If S is a set and $\mathcal{P}(S)$ is the power set of S, then $(\mathcal{P}(S), \cap, \cup, ^c, \emptyset, S)$ is a Boolean lattice with ^{c} denoting set complementation. Dropping complementation and nullary operations yields a *distributive lattice*, which is depicted in the upper section of Table 1.

We denote the Boolean lattice type as $\mathcal{F}_{\text{Bool}}$, and the distributive lattice type as $\mathcal{F}_{\text{Dist}}$.

4.2. Boolean lattice infeasibility

This section shows that it is impossible to define continuous operations on a Euclidean mirrored space $M = \mathbb{R}^l$ with the type $\mathcal{F}_{\text{Bool}}$ such that the laws in Table 1 are satisfied. Specifically, it is impossible to define a continuous involution with no fixed point, conflicting with complementation laws. We prove this using results from homology and provide both a general statement of the result and its specific implementations for Boolean lattices.

Restricting ourselves to continuous operations is important, as the complementation operation itself is continuous with respect to a natural topology on the space of sets; we explore this more thoroughly in Appendix A.2. A more intuitive justification arises by noting that small perturbations to a set A will yield commensurate perturbations to A^c.

Our first result shows, informally, that it is impossible to realize an algebra with a fixed point-free involution on the mirrored space using continuous operations. We refer the reader to Appendix A.2 for more details and proofs.

Table 1. Distributive and Boolean lattice laws.

Commutativity	$x \wedge y = y \wedge x$
Commutativity*	$x \vee y = y \vee x$
Associativity	$x \wedge (y \wedge z) = (x \wedge y) \wedge z$
Associativity*	$x \vee (y \vee z) = (x \vee y) \vee z$
Absorption	$x \vee (x \wedge y) = x$
Absorption*	$x \wedge (x \vee y) = x$
Distributivity	$x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$
Distributivity*	$x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$
↑ Distributive lattice (without 0, 1, $\neg$) ↑	
Identity	$x \wedge 1 = x$
Identity*	$x \vee 0 = x$
Complementation	$x \wedge (\neg x) = 0$
Complementation*	$x \vee (\neg x) = 1$
↑ Boolean lattice (with 0, 1, $\neg$) ↑	

Theorem 4.1. *Consider an algebra $\mathcal{A} = (A, \mathcal{F}^{\mathcal{A}})$ with a unary operation $\square^{\mathcal{A}}$. Assume $\mathcal{A}$ satisfies laws $R_1, \ldots, R_n$ which imply that $\square$ has no fixed point: $\square(x) \neq x$ for all $x \in A$. Furthermore, assume that one of the laws R_i is the involution law given by*

$$\square(\square(x)) = x.$$

Then, there exists no algebra $\mathcal{B} = (B, \mathcal{F}^{\mathcal{B}})$ on the Euclidean space $B = \mathbb{R}^l$ such that $\square^{\mathcal{B}}$ is continuous and $R_1, \ldots, R_n$ are all satisfied by $\mathcal{B}$.

We provide a specific instantiation of the above theorem for our considered case of Boolean lattices, leveraging the fact that the complementation operation is unrealizable.

Corollary 4.2. *The Boolean lattice type $\mathcal{F}_{\text{Bool}}$ cannot be realized on $M = \mathbb{R}^l$ with continuous operations such that the Boolean lattice laws in Table 1 are satisfied.*

Proof sketch. The Boolean not operator $\neg$ satisfies $\neg(\neg a) = a$ for any a in the domain. Furthermore, Boolean lattice laws show that if there were a fixed point $b = \neg b$, we must have both $b = 0$ and $b = 1$; this is a contradiction and thus $\neg$ is an involution with no fixed points. Applying Theorem 4.1 concludes the proof. □

4.3. Relaxing to a distributive lattice

Section 4.2 shows that a Boolean lattice structure cannot be realized on $M = \mathbb{R}^l$. We relax our requirements to that of a distributive lattice, and present a structure known as a Riesz algebra that realizes $\mathcal{F}_{\text{Dist}}$ and satisfies all associated laws.

Definition 4.3. The *Riesz mirrored algebra* is the distributive lattice $\mathcal{M} = (M, \mathcal{F}^{\mathcal{M}}_{\text{Dist}})$ with operations given by

$$a \wedge^{\mathcal{M}} b = \min(a, b) \quad \text{and} \quad a \vee^{\mathcal{M}} b = \max(a, b)$$

on $M = \mathbb{R}^l$, where $\min$ and $\max$ are defined elementwise. This algebra satisfies the distributive lattice laws in Table 1.

Since our specific application concerns the distributed lattice of sets, we can equivalently take our operation symbols to be $\cap$ and $\cup$ in place of $\wedge$ and $\vee$, respectively. With this notation, the realization $\cap^{\mathcal{S}} : S \times S \to S$ is standard set intersection on $S = \mathcal{P}(\mathbb{R}^d)$, the realization $\cap^{\mathcal{M}} : M \times M \to M$ is elementwise maximum on the mirrored space $M = \mathbb{R}^l$, and $\cap^{\mathcal{L}} : L \times L \to L$ is the operation on $L = \mathbb{R}^l$ induced via (1). Analogous notational identifications also hold for $\cup$.

5. Experiments

This section details our experimental results on transporting structure from algebras of sets to latent embeddings. Following the infeasibility result and subsequent structural relaxation in Section 4, we seek to transport the distributive lattice defined by set intersection and set union, disregarding complementation. Our desired laws are listed in the upper section of Table 1, identifying $\wedge$ with $\cap$, and $\vee$ with $\cup$.

Our experiments explore the impact of different choices for mirrored algebra operations $\cap^{\mathcal{M}}$ and $\cup^{\mathcal{M}}$. Section 5.1 shows that operations that are well-aligned with source algebra laws outperform those that satisfy few laws, affirming Hypothesis 1. Section 5.2 shows that well-designed mirrored algebras are crucial for ensuring *self-consistency*: the property that equivalent terms produce the same prediction.

We now introduce the shared portions of the experimental setup, with further details deferred to Appendix B.

Candidate operations. Our distributive lattice of sets contains two binary operations: meet ($\cap$) and join ($\cup$). We must realize these on the mirrored space as binary vector operations $\cap^{\mathcal{M}}$ and $\cup^{\mathcal{M}}$. We restrict ourselves to closed-form operations that are well-conditioned (as opposed to elementwise division or exponentiation, for example). The list of candidate operations in Table 2 includes the Riesz algebra $\min$ and $\max$ operations, as well as the standard vector operations of addition, subtraction, and Hadamard product. For diversity, we include an operation that is commutative but not associative (scaled addition), associative but not commutative (matrix product), and neither (cyclic addition).

We define the function $\texttt{sq}\colon \mathbb{R}^l \to \mathbb{R}^{\sqrt{l}\times\sqrt{l}}$ to reshape a vector into a square matrix (assuming l is a square number), and $\texttt{roll}\colon \mathbb{R}^l \to \mathbb{R}^l$ to cycle vector elements by one index. We denote the set of all candidate operations by

$$\mathcal{C} = \{\min, \max, +, -, \odot, +_s, \times_{\text{mat}}, +_c\}.$$

Table 2. List of candidate operations on M.

Element min ($\min$)	$(a, b) \mapsto \min(a, b)$
Element max ($\max$)	$(a, b) \mapsto \max(a, b)$
Addition ($+$)	$(a, b) \mapsto a + b$
Subtraction ($-$)	$(a, b) \mapsto a - b$
Hadamard prod. ($\odot$)	$(a, b) \mapsto a \odot b$
Scaled addition ($+_s$)	$(a, b) \mapsto 2a + 2b$
Matrix prod. ($\times_{\text{mat}}$)	$(a, b) \mapsto \texttt{sq}^{-1}(\texttt{sq}(a) \cdot \texttt{sq}(b))$
Cyclic addition ($+_c$)	$(a, b) \mapsto \texttt{roll}(a) + b$

Dataset. To generate a synthetic random subset of $\mathbb{R}^d$ for $d = 2$, we first uniformly sample two random integers n_i, n_o from $\{1, 2, \ldots, 10\}$. We then restrict ourselves to the zero-centered square and sample n_i and n_o points from $[-1, 1]^2$ to yield $I = \{v^i_1, \ldots, v^i_{n_i}\}$ and $O = \{v^o_1, \ldots, v^o_{n_o}\}$. We then generate a set U from these points as follows:

$$U = \left\{u \in [-1, 1]^2 : \min_{v \in I} \|v - u\|_2 \leq \min_{v \in O} \|v - u\|_2\right\}.$$

We generate 10^4 such random sets with an 80% training, 10% validation, and 10% testing split. For each set, an INR is trained on evaluations of the set indicator function using a SIREN architecture (Sitzmann et al., 2020). An `inr2vec` architecture (De Luigi et al., 2023) is then trained over this dataset, resulting in: 1) an encoder $E\colon S \to L$ mapping a set (as represented by the raw weight matrices of an INR) to a latent embedding space $L = \mathbb{R}^l$ with $l = 1024$, and 2) a decoder $D\colon [-1, 1]^2 \times L \to \mathbb{R}$ that predicts whether a particular point is in the set associated with a latent.

With some abuse of notation, we let $E(U) \in L$ denote the embedding of the INR trained on a set $U \subseteq [-1, 1]^2$ as described above. We precompute and store latent embeddings for all INRs, after which the encoder is no longer required. Decoder weights are also fixed for our later experiments.

Parameterizations. A particular training run starts with a fixed choice of operations $\cap^{\mathcal{M}}, \cup^{\mathcal{M}}\colon M \times M \to M$ on the mirrored space (e.g., $\cap^{\mathcal{M}} = \min$ and $\cup^{\mathcal{M}} = \max$ for the Riesz mirrored algebra). The learned bijection $\varphi\colon L \to M$ is constructed as a modified NICE architecture (Dinh et al., 2015). At training time, φ is the only learned component. Importantly, φ induces latent space operations $\cap^{\mathcal{L}}, \cup^{\mathcal{L}}\colon L \times L \to L$ from $\cap^{\mathcal{M}}, \cup^{\mathcal{M}}$ via (1).

For reference, we also try to *directly parameterize* operations on the latent space as $\cap^{\mathcal{L}} = f_\cap$ and $\cup^{\mathcal{L}} = f_\cup$, with learned functions $f_\cap, f_\cup\colon L \times L \to L$. We compare two options for this parameterization. The first is simply constructing $f_\cap$ and $f_\cup$ as multilayer perceptrons on the vector concatenation of inputs (no law guarantees). The second involves parameterizing in a symmetric, commutativity-

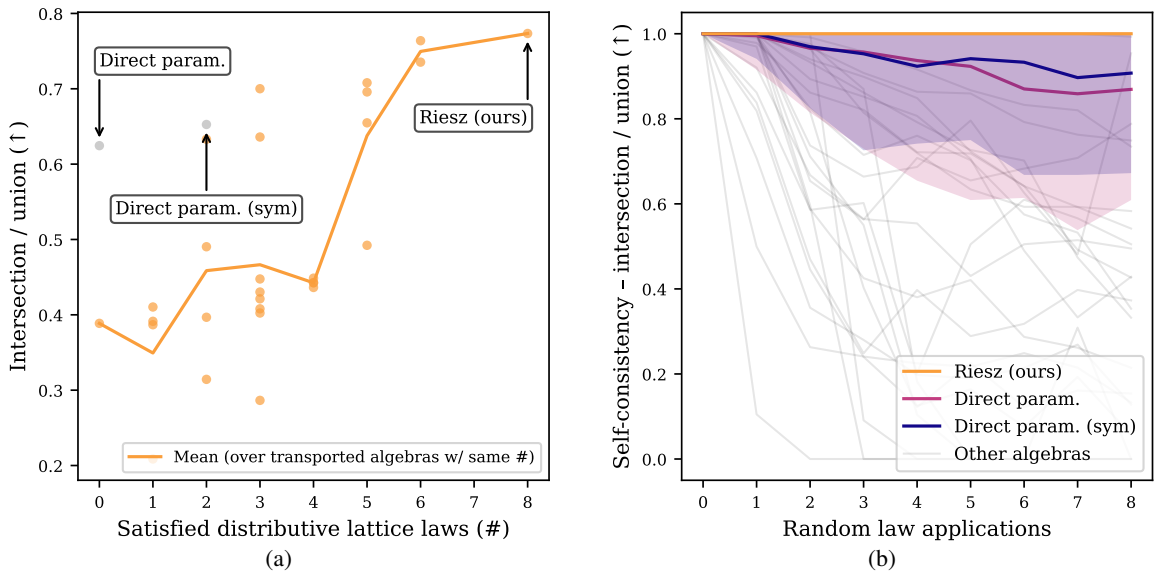

Figure 3. a) Learned operation performance vs. satisfaction of distributive lattice laws (solid line is mean). b) Self-consistency vs. number of random symbolic manipulations (i.e., law applications). Solid lines are medians, shaded areas capture 20th to 80th percentile ranges.

preserving manner via the form

$$f(z_1, z_2) = h\big(g(z_1) + g(z_2)\big),$$

where h and g are separate MLPs with compatible domains and codomains. We annotate this second parameterization using "sym" in our plots (see Zaheer et al. (2017)).

Loss and metrics. The training loss and evaluation metrics are computed over randomly constructed terms with a random number of starting symbols $\ell \in \{1, 2, \ldots, \ell_{\max}\}$ (in our experiments, $\ell_{\max} = 10$). We generate these by recursively combining random pairs of terms with either $\cap$ or $\cup$, starting with ℓ singleton terms (i.e., variables) and ending after $\ell - 1$ operations when only the final combined term remains.

For a particular such term $p(x_1, \ldots, x_\ell)$, we fetch ℓ sets $U_1, \ldots, U_\ell$ from data with corresponding precomputed `inr2vec` latent embeddings $z_1, \ldots, z_\ell$, recalling that $z_i = E(U_i) \in L$. We evaluate the ground truth set $U_{\text{true}} \subseteq [-1, 1]^2$ via the realized term value

$$U_{\text{true}} = p^{\mathcal{S}}(U_1, \ldots, U_\ell),$$

taking $\cap^{\mathcal{S}}$ and $\cup^{\mathcal{S}}$ to be standard set-theoretic intersection and union. We similarly evaluate the predicted latent

$$z_{\text{pred}} = p^{\mathcal{L}}(z_1, \ldots, z_\ell), \tag{3}$$

using $\cap^{\mathcal{L}}$ and $\cup^{\mathcal{L}}$, that are induced from $\cap^{\mathcal{M}}$ and $\cup^{\mathcal{M}}$ via (1). The predicted set is then given by

$$U_{\text{pred}} = \{u \in [-1, 1]^2 : D(u, z_{\text{pred}}) \geq 0\}. \tag{4}$$

All metrics are then approximated using uniformly sampled $u \in [-1, 1]^2$. Our loss is the expectation of the binary cross-entropy loss against the ground truth set indicator function

$$\text{Loss}(U_{\text{pred}}, U_{\text{true}}) = \mathbb{E}_u \left[\text{BCE}\big(D(u, z_{\text{pred}}), \mathbb{1}_{U_{\text{true}}}(u)\big)\right],$$

and our intersection over union (IoU) metric is written as

$$\text{IoU}(U_{\text{pred}}, U_{\text{true}}) = \frac{\mathbb{E}_u[\mathbb{1}_{U_{\text{true}} \cap U_{\text{pred}}}(u)]}{\mathbb{E}_u[\mathbb{1}_{U_{\text{true}} \cup U_{\text{pred}}}(u)]}.$$

The IoU score ranges from zero to one (perfect prediction).

5.1. Operation performance vs. structure choice

This experiment tests various candidate realizations of $\cap$ and $\cup$ on M, with the aim of evaluating whether satisfying distributed lattice laws induces superior performance. We consider all possible assignments $(\cap^{\mathcal{M}}, \cup^{\mathcal{M}}) \in \mathcal{C} \times \mathcal{C}$ with $\cap^{\mathcal{M}} \neq \cup^{\mathcal{M}}$, excluding flipped assignments (e.g., $(\max, \min)$ versus $(\min, \max)$) due to the exact symmetry of distributive lattice laws and our data generating process. This results in $\binom{|\mathcal{C}|}{2} = \binom{8}{2} = 28$ possible combinations. For each assignment $(\cap^{\mathcal{M}}, \cup^{\mathcal{M}})$, we determine which distributive lattice laws from Table 1 are satisfied using numerical testing. We provide some illustrative examples in Table 3, with the full list provided by Table 4 in the appendix.

Our results are depicted in Figure 3a. Each dot represents a particular choice of operations (i.e., a particular mirrored algebra). The x-axis groups together algebras which satisfy the same number of distributive lattice laws (# column in Table 4). The y-axis reports the mean IoU performance of a particular algebra, averaged over random terms.

Table 3. Selection of candidate operations on the mirrored space with the satisfied distributive lattice laws (fully reproduced in Table 4). Due to distributive lattice symmetries, we have two laws for each column (e.g., $a \cap^{\mathcal{M}} b = b \cap^{\mathcal{M}} a$ and $a \cup^{\mathcal{M}} b = b \cup^{\mathcal{M}} a$). The first row imposes a Riesz algebra structure. The second column counts how many laws are satisfied by a particular pair of operations.

Operations		#	Commutativity (*)	Associativity (*)	Absorption (*)	Distributivity (*)
$\cap^{\mathcal{M}} = \max$	$\cup^{\mathcal{M}} = \min$	8	✓✓	✓✓	✓✓	✓✓
$\cap^{\mathcal{M}} = \max$	$\cup^{\mathcal{M}} = \odot$	6	✓✓	✓✓	✗✓	✓✗
$\cap^{\mathcal{M}} = \min$	$\cup^{\mathcal{M}} = +$	6	✓✓	✓✓	✗✓	✓✗
$\cap^{\mathcal{M}} = \max$	$\cup^{\mathcal{M}} = +$	5	✓✓	✓✓	✗✗	✓✗
$\cap^{\mathcal{M}} = \min$	$\cup^{\mathcal{M}} = \odot$	5	✓✓	✓✓	✗✗	✓✗
$\cap^{\mathcal{M}} = \min$	$\cup^{\mathcal{M}} = +_s$	5	✓✓	✓✗	✗✓	✓✗
$\cap^{\mathcal{M}} = +$	$\cup^{\mathcal{M}} = \odot$	5	✓✓	✓✓	✗✗	✓✗
		⋮				
$\cap^{\mathcal{M}} = \times_{\text{mat}}$	$\cup^{\mathcal{M}} = +_c$	1	✗✗	✓✗	✗✗	✗✗
$\cap^{\mathcal{M}} = -$	$\cup^{\mathcal{M}} = +_c$	0	✗✗	✗✗	✗✗	✗✗

Figure 3a provides clear experimental support for Hypothesis 1: the accuracy of learned set operations is strongly tied to the number of satisfied source algebra laws. The Riesz algebra completely satisfies all 8 laws and achieves the best performance, while operations with few satisfied laws struggle. Despite significantly underperforming the Riesz algebra, the direct latent parameterizations surpass transported algebras with a similar number of satisfied laws, suggesting that the flexibility of their parameterization somewhat mitigates their lack of algebraic structure. Interestingly, algebras that only violate a few laws substantially outperform algebras that violate most or all; there is a notable increasing trend in performance. Thus even when not all laws can be satisfied, a reasonably well-aligned mirrored algebra can still provide substantial benefits.

5.2. Consistency under equivalent terms

This experiment adopts the same setting as above, but considers a different question: how *self-consistent* are the predictions of a model for terms that are distinct but equivalent with respect to $\mathcal{F}_{\text{Dist}}$? Naturally, we expect a good model to provide the same predicted set for $A \cap B$ and $B \cap A$.

Consider a random term $p(x_1, \ldots, x_\ell)$, with sampled latents $z_1, \ldots, z_\ell$ yielding a predicted set U_{pred} via (3) and (4). Instead of comparing U_{pred} to U_{true}, we generate a family of equivalent terms $q_i(x_1, \ldots, x_\ell)$ by randomly selecting laws and substituting their expressions into $p(x_1, \ldots, x_\ell)$ if such expressions are present in $p(x_1, \ldots, x_\ell)$. For each equivalent term, we compare the new predicted set V_{pred} (computed via (3) and (4) as before) with the original prediction U_{pred} and compute the corresponding IoU metric.

Figure 3b summarizes our results. The x-axis represents the number of law applications, and the y-axis represents the self-consistency IoU. The solid lines represent the median performance for each choice of candidate operations, with the shaded areas representing the direct parameterizations' 20-to-80th percentile ranges.

Our Riesz mirrored algebra is perfectly self-consistent, experimentally validating Proposition 3.1. While the median performance of the learned baselines degrades moderately as the terms diverge, the bottom quartile drops sharply with even just two random symbolic manipulations. Interestingly, the direct parameterizations have a higher self-consistency than most other algebras, despite satisfying only zero or two laws. This suggests that a flexible parameterization can learn the appropriate symmetries to some degree, although we note that the Riesz algebra is decidedly superior to both across all experiments.

6. Conclusion

Interesting mathematical objects generally carry additional algebraic structure, such as operations and laws. Machine learning methods often encode such objects (sets, functions, etc.) into latent embeddings for downstream tasks. This paper examines the possibility of learning latent space operations that provably satisfy the same structural laws as the source algebra of input data. We provide a general procedure for constructing *structural transport nets* to carry out such transport of structure, and we illustrate the method in a concrete case study of the algebra of sets. Experiment results support our key hypothesis: stronger alignment between latent space operations and source algebra laws improves the performance of learned operations. Exciting future research involves further developing the theory of realizable latent-space operations and exploring downstream applications of structural transport nets.

Impact Statement

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

References

Azizian, W. and Lelarge, M. Expressive power of invariant and equivariant graph neural networks. *arXiv preprint arXiv:2006.15646*, 2020.

Behrmann, J., Grathwohl, W., Chen, R. T., Duvenaud, D., and Jacobsen, J.-H. Invertible residual networks. In *International Conference on Machine Learning*, 2019.

Bronstein, M. M., Bruna, J., Cohen, T., and Veličković, P. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021.

Burris, S. and Sankappanavar, H. P. *A Course in Universal Algebra*, volume 78. Springer, 1981.

Chen, Z. and Zhang, H. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

Chiang, H.-T. L., Baxter, J. E., Sugaya, S., Yousefi, M. R., Faust, A., and Tapia, L. Fast deep swept volume estimator. *The International Journal of Robotics Research*, 40(10-11):1068–1086, 2021.

Cohen, T. and Welling, M. Group equivariant convolutional networks. In *International Conference on Machine Learning*, 2016.

Cohen, T. S., Geiger, M., and Weiler, M. A general theory of equivariant cnns on homogeneous spaces. *Advances in neural information processing systems*, 32, 2019.

De Luigi, L., Cardace, A., Spezialetti, R., Ramirez, P. Z., Salti, S., and Di Stefano, L. Deep learning on implicit neural representations of shapes. In *International Conference on Machine Learning*, 2023.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Dinh, L., Krueger, D., and Bengio, Y. NICE: Non-linear independent components estimation. *International Conference on Learning Representations (Workshop)*, 2015.

Fremlin, D. H. *Measure Theory*, volume 3. Torres Fremlin, 2002.

Ha, D., Dai, A., and Le, Q. HyperNetworks. In *International Conference on Learning Representations*, 2017.

He, Y.-H. and Kim, M. Learning algebraic structures: preliminary investigations. *International Journal of Data Science in the Mathematical Sciences*, 2023.

Jaworowski, J. On antipodal sets on the sphere and on continuous involutions. *Fundamenta Mathematicae*, 2 (43):241–254, 1956.

Kobyzev, I., Prince, S. J., and Brubaker, M. A. Normalizing flows: An introduction and review of current methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

Kondor, R. and Trivedi, S. On the generalization of equivariance and convolution in neural networks to the action of compact groups. In *International Conference on Machine Learning*, pp. 2747–2755. PMLR, 2018.

Mahjourian, R., Kim, J., Chai, Y., Tan, M., Sapp, B., and Anguelov, D. Occupancy flow fields for motion forecasting in autonomous driving. *IEEE Robotics and Automation Letters*, 7(2):5639–5646, 2022.

Maron, H., Ben-Hamu, H., Shamir, N., and Lipman, Y. Invariant and equivariant graph networks. *arXiv preprint arXiv:1812.09902*, 2018.

Martin-Maroto, F. and de Polavieja, G. G. Algebraic machine learning. *arXiv preprint arXiv:1803.05252*, 2018.

Meng, Y., Sun, D., Qiu, Z., Waez, M. T. B., and Fan, C. Learning density distribution of reachable states for autonomous systems. In *Conference on Robot Learning*, pp. 124–136. PMLR, 2022.

Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., and Geiger, A. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

Michaux, J., Chen, Q., Kwon, Y., and Vasudevan, R. Reachability-based trajectory design with neural implicit safety constraints. *arXiv preprint arXiv:2302.07352*, 2023.

Nichol, A., Dhariwal, P., Ramesh, A., Shyam, P., Mishkin, P., McGrew, B., Sutskever, I., and Chen, M. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.

Pan, J.-H., Hui, K.-H., Gao, X., Zhu, S., Liu, Y.-H., Heng, P.-A., and Fu, C.-W. Sdf-pack: Towards compact bin packing with signed-distance-field minimization. In *2023*

IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 10612–10619. IEEE, 2023.

Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S., and Lakshminarayanan, B. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 2021.

Park, J. J., Florence, P., Straub, J., Newcombe, R., and Lovegrove, S. DeepSDF: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

Sennesh, E., Xu, T., and Maruyama, Y. Computing with categories in machine learning. In *International Conference on Artificial General Intelligence*, 2023.

Shiebler, D., Gavranović, B., and Wilson, P. Category theory in machine learning. *arXiv preprint arXiv:2106.07032*, 2021.

Sitzmann, V., Martel, J., Bergman, A., Lindell, D., and Wetzstein, G. Implicit neural representations with periodic activation functions. In *Advances in Neural Information Processing Systems*, 2020.

Sohn, K., Lee, H., and Yan, X. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28, 2015.

Teshima, T., Ishikawa, I., Tojo, K., Oono, K., Ikeda, M., and Sugiyama, M. Coupling-based invertible neural networks are universal diffeomorphism approximators. *Advances in Neural Information Processing Systems*, 33:3362–3373, 2020.

Wang, L., Yang, N., Huang, X., Jiao, B., Yang, L., Jiang, D., Majumder, R., and Wei, F. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*, 2022.

Wang, Y., Li, Y., Driggs-Campbell, K., Fei-Fei, L., and Wu, J. Dynamic-resolution model learning for object pile manipulation. *arXiv preprint arXiv:2306.16700*, 2023.

Wechler, W. *Universal Algebra for Computer Scientists*, volume 25. Springer Science & Business Media, 2012.

Winkler, C., Worrall, D., Hoogeboom, E., and Welling, M. Learning likelihoods with conditional normalizing flows. *arXiv preprint arXiv:1912.00042*, 2019.

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

A. Proofs

A.1. Proofs for Section 3

Proposition 3.1. *Suppose that $L, M = \mathbb{R}^l$ and that $\varphi\colon L \to M$ is a bijection. Let $\mathcal{M} = (M, \mathcal{F}^{\mathcal{M}})$ be an algebra of type $\mathcal{F}$ and define the family $\mathcal{F}^{\mathcal{L}} \coloneqq \{f^{\mathcal{L}} : f \in \mathcal{F}\}$ of n-ary operations on L by (1). Then, φ is an isomorphism from the induced algebra $\mathcal{L} = (L, \mathcal{F}^{\mathcal{L}})$ to $\mathcal{M}$.*

Proof of Proposition 3.1. Let $f \in \mathcal{F}$, let $n = \operatorname{ar}(f)$, and consider the realization $f^{\mathcal{M}}$ on M and the realization $f^{\mathcal{L}}$ on L induced by (1). Let $z_1, \ldots, z_n \in L$. We have that

$$\begin{aligned}\varphi(f^{\mathcal{L}}(z_1, \ldots, z_n)) &= \varphi(\varphi^{-1}(f^{\mathcal{M}}(\varphi(z_1), \ldots, \varphi(z_n)))) \\ &= f^{\mathcal{M}}(\varphi(z_1), \ldots, \varphi(z_n))\end{aligned}$$

by construction of the operation $f^{\mathcal{L}}$. Hence, we see that φ is an isomorphism from the induced algebra $\mathcal{L}$ to the algebra $\mathcal{M}$. □

Theorem 3.2. *Consider a source algebra $\mathcal{S} = (S, \mathcal{F}^{\mathcal{S}})$ of type $\mathcal{F}$, and let $\mathcal{M} = (M, \mathcal{F}^{\mathcal{M}})$ be a mirrored space such that every law R satisfied by $\mathcal{S}$ is also satisfied by $\mathcal{M}$. Then, the induced latent algebra $\mathcal{L}$, defined by (1), also satisfies every such law R, for any bijection $\varphi\colon L \to M$.*

Proof of Theorem 3.2. Let $\varphi\colon L \to M$ be a bijection, and let $\mathcal{L}$ be the induced latent algebra defined by (1). Let R be a law that is satisfied by $\mathcal{S}$ (and hence satisfied by $\mathcal{M}$), given by

$$p(x_1, \ldots, x_n) = q(x_1, \ldots, x_n).$$

By Proposition 3.1, φ is an isomorphism from $\mathcal{L}$ to $\mathcal{M}$. Let $f \in \mathcal{F}$ be an arbitrary operation symbol. Then, by the properties of isomorphisms, it must be that

$$\varphi(f^{\mathcal{L}}(z_1, \ldots, z_n)) = f^{\mathcal{M}}(\varphi(z_1), \ldots, \varphi(z_n))$$

for all $z_1, \ldots, z_n \in L$. Thus, it holds that

$$\varphi(p^{\mathcal{L}}(z_1, \ldots, z_n)) = p^{\mathcal{M}}(\varphi(z_1), \ldots, \varphi(z_n))$$

for all $z_1, \ldots, z_n \in L$, and similarly,

$$\varphi(q^{\mathcal{L}}(z_1, \ldots, z_n)) = q^{\mathcal{M}}(\varphi(z_1), \ldots, \varphi(z_n))$$

for all such $z_1, \ldots, z_n$. Therefore, since $\mathcal{M}$ satisfies the law R, we conclude that

$$\varphi(p^{\mathcal{L}}(z_1, \ldots, z_n)) = \varphi(q^{\mathcal{L}}(z_1, \ldots, z_n))$$

for all $z_1, \ldots, z_n \in L$. Hence, by invertibility of φ, we also find that

$$p^{\mathcal{L}}(z_1, \ldots, z_n) = q^{\mathcal{L}}(z_1, \ldots, z_n)$$

for all $z_1, \ldots, z_n \in L$, and therefore $\mathcal{L}$ satisfies the law R. □

Proposition 3.3. *There exists a source algebra $\mathcal{S} = (S, \mathcal{F}^{\mathcal{S}})$ and a mirrored algebra $\mathcal{M} = (M, \mathcal{F}^{\mathcal{M}})$ with $M = \mathbb{R}^l$, both of the same type $\mathcal{F}$, such that $\mathcal{M}$ satisfies every law R that $\mathcal{S}$ satisfies, and, for all bijections $\varphi\colon L \to M$, there is no nontrivial homomorphism $\chi\colon S \to L$ when $L = \mathbb{R}^l$ is equipped with the algebra induced by $\mathcal{M}$ via (1).*

Proof of Proposition 3.3. We prove the claim by construction. Consider the source algebra $\mathcal{S} = (\mathbb{R}, \bullet)$, with a sole binary operation $\bullet$ defined by

$$s_1 \bullet s_2 = |s_1| s_2,$$

where, of course, $|s_1|$ represents the absolute value of the real number s_1, and $|s_1|s_2$ represents the usual product of the two real numbers $|s_1|$ and s_2. This source algebra is a semigroup, namely, $\mathbb{R}$ is closed under the binary operation $\bullet$, and $\bullet$ satisfies the associativity law, since

$$\begin{aligned} s_1 \bullet (s_2 \bullet s_3) &= |s_1|(s_2 \bullet s_3) \\ &= |s_1|(|s_2|s_3) \\ &= |s_1 s_2|s_3 \\ &= \big||s_1|s_2\big|s_3 \\ &= (|s_1|s_2) \bullet s_3 \\ &= (s_1 \bullet s_2) \bullet s_3 \end{aligned}$$

for all $s_1, s_2, s_3 \in \mathbb{R}$. Now, consider the mirrored algebra $\mathcal{M} = (\mathbb{R}, +)$, with $+$ being the standard addition operation on the real numbers. Obviously, $\mathcal{M}$ is also a semigroup, since $\mathbb{R}$ is closed under $+$, and $+$ is associative.

We now show that $\mathcal{M}$ satisfies every law R that $\mathcal{S}$ does. Let R be a law for type $\mathcal{F}$ defined by

$$R : p(x_1, \ldots, x_n) = q(x_1, \ldots, x_n)$$

for some arbitrary terms $p(x_1, \ldots, x_n), q(x_1, \ldots, x_n) \in T_{\mathcal{F}}(X)$. Suppose that $\mathcal{S}$ satisfies the law R. Then,

$$p^{\mathcal{S}}(s_1, \ldots, s_n) = q^{\mathcal{S}}(s_1, \ldots, s_n)$$

for all $s_i \in \mathbb{R}$. Since the associative binary operation $\bullet$ is the only operation in $\mathcal{F}^{\mathcal{S}}$, it must be that the term function $p^{\mathcal{S}}$ is given by some repeated application of $\bullet$:

$$p^{\mathcal{S}}(s_1, \ldots, s_n) = s_{i_1} \bullet s_{i_2} \bullet \cdots \bullet s_{i_{m_p}}$$

for some $m_p \in \mathbb{N}$ and some tuple $(i_1, \ldots, i_{m_p}) \in \{1, \ldots, n\}^{m_p}$. Similarly,

$$q^{\mathcal{S}}(s_1, \ldots, s_n) = s_{j_1} \bullet s_{j_2} \bullet \cdots \bullet s_{j_{m_q}}$$

for some $m_q \in \mathbb{N}$ and some tuple $(j_1, \ldots, j_{m_p}) \in \{1, \ldots, n\}^{m_p}$. Thus,

$$|s_{i_1} \cdots s_{i_{m_p}-1}|s_{i_{m_p}} = |s_{j_1} \cdots s_{j_{m_q}-1}|s_{j_{m_q}}. \tag{5}$$

If $m_p > m_q$, then there exists some factor s_i appearing in the product $|s_{i_1} \cdots s_{i_{m_p}-1}|s_{i_{m_p}}$ at least once more than it does in the product $|s_{j_1} \cdots s_{j_{m_q}-1}|s_{j_{m_q}}$. Thus, if the equality (5) holds for some $s_1, \ldots, s_n \in S$, doubling this particular value s_i would result in the law being violated, as the left-hand side of (5) would have an extra factor of 2 that the right-hand side would not. This implies that $m_p \leq m_q$. Analogous reasoning shows that $m_q \leq m_p$, and hence it must be that $m_p = m_q$; the same number of factors appear in the left-hand and right-hand sides of the law's realizations. Furthermore, the same reasoning goes to show that the left-hand and right-hand products in (5) actually must contain exactly the same factors with the same multiplicity (albeit in possibly different order), i.e., the ordered tuple $(s_{i_1}, \ldots, s_{i_{m_p}})$ of real numbers is some permutation of the ordered tuple $(s_{j_1}, \ldots, s_{j_{m_q}})$. Hence, it must be the case that

$$s_{i_1} + s_{i_2} + \cdots + s_{i_{m_p}} = s_{j_1} + s_{j_2} + \cdots + s_{j_{m_q}},$$

implying that

$$p^{\mathcal{M}}(s_1, \ldots, s_n) = p^{\mathcal{M}}(s_1, \ldots, s_n).$$

That is, the law is satisfied by $\mathcal{M}$ as well. Since R was arbitrarily chosen, we conclude that indeed $\mathcal{M}$ satisfies every law R that $\mathcal{S}$ does.

Now, let $\psi \colon S \to M$ be a homomorphism from $\mathcal{S}$ to $\mathcal{M}$. Then, it holds that

$$\psi(s_1 \bullet s_2) = \psi(s_1) + \psi(s_2)$$

for all $s_1, s_2 \in S = \mathbb{R}$. Therefore, for $s_1 = 0$ and $s_2 = s$ with $s \in S$ arbitrary, we conclude that

$$\psi(0) = \psi(|0|s) = \psi(0 \bullet s) = \psi(0) + \psi(s),$$

and hence

$$\psi(0) + \psi(s) = \psi(0) + \psi(t)$$

for all $s, t \in S$. However, since $+$ is the standard addition operation on $\mathbb{R}$, this is only possible if

$$\psi(s) = \psi(t)$$

for all $s, t \in S$, meaning the homomorphism ψ must be the trivial mapping $\psi\colon s \mapsto C$ with $C = \psi(0)$.

Now, let $\varphi\colon L \to M$ be an arbitrary bijection. Equip L with the algebra $\mathcal{L}$ induced by $\mathcal{M}$, as defined by (1). Let $\chi\colon S \to L$ be a homomorphism from $\mathcal{S}$ to $\mathcal{L}$. Then, since φ is an isomorphism from $\mathcal{L}$ to $\mathcal{M}$ (per Proposition 3.1), the composition $\varphi \circ \chi$ is a homomorphism from $\mathcal{S}$ to $\mathcal{M}$. Therefore, by our analysis above, $\varphi \circ \chi$ must be a trivial homomorphism given by $\varphi \circ \chi\colon s \mapsto C$ with $C = \varphi \circ \chi(0)$. Hence, for all $s \in S$, we conclude that

$$\chi(s) = \varphi^{-1}(C),$$

implying that χ must be a trivial homomorphism from $\mathcal{S}$ to $\mathcal{L}$. This concludes the proof. □

Proposition 3.4. *Consider a source algebra $\mathcal{S} = (S, \mathcal{F}^{\mathcal{S}})$ of type $\mathcal{F}$, the latent space $L = \mathbb{R}^l$, and an arbitrary encoder $E\colon S \to L$. If E is bijective and there exists a mirrored algebra $\mathcal{M} = (M, \mathcal{F}^{\mathcal{M}})$ with $M = \mathbb{R}^l$ and an isomorphism $\psi\colon S \to M$, then there exists a bijection $\varphi\colon L \to M$ such that $\varphi \circ E$ equals the isomorphism ψ.*

Proof of Proposition 3.4. Suppose that E is bijective and that there exists a mirrored algebra $\mathcal{M} = (M, \mathcal{F}^{\mathcal{M}})$ with $M = \mathbb{R}^l$ and an isomorphism $\psi\colon S \to M$. Define $\varphi\colon L \to M$ by $\varphi(z) = \psi \circ E^{-1}(z)$, which is well-defined since E is bijective. Then, it holds that

$$\varphi \circ E(s) = \psi(E^{-1}(E(s))) = \psi(s)$$

for all $s \in S$, which proves the result. □

A.2. Proofs for Section 4

Continuity of complementation. We briefly explain why we only consider continuous operations on the mirrored space to represent the complementation operation $\neg$ in the Boolean lattice type $\mathcal{F}_{\text{Bool}}$. Consider the complementation operation $^c : \mathcal{P}(\mathbb{R}^d) \to \mathcal{P}(\mathbb{R}^d)$. As there is no ambient topology on $\mathcal{P}(\mathbb{R}^d)$, the continuity of ^{c} is not well-defined in this context. However, a natural topology arises by passing first to the Borel sets Σ on $\mathbb{R}^d$ and then quotienting by the null ideal $\Sigma \cap \mathcal{N}$, with set intersection, union, and complementation inherited in the natural way. Following Fremlin (2002, 323A.iii.e), this allows us to define a topology via the *symmetric difference metric*:

$$d(A, B) \coloneqq \mu(A \triangle B) \text{ for all } A, B \in \Sigma/\Sigma \cap \mathcal{N},$$

where μ is inherited naturally on the quotiented space from the Borel measure on $\mathbb{R}^d$. Since the symmetric difference between A and B satisfies $A \triangle B = (A^c) \triangle (B^c)$, it is immediate that the inherited complementation operation is continuous with respect to this topology.

Negative result. Before proving our main result, we first present a key result from the algebraic topology literature.

Proposition A.1. *Any continuous involution on $\mathbb{R}^n$ has a fixed point.*

Proof of Proposition A.1. This is an easy application of Theorem 9 in Jaworowski (1956). Namely, $\mathbb{R}^n$ is a separable metric space, and it is acyclic because it is contractible. □

Lemma A.2. *Consider a Boolean lattice $\mathcal{A} = (A, \wedge, \vee, \neg, 0, 1)$ of type $\mathcal{F}_{\text{Bool}}$ and its associated laws in Table 1. Then, it holds that $\neg$ is an involution with no fixed points.*

Proof of Lemma A.2. Clearly, the Boolean lattice laws in Table 1 imply that $\neg(\neg a) = a$ for all $a \in A$, and thus $\neg$ is an involution. Now assume for the sake of contradiction that $\neg$ has a fixed point $b \in A$, so that $\neg b = b$. Then, by the Boolean lattice laws,

$$(\neg b) \wedge b = 0 \implies b \wedge b = 0 \implies b = 0,$$

and, similarly,

$$(\neg b) \vee b = 1 \implies b \vee b = 1 \implies b = 1.$$

This is a contradiction. □

We are now ready to prove a general negative result.

Theorem 4.1. *Consider an algebra $\mathcal{A} = (A, \mathcal{F}^{\mathcal{A}})$ with a unary operation $\Box^{\mathcal{A}}$. Assume $\mathcal{A}$ satisfies laws $R_1, \ldots, R_n$ which imply that $\Box$ has no fixed point: $\Box(x) \neq x$ for all $x \in A$. Furthermore, assume that one of the laws R_i is the involution law given by*

$$\Box(\Box(x)) = x.$$

Then, there exists no algebra $\mathcal{B} = (B, \mathcal{F}^{\mathcal{B}})$ on the Euclidean space $B = \mathbb{R}^l$ such that $\Box^{\mathcal{B}}$ is continuous and $R_1, \ldots, R_n$ are all satisfied by $\mathcal{B}$.

Proof of Theorem 4.1. Suppose for the sake of contradiction that there exists an algebra $\mathcal{B} = (B, \mathcal{F}^{\mathcal{B}})$ on $B = \mathbb{R}^l$ such that $\Box^{\mathcal{B}}$ is continuous and the laws $R_1, \ldots, R_n$ are all satisfied by $\mathcal{B}$. Then, by assumption it must be the case that $\Box^{\mathcal{B}}(b) \neq b$ for all $b \in B$. However, since $\Box^{\mathcal{B}}$ is a continuous involution on $B = \mathbb{R}^l$, by Proposition A.1, $\Box^{\mathcal{B}}$ has a fixed point, i.e., $\Box^{\mathcal{B}}(b) = b$ for some $b \in B$. This is a contradiction, and hence the result is proven. □

Corollary 4.2. *The Boolean lattice type $\mathcal{F}_{\text{Bool}}$ cannot be realized on $M = \mathbb{R}^l$ with continuous operations such that the Boolean lattice laws in Table 1 are satisfied.*

Proof of Corollary 4.2. This follows directly from Lemma A.2 and Theorem 4.1. □

B. Experiments

This appendix describes our experimental setup and provides additional figures conveying our results.

Reproducing these experiments takes approximately 5 GPU-days on an RTX A6000 GPU with an i7 core CPU. Our codebase was developed against PyTorch 2.1.2.

B.1. Hyperparameters and metrics

This section details the training hyperparameters and model architectures for reproducing our results. When training any model, we later reload model weights from the checkpoint with the best validation loss. All of our reported metrics are evaluated against a separate testing set.

Implicit neural representations. We train each INR using binary cross-entropy loss over the provided shape indicator function (Section 5) sampled at 5000 points in $[-1, 1]^2$. Our INR function is a standard sinusoidal activation SIREN network, with 128 hidden neurons, 3 layers, and $\omega_0 = 30$. Training uses the Adam optimizer with a learning rate of 0.01 and batch size of 10^3 for 10 epochs. These hyperparameters were chosen to enable fast convergence (on the order of seconds), as this training process is repeated 10^4 times in total. All INRs were trained using the same weight initialization for easier downstream embedding (De Luigi et al., 2023).

Latent embeddings (`inr2vec`). After producing the dataset of INRs, we train one latent embedder over the training split. We reuse the architecture of De Luigi et al. (2023) to train a model which takes in all MLP weights for the input INR for completeness (as opposed to just the hidden layer weights). Our latent embedding dimension is 1024–all other architecture hyperparameters are as in De Luigi et al. (2023). We use the Adam optimizer for 50 epochs with a learning rate of 10^{-4}, weight decay of 10^{-3}, and batch size of 16.

Structural transport nets. When learning a structural transport net, we have a fixed decoder D and fixed mirrored algebra $\mathcal{M}$. The only learned parameters are those defining the map $\varphi : \mathbb{R}^l \to \mathbb{R}^l$, which must be a bijection in order to achieve transport of structure and the isomorphism $\mathcal{L} \cong \mathcal{M}$. We parameterize φ as a series of additive coupling layers to enable easy differentiation through both the forward and inverse maps. We reuse the seminal NICE architecture, reducing the number of additive coupling layers to 2 and the number of nonlinear layers to 3 for efficiency (Dinh et al., 2015). Since our application requires differentiation through the function inverse, other architectures which rely on solving fixed-point iterations to compute inverses are not considered (Behrmann et al., 2019). Training runs for 10 epochs using Adam with a learning rate of 10^{-3}.

Direct latent space-parameterized baselines. Latent space-parameterized baselines are instantiated as functions $f_\cap, f_\cup \colon \mathbb{R}^l \times \mathbb{R}^l \to \mathbb{R}^l$. As described in Section 5, we compare two parameterizations:

1. Let each f be realized by a standard multilayer perceptron (MLP) with ReLU nonlinearities, operating on the vector concatenation of the inputs.
2. Let each f take the form
$$f(z_1, z_2) = h(g(z_1) + g(z_2)),$$
with $g \colon \mathbb{R}^l \to \mathbb{R}^{256}$ and $h \colon \mathbb{R}^{256} \to \mathbb{R}^l$ MLPs.

Notably, the second option satisfies commutativity via commutativity of the operation $+$, and thus satisfies two of the distributive lattice laws. We perform a hyperparameter sweep for each parameterization over the learning rates $\{10^{-4}, 10^{-3}, 10^{-2}\}$, layer counts $\{2, 3, 4\}$, and hidden dimension $\{64, 128, 256, 512\}$; the final optimal hyperparameters set the learning rate to 10^{-4}, layer count to 2, and hidden dimension to 256.

Computed metrics. We discuss a few details of our reported metrics. While the IoU metric is standard, it can occasionally be undefined if the union volume is zero; we exclude these datapoints from our computed statistics. We also tested an accuracy metric
$$\mathrm{Acc}(U_{\text{pred}}, U_{\text{true}}) = \mathbb{E}_u \left[\mathbb{1}_{U_{\text{pred}}}(u) \stackrel{?}{=} \mathbb{1}_{U_{\text{true}}}(u) \right],$$

where $\stackrel{?}{=}$ outputs 1 for equality and 0 otherwise. The results were qualitatively similar to that of the IoU metric so we omit them for simplicity.

B.2. Procedures

Random term generation. Section 3 and Section 5 describe a procedure for generating random terms of a particular type. Algorithm 2 provides more detailed pseudocode for the set distributive lattice type $\mathcal{F}_{\text{Dist}}$. We also work through an example for $\ell = 5$ starting symbols.

$$
\begin{array}{ccccccccccc}
v & & w & & x & & y & & z \\
v & & w & & x & & (y & \vee & z) \\
v & & (w & \wedge & x) & & (y & \vee & z) \\
(v & \vee & (w & \wedge & x)) & & (y & \vee & z) \\
(v & \vee & (w & \wedge & x)) & \wedge & (y & \vee & z)
\end{array}
$$

The final line is the finished random term.

Equivalent term generation. The experiment in Section 5.2 outlines a scheme for randomly generating a term $q(x_1, \ldots, x_\ell)$ which is equivalent to some original term $p(x_1, \ldots, x_\ell)$. At a high level, this works by successively substituting terms from distributive lattice laws. Algorithm 3 provides informal pseudocode for this procedure, and rests on the idea of applying laws to manipulate terms; a formal treatment of this is outside the scope of our paper, and we will explain our ideas informally.

Whether a law *applies* to a particular expression and how a law is then *applied* to that expression depends on which law is considered. We provide a concrete example for the distributive laws, with other laws following similarly.

Consider the distributive laws

$$x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z) \quad \text{and} \quad x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z). \tag{6}$$

Let X, Y, and Z be unspecified terms—either single variables, or more complicated compound terms—and consider a particular term $\hat{p}(X, Y, Z)$. Then we say that the distributive law *applies* to $\hat{p}(X, Y, Z)$ if $\hat{p}(X, Y, Z)$ is of one of the following forms:

$$
\begin{array}{llll}
X \vee (Y \wedge Z) & \text{or} & (X \vee Y) \wedge (X \vee Z) & \text{or} \\
X \wedge (Y \vee Z) & \text{or} & (X \wedge Y) \vee (X \wedge Z). &
\end{array}
\tag{7}
$$

If this is the case, we say that the distributive law is then *applied* to $\hat{p}(X, Y, Z)$ by applying the appropriate symbolic substitution to $\hat{p}(X, Y, Z)$ from (6). If multiple forms from (7) are appropriate, one is chosen at random.

Algorithm 3 essentially randomly applies a certain number of these transformations to random subterms of a particular starting term, yielding a chain of equivalent terms. We provide an example of such a chain here, starting from the operational polynomial $p(x, y, z) = x \wedge (y \vee z)$ and proceeding for three steps:

$$
\begin{aligned}
x \wedge (y \vee z) &= x \wedge (z \vee y) && \text{(commutativity)} \\
&= (x \wedge z) \vee (x \wedge y) && \text{(distributivity)} \\
&= ((x \wedge (x \vee y)) \wedge z) \vee (x \wedge y) && \text{(absorption)} \\
&=: q(x, y, z).
\end{aligned}
$$

In our self-consistency experiments, we then compare the realizations of our learned operations on the final term $q(x, y, z)$ against those of the starting term $p(x, y, z)$.

Algorithm 2 Random term generation

In: Number of symbols ℓ
Out: Term $p(x_1, \ldots, x_\ell)$
$\hat{P} \leftarrow (x_1, \ldots, x_\ell)$ `# Initialize list of terms`
while length$(\hat{P}) > 1$ **do**
  $i \leftarrow$ random integer in $\{1, \ldots,$ length$(\hat{P})\}$
  $\hat{p}_1 \leftarrow$ pop$(\hat{P}, i)$ `# Extract ith term and remove from list`
  $j \leftarrow$ random integer in $\{1, \ldots,$ length$(\hat{P})\}$
  $\hat{p}_2 \leftarrow$ pop$(\hat{P}, j)$ `# Extract jth term and remove from list`
  sym $\leftarrow$ random operation symbol in $\{\cap, \cup\}$ `# Select operation symbol`
  $\hat{q} \leftarrow \hat{p}_1$ sym $\hat{p}_2$ `# Apply operation symbol`
  append($\hat{P}$, $\hat{q}$) `# Append new term to list`
end
return $\hat{P}[1]$ `# Return remaining term in list`

Algorithm 3 Equivalent term generation

In: Starting term $p(x_1, \ldots, x_\ell)$, number of law applications J
Out: Equivalent term $q(x_1, \ldots, x_\ell)$
for $\_ \leftarrow 1$ **to** J **do**
  Recursively extract all subterms of p into a collection (tree search):
  $\hat{P} \leftarrow (\hat{p}_1, \ldots, \hat{p}_n)$ `# Specific subterms in p`
  $\hat{I} \leftarrow (\hat{i}_1, \ldots, \hat{i}_n)$ `# Indices for each p̂ in p`
  laws $\leftarrow$ ("associativity", "commutativity", "absorption", "distributivity")
  Randomly shuffle $\hat{P}$ and $\hat{I}$ (with the same shuffle)
  Randomly shuffle laws
  for $j \leftarrow 1$ **to** length(laws) **do**
    law $\leftarrow$ laws$[j]$
    for $k \leftarrow 1$ **to** length$(\hat{P})$ **do**
      $\hat{p}, \hat{i} \leftarrow \hat{P}[k], \hat{I}[k]$
      if law *applies to* $\hat{p}$ **then**
        $\hat{q} \leftarrow$ law applied to $\hat{p}$
        $p \leftarrow \hat{q}$ substituted for $\hat{p}$ at $\hat{i}$ in $\hat{P}$
      end
    end
  end
end
return $\hat{P}$

B.3. Additional visualizations

This section provides additional tables and graphics for our experiments.

Table of operations. Table 4 lists all combinations of candidate operations that we trained. For each combination, we numerically tested whether the 8 distributive lattice laws in Table 1 are satisfied. Summing the number of satisfied laws for a particular pair of combination yields the count (#) column.

The symmetric directly parameterized baseline satisfies two laws (commutativity in both directions), while the naive MLP satisfies no laws. These are not listed in the table.

Table 4. Candidate operations on the mirrored space with the satisfied distributive lattice laws. Due to distributive lattice symmetries, we have two laws for each column (e.g., $a \cap^{\mathcal{M}} b = b \cap^{\mathcal{M}} a$ and $a \cup^{\mathcal{M}} b = b \cup^{\mathcal{M}} a$). The first row imposes a Riesz algebra structure on the mirrored space. The second column counts how many laws are satisfied by a particular pair of candidate operations.

Operations		#	Commutativity (*)	Associativity (*)	Absorption (*)	Distributivity (*)
$\cap^{\mathcal{M}} = \max$	$\cup^{\mathcal{M}} = \min$	8	✓✓	✓✓	✓✓	✓✓
$\cap^{\mathcal{M}} = \max$	$\cup^{\mathcal{M}} = \odot$	6	✓✓	✓✓	✗✓	✓✗
$\cap^{\mathcal{M}} = \min$	$\cup^{\mathcal{M}} = +$	6	✓✓	✓✓	✗✓	✓✗
$\cap^{\mathcal{M}} = \max$	$\cup^{\mathcal{M}} = +$	5	✓✓	✓✓	✗✗	✓✗
$\cap^{\mathcal{M}} = \min$	$\cup^{\mathcal{M}} = \odot$	5	✓✓	✓✓	✗✗	✓✗
$\cap^{\mathcal{M}} = \min$	$\cup^{\mathcal{M}} = +_s$	5	✓✓	✓✗	✗✓	✓✗
$\cap^{\mathcal{M}} = +$	$\cup^{\mathcal{M}} = \odot$	5	✓✓	✓✓	✗✗	✓✗
$\cap^{\mathcal{M}} = \max$	$\cup^{\mathcal{M}} = +_s$	4	✓✓	✓✗	✗✗	✓✗
$\cap^{\mathcal{M}} = \min$	$\cup^{\mathcal{M}} = \times_{\text{mat}}$	4	✓✗	✓✓	✗✓	✗✗
$\cap^{\mathcal{M}} = +$	$\cup^{\mathcal{M}} = \times_{\text{mat}}$	4	✓✗	✓✓	✗✗	✓✗
$\cap^{\mathcal{M}} = \odot$	$\cup^{\mathcal{M}} = +_s$	4	✓✓	✓✗	✗✗	✗✓
$\cap^{\mathcal{M}} = \max$	$\cup^{\mathcal{M}} = -$	3	✓✗	✓✗	✗✓	✗✗
$\cap^{\mathcal{M}} = \max$	$\cup^{\mathcal{M}} = \times_{\text{mat}}$	3	✓✗	✓✓	✗✗	✗✗
$\cap^{\mathcal{M}} = \max$	$\cup^{\mathcal{M}} = +_c$	3	✓✗	✓✗	✗✗	✓✗
$\cap^{\mathcal{M}} = \min$	$\cup^{\mathcal{M}} = +_c$	3	✓✗	✓✗	✗✗	✓✗
$\cap^{\mathcal{M}} = +$	$\cup^{\mathcal{M}} = +_s$	3	✓✓	✓✗	✗✗	✗✗
$\cap^{\mathcal{M}} = \odot$	$\cup^{\mathcal{M}} = \times_{\text{mat}}$	3	✓✗	✓✓	✗✗	✗✗
$\cap^{\mathcal{M}} = +_s$	$\cup^{\mathcal{M}} = \times_{\text{mat}}$	3	✓✗	✗✓	✗✗	✓✗
$\cap^{\mathcal{M}} = \min$	$\cup^{\mathcal{M}} = -$	2	✓✗	✓✗	✗✗	✗✗
$\cap^{\mathcal{M}} = +$	$\cup^{\mathcal{M}} = -$	2	✓✗	✓✗	✗✗	✗✗
$\cap^{\mathcal{M}} = +$	$\cup^{\mathcal{M}} = +_c$	2	✓✗	✓✗	✗✗	✗✗
$\cap^{\mathcal{M}} = -$	$\cup^{\mathcal{M}} = \odot$	2	✗✓	✗✓	✗✗	✗✗
$\cap^{\mathcal{M}} = \odot$	$\cup^{\mathcal{M}} = +_c$	2	✓✗	✓✗	✗✗	✗✗
$\cap^{\mathcal{M}} = -$	$\cup^{\mathcal{M}} = +_s$	1	✗✓	✗✗	✗✗	✗✗
$\cap^{\mathcal{M}} = -$	$\cup^{\mathcal{M}} = \times_{\text{mat}}$	1	✗✗	✗✓	✗✗	✗✗
$\cap^{\mathcal{M}} = +_s$	$\cup^{\mathcal{M}} = +_c$	1	✓✗	✗✗	✗✗	✗✗
$\cap^{\mathcal{M}} = \times_{\text{mat}}$	$\cup^{\mathcal{M}} = +_c$	1	✗✗	✓✗	✗✗	✗✗
$\cap^{\mathcal{M}} = -$	$\cup^{\mathcal{M}} = +_c$	0	✗✗	✗✗	✗✗	✗✗

Operation performance and self-consistency scatter plots. We supplement our plots in Figure 3 with two in-depth scatter plots which elucidate interesting details in the data.

Figure 4a presents the same Section 5.1 experimental data as in Figure 3a. Namely, Figure 4a teases apart the influence of the length ℓ of the random symbol term, which is simply averaged in Figure 3a. The IoU performance metric, originally on the y-axis of Figure 3a, is now visualized in the color bar. The y-axis of Figure 4a starts with two-symbol terms (either $x \cup y$ or $y \cup x$) up to terms with 10 symbols. Each "column" in Figure 4a corresponds to one specific choice of algebra operations, which is represented as just a dot in Figure 3a. We annotate a few explicitly in both plots. These algebras are then grouped by how many distributive lattice laws they satisfy, which can be found in Table 4.

Figure 4a shows a moderate performance degradation as the length of the random term increases. This is less pronounced for methods with a low IoU to begin with, but can be more easily seen with better-performing methods, especially in the algebras with 5 and 6 satisfied laws. These algebras perform similarly to the Riesz algebra for terms with only a few symbols; however, as the length of the term increases the Riesz algebra emerges as a favorite. We believe this is attributable to the fact that the Riesz algebra exclusively satisfies all the desired distributive lattice laws.

Figure 4b similarly corresponds to Figure 3b and the experiment in Section 5.2. Each column in Figure 4b is a particular algebra, corresponding to one line in the line plot of Figure 3b. The color bar encodes the same self-consistency metric discussed in Section 5.2 and plotted on the y-axis of Figure 3b. However, the y-axis of Figure 4b corresponds to the x-axis of Figure 3b, and the x-axis of Figure 4b simply serves to arrange the different algebras in order with their alignment to the source algebra.

Figure 4b shows that all methods besides the Riesz algebra struggle to maintain self-consistency for equivalent terms. Naturally, for unchanged terms (0 on the y-axis) all methods are deterministic and hence self-consistent. The Riesz algebra is also perfectly self-consistent for all equivalent terms, as a direct consequence of Theorem 3.2. All other algebras degrade significantly as the equivalent terms become more complex. This stems from the fact that, unlike the Riesz algebra, all other operations must implicitly learn the underlying source algebra laws. This is always an inexact process and produces compounding errors for more complex terms.

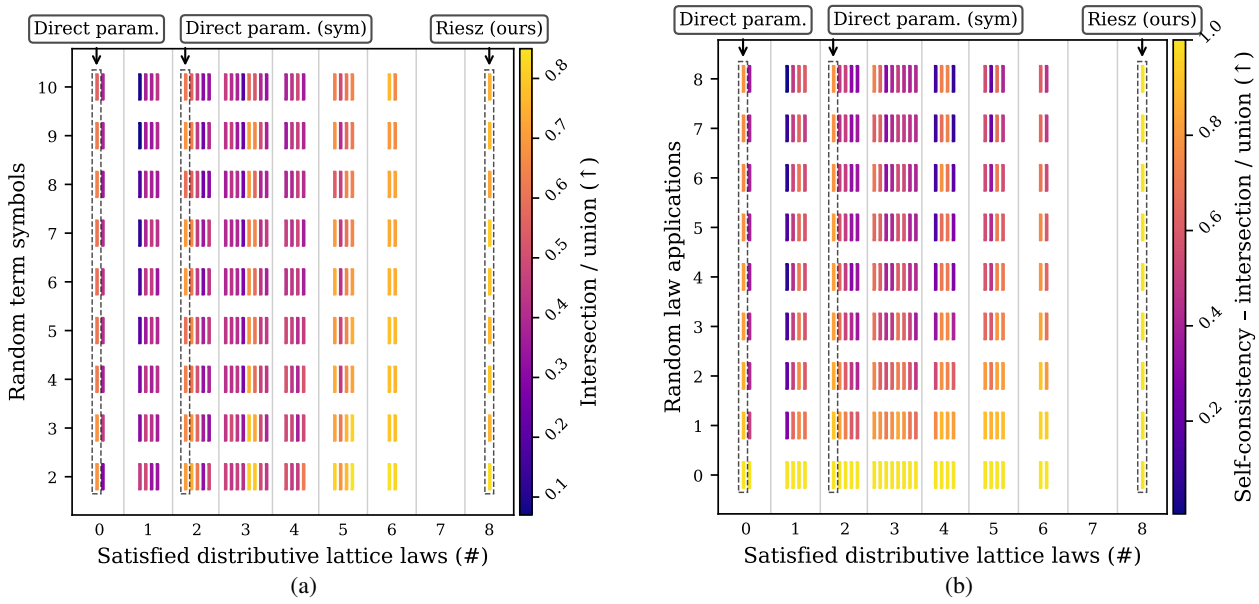

Figure 4. a) Average learned operation performance, plotted on the color bar, against number of satisfied distributive lattice laws and random term length. b) Average self-consistency, plotted on the color bar, against number of satisfied distributive lattice laws and number of random law applications.

Breakdown of specific laws. This series of plots breaks down the Section 5.1 data, plotted in Figure 3a and Figure 4a, at a more granular level. Instead of only considering the aggregate number of laws satisfied, Figure 5 analyzes the satisfaction of specific laws and its impact on learned operation performance.

Each point in any of the Figure 5 subplots represents a single algebra, with its color encoding the mean operation performance averaged over different term lengths, as in the y-axis of Figure 3a. For two particular distributive lattice laws, we group the algebras by the degree to which they satisfy each of the laws: this can include both of the forms in Table 1 (✓✓), just one or the other (✓✗/ ✗✓), or neither (✗✗). As with the other plots, we annotate our best-performing structural transport net and the two direct latent parameterization baselines.

Figure 5 broadly shows that satisfaction of any laws is positively related to learned operation performance. Specifically, moving from left-to-right and from bottom-to-top generally results in an increased average IoU score. However, conditioned on a particular law being satisfied, certain other laws seem to be less important. We enumerate a few such observations here:

1. Conditioned on the level of distributivity, increasing absorption seems to yield minimal benefits (Figure 5a).
2. Conditioned on the level of commutativity, increasing absorption seems to yield minimal benefits (Figure 5d).
3. Completely satisfying both commutativity and associativity is highly indicative of strong performance, but only partial satisfaction of either results in substantial degradation (Figure 5e).

Certain combinations of law satisfaction were not covered by our set of candidate operations. For example, no operations in $\mathcal{C} \times \mathcal{C}$ resulted in an algebra which was completely distributive and not at all absorptive Figure 5a. Future work can consider more sophisticated methods for constructing a set of candidate operations $\mathcal{C}$ which results in complete coverage of all possible law satisfactions.

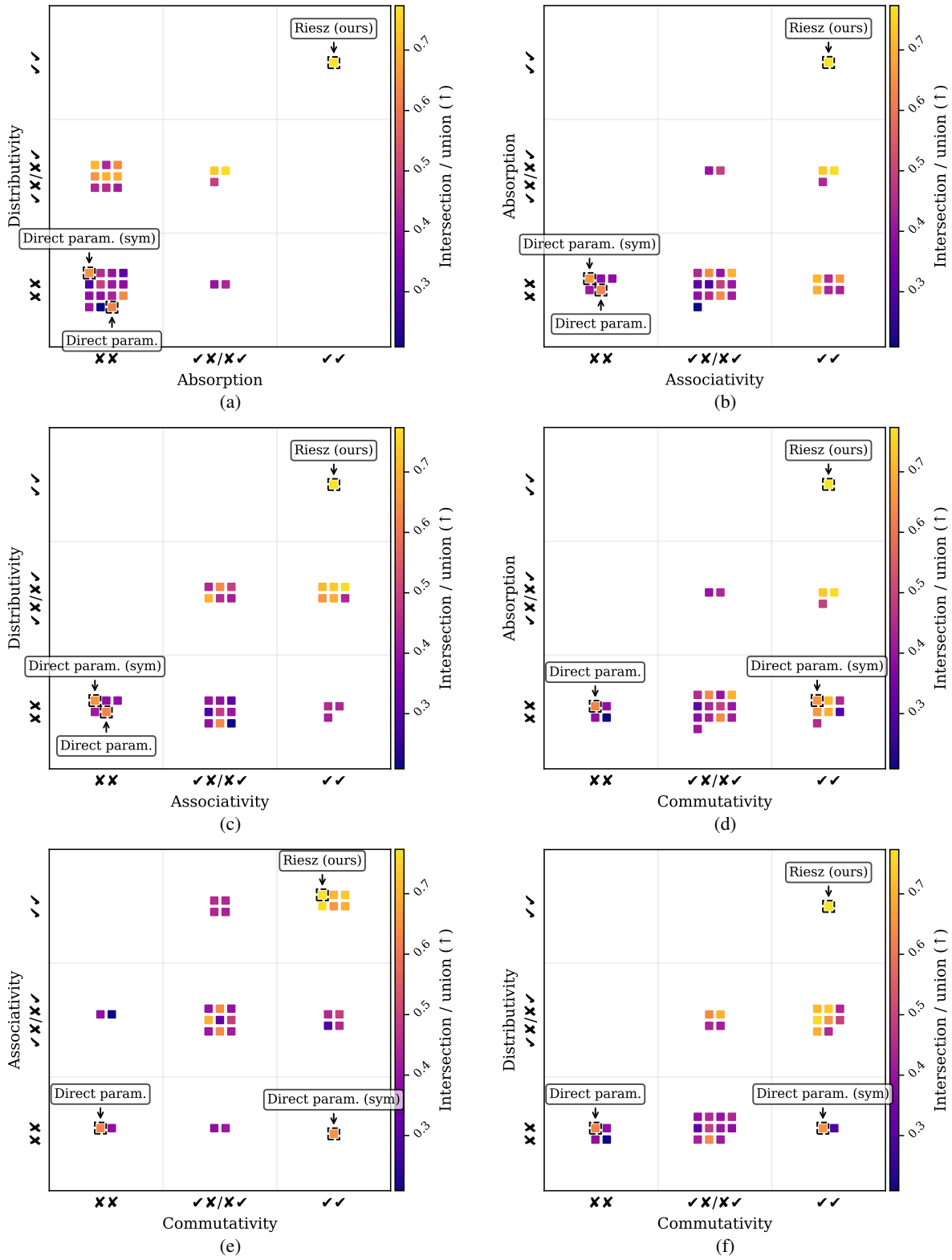

Figure 5. The performance of different algebras, grouped by their satisfaction of specific law combinations.