PAPER • OPEN ACCESS

Categorical representation learning: morphism is all you need

View the article online for updates and enhancements.

MACHINE LEARNING
Science and Technology

PAPER

Categorical representation learning: morphism is all you need

Artan Sheshmani[1,2,3,5,*] and **Yi-Zhuang You**[4,5,*]

1   Department of Mathematics, Center for Mathematical Sciences and Applications, Harvard University, 20 Garden Street, Cambridge, MA, 02139, United States of America
2   Institut for Matematik, Ny Munkegade 118 Building 1530, DK-8000 Aarhus C, Denmark
3   National Research University Higher School of Economics, Russian Federation, Laboratory of Mirror Symmetry, NRU HSE, 6 Usacheva Str., Moscow 119048, Russia
4   Department of Physics, University of California Sandiego, Condenced Matter Group, UC San Diego 9500 Gilman Drive, La Jolla, CA, 92093, United States of America
5   QGNai INC. (Quantum Geometric Networks for Artificial Intelligence), 83 Cambridge Parkway, Unit W806, Cambridge, MA, 02142, United States of America
*   Authors to whom any correspondence should be addressed.

E-mail: artan@cmsa.fas.harvard.edu and yzyou@physics.ucsd.edu

Keywords: category theory, categorical representation learning, natural language processing (NLP)

Abstract

We provide a construction for categorical representation learning and introduce the foundations of '*categorifier*'. The central theme in *representation learning* is the idea of everything to vector. Every object in a dataset $\mathcal{S}$ can be represented as a vector in $\mathbb{R}^n$ by an *encoding map* $E : \mathcal{O}bj(\mathcal{S}) \to \mathbb{R}^n$. More importantly, every morphism can be represented as a matrix $E : \mathcal{H}om(\mathcal{S}) \to \mathbb{R}^n_n$. The encoding map E is generally modeled by a *deep neural network*. The goal of representation learning is to design appropriate tasks on the dataset to train the encoding map (assuming that an encoding is optimal if it universally optimizes the performance on various tasks). However, the latter is still a *set-theoretic* approach. The goal of the current article is to promote the representation learning to a new level via a *category-theoretic* approach. As a proof of concept, we provide an example of a text translator equipped with our technology, showing that our categorical learning model outperforms the current deep learning models by 17 times. The content of the current article is part of a US provisional patent application filed by QGNai, Inc.

1. Introduction

The rise of category theory was a great revolution of mathematics in the 20th century. It enables us to see relationships between various fields that are otherwise imperceptible at ground level, attesting that seemingly unrelated areas of mathematics are not so different after all. A category describes a collection of *objects* together with the relations (called *morphisms*) between them. The concept of category has provided a unified template for different branches of mathematics. Category theory is expected to be suitable for modeling datasets with relational structures, such as semantic relations among words, phrases and sentences in language datasets, or social relations among people or organizations in social networks.

From the category theory perspective, *relationships are everything*. Objects must be defined and can only be defined through their interrelations. So far, the field of representation learning [1, 2] has remained focused on learning object encodings, following the idea of 'everything to vector'. More recently, graph neural network models [3, 4] and self-attentive mechanisms [5] have begun to model relationships between objects, but they still did not place relationships in the first place. This article aims to develop a novel machine learning framework, called *categorical representation learning*, that directly learns the *representation of relations* as feature matrices (or tensors). The learned relations will then be used in different tasks. For example, direct relations can be composed to reveal higher-order relations. Functor map between different categories can be learned by aligning the relations (morphisms). Relations can guide the algorithm to identify clusters of objects and to perform renormalization transformations to simplify the category.

1.1. Overview

The construction of the proposed architecture in the current article will contain three major steps:

- **Step 1: Mine the categorical structure from data**. In this step the objective is to develop the categorical representation learning approach, which enables the machine to extract the representation of objects and morphisms from data. The approach developed in this step will be the foundation of the construction which provides the morphism representation to drive further applications.
- **Step 2: Align the categorical structures between datasets**. The objective in this step is to develop the functorial learning approach, which can establish a functor between categories based on the learned categorical representations by aligning the morphism representations. The technique developed in this step will find applications in unsupervised (or semi-supervised) translation.
- **Step 3: Discover hierarchical structures with tensor categories**. The objective in this step is to combine the categorical representation learning and functorial learning in the setting of a tensor category to learn the tensor functor that fuses simple objects into composite objects. The approach here will enable the algorithm to perform renormalization group transformations on categorical dataset, which progressively simplifies the category structure. This will open up broad applications in classification and generation tasks of categories.

2. Categories and categorical embedding

2.1. Modeling the categorical structure in the data

A category $\mathcal{C}$ consists of a set of objects, $\mathcal{O}bj(\mathcal{C})$, and a set of morphisms $\mathcal{H}om(\mathcal{C})$ between objects. Each morphism f mapping a source object a to a target object b, is denoted as $f: a \rightarrow b$. It is possible that different pairs of objects can be related by the same morphism, in which case, we may think that the object-pairs share the same type of relations. It is also possible that the same pair of objects can be related by different morphisms, in which case, we may think that there are multiple relations between the objects. The hom-class $\mathcal{H}om(a,b)$ denotes the set of all morphisms from a to b. Morphisms benefit from composition laws. The composition is a binary operation; $\circ : \mathcal{H}om(a,b) \times \mathcal{H}om(b,c) \rightarrow \mathcal{H}om(a,c)$ defined for every three objects a,b,c. The composition of morphisms $f: a \rightarrow b$ and $g: b \rightarrow c$ is written as $g \circ f$. The composition map is associative, that is, $h \circ (g \circ f) = (h \circ g) \circ f$. Given a category $\mathcal{C}$, for every object $x \in \mathcal{O}bj(\mathcal{C})$ there exists an identity morphism $\mathrm{id}_x : x \rightarrow x$, such that for every morphism $f: a \rightarrow x$ and $g: x \rightarrow b$, one has $\mathrm{id}_x \circ f = f$ and $g \circ \mathrm{id}_x = g$, that is pre and post compositions of morphisms in the category with the identity morphism will leave them unchanged.

One can associate a vector space to a given category, as its representation space. For the latter one can think of such vector space as a category itself, and use a categorical functors as a map from a given category to the geometric space.
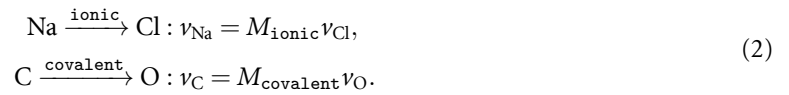
Take for instance a category $\mathcal{C}$ in which every *object*, a is mapped to a *vector* v_a in an ambient vector space R, and every morphisms between two objects in $\mathcal{C}$, $f: a \rightarrow b$, is mapped to a morphism between vectors v_a, v_b in R. Since R is a vector space, a morphism between two vectors is then realized in R as a *matrix $M_f : v_a \rightarrow v_b$*. For our purposes, the object and morphism embeddings, discussed above, will be implemented by separate embedding layers in a neural network. The map f is a morphism from object a to object b, iff the embedding matrix can transform the embedding vectors as $v_b = M_f v_a$ under matrix-vector multiplication. The composition of morphisms is then realized as matrix-matrix multiplications as $M_{g \circ f} = M_g M_f$, which is naturally associative. The identity morphism of an object $x \in \mathcal{C}$ is then represented as the projection operator $M_{\mathrm{id}_x} = v_x v_x^\mathsf{T}/|v_x|^2$, which preserves the vector representation v_x of the object. In this manner, the category structure is then represented respectively by the object, and morphism embeddings in the feature space.

Example 3.1. *For a language dataset, each object is given as a* word, *represented as a word vector, via the word-vector mapping functor. Each morphism is a* relation *between two words, represented by a matrix. For instance*

$$
\begin{aligned}
\mathtt{bright} &\xrightarrow{\mathrm{antonym}} \mathtt{dark} : v_{\mathtt{dark}} = M_{\mathrm{antonym}} v_{\mathtt{bright}}, \\
\mathtt{socks} &\xrightarrow{\mathrm{inside}} \mathtt{shoes} : v_{\mathtt{shoes}} = M_{\mathrm{inside}} v_{\mathtt{socks}}.
\end{aligned}
\tag{1}
$$

Given any pair of word vectors v_a, v_b, the hom-class $\mathcal{H}om(v_a, v_b)$ denotes the set of all matrices M which transform v_a to v_b, that is; $v_b = M v_a$.

Example 3.2. *For a chemical compound dataset, each object can be chosen to be a chemical element, represented as an element vector, via the embedding functor. Each morphism is a* relation *between two elements, which can, for example, stand for the type of chemical bound between elements, represented by a matrix. For instance*

$$\mathrm{Na} \xrightarrow{\texttt{ionic}} \mathrm{Cl} : v_{\mathrm{Na}} = M_{\texttt{ionic}} v_{\mathrm{Cl}},$$
$$\mathrm{C} \xrightarrow{\texttt{covalent}} \mathrm{O} : v_{\mathrm{C}} = M_{\texttt{covalent}} v_{\mathrm{O}}. \tag{2}$$

Different types of chemical bounds (ionic, covalent, metallic etc) are model by different matrices. Two elements can form the chemical bound, if their vector representations can be transformed from one to another by the matrix. Of course, this is an over-simplified illustration of relations between chemical elements. The realistic relations between elements in a chemical compound can be complicated, for which the categorical representation learning is proposed to learn useful representation of relations from data.

2.2. Mining the categorical structure from the data

2.2.1. Fuzzy morphisms

In machine learning, the relation between objects may not be strict, that is; $M_f v_a$ will not match v_b precisely, but only align with v_b with a high probability. To account for the fuzziness of relations then, the statement of $f : a \to b$ should be replaced by the probability of f belonging to the class $\mathcal{H}om(a,b)$. It can be written as

$$P(f \in \mathcal{H}om(a,b)) \equiv P(a \xrightarrow{f} b) \propto \exp z(a \xrightarrow{f} b), \tag{3}$$

which is parametrized by the logit $z(a \xrightarrow{f} b) \in \mathbb{R}$. The logit is proposed to be modeled by

$$z(a \xrightarrow{f} b) = v_b^\mathsf{T} M_f v_a, \tag{4}$$

such that if v_b and $M_f v_a$ align in similar directions, the logit will be positive, which results in a high probability for the morphism f to connect a to b. The objects a and b are said to be related (linked) if there exists at least one morphism connecting them.

The linking probability is then proportional to the sum of the likelihood of each candidate morphism, $P(a \to b) \propto \sum_f \exp z(a \xrightarrow{f} b)$. The probability is normalized by considering the unlinking likelihood as the unit. Thus the binary probability $P(a \to b)$ can be modeled by the sigmoid of a logit $z(a \to b)$ that aggregates the contributions from all possible morphisms,

$$P(a \to b) = \mathsf{sigmoid}(z(a \to b)) \equiv \frac{e^{z(a \to b)}}{1 + e^{z(a \to b)}},$$
$$z(a \to b) = \log \sum_f \exp z(a \xrightarrow{f} b) = \log \sum_f \exp(v_b^\mathsf{T} M_f v_a), \tag{5}$$

where it is assumed that each morphism contributes to the linking probability independently. More generally, the assumption of the independence of morphisms in determining the linking probability can be relaxed, such that the logits $z(a \xrightarrow{f} b)$ of different morphisms f is aggregated by a generic nonlinear function:

$$z(a \to b) = F\Big(\bigoplus_f z(a \xrightarrow{f} b) \Big) = F\Big(\bigoplus_f v_a^\mathsf{T} M_f v_b \Big), \tag{6}$$

where F may be realized by a deep neural network, and $\bigoplus_f$ denotes the concatenation of logit contributions from different morphisms.

2.2.2. Learning embedding from statistics

The key idea of unsupervised representation learning is to develop feature representations from data statistics. In the categorical representation learning, the object and morphism embeddings are learned from the concurrence statistics, which corresponds to the probability $p(a,b)$ that a pair of objects (a,b) occurs together in the same composite object. For example, two concurrent objects could stand for two elements in the same compound, or two words in the same sentence, or two people in the same organization. Concurrence does not happen for no reason. Assuming that two objects appearing together in the same structure can always be attributed to the fact that they are related by at least one type of relations, then the linking probability $P(a \to b)$ should be maximized for the observed concurrent pair (a,b) in the dataset.

The negative sampling method can be adopted to increase the contrast. For each observed (positive) concurrent pair (a,b), the object b will be replaced by a random object b', drawn from the negative sampling distribution $p_N(b')$ among all possible objects. The replaced pair (a,b') will be considered to be an unrelated pair of objects. Therefore the training objective is to maximize the linking probability $P(a \to b)$ for the

positive sample (a, b) and also maximize the unlinking probability $P(a \nrightarrow b') = 1 - P(a \rightarrow b')$ for a small set of negative samples (a, b'),

$$\mathcal{L} = \underset{(a,b) \sim p(a,b)}{\mathbb{E}} \left(\log P(a \rightarrow b) + \underset{b' \sim p_N(b')}{\mathbb{E}} \log(1 - P(a \rightarrow b')) \right). \tag{7}$$

With the model of $P(a \rightarrow b)$ in equation (5), the object embedding v_a and morphism embedding M_f can be trained by maximizing the objective function $\mathcal{L}$. The negative sampling distribution $p_N(b')$ can be engineered, but the most natural choice is to take the marginalized object distribution $p_N(b') = p(b') = \sum_a p(a, b')$. The theoretical optimum [6] is achieved with the following logit

$$z(a \rightarrow b) = \log \frac{p(a,b)}{p(a)p(b)} = \text{PMI}(a, b), \tag{8}$$

which learns the point-wise mutual information (PMI) between objects a and b. If the objects were not related at all, their concurrence probability should factorize to the product of object frequencies, i.e. $p(a, b) = p(a)p(b)$, implying zero PMI. So a non-zero PMI indicates non-trivial relations among objects: a positive relation (PMI > 0) enhances $p(a, b)$ while a negative relation (PMI < 0) suppresses $p(a, b)$ relative to $p(a)p(b)$. By training the logit $z(a \rightarrow b)$ to approximate the PMI, the relations between objects are discovered and encoded as the feature matrix M_f of morphisms.

2.2.3. Scope of concurrence

It is noted that the definition of concurrence depends on the scope (or the context scale). For example, the concurrence of two words can be restricted in the scope of phrases or sentences or paragraphs. Different choices for the concurrence scope (say length of sentences, or phrases being considered) will affect the concurrent pair distribution $p(a, b)$, which then leads to different results for the object and morphism embeddings. This implies that objects may be related by different types of morphisms in different scopes. The categorical representation learning approach mentioned above can be applied to uncover the morphism embeddings in a scope-dependent manner. Eventually, the different object and morphism embeddings across various scopes can further be connected by the renormalization functor, which maps the category structure between different scales, detail of which will be elaborated in later sections. The scope-dependent categorical representation learning will form the basis for the development of unsupervised renormalization technique for categories with hierarchical structures, which will find broad applications in translation, classification, and generation tasks.

2.2.4. Connection to multi-head attention

The multi-head attention mechanism [5] consists of two steps. The first step is a dynamic link prediction by the '*query-key matching*', that the probability to establish an attention link from a to b is given by $P(a \xrightarrow{f} b) \propto \exp z(a \xrightarrow{f} b)$ with f labeling the attention head. The logit is computed from the inner product between the query vector, $Q_f v_b$, and the key vector, $K_f v_a$, as

$$z(a \xrightarrow{f} b) = v_b^\mathsf{T} Q_f^\mathsf{T} K_f v_a, \tag{9}$$

which can be written in the form of equation (4) if $M_f = Q_f^\mathsf{T} K_f$. The second step is the value propagation along the attention link (weighted by the linking probability). Focusing on the first step of dynamic link prediction, the linking probability model in equation (5) resembles the multi-head attention mechanism with the morphism embedding $M_f = Q_f^\mathsf{T} K_f$ given by the product of query and key matrices for each attention head. The proposal of the categorical representation learning is to keep the learned morphism matrix M_f as encoding of relations in the feature space, which can be further used in other task applications.

    The matrix M_f can also be viewed as a metric in the feature space, which defines the inner product of object vectors. Different morphisms f correspond to different metrics M_f, which distort the geometry of the feature space differently (bring object embeddings together or pushing them apart). When there are multiple relations in the action, the feature space is equipped with different metrics simultaneously, which resembles the idea of superposition of geometries in quantum gravity. If the single-head logit $z(a \xrightarrow{f} b) = v_b^\mathsf{T} M_f v_a$ is considered as an (negative) energy of two objects a and b embedded in a specific geometry, the aggregated multi-head logit $z(a \rightarrow b) = \log \sum_f \exp z(a \xrightarrow{f} b)$ is analogous to the (negative) free energy that the objects will experience in the ensemble of fluctuating geometries.

2.3. Aligning the categorical structures between data sets

2.3.1. Tasks as functors

Functor is a fundamental concept in the category theory. It denotes the structure-preserving map between two categories. A functor $\mathcal{F}$ from a source category $\mathcal{C}$ to a target category $\mathcal{D}$ is a mapping that associates to each object a in $\mathcal{C}$ an object $\mathcal{F}(a)$ in $\mathcal{D}$, and associates to each morphism $f: a \to b$ in $\mathcal{C}$ a morphism $\mathcal{F}(f): \mathcal{F}(a) \to \mathcal{F}(b)$ in $\mathcal{D}$, such that $\mathcal{F}(\mathrm{id}_a) = \mathrm{id}_{\mathcal{F}(a)}$ for every object a in $\mathcal{C}$ and $\mathcal{F}(g \circ f) = \mathcal{F}(g) \circ \mathcal{F}(f)$ for all morphisms $f: a \to b$ and $g: b \to c$ in $\mathcal{C}$. The definition can be illustrated with the following commutative diagram.

$$
\begin{array}{ccc}
\mathcal{C} & & a \xrightarrow{\;\;f\;\;} b \\
\big\downarrow{\scriptstyle \mathcal{F}} & & \big\downarrow{\scriptstyle \mathcal{F}} \qquad \big\downarrow{\scriptstyle \mathcal{F}} \\
\mathcal{D} & & \mathcal{F}(a) \xrightarrow{\;\mathcal{F}(f)\;} \mathcal{F}(b)
\end{array}
\tag{10}
$$

A functor between two categories not only maps objects to objects, but also preserves their relations, which is essential in category theory. Many machine learning tasks can be generally formulated as functors between two data categories. For instance, machine translation is a functor between two language categories, which not only maps words to words but also preserves the semantic relations between words. Image captioning is a functor from image to language categories, that transcribes objects in the image as well as their interrelations.

In what follows, we discuss *Functorial learning* as an approach, providing the means for the machine to learn the functorial maps between categories in an unsupervised or semisupervised manner.

2.3.2. Functorial learning

In functorial learning, each functor $\mathcal{F}$ is represented by a transformation $V_{\mathcal{F}}$ that transforms the vector embedding v_a of each object a in the source category to the vector embedding $v_{\mathcal{F}(a)}$ of the corresponding object $\mathcal{F}(a)$ in the target category

$$
v_{\mathcal{F}(a)} = V_{\mathcal{F}} v_a,
\tag{11}
$$

and also transforms the matrix embedding M_f of each morphism in the source category to the matrix embedding $M_{\mathcal{F}(f)}$ of the corresponding morphism $\mathcal{F}(f)$ in the target category

$$
M_{\mathcal{F}(f)} V_{\mathcal{F}} = V_{\mathcal{F}} M_f,
\tag{12}
$$

represented by the following commutative diagram.

$$
\begin{array}{ccc}
\mathcal{C} & & v_a \xrightarrow{\;\;M_f\;\;} v_b \\
\big\downarrow{\scriptstyle \mathcal{F}} & & \big\downarrow{\scriptstyle V_{\mathcal{F}}} \qquad \big\downarrow{\scriptstyle V_{\mathcal{F}}} \\
\mathcal{D} & & V_{\mathcal{F}} v_a \xrightarrow{\;M_{\mathcal{F}(f)}\;} V_{\mathcal{F}} v_b
\end{array}
$$

In the case where the objects are embedded as unit vectors on a hypersphere, the functorial transformation $V_{\mathcal{F}}$ will be represented by orthogonal matrices, whose inverses are simply given by the transpose matrices $V_{\mathcal{F}}^{\mathsf{T}}$, which admit efficient implementation in the algorithm. The proposed representation for the functor automatically satisfies the functor axioms, as

$$
\begin{aligned}
M_{\mathcal{F}(\mathrm{id}_a)} &= v_{\mathcal{F}(a)} v_{\mathcal{F}(a)}^{\mathsf{T}} = V_{\mathcal{F}} v_a v_a^{\mathsf{T}} V_{\mathcal{F}}^{\mathsf{T}} = V_{\mathcal{F}} M_{\mathrm{id}_a} V_{\mathcal{F}}^{\mathsf{T}}, \\
M_{\mathcal{F}(g \circ f)} &= V_{\mathcal{F}} M_{g \circ f} V_{\mathcal{F}}^{\mathsf{T}} = V_{\mathcal{F}} M_g M_f V_{\mathcal{F}}^{\mathsf{T}} \\
&= V_{\mathcal{F}} M_g V_{\mathcal{F}}^{\mathsf{T}} V_{\mathcal{F}} M_f V_{\mathcal{F}}^{\mathsf{T}} = M_{\mathcal{F}(g)} M_{\mathcal{F}(f)} = M_{\mathcal{F}(g) \circ \mathcal{F}(f)}.
\end{aligned}
$$

So as long as the optimal transformation $V_{\mathcal{F}}$ can be found, our design will ensure that it parametrizes a legitimate functor between categories.

2.3.3. Universal structure loss

Now we explain how to find the optimal transformation $V_{\mathcal{F}}$. The most important requirement for a functor is to preserve the morphisms between two categories. Hence we propose to train the functor by minimizing a loss function which ensures the structural compatibility in equation (12). We denote the loss function by *universal structure loss*,

$$
\mathcal{L}_{\mathrm{struc}} = \sum_f \| M_{\mathcal{F}(f)} V_{\mathcal{F}} - V_{\mathcal{F}} M_f \|^2,
\tag{13}
$$

given the matrix representation $M_f, M_{\mathcal{F}(f)}$ of morphisms in both the source and target categories, which were obtained from the categorical representation learning. The loss function is universal in the sense that it is independent of the specific task that the functor is trying to model. In this approach, the morphism embeddings are all that we need to drive the learning of functor transformation $V_{\mathcal{F}}$. This is precisely in line with the spirit of the category theory: objects are illusions, they must be defined and can only be defined by morphisms. Therefore

<div align="center">

'***Morphism is all you need!***'.

</div>

2.3.4. Alignment loss

However, in reality, the learned morphism matrices M_f may not be of full rank. In such cases, the solution of the transformation $V_{\mathcal{F}}$ is not unique, because arbitrary transformation within the null space of the morphism matrix can be composed with $V_{\mathcal{F}}$ without affecting the structure lost $\mathcal{L}_{\text{struc}}$. To overcome this difficulty, we propose to subsidize the structure loss with additional alignment loss over a few pair of aligned pairs of objects $(a, \mathcal{F}(a))$,

$$\mathcal{L}_{\text{align}} = \sum_{a \in \mathcal{A}} \| v_{\mathcal{F}(a)} - V_{\mathcal{F}} v_a \|^2, \tag{14}$$

where $\mathcal{A}$ is only a subset of objects in the source category. This provides additional supervised signals to train $V_{\mathcal{F}}$ by partially enforcing equation (11). The total loss will be a weighted combination of the structure loss and the alignment loss

$$\mathcal{L} = \mathcal{L}_{\text{struc}} + \lambda \mathcal{L}_{\text{align}}. \tag{15}$$

By minimizing the total loss, $V_{\mathcal{F}}$ will be trained, and the functor between categories can be established. The advantage of the categorical approach is that the structural loss already puts constrains on the parameters in $V_{\mathcal{F}}$, so the effective parameter space for the alignment loss is reduced, such that the model can be trained with much less supervised samples when the category structure is rigid enough.

2.4. Discovering hierarchical structures with tensor categories

2.4.1. Renormalization as tensor bifunctor

Renormalization group plays an essential role in analyzing the hierarchical structures in physics and mathematics. It provides an efficient approach to extract the essential information of a system by progressively coarse graining the objects in the system. The categorical representation learning and functorial learning proposed in the previous two sections provide solid foundation to develop hierarchical renormalization approach for machine learning. This will enable the machine to summarize the feature representation of composite objects from that of simple objects.

The elementary step of a coarse graining procedure is to fuse two objects into one composite object (or 'higher' object). Multiple objects can then be fused together in a pair-wise manner progressively. In category theory, the pair-wise fusion of objects is formulated as a tensor bifunctor $\otimes : \mathcal{C} \times \mathcal{C} \to \mathcal{C}$. The tensor bifunctor maps each pair of objects (a, b) to a composite object $a \otimes b$ and each pair of morphisms (f, g) to a composite morphism $f \otimes g$ while preserving the morphisms between objects, as presented in the following diagram.
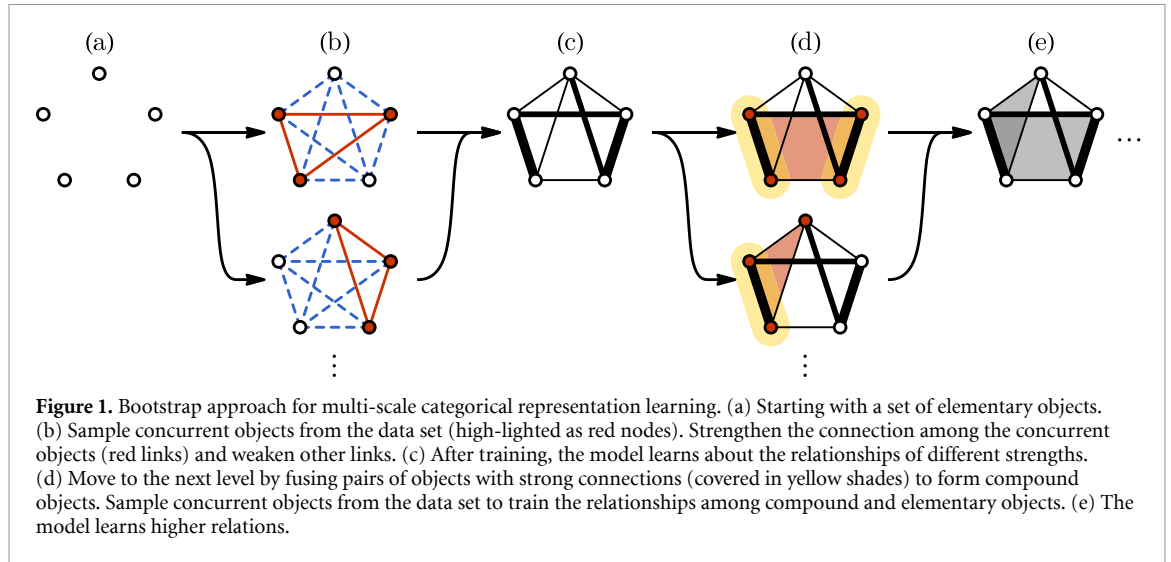
$$
\begin{array}{ccc}
\mathcal{C} \times \mathcal{C} & \qquad (a,b) \xrightarrow{\ (f,g)\ } (f(a), g(b)) \\
\Big\downarrow & \qquad \Big\downarrow \qquad\qquad\qquad \Big\downarrow \\
\mathcal{C} & \qquad a \otimes b \xrightarrow{\ f \otimes g\ } (f \otimes g)(a \otimes b)
\end{array}
\tag{16}
$$

The category equipped with the tensor bifunctor is called a tensor category (or monoidal category). Objects and morphisms of different hierarchies are treated within the same framework systematically. This enables the algorithm to model multi-scale structure in the data set with the universal approach of categorical representation learning.

2.4.2. Representing tensor bifunctor

As a special case of general functors, the tensor bifunctor can be represented by a *fusion operator* Θ, such that the object embeddings are fused by

$$v_{a \otimes b} = \Theta(v_a \otimes v_b), \tag{17}$$

Figure 1. Bootstrap approach for multi-scale categorical representation learning. (a) Starting with a set of elementary objects. (b) Sample concurrent objects from the data set (high-lighted as red nodes). Strengthen the connection among the concurrent objects (red links) and weaken other links. (c) After training, the model learns about the relationships of different strengths. (d) Move to the next level by fusing pairs of objects with strong connections (covered in yellow shades) to form compound objects. Sample concurrent objects from the data set to train the relationships among compound and elementary objects. (e) The model learns higher relations.

and the morphism embeddings are fused by

$$M_{f \otimes g} \Theta = \Theta(M_f \otimes M_g), \tag{18}$$

following the general scheme in equations (11) and (12). The tensor product of the vector and matrix representations are implemented as Kronecker products. Θ can be viewed as an operator which projects the tensor-product feature space back to the original feature space.

To simplify the construction, we will assume that the representation of the tensor bifunctor is strict in the feature space, meaning that the fusion is strictly associative without non-trivial natural isomorphisms,

$$v_{a \otimes b \otimes c} = \Theta(\Theta(v_a \otimes v_b) \otimes v_c) = \Theta(v_a \otimes \Theta(v_b \otimes v_c)). \tag{19}$$

Such a strict representation will always be possible given a large enough feature space dimension, since every monoidal category is equivalent to a strict monoidal category. To impose the strict associativity in the learning algorithm, we propose to fuse objects in different orders, such that the machine will not develop any preference over a particular fusion tree and will learn to construct an associative fusion operator Θ. With the fusion operator, we establish the embedding for all composite objects (and their morphisms) in the category given the embedding of fundamental objects (and their morphisms).

2.4.3. Multi-scale categorical representation learning

The tensor bifunctor learning can be combined with the categorical representation learning as an integrated learning scheme, which allows the algorithm to mine the category structure at multiple scales. If the data set has naturally-defined levels of scopes, one can learn the objects and morphism embeddings in different scopes. The objective is to learn the concurrence of objects within their scope according to the loss function equation (7). For each pair of objects (a, b) drawn from the same scope, we want to maximize the linking probability $P(a \to b)$. For randomly sampled pairs of objects (a, b'), we want to maximize the unlinking probability $P(a \nrightarrow b') = 1 - P(a \to b')$. The linking probability $P(a \to b)$ is modeled by equation (5), based on the object and morphism embeddings. The vector embedding v_a of a composite object a is constructed by recursively applying Θ to fuse from fundamental objects as in equation (19).

A more challenging situation is that the data set has no naturally-defined levels of scopes, such that the hierarchical representation must be established with a bootstrap approach. We assume that at least a set of elementary objects can be specified in the data set (such as words in the language data set), illustrated as small circles in figure 1(a). We first apply the categorical representation learning approach to learn the linking probability $P(a \to b)$ between objects. The algorithm first samples different clusters of objects from the data set within a certain scale. The linking probability is enhanced for the pair of objects concurring in the same cluster, and is suppressed otherwise, as shown in figure 1(b). In this way, the algorithm learns to find the embedding v_a for every fundamental object a. After a few rounds of training, fuzzy morphisms among objects will be established, as depicted in figure 1(c). The thicker link represents higher linking probability $P(a \to b)$ and stronger relations. In later rounds of training, when the sampled cluster contains a pair of objects (a, b) connected by a strong relation (with $P(a \to b)$ beyond a certain threshold), it will be considered as a composite object $a \otimes b$, as yellow groups in figure 1(d). The embedding $v_{a \otimes b}$ of composite

object will be calculated from that of the fundamental objects as $v_{a \otimes b} = \Theta(v_a \otimes v_b)$, based on the fusion operator Θ given by the tensor bifunctor model. The model will continue to learn the linking probability $P(a \otimes b \rightarrow c)$ between the composite object $a \otimes b$ and the other object c in the cluster, as red polygons in figure 1(d). In this way, the fusion operator Θ will get trained together with the object and morphism embeddings. This will establish higher morphisms between composite objects like figure 1(e). The approach can then be carried on progressively to higher levels, which will eventually enable the machine to learn the hierarchical structures in the data set and establish the representations for objects, morphisms and tensor bifunctors all under the same approach.

3. Summary

In this work, we propose the general framework for categorical representation learning—a relationship-oriented representation learning approach. It allows machine learning algorithms to uncover the categorical structure in the dataset, and to utilize the learned representation of objects and morphisms for down-stream tasks. A key ingredient of the categorial representation learning beyond conventional approach lies in its ability to obtain explicit matrix representations of morphisms (apart from vector representations of objects), which can be applied to learn the functorial map (relational-structure-preserving map) between categorical datasets.

Data availability statement

The data generated and/or analysed during the current study are not publicly available for legal/ethical reasons but are available from the corresponding author on reasonable request.

Acknowledgment

Appendix. Proof of concept results

Learning chemical compounds
Data set
To demonstrate the proposed categorical learning framework, we apply our approach to the inorganic chemical compound data set. The data set for our proof of concept (POC) contains 61 023 inorganic compounds, covering 89 elements in the periodic table figure 2(a). The data set can be modeled as a category, which contains elements as fundamental objects, as well as functional groups and compounds as composite objects. The morphisms represent the relations (such as chemical bonds) between atoms or groups of atoms. We assume that the concurrence of two elements in a compound is due to the underlying relations, such that the morphisms can emerge from learning the elements' concurrence.

On the data set level, we collect the point-wise mutual information (PMI) $\mathrm{PMI}(a, b) = \log p(a, b) - \log p(a) - \log p(b)$, where $p(a, b)$ is the probability for the pair of elements a, b to appear in the same chemical compound, and $p(a)$ is the marginal distribution. Performing a principal component analysis of the PMI and taking the leading three principal components, we can obtain three-dimensional vector encodings of elements, as shown in figure 2(b). We observe that elements of similar chemical properties are close to each other, because they share similar context in the compound. This observation indicates that our algorithm is likely to uncover such relations among elements from the data set.

Unsupervised/semisupervised translation
To demonstrate the categorical representation learning and the functorial learning in sections 1 and 2, we designed an unsupervised translation task with the chemical compound data set. We take the data set in English and translate each element into Chinese. Figures 2(c) and (d) shows some samples from both the English and Chinese data set. The task of unsupervised (or semisupervised) translation is to learn to translate chemical compounds from one language to another without aligned samples (or with only a few aligned samples). The unsupervised translation is possible since the chemical relation between elements are identical in both languages. The categorical representation learning can capture these relations and represent them as morphism embeddings. By aligning the morphism embeddings, using the functorial learning approach, the translator can be learned as a functor that maps between the English and Chinese compound categories. The translation functor is required to map elements to elements while preserving their relations.
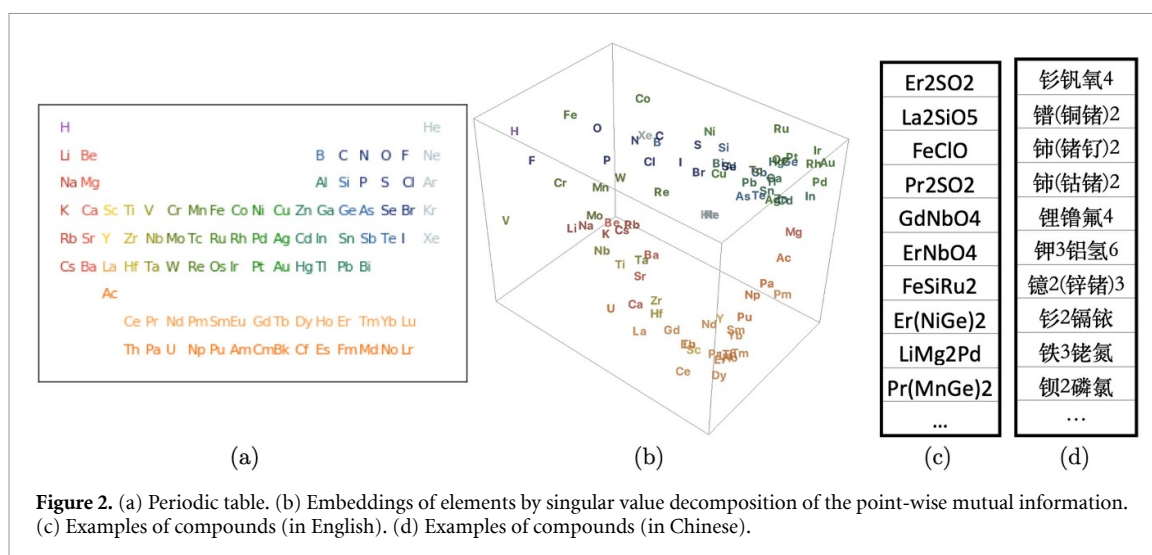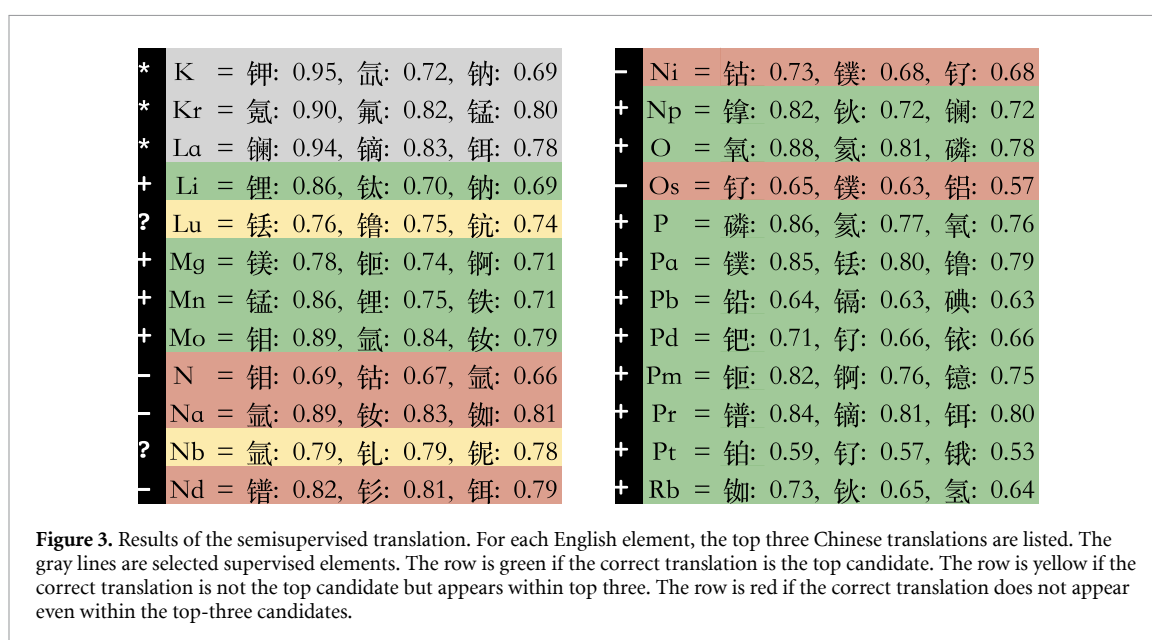
Figure 2. (a) Periodic table. (b) Embeddings of elements by singular value decomposition of the point-wise mutual information. (c) Examples of compounds (in English). (d) Examples of compounds (in Chinese).

Figure 3. Results of the semisupervised translation. For each English element, the top three Chinese translations are listed. The gray lines are selected supervised elements. The row is green if the correct translation is the top candidate. The row is yellow if the correct translation is not the top candidate but appears within top three. The row is red if the correct translation does not appear even within the top-three candidates.

We demonstrate the semisupervised translation in figure 3. We assign 15 elements as supervised data and provide the aligned English-Chinese element pairs to the machine. By minimizing the structure and alignment loss together.

More on model structure for translator (by Ahmadreza Azizi[6])

Many of the state-of-the-art translation models (usually called sequence to sequence or seq-2-seq models) incorporate two blocks of Encoder and Decoder that are connected to the source input and the target input respectively. Also in different models the Encoder and Decoder blocks are connected to each other in different ways. For the purpose of POC, we hence use the most relevant design in the seq-2-seq models and compare its results with our categorical learning model. Recently, text translation models with transformers have had impressive achievements on the datasets with long inputs (usually more than 250 tokens). However we do not use transformers, due to the name of compounds not being long and hence transformers do not have specific supremacy over their counterparts with recurrent neural network (RNN) based cells in this case. Therefore our choice is to design a seq-2-seq model which includes two blocks of Encoder and Decoder with general recurrent unit (GRU) cells which benefits from the attention mechanism (see figure 4). We add attention mechanism to the model as there is no significant language pattern in the compounds, hence the connection between Encoder and Decoder must be empowered with attention mechanism so that almost all
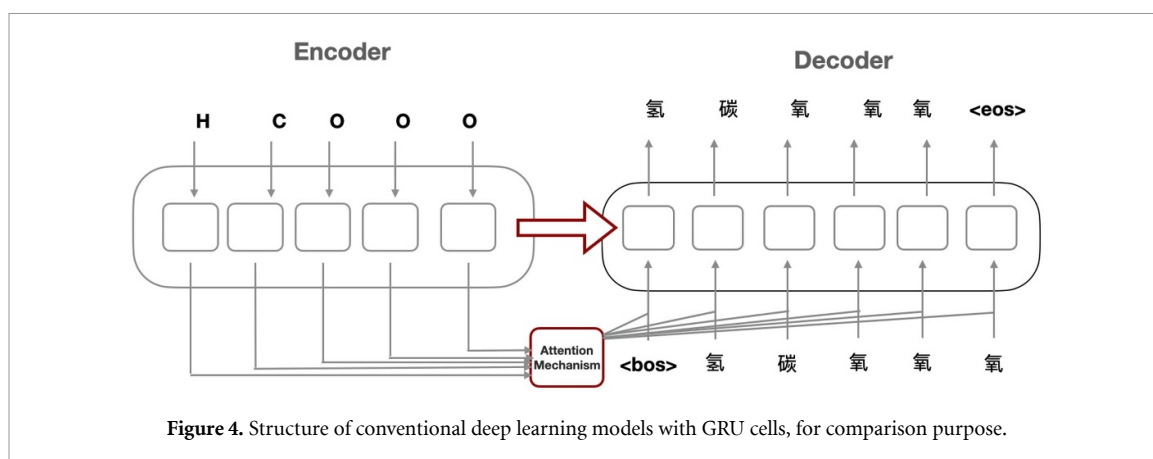
---

[6] Ahmadreza Azizi contributes to the appendix.

Figure 4. Structure of conventional deep learning models with GRU cells, for comparison purpose.

Table 1. Performance of deep learning models and categorical learning model after training on compounds data.

Models	Categorical learning	Seq2Seq	Seq2Seq	Seq2Seq
Number of parameters	16 600	25 626	77 222	287 020
Number of supervised elements	15	15	15	15
Number of correct translations	57	12	38	59

the information in the Encoder can be expressed to the Decoder. Finally, we apply the teacher-force technique to provide aid to the Decoder for translating the English name of compounds to the Chinese counterparts, more robustly.

Learning method for translator
Formally, the source sentence s is a set of words $s = \{s_1, s_2, \ldots, s_T\}$ and the target sentence is $t = \{t_1, t_2, \ldots, t_T\}$ both with length T. The embedding layers of source sentence and target sentence represent them as two sets of vectors $x = \{x_1, x_2, \ldots, x_T\} \in X$ and $y = \{y_1, y_2, \ldots, y_T\} \in Y$ respectively. For each element in x, the Encoder E reads x_t and encapsulates its information in the hidden states

$$h_t = E(h_{t-1}, x_{t-1}).$$

After reading all elements in x, the Encoder outputs the last hidden layer $h_T = E(h_{T-1}, x_{T-1})$ to be passed to the first cell in Decoder. Therefore for each element y_t in y, the hidden state of the decoder cell at t is given by

$$s_t = D(s_{t-1}, y_t, c_t).$$

In this formula, the attention mechanism is expressed as the context vector c_t. Conceptually, the context vector $c_t = \sum_{i=1}^{T} \alpha_{t,i} h_i$ carries out the sum of information of the encoder cells (h_i) weighted by the alignment score $\alpha_{t,i}$. Here the score function $\alpha_{t,i}$ indicates how much information from each input x_i should be contributed to the output y_t and can have different forms [5, 7].

Finally, given the output of Decoder cells $\hat{y}$, its similarity will be compared to the true translation y thorough the cross entropy loss function:

$$L(\theta_D, \theta_E) = -\frac{1}{N} \sum_i y_i \, log \, \hat{y}_i \tag{20}$$

with θ_E and θ_D the trainable parameters in the Encoder block and the Decoder block respectively. The equation (20) can be seen as a metric function that measures the distance between y and $\hat{y}$.

Translator results
According to equation (20), in the process of training, the parameters Θ_E and Θ_D are optimized so that the model output $\hat{y}$ becomes very similar to the ground truth y. We train the deep learning model with various sizes of GRU cells and count number of correct translations that are made in each model.

Note that the categorical learning model has only 16 600 parameters and we believe it is only fair if only we compare the deep learning model with similar number of parameters. Table 1 elaborates on our results for deep learning models and our categorical learning model. These results indicate that the deep learning models are unable to outperform our model unless the number of parameters they use is 17 times more than

our categorical learning model, in which case the deep learning model has the capability of achieving a better performance. This comparison is particularly interesting, once one decreases the number of parameters to about 25 000. Our experiments clearly indicate that the categorical learning methods completely outperform the deep learning models.

ORCID iD

Yi-Zhuang You ● https://orcid.org/0000-0003-4080-5340

References

[1] Bengio Y, Courville A and Vincent P 2013 Representation learning: a review and new perspectives *IEEE Trans. Pattern Anal. Mach. Intell.* **35** 1798–828
[2] Goodfellow I, Bengio Y, Courville A and Bengio Y 2016 *Deep Learning* vol 1 (Cambridge: MIT Press)
[3] Scarselli F, Gori M, Tsoi A C, Hagenbuchner M and Monfardini G 2008 The graph neural network model *IEEE Trans. Neural Netw.* **20** 61–80
[4] Zhou J, Cui G, Zhang Z, Yang C, Liu Z, Wang L, Li C and Sun M 2018 Graph neural networks: a review of methods and applications (arXiv:1812.08434)
[5] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A N, Kaiser Łukasz and Polosukhin I 2017 Attention is all you need *Advances in Neural Information Processing Systems* pp 5998–6008
[6] Levy O and Goldberg Y 2014 Neural word embedding as implicit matrix factorization *Advances in Neural Information Processing Systems* pp 2177–85
[7] Bahdanau D, Cho K and Bengio Y 2014 Neural machine translation by jointly learning to align and translate (arXiv:1409.0473)