
ON THE POWER OF FOUNDATION MODELS

Yang Yuan^{1,2,3}

¹IIS, Tsinghua University

²Shanghai Artificial Intelligence Laboratory

³Shanghai Qi Zhi Institute

ABSTRACT

With infinitely many high-quality data points, infinite computational power, an infinitely large foundation model with a perfect training algorithm and guaranteed zero generalization error on the pretext task, can the model be used for everything? This question cannot be answered by the existing theory of representation, optimization or generalization, because the issues they mainly investigate are assumed to be nonexistent here. In this paper, we show that category theory provides powerful machinery to answer this question. We have proved three results. The first one limits the power of prompt-based learning, saying that the model can solve a downstream task with prompts if and only if the task is representable. The second one says fine tuning does not have this limit, as a foundation model with the minimum required power (up to symmetry) can theoretically solve downstream tasks for the category defined by pretext task, with fine tuning and enough resources. Our final result can be seen as a new type of generalization theorem, showing that the foundation model can represent unseen objects from the target category (e.g., images) using the structural information from the source category (e.g., texts). Along the way, we provide a categorical framework for supervised and self-supervised learning, which might be of independent interest.

1 Introduction

Foundation models have recently exhibited remarkable proficiency in addressing a myriad of complex downstream tasks that were very difficult or impossible for the previous methods (Ramesh et al., 2021; Rombach et al., 2022; Ramesh et al., 2022; Sohl-Dickstein et al., 2015; Brown et al., 2020; Radford et al., 2018, 2019; He et al., 2022). Being different in network structure, dataset and training algorithms, the foundation models are similar in terms of the training process: the model is first trained with a large unlabeled dataset on a pretext task, and then being applied to other downstream tasks with the parameters frozen. Freezing the parameter is necessary because training a foundation model is very expensive, and the downstream tasks usually have limited data samples, which makes retraining less attractive.

Given a set of downstream tasks, which of them are solvable by the foundation models, and which are not? This is a very fundamental question. The existing theory attacks this question through various perspectives like dataset quality and quantity, computational power, network structure, etc. However, as we collect trillions of data points, build gigantic GPU data centers, and optimize huge networks with billions of parameters, a simple question just pops up:

Where are we heading to?

Specifically, with infinitely many high-quality data points, infinite computational power, an infinitely large foundation model with a perfect training algorithm and guaranteed zero generalization error on the pretext task, can the model be used for every downstream task? As we will see, this question is not about model representation, optimization or generalization, but structural representation of the tasks. Category theory, as the theory of mathematical structures, is the ideal tool for answering this question.

For the downstream tasks, there are mainly two types of methods: prompt tuning and fine tuning. Prompt tuning does not train with the downstream tasks. Instead, it only sends a task specific prompt to the model, so that the model can

“switch” its working mode for solving the task. Fine tuning trains a small network connecting to the foundation model with the labeled dataset for the downstream task.

In Theorem 3, we show that with prompt tuning, the model can solve the task if and only if the task is “representable” in the category defined by the pretext task. If the category does not have complicated structures, our theorem indicates that the power of prompt tuning is limited. For example, in Corollary 1, we show that the rotation pretext task Gidaris et al. (2018) alone does not guarantee that prompt tuning can solve complicated downstream tasks such as segmentation or classification.

On the other hand, the results on fine tuning are more promising. Our Theorem 5 proves that for the foundation model with the minimum required power (up to symmetry) for the pretext task and enough resources including training data, it can potentially solve any downstream tasks for the category defined by pretext task. The role of pretext task is crucial in the sense that if the pretext task fails to extract adequate information from the unlabeled dataset, the power of fine tuning remains restricted.

Along the way, we have provided a categorical framework for machine learning. Interestingly, the framework injects the learning perspective to category theory as well. Therefore, we also proved a generalization theorem for structural learning (Theorem 6), which explains why self-supervised learning for text-image generation tasks can represent image objects such as avocado chair, which do not exist in the dataset or real world, and can support their generation when combined with a target-side decoder or generator. Theorem 6 can be easily generalized to the compositional theorem of multiple categories, see Theorem 7.

Unlike most machine learning theory papers, our paper does not have any assumptions on the data distribution or network structure. Instead, we take the bird’s-eye view that is model oblivious, and only focuses on the structure defined by the pretext task. It is indeed possible that by designing a special network, one may get a more powerful model with better performance. However, we stick with our setting because:

- **Empirically, people do not customize network structures for different tasks.** Instead, they tend to use similar structures like ResNet He et al. (2016) or Transformer Vaswani et al. (2017). By the no free lunch theorem Shalev-Shwartz and Ben-David (2014), if the model does not contain task-specific prior information, it will not be able to completely solve all the tasks. In other words, the standard models are not universally competent, and it is likely that the limitation that we derived from the model oblivious setting, also applies to the practical settings.
- **Pretext task design is a central problem in self-supervised learning.** Currently, there are various datasets floating around, and a wide range of diverse tasks to solve, but the practitioners do not really know what kinds of pretext tasks to pick for solving a given task, or the limitations of each pretrained model. They get the intuition by trial and error with experiments, which are both expensive and noisy. Our framework will help them to think about this problem in a more mathematical and systematic way, and provide guidance for better pretext task design.

2 Related Work

2.1 Self-supervised learning

Self-supervised learning. Recently, researchers have proposed many self-supervised learning algorithms for foundation models, including contrastive methods (Chen et al., 2020; He et al., 2020; Grill et al., 2020; Chen and He, 2021; Noroozi and Favaro, 2016; Zbontar et al., 2021), masked image models (He et al., 2022; Dosovitskiy et al., 2020; Doersch et al., 2015; Pathak et al., 2016), masked language models (Devlin et al., 2018; Raffel et al., 2020), pure language models (Brown et al., 2020; Radford et al., 2018, 2019), and with other pretext tasks (Oord et al., 2018; Gidaris et al., 2018; Clark et al., 2020; Noroozi and Favaro, 2016; Pathak et al., 2017).

Multimodal learning. Self-supervised learning can also be applied to multimodal learning, including text + image (Ramesh et al., 2021; Rombach et al., 2022; Ramesh et al., 2022; Sohl-Dickstein et al., 2015), video + audio (Arandjelovic and Zisserman, 2018). For generating images for the multimodal tasks, diffusion model is the state-of-the-art approach (Rombach et al., 2022; Sohl-Dickstein et al., 2015; Dhariwal and Nichol, 2021; Ho et al., 2020).

Prompt tuning. There are various prompt tuning methods, including the discrete prompts (Brown et al., 2020; Jiang et al., 2020; Shin et al., 2020; Gao et al., 2020) and continuous prompts (Liu et al., 2021; Li and Liang, 2021).

2.2 Theory for deep learning and self-supervised learning

Theory for deep learning has been an active research area recently. Optimization theory focuses on how and why first order methods like stochastic gradient descent finds the local/global optimum of the neural networks (Du et al., 2019b; Arora et al., 2019a; Du et al., 2018; Allen-Zhu et al., 2019b,a; Zou et al., 2020; Li and Yuan, 2017).

Generalization theory focuses on how the performance of the model in the training set transfers to the population distribution (Allen-Zhu et al., 2019a; Arora et al., 2019a; Bartlett et al., 2017, 2020; Yin et al., 2019). Representation theory focuses on the representation power of the neural networks (Hornik et al., 1989; Cybenko, 1989; Raghu et al., 2017). There are also theoretical results on analyzing various aspects of reinforcement learning (Du et al., 2019a,c; Jin et al., 2018; Cai et al., 2020).

Theory for self-supervised learning. There are many interesting theory results for self-supervised learning (Wen and Li, 2021, 2022; Luo et al., 2022; HaoChen et al., 2021; Arora et al., 2019b; Tosh et al., 2021; Lee et al., 2021; Zimmermann et al., 2021). For example, HaoChen et al. (2021); Tan et al. (2023) show that SimCLR is essentially computing spectral graph clustering on the unlabeled dataset. Liu (2022) gives a better rectified flow algorithm with elegant theoretical guarantees for improving the diffusion models.

Application of category theory. Category theory has been applied to many research areas (Fong and Spivak, 2018), including physics (Marquis, 2008; Kuś and Skowron, 2019), design (Censi, 2015), and machine learning (Shiebler et al., 2021; Mahadevan, 2022a,b). The most relevant paper on category theory might be Bradley et al. (2022), where they provide an enriched category theory of natural language.

3 Preliminaries

Category theory is used in almost all areas of mathematics. Here we only introduce the necessary notions for understanding the results of our paper, and skip many important details (e.g., universe and diagram). Curious readers may check Mac Lane (2013); Riehl (2017); Adámek et al. (1990) for more comprehensive introductions.

A category $\mathcal{C}$ has a set of objects $\text{Ob}(\mathcal{C})$, and a set of morphisms $\text{Hom}_{\mathcal{C}}(X, Y)$ from X to Y for every $X, Y \in \text{Ob}(\mathcal{C})$. Given $f \in \text{Hom}_{\mathcal{C}}(X, Y), g \in \text{Hom}_{\mathcal{C}}(Y, Z)$, we define their composition as $g \circ f \in \text{Hom}_{\mathcal{C}}(X, Z)$. Notice that $\circ$ is associative, i.e., $(h \circ g) \circ f = h \circ (g \circ f)$. For every $X \in \text{Ob}(\mathcal{C})$, there exists a unique identity morphism $\text{id}_X \in \text{Hom}_{\mathcal{C}}(X, X)$. A morphism $f : X \rightarrow Y$ is an isomorphism if there exists $g : X \leftarrow Y$ such that $f \circ g = \text{id}_Y$ and $g \circ f = \text{id}_X$. In this case, we say X and Y are isomorphic and write $X \simeq Y$.

Given a category $\mathcal{C}$, we define its opposite $\mathcal{C}^{\text{op}}$ by setting $\text{Ob}(\mathcal{C}^{\text{op}}) = \text{Ob}(\mathcal{C})$ and $\text{Hom}_{\mathcal{C}^{\text{op}}}(X, Y) = \text{Hom}_{\mathcal{C}}(Y, X)$. Moreover, given $f \in \text{Hom}_{\mathcal{C}^{\text{op}}}(X, Y), g \in \text{Hom}_{\mathcal{C}^{\text{op}}}(Y, Z)$, the new composition is $g \overset{\text{op}}{\circ} f = f \circ g \in \text{Hom}_{\mathcal{C}^{\text{op}}}(X, Z)$.

We define **Set** to be the category of sets, where the objects are sets, and $\text{Hom}_{\text{Set}}(X, Y)$ is the set of all functions with domain X and codomain Y . Notice that we ignore the subtleties about the universe for better presentation, so here just assume that **Set** does not contain strange objects like a set containing all sets.

A functor is a structure-preserving map between categories. Given two categories $\mathcal{C}$ and $\mathcal{C}'$, a functor $F : \mathcal{C} \rightarrow \mathcal{C}'$ assigns to each object $X \in \text{Ob}(\mathcal{C})$ an object $F(X) \in \text{Ob}(\mathcal{C}')$, and to each morphism $f \in \text{Hom}_{\mathcal{C}}(X, Y)$ a morphism $F(f) \in \text{Hom}_{\mathcal{C}'}(F(X), F(Y))$. It preserves identities and composition: for every $X \in \text{Ob}(\mathcal{C})$, $F(\text{id}_X) = \text{id}_{F(X)}$, and for every pair of composable morphisms $f : X \rightarrow Y, g : Y \rightarrow Z$, we have $F(g \circ f) = F(g) \circ F(f)$. A functor $F : \mathcal{C} \rightarrow \mathcal{C}'$ is faithful if, for every $X, Y \in \text{Ob}(\mathcal{C})$, the induced map $F_{X,Y} : \text{Hom}_{\mathcal{C}}(X, Y) \rightarrow \text{Hom}_{\mathcal{C}'}(F(X), F(Y))$ is injective. It is full if $F_{X,Y}$ is surjective for every $X, Y \in \text{Ob}(\mathcal{C})$. It is fully faithful if it is both full and faithful. In this paper, we call F an embedding functor if it is faithful and injective on objects. We call F a full embedding if it is full, faithful, and injective on objects.

The morphisms between functors are called natural transformations. Given two functors $F_1, F_2 : \mathcal{C} \rightarrow \mathcal{C}'$, a natural transformation $\theta : F_1 \Rightarrow F_2$ assigns to every object $X \in \text{Ob}(\mathcal{C})$ a morphism $\theta_X : F_1(X) \rightarrow F_2(X)$ in $\mathcal{C}'$, such that for every morphism $f : X \rightarrow Y$ in $\mathcal{C}$, the following naturality condition holds: $\theta_Y \circ F_1(f) = F_2(f) \circ \theta_X$. We write $F_1 \simeq F_2$ if there exists a natural isomorphism $\theta : F_1 \Rightarrow F_2$, i.e., a natural transformation whose components θ_X are isomorphisms in $\mathcal{C}'$. A functor $F : \mathcal{C} \rightarrow \mathcal{C}'$ is an isomorphism of categories if there exists a functor $G : \mathcal{C}' \rightarrow \mathcal{C}$ such that $G \circ F = \text{id}_{\mathcal{C}}, F \circ G = \text{id}_{\mathcal{C}'}$. In this case, we say $\mathcal{C}$ and $\mathcal{C}'$ are isomorphic and write $\mathcal{C} \simeq \mathcal{C}'$.

A category $\mathcal{A}$ is a subcategory of a category $\mathcal{B}$, if

1. $\text{Ob}(\mathcal{A}) \subseteq \text{Ob}(\mathcal{B})$,
2. for each $A, A' \in \text{Ob}(\mathcal{A})$, $\text{Hom}_{\mathcal{A}}(A, A') \subseteq \text{Hom}_{\mathcal{B}}(A, A')$,
3. for each $\mathcal{A}$ -object A , the $\mathcal{B}$ -identity on A is the $\mathcal{A}$ -identity on A ;
4. the composition law in $\mathcal{A}$ is the restriction of the composition law in $\mathcal{B}$ to the morphisms of $\mathcal{A}$.

Moreover, $\mathcal{A}$ is a full subcategory of $\mathcal{B}$, if it is a subcategory of $\mathcal{B}$ and for each $A, A' \in \text{Ob}(\mathcal{A})$, $\text{Hom}_{\mathcal{A}}(A, A') = \text{Hom}_{\mathcal{B}}(A, A')$. Every subcategory $\mathcal{A}$ of category $\mathcal{B}$ naturally defines an inclusion functor $E : \mathcal{A} \hookrightarrow \mathcal{B}$, which is an embedding. For such embeddings, we have the following lemma.

Lemma 1 (Adámek et al. (1990)). *A functor $F : \mathcal{C} \rightarrow \mathcal{B}$ is a full embedding if and only if there exists a full subcategory $\mathcal{A}$ of $\mathcal{B}$ with inclusion functor $E : \mathcal{A} \hookrightarrow \mathcal{B}$ and an isomorphism $G : \mathcal{C} \rightarrow \mathcal{A}$ with $F = E \circ G$.*

3.1 Contrastive methods

Contrastive methods (like SimCLR (Chen et al., 2020) and MoCo (He et al., 2020)) take a query point q , and one positive sample p_1 to q , as well as $N - 1$ other objects $\{p_i\}_{i=2}^N$. Then they aim to find a function f to minimize the following loss function:

$$\mathcal{L}(q, p_1, \{p_i\}_{i=2}^N) = -\log \frac{\exp(\text{sim}(f(q), f(p_1))/\tau)}{\sum_{i=1}^N \exp(\text{sim}(f(q), f(p_i))/\tau)}$$

The actual loss function takes summation over different q , and τ is a temperature hyperparameter. $\text{sim}(Z_i, Z_j) = -\|Z_i - Z_j\|^2/2$, so $\exp(\text{sim}(f(q), f(p_1))/\tau)$ can be seen as a Gaussian kernel for $f(q) - f(p_1)$, whose normalizing constant $\frac{1}{\sqrt{2\pi\tau}}$ is canceled out in the fraction.

3.2 Language Modeling

Given a sentence $S = (w_1, w_2, \dots, w_N)$, where each w_i is a token in the sentence, language modeling aims to learn a function f that estimates the probability distribution over the next token based on the preceding context. The objective is to minimize the negative log-likelihood of the observed sequence of tokens. This is typically expressed through the following loss function:

$$\mathcal{L}(S) = -\sum_{i=2}^N \log f(w_i \mid w_1, w_2, \dots, w_{i-1})$$

Here, $f(w_i \mid w_1, w_2, \dots, w_{i-1})$ represents the conditional probability that the next token in the sequence is w_i , given the previous tokens $w_1, w_2, \dots, w_{i-1}$. The function f takes the preceding tokens as input and outputs a probability distribution over the possible next tokens. The goal is to maximize the likelihood that the model assigns to the actual sequence of tokens observed in the sentence.

3.3 Reproducing Kernel Hilbert Space (RKHS)

Given two objects $X, Y \in \mathcal{C}$, consider a feature map $f : \mathcal{C} \rightarrow \mathcal{H}$, where the feature space $\mathcal{H}$ is usually much larger than $\mathcal{C}$. We may define a kernel k that measures the similarity of X and Y as $k(x, y) \triangleq \langle f(x), f(y) \rangle_{\mathcal{H}}$, i.e., the inner product between the two object after mapping them to the feature space. For any vector $T \in \mathcal{H}$, it also corresponds to a function $T(\cdot) : \mathcal{C} \rightarrow \mathbb{R}$, defined as $T(x) = \langle T, f(x) \rangle_{\mathcal{H}}$. Specifically, $f(y)$ as a vector in $\mathcal{H}$ also represents the function $k(\cdot, y) : \mathcal{C} \rightarrow \mathbb{R}$, because for any $x \in \mathcal{C}$, we have $k(x, y) = \langle f(x), f(y) \rangle_{\mathcal{H}}$. Formally, we have:

Definition 1 (Reproducing kernel Hilbert space). *Let $\mathcal{H}$ be a Hilbert space of $\mathbb{R}$ -valued functions defined on a non-empty set $\mathcal{C}$. A function $k : \mathcal{C} \times \mathcal{C} \rightarrow \mathbb{R}$ is called a reproducing kernel of $\mathcal{H}$, and $\mathcal{H}$ is a reproducing kernel Hilbert space (RKHS), if k satisfies*

- $\forall x \in \mathcal{C}, k(\cdot, x) \in \mathcal{H}$,
- $\forall x \in \mathcal{C}, \forall f \in \mathcal{H}, \langle f, k(\cdot, x) \rangle_{\mathcal{H}} = f(x)$.

4 Categorical Framework of Supervised Learning

In supervised learning, we have a population distribution $D = (D_X, D_Y)$, representing the ground truth distribution of the input and output data points. The training set $(X_{\text{train}}, Y_{\text{train}})$ and test set $(X_{\text{test}}, Y_{\text{test}})$ are uniformly sampled from (D_X, D_Y) . We hope to learn a function $f : X \rightarrow Y$ so that $f(x)$ accurately predicts the label $x \in X$. We also define a loss function $L(f, x, y)$ to measure the distance between the prediction $f(x)$ and the correct label, which is hopefully close to 0. The loss function on a dataset (X, Y) is $L(f, X, Y) \triangleq \mathbb{E}_{(x, y) \sim (X, Y)} L(f, x, y)$. The task of supervised learning, is to minimize the population loss $L_{\text{population}} \triangleq \mathbb{E}_{(X, Y) \sim (D_X, D_Y)} L(f, X, Y)$, with access of the training set $(X_{\text{train}}, Y_{\text{train}})$.

Using the language of category theory, we have two categories $\mathcal{X}$ and $\mathcal{Y}$, with objects x, y as input and output data points, respectively. To avoid confusion, below we switch notation from x, y to X, Y to represent the objects, as later we will not use D_X, D_Y any more. The population distribution can be seen as a functor F from $\mathcal{X}$ to $\mathcal{Y}$, representing

the correct label Y given the input X . Due to the inherent noise in the real world, there may not always exist the unique correct label for each input X . In other words, the Bayesian optimal solution does not give zero population loss. We will discuss this issue in Section 7.2, and for now let us simply assume that the unique correct labels always exist.

With this formulation, training/test set can be seen as samples over objects in $\mathcal{X}$ with correct labels in $\mathcal{Y}$. Therefore, supervised learning investigates the following question: can we learn a functor F with samples of $X \in \mathcal{X}$ and $F(X) \in \mathcal{Y}$?

Consider the special case that both categories are discrete, meaning that the only morphisms existed are identity morphisms like $\text{id}_X \in \text{Hom}_{\mathcal{X}}(X, X)$ and $\text{id}_Y \in \text{Hom}_{\mathcal{Y}}(Y, Y)$. In other words, both categories are sets without any morphisms between different objects. In this case, learning F is impossible, because it maps a set to another set without any prior knowledge. The no free lunch theorem tells us that unless we have sample size larger than half of the set size, the functor we learned will have constant generalization error with constant probability.

Generalization theory deals with this problem by assuming that F is in a predefined hypothesis class $\mathcal{H}$, or close to a function in $\mathcal{H}$. In category theory, we do not add restrictions to the functors, but to the structure of the categories instead. For example, if we know $\mathcal{X}$ has a linear structure, and F preserves the structure, it is possible to learn F with a few samples.

This formulation seems useless, as it does not even characterize the loss function, which is crucial in supervised learning. This is because **category theory takes the bird’s-eye view**. Our categorical framework will not replace the classical framework, or generate better supervised learning algorithms. Instead, it treats the existing supervised learning algorithms a subroutine or a building block, which can be incorporated into a bigger picture. Therefore, it does not care about the loss functions or optimization process, which are treated as the implementation details. Instead, it focuses on the structure of the categories and functors, and tries to understand whether certain functors are learnable or not. It also investigates whether the ability of mastering at one or more tasks can be generalized to other tasks.

5 Categorical Framework of Self-supervised Learning

5.1 Pretext tasks and relationships

In self-supervised learning, we have a population distribution D of data points without labels, and try to extract useful information from D by setting up pretext tasks. Different pretext tasks give different kinds of relationships between two data points, for example:

- Contrastive methods. Contrastive methods like SimCLR (Chen et al., 2020) modify data point $X \in D$ to get two semantically similar copies $X', X'' \in D$ as the positive pair, and pick another different $Y \in D$ as the negative sample. The pretext task says X', X'' should be close and both of them should be far away from Y . It has been proved that this task is equivalent to learning similarities between data points (Tan et al., 2023; HaoChen et al., 2021).
- Language modeling. For every sentence (or sub-sentence) $X \in D$, the pretext task models the distribution of the next token following X . By iteratively applying this task, we are learning the probability of a sentence $Y \in D$ is successive to sentence X , where X is a prefix of Y .

We remark that the relationships defined by the pretext tasks are not necessarily consistent, therefore cannot be directly converted into morphisms of a category. A natural question is:

Can we find a category that approximates the relationships introduced by pretext tasks?

To answer this question, we should first define how a category is induced by a foundation model.

5.2 Foundation model induced category

A foundation model maps an input to an internal representation, and subsequent prediction heads or task-specific readouts compute outputs from this representation. In this paper, we write $f : D \rightarrow \mathcal{H}$ for the representation map of a trained foundation model. When the pretext task provides additional fixed forward operations on representation states, such as similarity computations, prediction heads, or autoregressive state transitions, we regard these operations as part of the model’s representation interface.

In general, different inputs may have the same representation, and such inputs cannot be distinguished by any readout that only has access to the representation. Therefore, the category induced by a foundation model should be defined on observable representation states, or equivalently on inputs modulo representation equivalence.

Definition 2 (Foundation model induced category). *Let $f : D \rightarrow \mathcal{H}$ be a foundation model representation. Define an equivalence relation on D by*

$$X \sim_f X' \iff f(X) = f(X').$$

Let $\mathcal{H}_f \triangleq f(D) \subseteq \mathcal{H}$ be the set of reachable representation states. We say f induces a category $\mathcal{C}_f$ if there exists a data-oblivious readout

$$k_f : \mathcal{H}_f \times \mathcal{H}_f \rightarrow \mathbf{Set}$$

and a family of composition maps

$$\circ_{X,Y,Z} : k_f(f(Y), f(Z)) \times k_f(f(X), f(Y)) \rightarrow k_f(f(X), f(Z)),$$

such that:

- $\text{Ob}(\mathcal{C}_f) \triangleq D/\sim_f$. By abuse of notation, we write an object as X , and identify it with its reachable representation state $f(X) \in \mathcal{H}_f$.
- For any $X, Y \in \text{Ob}(\mathcal{C}_f)$,

$$\text{Hom}_{\mathcal{C}_f}(X, Y) \triangleq k_f(f(X), f(Y)).$$

This definition is well-defined because f is constant on each equivalence class.

- *Composable: for any $X, Y, Z \in \text{Ob}(\mathcal{C}_f)$, $g \in \text{Hom}_{\mathcal{C}_f}(X, Y)$, and $h \in \text{Hom}_{\mathcal{C}_f}(Y, Z)$, the composition $h \circ g$ is an element of $\text{Hom}_{\mathcal{C}_f}(X, Z)$.*
- *Associative: for any composable morphisms g, h, u ,*

$$(u \circ h) \circ g = u \circ (h \circ g).$$

- *Unique identity: for every $X \in \text{Ob}(\mathcal{C}_f)$, there exists a unique identity morphism*

$$\text{id}_X \in \text{Hom}_{\mathcal{C}_f}(X, X)$$

such that

$$g \circ \text{id}_X = g \quad \text{and} \quad \text{id}_Y \circ g = g$$

for every $g \in \text{Hom}_{\mathcal{C}_f}(X, Y)$.

Here, data-oblivious means that the readout k_f and the composition law are specified before seeing the empirical data distribution. The subscript f indicates that the readout may depend on the trained representation and on the pretext-task structure associated with the model. Thus the induced category only contains distinctions that are observable from the representation. If two inputs are collapsed by f , then they are identified as the same object in $\mathcal{C}_f$.

A useful general way to construct such readouts is through a pretext-task-induced action on the representation space. Let M_f be a monoid with unit e , determined by the fixed interface of the trained model, and suppose there is a right action

$$\alpha_f : \mathcal{H}_f \times M_f \rightarrow \mathcal{H}_f$$

satisfying

$$\alpha_f(h, e) = h, \quad \alpha_f(\alpha_f(h, m), n) = \alpha_f(h, mn).$$

The subscript f emphasizes that the action may use the trained model's fixed operations on representation states.

Given such an action, define

$$k_f(f(X), f(Y)) \triangleq \{m \in M_f : \alpha_f(f(X), m) = f(Y)\}.$$

For

$$m \in k_f(f(X), f(Y)), \quad n \in k_f(f(Y), f(Z)),$$

define

$$n \circ m \triangleq mn.$$

Then

$$\alpha_f(f(X), mn) = \alpha_f(\alpha_f(f(X), m), n) = f(Z),$$

so $mn \in k_f(f(X), f(Z))$. The identity morphism at X is the monoid unit e . The action law implies composability and associativity, and the unit law gives the identity morphisms. Hence any such action induces a category on the reachable representation states.

This construction should be viewed as a representation-level version of the relationships specified by the pretext task. For contrastive learning, the relevant action is translation in feature space. For language modeling, the relevant action is the autoregressive transition obtained by appending tokens to a state.

As we will see shortly, category theory provides many great tools to analyze our model. However, we still need to show Definition 2 is not vacuous, as demonstrated by the following two theorems.

Theorem 1. *Contrastive methods induce categories.*

Proof. For contrastive methods, the foundation model maps each input X to a feature vector $f(X) \in \mathcal{H}$. Let

$$\mathcal{H}_f = f(D) \subseteq \mathcal{H}$$

be the set of reachable feature states. As in Definition 2, we regard the objects of the induced category as inputs modulo representation equivalence, and write them as X, Y, Z .

Let

$$M_f = (\mathcal{H}, +)$$

be the additive monoid of the feature space. It acts on reachable feature states by translation:

$$\alpha_f(z, v) = z + v.$$

Define

$$k_f(f(X), f(Y)) \triangleq \{v \in \mathcal{H} : \alpha_f(f(X), v) = f(Y)\}.$$

Equivalently,

$$\text{Hom}_{\mathcal{C}_f}(X, Y) = k_f(f(X), f(Y)) = \{f(Y) - f(X)\}.$$

This is well-defined on objects of $D/\sim_f$ because f is constant on each equivalence class.

For

$$g = f(Y) - f(X) \in \text{Hom}_{\mathcal{C}_f}(X, Y), \quad h = f(Z) - f(Y) \in \text{Hom}_{\mathcal{C}_f}(Y, Z),$$

define the composition by

$$h \circ g \triangleq g + h.$$

Then

$$h \circ g = (f(Y) - f(X)) + (f(Z) - f(Y)) = f(Z) - f(X) \in \text{Hom}_{\mathcal{C}_f}(X, Z).$$

Therefore composition is well-defined.

We now verify the category axioms.

- **Composable.** The calculation above shows that if $g \in \text{Hom}_{\mathcal{C}_f}(X, Y)$ and $h \in \text{Hom}_{\mathcal{C}_f}(Y, Z)$, then $h \circ g \in \text{Hom}_{\mathcal{C}_f}(X, Z)$.
- **Associative.** Composition is vector addition in $\mathcal{H}$, and vector addition is associative. Hence for any composable morphisms g, h, u ,

$$(u \circ h) \circ g = u \circ (h \circ g).$$

- **Unique identity.** For every object X , the identity morphism is the zero vector

$$\text{id}_X = \mathbf{0} \in \text{Hom}_{\mathcal{C}_f}(X, X),$$

because

$$\alpha_f(f(X), \mathbf{0}) = f(X).$$

For any $g \in \text{Hom}_{\mathcal{C}_f}(X, Y)$,

$$g \circ \text{id}_X = g, \quad \text{id}_Y \circ g = g.$$

Since

$$\text{Hom}_{\mathcal{C}_f}(X, X) = \{f(X) - f(X)\} = \{\mathbf{0}\},$$

this identity morphism is unique.

Therefore contrastive methods with foundation model f induce a category $\mathcal{C}_f$ on reachable feature states. $\square$

Theorem 2. *Language modeling induces categories.*

Proof. Let V be the token vocabulary and let V^* be the free monoid of finite token strings under concatenation. The unit element is the empty string ϵ .

For language modeling, we take the foundation model representation $f(X) \in \mathcal{H}$ to be an autoregressive state of the model after reading the context X . Such a state may be, for example, a sufficient hidden state or a key-value cache state. The important property is that the model can be advanced forward after appending tokens.

Let

$$M_f = V^*.$$

The fixed autoregressive transition of the trained language model gives a right action

$$\alpha_f : \mathcal{H}_f \times V^* \rightarrow \mathcal{H}_f$$

satisfying

$$\alpha_f(f(X), u) = f(Xu)$$

for every context X and every token string $u \in V^*$.

Indeed,

$$\alpha_f(f(X), \epsilon) = f(X),$$

and for any $u, v \in V^*$,

$$\alpha_f(\alpha_f(f(X), u), v) = \alpha_f(f(Xu), v) = f(Xuv) = \alpha_f(f(X), uv).$$

Therefore α_f is a right action of V^* on reachable autoregressive states.

For each object Y , denote by P_Y the distribution of all future continuations generated after the model is in state $f(Y)$. This distribution is determined only by the state $f(Y)$ and the model's fixed next-token readout, not by how the state $f(Y)$ was reached.

More explicitly, let

$$o_f : \mathcal{H}_f \rightarrow \Delta(V)$$

be the next-token readout of the trained language model. Then P_Y is the autoregressive continuation law starting from $f(Y)$. For a finite continuation

$$w = w_1 w_2 \cdots w_m \in V^*,$$

its prefix probability is

$$P_Y(w_1, \dots, w_m) = \prod_{i=1}^m o_f(\alpha_f(f(Y), w_{<i}))(w_i),$$

where $w_{<i} = w_1 \cdots w_{i-1}$. If one includes an end-of-sequence token, this gives a probability distribution over finite continuations; otherwise it defines the usual cylinder probabilities over infinite continuations.

Now define the readout k_f by

$$k_f(f(X), f(Y)) \triangleq \begin{cases} \{(X, Y, P_Y)\}, & \text{if there exists } u \in V^* \text{ such that } \alpha_f(f(X), u) = f(Y), \\ \emptyset, & \text{otherwise.} \end{cases}$$

Equivalently,

$$\text{Hom}_{\mathcal{C}_f}(X, Y) = \begin{cases} \{(X, Y, P_Y)\}, & \text{if } Y \text{ is reachable from } X \text{ by the forward autoregressive transition,} \\ \emptyset, & \text{otherwise.} \end{cases}$$

Thus a morphism from X to Y records the future-continuation distribution available after reaching Y . It does not record the probability of generating Y from X .

Given

$$g = (X, Y, P_Y) \in \text{Hom}_{\mathcal{C}_f}(X, Y), \quad h = (Y, Z, P_Z) \in \text{Hom}_{\mathcal{C}_f}(Y, Z),$$

define the composition function by

$$h \circ g \triangleq (X, Z, P_Z).$$

We verify that this is well-defined. Since $g \in \text{Hom}_{\mathcal{C}_f}(X, Y)$, there exists $u \in V^*$ such that

$$\alpha_f(f(X), u) = f(Y).$$

Since $h \in \text{Hom}_{\mathcal{C}_f}(Y, Z)$, there exists $v \in V^*$ such that

$$\alpha_f(f(Y), v) = f(Z).$$

Therefore, by the action law,

$$\alpha_f(f(X), uv) = \alpha_f(\alpha_f(f(X), u), v) = \alpha_f(f(Y), v) = f(Z).$$

Hence there exists a token string $uv \in V^*$ such that

$$\alpha_f(f(X), uv) = f(Z).$$

By the definition of k_f , this implies

$$(X, Z, P_Z) \in \text{Hom}_{\mathcal{C}_f}(X, Z).$$

Therefore composition is well-defined.

We now verify the category axioms.

- **Composable.** The argument above shows that if $g \in \text{Hom}_{\mathcal{C}_f}(X, Y)$ and $h \in \text{Hom}_{\mathcal{C}_f}(Y, Z)$, then $h \circ g \in \text{Hom}_{\mathcal{C}_f}(X, Z)$.
- **Associative.** Consider composable morphisms

$$g = (X, Y, P_Y), \quad h = (Y, Z, P_Z), \quad r = (Z, W, P_W).$$

Then

$$(r \circ h) \circ g = (X, W, P_W),$$

and

$$r \circ (h \circ g) = (X, W, P_W).$$

Hence

$$(r \circ h) \circ g = r \circ (h \circ g).$$

- **Unique identity.** For every object X , since

$$\alpha_f(f(X), \epsilon) = f(X),$$

we have

$$\text{Hom}_{\mathcal{C}_f}(X, X) = \{(X, X, P_X)\}.$$

Define

$$\text{id}_X \triangleq (X, X, P_X).$$

For any

$$g = (X, Y, P_Y) \in \text{Hom}_{\mathcal{C}_f}(X, Y),$$

we have

$$g \circ \text{id}_X = (X, Y, P_Y) = g.$$

Similarly,

$$\text{id}_Y \circ g = (X, Y, P_Y) = g.$$

Since $\text{Hom}_{\mathcal{C}_f}(X, X)$ is a singleton, this identity morphism is unique.

Therefore language modeling with foundation model f induces a category $\mathcal{C}_f$ on reachable autoregressive states. $\square$

5.3 Category approximates pretext task

For the two constructions above, once the corresponding readout k_f and forward interface are fixed, any choice of model parameters gives a well-defined category, although the resulting category may be degenerate or poorly aligned with the intended pretext-task structure. In particular, even when the weights of f are random, there still exists a possibly strange category for f . Therefore, if we name f with its weight w_t as f_t for the step t during the training process, then in every step t we will get a difference induced category $\mathcal{C}_{f_t}$. In other words, the training process of f_t also corresponds to the training process of categories $\mathcal{C}_{f_t}$. Hence, as the answer to the question in Section 5.1, we can find a category that approximates the relationships introduced by the pretext task through the learning process.

A common question might arise: “Why approximate pretext tasks using categories in the first place?” However, as explained earlier, since foundation models **always** correspond to well defined categories in the two scenarios, the right question should be “how is it possible that we can always learn a good category when pretext tasks are not consistent?”

The answer lies in the nature of the learning process: it ensures that $\mathcal{C}_{f_t}$ is always a well-defined category, even if the morphisms do not perfectly align with the pretext task. For example, as shown in Tan et al. (2023), the model learns the spectral clustering solution of the similarity graph derived from the pretext task. However, the final solution may establish pairwise similarities between all objects, which may not precisely match the original pretext task.

Consequently, whether the pretext task is consistent or not does not impact our analysis, since the model ultimately learns a consistent category that best approximates the relationships specified by the pretext task. Given any pretext task dataset and a learning algorithm, such a category always exists¹. For the rest of the paper, we will refer to the category that best matches the pretext task as the “ideal category $\mathcal{C}$ ” and the corresponding foundation model as the “ideal foundation model” for $\mathcal{C}$.

5.4 Yoneda Lemma

By the previous definitions, when the pretext task is given, even if the training process is optimal, an ideal foundation model f is all we expect to get, which corresponds to an ideal category $\mathcal{C}$. We raise the following question,

Is f guaranteed to have some property, if it is ideal with a category $\mathcal{C}$?

This is the main motivation of our paper. Below we first introduce an important notion called Yoneda embedding.

Definition 3 (Yoneda embedding $h_{\mathcal{C}}$). *Given any $X \in \mathcal{C}$, $h_{\mathcal{C}}(X) \triangleq \text{Hom}_{\mathcal{C}}(X, \cdot)$, which takes input $Y \in \mathcal{C}$ and outputs $\text{Hom}_{\mathcal{C}}(X, Y)$.²*

Define $\mathcal{C}^{\vee}$ as the category of functors from $\mathcal{C}^{op}$ to $\mathbf{Set}^{op}$. Then $h_{\mathcal{C}} : \mathcal{C} \rightarrow \mathcal{C}^{\vee}$ is well-defined. Notice that we can define $\text{Hom}_{\mathcal{C}}(\cdot, X)$ similarly, which gives another kind of Yoneda embedding.

How do we represent $\mathcal{C}^{\vee}$ empirically? To answer this question, we should first learn arguably the most important theorem in category theory, as follows.

Lemma 2 (Yoneda lemma). *For $A \in \mathcal{C}^{\vee}$, and $X \in \mathcal{C}$, $\text{Hom}_{\mathcal{C}^{\vee}}(A, h_{\mathcal{C}}(X)) \simeq A(X)$.*

In category theory, when two objects (or functors) are isomorphic, we treat them as equal. The technical details for making them equal to each other is omitted, as we observe that this is not a problem for modern networks empirically. For example, if there exists an isomorphism between two objects X and Y , so that $\phi(X) = Y$, empirically a deep neural network can easily fit this ϕ . More generally, techniques like rectified flow (Liu, 2022) can fit very complicated isomorphisms between random noise and images, using neural network with ODE.

Let $\mathcal{C} = \mathcal{C}_f$ be the category induced by the foundation model f and the readout k_f . By construction, the objects of $\mathcal{C}$ are observable representation states, written as X, Y , and

$$\text{Hom}_{\mathcal{C}}(X, Y) = k_f(f(X), f(Y)).$$

Once $\mathcal{C}$ is constructed, it has the canonical Yoneda embedding

$$h_{\mathcal{C}} : \mathcal{C} \rightarrow \mathcal{C}^{\vee}, \quad h_{\mathcal{C}}(X) = \text{Hom}_{\mathcal{C}}(X, \cdot).$$

Strictly speaking, the feature vector $f(X) \in \mathcal{H}_f$ is not itself a functor in $\mathcal{C}^{\vee}$. Rather, since objects of $\mathcal{C}$ are identified with reachable representation states, there is a canonical realization map

$$\Phi_f : \mathcal{H}_f \rightarrow \mathcal{C}^{\vee}$$

defined by

$$\Phi_f(f(X)) \triangleq h_{\mathcal{C}}(X).$$

This is well-defined because objects are taken modulo representation equivalence: if $f(X) = f(X')$, then X and X' denote the same object of $\mathcal{C}$.

Under this realization, $f(X)$ is interpreted as a code for the representable functor $h_{\mathcal{C}}(X)$. Therefore,

$$k_f(f(X), f(Y)) = \text{Hom}_{\mathcal{C}}(X, Y) = h_{\mathcal{C}}(X)(Y).$$

¹There may even exist multiple different categories due to the randomness of the training process, which can be analyzed similarly and we omit the discussion here.

²In the notation of Masaki Kashiwara (2006), this functor is denoted by $k_{\mathcal{C}}$ rather than $h_{\mathcal{C}}$. We use $h_{\mathcal{C}}$ here to avoid conflict with the kernel notation k .

By the Yoneda lemma, we also have

$$h_{\mathcal{C}}(X)(Y) \simeq \text{Hom}_{\mathcal{C}^\vee}(h_{\mathcal{C}}(X), h_{\mathcal{C}}(Y)).$$

Equivalently,

$$k_f(f(X), f(Y)) \simeq \text{Hom}_{\mathcal{C}^\vee}(\Phi_f(f(X)), \Phi_f(f(Y))).$$

From now on, when no confusion arises, we suppress the realization map Φ_f from the notation and identify each reachable representation $f(X)$ with its realization

$$\Phi_f(f(X)) = h_{\mathcal{C}}(X)$$

in $\mathcal{C}^\vee$. Thus expressions such as $f(X) \in \mathcal{C}^\vee$, $T \simeq f(P)$, or $k_f(T, f(X))$ should be understood after applying this canonical realization.

Equivalently, after this point we use $h_{\mathcal{C}}$ as the Yoneda-realized form of the ideal foundation model for the category $\mathcal{C}$. That is, when we refer to $h_{\mathcal{C}}$ as a foundation model, we mean the realized representation

$$h_{\mathcal{C}} \simeq \Phi_f \circ f,$$

rather than the raw feature map $f : D \rightarrow \mathcal{H}$ itself. This is only a notational convention: the actual logical direction remains

$$(f, k_f) \implies \mathcal{C}_f \implies h_{\mathcal{C}},$$

and the feature vector $f(X) \in \mathcal{H}_f$ is not literally a functor before this realization is applied.

Under this convention, $f(X)$ encodes, and will be identified with, the representable functor $h_{\mathcal{C}}(X) = \text{Hom}_{\mathcal{C}}(X, \cdot)$, which takes input $Y \in \mathcal{C}$ and outputs $\text{Hom}_{\mathcal{C}}(X, Y)$.

The above interpretation may look redundant at first glance, because after applying the convention above, one may simply write $f(X)$ as if it were $h_{\mathcal{C}}(X)$. The reason for first introducing Φ_f is to make the logical direction explicit. We do not start from a given category $\mathcal{C}$ and then choose a representation of its objects. Instead, we start from the foundation model f and the readout k_f , construct the category $\mathcal{C}_f$, and only then interpret the reachable representation $f(X)$ as the representable functor $h_{\mathcal{C}}(X)$.

5.5 Downstream tasks

After the model is trained, we apply it to the downstream tasks with two different approaches: prompt tuning and fine tuning. In this section, we investigate the power of both approaches, with different outcomes. We first define what we mean by solving a downstream task.

Definition 4 (Task). *A task T is a functor in $\mathcal{C}^\vee$.*

Definition 5 (Task solving). *We say the model solves a task T , if the downstream functor induced by the model is isomorphic to T . Equivalently, for any input $X \in \mathcal{C}$, the model outputs a solution that is isomorphic to $T(X)$.*

Given a task defined as a functor $T \in \mathcal{C}^\vee$, prompt tuning means we freeze the parameters of the model, and only use a task specific prompt P (usually in text or image), followed by the actual input of the task X , to get the output $T(X)$. Therefore, the prompt P and input X are the only two inputs to the model. By Lemma 2, if we directly send T and $h_{\mathcal{C}}(X)$ to k_f , we have

$$T(X) \simeq \text{Hom}_{\mathcal{C}^\vee}(T, h_{\mathcal{C}}(X)) \triangleq k_f(T, f(X)). \quad (1)$$

That is, k_f can accurately compute $T(X)$ from these two representations. However, in prompt tuning, the task representation must itself be realized by a prompt. Therefore, instead of sending T directly to k_f , we send the representation $h_{\mathcal{C}}(P)$ induced by the prompt P . This brings up another important definition in category theory.

Definition 6 (Representable functor). *A functor $T \in \mathcal{C}^\vee$ is representable if there is an isomorphism $h_{\mathcal{C}}(X) \simeq T$ for some $X \in \mathcal{C}$. Such an object X is called a representative of T .*

Based on this definition, we have the following characterization of prompt tuning.

Theorem 3 (Power on prompt tuning). *f can solve T with prompt tuning, if and only if task $T \in \mathcal{C}^\vee$ is representable. When T is representable, the optimal prompt is the representative of T .*

Proof. When the task T is representable, let $P \in \mathcal{C}$ be a representative of T , so that $T \simeq h_{\mathcal{C}}(P)$. Then by (1),

$$T(X) \simeq \text{Hom}_{\mathcal{C}^\vee}(T, h_{\mathcal{C}}(X)) \simeq \text{Hom}_{\mathcal{C}^\vee}(h_{\mathcal{C}}(P), h_{\mathcal{C}}(X)) \triangleq k_f(f(P), f(X)).$$

Hence the model solves T with prompt P .

On the other hand, suppose the model solves T with prompt tuning using a prompt P . Then for any input $X \in \mathcal{C}$, the model output is

$$k_f(f(P), f(X)) \triangleq \text{Hom}_{\mathcal{C}^\vee}(h_{\mathcal{C}}(P), h_{\mathcal{C}}(X)) \simeq h_{\mathcal{C}}(P)(X).$$

Therefore, the downstream functor induced by the prompt P is exactly $h_{\mathcal{C}}(P)$. Since the model solves T , this functor must be isomorphic to T . Hence T is representable. $\square$

Remark. There are some interesting results on continuous prompts (Liu et al., 2021; Li and Liang, 2021), which directly tunes the prompts in the feature space of the neural network, therefore the resulting prompt is not a representation of any real words/sentences. By doing this, it is possible to obtain more power than the representable tasks, but the enhancement depends on the expressive power of the feature space. Below we provide a simple application of Theorem 3.

Corollary 1 (Pretext-only guarantee of rotation prediction). *For the pretext task of predicting image rotations (Gidaris et al., 2018), the only structure guaranteed by the pretext objective itself is the four-fold rotational structure generated by*

$$0^\circ, 90^\circ, 180^\circ, 270^\circ.$$

Consequently, without additional assumptions on the architecture, optimizer, data distribution, or implicit bias of the learned representation, prompt tuning is not guaranteed to solve content-level downstream tasks such as segmentation or classification.

Proof. The rotation-prediction pretext task only requires the model to distinguish the four rotations of an image:

$$0^\circ, 90^\circ, 180^\circ, 270^\circ.$$

Therefore, the structure specified by the pretext task is the four-fold rotational action generated by these transformations.

Since we make no assumptions on the implicit bias of the training algorithm or on additional information preserved by the representation, the only structure that is guaranteed to be learned from the pretext task alone is this rotational structure. A learned representation may in fact retain more information, such as semantic or spatial content. However, such information is not forced by the rotation-prediction objective and therefore cannot be used as a pretext-only guarantee.

Under the pretext-guaranteed rotational structure, the representable functors

$$h_{\mathcal{C}}(X) = \text{Hom}_{\mathcal{C}}(X, \cdot)$$

can only express relationships determined by the four-fold rotation action. By Theorem 3, prompt tuning can solve a downstream task only when that task is representable in the induced category. Hence, from the rotation pretext task alone, we can only guarantee prompt-tuning solutions for tasks representable by this rotational structure. $\square$

When applying Theorem 3 to large language models, the statement is slightly different, because we have to send “prompt + X ” to the model.

Consider task T that takes X as the input, and outputs the distribution of sentences representing correct answers. For any prompt P , we can define a task T_P , such that $T_P(P + X) = T(X)$, and $T_P(X') = \emptyset$ if X' is not started with P . This can be a bit confusing, but T_P is defined by T , and not necessarily related to the prompt P . For example, if T is “count the number of words in X ”, but prompt P = “repeat the following sentence:”, then T_P (“repeat the following sentence: hello”) should give the answer “1”, instead of “hello”.

Theorem 4 (Prompt tuning on LLM). *LLM f can solve the task T with prompt tuning, if and only if there exists a prompt P , such that $T_P \simeq f(P)$.*

Proof. If there exists a prompt P such that $T_P \simeq f(P)$, then we identify $f(P)$ with $h_{\mathcal{C}}(P)$. Hence, by Yoneda Lemma,

$$k_f(f(P), f(P + X)) \simeq k_f(T_P, f(P + X)) \simeq T_P(P + X) = T(X).$$

On the other hand, suppose there exists a prompt P such that

$$T(X) = T_P(P + X) = k_f(f(P), f(P + X))$$

for any input X . Then for every object of the form $P + X$, we have

$$T_P(P + X) \simeq k_f(f(P), f(P + X)) \simeq h_{\mathcal{C}}(P)(P + X).$$

If X' does not start with P , then by definition

$$T_P(X') = \emptyset,$$

and by the prefix structure of the language-model category,

$$h_C(P)(X') = \text{Hom}_C(P, X') = \emptyset.$$

Therefore, T_P and $h_C(P)$ agree on all objects. Since their functor structures are determined by the prefix structure of the language-model category, it follows that

$$T_P \simeq h_C(P).$$

this is equivalently written as

$$T_P \simeq f(P). \quad \square$$

The power of prompt tuning is limited, and can be characterized by representable functors. What about fine tuning? We have the following theorem:

Lemma 3 (Yoneda Extension of Functors). *Let $T \in \mathcal{C}^\vee$. Then there exists an extension functor $h_C^\dagger T : \mathcal{C}^\vee \rightarrow \mathbf{Set}$ such that $h_C^\dagger T \circ h_C \simeq T$. In fact, $h_C^\dagger T$ is given by $h_C^\dagger T(A) \triangleq \text{Hom}_{\mathcal{C}^\vee}(T, A)$, $A \in \mathcal{C}^\vee$.*

The actual Yoneda extension of functors theorem is more general (see e.g., Proposition 2.7.1 in Masaki Kashiwara (2006)), it can be applied to any category $\mathcal{A}$, but here we only use $\mathbf{Set}$ to avoid unnecessary technical details of inductive limits. Using Lemma 3, we immediately get the following theorem.

Theorem 5 (Power on fine tuning). *Ideally, given enough resources, h_C can solve any task $T \in \mathcal{C}^\vee$.*

Proof. Applying Lemma 3, we know that with enough training data for the downstream task, computational power, and a fine tuning model h that learns $h_C^\dagger T$ perfectly, we can concatenate h with f , to solve the task T . $\square$

The downstream tasks considered in Theorem 5 are based on the structure in $\mathcal{C}$, not the data content in the dataset. As a result, the category defined by Gidaris et al. (2018) still has very simple group-structure, but with Theorem 5 it is possible to solve more diverse tasks. For example, we can map all the objects to the same output, which cannot be achieved with prompt tuning. Theorem 5 conveys the importance of pretext task, as more informative pretext tasks will create more structured category $\mathcal{C}$, which further improves the power of fine tuning on $\mathcal{C}^\vee$.

Even with very informative $\mathcal{C}$, Theorem 5 only provides a raw upper bound with strong assumptions on the resources needed. By contrast, empirically the fine tuning step is usually done with a small network. Therefore, instead of treating it as a strong backup of what people are doing right now, it is more like exhibiting the future possibility. In other words, it implies that by transforming the objects to the corresponding feature space $\mathcal{C}^\vee$, we have captured all the information encoded in $\mathcal{C}$.

For readers who are familiar with machine learning theory, Theorem 5 may look similar to the theory of over-parameterization at first glance. However, they are analyzing different steps of self-supervised learning. Over-parameterization analyzes the pretraining step, saying that under certain assumptions, the optimization and generalization error will be very small for the pretext task, as long as the model is big enough and the learning rate is small enough. By contrast, Theorem 5 analyzes the fine tuning step after pretraining. Even if we have successfully pretrained a network on Imagenet with contrastive learning and zero generalization error, it remains unclear whether the model can be used for image segmentation as the downstream task, unless someone verifies it empirically. But Theorem 5 says, as long as the model is ideal, the feature layer contains all the information of $\mathcal{C}$ for any downstream tasks.

6 Multimodal Learning

In the previous section, we have seen how the foundation models are related to learning a category defined by a pretext task. What happens if we have multiple categories? We can first use the pretext tasks to learn different foundation models separately, then connect these models together in the embedding space. This is exactly how multimodal models like CLIP Radford et al. (2021), Dall-E 2 Ramesh et al. (2022), Multilingual CLIP Carlsson et al. (2022) and AltCLIP Chen et al. (2022) work.

In this section, we analyze the functors between different categories. Similar to the previous section, we consider the case that the functors are perfectly learned, and investigate the implications under this assumption. This setting hides unnecessary details like how the loss is defined or how network structure is designed. However, it provides interesting insights of how to connect different categories together, and what to expect after the functor connection.

As the starting point, we would like to emphasize that we assume the categories we consider are “natural”, which means they are not only self-consistent, but also consistent with each other. For example, when we use texts like “white, red, blue” or “big, small, tiny” to describe a chair in the language category, there are corresponding images in the image category. Such connection can be described by functors.

6.1 Generalization Theorem

We first assume that a functor between embedding spaces learns the object mapping between two categories perfectly, and then show that the structural information is preserved by this functor. Below we use the notations $h_{\mathcal{B}}, h_{\mathcal{C}}$ and $\mathcal{B}^{\vee}, \mathcal{C}^{\vee}$ to denote the ideal foundation models and the corresponding embedding spaces.

Definition 7 (feature-aligned functor). *Given two categories $\mathcal{B}, \mathcal{C}$, a full embedding $F : \mathcal{C} \rightarrow \mathcal{B}$, denote the corresponding foundation model as $h_{\mathcal{B}}, h_{\mathcal{C}}$. A functor $\hat{F} : \mathcal{C}^{\vee} \rightarrow \mathcal{B}^{\vee}$ is feature-aligned with F if for any $X \in \text{Ob}(\mathcal{C})$, $\hat{F}(h_{\mathcal{C}}(X)) \simeq h_{\mathcal{B}}(F(X))$.*

Theorem 6 (Generalization theorem for structural learning). *Consider two categories $\mathcal{B}$ and $\mathcal{C}$, and a full embedding*

$$F : \mathcal{C} \rightarrow \mathcal{B}.$$

In the learning scenario, an ideal foundation model $h_{\mathcal{C}}$ for $\mathcal{C}$ together with a feature-aligned functor

$$\hat{F} : \mathcal{C}^{\vee} \rightarrow \mathcal{B}^{\vee}$$

preserves the structure of $\mathcal{C}$ inside $\mathcal{B}$: for any $X, Y \in \mathcal{C}$,

$$\text{Hom}_{\mathcal{C}}(X, Y) \simeq \text{Hom}_{\mathcal{B}^{\vee}}(\hat{F}(h_{\mathcal{C}}(X)), \hat{F}(h_{\mathcal{C}}(Y))).$$

Moreover, for any $X \in \mathcal{C}$, the object $\hat{F}(h_{\mathcal{C}}(X)) \in \mathcal{B}^{\vee}$ lies in the representable image of $h_{\mathcal{B}}$. Hence there exists an object $Y_X \in \mathcal{B}$, unique up to isomorphism, such that

$$h_{\mathcal{B}}(Y_X) \simeq \hat{F}(h_{\mathcal{C}}(X)).$$

In fact, one may take $Y_X \simeq F(X)$.

Proof. Since $F : \mathcal{C} \rightarrow \mathcal{B}$ is a full embedding, for any $X, Y \in \mathcal{C}$ we have

$$\text{Hom}_{\mathcal{C}}(X, Y) \simeq \text{Hom}_{\mathcal{B}}(F(X), F(Y)).$$

By the full faithfulness of the covariant Yoneda embedding, we further obtain

$$\text{Hom}_{\mathcal{B}}(F(X), F(Y)) \simeq \text{Hom}_{\mathcal{B}^{\vee}}(h_{\mathcal{B}}(F(X)), h_{\mathcal{B}}(F(Y))).$$

Since $\hat{F}$ is feature-aligned, we have

$$h_{\mathcal{B}}(F(X)) \simeq \hat{F}(h_{\mathcal{C}}(X)), \quad h_{\mathcal{B}}(F(Y)) \simeq \hat{F}(h_{\mathcal{C}}(Y)).$$

Hence

$$\text{Hom}_{\mathcal{C}}(X, Y) \simeq \text{Hom}_{\mathcal{B}^{\vee}}(\hat{F}(h_{\mathcal{C}}(X)), \hat{F}(h_{\mathcal{C}}(Y))).$$

The representability statement follows immediately from feature alignment:

$$\hat{F}(h_{\mathcal{C}}(X)) \simeq h_{\mathcal{B}}(F(X)).$$

Therefore $\hat{F}(h_{\mathcal{C}}(X))$ lies in the representable image of $h_{\mathcal{B}}$. By the uniqueness of representatives up to isomorphism, there exists an object $Y_X \in \mathcal{B}$, unique up to isomorphism, such that

$$h_{\mathcal{B}}(Y_X) \simeq \hat{F}(h_{\mathcal{C}}(X)).$$

In particular, one may take $Y_X \simeq F(X)$. □

Theorem 6 is much more powerful than it appears. We call it the generalization theorem, because it provides another kind of generalization, different from the existing generalization theory on stability or Rademacher complexity. It tells us that, the structural information of one category can be recovered in the feature space by the structural information of another category with a feature-aligned functor.

6.2 Application of Theorem 6

In this subsection, we apply Theorem 6 to analyze two models: CLIP and Dall-E 2.

CLIP. The dataset of CLIP contains millions of image-text pairs. During pretraining, for each batch of N pairs of data points, CLIP uses an image encoder and a text encoder to obtain N pairs of embeddings, and learns a function for matching the N correct pairs out of the $N \times N$ possible connections. Theorem 6 then gives the following corollary.

Corollary 2 (Creativity of CLIP). *Let $\mathcal{C}$ be a category of language descriptions that admit corresponding images, viewed as a full subcategory of a language category $\mathcal{C}'$. Let $\mathcal{B}$ be the image category. Assume CLIP learns a feature-aligned functor*

$$\hat{F} : \mathcal{C}^\vee \rightarrow \mathcal{B}^\vee$$

for a full embedding

$$F : \mathcal{C} \rightarrow \mathcal{B}.$$

Then for any $X \in \mathcal{C}$, the object

$$\hat{F}(h_{\mathcal{C}}(X)) \in \mathcal{B}^\vee$$

lies in the representable image of $h_{\mathcal{B}}$. Hence there exists an image object $Y_X \in \mathcal{B}$, unique up to isomorphism, such that

$$h_{\mathcal{B}}(Y_X) \simeq \hat{F}(h_{\mathcal{C}}(X)).$$

In particular, one may take $Y_X \simeq F(X)$. Therefore, CLIP can represent new images that can be described in $\mathcal{C}$, even if they do not appear in the training set.

Remark. Corollary 2 explains why CLIP-style models can support images such as “avocado chair”, even when such images do not explicitly appear in the dataset. The full embedding assumption is crucial: otherwise, the feature computed in $\mathcal{C}^\vee$ cannot be transferred faithfully to the image side $\mathcal{B}^\vee$. Strictly speaking, Corollary 2 shows that the corresponding image object exists in the representable image of $h_{\mathcal{B}}$. To generate a concrete image, one still needs an image-side decoder or generator.

Dall-E 2. DALL-E 2 is the combination of CLIP and a diffusion model (Rombach et al., 2022; Sohl-Dickstein et al., 2015; Dhariwal and Nichol, 2021; Ho et al., 2020). If CLIP can be seen as a feature-aligned functor between suitable subcategories, why do we still need an extra diffusion model? The reason is that these two categories are not purely isomorphic at the object level. When we type “a photo of dog”, there are millions of different matching images. Therefore, DALL-E 2 effectively modifies the image side so that each object is no longer a single image, but a probability distribution of images. From this perspective, the diffusion model is a generator that samples concrete images from such an image-distribution object, while CLIP learns the functor from the category of texts to the category of image distributions.

6.3 Compositional Theorem

By applying Theorem 6 multiple times through a list of categories, we immediately get the following theorem.

Theorem 7 (Compositional Theorem). *Consider a list of categories $\{\mathcal{B}_i\}_{i=1}^n$, and $n - 1$ full embeddings $\{F_i\}_{i=1}^{n-1}$, where*

$$F_i : \mathcal{A}_i \rightarrow \mathcal{B}_{i+1},$$

with $\mathcal{A}_1 = \mathcal{B}_1$, and for $i > 1$, $\mathcal{A}_i$ is the full subcategory of $\mathcal{B}_i$ induced by F_{i-1} . Denote the foundation model for $\mathcal{B}_i$ by $h_{\mathcal{B}_i}$, and the feature-aligned functors by $\{\hat{F}_i\}_{i=1}^{n-1}$, where

$$\hat{F}_i : \mathcal{B}_i^\vee \rightarrow \mathcal{B}_{i+1}^\vee.$$

Define

$$F \triangleq F_{n-1}F_{n-2} \cdots F_1, \quad \hat{F} \triangleq \hat{F}_{n-1}\hat{F}_{n-2} \cdots \hat{F}_1.$$

Then, for any $X, Y \in \mathcal{B}_1$,

$$\text{Hom}_{\mathcal{B}_1}(X, Y) \simeq \text{Hom}_{\mathcal{B}_n}(\hat{F}(h_{\mathcal{B}_1}(X)), \hat{F}(h_{\mathcal{B}_1}(Y))).$$

Moreover, for any $X \in \mathcal{B}_1$, the object

$$\hat{F}(h_{\mathcal{B}_1}(X)) \in \mathcal{B}_n^\vee$$

lies in the representable image of $h_{\mathcal{B}_n}$. Hence there exists an object $Y_X \in \mathcal{B}_n$, unique up to isomorphism, such that

$$h_{\mathcal{B}_n}(Y_X) \simeq \hat{F}(h_{\mathcal{B}_1}(X)).$$

In particular, one may take $Y_X \simeq F(X)$.

Proof. Apply Theorem 6 recursively along the chain

$$\mathcal{B}_1 \xrightarrow{F_1} \mathcal{B}_2 \xrightarrow{F_2} \dots \xrightarrow{F_{n-1}} \mathcal{B}_n.$$

At each step, the corresponding feature-aligned functor preserves the morphism structure in the next embedding space. Composing these identifications yields

$$\mathrm{Hom}_{\mathcal{B}_1}(X, Y) \simeq \mathrm{Hom}_{\mathcal{B}_n}(\hat{F}(h_{\mathcal{B}_1}(X)), \hat{F}(h_{\mathcal{B}_1}(Y))).$$

The representability statement also follows recursively: at the final step, the composed feature

$$\hat{F}(h_{\mathcal{B}_1}(X))$$

is identified with

$$h_{\mathcal{B}_n}(F(X)),$$

and therefore lies in the representable image of $h_{\mathcal{B}_n}$. By uniqueness of representatives up to isomorphism, there exists an object $Y_X \in \mathcal{B}_n$, unique up to isomorphism, such that

$$h_{\mathcal{B}_n}(Y_X) \simeq \hat{F}(h_{\mathcal{B}_1}(X)).$$

In particular, one may take $Y_X \simeq F(X)$. □

If we want to map objects from $\mathcal{B}_1$ to $\mathcal{B}_n$, but the direct training data between the two categories is limited, Theorem 7 provides an alternative route. It suffices to find a path between the two categories and learn the functors for each edge of the path. This is especially useful when some of the categories are pretrained, such as GPT or CLIP. For example, Carlsson et al. (2022) and Chen et al. (2022) use multilingual text pairs to train the functors between various language categories and the English category, which are then connected to the image category using CLIP. In this way, they naturally extend CLIP to multilingual settings.

7 Discussion

7.1 Applying to small categories

In our abstract and introduction, we raised the question about an infinitely large model with infinite resources, which makes our theorem look unrealistic. However, these assumptions are purely rhetoric, and our real assumption is simply the model being ideal, i.e., the model perfectly solves the pretext task. This assumption is much weaker than the infinite one, because it means our theorems can be directly applied to small datasets, which correspond to smaller categories. For instance, consider a dataset containing only 100k sentences, and a model is trained to learn the data using a language model where each sentence is connected to its neighboring sentences with certain probabilities. While the sentences or words in the corresponding category $\mathcal{C}$ may not be as informative as the language category that humans possess, it is well-defined, and the ideal model is also well-defined. Training an ideal model for a small dataset of this scale is feasible. According to Theorem 3, such a model can only perfectly solve tasks that are representable by objects in $\mathcal{C}$.

7.2 Bayesian optimal classifier

The notion of Bayesian optimal classifier is widely used in the generalization theory for supervised learning. It simply means that due to the inherent noise in the labels, even the best model will not be able to get zero loss in the population distribution. For example, given a blurry image of a dog, it might be difficult to tell its exact breed.

However, in our categorical framework, we take the physicist’s view of the world, and assume that **all the objects and relationships are self consistent**, without any noise. For example, Alaskan Husky and Siberian Husky indeed look similar from their photos, but they are fundamentally different breeds, in terms of their origins, size, appearance, temperament and so on. Our framework models these conceptual differences, and treats the photos of the dogs (which have noise) as the outcome of the last step of diffusion as discussed in Section 6. If the reader is familiar with the functional programming, the categorical framework can be seen as the “pure functional” part, which is predictable and precise. By contrast, the process that deals with the actual input and output, can be seen as the “side effects”, where the notion of Bayesian optimal classifier comes in.

7.3 Relationship to RKHS

Our categorical framework can be seen as a natural generalization of the RKHS framework, where in RKHS a kernel function outputs a value in $\mathbb{R}$, while our framework generalizes this to **Set**.

For example, by analogy with the Yoneda perspective to RKHS, we obtain the reproducing property

$$\forall x \in \mathcal{C}, \forall f \in \mathcal{H}, \langle k(x, \cdot), f \rangle_{\mathcal{H}} = f(x),$$

where f can be viewed as a functor in $\mathcal{C}^{\vee}$, $k(x, \cdot)$ is a representable functor, and $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ computes $\text{Hom}_{\mathcal{C}}$. Therefore, in RKHS the morphisms between two objects are represented by a single real number computed by $\langle \cdot, \cdot \rangle_{\mathcal{H}}$. This case works perfectly for algorithms like SimCLR, where the relationship between two objects is a real number representing similarity. However, in practice, the relationship between two objects can be much more than a real number, especially for NLP tasks. In that case, RKHS is insufficient, and the general Yoneda lemma is necessary.

Acknowledgements

This paper was greatly inspired by the fruitful interactions with the Qianfang (functor) team at the Beijing Academy of Artificial Intelligence. We would like to extend special thanks to Yue Cao for insightful discussions and the anonymous reviewers for their constructive feedback and suggestions. Our sincere appreciation also goes to Dun Liang and Yiyang Jia for their help with clarifications of the underlying category of SimCLR, and to Haozhe Jiang for suggesting a better narrative of the paper, as well as identifying a bug in the application of Theorem 3 to large language models. We also acknowledge the assistance of large language models in improving the exposition and helping identify several issues in earlier drafts of the manuscript. This paper is supported by the Ministry of Science and Technology of the People’s Republic of China, the 2030 Innovation Megaprojects “Program on New Generation Artificial Intelligence” (Grant No. 2021AAA0150000).

References

- Adámek, J., Herrlich, H., and Strecker, G. (1990). *Abstract and concrete categories*. Wiley-Interscience.
- Allen-Zhu, Z., Li, Y., and Liang, Y. (2019a). Learning and generalization in overparameterized neural networks, going beyond two layers. *Advances in neural information processing systems*, 32.
- Allen-Zhu, Z., Li, Y., and Song, Z. (2019b). A convergence theory for deep learning via over-parameterization. In *International Conference on Machine Learning*, pages 242–252. PMLR.
- Arandjelovic, R. and Zisserman, A. (2018). Objects that sound. In *Proceedings of the European conference on computer vision (ECCV)*, pages 435–451.
- Arora, S., Du, S., Hu, W., Li, Z., and Wang, R. (2019a). Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *International Conference on Machine Learning*, pages 322–332. PMLR.
- Arora, S., Khandeparkar, H., Khodak, M., Plevrakis, O., and Saunshi, N. (2019b). A theoretical analysis of contrastive unsupervised representation learning. *arXiv preprint arXiv:1902.09229*.
- Bartlett, P. L., Foster, D. J., and Telgarsky, M. J. (2017). Spectrally-normalized margin bounds for neural networks. *Advances in neural information processing systems*, 30.
- Bartlett, P. L., Long, P. M., Lugosi, G., and Tsigler, A. (2020). Benign overfitting in linear regression. *Proceedings of the National Academy of Sciences*, 117(48):30063–30070.
- Bradley, T.-D., Terilla, J., and Vlassopoulos, Y. (2022). An enriched category theory of language: from syntax to semantics. *La Matematica*, pages 1–30.
- Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.
- Cai, Q., Yang, Z., Jin, C., and Wang, Z. (2020). Provably efficient exploration in policy optimization. In *International Conference on Machine Learning*, pages 1283–1294. PMLR.

- Carlsson, F., Eisen, P., Rekathati, F., and Sahlgren, M. (2022). Cross-lingual and multilingual clip. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 6848–6854.
- Censi, A. (2015). A mathematical theory of co-design. *arXiv preprint arXiv:1512.08055*.
- Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. (2020). A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.
- Chen, X. and He, K. (2021). Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15750–15758.
- Chen, Z., Liu, G., Zhang, B.-W., Ye, F., Yang, Q., and Wu, L. (2022). Altclip: Altering the language encoder in clip for extended language capabilities. *arXiv preprint arXiv:2211.06679*.
- Clark, K., Luong, M.-T., Le, Q. V., and Manning, C. D. (2020). Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.
- Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314.
- Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
- Dhariwal, P. and Nichol, A. (2021). Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794.
- Doersch, C., Gupta, A., and Efros, A. A. (2015). Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE international conference on computer vision*, pages 1422–1430.
- Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.
- Du, S., Krishnamurthy, A., Jiang, N., Agarwal, A., Dudik, M., and Langford, J. (2019a). Provably efficient rl with rich observations via latent state decoding. In *International Conference on Machine Learning*, pages 1665–1674. PMLR.
- Du, S., Lee, J., Li, H., Wang, L., and Zhai, X. (2019b). Gradient descent finds global minima of deep neural networks. In *International conference on machine learning*, pages 1675–1685. PMLR.
- Du, S. S., Kakade, S. M., Wang, R., and Yang, L. F. (2019c). Is a good representation sufficient for sample efficient reinforcement learning? *arXiv preprint arXiv:1910.03016*.
- Du, S. S., Zhai, X., Poczos, B., and Singh, A. (2018). Gradient descent provably optimizes over-parameterized neural networks. *arXiv preprint arXiv:1810.02054*.
- Fong, B. and Spivak, D. I. (2018). Seven sketches in compositionality: An invitation to applied category theory. *arXiv preprint arXiv:1803.05316*.
- Gao, T., Fisch, A., and Chen, D. (2020). Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*.
- Gidaris, S., Singh, P., and Komodakis, N. (2018). Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*.
- Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., Doersch, C., Avila Pires, B., Guo, Z., Gheshlaghi Azar, M., et al. (2020). Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284.
- HaoChen, J. Z., Wei, C., Gaidon, A., and Ma, T. (2021). Provable guarantees for self-supervised deep learning with spectral contrastive loss. *Advances in Neural Information Processing Systems*, 34:5000–5011.
- He, K., Chen, X., Xie, S., Li, Y., Dollár, P., and Girshick, R. (2022). Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009.
- He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. (2020). Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738.

- He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.
- Ho, J., Jain, A., and Abbeel, P. (2020). Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851.
- Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366.
- Jiang, Z., Xu, F. F., Araki, J., and Neubig, G. (2020). How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.
- Jin, C., Allen-Zhu, Z., Bubeck, S., and Jordan, M. I. (2018). Is q-learning provably efficient? *Advances in neural information processing systems*, 31.
- Kuś, M. and Skowron, B. (2019). *Category Theory in Physics, Mathematics, and Philosophy*. Springer.
- Lee, J. D., Lei, Q., Saunshi, N., and Zhuo, J. (2021). Predicting what you already know helps: Provable self-supervised learning. *Advances in Neural Information Processing Systems*, 34:309–323.
- Li, X. L. and Liang, P. (2021). Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.
- Li, Y. and Yuan, Y. (2017). Convergence analysis of two-layer neural networks with relu activation. *Advances in neural information processing systems*, 30.
- Liu, Q. (2022). Rectified flow: A marginal preserving approach to optimal transport. *arXiv preprint arXiv:2209.14577*.
- Liu, X., Zheng, Y., Du, Z., Ding, M., Qian, Y., Yang, Z., and Tang, J. (2021). Gpt understands, too. *arXiv preprint arXiv:2103.10385*.
- Luo, Z., Weng, C., Wu, S., Zhou, M., and Ge, R. (2022). One objective for all models—self-supervised learning for topic models. *arXiv preprint arXiv:2203.03539*.
- Mac Lane, S. (2013). *Categories for the working mathematician*, volume 5. Springer Science & Business Media.
- Mahadevan, S. (2022a). Categoroids: Universal conditional independence. *arXiv preprint arXiv:2208.11077*.
- Mahadevan, S. (2022b). Unifying causal inference and reinforcement learning using higher-order category theory. *arXiv preprint arXiv:2209.06262*.
- Marquis, J.-P. (2008). *From a geometrical point of view: A study of the history and philosophy of category theory*, volume 14. Springer Science & Business Media.
- Masaki Kashiwara, P. S. (2006). *Categories and Sheaves*. Springer.
- Noroozi, M. and Favaro, P. (2016). Unsupervised learning of visual representations by solving jigsaw puzzles. In *European conference on computer vision*, pages 69–84. Springer.
- Oord, A. v. d., Li, Y., and Vinyals, O. (2018). Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.
- Pathak, D., Girshick, R., Dollár, P., Darrell, T., and Hariharan, B. (2017). Learning features by watching objects move. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2701–2710.
- Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T., and Efros, A. A. (2016). Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2536–2544.
- Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. (2021). Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR.
- Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al. (2018). Improving language understanding by generative pre-training. *OpenAI blog*.

- Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.
- Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P. J., et al. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67.
- Raghu, M., Poole, B., Kleinberg, J., Ganguli, S., and Sohl-Dickstein, J. (2017). On the expressive power of deep neural networks. In *international conference on machine learning*, pages 2847–2854. PMLR.
- Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. (2022). Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*.
- Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., and Sutskever, I. (2021). Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pages 8821–8831. PMLR.
- Riehl, E. (2017). *Category theory in context*. Courier Dover Publications.
- Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. (2022). High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695.
- Shalev-Shwartz, S. and Ben-David, S. (2014). *Understanding machine learning: From theory to algorithms*. Cambridge university press.
- Shiebler, D., Gavranović, B., and Wilson, P. (2021). Category theory in machine learning. *arXiv preprint arXiv:2106.07032*.
- Shin, T., Razeghi, Y., Logan IV, R. L., Wallace, E., and Singh, S. (2020). Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*.
- Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. (2015). Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR.
- Tan, Z., Zhang, Y., Yang, J., and Yuan, Y. (2023). Contrastive learning is spectral clustering on similarity graph.
- Tosh, C., Krishnamurthy, A., and Hsu, D. (2021). Contrastive learning, multi-view redundancy, and linear models. In *Algorithmic Learning Theory*, pages 1179–1206. PMLR.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.
- Wen, Z. and Li, Y. (2021). Toward understanding the feature learning process of self-supervised contrastive learning. In *International Conference on Machine Learning*, pages 11112–11122. PMLR.
- Wen, Z. and Li, Y. (2022). The mechanism of prediction head in non-contrastive self-supervised learning. *arXiv preprint arXiv:2205.06226*.
- Yin, D., Kannan, R., and Bartlett, P. (2019). Rademacher complexity for adversarially robust generalization. In *International conference on machine learning*, pages 7085–7094. PMLR.
- Zbontar, J., Jing, L., Misra, I., LeCun, Y., and Deny, S. (2021). Barlow twins: Self-supervised learning via redundancy reduction. In Meila, M. and Zhang, T., editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 12310–12320. PMLR.
- Zimmermann, R. S., Sharma, Y., Schneider, S., Bethge, M., and Brendel, W. (2021). Contrastive learning inverts the data generating process. In *International Conference on Machine Learning*, pages 12979–12990. PMLR.
- Zou, D., Cao, Y., Zhou, D., and Gu, Q. (2020). Gradient descent optimizes over-parameterized deep relu networks. *Machine learning*, 109(3):467–492.