Know2Vec: A Black-Box Proxy for Neural Network Retrieval

Zhuoyi Shang[1,2,3], **Yanwei Liu**[1,3*], **Jinxia Liu**[4], **Xiaoyan Gu**[1,3], **Ying Ding**[1 3], **Xiangyang Ji**[5]

[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3]Key Laboratory of Cyberspace Security Defense, Beijing, China
[4]College of Information and Intelligence Engineering, Zhejiang Wanli University, Ningbo, China
[5]Tsinghua University, Beijing, China
{shangzhuoyi,liuyanwei}@iie.ac.cn, liujinxia@zjwu.edu.cn, {guxiaoyan,dingying}@iie.ac.cn, xyji@tsinghua.edu.cn

Abstract

For general users, training a neural network from scratch is usually challenging and labor-intensive. Fortunately, neural network zoos enable them to find a well-performing model for directly use or fine-tuning it in their local environments. Although current model retrieval solutions attempt to convert neural network models into vectors to avoid complex multiple inference processes required for model selection, it is still difficult to choose a suitable model due to inaccurate vectorization and biased correlation alignment between the query dataset and models. From the perspective of knowledge consistency, i.e., whether the knowledge possessed by the model can meet the needs of query tasks, we propose a model retrieval scheme, named Know2Vec, that acts as a black-box retrieval proxy for model zoo. Know2Vec first accesses to models via a black-box interface in advance, capturing vital decision knowledge from models while ensuring their privacy. Next, it employs an effective encoding technique to transform the knowledge into precise model vectors. Secondly, it maps the user's query task to a knowledge vector by probing the semantic relationships within query samples. Furthermore, the proxy ensures the knowledge-consistency between query vector and model vectors within their alignment space, which is optimized through the supervised learning with diverse loss functions, and finally it can identify the most suitable model for a given task during the inference stage. Extensive experiments show that our Know2Vec achieves superior retrieval accuracy against the state-of-the-art methods in diverse neural network retrieval tasks.

Extended version — https://arxiv.org/pdf/2412.16251

Introduction

Well-trained models in many domains have demonstrated promising performance in various downstream tasks. The training process refines knowledge from dataset into general rules and patterns, enabling the model to make accurate predictions on new data (Tian et al. 2023). However, their performances vary widely for a targeted downstream application (Zhang et al. 2023). The model whose knowledge is more closely aligned with the task requirements tends to perform better. For example, a model with numerical knowledge would find it easier to complete the MNIST(Deng 2012) classification task than a model specialized in flower classification. Assessing the suitableness of a Deep learning(DL) model by uploading the entire dataset to the huge model market for comparison against inference results is risky and impractical due to data disclosure and resource constraints. Therefore, further research is needed to evaluate the correlation between neural network models and query tasks.

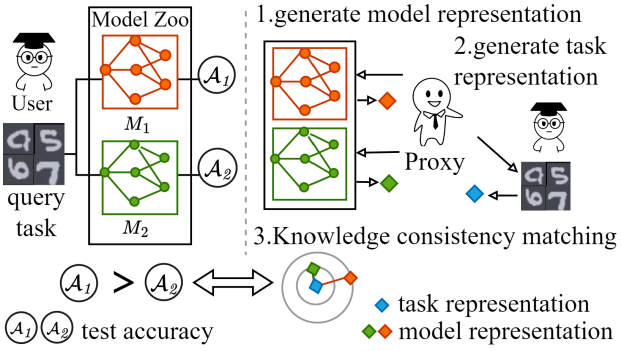

Figure 1: Knowledge-consistency-based black-box proxy for model retrieval.

Source-Free model transferability estimation (SF-MTE) (Bao et al. 2019; Nguyen et al. 2020; Zhang et al. 2023) methods are designed to rank the suitability of pre-trained models for fine-tuning in downstream tasks. Traditional methods (Bao et al. 2019; Nguyen et al. 2020) directly score the candidate models by utilizing statistical data like features or joint distribution of models and query task. Typically, Model Spider(Zhang et al. 2023) vectorizes both neural network models and query tasks to avoid the high computational costs of forward propagation increased by traditional methods.

With the vectorization idea, Neural Network Retrieval (NNR) (Jeong et al. 2021; Zhong, Qian, and Zhang 2021) tries to transform models and datasets into specific embeddings that facilitates their matching. Generating vectors for models and datasets, and calculating their correlations require an accurate understanding of the key knowledge of both models and query tasks. The pioneering NNR study, DNNR (Zhong, Qian, and Zhang 2021) utilizes litmus im-

*Corresponding authors.

ages to construct models' semantic vectors, while TANS (Jeong et al. 2021) further advances the field by searching for a cross-modal space to minimize the semantic discrepancy between model representations and query images. These techniques, while improving retrieval efficiency, still encounter various problems, such as laborious and rough vector generation process (Zhong, Qian, and Zhang 2021; Zhang et al. 2023), imprecise alignment (Jeong et al. 2021), and the necessity for privacy protection (Zhang et al. 2023; Jeong et al. 2021).

In particular, the primary challenge of the correlation calculation methods(SF-MTE or NNR) lies in the two aspects as follows. (1) **Transforming the unstructured nature of neural network models into a vectorial format**, which must capture the intrinsic knowledge in models for effective retrieval. (2) **The establishment of a quantifiable mapping space**, where query vectors align with model representations, ensuring semantically similar vectors are proximate. Existing methods often rely on complex and suboptimal vector generation processes, failing to fully capture critical model knowledge or achieve seamless alignment.

To address the above challenges, we propose a novel knowledge-consistency-based black-box proxy for model retrieval, named Know2Vec, and it is shown in Fig. 1. The objective of Know2Vec is to establish a consistent representation of knowledge, allowing semantic alignment between the query task and models. Firstly, it abstracts the intrinsic knowledge acquired by neural network models into a generalized representation in a black-box way. Next, it interacts with users to generate effective task representation by understanding the differences between query samples. Lastly, the proxy is designed to perform a knowledge consistency matching between the abstracted model representation and the task representation, facilitating efficient model retrieval.

Our key contributions are:

- We propose a model knowledge vectorization scheme for parameter-agnostic scenarios, which is designed to capture the implicit model knowledge and further vectorize it to support accurate model retrieval. We further prove in theory that it is feasible to obtain model information with randomly selected probes.
- A carefully designed measure function is proposed to align the heterogeneous knowledge embeddings, which correspond to the knowledge of the query task and those of known models, assisting users in accurately defining their needs and retrieving the most suitable neural network model.
- Know2Vec achieves superior retrieval performance across various NNR tasks, outperforming state-of-the-art baselines in our experiments. Additionly, it accesses neural network models in a black-box manner, eliminating the need to understand internal parameters, thus preserving privacy.

Related Work

Neural Network Retrieval

NNR addresses the model selection issue by mapping query entries and neural network models into vectors, enabling users to find a satisfactory pre-trained model from model markets (Zhou 2016). Deep Neural Network Retrieval (DNNR)(Zhong, Qian, and Zhang 2021) initially achieves model vectorization through feeding random litmus images to the candidate models. However, it needs extensive datasets and computational resources, making it impractical for online retrieval. TANS (Jeong et al. 2021) aims to align query datasets with similar neural network representations, but it overlooks the subtle differences within categories that are key for aligning knowledge. By representing key decision-making knowledge from models without accessing to their internal parameters and aligning it with the need of a query task, our method achieves both privacy protection and precise retrieval goals.

Source-Free Model Transferability Estimation

For a given target task and a model library, Source-Free Model Transferability Estimation(SF-MTE) (Ding et al. 2024) aims to propose a metric to quantify the transferability score without the need for individual training. Static SF-MTE methods, such as LEEP(Nguyen et al. 2020), H-score(Bao et al. 2019), compute scores directly from statistical data like features and logits. In contrast, Dynamic SF-MTE methods aim to project static features into tailored spaces to facilitate superior approximation. They try to estimate the maximum average log evidence(You et al. 2021), or they endeavor to identify a model/task vectorlization technique such as the classical Model Spider (Zhang et al. 2023). However, despite enhancing computational efficiency to a certain extent, these methods still necessitate a complex training process.

Boundary Supporting Samples

Boundary supporting samples are identified as those close to the decision boundary of neural network models. Assume the target model is a k-class DNN classifier, where the output layer is an active layer. Formally, we denote by $\{g_i\}$ the decision functions of the target classifier, and a data point x is on the target classifier's classification boundary if at least two labels have the largest discrimination probability, i.e., $g_a(x) = g_b(x) \geq \max_{c \neq a,b} g_c(x)$, where a, b, c are category index, and $g_a(x)$ is the probability that sample x belongs to category A (Cao, Jia, and Gong 2021). Tian et al. (Tian et al. 2023) claimed that the knowledge transferred from a training dataset to a DL model can be uniquely represented by the model's decision boundary samples, providing feasibility for us to acquire model knowledge in a black-box setting. However, this method only acquires partial model knowledge and requires target training dataset, which is illogical in NNR problem. Accordingly, we propose a parameter-agnostic model knowledge vectoring approach without demanding training dataset.

Method

Taking NNR problem as an example, we will elaborate on calculating model-dataset correlation when the model parameters are agnostic. We start by vectorizing models and query tasks, which helps to distill the models' knowledge

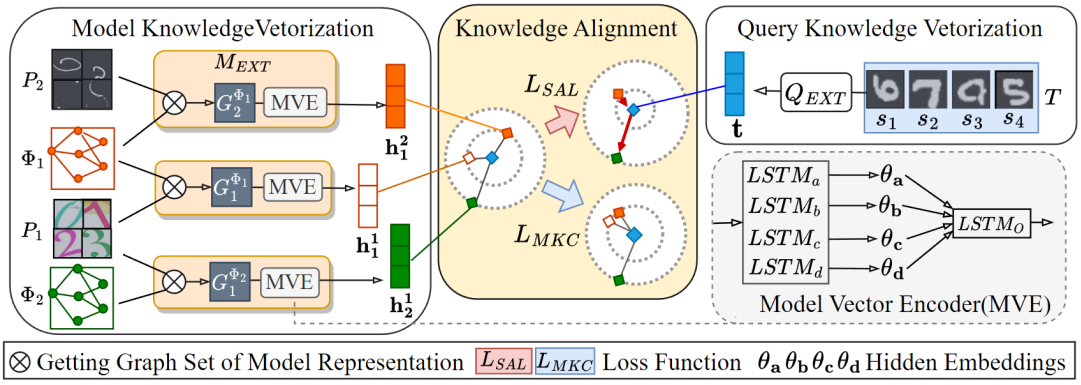

Figure 2: Model Retrieval Framework.

and clarify the requirements of the tasks. We tackle the new issues that incurred from the limited known information. Next, we seek a knowledge-consistent space that acts as a bridge, which connects the two modalities despite their differences in structures and semantic parameters, and providing a way to measure their semantic similarity.

Problem Formulation

We consider an arbitrary query task $T = \{s_i, l_i\}_{i=1}^{n}$ with n samples $\{s_i\}$ and the corresponding target labels $\{l_i\}$. Given a large model hub $M = \{\Phi_i\}_{i=1}^{m}$ with a total of m well-trained models, the goal of NNR is to choose a DNN model Φ_j that performs well on T. We define $\mathcal{A}(\Phi_i, T)$ the verification accuracy of T on model Φ_i. Mathematically, NNR aims to search for the best-fitted model Φ_j that satisfies

$$j = \arg\max_i \mathcal{A}(\Phi_i, T) \tag{1}$$

As mentioned earlier, we assume there is a virtually perfect proxy $\mathcal{P}$ that serves as a good communication intermediary: (1) It distills model knowledge and obtains vector $\mathbf{h_i}$ for each candidate model Φ_i; (2) It gets the requirements of query tasks T and generates the corresponding query knowledge vector $\mathbf{t}$; (3) It selects a suitable model Φ_j through a semantic measurement $\mathcal{DIS}()$. For an effective NNR method, maximizing $\mathcal{A}(\Phi_i, T)$ is equivalent to minimizing $\mathcal{DIS}(\mathbf{h_i}, \mathbf{t})$

$$j = \arg\max_i \mathcal{A}(\Phi_i, T) \Leftrightarrow j = \arg\min_i \mathcal{DIS}(\mathbf{h_i}, \mathbf{t}) \tag{2}$$

Fig. 2 illustrates the well-designed model retrieval framework. $\mathcal{P}$ includes three components: a model knowledge extractor M_{EXT}, a query knowledge extractor Q_{EXT}, and a knowledge alignment space with measurement function $\mathcal{DIS}()$. Firstly, M_{EXT} vectorizes model knowledge assisted by a series of additional probe datasets, denoted as $\mathbf{h}_\mathbf{i}^{\mathbf{i'}} = M_{EXT}(P_{i'}, \Phi_i)$, where $P_{i'}$ is the i'th probe dataset. Specifically for model Φ_i, M_{EXT} starts by creating a graph set $G_{i'}^{\Phi_i}$ with $P_{i'}$, and then encodes $G_{i'}^{\Phi_i}$ into $\mathbf{h}_\mathbf{i}^{\mathbf{i'}}$ through a model vector encoder. Next, Q_{EXT} extracts semantic correlations from query task T, producing a task knowledge vector $\mathbf{t} = Q_{EXT}(T)$. After that, the knowledge alignment space assesses the consistency of knowledge between model vectors $\{\mathbf{h}_i\}$ and query vector $\mathbf{t}$, for selecting the suitable model to the task.

Model Knowledge Vectorization

The previous NNR methods attempted to break down candidate model and explore the semantic information through exposed parameters. However, this is laborious and privacy-unfriendly. Fortunately, Theorem 1 proves that model knowledge that is transferred from the training dataset can be encapsulated by a matrix, denoted as knowledge representation matrix (KRM), providing the possibility for more efficient and privacy-preserving model knowledge embedding. Given the neural network Φ and its centroid samples of training dataset P, KRM can be generated based on the model's response to input samples in advance, which only requiring black-box access to Φ.

KRM is formed by two kind of representative samples: centroid samples and decision boundary samples. As illustrated in Fig. 3, taking binary classification that contains categories A and B as an example, the transferred knowledge KRM consists of two vectors $\{\mathbf{r_a^b} = x_a^b - x_a, \mathbf{r_b^a} = x_b^a - x_b\}$, where x_a and x_b are centroid samples of A and B, respectively, x_a^b is the decision boundary sample from A to B, and x_b^a is the decision boundary sample from B to A. x_a^b can be generated by points that respectively belong to categories A and B(such as x_a, x_b), and similarly for x_b^a.

Yet even with KRM, obtaining an effective representation $\mathbf{h}$ is still challenging. The primary obstacle lies in the design of model knowledge extractor M_{EXT}, which is responsible for converting information-limited KRM into measurable vectors to enable model retrieval tasks. These vectors in KRM encapsulates the incomplete decision knowledge of the model. Considering the dynamic changes in features, we further collect the information both in KRM and representative samples as a graph set and design a specialized DL framework to generate $\mathbf{h}$.

Furthermore, obtaining central samples is almost impossible because model owners tend to withhold their training datasets due to privacy concerns or copyright restrictions, which complicating the KRM generation process. We solve this issue by proving that alternative datasets can effectively generate a model's knowledge representation vector.

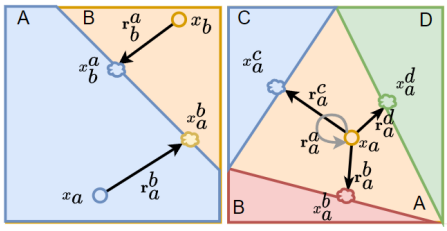

Figure 3: Decision Boundary Sample.

Theorem 1 *(Tian et al. 2023) The knowledge transferred from a training dataset to a deep learning model can be represented by the knowledge representation matrix KRM formed by perturbation vectors across different classes. For a k-class classifier, let the centroid sample of category A be denoted as x_a, the perturbation vector $\mathbf{r_a^k} = x_a^k - x_a$ from category A to category K is defined as the offsets between x_a and x_a^k, where x_a^k is the boundary sample from category A to category K. Collectively, KRM is defined by:*

$$KRM = \begin{bmatrix} \mathbf{0} & \mathbf{r_a^b} & ... & \mathbf{r_a^k} \\ \mathbf{r_b^a} & \mathbf{0} & ... & \mathbf{r_b^k} \\ .. & ... & .. & \mathbf{r_c^k} \\ \mathbf{r_k^a} & \mathbf{r_k^b} & ... & \mathbf{0} \end{bmatrix} \tag{3}$$

Getting Graph Set of Model Representation Relying solely on KRM to obtain model knowledge may lead to information loss since the knowledge transfer vector $\mathbf{r_b^a} = x_b^a - x_b$ overlooks crucial details such as the starting point x_b and ending point x_b^a. The central sample holds key feature about its category, while the boundary samples imply transition features between categories. First, it's difficult to pinpoint exactly how features changed as the sample moves from x_b to x_b^a within $\mathbf{r_b^a}$. Second, KRM offers limited insight of the distinctive intra-class knowledge.

As illustrated in Fig. 3, for a classification model Φ_i with 4 categories, the centroid sample x_a of category A is interconnected with boundary samples $\overline{x_a} = \{x_a^b, x_a^c, x_a^d\}$, forming a directed graph structure. Within this structure, the directed edges $\{\mathbf{r_a^b}, \mathbf{r_a^c}, \mathbf{r_a^d}\}$ represent the specific connections of x_a. This graph is formally defined as $G_a = \{x_a, \overline{x_a}, \{\mathbf{r_a^b}, \mathbf{r_a^c}, \mathbf{r_a^d}\}\}$. Among them, $\overline{x_a}$ and x_a are two different types of nodes. Expand to other categories, a total of 4 sets of such connection relationships can be modeled: $G^{\Phi_i} = \{G_a, G_b, G_c, G_d\}$. Undoubtedly, G^{Φ_i} offers a richer semantic representation than KRM.

G^{Φ_i} implicitly links G_a, G_b, G_c, G_d through relationships between categories. Specifically, the information of category A can be obtained from these three types of nodes: (1) The central sample x_a which embodies the unique features about A; (2) The boundary samples $\overline{x_a}$ from A to other categories, and they explain which features need to change for the transition from A to other categories; (3) Boundary samples x_b^a, x_c^a, x_d^a from other categories to A that suggest why the model might incorrectly classify as A. These points are present in G_b, G_c, G_d. By encoding all nodes in G^{Φ_i} through inter-category relationships, we facilitate a transformation from G^{Φ_i} into the model knowledge vector $\mathbf{h}$.

Implementation of Model Vector Encoder We consider the relationships as the dependencies of sequential data, in which each sequence corresponds to one category. As shown in Fig. 2, the model vector encoder is implemented with an inner-outer encoder. The inner encoder processes individual subgraphs, while the outer encoder integrates these subgraphs. Both are completed by a bidirectional Long Short Term Memory(LSTM)(Yu et al. 2019) network to handle variable long term dependencies. For category A, the information from the above-mentioned first two types of nodes (1) and (2) has been successfully encoded to the hidden embeddings θ_a by inner layer $LSTM_a$, as mentioned in Eq. 4. Additionally, the embeddings $\theta_b, \theta_c, \theta_d$ already include information from the third type of nodes (3). These embeddings are further aggregated as the model knowledge vector $\mathbf{h}$ by the outer-layer $LSTM_O$, aligned through sequence correspondence.

$$\begin{aligned} \theta_{\mathbf{a}} &= LSTM_a(x_a, x_a^b, x_a^c, x_a^d; W, b) \\ \theta_{\mathbf{b}} &= LSTM_b(x_b^a, x_b, x_b^c, x_b^d; W, b) \\ \theta_{\mathbf{c}} &= LSTM_c(x_c^a, x_c^b, x_c, x_c^d; W, b) \\ \theta_{\mathbf{d}} &= LSTM_d(x_d^a, x_d^b, x_d^c, x_d; W, b) \\ \mathbf{h_i} &= LSTM_O(\theta_{\mathbf{a}}, \theta_{\mathbf{b}}, \theta_{\mathbf{c}}, \theta_{\mathbf{d}}; W, b) \end{aligned} \tag{4}$$

where W, b are the optimizable parameters.

Thus, M_{EXT} has encoded G^{Φ} into vector $\mathbf{h}$, as the bidirectional network ensures all edges in G^{Φ} are reachable, either directly or indirectly.

Using probe datasets instead of training datasets Access to the training dataset of a neural network is sometimes impractical, but we can still use other data to probe and obtain boundary samples. Lemma 1 theoretically proves the feasibility that we can still get the semantic relationships of G^{Φ_i} with probe samples.

Lemma 1 *The perturbation vectors in KRM can also be obtained from the target model with associating the external datasets.*

Proof 1 *Taking binary classification with categories A and B as an example, we consider a neural network model Φ as*

$$\Phi(x) = \delta(\mathbf{w} * x + \mathbf{b}) \tag{5}$$

where δ is the active function, $\mathbf{w}$ and $\mathbf{b}$ are the weights and biases, respectively, x is any input sample, and $*$ denotes multiplication between vectors.

δ is composed of g_A and g_B, where $g_A(x)$ is the probability that sample x belongs to category A, and similarly $g_B(x)$ for category B. For the convenience of narration, it may be helpful to set $\delta = g_A - g_B$. Assuming that the centroid samples of training dataset for Φ are x_a and x_b, the boundary sample from A to B is x_a^b , then

$$\delta(\mathbf{w} * x_a^b + \mathbf{b}) = 0 \tag{6}$$

$$\delta(\mathbf{w} * x_a + \mathbf{b}) = 1 \tag{7}$$

$$\delta(\mathbf{w} * x_b + \mathbf{b}) = -1 \tag{8}$$

There must exist two selected samples z_a and z_b that satisfy $\delta(\mathbf{w} * z_a + b) = 1 - \lambda_1, \delta(\mathbf{w} * z_b + b) = -1 + \lambda_2$, where λ_1, λ_2

are very small values that can be ignored. Correspondingly, a boundary sample z_a^b from A to B satisfies $\delta(\mathbf{w} * z_a^b + b) = 0 - \lambda_3$, with λ_3 being a very small value. Then,

$$\delta(\mathbf{w} * x_a^b + \mathbf{b}) - \delta(\mathbf{w} * z_a^b + \mathbf{b}) = \lambda_3 \quad (9)$$

$$\delta(\mathbf{w} * x_a + \mathbf{b}) - \delta(\mathbf{w} * z_a + \mathbf{b}) = \lambda_1 \quad (10)$$

$$\delta(\mathbf{w} * x_b + \mathbf{b}) - \delta(\mathbf{w} * z_b + \mathbf{b}) = \lambda_2 \quad (11)$$

Since δ is continuous and differential in the regions of interest, there exists a value σ that satisfies $z_a^b = x_a^b + \sigma$ due to the Mean Value Theorem. Similarly, there must also be a disturbance σ_a such that $x_a = z_a + \sigma_a$.

Therefore, the perturbation vector $\mathbf{r_a^b} = x_a^b - x_a$ can also alternatively represented by Eq.(12), where σ and σ_a are the offsets.

$$\mathbf{r_a^b} = z_a^b - z_a + \sigma + \sigma_a \quad (12)$$

For a fixed model Φ, x_a and x_a^b are unique, and also σ and σ_a are only related to z_a and z_a^b, respectively. Therefore, $\mathbf{r_a^b}$ can be represented by z_a and z_a^b. This principle is applicable to other vectors in KRM, and the proven conclusion can be extended to other classification models.

Query Knowledge Vectorization

For the query task $T = \{s_i, l_i\}_{i=1}^n$, we implement the query encoder Q_{EXT} to discern correlations both within and across categories within the query samples. We first average the samples of each class to identify features unique to that class, denoted as $\theta_\mathbf{k}$ for class k, then we feed these features of distinct classes as separate sequences into a bidirectional LSTM-based network $LSTM_t$ to investigate how they relate to one another, as detailed in the following equation,

$$\begin{aligned} \theta_\mathbf{k} &= \frac{1}{|\mathbf{I}(l_i = k)|} \sum [s_i * \mathbf{I}(l_i = k)] \\ \mathbf{t} &= LSTM_t(\theta_\mathbf{1}, \theta_\mathbf{2}, ..., \theta_\mathbf{k}; W, b) \end{aligned} \quad (13)$$

where k is the category index, $|\mathbf{I}(l_i = k)|$ is the number of class k, $\mathbf{I}()$ is one if the logical expression in the bracket is true, otherwise is zero.

Knowledge Alignment

We develop an effective loss function that encourages the alignment between model embedding $\mathbf{h}$ and task embedding $\mathbf{t}$ within our retrieval proxy, enabling knowledge-consistent model retrieval. As shown in Fig. 2, a model embedding consistency function L_{MKC} is used to encourage neural networks to overcome the noise caused by external datasets, and a spatial alignment loss function L_{SAL} is used to overcome various biases between $\mathbf{h}$ and $\mathbf{t}$.

Model Embedding Consistency Loss. Assuming $\mathbf{h_i^{i'}}$ is the generated embedding of Φ_i by using probe dataset $P_{i'}$, and it contains both the model's inherent knowledge and noise from $P_{i'}$. To address this, a category loss L_{MKC} is used to incentivize M_{EXT} to learn the knowledge specific to Φ_i, using a distinct index i for each model as the training label,

$$L_{MKC} = CE(\mathbf{h_i^{i'}}, i) \quad (14)$$

where CE is the well-established cross-entropy loss function(Ho and Wookey 2020).

Spatial Alignment Loss. After vectorization, there are still semantic and mapping space bias between $\mathbf{h}$ and $\mathbf{t}$. $\mathbf{h}$ is encoded from two types of samples, while $\mathbf{t}$ aggregates the features of each category, that leads to semantic differences. These differences in mapping space due to M_{EXT} and Q_{EXT} further contribute to the alignment biases. To suppress the biases, we characterize the knowledge consistency with cosine similarity incorporating a margin of 0.4.

$$L_{SAL}(\mathbf{t_i}, \mathbf{h_j}) = \begin{cases} 1 - cos(\mathbf{t_i}, \mathbf{h_j}), & \text{if } i = j \\ \max(0, cos(\mathbf{t_i}, \mathbf{h_j}) - 0.4), & \text{else} \end{cases} \quad (15)$$

where i and j are the indexes of the query task and candidate model, respectively, cos is the cosine distance. Therefore, the final objective function is defined as follows:

$$L = L_{MKC} + \alpha \cdot L_{SAL} \quad (16)$$

where α is a is a constant parameter to balance the different losses, and it is set to 1 in our experiment.

Finally in the well-established knowledge alignment space after training, the model index j with the strongest semantic correlation between $\mathbf{t}$ and candidate model embeddings $\{\mathbf{h_i}\}_{i=1}^m$ can be obtained by the semantic measurement $\mathcal{DIS}()$, which is implemented by the cosine distance.

$$j = \arg\min_i \mathcal{DIS}(\mathbf{t}, \mathbf{h_i}) \quad (17)$$

Experiments

We compare our Know2Vec with several state-of-the-art methods in two scenarios: NNR and SF-MTE. There are four groups of comparison methods.

- **Statistical SF-MTE methods:** H-Score (Bao et al. 2019), NCE (Tran, Nguyen, and Hassner 2019), Leep (Nguyen et al. 2020), NLeep (Li et al. 2021), and LFC (Deshpande et al. 2021).
- **Dynamic SF-MTE methods:** LogME (You et al. 2021) and Model Spider (Zhang et al. 2023).
- **General NNR methods:** TANS (Jeong et al. 2021) and DNNR (Zhong, Qian, and Zhang 2021). Since DNNR requires to train a considerable neural network model for querying data, we do not compare with it.
- **Universal Large language models (LLMs)**: We also examine GPT-4 (Achiam et al. 2023) and Gemini (Team et al. 2023) due to their powerful generation capability.

Performance Comparison on NNR Tasks

Experimental Setup. The evaluation experiment is carried on a modified model-hub created from Kaggle[1] with diverse real-world datasets/models following the methodology outlined in TANS(Jeong et al. 2021). We developed 58 classification models and 232 distinct testing tasks. The probe images are randomly selected from the training dataset of Know2Vec. For fairness, we fine-tuned the model generated from LLMs for 500 steps, as LLMs typically generate neural network rather than select them. In the specific vector computation, the vectors of our approach and TANS method

[1]https://www.kaggle.com/

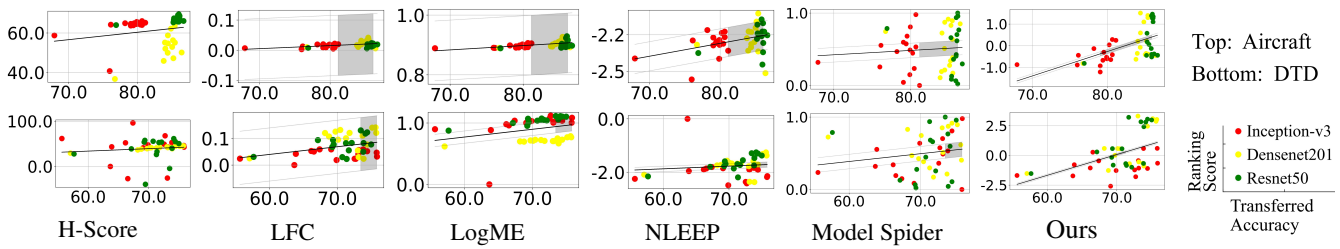

Figure 4: Visualization description of Pearson correlation on SF-MTE experiments.

	R@1	R@3	V. Acc	Ft. Acc	Time	Pri.
H-Score	3.02	7.76	29.07	58.94	23.21	γ
NCE	91.81	**100**	94.03	90.22	10.09	γ
Leep	93.10	**100**	94.33	91.66	11.28	$\gamma\gamma$
NLeep	75.86	92.24	83.84	85.99	10.60	$\gamma\gamma$
LFC	91.38	**100**	92.79	90.25	10.03	$\gamma\gamma$
LogME	50.43	62.93	64.68	77.30	11.32	$\gamma\gamma$
Model Spider	3.87	5.60	27.23	39.18	4.28	$\gamma\gamma$
TANS	82.75	**100**	93.70	94.22	$\leq$ **0.1**	γ
Ours	**94.82**	**100**	**94.87**	**95.67**	$\leq$ **0.1**	$\gamma\gamma\gamma$
GPT-4	-	-	-	46.37	34.48	$\gamma\gamma\gamma$
Gemini	-	-	-	33.87	70.93	$\gamma\gamma\gamma$

Table 1: Performance comparison of NNR tasks.

are 256 dimensions long, while that of Model Spider is 1024 dimensions long.

To thoroughly assess our method against benchmarks, we adopt a suite of established metrics, including: (1) Top-k hitting ratio (R@k,%), that measures the overlap percentage between the top-k prediction results and the ground truth. (2) Valid Accuracy(V.Acc,%) and Fine-Tuned Accuracy(Ft.Acc,%), which quantify the accuracy of the query task on the top-1 selected model and the results after fine-tuning over 50 trials, respectively. (3) Search Time(Time, s). (4) Privacy(Pri.). We categorize privacy into three tiers of model access permissions: white-box access γ, grey-box access $\gamma\gamma$ and black-box access $\gamma\gamma\gamma$.

Experimental Analysis. The quantitative comparison results of NNR task are shown in Table 1. The best score is in bold. As can be seen, in the evaluation of statical methods, Leep achieves higher retrieval accuracy among the evaluated methods, due to its focus on average loglikelihood. However, it falls slightly behind in search time compared to NCE, NLeep, and LFC. H-Score, unfortunately, underperforms in both search time and accuracy, possibly due to its complex calculation and lack of consideration for similarities within categories. Dynamic methods such as Model Spider and LogME also struggle, possibly because of their focus on ranking order of transferability on abundant downstream data, whereas NNR task is more concerned with the performance of the selected top-1 model. Despite its lower accuracy than LogME, Model Spider benefits from vector-based computations, consuming less search time. The same beneficiaries also include our method and TANS. Fortunately, TANS excels in search time and provides substantial retrieval accuracy although it offers only a sub-optimal level of privacy. Our method, while maintains superiority in terms of computation time, offers superior retrieval performance and maintains privacy. Notably, our method achieved a 1.72% increase in retrieval accuracy over the suboptimal result, which demonstrates the superior precision of the knowledge alignment space embedded in our proposed proxy. Undoubtedly, although GPT-4 and Gemini ensure a strong privacy since they do not require access to model zoo, their performances in statistical data falls short of expectations.

Performance Comparison on SF-MTE tasks

Experimental Setup. We construct a heterogeneous model zoo similar to previous work(Zhang et al. 2023), where we collect 48 publicly available pre-trained models trained on diverse datasets[2], covering various neural network architectures. The probe dataset is filtered from several publicly available datasets. We evaluate various methods on 4 different downstream tasks, Aircraft(Maji et al. 2013) and DTD(Cimpoi et al. 2014) for classification, UTKFace (Zhang, Song, and Qi 2017) and dSprites(Matthey et al. 2017) for regression. We leave blank for the regression column of NCE, Leep, NLeep, and LFC since they cannot be used for regression tasks.

We measure the performance of SF-MTE with Pearson(P.)(Cohen et al. 2009) and Spearman(S.)(Hauke and Kossowski 2011) correlation scores, as they are widely adopted (Nguyen et al. 2020; Li et al. 2021; Zhang et al. 2023; Jeong et al. 2021; Deshpande et al. 2021) to evaluate the relationship between the predicted transferability scores and test accuracy.

Experimental Analysis The statistical evaluations of model transferability over classification and regression tasks are shown in Table 2. To provide a clear representation of the correlation between the baseline predictions and the actual accuracy, we visualize the top-6 results of Pearson correlation scores in Fig. 4. In the static methods, LFC and NLeep show a consistent performance, achieving positive and satisfactory scores in both Pearson and Spearman evaluations. By comparison, NCE and Leep show negative correlation coefficients. H-Score performs better on Spearman score than Pearson score, probably because Pearson score is more sensitive to the predicted outlier's scores. In dynamic SF-MTE methods, Model Spider performs poorly, perhaps because its insufficient robustness. In contrast, LogME excels in classification tasks, a testament to the precision of its linear esti-

[2]https://bmwu.cloud//

	Classification				Regression				Mean	
	DTD		Aircraft		UTKFace		dSprites			
	P.	S.	P.	S.	P.	S.	P.	S	P.	S.
H-Score	0.1081	0.2311	0.1915	0.4967	-0.0011	-0.0012	**0.2243**	0.2014	0.0945	0.2320
NCE	-0.1650	-0.2559	0.0845	-0.0229	-	-	-	-	-0.0402	-0.1394
Leep	-0.1672	-0.2229	-0.2079	-0.1718	-	-	-	-	-0.1875	-0.1973
NLeep	0.1153	0.2298	0.3852	0.3213	-	-	-	-	0.2502	0.2755
LFC	0.3508	0.2383	0.4323	0.4733	-	-	-	-	**0.3915**	0.3558
LogME	0.2367	0.3280	0.5310	0.5337	-0.0038	-0.0031	0.1082	0.1114	0.2180	0.2425
Model Spider	0.1705	0.2937	0.0793	0.1263	0.1763	0.1599	-0.0365	-0.0483	0.0974	0.1329
TANS	0.2365	0.2738	-0.3804	-0.3110	-0.0057	0.0091	0.1054	0.1126	-0.0110	0.2112
Ours	**0.4942**	**0.5122**	**0.5545**	**0.5779**	**0.1900**	**0.1909**	0.1917	**0.2608**	0.3576	**0.3854**
GPT-4	-0.3228	-0.1030	-0.2093	-0.0273	0.0475	0.0412	-0.0759	-0.0814	-0.1401	-0.0426
Gemini	-0.0099	0.0828	0.0039	-0.0184	0.0269	0.0764	0.0797	0.1475	0.0251	0.0720

Table 2: Performance comparison on source-free model transferability estimation tasks.

	Q_{EXT}	LSTM	ConCat	Avg.
M_{EXT}	LSTM	**94.82**	92.54	88.14
	ConCat	90.87	94.05	87.37
	Avg.	90.33	89.34	89.35

Table 3: Ablation study of retrieval architecture.

w/o L_{MKC}	L_{SAL}(Cos.)	L_{SAL}(Con.)	P_{train}
82.97	**94.82**	93.53	95.25

Table 4: Ablation study of loss functions and probe dataset.

mation model. TANS struggles in the SF-MTE tasks, while GPT-4 and Gemini display negative or near-zero correlation coefficients on most datasets, indicating a less competitive performance compared to other methods. Our method excels in classification tasks, with improvements of Spearman coefficient reaching 0.1842 and 0.0442 over the sub-optimal result. This indicates a strong correlation between predicted transferability scores and actual test accuracy, as shown in Fig.5, thanks to the semantically rich knowledge vectors and precise matching process. Although our method is not the best in every evaluation dimensions, which may due to being trained only on classification tasks, it still delivers satisfactory results across all tested tasks.

Ablation Study

We assess the performance of each component designed in the proposed proxy on the Kaggle-hub.

Analysis of Knowledge Vectorization Architecture. As shown in Table 3, we explored alternative sequence encoding methods, transitioning from an LSTM network to simpler network such as averaging (Avg.) and concatenation (ConCat). Fixing the structure of M_{EXT} to an LSTM, we found that the result in the first column (94.82%) is significantly higher than those in the second column (92.54%) and third column (88.14%), suggesting that the $LSTM_t$ in Q_{EXT} captures query knowledge more accurately. Likewise, the first row's retrieval accuracy in the first column significantly outperforms the other rows, highlighting the effectiveness of LSTM-based vector encoder in M_{EXT} in extracting detailed model information. We further made T-SNE visualization of model representations before(left) and after(right) encoding in M_{EXT}. In Fig. 5, different colors correspond to the knowledge vectors for different models. The right figure shows an improvement over the left by correctly separating models that were incorrectly clustered together based on their semantics.

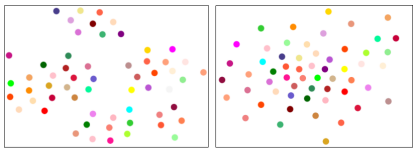

Figure 5: Visual description of model knowledge vectors.

Analysis of Different Loss Functions. First, we tested two unsupervised loss functions, cosine loss (Cos.) and contrastive loss (Con.) for spatial alignment loss L_{SAL}. It can be seen from Table 4, Know2Vec achieved the highest accuracy of 94.82% with L_{SAL}(Cos), allowing for the natural knowledge alignment. Moreover, we observe a slight drop in performance without L_{MKC}, and this highlights the importance of L_{MKC} in filtering noise from model vectors.

Analysis of Different Probe Datasets. In Table 4, the value of P_{train} indicates the model retrieval accuracy when the target model's training dataset is used as the probe dataset, suggesting that alternative dataset might be as effective as the training dataset in generating model knowledge vectors. The same image can serve as a probe to further generate knowledge vectors for different models with alternative dataset, thereby accurately depicting the semantic differences of models in the knowledge consistency space.

Conclusion

In this paper, we propose Know2Vec, a novel proxy for neural network retrieval under a black-box situation. This proxy translates both model knowledge and query data knowledge into vectors, and thus enhancing the accuracy of the retrieval process by ensuring the knowledge consistency among them. The experimental results from NNR and SF-MTE tasks confirm that Know2Vec surpasses the state-of-the-art baseline methods in retrieval precision with acceptable retrieval speed, while also addressing privacy concerns.

Acknowledgments

This work was supported in part by National Natural Science Foundation of China under grant No. 62371450, Ningbo Natural Science Foundation under contract 2022J189, and the Cooperation Project Between Chongqing Municipal Undergraduate Universities and Institutes affiliated to Chinese Academy of Sciences under grant HZ2021015. Additionally, this work was supported by the Chinese Academy of Sciences under grant No. XDB0690302.

References

Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Bao, Y.; Li, Y.; Huang, S.; Zhang, L.; Zheng, L.; Zamir, A.; and Guibas, L. J. 2019. An Information-Theoretic Approach to Transferability in Task Transfer Learning. In *2019 IEEE International Conference on Image Processing, ICIP 2019, Taipei, Taiwan, September 22-25, 2019*, 2309–2313. IEEE.

Cao, X.; Jia, J.; and Gong, N. Z. 2021. IPGuard: Protecting Intellectual Property of Deep Neural Networks via Fingerprinting the Classification Boundary. In Cao, J.; Au, M. H.; Lin, Z.; and Yung, M., eds., *ASIA CCS '21: ACM Asia Conference on Computer and Communications Security, Virtual Event, Hong Kong, June 7-11, 2021*, 14–25. ACM.

Cimpoi, M.; Maji, S.; Kokkinos, I.; Mohamed, S.; and Vedaldi, A. 2014. Describing textures in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3606–3613.

Cohen, I.; Huang, Y.; Chen, J.; Benesty, J.; Benesty, J.; Chen, J.; Huang, Y.; and Cohen, I. 2009. Pearson correlation coefficient. *Noise reduction in speech processing*, 1–4.

Deng, L. 2012. The MNIST Database of Handwritten Digit Images for Machine Learning Research [Best of the Web]. *IEEE Signal Processing Magazine*, 29(6): 141–142.

Deshpande, A.; Achille, A.; Ravichandran, A.; Li, H.; Zancato, L.; Fowlkes, C. C.; Bhotika, R.; Soatto, S.; and Perona, P. 2021. A linearized framework and a new benchmark for model selection for fine-tuning. *CoRR*, abs/2102.00084.

Ding, Y.; Jiang, B.; Yu, A.; Zheng, A.; and Liang, J. 2024. Which Model to Transfer? A Survey on Transferability Estimation. *CoRR*, abs/2402.15231.

Hauke, J.; and Kossowski, T. 2011. Comparison of values of Pearson's and Spearman's correlation coefficients on the same sets of data. *Quaestiones geographicae*, 30(2): 87–93.

Ho, Y.; and Wookey, S. 2020. The Real-World-Weight Cross-Entropy Loss Function: Modeling the Costs of Mislabeling. *IEEE Access*, 8: 4806–4813.

Jeong, W.; Lee, H.; Park, G.; Hyung, E.; Baek, J.; and Hwang, S. J. 2021. Task-Adaptive Neural Network Retrieval with Meta-Contrastive Learning. *CoRR*, abs/2103.01495.

Li, Y.; Jia, X.; Sang, R.; Zhu, Y.; Green, B.; Wang, L.; and Gong, B. 2021. Ranking Neural Checkpoints. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, 2663–2673. Computer Vision Foundation / IEEE.

Maji, S.; Rahtu, E.; Kannala, J.; Blaschko, M.; and Vedaldi, A. 2013. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*.

Matthey, L.; Higgins, I.; Hassabis, D.; and Lerchner, A. 2017. dsprites: Disentanglement testing sprites dataset.

Nguyen, C. V.; Hassner, T.; Seeger, M. W.; and Archambeau, C. 2020. LEEP: A New Measure to Evaluate Transferability of Learned Representations. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, 7294–7305. PMLR.

Team, G.; Anil, R.; Borgeaud, S.; Wu, Y.; Alayrac, J.-B.; Yu, J.; Soricut, R.; Schalkwyk, J.; Dai, A. M.; Hauth, A.; et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Tian, Z.; Wang, Z.; Abdelmoniem, A. M.; Liu, G.; and Wang, C. 2023. Knowledge Representation of Training Data With Adversarial Examples Supporting Decision Boundary. *IEEE Transactions on Information Forensics and Security*, 18: 4116–4127.

Tran, A. T.; Nguyen, C. V.; and Hassner, T. 2019. Transferability and Hardness of Supervised Classification Tasks. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, 1395–1405. IEEE.

You, K.; Liu, Y.; Wang, J.; and Long, M. 2021. LogME: Practical Assessment of Pre-trained Models for Transfer Learning. In Meila, M.; and Zhang, T., eds., *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*, 12133–12143. PMLR.

Yu, Y.; Si, X.; Hu, C.; and Zhang, J. 2019. A review of recurrent neural networks: LSTM cells and network architectures. *Neural computation*, 31(7): 1235–1270.

Zhang, Y.; Huang, T.; Ding, Y.; Zhan, D.; and Ye, H. 2023. Model Spider: Learning to Rank Pre-Trained Models Efficiently. In Oh, A.; Naumann, T.; Globerson, A.; Saenko, K.; Hardt, M.; and Levine, S., eds., *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Zhang, Z.; Song, Y.; and Qi, H. 2017. Age progression/regression by conditional adversarial autoencoder. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 5810–5818.

Zhong, N.; Qian, Z.; and Zhang, X. 2021. Deep Neural Network Retrieval. In Shen, H. T.; Zhuang, Y.; Smith, J. R.; Yang, Y.; César, P.; Metze, F.; and Prabhakaran, B., eds., *MM '21: ACM Multimedia Conference, Virtual Event, China, October 20 - 24, 2021*, 3455–3463. ACM.

Zhou, Z. 2016. Learnware: on the future of machine learning. *Frontiers Comput. Sci.*, 10(4): 589–590.