Towards Unified Task Embeddings Across Multiple Models: Bridging the Gap for Prompt-Based Large Language Models and Beyond

Xinyu Wang[1,2], **Hainiu Xu**[2], **Lin Gui**[2], **Yulan He**[1,2,3]

[1]Department of Computer Science, University of Warwick
[2]Department of Informatics, King's College London
[3]The Alan Turing Institute
Xinyu.Wang.11@warwick.ac.uk
{hainiu.xu, lin.1.gui, yulan.he}@kcl.ac.uk

Abstract

Task embedding, a meta-learning technique that captures task-specific information, has gained popularity, especially in areas such as multi-task learning, model editing, and interpretability. However, it faces challenges with the emergence of prompt-guided Large Language Models (LLMs) operating in a gradient-free manner. Existing task embedding methods rely on fine-tuned, task-specific language models, which hinders the adaptability of task embeddings across diverse models, especially prompt-based LLMs. To hardness the potential of task embeddings in the era of LLMs, we propose a framework for unified task embeddings (FUTE), harmonizing task embeddings from various models, including smaller language models and LLMs with varied prompts, within a single vector space. Such uniformity enables comparison and analysis of similarities amongst different models, broadening the scope and utility of existing task embedding methods in multi-model scenarios, while maintaining their performance comparable to architecture-specific methods.[1]

1 Introduction

Task embeddings serve as a valuable meta-learning tool for various downstream tasks such as multi-task learning (Vu et al., 2020; Li et al., 2022; Zhou et al., 2022; Wu et al., 2023), model editing (Ilharco et al., 2023; Ortiz-Jimenez et al., 2023), and interpretability (Gueta et al., 2023). Current task embedding methods focus on encapsulating the task-specific information into a vector representation, which is derived from parameters of task-specific, fine-tuned models (Vu et al., 2020; Zhou et al., 2022; Ilharco et al., 2023). However, the existing paradigm's reliance on the parameters of a single model restricts the comparison and adaptability of task embeddings across various models, especially hindering their applicability

to prompt-guided, gradient-free Large Language Models (LLMs).

To overcome these challenges, we propose a model-agnostic **F**ramework for **U**nified **T**ask **E**mbedding (**FUTE**), which enables the application of task embedding across diverse models including language models of various architectures and LLMs with different prompts, as illustrated in Figure 1. FUTE generates task embeddings by treating these models as "black boxes" and does not require knowledge of the original models' training datasets. Specifically, we decouple the concept of task embedding into two distinct forms: *Dataset Task Embedding* (DTE) and *Model Task Embedding* (MTE). DTE is designed to capture the task information specific to a dataset, focusing on its unique characteristics. In contrast, MTE captures the model's behavior according to the task performed, independent of its training dataset. By decoupling MTE and DTE, our framework facilitates a more granular analysis of task characteristics, providing insights into both the data and models employed. Moreover, the DTEs and MTEs of various models are computed using a single surrogate model, which replaces the role of the original model in existing methods. This consistent approach to computing DTEs and MTEs ensures that the learned task embeddings reside in the same embedding space, hence facilitating the comparison and analysis of similarities among them.

Building upon the aforementioned improvements, FUTE conceptualizes the combination of a prompt and a LLM as a single model unit, enabling the computation of task embeddings for LLMs guided to perform specific tasks through prompts. Experimental results show that, while being more adaptable and covering a wider range of models, our proposed framework performs on par with existing model-specific embedding methods. FUTE primarily focuses on extending task embedding methods to multi-model scenarios rather

---

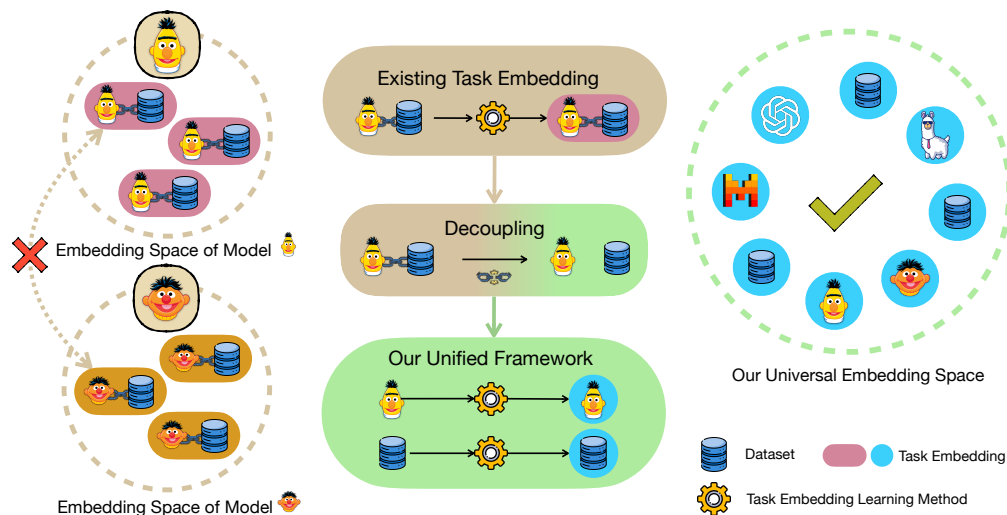[1]Code is available at https://github.com/xnyuwg/fute.

Figure 1: Overview of FUTE. Existing methods generate task embeddings by leveraging model parameters, confining to models of identical architecture and initialization. This limitation, depicted on the left, results in embeddings that reside in separate vector spaces depending on the model, making cross-model similarity comparisons impossible. Our unified framework introduces a method for independently computing task embeddings, enabling the incorporation of diverse models, ranging from different neural architectures to Large Language Models (LLMs) with distinct prompts, into a unified vector space.

than directly improving the embedding methods themselves. Although the performance of FUTE relies on the employed task embedding methods, this aspect represents a strength, allowing FUTE to leverage future advancements in task embedding techniques without significant modifications. Our main contributions are summarized as follows:

- We introduce FUTE, a framework capable of learning unified task embeddings from diverse models, including language models of different architectures, and LLMs with various prompts, within a single vector space.
- We decouple the concept of task embedding into data task embedding and model task embedding.
- Experimental results show that FUTE, while being more versatile, retains a performance to be comparable to existing model-specific methods.

2 Related Work

Task embedding, initially introduced as a task vector (Achille et al., 2019), as a meta-learning method for visual classification tasks. This concept utilizes the empirical Fisher information derived from model gradients to encode task-specific information into a vector representation, demonstrating that the model parameter can provide detailed task information. Expanding into NLP, Vu et al. (2020) applied Fisher information to capture task-specific details in fine-tuned language models, formalizing it as task embedding. Zhou et al.

(2022) explored three Parameter-Efficient Fine-Tuning (PEFT) methods, Prompt Tuning (Liu et al., 2022b), Bias Tuning (Ben Zaken et al., 2022), and Low-Rank Tuning (Hu et al., 2022) to generate task embeddings. They discovered that PEFT parameters, being the only components adjusted during fine-tuning, inherently encapsulate task-specific information. Ilharco et al. (2023) introduced a method with arithmetic operations by subtracting the original parameters from its fine-tuned parameters to extract task vectors. Studies by (Alvarez-Melis and Fusi, 2020; Tan et al., 2021) proposed direct computation of task similarity via Wasserstein Distance, bypassing the need for embedding similarity measures. Additional research efforts, such as (Lv et al., 2023; Huang et al., 2023a,b) have indirectly engaged with concepts akin to task embedding for multi-task learning, albeit without explicitly naming them as such. Recent investigations into the task embedding of LLMs by (Hendel et al., 2023) extracted task vectors from parameters and activations of LLMs. However, similar to previous approaches, this method is confined to a single model and is unable to generate task embeddings for datasets, thereby limiting its applicability. All the aforementioned methods predominantly focused on learning task embeddings for individual models, limiting their application in scenarios involving multiple models.

3 Methodology

3.1 Problem Setup

Existing task embedding learning methods employ a singular embedding to represent both dataset and model task information. In this paper, we refine the concept into two distinct forms: *Dataset Task Embedding* (DTE) and *Model Task Embedding* (MTE). The learning of them is independent and parallel.

Formally, we define a model as M and a dataset as $D = \{(x_i, y_i)\}_{i=1}^n$, where x_i denotes the input and y_i denotes the label. Task embedding is denoted as e, with DTE specified as e_D and MTE as e_M. Typically, the task embedding learning method requires a base model, denoted as Θ. The task embedding learning method $f(\cdot, \cdot)$ is formulated as:

$$e = f\big(\{(x_i, y_i)\}_{i=1}^n, \Theta\big), \tag{1}$$

where the function $f(\cdot, \cdot)$ employs the base model Θ and a set of data instances $\{(x_i, y_i)\}_{i=1}^n$ to compute the corresponding task embedding e. In this paper, $f(\cdot, \cdot)$ encapsulates the complete pipeline of a task embedding learning method, integrating all steps from initial data to the final embedding output. It is important to note that to ensure all task embeddings reside within the same vector space, it is crucial that the function $f(\cdot, \cdot)$, responsible for learning task embeddings, utilizes a consistent base model Θ. This requirement ensures the consistency of the embedding process across different tasks.

3.2 Background

In this paper, we concentrate on extending two task embedding learning methods: TaskEmb (Vu et al., 2020) and TuPaTE (Zhou et al., 2022). TaskEmb employs the Fisher Information Matrix (FIM), which captures the curvature of the loss surface, to learn task embeddings. TuPaTE, on the other hand, introduces Parameter-Efficient Fine-Tuning (PEFT) methods to learn task embedding.

TaskEmb TaskEmb (Vu et al., 2020) employs the FIM with respect to the parameters of a language model to compute task embeddings. The FIM is derived from the expected covariance of the gradients concerning the parameter θ of a language model Θ_M where $\theta \in \Theta_M$ as:

$$F_\theta = \mathop{\mathbb{E}}_{x, y \sim P_\theta(x,y)} \nabla_\theta \log P_\theta(y|x) \nabla_\theta \log P_\theta(y|x)^T. \tag{2}$$

Specifically, TaskEmb computes the empirical Fisher on a fine-tuned model Θ_M using the labels:

$$F_\theta = \frac{1}{n} \sum_{i=1}^n \left[\nabla_\theta \log P_\theta(y_i|x_i) \nabla_\theta \log P_\theta(y_i|x_i)^T \right]. \tag{3}$$

Subsequently, the diagonal entries are extracted and averaged across the entire dataset, and then concatenated to form the task embedding e. The complete pipeline of TaskEmb is denoted as $e = f_{\text{TaskEmb}}(\{(x_i, y_i)\}_{i=1}^n, \Theta_M)$.

TuPaTE TuPaTE demonstrates the application of three PEFT methods: Prompt Tuning (Liu et al., 2022b), Bias Tuning (Ben Zaken et al., 2022), and Low-Rank Tuning (Hu et al., 2022). In this paper, our focus is specifically on TuPaTE with Prompt Tuning. However, FUTE can be easily adapted to other methods in a similar manner.

Specifically, with a language model Θ_M, TuPaTE with Prompt Tuning incorporates additional attention prefix matrices K_p and V_p as parameters. For each layer, these augmented vectors are concatenated with the existing key K and value V to form $K' = [K_p, K]$ and $V' = [V_p, V]$, where $[\cdot, \cdot]$ denotes concatenation. Subsequently, the attention mechanism employs the modified K' and V' for computation. Following PEFT using the prompt K_p and V_p in the language model Θ_M on dataset $\{(x_i, y_i)\}_{i=1}^n$, these prompt vectors are averaged across all layers and concatenated to form the task embedding e. The entire process of TuPaTE is denoted as $e = f_{\text{TuPaTE}}(\{(x_i, y_i)\}_{i=1}^n, \Theta_M)$.

3.3 Model Task Embedding

Model Task Embedding (MTE) is designed to encapsulate the task-specific capabilities of a model within an embedding. Previous methods require that task embeddings be derived from the same base model Θ_M. Furthermore, these methods are closely tied to datasets, constraining the embeddings to models trained on specific datasets, as illustrated in Figure 2A. The dependency on datasets restricts applicability in scenarios where datasets, such as those for LLMs, are unavailable or inaccessible due to privacy concerns. To overcome these limitations, we introduce a model-agnostic and dataset-agnostic approach for extracting MTE. This method offers advantages in scenarios involving varied models and in contexts where training data is inaccessible.

MTE Framework FUTE eliminates the dependency on a specific dataset D for understanding the task information of a model, as depicted in Figure 2C, by utilizing an unsupervised text data $U = \{x_i'\}_{i=1}^m$, unrelated to specific tasks, to substitute the conventional task-specific dataset $D = \{(x_i, y_i)\}_{i=1}^n$. By applying the model Θ_M
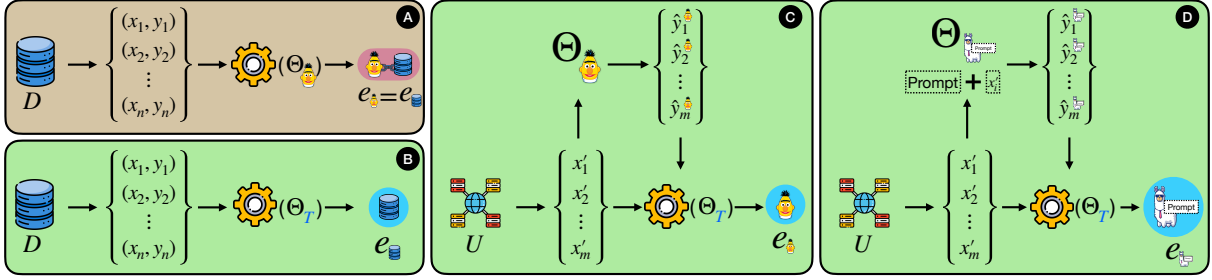
Figure 2: Comparison between FUTE and existing methods. (A) Existing methods typically utilize data $D = \{(x_i, y_i)\}_{i=1}^{n}$ and model Θ_M to generate task embedding e for both the dataset and the model. (B) FUTE derives dataset task embedding (DTE) e_D by introducing an independent surrogate base model Θ_T. (C) FUTE further advances by deriving model task embedding (MTE) e_M by incorporating unsupervised data U to produce alternative input $\{(x_i', \hat{y}_i^M)\}_{i=1}^{m}$, enabling model-specific embeddings without direct dependency on task data. (D) Additionally, FUTE computes MTE for Large Language Models (LLMs) with prompts by treating the combination of a prompt and an LLM as a single model.

on $U = \{x_i'\}_{i=1}^{m}$, we first generate predictions $\{\hat{y}_i^M\}_{i=1}^{m}$ according to:

$$\hat{y}_i^M = \Theta_M(x_i'). \tag{4}$$

This approach is based on the premise that different task-specific models, such as sentiment analysis or question answering, will yield divergent outcomes even for identical text inputs. Here, $\Theta_M(x_i')$ captures task-related information inherent to Θ_M.

Additionally, FUTE introduces a surrogate base model for task embedding learning, denoted as Θ_T, as a substitute for Θ_M. This substitution preserves the uniformity of Θ_T throughout the task embedding learning process, eliminating the need for the same Θ_M. The T5-base (Raffel et al., 2020) is employed as Θ_T in our experiments. We pair each x_i' with its corresponding task-related $\hat{y}_i^M$ to feed into the task embedding learning method $f(\cdot, \cdot)$, deriving MTE e_M as:

$$e_M = f(\{(x_i', \hat{y}_i^M)\}_{i=1}^{m}, \Theta_T). \tag{5}$$

This mechanism allows for the adaptation of the MTE learning process to various models, with the model output $\{\hat{y}_i^M\}_{i=1}^{m}$ reflecting the nature of the task of Θ_M, be it a probability, class, or sentence.

MTE for Prompt-based LLMs Previous methods typically rely on task-specific model parameters and datasets, which are not applicable or accessible for LLMs, especially those closed-source LLMs. Conversely, FUTE can be easily extended to accommodate prompt-based LLMs. In essence, a prompt in conjunction with an LLM can be treated as a unified model from which its MTE can be derived. Prompts can vary from simple instructions

to more complex In-Context Learning examples (ICL) and Chain of Thought (CoT) prompts.

We use the same MTE framework proposed above, and the only difference is that the input to Eq. (4) includes a prompt in addition to the original input x', as shown in Figure 2D. Even though the LLM here, Θ_M, may be significantly larger in size compared to the MTE learning model, Θ_T, an LLM guided by a prompt, which is limited to performing a specific task, can be managed by Θ_T.

This method holds particular promise for LLMs, where different prompts can trigger varied task-specific performances, thereby offering a nuanced understanding of the task capability of LLMs without being limited to specific fine-tuned parameters.

Sourcing Unsupervised Data For sourcing unsupervised text data U, we opt for the CrossFit dataset (Ye et al., 2021). CrossFit encompasses a dataset collection of 160 diverse NLP tasks, offering a rich variety of text styles and structures. This diversity stands in contrast to large, unsupervised text corpora such as Wikipedia, which predominantly feature formal and neutral tones, and consistent style. Such uniformity in text data can skew the performance of models towards, for example, neutral predictions in tasks such as sentiment analysis. This bias may require a much larger volume of data to develop a comprehensive understanding of the model's capabilities. Given this context, the broad spectrum of text characteristics found in CrossFit makes it an ideal choice for U in our study, especially when considering the efficiency of MTE learning.

Discussion of MTE   Viewed from an alternative perspective, this MTE learning approach resembles the principles of knowledge distillation, as described in (Hinton et al., 2015), by utilizing extensive unsupervised data. It effectively facilitates the transfer of knowledge from the original model, Θ_M, to the surrogate base model, Θ_T.

Theoretically, FUTE can learn MTEs for any models with accessible outputs $\{\hat{y}_i^M\}_{i=1}^m$, which potentially encompass not only language models and LLMs but also traditional neural networks like CNN (Fukushima, 1980) and LSTM (Hochreiter and Schmidhuber, 1997), as well as classical models such as decision trees and Support Vector Machines (Hearst et al., 1998).

3.4   Dataset Task Embedding

Dataset Task Embedding (DTE) aims to encapsulate the task information of a dataset within an embedding. In this paper, DTE is conceptualized similarly to the task embedding methods presented in prior studies. The process of learning DTE closely aligns with that of MTE utilizing Θ_T. Specifically, as illustrated in Figure 2B, given a dataset $D = \{(x_i, y_i)\}_{i=1}^n$, the computation of DTE e_D is revised as follows:

$$e_D = f\big(\{(x_i, y_i)\}_{i=1}^n, \Theta_T\big). \tag{6}$$

This reformulation facilitates a more flexible and adaptable approach for generating task embeddings, by leveraging Θ_T without being bounded by the constraints of architectural and initialization uniformity imposed by Θ_M.

4   Experiments

We conducted two sets of experiments focused on smaller language models and LLMs.

4.1   Language Model Experiments

4.1.1   Experimental Setup

To assess the effectiveness of FUTE with language models, we follow (Vu et al., 2020; Zhou et al., 2022) to evaluate the transferability across 11 Classification or Regression (CR) and 11 Question Answering (QA) datasets. Vu et al. (2020); Zhou et al. (2022) conducted a transfer learning experiment, where initially, a language model is fine-tuned on a source dataset, followed by further fine-tuning on a target dataset. Based on the similarity between task embeddings learned from BERT-base (Devlin et al., 2019), a source dataset is selected for a given

target dataset, promising to achieve optimal transfer learning gain. Evaluating the selection process highlights the effectiveness of task embedding.

Our approach to the experiment diverges from the original method by shifting the focus from selecting the most appropriate source dataset to identifying the most compatible models for a target dataset. This is achieved by matching the MTEs of candidate models with the DTE of the target dataset based on their embedding similarity. More experimental setup details are given below:

Datasets   CR datasets comprise nine datasets from the GLUE benchmark (Wang et al., 2019), in addition to SNLI (Bowman et al., 2015) and Sci-Tail (Khot et al., 2018). QA datasets contain eleven datasets from the MultiQA repository (Talmor and Berant, 2019). See Appendix A.1 for more details.

Unsupervised Dataset   We randomly sampled 100 text entries from each task within CrossFit (Ye et al., 2021) to ensure broad text styles and structures. We removed any datasets that overlap with the CR or QA datasets mentioned above, discarding any associated labels. This process resulted in the compilation of 104,162 text entries for U. In our experimental setup, we fine-tune Θ_T on the unsupervised data for one epoch.

Metric   We follow (Vu et al., 2020; Zhou et al., 2022) and employ two metrics: (1) the average rank ρ of the candidate source model that yields the highest transfer gain; (2) the Normalized Discounted Cumulative Gain (NDCG) (Järvelin and Kekäläinen, 2002), which assesses the overall quality of the entire ranking.

We present two extensions for TaskEmb and TuPaTE with FUTE: (1) **FUTE+TaskEmb**: It combines FUTE with the TaskEmb method, utilizing Eq. (5) and (6), in conjunction with the TaskEmb method $f_{\text{TaskEmb}}(\cdot, \cdot)$. (2) **FUTE+TuPaTE**: It applies FUTE on TuPaTE, employing Eq. (5) and (6) alongside TuPaTE method $f_{\text{TuPaTE}}(\cdot, \cdot)$.

4.1.2   Baselines

In language model experiment, we compare FUTE with the following baselines: (1) **DataSize** (Vu et al., 2020) prioritises dataset based on the size of dataset. (2) **CurveGrad** (Bingel and Søgaard, 2017; Vu et al., 2020) analyzes the gradients of the loss curve associated with the language model, and is adapted by (Vu et al., 2020) for transferability prediction. (3) **TextEmb** (Vu et al., 2020)

Method	CR				QA			
	in-class		*all-class*		*in-class*		*all-class*	
	$\rho\downarrow$	NDCG $\uparrow$	$\rho\downarrow$	NDCG $\uparrow$	$\rho\downarrow$	NDCG $\uparrow$	$\rho\downarrow$	NDCG $\uparrow$
DataSize	3.6	80.4	7.8	75.2	3.2	84.4	11.4	65.8
CurveGrad	5.5	68.6	-	-	8.3	64.8	-	-
TextEmb	5.2	76.4	9.8	74.7	4.1	81.1	5.8	82.0
TaskEmb	2.8	82.3	5.4	78.3	3.2	84.5	5.4	82.8
TuPaTE	**2.5**	83.7	**4.5**	81.0	**3.0**	**85.7**	4.8	83.3
FUTE + TaskEmb	4.4	79.4	7.0	77.9	4.5	83.5	5.3	84.3
FUTE + TuPaTE	3.3	**83.8**	6.2	**82.0**	3.3	85.6	**4.1**	**84.8**

Table 1: Transferability results of average rank ρ and NDCG (%) on classification or regression (CR) and question answering (QA) datasets. The terms *in-class* and *all-class* denote scenarios where the source datasets are within the same category (CR or QA) or across the two categories, respectively. For example, *in-class* results for CR indicate that the source datasets are exclusively from the CR category.

computes an average of the text representations extracted from the last layer of the language model across the entire dataset. (4) **TaskEmb** (Vu et al., 2020), as detailed in Section 3.2, leverages Finisher information to derive task embedding. (5) **TuPaTE** (Zhou et al., 2022), outlined in Section 3.2, utilize PEFT with the language model, extracting the tuned prompt as task embedding.

All these baselines are unavailable in the multi-model scenario. Dataset-centric methods, like DataSize, inherently cannot differentiate between models trained on the same dataset. Model-specific approaches, CurveGrad, TextEmb, TaskEmb, and TuPaTE, all require a single model initialization, limiting their adaptability across diverse models. CurveGrad computes the gradients from the same model. TextEmb extracts the text representations from the same model. For example, similarly, text embedding derived from BERT-base, BERT-large, and GPT2 do not reside in the same vector space. However, FUTE can be potentially adapted to extend text embedding methods as well. TaskEmb and TuPaTE, as mentioned above, are also unavailable for multiple models.

4.1.3 Transferability Results

The Transferability results are presented in Table 1. Upon comparison, FUTE exhibits competitive performance relative to the original methods. Specifically, when integrated with TaskEmb, FUTE achieves better results in the QA *all-class* setting compared to the original TaskEmb method, albeit with slightly less optimised outcomes in other settings. Similarly, the combination of FUTE with TuPaTE demonstrates improved performance in the *all-class* setting, while yielding very similar outcomes in the *in-class* setting. Overall, the dis-

crepancies between FUTE and the original methods are generally confined to less than a 2% difference.

4.1.4 Cross-Domain Results

FUTE utilizes consistent unsupervised data U for MTEs across all models. This approach diverges from previous methodologies, where each task embedding is tied to a specific dataset. We hypothesise that task embeddings generated by previous methods may encapsulate domain-specific information due to their dataset-centric nature. To test this hypothesis, we conduct experiments using three datasets from the finance and medical domains: FinancialPhraseBank (Malo et al., 2014), Medical Question Pairs (MQP) (McCreery et al., 2020), and MedQA (Jin et al., 2021). We randomly split 20% of the training set to serve as the validation set. These datasets exhibit text characteristics distinct from the general text content presented in the datasets used in Section 4.1.3.

The comparative results, displayed in Table 2, reveal that TaskEmb maintains consistent performance across this cross-domain experiment, closely mirroring the results obtained from FUTE. TaskEmb leverages the parameters of the entire language model, potentially embedding within it a wider range of domain information. Conversely, TuPaTE extracts task embeddings from the PEFT parameters, which may be more narrowly tailored to specific domain characteristics. This difference suggests that TaskEmb's embeddings could carry broader information due to its reliance on the comprehensive model parameters.

Remarkably, FUTE using TuPaTE significantly outperforms the original method. This improvement can be attributed to the utilization of consistent unsupervised data U across all models, thereby

Method	$\rho\downarrow$	NDCG $\uparrow$
TaskEmb	5.6	80.9
TuPaTE	10.7	73.8
FUTE + TaskEmb	**4.7**	**81.8**
FUTE + TuPaTE	7.0	81.1

Table 2: Cross-domain results of average rank ρ and NDCG (%) on financial and medical domain datasets.

reducing the domain influence from the data source. The findings from Table 2 indicate the promising potential of FUTE to enhance the adaptability and effectiveness in cross-domain applications.

4.1.5 Visualization

In this section, we present a visualization in Figure 3 by extending TuPaTE to three language models, BERT-base (Devlin et al., 2019), GPT2 (Radford et al., 2019), and T5-base (Raffel et al., 2020), based on with CR and QA datasets. A notable observation from the visualization is the emergence of two main clusters based on task information, highlighting the significant distinctions of MTEs between task categories. MTEs associated with the same language model, particularly for QA tasks, demonstrate a tendency to cluster together, which indicates that the inherent characteristics of a language model also influence the task embedding space. An interesting detail is the presence of three blue points within the CR task cluster, representing MTEs associated with the BoolQ dataset (Clark et al., 2019). This positioning matches BoolQ's characteristics with CR tasks due to its requirement for boolean answer generation for the question, which closely resembles CR tasks. This positioning matches with the characteristics of BoolQ, which is formatted to predict a boolean answer for each question and closely resembles CR tasks. See Appendix A.2 for more details.

4.2 LLMs Experiments

4.2.1 Experimental Setup

In this section, we assess the effectiveness of FUTE for LLMs by focusing on the task of zero-shot prompt instruction selection. The objective is to identify the most suitable prompt instruction that enables an LLM to achieve optimal performance on a given dataset. We utilize FUTE to generate MTEs of LLMs associated with prompts using Eq. (5), and compute DTE for the target dataset using Eq. (6). We then compute the similarity between MTEs and DTEs for prompt selection.
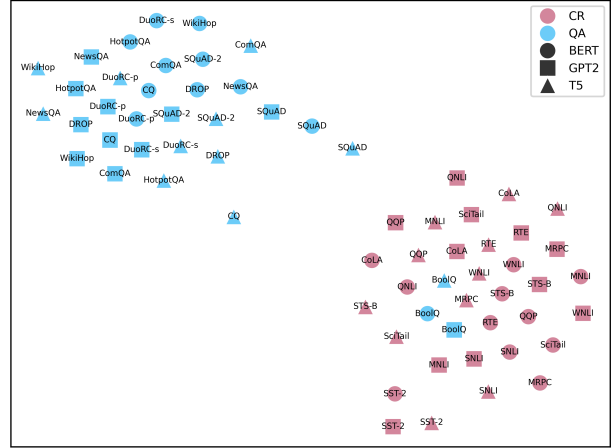

Figure 3: T-SNE visualization of language model MTEs using FUTE with TuPaTE. Different task categories are represented by various colors, while different models are indicated by distinct shapes. E.g., a red circle highlights an MTE of BERT trained on a CR dataset.

Datasets   We utilize two categories of tasks for evaluation: Sentiment Analysis (SA) and Natural Language Inference (NLI). For SA, we employ: Rotten Tomatoes (Pang and Lee, 2005), IMDB (Maas et al., 2011), and SST-2 (Socher et al., 2013). For NLI, we use: WANLI (Liu et al., 2022a), CB (De Marneffe et al., 2019), and ANLI (Round 3) (Nie et al., 2020). We randomly sample 1,000 data from the test set of each dataset for evaluation.

Unsupervised Dataset   We employ CrossFit (Ye et al., 2021) as the source of unsupervised data U and randomly sample 10,000 diverse text entries from U in Section 4.1. Again, we ensure no overlap between U and datasets D, and exclusively use text content, discarding any associated labels.

LLMs   We employ Llama2-13B (Touvron et al., 2023) in 8 bit, Llama2-70B (Touvron et al., 2023) in 4 bit, and Mixtral 8x7B (Jiang et al., 2024) in 8 bit. Mixtral 8x7B is a Sparse Mixture of Experts (SMoE) architecture, featuring 8 experts with each token processed by two experts.

Prompts   We incorporate a diverse set of 13 zero-shot prompts for each category of tasks, which comprises 10 vanilla instruction prompts sourced from PromptSource (Bach et al., 2022) and 3 Chain of Thought (CoT) prompts. Each prompt is structured to include instructions only, without any ICL examples. See Appendix A.3 for more details.

Metric   We utilize three metrics: (1) the performance of the selected prompt; (2) performance rate,

Category	Method	Llama 2 13B			Llama 2 70B			Mixtral 8x7B		
		Performance	Rate	NDCG	Performance	Rate	NDCG	Performance	Rate	NDCG
SA	MI	88.0	94.4	59.0	85.3	90.9	72.9	87.2	85.1	65.1
	LocalE	84.3	90.2	47.5	88.3	88.7	57.5	**88.5**	**96.7**	78.6
	GlobalE	89.2	95.7	88.8	91.9	97.9	**82.7**	88.4	96.6	**86.7**
	ZPS-Log	54.3	58.2	38.5	78.0	83.0	54.0	57.0	62.1	33.9
	ZPS-Prob	54.3	58.2	38.5	78.0	83.0	50.8	57.0	62.1	33.9
	ZPS-Vote	54.3	58.2	38.5	78.0	83.0	50.8	57.0	52.1	33.9
	Self-Select	54.3	58.2	42.8	85.3	90.9	69.3	57.0	62.1	38.5
	SPELL	89.2	95.7	**89.6**	79.6	84.6	65.4	57.0	62.1	38.2
	FUTE + TaskEmb	**89.6**	**96.1**	89.4	**93.0**	**99.0**	74.5	86.9	94.9	71.1
	FUTE + TuPaTE	89.2	95.7	55.6	92.4	98.4	67.9	87.9	95.8	52.2
NLI	MI	**46.8**	**90.1**	61.7	48.6	94.8	74.6	37.2	73.6	36.7
	LocalE	37.5	74.4	56.4	43.9	84.6	66.6	39.1	78.6	43.5
	GlobalE	40.4	80.4	65.3	48.2	93.7	76.9	40.2	79.1	44.1
	ZPS-Log	34.8	70.2	48.1	34.9	66.8	49.5	39.0	76.5	41.7
	ZPS-Prob	32.6	65.6	39.7	38.0	73.2	51.2	39.5	78.9	39.3
	ZPS-Vote	32.6	65.6	39.7	33.7	64.3	48.4	39.5	78.9	39.3
	Self-Select	33.9	68.1	39.5	39.7	76.7	53.6	39.1	78.6	43.9
	SPELL	42.4	84.4	**78.1**	48.6	94.8	77.2	39.1	78.6	41.5
	FUTE + TaskEmb	35.8	72.0	47.4	41.1	78.8	60.8	**43.2**	**85.6**	**49.1**
	FUTE + TuPaTE	37.0	75.0	71.8	**50.6**	**98.4**	**81.8**	40.8	81.3	44.4

Table 3: Zero-shot prompt selection results of Performance (%), Performance Rate (%), and NDCG (%) on Sentiment Analysis (SA) and Natural Language Inference (NLI) datasets.

computed as the ratio of the achieved performance with the selected prompt to the maximum performance observed across all prompts. A performance rate of 100% indicates perfect selection; (3) NDCG (Järvelin and Kekäläinen, 2002) evaluates the overall quality of prompt ranking, where relevance is determined by performance. An NDCG of 100% represents an ideal ranking. The final results are averaged across datasets.

4.2.2 Baselines

We compare FUTE against the following baselines: (1) **MI** (Sorensen et al., 2022) selects the prompt that maximizes the mutual information between the input and the LLM output. (2) **LocalE** (Lu et al., 2022) calculates the local entropy of the prediction probabilities for each entry. (3) **GlobalE** (Lu et al., 2022) computes the global entropy across all prediction categories in a dataset. **ZPS** (Liao et al., 2022) leverages an ensemble of prompts to generate pseudo labels and then predicts the performance of each prompt on these pseudo labels. Based on the ensembling technique, it has three variants: (4) **ZPS-Log** using the sum of log probabilities, (5) **ZPS-Prob** using the sum of probabilities, and (6) **ZPS-Vote** using majority vote. (7) **Self-Select** (Ramji and Kyimpopkin, 2023) introduces a meta-prompt that queries LLM for selecting the most effective prompt. (8) **SPELL** (Gonen et al., 2023) predicts prompt performance based on

the perplexity of the prompt.

Here, we also use two extensions with FUTE: **FUTE+TaskEmb** with $f_{TaskEmb}(\cdot, \cdot)$ and **FUTE+TuPaTE** with $f_{TuPaTE}(\cdot, \cdot)$.

4.2.3 Overall Results

Table 3 presents the comparative results. No single baseline consistently outperforms the others across all scenarios. Focusing on performance and rate performance, MI excels on the Llama2-13B for the NLI task. LocalE stands out on the Mixtral for the SA task. FUTE+TaskEmb achieves superior performance on Llama2-13B and Llama2-70B for the SA task. However, given that performance and rate performance only focus on top-1 selection, we delve deeper into the NDCG scores for a comprehensive quality assessment.

GlobalE obtains the highest NDCG on Llama2-70B and Mixtral for the SA task. Its methodology, based on model entropy and confidence, appears less effective for the more challenging NLI task, likely due to the increased uncertainty even for correct prediction in LLM responses. SPELL achieves the best NDCG on Llama2-13B for both SA and NLI task, but underperforms on Llama2-70B and Mixtral. Given SPELL's reliance on perplexity, this outcome may suggest that larger models possess greater resilience in delivering accurate predictions, even in scenarios of poor perplexity.

FUTE outperforms others on Llama2-70B and

Mixtral for NLI tasks, demonstrating the effectiveness of our framework when handling difficult tasks. For the Llama2 model, FUTE+TaskEmb outperforms FUTE+TuPaTE on the SA task, but underperforms on NLI task. This disparity may stem from the inherent complexity of the NLI task, which tends to produce more unconfident and potentially noisier predictions from LLMs. TaskEmb leverages the parameters of the entire model, learning a broad knowledge but also risking overfitting to noisy data. TuPaTE utilizing the PEFT parameter, has a more constrained capacity that may better isolate relevant, non-noisy information. This specialization likely accounts for TaskEmb's superior performance in SA tasks and vice versa. On Mixtral, which is based on SMoE architecture, FUTE+TaskEmb demonstrates superior results. The diverse experts within Mixtral may require more information storage capacity, a requirement that TaskEmb meets. In summary, while each baseline excels in certain tasks/models, FUTE emerges as a stable approach, achieving top-3 results across various scenarios.
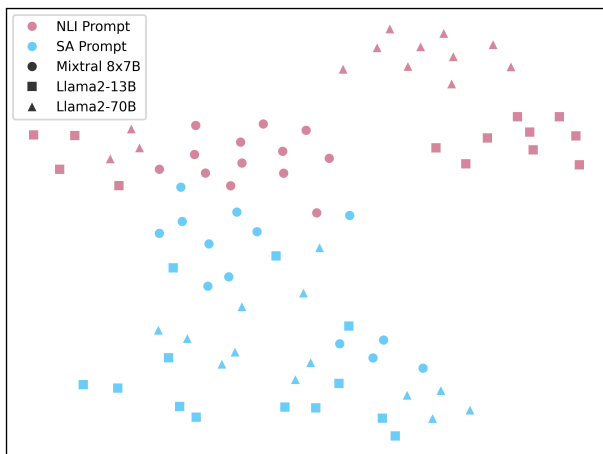
4.2.4 Visualization

Figure 4: T-SNE visualization of LLMs MTE using FUTE with TuPaTE. Prompts related to different task categories are represented by various colors, while different LLMs are indicated by distinct shapes. For example, a blue square indicates the MTE of Llama2-13B guided by an SA prompt.

Figure 4 illustrates the visualization of learned MTEs for LLMs utilizing diverse prompts. This visualization reveals two principal clusters formed based on the prompts, highlighting the significant role prompts play in differentiating LLMs performance across task categories. While prompts across various tasks significantly influence MTEs,

the distinction becomes less pronounced within a single task category, where specific LLMs have a more substantial impact on the embedding outcomes. Especially within the NLI task category, MTEs are further divided into roughly three clusters, each corresponding to different LLMs. Mixtral MTEs also generally cluster towards the centre. See Appendix A.4 for a more detailed visualization, where prompts also slightly influence MTEs within the same task.

4.2.5 Efficiency Analysis

FUTE extends the existing task embedding method and utilizes embedding similarity to select prompts, leading to a distinctive computational complexity profile. The process requires evaluating a total of k_p prompts across k_D datasets. The computational complexity of FUTE is represented as $O((k_p + k_D)c_o)$, where c_o represents the computation cost associated with the task embedding method. In contrast, the computational complexity for other baselines evaluated in this study is expressed as $O(k_p k_D c_b)$, where c_o represents the computation cost associated with baselines. The difference in complexity becomes significant with increasing values of k_p and k_D, where the product $k_p k_D$ will exceed the sum $(k_p + k_D)$. Thus, FUTE presents a potentially more efficient approach to prompt selection, especially as the number of prompts and datasets grows. However, the existing task embedding methods used in FUTE require fine-tuning, thereby $c_o > c_b$. With a relatively small number of prompts (k_p) and datasets (k_D), FUTE does not demonstrate superior running efficiency in comparison to baselines. Nonetheless, c_o is more closely related to the inherent efficiency of the task embedding method itself, whereas FUTE focuses on extending these methods.

5 Conclusion and Future Work

This paper introduces a unified framework FUTE that separate task embeddings into DTE and MTE, by employing an additional surrogate base model and unsupervised data, without altering existing task embedding learning methodologies. FUTE can effectively extend current methods to learn MTEs, which are positioned in a unified vector space, for various language models and LLMs with distinct prompts. FUTE is, theoretically, capable of learning MTEs for any model with accessible outputs, within a single vector space. We leave the exploration to other models in the future.

Limitations

FUTE is designed to extend, rather than enhance, current task embedding learning methods. Consequently, it inherits potential limitations from the methodologies it builds on. However, we argue that this is more of a feature of FUTE rather than a limitation. By not altering current procedures, our framework leverages the strengths of existing task embedding methods, allowing for seamless integration of future advancements in the field. This adaptability means that as stronger task embedding methods emerge in the future, our framework will also exhibit improved performance without requiring modifications. We also acknowledge that any alteration to the surrogate base model requires the re-learning of task embeddings, which could be computationally intensive. However, we consider this issue intrinsic to the existing task embedding learning methods. If future developments in task embedding learning methods eliminate dependency on the base model, FUTE can be adjusted by simply omitting the surrogate base model as well.

Ethics Statement

FUTE aims to address challenges related to the interpretability of task embeddings across diverse models, especially for prompt-guided LLMs. While this framework holds promise for enhancing the utility and comparability of task embeddings across different models, it also raises some ethical concerns. For example, learning task embeddings requires the use of unsupervised data, which may not adequately encapsulate the essential attributes of a particular task, potentially resulting in biased task embeddings. Moreover, the reliance on a surrogate T5 model introduces potential ethical issue related to transparency. If the surrogate model fails to accurately replicate the behavior of the target model, it could lead to misleading results.

Acknowledgements

References

Abdalghani Abujabal, Rishiraj Saha Roy, Mohamed Yahya, and Gerhard Weikum. 2019. ComQA: A community-sourced dataset for complex factoid question answering with paraphrase clusters. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 307–317, Minneapolis, Minnesota. Association for Computational Linguistics.

Alessandro Achille, Michael Lam, Rahul Tewari, Avinash Ravichandran, Subhransu Maji, Charless Fowlkes, Stefano Soatto, and Pietro Perona. 2019. Task2vec: Task embedding for meta-learning. In *2019 IEEE/CVF International Conference on Computer Vision*, pages 6429–6438.

David Alvarez-Melis and Nicoló Fusi. 2020. Geometric dataset distances via optimal transport. In *Advances in Neural Information Processing Systems*.

Stephen H. Bach, Victor Sanh, Zheng-Xin Yong, Albert Webson, Colin Raffel, Nihal V. Nayak, Abheesht Sharma, Taewoon Kim, M Saiful Bari, Thibault Fevry, Zaid Alyafeai, Manan Dey, Andrea Santilli, Zhiqing Sun, Srulik Ben-David, Canwen Xu, Gunjan Chhablani, Han Wang, Jason Alan Fries, Maged S. Al-shaibani, Shanya Sharma, Urmish Thakker, Khalid Almubarak, Xiangru Tang, Xiangru Tang, Mike Tian-Jian Jiang, and Alexander M. Rush. 2022. Promptsource: An integrated development environment and repository for natural language prompts.

Junwei Bao, Nan Duan, Zhao Yan, Ming Zhou, and Tiejun Zhao. 2016. Constraint-based question answering with knowledge graph. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2503–2514, Osaka, Japan. The COLING 2016 Organizing Committee.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1–9, Dublin, Ireland. Association for Computational Linguistics.

Joachim Bingel and Anders Søgaard. 2017. Identifying beneficial task relations for multi-task learning in deep neural networks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 164–169, Valencia, Spain. Association for Computational Linguistics.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota. Association for Computational Linguistics.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The pascal recognising textual entailment challenge. In *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, pages 177–190, Berlin, Heidelberg. Springer Berlin Heidelberg.

Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The commitmentbank: Investigating projection in naturally occurring discourse. In *proceedings of Sinn und Bedeutung*, volume 23, pages 107–124.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Kunihiko Fukushima. 1980. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36:193–202.

Hila Gonen, Srini Iyer, Terra Blevins, Noah Smith, and Luke Zettlemoyer. 2023. Demystifying prompts in language models via perplexity estimation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10136–10148, Singapore. Association for Computational Linguistics.

Almog Gueta, Elad Venezian, Colin Raffel, Noam Slonim, Yoav Katz, and Leshem Choshen. 2023. Knowledge is a region in weight space for fine-tuned language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 1350–1370, Singapore. Association for Computational Linguistics.

M.A. Hearst, S.T. Dumais, E. Osuna, J. Platt, and B. Scholkopf. 1998. Support vector machines. *IEEE Intelligent Systems and their Applications*, 13(4):18–28.

Roee Hendel, Mor Geva, and Amir Globerson. 2023. In-context learning creates task vectors. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9318–9333, Singapore. Association for Computational Linguistics.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. *ArXiv*, abs/1503.02531.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9:1735–1780.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Chengsong Huang, Qian Liu, Bill Yuchen Lin, Tianyu Pang, Chao Du, and Min Lin. 2023a. Lorahub: Efficient cross-task generalization via dynamic lora composition.

Kuan-Hao Huang, Liang Tan, Rui Hou, Sinong Wang, Amjad Almahairi, and Ruty Rinott. 2023b. Learning easily updated general purpose text representations with adaptable task-specific prefix. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 7422–7430, Singapore. Association for Computational Linguistics.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2023. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations*.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample,

L'elio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of experts. *ArXiv*, abs/2401.04088.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. 2021. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14):6421.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. Scitail: A textual entailment dataset from science question answering. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Hector J. Levesque, Ernest Davis, and L. Morgenstern. 2011. The winograd schema challenge. In *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*.

Geng Li, Boyuan Ren, and Hongzhi Wang. 2022. Eeml: Ensemble embedded meta-learning. In *Web Information Systems Engineering – WISE 2022*, pages 433–442, Cham. Springer International Publishing.

Chonghua Liao, Yanan Zheng, and Zhilin Yang. 2022. Zero-label prompt selection. *ArXiv*, abs/2211.04668.

Alisa Liu, Swabha Swayamdipta, Noah A. Smith, and Yejin Choi. 2022a. WANLI: Worker and AI collaboration for natural language inference dataset creation. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 6826–6847, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022b. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68, Dublin, Ireland. Association for Computational Linguistics.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8086–8098, Dublin, Ireland. Association for Computational Linguistics.

Xingtai Lv, Ning Ding, Yujia Qin, Zhiyuan Liu, and Maosong Sun. 2023. Parameter-efficient weight ensembling facilitates task-level knowledge transfer. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 270–282, Toronto, Canada. Association for Computational Linguistics.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

P. Malo, A. Sinha, P. Korhonen, J. Wallenius, and P. Takala. 2014. Good debt or bad debt: Detecting semantic orientations in economic texts. *Journal of the Association for Information Science and Technology*, 65.

Clara H. McCreery, Namit Katariya, Anitha Kannan, Manish Chablani, and Xavier Amatriain. 2020. Effective transfer learning for identifying similar questions: Matching user questions to covid-19 faqs.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901, Online. Association for Computational Linguistics.

Guillermo Ortiz-Jimenez, Alessandro Favero, and Pascal Frossard. 2023. Task arithmetic in the tangent space: Improved editing of pre-trained models. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 115–124, Ann Arbor, Michigan. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Keshav Ramji and Alexander Kyimpopkin. 2023. Self-select: Optimizing instruction selection for large language models. In *NeurIPS 2023 Foundation Models for Decision Making Workshop*.

Amrita Saha, Rahul Aralikatte, Mitesh M. Khapra, and Karthik Sankaranarayanan. 2018. DuoRC: Towards complex language understanding with paraphrased reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1683–1693, Melbourne, Australia. Association for Computational Linguistics.

Nikhil Dandekar Shankar Iyer and Kornél Csernai. 2017. First Quora Dataset Release: Question pairs.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Taylor Sorensen, Joshua Robinson, Christopher Rytting, Alexander Shaw, Kyle Rogers, Alexia Delorey, Mahmoud Khalil, Nancy Fulda, and David Wingate. 2022. An information-theoretic approach to prompt engineering without ground truth labels. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 819–862, Dublin, Ireland. Association for Computational Linguistics.

Alon Talmor and Jonathan Berant. 2019. MultiQA repository.

Yang Tan, Yang Li, and Shao-Lun Huang. 2021. Otce: A transferability metric for cross-domain cross-task representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15779–15788.

Hugo Touvron, Louis Martin, Kevin R. Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Daniel M. Bikel, Lukas Blecher, Cristian Cantón Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony S. Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel M. Kloumann, A. V. Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, R. Subramanian, Xia Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zhengxu Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and

Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *ArXiv*, abs/2307.09288.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. NewsQA: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200, Vancouver, Canada. Association for Computational Linguistics.

Tu Vu, Tong Wang, Tsendsuren Munkhdalai, Alessandro Sordoni, Adam Trischler, Andrew Mattarella-Micke, Subhransu Maji, and Mohit Iyyer. 2020. Exploring and predicting transferability across NLP tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 7882–7926, Online. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Constructing datasets for multi-hop reading comprehension across documents. *Transactions of the Association for Computational Linguistics*, 6:287–302.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Chengyue Wu, Teng Wang, Yixiao Ge, Zeyu Lu, Ruisong Zhou, Ying Shan, and Ping Luo. 2023. π-tuning: Transferring multimodal foundation models with optimal multi-task interpolation. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 37713–37727. PMLR.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Qinyuan Ye, Bill Yuchen Lin, and Xiang Ren. 2021. CrossFit: A few-shot learning challenge for cross-task generalization in NLP. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7163–7189, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Wangchunshu Zhou, Canwen Xu, and Julian McAuley. 2022. Efficiently tuned parameters are task embeddings. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5007–5014, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

A Appendix

A.1 Datasets in Language Model Experiment

Text classification or regression (CR) contain: SNLI (Bowman et al., 2015), MNLI (Williams et al., 2018), QQP (Shankar Iyer and Csernai, 2017), QNLI (Wang et al., 2019), SST-2 (Socher et al., 2013), SciTail (Khot et al., 2018), CoLA (Warstadt et al., 2019), STS-B (Cer et al., 2017), MRPC (Dolan and Brockett, 2005), RTE (Dagan et al., 2006), WNLI (Levesque et al., 2011).

Question answering (QA) datasets contain: SQuAD-2 (Rajpurkar et al., 2018), NewsQA (Trischler et al., 2017), HotpotQA (Yang et al., 2018), SQuAD-1 (Rajpurkar et al., 2016), DuoRC-p (Saha et al., 2018), DuoRC-s (Saha et al., 2018), DROP (Dua et al., 2019), WikiHop (Welbl et al., 2018), BoolQ (Clark et al., 2019), ComQA (Abujabal et al., 2019), CQ (Bao et al., 2016).

A.2 Visualization of Language Model Task Embedding

Figure A1 presents the visualization of task embeddings for language models learned by FUTE with TuPaTE. This visualization highlights the emergence of distinct clusters that reflect task-specific information, revealing that task embeddings are influenced by both language models and trained datasets:

Datasets Beyond the primary task clusters, some datasets demonstrate a pronounced impact on the formation of task embeddings. For example, models trained on SNLI and SST-2 exhibit a tendency to cluster together within the same task. A particular observation is the grouping of three blue points, representing task embeddings associated with the BoolQ, within the CR task cluster. This matches the format of BoolQ, a boolean answer for a question, which aligns more closely with CR tasks.

Language Model MTEs associated with the same language model, particularly BERT and GPT2, demonstrate a propensity to cluster together. This phenomenon indicates that the inherent characteristics of a language model also influence the task embedding space.

A.3 Prompt Template

Table A1 and Table A2 list used prompts in SA and NLI in Section 4.2. Each contains 10 vanilla prompts and 3 CoT prompts.

A.4 Visualization of LLM Task Embedding with Prompts

Figure A2 demonstrates a visualization where prompt numbers are detailed in Table A1 and Table A2. Prompt task categories are differentiated by colors, and distinct shapes represent various LLMs. This visualization reveals that task embeddings are influenced by both the LLMs and the prompts used as follows:

Influence of Prompt This visualization demonstrates two influences: inter-task and intra-task influences. For inter-task influences, task embeddings are clearly separated into two main categories based on the task, indicating that prompts have a substantial impact on LLM performance across different tasks. For intra-task influences, within the same task, prompts also slightly influence task embeddings. For example, in the NLI task, all MTEs of prompt#9 are closer to the SA cluster than MTEs from the same LLM. Conversely, MTEs of prompt#10 diverge from the SA cluster. Also, the Chain of Thought (CoT) prompts#11, 12, and 13 form their own cluster, distinct from the main LLM-based clusters. A similar phenomenon is observed within the SA task, where prompts#7 and 8 diverge from the NLI cluster.

Influence of LLM Task embeddings tend to cluster by LLM, particularly for the NLI task, where embeddings from the same LLM cluster together. Mixtral's task embeddings also generally cluster towards the center.
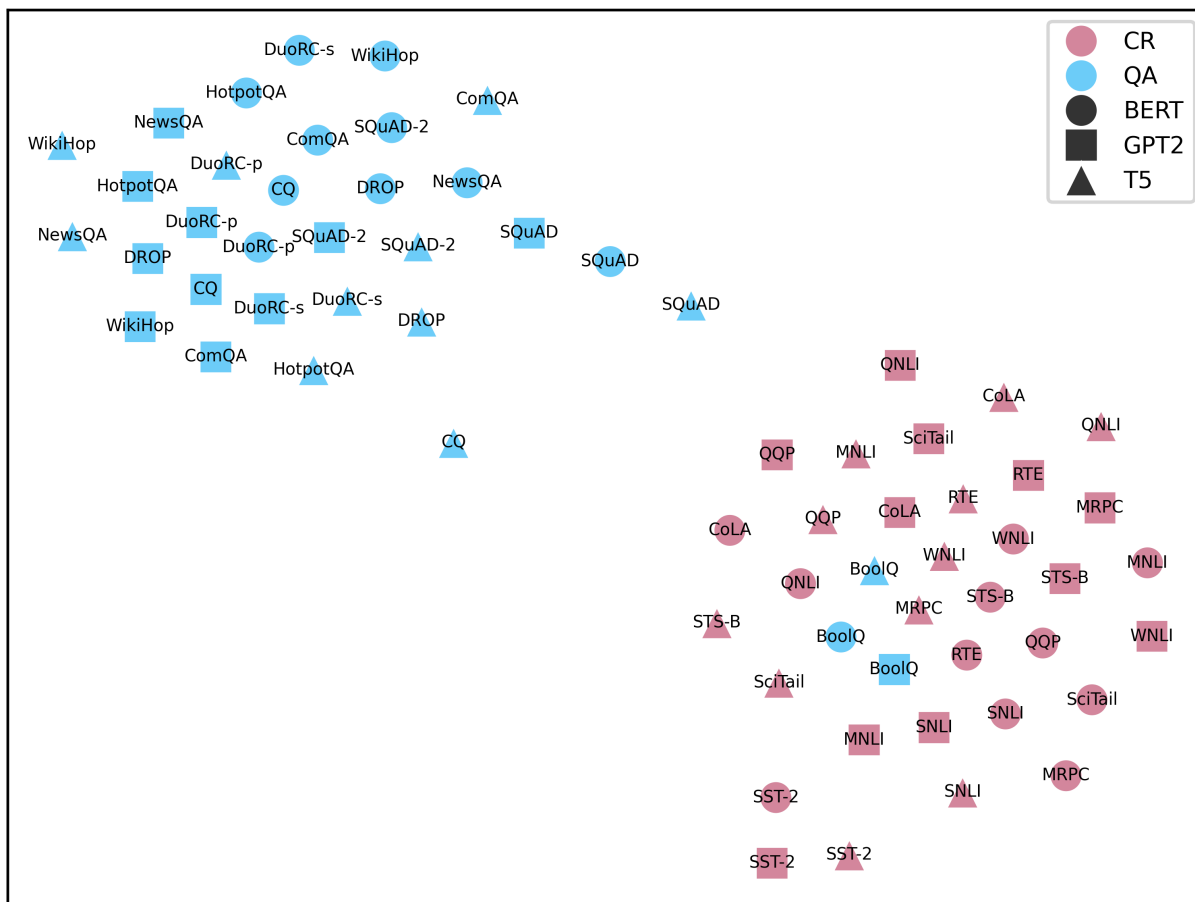
Figure A1: T-SNE visualization of language model MTEs using FUTE with TuPaTE. Different task categories are represented by various colors, while different models or datasets are indicated by distinct shapes.

No.	Prompt
1	Does the movie review below make someone want to watch it? {text}
2	{text} How does the viewer feel about the movie?
3	{text} Did the reviewer enjoy the movie?
4	Does it seem like the reviewer who wrote this review liked the movie? {text}
5	Is the movie review below positive? {text}
6	{text} Did the reviewer find this movie good or bad?
7	{text} Is this review positive or negative?
8	{text} This is definitely not a positive or negative review
9	{text} Did the reviewer enjoy the movie?
10	{text} Did the reviewer find this movie good or bad?
11	Does the movie review below make someone want to watch it? {text} Reason step by step before answering.
12	{text} How does the viewer feel about the movie? Reason step by step before answering.
13	{text} Did the reviewer enjoy the movie? Reason step by step before answering.

Table A1: Prompts used for SA

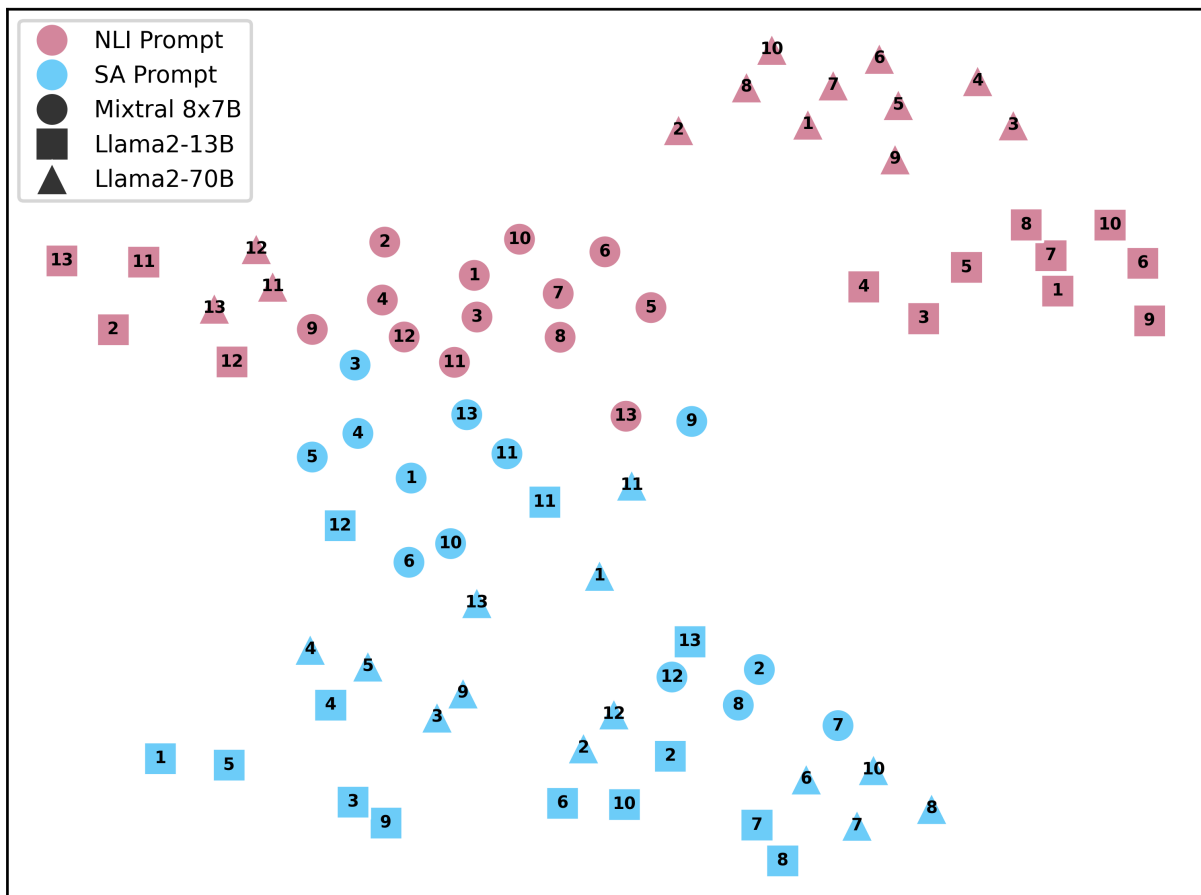| No.	Prompt
1	Given that {premise} Does it follow that hypothesis yes, no, or maybe?
2	{premise} Using only the above description and what you know about the world, "{hypothesis}" is definitely correct, incorrect, or uncertain?
3	{premise} Question: {hypothesis} true, false, or maybe?
4	Take the following as truth: premise Then the following statement: ¨hypothesisïs true, false, or maybe?
5	{premise} Question: Does this imply that "{hypothesis}"? Yes, no, or maybe?
6	Given {premise} Should we assume that "{hypothesis}" is true? Yes, no, or maybe?
7	{premise} Based on the previous passage, is it true that "{hypothesis}"? Yes, no, or maybe?
8	{premise} Are we justified in saying that "{hypothesis}"? yes, no, or maybe?
9	Suppose it's true that {premise} Then, is "{hypothesis}" always, sometimes, or never true?
10	Given {premise} Is it guaranteed true that "{hypothesis}"? yes, no, or maybe?
11	Given that {premise} Does it follow that hypothesis yes, no, or maybe? Reason step by step before answering.
12	{premise} Using only the above description and what you know about the world, "{hypothesis}" is definitely correct, incorrect, or uncertain? Reason step by step before answering.
13	{premise} Question: {hypothesis} true, false, or maybe? Reason step by step before answering.

Table A2: Prompts used for NLI

Figure A2: T-SNE visualization of LLMs MTE using FUTE with TuPaTE. Prompts in different task categories are represented by colors, while different LLMs are indicated by distinct shapes. The numbers correspond to the prompt number as listed in Table A1 and Table A2.