Two Heads are Better than One: Distilling Large Language Model Features Into Small Models with Feature Decomposition and Mixture

Tianhao Fu,[*], Xinxin Xu,[*]Xuanmeng Zhang, Weichen Xu, Ruilong Ren, Bowen Deng, Xinyu Zhao, Jian Cao, Xixin Cao

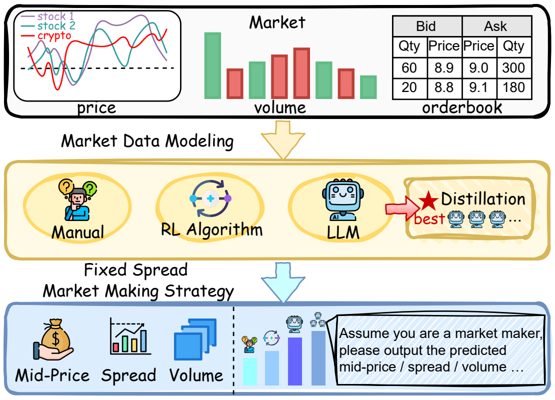
Peking University

Abstract

Market making (MM) through Reinforcement Learning (RL) has attracted significant attention in financial trading. With the development of Large Language Models (LLMs), more and more attempts are being made to apply LLMs to financial areas. A simple, direct application of LLM as an agent shows significant performance. Such methods are hindered by their slow inference speed, while most of the current research has not studied LLM distillation for this specific task. To address this, we first propose the normalized fluorescent probe to study the mechanism of the LLM's feature. Based on the observation found by our investigation, we propose Cooperative Market Making(CMM), a novel framework that decouples LLM features across three orthogonal dimensions: layer, task, and data. Various student models collaboratively learn simple LLM features along with different dimensions, with each model responsible for a distinct feature to achieve knowledge distillation. Furthermore, CMM introduces an Hájek-MoE to integrate the output of the student models by investigating the contribution of different models in a kernel function-generated common feature space. Extensive experimental results on four real-world market datasets demonstrate the superiority of CMM over the current distillation method and RL-based market-making strategies. The code is provided in the supplementary material.

Introduction

Market making is a core task in today's financial area, which could provide liquidity to each financial asset (Guéant, Lehalle, and Fernandez-Tapia 2013). Today, algorithmic systems handle more than 60% of the trading volume in active markets (Othman 2012). Recent breakthroughs in LLM have demonstrated exceptional potential in financial data analysis (Brown et al. 2020; Radford et al. 2019). We conducted an exploratory experiment that shows that Gemini-2.5-Pro (Team et al. 2023) and Llama-3.1 (Grattafiori et al. 2024) perform well in all indicators, surpassing traditional RL methods, as depicted in Figure 1. More details are available in the supplementary material. However, LLM inference speeds cannot meet the demands of real-time trading, where subsecond latency is essential (Vaswani 2017; Kaplan et al. 2020). Knowledge distillation (KD) (Hinton 2015; Gou et al. 2021; Xu et al. 2024) is a technique used to mitigate latency issues. However, existing approaches focus on distilling knowledge from large LLMs to small LLMs, these small LLMs still exhibit slower inference speeds compared to traditional lightweight models (Jiao et al. 2019). There are no cross-architecture distillation methods.



Bid		Ask	
Qty	Price	Price	Qty
60	8.9	9.0	300
20	8.8	9.1	180

Assume you are a market maker, please output the predicted mid-price / spread / volume …

Figure 1: Overview of the market-making workflow. A standard market-making algorithm analyzes historical market data and outputs future ordering strategies. We do a simple experiment that utilizes an LLM prompted with input to directly predict the future mid-price, spread, and volume, which is used to construct future orders via classic price and volume arithmetic sequences. We find that the LLM-based approach surpasses the performance of traditional RL algorithms. Furthermore, with our proposed distilled method, the small model demonstrates further significant improvements that could be used in a real-time scenario.

We then conduct an exploratory experiment that directly distills the LLM feature to a small model using traditional distillation methods, resulting in poor performance. Detailed results are provided in the Supplementary Materials. This failure results in the single small model lacking the representational capacity to capture an LLM's deep and high-dimensional complex features. Therefore, we try to decompose complex LLM features into simpler components and assign them to an ensemble of small learning models to im-

[*] These authors contributed equally.

prove the effectiveness of small models in learning LLM features. We believe that layers, tasks, and input data types, such as these three variables, have the potential to decouple complex features. To further validate this, we propose a Normalized Fluorescent Probe to analyze the complex feature representations of LLMs (Yu and Ananiadou 2024). Our analysis shows that with stronger decoupling conditions of these dimensions, the LLM feature exhibits more apparent separation between clusters. Furthermore, a specialization across the model depth is observed: the shallow layers prioritize the prediction of mid-price, the middle layers focus on the spread, and the deep layers are geared towards the total volume(Bouchaud, Farmer, and Lillo 2009; Kyle 1985). Building on these insights, we introduce **Cooperative Market Making (CMM)**, a two-stage framework which contains: (1) **Orthogonal Feature Decomposition Distillation (OFDD)**: We decouple LLM features into three dimensions: layer feature hierarchy, task objectives, and data type/market regime to decompose complex feature spaces into specialized clusters (LeCun, Bengio, and Hinton 2015; Schmidhuber 2015). Each cluster is distilled into dedicated lightweight models (Ba and Caruana 2014). (2) **Hájek Projection-based Mixture-of-Experts (Hájek-MoE)**: We design a projection mechanism to quantify the contribution of each lightweight expert model.(Hájek 1968; Wager and Athey 2018)

In summary, the contributions of this paper are as follows:

- We propose a Normalized Fluorescent Probe to perform a mechanistic analysis of LLM features and reveal two critical LLM features: (1) Features of different layers govern different outputs of the tasks. (2) Features within the same layer exhibit significant discrepancies when processing heterogeneous input data.
- We propose Orthogonal Feature Decomposition Distillation (OFDD) to decompose LLM features along with three complementary variables: (1) layer hierarchy, (2) task specialization, and (3) data market regime. This decomposition simplifies the LLM features and enables small models to learn from the LLM more effectively.
- We propose the Hájek projection-based Mixture-of-Experts (Hájek-MoE) to integrate the outputs of different small models. Hájek-MoE quantifies the contribution of each model, considering the projection length of each model vector in the same feature space generated by a kernel function.
- We demonstrate the superiority of our approach through extensive experiments conducted in challenging market environments. In contrast to LLM-based methods, our approach achieves higher accuracy and reduced computational cost. Compared to RL-based methods, it exhibits a much greater sample efficiency.

Related Works

Market Making

Market making involves the continuous submission of buy and sell orders in the limit order book (Gould et al. 2013) to maximize returns while managing risk (Vicente, Fernández, and García 2023; Amihud and Mendelson 1980). Traditional MM frameworks, including those introduced by (Avellaneda and Stoikov 2008; Guéant 2017), are based on mathematical models that often assume static market conditions. In recent years, RL has gained traction as a flexible and adaptive approach for designing MM strategies that can respond to real-time market fluctuations (Chan and Shelton 2001; Cartea, Jaimungal, and Penalva 2015; Patel 2018; Zhong, Bergstrom, and Ward 2021; Fang et al. 2021; Niu, Li, and Li 2022; Sun et al. 2023), but a significant portion of this research has focused on the refinement of strategies for a single price level (Sadighian 2019). Many RL-based approaches use the mid-price as a dynamic pricing reference, while this reliance on a volatile signal can lead to excessive order cancellations (Kumar 2020).

LLM Distillation

Recent LLMs, such as PaLM 540b (Chowdhery et al. 2023), pose significant challenges to both inference and fine-tuning due to their substantial computational demands. These dependencies highlight the importance of knowledge distillation. Furthermore, the Chain-of-Thought (Wei et al. 2022) framework has enabled the generation of rich reasoning outputs from teacher models (Ho, Schmid, and Yun 2022), allowing student models to learn not only the answers but also the reasoning processes. This approach enhances the student model through multitask learning (Chu et al. 2023). In addition, efforts have been made to generate various rationales to improve the consistency of predictions (Chen et al. 2024). Although these systems are designed to leverage knowledge diversity, they remain underexplored.

Mixture of Expert

Mixture of Experts (MoE) was introduced by (Jordan and Jacobs 1994), with distinct experts handling different input sections. It was extended by deep MoE and conditional computation (Bertsekas, Tsitsiklis, and Athans 2005). (Shazeer et al. 2017) integrated MoE with LSTMs (Hochreiter 1997), and Switch Transformers (Fedus, Zoph, and Shazeer 2022) combined MoE with transformers, enabling MoE in architectures like MoE-LLaVA (Lin et al. 2024). The core principle of MoE is to expand the capacity of the model efficiently (Dou et al. 2019). Unlike previous MoE techniques (Cai et al. 2024; Yuksel, Wilson, and Gader 2012; Vats et al. 2024) where specialization emerges during training, MoAI (Lee et al. 2024) explicitly defines experts for specific input segments. Recent advances enable training trillion-parameter models in natural language processing (Li, Thomas, and Liu 2021) and computer vision (Ochs et al. 2015).

Method

In this section, we describe the proposed normalized fluorescent probe, the orthogonal feature decomposition distillation, and Hájek-MoE. Figure 2 is an overview of our whole framework. Firstly, we decouple LLM features across three orthogonal dimensions: layer, task, and data. Various student models collaboratively learn complex LLM features along

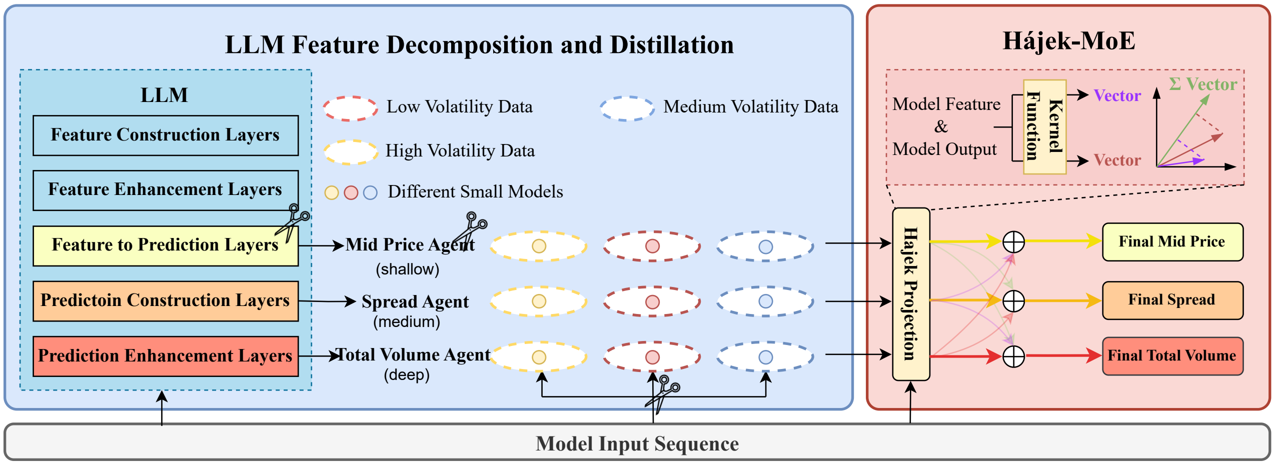


Figure 2: Overview of the CMM Framework. Left: LLM Feature Decomposition and Distillation. The complex feature space of an LLM is decomposed across three dimensions: layer, task, and data. Such three variables result in various types of features, where each feature type is learned by a specialized small model, thereby effectively representing the comprehensive LLM feature space through a collection of smaller models. Right: Inference with Hájek-MoE. Hájek-MoE employs a kernel function to project the output and feature of each small model into a shared feature space to obtain each model's confidence score. The final prediction is computed by aggregating each model's output with the scores.

these dimensions, and each model is responsible for a specific type of feature. Next, through H'ajek-MoE, we integrate the output of the student models by investigating the contribution of the projection of different models along different input data vectors.

Normalized Fluorescent Probe

To study the LLM feature, we propose a normalized fluorescent probe that could accurately identify and quantify the influence of any position feature on the model's outputs. The normalized fluorescent probe follows a traditional LLM mechanistic interpretation approach that leverages noise perturbation techniques to assess the sensitivity and robustness of each feature and analyze the resultant changes in outputs. However, the traditional approach is easily affected by noise distributions, amplitudes, and output variation metrics. For example, when we study which feature has a greater impact on the specific task output, the results observed are different when adding Gaussian-distributed noise and adding uniformly distributed noise. Therefore, we proposed a noise-normalized probe that can ensure the robustness and comprehensiveness of our probe by doing multiple experiments with different noise distributions and averaging the results.

We compare perturbation effects within each module to identify its most influenced outputs. We ensure the robustness of the results by integrating different types of noise. The implementation steps of the normalized fluorescent probe are as Algorithm 1. As shown in lines 4 to 13, we first generate Gaussian and uniformly distributed noise and randomly sample each noise within a given amplitude range $[a_{min}, a_{max}]$ to construct noise samples with different amplitudes and distributions. Subsequently, we inject these noise samples into the parameters θ of the module one by one, as can be seen in line 14. By measuring and recording the output change values Δ_o after the changes of the corresponding parameters, we can evaluate the impact of the module on the output. As illustrated in lines 16 to 19, in order to eliminate the interference of noise amplitude differences and examine the effects of different noise forms, after the changes of the corresponding parameters, we can evaluate the impact of the module on the output. As illustrated in lines 16 to 19, in order to eliminate the interference of noise amplitude differences and examine the effects of different noise forms, we normalize the noise amplitude of each output Δ_o to obtain Δ_{norm} and calculate its average effect Avg_o under different noise forms. Then we get an M×O matrix S, where the row index represents different modules, the column index represents a specific output, and each element in the matrix reflects the contribution or influence of the corresponding module on the output. The Causal Attribution Map C is obtained by taking the maximum value of each column in the S matrix, which reveals the module with the greatest impact on each output.

After applying noise perturbations and computing the influence scores for each pair (module, output), we then quantify the impact of each module on each output. Based on the normalized fluorescent probe, we find that in market making, shallow LLM features are most useful for predicting mid-price, middle features for spread, and deep features for total volume. We also find that even in the same layer, representations exhibit significant discrepancies when processing heterogeneous input data. For example, feature clustering is more pronounced when visualizing data with low volatility. Figure 3 shows the visualization results.

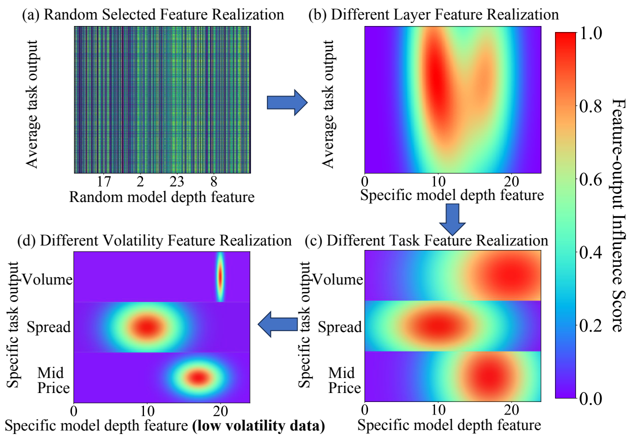


Figure 3: Progressive feature decomposition visualization by our probe results. With stronger decoupling conditions, the LLM features exhibit clearer separation between clusters. Furthermore, a specialization across model depth is observed: shallow layers prioritize mid-price prediction, middle layers focus on the spread, and deep layers are geared towards total volume.

Algorithm 1: Normalized Fluorescent Probe

```
Input: Model parameters $\theta$, feature set $F$, output variables $O$, noise types $\{\mathcal{N}, \mathcal{U}\}$, amplitude ranges $A$.
Output: Causal Attribution Map $C$ (each element of $C$ is the index of the corresponding most influential module).
for each module $m \in M$ do
    $\theta_{original} \leftarrow \theta$ {Preserve original parameters}
    1. Normalize over Noise Distributions
    for each distribution $d \in \{\mathcal{N}, \mathcal{U}\}$ do
        if $d =$ Gaussian then
            $\epsilon \sim \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$ $\{\mu = 0, \sigma = 1\}$
        else if $d =$ Uniform then
            $\epsilon \sim \frac{1}{b-a} - 1 \leq x \leq 1$
        end if
    end for
    2. Normalize over Output Amplitude
    for each amplitude range $[a_{min}, a_{max}] \in A$ do
        $A_{scale} \leftarrow$ random$(a_{min}, a_{max})$
        $\theta_{perturbed} \leftarrow \theta + A_{scale} \cdot \epsilon$
        for each output $o \in O$ do
            $\Delta_o = |f(\theta_{perturbed})_o - f(\theta_{original})_o|$
            $\Delta_{norm} \leftarrow \frac{\Delta_o - a_{min}}{a_{max} - a_{min}}$
        end for
        $S_{m,o} \leftarrow \frac{1}{|A| \cdot N} \sum_{i=1}^{N} \Delta_{norm}^{(i)}$
    end for
end for
$C_o \leftarrow \arg\max_{m \in M} S_{m,o}$ for each output $o \in O$ {Find the module with max influence for each output}
return Causal Attribution Map $C$
```

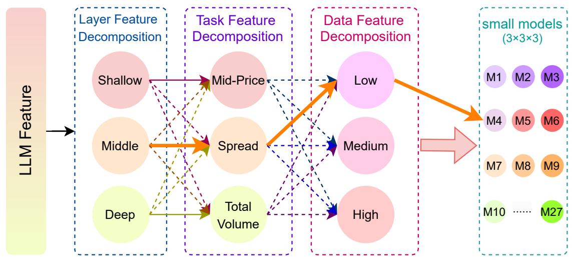


Figure 4: Tree diagram of Orthogonal Feature Decomposition Distillation. By varying three decoupling variables, complex LLM features are decomposed into simpler components and distilled into specialized small models.

Orthogonal Feature Decomposition Distillation

As mentioned above, our normalized fluorescent probe analysis reveals that LLM features can be classified along three dimensions. Therefore, we reduce the learning difficulty for small models by decomposing the complex features along these dimensions, and each class of features is distilled with a specific small model.

Base Distillation. We perform base distillation by distilling the features during the training process. However, our initial experiments did not show good results. This is because the small model's architecture is too simple to capture the complexity of the LLM's advanced features. Since we have shown above that the features of different modules of LLM mainly affect different outputs, we use various small models to learn these different features. In view of this, we propose a strategy to decouple the complex features of LLM from three dimensions, and thereby improve the feature representation capabilities of small models. Specifically, we decompose LLM features into three independent types, each of which is learned by a dedicated small model. Notably, because of the observation that a high degree of correlation exists between layer-wise and task-wise decompositions, for the layer and task variables, the corresponding pair distilled small model has a large weight when doing MoE. Figure 4 shows the specific decomposition process. In this way, the features learned by multiple small models can effectively approximate the overall features of the LLM, significantly reducing the difficulty of training small models.

Layer Feature Decomposition. Previous work has shown that the different layers of LLM are characterized at different levels, from shallow to deep (Yu and Ananiadou 2024). Our normalized fluorescent probe also identified the dividing lines between different layers. Based on this intuition, we make different models learn different parts of the feature during Distillation, which can effectively reduce the difficulty of feature learning. In this way, we initially decompose the features based on the layer. The last three levels that we have discovered with the probe are the output of the separate controls, mid-price, spread, and total volume.

Task Feature Decomposition. In addition to decomposing the structure of the LLM, we can further enhance the distillation performance of the small model by decomposing our task. Our specific task involves determining the mid-price of market making orderbook, the spread of market making orderbook, and the total volume of market making orderbook.

Referring to Figure 3(c), we found that different LLM layers specialize in predicting different outputs. As mentioned in Layer Feature Decomposition, we identified that the LLM Feature to Prediction Layers (Shallow Layer), LLM Prediction Construction Layers (Middle Layer), and LLM Prediction Enhancement Layer (Deep Layer) exhibit the strongest correlations with the outputs mid-price, spread, and total volume, respectively. Leveraging these insights, the lightweight model, which aims to learn shallow features, will additionally learn the LLM mid-price output logits through both feature distillation and logit distillation. In this way, we further decompose the LLM features based on the task.

Data(Market Type) Feature Decomposition. From the normalized fluorescent probe, we recognized that the features of the same layer exhibit significant discrepancies when processing input data with different market volatility, as demonstrated in Figure 3(d). Therefore, based on the input data, we calculate the historical 5-day volatility of each data sample and then categorize the input data into three types by dividing the volatility into three bands. Thus, we can further decompose the features according to the kind of data. Training a market maker model on low, medium, or high volatility data results in conservative, neutral, and aggressive strategies, respectively. In this way, we further decompose the LLM features based on the type of market.

To further improve the performance of small models, we also experimented with different architectures that learn the same feature type to identify the optimal ones. A detailed analysis is provided in the Supplementary Materials.

Hájek Projection-based Mixture-of-Experts

After the OFDD, each small model could and only could capture a specific type of feature of the original LLM, resulting in poor performance when processing other types of data features. Therefore, we urgently need to find an effective way to combine these highly specialized small models. When choosing a model fusion strategy, we did not adopt the conventional MoE framework. Because our situation is unique: The small models that we built have unique architectures and inherent correlations, while they inherit specific knowledge at different levels from the same LLM.

Drawing inspiration from the Hájek projection (Hájek 1968; Wager and Athey 2018), which utilizes a kernel function to map the different spaces into the same space, to manage and leverage various small models effectively, we employ a framework based on the mixture of experts (Hájek-MoE) based on the Hájek projection. Our approach similarly maps the feature spaces of different models into a unified representation space through a kernel function that accepts each model's features and prediction and outputs each model vector's coordinates in the mapped kernel space.

In practice, the Hájek Projection is realized through a concrete strategy of dimensionality reduction and geometric projection to compute the confidence score of each expert. The key steps are as follows:

First, we define a kernel function ϕ, which maps high-dimensional expert features and predictions into a 2D vector space. In practice, ϕ is implemented as a shallow neural network.

Next, we compute the consensus feature by computing the average feature and prediction across all N experts, denoted as $\bar{E}(X)$:

$$\bar{E}(X) = \frac{1}{N}\sum_{j=1}^{N} E_j(X) \tag{1}$$

This average output is then mapped to a 2D vector using the function ϕ, which we call the consensus vector V_{avg}. This vector serves as the reference axis for our projection.

$$V_{\text{avg}} = \phi(\bar{E}(X)) \tag{2}$$

Similarly, for each expert E_i, its output $E_i(X)$ is mapped to a 2D feature vector V_i using the same function ϕ.

The Hájek confidence score C_i for each expert E_i is defined as the scalar projection of its feature vector V_i onto the consensus vector V_{avg}. This value quantifies the degree of alignment between the expert's contribution and the collective consensus. The confidence score is computed as:

$$C_i = \frac{V_i \cdot V_{\text{avg}}}{\|V_{\text{avg}}\|} \tag{3}$$

where $V_i \cdot V_{\text{avg}}$ is the dot product of the two vectors, and $\|V_{\text{avg}}\|$ is the Euclidean norm of the consensus vector. This ensures that experts whose outputs are more aligned with the group consensus receive a higher Hájek confidence score.

Experiments

Experimental Setup

Dataset and Setting. The data we use are historical orders and trades on the Shanghai Futures Exchange from July 2021 to July 2022, covering 186 trading days follows IMM (Niu et al. 2023). We select four contracts as 4 datasets: *FU*, *RB*, *CU*, and *AG* and reconstruct their historical 5-depth limit order books with a 500-millisecond real-time financial period. Our model takes the order book of the previous timestep as input. It directly predicts the mid-price, spread, and volume for the subsequent timestep, which are then used to construct the new order book via a traditional trapezoidal algorithm. We follow the IMM (Niu et al. 2023) to do the dataset split to train and test. We use LLaMA3.1 as our LLM. We use MLP with 2 layers as hajak kernel function. All experiments are conducted on 64 NVIDIA H100 GPUs.

Compared Methods and Metrics. We evaluate the effect of our method on multiple rule-based KD approaches, including FOIC (Gašperov and Kostanjčar 2021), LIIC (Gašperov and Kostanjčar 2021), LIIIC (Niu et al. 2023). We also compare with various RL-based MM methods including RL_{DS} (Beysolow II and Beysolow II 2019), DRL_{OS}

	RB			FU			CU			AG		
	EPnL [10^3] **↑**	**MAP [unit] ↓**	**PnLMAP ↑**	**EPnL** [10^3] **↑**	**MAP [unit] ↓**	**PnLMAP ↑**	**EPnL** [10^3] **↑**	**MAP [unit] ↓**	**PnLMAP ↑**	**EPnL** [10^3] **↑**	**MAP [unit] ↓**	**PnLMAP ↑**
FOIC	3.23 ± 4.35	255 ± 111	14 ± 22	-7.79 ± 9.25	238 ± 135	-43 ± 56	-33.05 ± 27.63	206 ± 141	-161 ± 224	-48.39 ± 28.83	189 ± 154	-250 ± 335
LIIC	2.26 ± 3.32	123 ± 32	20 ± 29	-6.89 ± 6.66	115 ± 30	-66 ± 69	-24.19 ± 14.83	150 ± 20	-164 ± 513	-38.9 ± 26.2	142 ± 45	-302 ± 243
LTIIIC	9.16 ± 4.87	65 ± 6	139 ± 68	8.26 ± 2.64	52 ± 3	160 ± 50	-16.74 ± 15.81	112 ± 109	-190 ± 203	-32.57 ± 22.8	128 ± 22	-264 ± 166
RL_{SD}	4.36 ± 1.64	38 ± 4	114 ± 38	7.31 ± 5.38	76 ± 29	90 ± 46	-19.7 ± 17.2	214 ± 109	-92 ± 298	-25.43 ± 23.83	107 ± 37	-237 ± 235
DRL_{OS}	8.22 ± 3.70	51 ± 4	156 ± 61	11.03 ± 13.87	37 ± 3	30 ± 36	-18.9 ± 18.02	647 ± 2367	-99 ± 147	-28.39 ± 27.92	169 ± 154	-167 ± 135
IMM	16.46 ± 9.1	96 ± 13	165 ± 74	28.1 ± 10.27	102 ± 14	274 ± 89	-4.86 ± 10.17	111 ± 28	-43 ± 87	-14.5 ± 20.2	102 ± 14	-274 ± 89
CMM	**22.69 ± 1.96**	**34 ± 3**	**179 ± 14**	**31.39 ± 4.18**	**32 ±2**	**298 ± 19**	**-1.63 ± 0.41**	**30 ± 8**	**-16 ± 17**	**-8.95 ± 11.22**	**64 ± 2**	**-143 ± 102**

Table 1: Overall Results.

(Chung et al. 2022), IMM (Niu et al. 2023) and CMM. The following performance and risk metrics are employed: 1.Episodic PnL is a natural choice to evaluate the profitability of a MM agent (Sutton 2018). 2.Mean Absolute Position (MAP) accounts for the inventory risk (Gârleanu and Pedersen 2013). 3.Return Per Trade (RPT) evaluates the agent's capability of capturing the spread (Easley, De Prado, and O'Hara 2011). It is normalized across different markets by the average market spread. 4. PnLMAP defined as PnL divided by the mean absolute position (MAP) in this period (Hull and Basu 2016). It means the PnL in per unit of inventory and can measure the ability of the agent to profit against inventory risk.

Overall Results

The experimental results in Table 1 demonstrate the superior performance of CMM in all 4 futures contracts. For instance, on the RB dataset, CMM not only achieves significantly higher profitability than the strongest baseline IMM, but also substantially reduces inventory risk. This was accomplished through layer-task aligned distillation guided by the normalized fluorescent probe. The FU contract results further validate the framework's robustness, where our method attains the highest PnLMAP of 298, a significant improvement over IMM. This superior performance is achieved by effectively integrating volatility-conditioned specialization, thereby demonstrating stable profitability across varying market regimes. Notably, in challenging low-liquidity markets such as CU and AG contracts, CMM significantly outperforms the benchmarks in terms of the terminal wealth, risk-adjusted return, and spread-capturing ability. This performance stems from Hájek-MoE's adaptive fusion mechanism, which dynamically obtains the confidence scores of experts based on the real-time market.

Model Component Ablation Study

Table 2 demonstrates the progressive impact of our decomposition components. SR refers to the Sharpe Ratio. The layer-wise decomposition baseline, FD(layer), delivers the poorest performance across all metrics. This result indicates that only isolating hierarchical LLM features is insufficient. An improvement in profitability is achieved by introducing task decomposition with specialized prediction heads. This enhancement arrives from the model's newfound ability to develop distinct, focused strategies for different predictive sub-tasks, moving beyond a monolithic prediction approach and thus capturing market opportunities more effectively. Subsequently, volatility-type decomposition improves risk control through market-regime adaptation, and the model successfully mitigates potential losses, as reflected by a marked reduction in adverse outcomes. The complete framework, FD(Hájek-MoE), achieves optimal balance between profitability and risk-adjusted returns, which demonstrates the superiority of its input-adaptive fusion mechanism. Unlike static aggregation, our approach dynamically allocates confidence scores to the experts in real-time, achieving a robust fusion of all decomposed components.

Models	**EPnL** [10^3] **↑**	**MAP [unit] ↓**	**PnLMAP ↑**	**SR ↑**
$\text{CMM}_{\text{FD(layer)}}$	12.54 ± 3.64	67 ± 5	162 ± 46	1.98
$\text{CMM}_{\text{FD(task)}}$	17.25 ± 9.72	58 ± 47	195 ± 56	2.33
$\text{CMM}_{\text{FD(type)}}$	22.41 ± 5.11	45 ± 5	241 ± 57	2.68
$\text{CMM}_{\text{FD(hájek-moe)}}$	**31.39 ± 4.18**	**32 ± 2**	**298 ± 19**	**2.98**

Table 2: Comparison results of ablation study on FU dataset.

Comparison of Different Distillation Methods and MoE Methods

As shown in Table 3, our framework achieves a higher EPnL of 31.39, an improvement of 3.02% compared to the original LLM, while also demonstrating 6.3× lower latency at just 0.3s. This validates the effectiveness of our orthogonal decomposition strategy in preserving critical features identified by the normalized fluorescent probe. Conventional KD methods such as ReviewKD and CAT-KD suffer severe performance degradation due to entangled feature learning, while standard MoE methods like X-MoE and MH-MoE exhibit poor risk control from static expert fusion. CMM's superiority stems from its three-dimensional decomposition.

Models	**EPnL** [10^3] **↑**	**MAP [unit] ↓**	**PnLMAP ↑**	**SR ↑**	**Latency(s) ↓**
LLM-Base	30.47 ± 5.52	64 ± 13	283 ± 66	3.01 ± 0.3	1.9
ReviewKD	10.52 ± 3.21	2230 ± 112	48 ± 9	2.00 ± 0.0	0.22
Sim-KD	15.34 ± 4.15	1895 ± 85	85 ± 14	2.25 ± 0.1	0.22
CAT-KD	20.81 ± 5.03	1570 ± 143	132 ± 18	2.48 ± 0.2	0.22
X-MoE	19.63 ± 6.17	1180 ± 205	195 ± 23	1.67 ± 0.3	0.46
MH-MoE	20.92 ± 3.52	785 ± 315	155 ± 28	1.89 ± 0.1	0.57
CMM	**31.39 ± 4.18**	**32 ± 2**	**298 ± 19**	**2.98**	**0.3**

Table 3: Comparison of Different Distillation Methods and MoE Methods on FU.

Feature Decomposition Analysis

Figure 5 presents a 2D visualization of the feature space, obtained via PCA, to demonstrate the effectiveness of orthog-

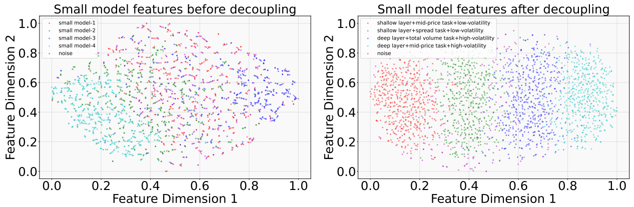


Figure 5: Feature Decomposition Analysis

onal decomposition. Each point in the figure corresponds to the feature representation from a small model. The pre-decomposition features are chaotically overlapped. In contrast, the post-decomposition features reveal a highly organized structure. This structure is visually articulated through color, where models sharing the same volatility, task, and layer depth are assigned the same color and visibly cluster together. For example, all models for the spread task in a low-volatility regime using a shallow layer appear as a distinct green cluster, while models for the total volume task in a high-volatility regime with a deep layer form a blue cluster.

Robustness Test under Extreme Market Conditions

To evaluate the robustness of CMM under extreme market, we conducted experiments using historical data augmented with simulated flash crash and sudden reversal scenarios.

The results in Table 4 show that CMM achieves a strong balance of profitability and risk mitigation under duress, where baseline models proved vulnerable. This resulted in a healthier risk-adjusted return. These results demonstrate that CMM is more robust under extreme market, likely due to its dynamic expert confidence score mechanism, which prioritizes risk control experts during market turbulence.

Models	EPnL [10^3] ↑	MAP (unit) ↓	PnLMAP ↑	SR ↑
LLM-Base	6.50 ± 2.45	150 ± 30	0.43 ± 0.08	0.60 ± 0.1
IMM	7.80 ± 3.05	120 ± 25	0.65 ± 0.12	0.80 ± 0.1
CMM	**10.50 ± 2.10**	**45 ± 8**	**2.33 ± 0.46**	**1.20 ± 0.2**

Table 4: Robustness Test Results under Extreme Market.

Long-Term Market Adaptability Experiment

To assess the long-term adaptability of CMM across different market regimes, we ran the model over a 1-week period covering various market conditions, including bull, bear, and sideways markets. The results in Table 5 show that CMM achieves superior performance compared to LLM-Base and IMM in terms of profitability, risk control, and computational efficiency across different market regimes. These results indicate that CMM is better at adapting to changing market, likely due to its orthogonal decomposition strategy, which allows CMM to specialize in different market types.

Models	EPnL [10^3] ↑	MAP (unit) ↓	PnLMAP ↑	SR ↑
LLM-Base	12.00 ± 1.50	80 ± 15	0.56 ± 0.12	1.80 ± 0.20
IMM	10.00 ± 1.00	60 ± 10	0.45 ± 0.10	1.50 ± 0.10
CMM	**14.00 ± 1.20**	**30 ± 3**	**0.80 ± 0.05**	**2.20 ± 0.15**

Table 5: Long-Term Market Adaptability Results.

Performance Test under Low-Data Conditions

To evaluate the performance of CMM when trained with limited data, we conducted experiments using 10%, 20%, and 50% of the available data. The results in Table 6 show that CMM consistently outperforms the baselines across all data percentages. Even when trained on a minimal fraction of the data, CMM establishes a commanding lead by delivering markedly higher returns while simultaneously maintaining much stricter risk control. This may be because baseline models are prone to overfitting or failing to discern clear patterns from limited information. CMM's architecture allows it to learn more generalizable market behaviors. This data efficiency can likely be attributed to its orthogonal decomposition strategy, which simplifies the feature space and reduces the complexity of the learning task.

Data %	Models	EPnL [10^3] ↑	MAP (unit) ↓	PnLMAP ↑
10%	LLM-Base	2.50 ± 0.85	80 ± 15	0.31 ± 0.06
	IMM	2.75 ± 0.95	52 ± 8	0.53 ± 0.10
	CMM	**4.50 ± 1.20**	**35 ± 5**	**1.29 ± 0.26**
20%	LLM-Base	3.20 ± 1.05	72 ± 12	0.44 ± 0.08
	IMM	3.50 ± 1.10	48 ± 7	0.73 ± 0.14
	CMM	**5.20 ± 1.30**	**32 ± 4**	**1.63 ± 0.32**
50%	LLM-Base	4.00 ± 1.20	65 ± 10	0.62 ± 0.12
	IMM	4.30 ± 1.05	40 ± 6	1.08 ± 0.21
	CMM	**6.00 ± 1.40**	**28 ± 3**	**2.14 ± 0.43**

Table 6: Performance under Low-Data Conditions.

Energy Efficiency and Inference Speed Experiment

To compare the energy efficiency and inference speed of CMM with baseline methods, we measured power consumption and inference latency during model deployment. The results in Table 7 show that CMM significantly outperforms the baselines in terms of energy efficiency and inference speed. These results indicate that CMM is more suitable for real-time trading environments, where low latency and energy efficiency are critical.

Models	Power Usage (W) ↓	Inference Latency (ms) ↓
LLM-Base	150 ± 10	120 ± 15
IMM	90 ± 5	80 ± 10
CMM	**45 ± 3**	**20 ± 5**

Table 7: Energy Efficiency and Inference Speed Results.

Conclusion

This paper addresses market-making challenges by proposing CMM, a framework that decouples LLM knowledge through orthogonal decomposition. It starts by using a probe to analyze how different layers of LLMs are related to specific tasks and how features diverge under varying volatility conditions. Then, it introduces OFDD, which transfers LLM knowledge by breaking it down through layers, tasks, and input data types. Finally, it designs Hájek-MoE to dynamically integrate these decomposed experts, adjusting their contributions based on input data. Extensive experiments demonstrate that CMM outperforms other methods.

References

Amihud, Y.; and Mendelson, H. 1980. Dealership market: Market-making with inventory. *Journal of financial economics*, 8(1): 31–53.

Avellaneda, M.; and Stoikov, S. 2008. High-frequency trading in a limit order book. *Quantitative Finance*, 8(3): 217–224.

Ba, J.; and Caruana, R. 2014. Do deep nets really need to be deep? *Advances in neural information processing systems*, 27.

Bertsekas, D. P.; Tsitsiklis, J. N.; and Athans, M. 2005. Convergence theories of distributed iterative processes: A survey. In *Stochastic Programming*, 107–139. Springer.

Beysolow II, T.; and Beysolow II, T. 2019. Market making via reinforcement learning. *Applied Reinforcement Learning with Python: With OpenAI Gym, Tensorflow, and Keras*, 77–94.

Bouchaud, J.-P.; Farmer, J. D.; and Lillo, F. 2009. How markets slowly digest changes in supply and demand. In *Handbook of financial markets: dynamics and evolution*, 57–160. Elsevier.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901.

Cai, W.; Jiang, J.; Wang, F.; Tang, J.; Kim, S.; and Huang, J. 2024. A survey on mixture of experts. *arXiv preprint arXiv:2407.06204*.

Cartea, Á.; Jaimungal, S.; and Penalva, J. 2015. *Algorithmic and high-frequency trading*. Cambridge University Press.

Chan, N. T.; and Shelton, C. 2001. An electronic market-maker.

Chen, Y.; Singh, C.; Liu, X.; Zuo, S.; Yu, B.; He, H.; and Gao, J. 2024. Towards consistent natural-language explanations via explanation-consistency finetuning. *arXiv preprint arXiv:2401.13986*.

Chowdhery, A.; Narang, S.; Devlin, J.; Bosma, M.; Mishra, G.; Roberts, A.; Barham, P.; Chung, H. W.; Sutton, C.; Gehrmann, S.; et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240): 1–113.

Chu, E.-P.; Liu, K.-C.; Hsieh, C.-Y.; Chang, C.-Y.; Tsao, Y.; and Chan, C.-T. 2023. Multi-Task Learning U-Net for Functional Shoulder Sub-Task Segmentation. In *2023 45th Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, 1–5. IEEE.

Chung, G.; Chung, M.; Lee, Y.; and Kim, W. C. 2022. Market Making under Order Stacking Framework: A Deep Reinforcement Learning Approach. In *Proceedings of the Third ACM International Conference on AI in Finance*, 223–231.

Dou, Z.-Y.; Tu, Z.; Wang, X.; Wang, L.; Shi, S.; and Zhang, T. 2019. Dynamic layer aggregation for neural machine translation with routing-by-agreement. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 86–93.

Easley, D.; De Prado, M. M. L.; and O'Hara, M. 2011. The microstructure of the" flash crash": Flow toxicity, liquidity crashes, and the probability of informed trading. *Journal of Portfolio Management*, 37(2): 118.

Fang, Y.; Ren, K.; Liu, W.; Zhou, D.; Zhang, W.; Bian, J.; Yu, Y.; and Liu, T.-Y. 2021. Universal trading for order execution with oracle policy distillation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 107–115.

Fedus, W.; Zoph, B.; and Shazeer, N. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120): 1–39.

Gârleanu, N.; and Pedersen, L. H. 2013. Dynamic trading with predictable returns and transaction costs. *The Journal of Finance*, 68(6): 2309–2340.

Gašperov, B.; and Kostanjčar, Z. 2021. Market making with signals through deep reinforcement learning. *IEEE access*, 9: 61611–61622.

Gou, J.; Yu, B.; Maybank, S. J.; and Tao, D. 2021. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6): 1789–1819.

Gould, M. D.; Porter, M. A.; Williams, S.; McDonald, M.; Fenn, D. J.; and Howison, S. D. 2013. Limit order books. *Quantitative Finance*, 13(11): 1709–1742.

Grattafiori, A.; Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Vaughan, A.; et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Guéant, O. 2017. Optimal market making. *Applied Mathematical Finance*, 24(2): 112–154.

Guéant, O.; Lehalle, C.-A.; and Fernandez-Tapia, J. 2013. Dealing with the inventory risk: a solution to the market making problem. *Mathematics and financial economics*, 7: 477–507.

Hájek, J. 1968. Asymptotic normality of simple linear rank statistics under alternatives. *The Annals of Mathematical Statistics*, 325–346.

Hinton, G. 2015. Distilling the Knowledge in a Neural Network. *arXiv preprint arXiv:1503.02531*.

Ho, N.; Schmid, L.; and Yun, S.-Y. 2022. Large language models are reasoning teachers. *arXiv preprint arXiv:2212.10071*.

Hochreiter, S. 1997. Long Short-term Memory. *Neural Computation MIT-Press*.

Hull, J. C.; and Basu, S. 2016. *Options, futures, and other derivatives*. Pearson Education India.

Jiao, X.; Yin, Y.; Shang, L.; Jiang, X.; Chen, X.; Li, L.; Wang, F.; and Liu, Q. 2019. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*.

Jordan, M. I.; and Jacobs, R. A. 1994. Hierarchical mixtures of experts and the EM algorithm. *Neural computation*, 6(2): 181–214.

Kaplan, J.; McCandlish, S.; Henighan, T.; Brown, T. B.; Chess, B.; Child, R.; Gray, S.; Radford, A.; Wu, J.; and Amodei, D. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Kumar, P. 2020. Deep reinforcement learning for market making. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, 1892–1894.

Kyle, A. S. 1985. Continuous auctions and insider trading. *Econometrica: Journal of the Econometric Society*, 1315–1335.

LeCun, Y.; Bengio, Y.; and Hinton, G. 2015. Deep learning. *nature*, 521(7553): 436–444.

Lee, B.-K.; Park, B.; Won Kim, C.; and Man Ro, Y. 2024. Moai: Mixture of all intelligence for large language and vision models. In *European Conference on Computer Vision*, 273–302. Springer.

Li, Y.; Thomas, M. A.; and Liu, D. 2021. From semantics to pragmatics: where IS can lead in Natural Language Processing (NLP) research. *European Journal of Information Systems*, 30(5): 569–590.

Lin, B.; Tang, Z.; Ye, Y.; Cui, J.; Zhu, B.; Jin, P.; Zhang, J.; Ning, M.; and Yuan, L. 2024. Moe-llava: Mixture of experts for large vision-language models. *arXiv preprint arXiv:2401.15947*.

Niu, H.; Li, S.; and Li, J. 2022. MetaTrader: An reinforcement learning approach integrating diverse policies for portfolio optimization. In *Proceedings of the 31st ACM international conference on information & knowledge management*, 1573–1583.

Niu, H.; Li, S.; Zheng, J.; Lin, Z.; Li, J.; Guo, J.; and An, B. 2023. IMM: An Imitative Reinforcement Learning Approach with Predictive Representation Learning for Automatic Market Making. *arXiv preprint arXiv:2308.08918*.

Ochs, P.; Dosovitskiy, A.; Brox, T.; and Pock, T. 2015. On iteratively reweighted algorithms for nonsmooth nonconvex optimization in computer vision. *SIAM Journal on Imaging Sciences*, 8(1): 331–372.

Othman, A. 2012. *Automated market making: Theory and practice*. Carnegie Mellon University.

Patel, Y. 2018. Optimizing market making using multi-agent reinforcement learning. *arXiv preprint arXiv:1812.10252*.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I.; et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8): 9.

Sadighian, J. 2019. Deep reinforcement learning in cryptocurrency market making. *arXiv preprint arXiv:1911.08647*.

Schmidhuber, J. 2015. On learning to think: Algorithmic information theory for novel combinations of reinforcement learning controllers and recurrent neural world models. *arXiv preprint arXiv:1511.09249*.

Shazeer, N.; Mirhoseini, A.; Maziarz, K.; Davis, A.; Le, Q.; Hinton, G.; and Dean, J. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*.

Sun, S.; Wang, X.; Xue, W.; Lou, X.; and An, B. 2023. Mastering stock markets with efficient mixture of diversified trading experts. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2109–2119.

Sutton, R. S. 2018. Reinforcement learning: An introduction. *A Bradford Book*.

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

Vaswani, A. 2017. Attention is all you need. *Advances in Neural Information Processing Systems*.

Vats, A.; Raja, R.; Jain, V.; and Chadha, A. 2024. The Evolution of Mixture of Experts: A Survey from Basics to Breakthroughs. *Preprints (August 2024)*.

Vicente, Ó. F.; Fernández, F.; and García, J. 2023. Automated market maker inventory management with deep reinforcement learning. *Applied Intelligence*, 53(19): 22249–22266.

Wager, S.; and Athey, S. 2018. Estimation and inference of heterogeneous treatment effects using random forests. *Journal of the American Statistical Association*, 113(523): 1228–1242.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q. V.; Zhou, D.; et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35: 24824–24837.

Xu, X.; Li, M.; Tao, C.; Shen, T.; Cheng, R.; Li, J.; Xu, C.; Tao, D.; and Zhou, T. 2024. A survey on knowledge distillation of large language models. *arXiv preprint arXiv:2402.13116*.

Yu, Z.; and Ananiadou, S. 2024. Interpreting arithmetic mechanism in large language models through comparative neuron analysis. *arXiv preprint arXiv:2409.14144*.

Yuksel, S. E.; Wilson, J. N.; and Gader, P. D. 2012. Twenty years of mixture of experts. *IEEE transactions on neural networks and learning systems*, 23(8): 1177–1193.

Zhong, Y.; Bergstrom, Y. M.; and Ward, A. 2021. Data-driven market-making via model-free learning. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, 4461–4468.

Reproducibility Checklist

1. General Paper Structure

1.1. Includes a conceptual outline and/or pseudocode description of AI methods introduced (yes/partial/no/NA) yes

1.2. Clearly delineates statements that are opinions, hypothesis, and speculation from objective facts and results (yes/no) yes

1.3. Provides well-marked pedagogical references for less-familiar readers to gain background necessary to replicate the paper (yes/no) yes

2. Theoretical Contributions

2.1. Does this paper make theoretical contributions? (yes/no) no

If yes, please address the following points:

2.2. All assumptions and restrictions are stated clearly and formally (yes/partial/no) no

2.3. All novel claims are stated formally (e.g., in theorem statements) (yes/partial/no) no

2.4. Proofs of all novel claims are included (yes/partial/no) no

2.5. Proof sketches or intuitions are given for complex and/or novel results (yes/partial/no) no

2.6. Appropriate citations to theoretical tools used are given (yes/partial/no) no

2.7. All theoretical claims are demonstrated empirically to hold (yes/partial/no/NA) no

2.8. All experimental code used to eliminate or disprove claims is included (yes/no/NA) no

3. Dataset Usage

3.1. Does this paper rely on one or more datasets? (yes/no) yes

If yes, please address the following points:

3.2. A motivation is given for why the experiments are conducted on the selected datasets (yes/partial/no/NA) yes

3.3. All novel datasets introduced in this paper are included in a data appendix (yes/partial/no/NA) yes

3.4. All novel datasets introduced in this paper will be made publicly available upon publication of the paper with a license that allows free usage for research purposes (yes/partial/no/NA) partial

3.5. All datasets drawn from the existing literature (potentially including authors' own previously published work) are accompanied by appropriate citations (yes/no/NA) yes

3.6. All datasets drawn from the existing literature (potentially including authors' own previously published work) are publicly available (yes/partial/no/NA) yes

3.7. All datasets that are not publicly available are described in detail, with explanation why publicly available alternatives are not scientifically satisficing (yes/partial/no/NA) yes

4. Computational Experiments

4.1. Does this paper include computational experiments? (yes/no) yes

If yes, please address the following points:

4.2. This paper states the number and range of values tried per (hyper-) parameter during development of the paper, along with the criterion used for selecting the final parameter setting (yes/partial/no/NA) yes

4.3. Any code required for pre-processing data is included in the appendix (yes/partial/no) yes

4.4. All source code required for conducting and analyzing the experiments is included in a code appendix (yes/partial/no) yes

4.5. All source code required for conducting and analyzing the experiments will be made publicly available upon publication of the paper with a license that allows free usage for research purposes (yes/partial/no) yes

4.6. All source code implementing new methods have comments detailing the implementation, with references to the paper where each step comes from (yes/partial/no) yes

4.7. If an algorithm depends on randomness, then the method used for setting seeds is described in a way sufficient to allow replication of results (yes/partial/no/NA) yes

4.8. This paper specifies the computing infrastructure used for running experiments (hardware and software), including GPU/CPU models; amount of memory; operating system; names and versions of relevant software libraries and frameworks (yes/partial/no) yes

4.9. This paper formally describes evaluation metrics used and explains the motivation for choosing these metrics (yes/partial/no) yes

4.10. This paper states the number of algorithm runs used to compute each reported result (yes/no) yes

4.11. Analysis of experiments goes beyond single-dimensional summaries of performance (e.g., average; median) to include measures of variation, confidence, or other distributional information (yes/no) yes

4.12. The significance of any improvement or decrease in performance is judged using appropriate statistical tests (e.g., Wilcoxon signed-rank) (yes/partial/no) partial

4.13. This paper lists all final (hyper-)parameters used for each model/algorithm in the paper's experiments (yes/partial/no/NA) partial