Pricing Online LLM Services with Data-Calibrated Stackelberg Routing Game

Zhendong Guo, Wenchao Bai, Jiahui Jin*

School of Computer Science and Engineering, Southeast University
{zdguo, wbai, jjin}@seu.edu.cn

Abstract

The proliferation of Large Language Models (LLMs) has established LLM routing as a standard service delivery mechanism, where users select models based on cost, Quality of Service (QoS), among other things. However, optimal pricing in LLM routing platforms requires precise modeling for dynamic service markets, and solving this problem in real time at scale is computationally intractable. In this paper, we propose PriLLM, a novel practical and scalable solution for real-time dynamic pricing in competitive LLM routing. PriLLM models the service market as a Stackelberg game, where providers set prices and users select services based on multiple criteria. To capture real-world market dynamics, we incorporate both objective factors (e.g., cost, QoS) and subjective user preferences into the model. For scalability, we employ a deep aggregation network to learn provider abstraction that preserve user-side equilibrium behavior across pricing strategies. Moreover, PriLLM offers interpretability by explaining its pricing decisions. Empirical evaluation on real-world data shows that PriLLM achieves over 95% of the optimal profit while only requiring less than 5% of the optimal solution's computation time.

Code — https://github.com/luck-seu/PriLLM
Extended version — https://arxiv.org/abs/2511.09062

1 Introduction

The landscape of Large Language Models (LLMs) is rapidly evolving, with service providers continuously introducing new models. This proliferation complicates the task of selecting the optimal model for users (Song et al. 2025; Yue et al. 2025). To address this challenge, LLM routing platforms, such as OpenRouter, Eden AI, and Martian, have been developed. These platforms provide a centralized interface that consolidates key metrics for each model, including per-token cost and Quality of Service (QoS) parameters. Additionally, they offer a unified API, enabling seamless access to multiple models through a standardized interface. *This paper investigates the dynamic service pricing strategies for a service provider within an LLM routing system (Figure 1)*, where user requests are dynamically routed to the most appropriate service provider based on user preferences.

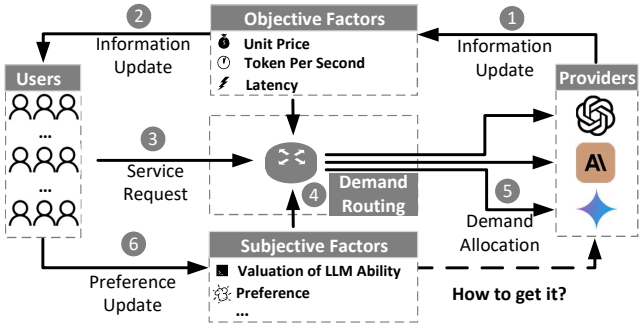


Figure 1: An example of an LLM routing system that dynamically matches user requests to the most suitable providers based on individual preferences. In this example, users receive updates on key metrics of service providers and then adjust their allocation strategies accordingly (Steps 1–4). Their subjective preferences will be updated after providers accept and fulfill the requests (Steps 5–6).

When selecting services, users' routing preferences are shaped by both objective and subjective factors. Objective attributes include per-token cost, model parameter count, and real-time quality-of-service metrics (QoS), namely, access delay (time to first token, TTFT) and service congestion (tokens generated per second, TPS). Subjective attributes encompass the user-perceived values derived from LLMs' practical utilities and their brand reputations (Song et al. 2025; Hu and Zhou 2024; Mizrahi et al. 2024). Users aggregate the factors to minimize individual costs, allowing the routing system to allocate requests effectively across providers. The pricing strategies in LLM service markets are crucial for both providers and users, yet remain an open problem. On one hand, setting lower prices can attract more users, but may degrade QoS due to higher demand. On the other hand, higher prices may increase QoS but reduce market competitiveness. Even worse, the service provider can only obtain partial information about the system without knowing actual users' preferences and LLMs' user-perceived values.

Traditional approaches for cloud service pricing (Saxena and Singh 2024; Chakraborty et al. 2021; Huang et al. 2023; Tütüncüoğlu and Dán 2024; Ding, Gao, and Huang 2023)

*Corresponding author

cannot be directly applied to LLM routing services due to a fundamental misalignment with the market's unique dynamics. Specifically, LLM services exhibit heightened sensitivity to QoS factors such as service congestion and network latency, while user preferences are additionally shaped by user-perceived value. As a counter-intuitive example, a provider's price increase may actually expand its market share if accompanied by substantial performance improvements. Without a calibrated market model that jointly captures these system-level sensitivities and user-side heterogeneity, traditional methods fail to reflect real market dynamics. The Stackelberg game framework naturally models leader-follower dynamics in service markets, where providers act as leaders setting prices and users respond as followers. However, applying this framework to LLM service pricing introduces substantial computational challenges. In particular, the leader's optimization problem depends on the follower-side Nash equilibrium, which is often non-convex, leading to computational intractability. Existing approaches mitigate this complexity through macro-level approximations (Harks and Schedel 2021; Cui, Hu, and Luo 2020; Wu, Barati, and Lim 2020; Fotouhi and Miller-Hooks 2021; Wu et al. 2021; Xiong et al. 2016) or equilibrium relaxations (Li et al. 2019; Naoum-Sawaya and Elhedhli 2011; Böhnlein, Kratsch, and Schaudt 2021; Briest, Hoefer, and Krysta 2008). However, these simplifications sacrifice the modeling precision necessary to capture the nuanced dynamics inherent in LLM routing systems. *Can we calibrate the market model to reflect real-world dynamics by incorporating both objective metrics and subjective preferences? If so, how can we characterize the resulting market equilibrium? Can we design scalable algorithms to handle many users and providers in real-time?* To the best of our knowledge, these questions remain unexplored. An extended version with full technical appendices is available on arXiv.

Contributions & Organization. To answer these questions, we propose PriLLM, the first practical and scalable solution for real-time service pricing in LLM routing.

(1) Problem formulation (Section 3). We formulate the service pricing problem as a Stackelberg game, where providers act as leaders and customers as followers. Given the initial pricing strategy profile $\mathcal{P}$, service selection strategy profile $\mathcal{F}$, user-defined routing functions, and a target provider s, it alternates between updating $\mathcal{F}$ to reach a Nash equilibrium under $\mathcal{P}$, and adjusting s's pricing strategy to maximize its profit. This yields a mathematical program with equilibrium constraints (MPEC), which is usually NP-hard. We prove the existence and uniqueness of the Nash equilibrium.

(2) Game calibration (Section 4). We calibrate our Stackelberg routing model to real LLM routing data by learning user-preference parameters $\boldsymbol{\theta}$. At any given price and market condition, the user side admits a single Nash equilibrium flow, delivered by a predictor function $\mathcal{F}^*(\cdot)$. Fitting the game model to observed traffic allows us to recover latent preferences and align the model with real behaviors.

(3) Game abstraction (Section 5). To reduce complexity in large-scale markets, we aggregate both users and service providers into abstracted groups. On the user side, we group users by the apps they use, based on the assumption that users of the same app share similar preferences. On the provider side, we propose a deep aggregation network that learns to abstract service providers. To prevent overfitting, PriLLM samples multiple pricing strategies for s and minimizes the discrepancy in selection strategies across them.

(4) Experimental study (Section 6). Using real-life market data from OpenRouter, we empirically find the following. (a) PriLLM accurately captures real-world market dynamics parameters, achieving a high R^2 score of 0.8982 in fitting user traffic flows. (b) PriLLM outperforms pMPEC and Smooth, the best baselines, by 9.2% and 14.3% on average, respectively, up to 11.4% and 17.3%. (c) PriLLM achieves over 95% of the optimal profit, requiring less than 5% of the computation time of the brute-force optimal solver.

2 Related Work

2.1 Stackelberg Game in Service Pricing

In traditional cloud and edge computing markets, dynamic pricing has long been modeled as a Stackelberg game, with service providers acting as leaders and users acting as followers. However, these traditional models typically assume that users' utilities depend solely on coarse-grained quality of service (QoS) metrics (Saxena and Singh 2024; Chakraborty et al. 2021; Tütüncüoğlu and Dán 2024; Ding, Gao, and Huang 2023), such as average latency, while ignoring users' subjective values of model quality. In the LLM service domain, user decisions additionally weigh a range of fine-grained metrics, such as TTFT, TPS, and user-perceived values for different models (Ding et al. 2024; Hu et al. 2024; Feng, Shen, and You 2024). This richer utility structure makes classical pricing models difficult to apply. To bridge this gap, our work introduces a Stackelberg congestion formulation explicitly parameterized by both objective and subjective preferences for the LLM servicing market.

2.2 LLM Service Evaluation

Existing evaluations of LLM services fall into two broad categories: (i) human-centric studies that elicit subjective quality judgements from small, controlled user panels (Chiang et al. 2024; Shankar et al. 2024; Ni et al. 2024); and (ii) automated benchmarks that score models on curated reference corpora with ground-truth answers (Hu and Zhou 2024; Mizrahi et al. 2024). Both streams produce leaderboards, yet neither is suitable for informing pricing decisions. Human-centric studies suffer from limited and non-representative samples, preventing the generalization of findings to the heterogeneous user base encountered in production systems. Automated benchmarks, conversely, rely on static test suites that may deviate from the task distributions encountered in live deployments; consequently, their reported scores exhibit low correlation with the utility users actually derive from the service. In contrast to these methods, we evaluate LLM performance through the lens of user-perceived value. This value is derived from the collective behavior of users in the LLM market and is primarily measured by the models' ability to enhance real-world task performance.

2.3 Model Relaxation in MPEC

A prevalent approach for investigating the optimal pricing strategy for LLM service providers within a congestion-aware Stackelberg game is to reformulate the task as an MPEC (Naoum-Sawaya and Elhedhli 2011; Cardellini, Di Valerio, and Presti 2016). Even a bilevel linear program is NP-hard (Friesz and Harker 1985). Existing research, therefore, either simplifies the problem itself (Harks and Schedel 2021; Cui, Hu, and Luo 2020; Wu, Barati, and Lim 2020; Fotouhi and Miller-Hooks 2021; Wu et al. 2021; Xiong et al. 2016) or relaxes the MPEC constraints (Li et al. 2019; Naoum-Sawaya and Elhedhli 2011; Böhnlein, Kratsch, and Schaudt 2021; Briest, Hoefer, and Krysta 2008), while some studies employ reinforcement learning to approximate these constraints so as to alleviate computational burden (Liu et al. 2020; Kuang et al. 2025). Nevertheless, users of LLM services are acutely sensitive to quality-of-service (QoS) factors such as latency and congestion. Consequently, model-simplification techniques that disregard heterogeneous user preferences or congestion effects are inadequate for this task (Harks and Schedel 2021). By contrast, methods that relax the MPEC constraints often require considerable computation time to attain high precision, whereas the complexity of the MPEC renders the underlying logic difficult for the RL network to master directly. Our approach is grounded in the analysis of the routing game's properties, which allows us to formulate a fully differentiable game abstraction which enables direct, end-to-end training of a neural network.

3 Stackelberg Routing Game

We model the LLM routing system as a Stackelberg routing game. We begin with an overview of the game setting, followed by a formal problem definition.

3.1 Preliminaries

The Stackelberg routing game comprises two types of entities: service providers and users.

Service Providers. The service providers develop, train, and maintain the LLMs, as well as host the LLM services. We denote them as $\mathcal{S}$. For each SP $s_j \in \mathcal{S}$, its service is characterized by user-perceived value b_j , service capacity α_j and a certain QoS, including metrics: transmission delay between users and congestion factor. The SP aims to set an optimal unit price p_j (e.g., dollars per million tokens) to maximize its profit. All the SPs' prices compose a pricing strategy profile $\mathcal{P}$ of the system, such that $\mathcal{P} = \{p_j\}_{j=1}^{m}$.

We focus on the pricing decision problem for a (given) target provider. Thus, we divide the set of SPs into a target provider s and the set of rival providers R, such that $\mathcal{S} = \{s\} \cup R$: (i) *Target Provider* (s): This is the specific provider whose pricing strategy is the central focus of our study. (ii) *Rival Providers* (R): This is a set of $m-1$ competing service providers, denoted as $R = \{r_1, r_2, \ldots, r_{m-1}\}$. Without loss of generality, we let $s_j \in \mathcal{S}$ be a target provider if $j = m$, otherwise s_j is a rival provider.

Users. We consider a set of n users, denoted as $U = \{u_1, u_2, \ldots, u_n\}$. A user is a group of population and is aggregated as an entity, like a commercial application or an enterprise client with a total token demand D_i. User u_i decides her allocation strategy $\boldsymbol{f}_i = \{f_{ij}\}_{j\in S}$, where f_{ij} indicates the amount of tokens allocated to SP s_j, which is determined by minimizing her routing-cost function $\mathcal{C}_i$:

$$\begin{aligned} \mathcal{C}_i &= \sum_{j\in S} f_{ij}\left(w_p p_j + w_q Q_j + w_d d_{ij} - b_j\right) \\ \text{s.t.} &\sum_{j\in S} f_{ij} = D_i, \quad f_{ij} \geq 0. \end{aligned} \tag{1}$$

where p_j, d_{ij}, and Q_j represent objective factors: unit price, transmission delay between s_j and u_i, and s_j's congestion factor, respectively. Subjective preferences are captured by b_j (user-perceived value of s_j's quality/brand) and w_p, w_q, w_d (subjective weights of the objective factors).

3.2 Game Formulation

We use the Stackelberg game to model the pricing decision process of s. In this game, s as the leader sets the price p_s according to the market conditions of U and R, and then the users as followers make their allocations based on unit price, TTFT and congestion factor of each service provider. We assume $Q_j = \sum_{i\in U} f_{ij}/\alpha_j$ in our model, where α_j is the service capacity of s_j. We can get the following definitions. The presence of service congestion means that users' selections are interdependent, as one user's choice can impact the service quality experienced by others. This interaction among users forms a non-cooperative game. Therefore, s's price decision-making process is also constrained by the Nash Equilibrium (NE) of this user-side game.

User-side Game Given the fixed price strategy profile $\mathcal{P}$ of all service providers, users strategically split their demands to minimize their expected acquisition cost. Let the strategy profile of u_i be $\boldsymbol{f}_i = \{f_{ij}\}_{j\in R\cup\{s\}}$. Denote the joint allocation strategy profile of all n users by $\mathcal{F} = (\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_n)$. For convenience, write $\mathcal{F}_{-i} = (\boldsymbol{f}_1, \ldots, \boldsymbol{f}_{i-1}, \boldsymbol{f}_{i+1}, \ldots, \boldsymbol{f}_n)$. Next, we define NE of the user-side game and prove the existence and uniqueness.

Definition 1 (Nash Equilibrium of the user-side game)**.** *A strategy profile $\mathcal{F} = (\boldsymbol{f}_i^*, \boldsymbol{f}_{-i}^*)$ is a Nash equilibrium of the user-side game if for any user u_i, it is true that $\mathcal{C}_i(\boldsymbol{f}_i^*; \mathcal{P}, \boldsymbol{f}_{-i}^*) \leq \mathcal{C}_i(\boldsymbol{f}_i'; \mathcal{P}, \boldsymbol{f}_{-i}^*)$ for any $\boldsymbol{f}_i' \neq \boldsymbol{f}_i^*$.*

Theorem 1. *A unique NE exists in the user-side game.*

Proof Sketch. The objective function of u_i (i.e.,Eq. (1)) is continuous, and the inequality and equality constraints are convex. Therefore, the feasible sets of Eq. (1) are closed, nonempty, and convex. The Hessian matrix of the cost function $\mathcal{C}_i$ is positive definite, which means that the second-order partial derivatives are greater than zero, i.e.,, $\nabla^2\mathcal{C}_i \succ 0$ and the cost function $\mathcal{C}_i$ is strictly convex. Thus, the followers form a concave n-person game. By Rosen's uniqueness theorem (Rosen 1964), the lower-layer game always has a unique equilibrium, and each follower's optimization problem can converge to a unique solution in the equilibrium state, regardless of the pricing strategy $\mathcal{P}$. □

Stackelberg Routing Game Based on user-side NE, we define the target s's pricing problem, which maximizes its profit Ψ_s by determining the best unit price p_s as below.

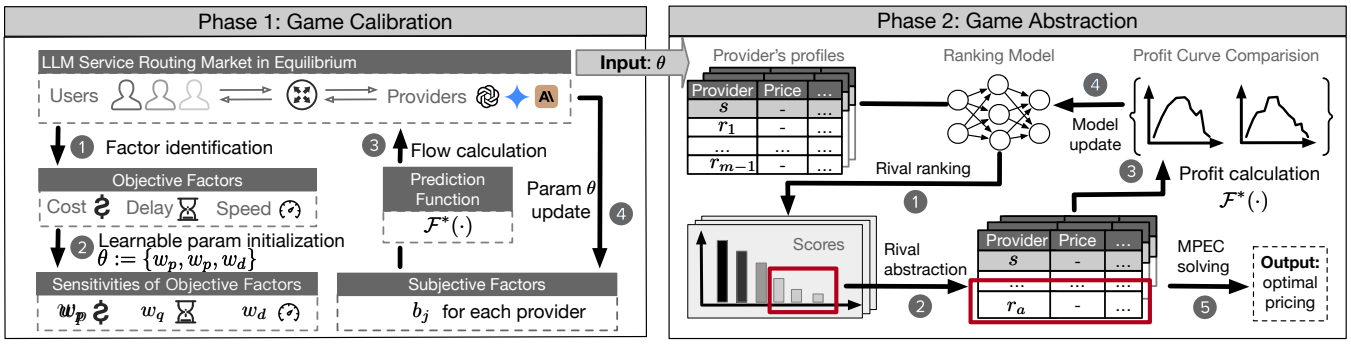


Figure 2: Overview of PriLLM.

Problem 1 (LLM Service Pricing Problem)**.** *Given the user-side NE, s determines p_s to maximize its profit Ψ_s.*

$$\begin{aligned} \max_{p_s} \quad & \Psi_s(p_s) = p_s \cdot Q_s^*(p_s) \cdot \alpha_s \\ \text{s.t.} \quad & \textit{User-side game reaches equilibrium} \\ & 0 \le p_s \le p^{max}, \end{aligned} \tag{2}$$

where $Q_s^*(p_s)\alpha_s$ is the equilibrium service demand for s resulting from user-side NE.

The Stackelberg routing game (Problem 1) is intractable: its leader–follower structure, coupled with the user-side Nash equilibrium, yields an MPEC that remains NP-hard even in the linear case (Friesz and Harker 1985). Moreover, the user cost function $\mathcal{C}_i$ in Equation 1 contains preference parameters $\boldsymbol{\theta} = \{w_p, w_q, w_d, \{b_j\}\}$ that encode latent sensitivities; unless these parameters are calibrated on real market data, the model suffers a pronounced sim-to-real gap and delivers pricing policies of inferior quality.

Accordingly, we introduce PriLLM, a two-phase framework. An overview of PriLLM is provided in Figure 2: Phase 1 refines the game model by extracting latent user preferences from observational data; Phase 2 scales the market efficiently and compresses the pricing problem to a simplified form for live LLM providers.

4 Data-Driven Game Calibration

To calibrate the game model with the data from the real-life LLM routing platforms, we propose a paradigm shift from traditional model-first approaches to a data-driven framework. Instead of manually setting the user preference parameters θ, we learn them directly from routing data. We consolidate these learnable parameters into a single set $\boldsymbol{\theta}$ to simplify notation, after normalizing the price weight to $w_p = 1$ for model identifiability: $\boldsymbol{\theta} = \{1, w_q, w_d, \{b_j\}_{j\in S}\}$. We demonstrate the learnability of the parameter set $\boldsymbol{\theta}$.

To achieve game calibration, Phase 1 of PriLLM contains four key steps. First, PriLLM identifies objective factors from the real-world data and generates a strong initial estimate for θ. Subsequently, PriLLM leverages a prediction function to compute the expected routing flows under the current parameters $\boldsymbol{\theta}$. The discrepancy between these predicted flows and the actual routing data is then quantified as an error signal, which is backpropagated to refine $\boldsymbol{\theta}$. This refinement process is repeated until we obtain the final parameters that best explain the real-world LLM routing flows.

4.1 Learnability of Game Parameters

Theorem 1 implies that for any given S and its pricing strategy $\mathcal{P}$, the user-side game NE is a single, routing flow $\mathcal{F}^*$. This allows us to treat the user-side game as a function $\mathcal{F}^*(\cdot)$ mapping objective factors of each s_j, $\boldsymbol{O} = \{p_j, \alpha_j, \{d_{ij}\}_{i\in U}\}_{j\in S}$, demnad of users, $\{D_i\}_{i\in U}$ and user preference parameters, $\boldsymbol{\theta}$ to a unique equilibrium flow.

$$\mathcal{F}^* = \mathcal{F}^*(\boldsymbol{\theta}, \boldsymbol{O}, \{D_i\}_{i\in U}) \tag{3}$$

To achieve $\mathcal{F}^*(\cdot)$, we construct a potential function $\Phi(\mathcal{F})$ for the game. To simplify its presentation, we define it as the sum of two components: a fixed cost component $\Phi_{\text{Fixed}}(\mathcal{F})$ and a congestion cost component $\Phi_{\text{Congestion}}(\mathcal{F})$. Let $\Phi_{\text{Fixed}}(\mathcal{F})$ be defined as:

$$\Phi_{\text{Fixed}}(\mathcal{F}) = \sum_{i=1}^{n} \sum_{j\in S} \left(w_p p_j + w_d d_{ij} - b_j\right) f_{ij} \tag{4}$$

And let $\Phi_{\text{Congestion}}(\mathcal{F})$ be the total delay cost:

$$\Phi_{\text{Congestion}}(\mathcal{F}) = \sum_{j\in S} \frac{w_q}{2\alpha_j} \left((Q_j \alpha_j)^2 + \sum_{i=1}^{n} f_{ij}^2 \right) \tag{5}$$

The potential function is the sum of these two parts: $\Phi(\mathcal{F}) = \Phi_{\text{Fixed}}(\mathcal{F}) + \Phi_{\text{Congestion}}(\mathcal{F})$, where $\mathcal{F}^*$ is the uniqueness solution to minimization of $\Phi(\mathcal{F}; \boldsymbol{\theta}, \boldsymbol{O}, \{D_i\}_{i\in U})$. A proof is provided in the extended version. To learn $\boldsymbol{\theta}$ via gradient-based methods, we also need compute the gradient of a loss function through $\mathcal{F}^*(\cdot)$.

Theorem 2. *The user-side NE prediction function $\mathcal{F}^*(\boldsymbol{\theta})$ is piecewise differentiable. Consequently, for any loss function $\mathcal{L}(\mathcal{F}^*)$, the gradient $\nabla_{\boldsymbol{\theta}}\mathcal{L}$ exists (as a sub-gradient at non-differentiable points) and can be learned end-to-end using modern automatic differentiation frameworks.*

Proof Sketch. The unique NE is the solution to a strictly convex quadratic program (QP) derived from the user-side game(a potential game), with $\boldsymbol{\theta}$ appearing as coefficients in the $\mathcal{C}_i$. The solution $\mathcal{F}^*$ is characterized by the Karush-Kuhn-Tucker (KKT) conditions, which form an implicit

function of $\boldsymbol{\theta}$. Automatic differentiation libraries can differentiate through the solution of such convex optimization problems. The piecewise nature, which arises from changes in the set of active constraints (i.e., users' service routing), is handled by these frameworks, which compute valid subgradients at points of non-differentiability. □

4.2 Initialize and Learn Game Parameters

The learning process is sensitive to the initial values of $\boldsymbol{\theta}$, as poor initialization often leads to a failure to converge. Hence, we propose an initialization strategy to get the high quality initial values $\{b_j\}$ for $\boldsymbol{\theta}$. The method takes representative days of real-world traffic data ($\mathcal{F}_t^{\text{real}}$) as input, and outputs a initial parameters $\boldsymbol{\theta}_{\text{init}}$, detailed in extended version. The core idea is to assume the observed data $\mathcal{F}_t^{\text{real}}$ already represents a user NE. Based on this assumption, we work backward to find the inherent biases $\{b_j\}$ of the services with fixed weights $\{w_p = 1, w_d = 1, w_q = 1\}$. In a NE, for any user, all the services they actually use must be equally costly or attractive, and these must be more attractive than any service they do not use. This principle allows us to formulate a simple Linear Program to find the smallest non-negative biases $\{b_j^*\}$ that make the real-world data calibrated with our game model. We then form our initial parameter vector:$\{w_p = 1.0, w_d = 1.0, w_p = 1.0, \{b_j^*\}\}$.

Given initial parameter vector and T days' real-world routing data as T NE of user-side game, i.e.,, the real daily traffic distributions, objective factors of S, demand of each user, our objective is to find the parameters $\boldsymbol{\theta}^*$ that best account for the observed data. Hence we minimize a loss function that quantifies the discrepancy between the model's predicted NE and the actual NE:

$$\min_{\boldsymbol{\theta}} \quad \mathcal{L}(\boldsymbol{\theta}) = \sum_{t=1}^{T} \left\| \mathcal{F}^*(\boldsymbol{\theta}; \{D_{it}\}_{i\in U}, \mathbf{O}_t) - \mathcal{F}_t^{\text{real}} \right\|_2^2 \quad (6)$$

where $\mathcal{F}_t^{\text{real}}, \{D_{it}\}_{i\in U}$ is the routing data and demand data of day t. By minimizing the objective function $\mathcal{L}(\boldsymbol{\theta})$, we can find a local optima using gradient-based methods.

5 Dynamic Pricing with Game Abstraction

To tackle the Stackelberg routing game, PriLLM uses a novel learning framework to simplify the routing market while preserving the user-side NE for the target providers s. Within the simplified market, we can efficiently solve the MPEC problem to find a near-optimal price for target s.

As illustrated in Phase 2 of Fig 2, the process is as follows. First, PriLLM employs a ranking model to assign a score to each rival and aggregates the rivals with lower scores. PriLLM then assesses the quality of the abstracted routing game by comparing s's profit curve to that of the original game, and updates the ranking model accordingly. By using routing game abstraction, we can efficiently solve for a near-optimal pricing to the s's routing game.

5.1 Rival Ranking and Abstraction

The main idea of PriLLM's abstraction is to preserve the most influential $K-1$ rivals while aggregating the less significant ones. It can maintain the complexity of the market competition while reducing the computational load.

To rank the rival providers, PriLLM encodes each rival $r_j \in R$ into a feature vector encapsulating its objective factors, user-perceived value and related factor with s. These embeddings are processed by a stack of Set Attention Blocks (SABs) (Lee et al. 2019). To accommodate different aggregation needs, our SAB module is designed to compute two parallel sets of importance scores for each rival:

$$(score_j^{\text{sum}}, score_j^{\text{avg}}) = SAB(p_j, \{d_{ij}\}, \alpha_j, b_j; \boldsymbol{\theta}, \boldsymbol{O}, \{D_i\})$$

PriLLM preserves the top $K-1$ rivals identified by the averaging score. All remaining rivals R_{agg} are then aggregated into a single representative rival, r_a. The attributes of r_a are synthesized using the two learned score types: cumulative properties like α_a are calculated via a weighted sum:

$$\alpha_a = \sum_{r_j \in R_{\text{agg}}} score_j^{\text{sum}} \cdot \alpha_j$$

while attributes like $p_a, \{d_{ia}\}, b_a$ are computed through a weighted average:

$$(p_a, \{d_{ia}\}, b_a) = \frac{\sum_{r_j \in R_{\text{agg}}} (score_j^{\text{avg}} \cdot (p_j, \{d_{ij}\}, b_j))}{\sum_{r_j \in R_{\text{agg}}} score_j^{\text{avg}}}.$$

This process transforms the original market into a simplified market by reducing the size of the rival set.

5.2 Loss Function and Model Solving

Our central hypothesis is that if the profit curve of target s in an abstracted routing game aligns with the original profit curve, the abstracted routing game's derived optimal price will approximate the true optimum. Therefore, we evaluate the quality of an abstracted game by comparing the two profit curves of target s. We employ a sampling-based method. By sampling multiple price points for target s, we repeatedly compute and compare its profit by $\mathcal{F}^*(\cdot)$ in both the original and the abstracted routing game.

Given the orginal profit vector $\mathbf{Y}$ and the predicted profit vector $\hat{\mathbf{Y}}(\phi)$ over a set of sampled prices points, we train PriLLM, parameterized by ϕ, by minimizing:

$$\mathcal{L}\text{curve}(\phi) = \frac{1}{L} \sum_{k=1}^{L} \left(\frac{Y_k}{||\mathbf{Y}||_\infty} - \frac{\hat{Y}k(\phi)}{||\mathbf{Y}||\infty} \right)^2, \quad (7)$$

where ϕ represents the parameters of our deep aggregation network. X is the total number of candidate price points sampled for the target provider s. $\mathbf{Y} = [Y_1, \ldots, Y_L]$ is the profit vector in which each Y_k is the profit for provider s at the k-th candidate price, computed in the original routing game. But $\hat{\mathbf{Y}}(\phi) = [\hat{Y}_1(\phi), \ldots, \hat{Y}_L(\phi)]$ is computed in the simplified game generated by PriLLM's aggregator. $||\mathbf{Y}||_\infty$ is the L-infinity norm (i.e., the maximum absolute value) of the original profit vector. We use it to normalize both curves, which stabilizes the training process against varying scales of profit. Detail calculation of NE is provided in extended version.

Scenario	Premium Market		Economy Market		Coding Market		Translation Market	
Problem Size	(4 LLMs, 900B tokens)		(8 LLMs, 1200B tokens)		(13 LLMs, 800B tokens)		(10 LLMs, 40B tokens)	
Metric	**Time(s)**	**Profit(%)**	**Time(s)**	**Profit(%)**	**Time(s)**	**Profit(%)**	**Time(s)**	**Profit(%)**
PriLLM	**1.04**	**98.5%**	**3.01**	**96.2%**	**4.95**	**95.1%**	**3.98**	**95.8%**
pMPEC	1.64	91.9%	21.02	88.3%	95.04	85.4%	35.17	87.5%
Smooth	3.29	88.1%	135.2	84.5%	680.5	81.1%	210.8	83.8%
SPGCE	2.24	89.2%	18.04	85.1%	105.22	80.5%	32.66	64.2%
ODCA	2.37	62.8%	28.15	51.3%	101.17	64.7%	39.49	78.6%

Table 1: Performance comparison of pricing algorithms with optimal profit across different market scenarios.

After abstraction, the problem can be formulated as an MPEC via the KKT conditions. We can split the price domain of p_s into ordered intervals, solve the resulting subproblems, and keep the best price; As every feasible price is examined, the solution is optimal. All solution methods are given in Baselines of Evaluation section.

6 Evaluation

Using real-life datasets, this section experimentally evaluate the effectiveness and efficiency of PriLLM framework.

Dataset. We collected three months of historical data for the 20 most popular LLM services and 20 APPs from OpenRouter. We train PriLLM's deep aggregation network using a dataset of over 2,000 market scenarios constructed through data augmentation. Each scenario is derived from a real daily market snapshot, with perturbations applied to the attributes of rivals to simulate diverse market conditions.

Configurations. From this dataset, one provider is randomly selected as the target provider, s, with the rest forming the set of rivals, R. We extract the API input unit prices as p_j and total daily token usage as the total user demands. To simulate a market of enterprise clients, we model the total usage flow of one commercial APP as one user. Key QoS metrics are simulated based on empirical data: user-specific TTFTs are sampled from the observed distribution on OpenRouter, and service capacities $\alpha_j = \sum_{i \in U} f_{ij}/v_j$ where v_j are the models' official TPS ratings. We report profit as a relative metric, defined as the ratio to the optimal profit.

Enviroment settings. We ran experiments on a 64-bit machine with an Intel i7-8550U CPU and 24GB RAM. Our framework is implemented in Python 3.8. The MPEC problems are modeled using Pyomo 6.1.2. The learned parameters of PriLLM are based on a portion of the historical data, and evaluation is performed on a held-out test set.

Baselines. We use the following MPEC solvers to solve the Stackelberg routing game. (1) SPITER (Jin et al. 2024), the default solver integrated in PriLLM. (2) BF (Brute-Force), which enumerats all KKT conditions of the user-side game. It provides the theoretical optimum. (3) pMPEC (Hart et al. 2017), a generic solvers implemented via the Pyomo library. (4) Smooth (Wu et al. 2021), which relaxes the complementarity constraints into smooth inequalities, allowing the use of standard nonlinear solvers. (5) ODCA (Chen et al. 2020) & SPGCE (Harks and Schedel 2021), a simplified MPEC solver by ignoring user interaction effects.

For market simulation, we include the following baselines: (6) NPM, which models the user cost function solely based on observable, objective metrics. (7) XGBoost (Chen and Guestrin 2016), a non-game-theoretic approach that directly predicts user choices from market features.

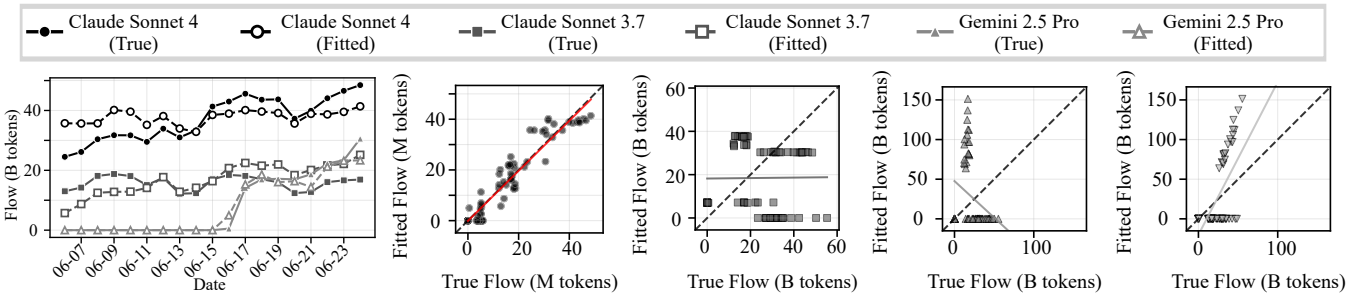
We also compare with baselines for subjective preference learning: (8) FD, a black-box optimization method that approximates the gradients of the user equilibrium through finite differences. (9) A2C (Liu et al. 2020), an actor-critic reinforcement learning method that learns an optimal policy for selecting preference parameters.

6.1 Overall Performance of PriLLM

We conduct experiments on PriLLM to validate the game calibration and the game abstraction method. To evaluate PriLLM under diverse competitive conditions, we formulated the LLM market into four typical scenarios: Premium and Economy markets are delineated by API price points (e.g., above/below $1 per million tokens), while Coding and Translation markets are formulated based on application-specific usage data. This partitioning varies problem sizes and competitive dynamics, as summarized in Table 1.

Effectiveness of game calibration. We tested PriLLM on the coding market which includes 13 popular LLM service providers and 8 coding apps with token usage exceeding 2 B tokens. Fig. 3a shows that PriLLM achieves good fitting performance in both the 2-LLM Model and 3-LLM Model coding market scenarios based on the learned parameters. Figure 3b show that our model achieves a high R^2 score of 0.8982 and a low Mean Absolute Error (MAE) of 3.56M tokens. Comparing with Fig. 3c, we conclude that PriLLM has stronger prior knowledge than traditional machine learning methods and can learn effective information when relevant data is sparse. Comparing with Fig. 3d and Fig. 3e, we see that the b_j and congestion effect Q_j considered in the modeling of PriLLM improve the ability of calibrating.

Effectiveness of game abstraction. We validate PriLLM's ability to simplify the Stackelberg routing game. We use the deep aggregation network to get a simplified market with $K = 2$ rivals, for which $K = 2$ is a good choice in terms of solution efficiency and aggregation accuracy. Then We compare PriLLM's efficiency and accuracy with different methods on the various market



(a) Calibration of LLM Routing Flows. (b) Fit by PriLLM. (c) Fit by XGBoost. (d) Fit by NPM. (e) Fit by PriLLM$_{\text{noQ}}$.

Figure 3: Experiment of PriLLM's Game Calibration.

settings. The results are mainly represented in Table 1. Comparing PriLLM with pMPEC and Smooth methods, we can find that according to the aggregation results obtained by PriLLM, the SPITER method can solve the problem in a faster time and obtain profit close to the optimal solution.

6.2 Impacts of Key Components

Impact of game-parameter learning method. We study the impact of the game-parameter learning methods on game calibration. First, we collected data from the four most popular LLMs for 19 consecutive days, including price, TTFT, TPS, and usage tokens from various apps. We then used this data to learn user subjective parameters in our game and then calibrate them against the real market. Parameters learned using the A2C or FD methods consistently failed to effectively calibrate the model to real-world data. Due to the complexity of the underlying user game, A2C methods struggle to learn. FD methods treat the underlying user solution process as a black box, requiring a significant time-consuming gradient calculation and manual adjustment of the update step size Table 2 shows that only our method can stably learn parameters. On the dataset, the MSE error can be reduced by 45.73 within 25.88 seconds.

Method	Converges?	Final MSE	Time (s)
PriLLM	**Yes**	**45.73**	**25.88**
FD	No	1.63×10^6	—
A2C	No	3.14×10^{11}	—

Table 2: Impact of Game-parameter Learning Methods.

Impact of aggregation methods. We evaluate the impacts of aggregation methods on game abstraction. First, we selected data from eight LLMs in the Economy market as the baseline experimental setup, including unit price, TTFT, TPS, and usage tokens from various apps. Second, we varied the number of LLMs from small to large (5 and 7 rival providers). Finally, based on the resulting markets of varying sizes, we simplified the market using different aggregation methods and uniformly calculated the optimal price using the SPITER algorithm. In Table 3, we compared the profits obtained using different aggregation methods. DA$_{K=2}$ We also compared the time taken to calculate the price using the BF method directly without using any aggregation method. In Table 3, we can see that PriLLM achieves near-optimal profits (over 97% of the optimum) while drastically reducing computation time compared to the exact solver.

	6 LLMs Market		8 LLMs Market	
Method	**Profit(%)**	**Time(s)**	**Profit(%)**	**Time(s)**
BF	100.0%	420.63	100.0%	612.24
DA $_{K=1}$	93.8%	5.52	92.1%	9.85
DA $_{K=2}$	95.2%	9.18	94.5%	13.24
DA $_{K=3}$	99.7%	19.17	99.5%	20.21
DA $_{K=4}$	99.9%	62.65	99.9%	61.22
MIN $_{K=2}$	58.1%	6.37	51.7%	14.09
AVG	18.9%	5.15	15.3%	9.98

Table 3: Impact of Aggregation Methods. BF is the brute-force method without aggregation. DA stands for the deep aggregation network. MIN selects the K cheapest rivals. AVG returns K identical providers with averaged attributes.

Impact of Parameter K. We also analyze the impact of the number of aggregated rivals K on the model's effectiveness and efficiency. We conduct this experiment on the 8 LLM market settings and compare with heuristic method. We vary K from 1 to 4 for PriLLM and measure the resulting profit and total computation time. Table 3 shows that increasing K from 1 to 4 reduces the profit gap from 11.53% to a near-zero 0.06%. Our experiment shows that using $K = 2$ aggregated rivals offers a profit improvement over $K = 1$ for a increase in runtime.

7 Conclusion

We introduced PriLLM, a framework for dynamic pricing in LLM service markets. By formulating the problem as a Stackelberg game and leveraging novel data-driven calibration and a deep aggregation network, PriLLM overcomes the limitations of traditional approaches. Our evaluation on real-world data confirms that PriLLM achieves near-optimal profit with efficiency. As future work, we will expand PriLLM from single leader pricing to multiple leaders.

Acknowledgments

This work is supported by National Natural Science Foundation of China under Grants No. 62572119 and 62232004, Jiangsu Provincial Key Laboratory of Network and Information Security under Grants No.BM2003201, Key Laboratory of Computer Network and Information Integration of Ministry of Education of China under Grants No.93K-9, and partially supported by Collaborative Innovation Center of Novel Software Technology and Industrialization, Collaborative Innovation Center of Wireless Communications Technology. We also thank the Big Data Computing Center of Southeast University for providing the experiment environment and computing facility.

References

Böhnlein, T.; Kratsch, S.; and Schaudt, O. 2021. Revenue Maximization in Stackelberg Pricing Games: Beyond the Combinatorial Setting. *Mathematical Programming*, 187(1-2): 653–695.

Briest, P.; Hoefer, M.; and Krysta, P. 2008. Stackelberg Network Pricing Games. arxiv:0802.2841.

Cardellini, V.; Di Valerio, V.; and Presti, F. L. 2016. Game-theoretic resource pricing and provisioning strategies in cloud systems. *IEEE Transactions on Services Computing*, 13(1): 86–98.

Chakraborty, A.; Mondal, A.; Roy, A.; and Misra, S. 2021. Dynamic Trust Enforcing Pricing Scheme for Sensors-as-a-Service in Sensor-Cloud Infrastructure. *IEEE Transactions on Services Computing*, 14(5): 1345–1356.

Chen, T.; and Guestrin, C. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 785–794.

Chen, Y.; Li, Z.; Yang, B.; Nai, K.; and Li, K. 2020. A Stackelberg Game Approach to Multiple Resources Allocation and Pricing in Mobile Edge Computing. *Future Generation Computer Systems*, 108: 273–287.

Chiang, W.-L.; Zheng, L.; Sheng, Y.; Angelopoulos, A. N.; Li, T.; Li, D.; Zhu, B.; Zhang, H.; Jordan, M.; Gonzalez, J. E.; et al. 2024. Chatbot arena: An open platform for evaluating llms by human preference. In *Forty-first International Conference on Machine Learning*.

Cui, Y.; Hu, Z.; and Luo, H. 2020. Optimal Day-Ahead Charging and Frequency Reserve Scheduling of Electric Vehicles Considering the Regulation Signal Uncertainty. *IEEE Transactions on Industry Applications*, 56(5): 5824–5835.

Ding, D.; Mallick, A.; Wang, C.; Sim, R.; Mukherjee, S.; Ruhle, V.; Lakshmanan, L. V.; and Awadallah, A. H. 2024. Hybrid llm: Cost-efficient and quality-aware query routing. *arXiv preprint arXiv:2404.14618*.

Ding, N.; Gao, L.; and Huang, J. 2023. Optimal Pricing Design for Coordinated and Uncoordinated IoT Networks. *IEEE Transactions on Mobile Computing*, 22(12): 7121–7137.

Feng, T.; Shen, Y.; and You, J. 2024. Graphrouter: A graph-based router for llm selections. *arXiv preprint arXiv:2410.03834*.

Fotouhi, H.; and Miller-Hooks, E. 2021. Optimal Time-Differentiated Pricing for a Competitive Mixed Traditional and Crowdsourced Event Parking Market. *Transportation Research Part C: Emerging Technologies*, 132: 103409.

Friesz, T. L.; and Harker, P. T. 1985. Properties of the iterative optimization-equilibrium algorithm. *Civil Engineering Systems*, 2(3): 142–154.

Harks, T.; and Schedel, A. 2021. Stackelberg Pricing Games with Congestion Effects. *Mathematical Programming*.

Hart, W. E.; Laird, C. D.; Watson, J.-P.; Woodruff, D. L.; Hackebeil, G. A.; Nicholson, B. L.; Siirola, J. D.; et al. 2017. *Pyomo-optimization modeling in python*, volume 67. Springer.

Hu, Q. J.; Bieker, J.; Li, X.; Jiang, N.; Keigwin, B.; Ranganath, G.; Keutzer, K.; and Upadhyay, S. K. 2024. Routerbench: A benchmark for multi-llm routing system. *arXiv preprint arXiv:2403.12031*.

Hu, T.; and Zhou, X.-H. 2024. Unveiling llm evaluation focused on metrics: Challenges and solutions. *arXiv preprint arXiv:2404.09135*.

Huang, S.; Huang, H.; Gao, G.; Sun, Y.-E.; Du, Y.; and Wu, J. 2023. Edge Resource Pricing and Scheduling for Blockchain: A Stackelberg Game Approach. *IEEE Transactions on Services Computing*, 16(2): 1093–1106.

Jin, J.; Guo, Z.; Bai, W.; Wu, B.; Liu, X.; and Wu, W. 2024. Congestion-aware Stackelberg pricing game in urban Internet-of-Things networks: A case study. *Computer Networks*, 246: 110405.

Kuang, Y.; Li, X.; Wang, J.; Zhu, F.; Lu, M.; Wang, Z.; Zeng, J.; Li, H.; Zhang, Y.; and Wu, F. 2025. Accelerate presolve in large-scale linear programming via reinforcement learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Lee, J.; Lee, Y.; Kim, J.; Kosiorek, A.; Choi, S.; and Teh, Y. W. 2019. Set Transformer: A Framework for Attention-based Permutation-Invariant Neural Networks. In Chaudhuri, K.; and Salakhutdinov, R., eds., *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, 3744–3753. PMLR.

Li, R.; Wei, W.; Mei, S.; Hu, Q.; and Wu, Q. 2019. Participation of an Energy Hub in Electricity and Heat Distribution Markets: An MPEC Approach. *IEEE Transactions on Smart Grid*, 10(4): 3641–3653.

Liu, Y.; Wang, W.; Hu, Y.; Hao, J.; Chen, X.; and Gao, Y. 2020. Multi-Agent Game Abstraction via Graph Attention Neural Network. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05): 7211–7218.

Mizrahi, M.; Kaplan, G.; Malkin, D.; Dror, R.; Shahaf, D.; and Stanovsky, G. 2024. State of what art? a call for multi-prompt llm evaluation. *Transactions of the Association for Computational Linguistics*, 12: 933–949.

Naoum-Sawaya, J.; and Elhedhli, S. 2011. Controlled Predatory Pricing in a Multiperiod Stackelberg Game: An MPEC Approach. *Journal of Global Optimization*, 50(2): 345–362.

Naoum-Sawaya, J.; and Elhedhli, S. 2011. Controlled predatory pricing in a multiperiod Stackelberg game: an MPEC approach. *Journal of Global Optimization*, 50(2): 345–362.

Ni, J.; Xue, F.; Yue, X.; Deng, Y.; Shah, M.; Jain, K.; Neubig, G.; and You, Y. 2024. Mixeval: Deriving wisdom of the crowd from llm benchmark mixtures. *Advances in Neural Information Processing Systems*, 37: 98180–98212.

Rosen, J. B. 1964. Existence and uniqueness of equilibrium points for concave n-person games. *Econometrica*, 33: 520–534.

Saxena, D.; and Singh, A. K. 2024. An Oversubscription and Service Pricing Exploitation-Based Profit Maximization Framework for Industry Cloud Resource Management. *IEEE Transactions on Services Computing*, 17(5): 2041–2053.

Shankar, S.; Zamfirescu-Pereira, J.; Hartmann, B.; Parameswaran, A.; and Arawjo, I. 2024. Who validates the validators? aligning llm-assisted evaluation of llm outputs with human preferences. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*, 1–14.

Song, W.; Huang, Z.; Cheng, C.; Gao, W.; Xu, B.; Zhao, G.; Wang, F.; and Wu, R. 2025. IRT-Router: Effective and Interpretable Multi-LLM Routing via Item Response Theory. In Che, W.; Nabende, J.; Shutova, E.; and Pilehvar, M. T., eds., *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 15629–15644. Vienna, Austria: Association for Computational Linguistics. ISBN 979-8-89176-251-0.

Tütüncüoğlu, F.; and Dán, G. 2024. Optimal Service Caching and Pricing in Edge Computing: A Bayesian Gaussian Process Bandit Approach. *IEEE Transactions on Mobile Computing*, 23(1): 705–718.

Wu, B.; Zhu, X.; Liu, X.; Jin, J.; Xiong, R.; and Wu, W. 2021. Revenue Maximization of Electric Vehicle Charging Services with Hierarchical Game. In *Proceedings of the 16th International Conference on Wireless Algorithms, Systems, and Applications (WASA)*, 417–429. Springer.

Wu, Y.; Barati, M.; and Lim, G. J. 2020. A Pool Strategy of Microgrid in Power Distribution Electricity Market. *IEEE Transactions on Power Systems*, 35(1): 3–12.

Xiong, Y.; Gan, J.; An, B.; Miao, C.; and Soh, Y. C. 2016. Optimal Pricing for Efficient Electric Vehicle Charging Station Management. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, AAMAS '16, 749–757. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450342391.

Yue, Y.; Zhang, G.; Liu, B.; Wan, G.; Wang, K.; Cheng, D.; and Qi, Y. 2025. MasRouter: Learning to Route LLMs for Multi-Agent Systems. In Che, W.; Nabende, J.; Shutova, E.; and Pilehvar, M. T., eds., *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 15549–15572. Vienna, Austria: Association for Computational Linguistics. ISBN 979-8-89176-251-0.