# Auctionformer: A Unified Deep Learning Algorithm for Solving Equilibrium Strategies in Auction Games

**Kexin Huang** [† 1] **Ziqian Chen** [2] **Xue Wang** [2] **Chongming Gao** [1] **Jinyang Gao** [2] **Bolin Ding** [2] **Xiang Wang** [‡ 1]

## Abstract

Auction games have been widely used in plenty of trading environments such as online advertising and real estate. The complexity of real-world scenarios, characterized by diverse auction mechanisms and bidder asymmetries, poses significant challenges in efficiently solving for equilibria. Traditional learning approaches often face limitations due to their specificity to certain settings and high resource demands. Addressing this, we introduce *Auctionformer*, an efficient transformer-based method to solve equilibria of diverse auctions in a unified framework. Leveraging the flexible tokenization schemes, Auctionformer translates varying auction games into a standard token series, making use of renowned Transformer architectures. Moreover, we employ Nash error as the loss term, sidestepping the need for underlying equilibrium solutions and enabling efficient training and inference. Furthermore, a few-shot framework supports adaptability to new mechanisms, reinforced by a self-supervised fine-tuning approach. Extensive experimental results affirm the superior performance of Auctionformer over contemporary methods, heralding its potential for broad real-world applications.

## 1. Introduction

Auctions serve as a cornerstone in various trading environments including art acquisitions (Louargand & McDaniel, 1991), radio spectrum allocations (Chen et al., 2010), real estate sales (Mayer, 1998), and online advertising (Bigler, 2008; Chen, 2017). Notably, online advertising has witnessed an exponential surge in trading volume, amounting to an astonishing 209.7 billion in recent decades, facilitated by sophisticated auction mechanisms (Interactive Advertising Bureau & PwC, 2023). A typical auction includes two components, the bidders and the mechanism. Illustratively, consider a classic single-item auction where bidders are availed information related to the underlying value of the item on offer. Based on this revelation, each bidder is tasked with setting a price he/she is willing to pay for the item. After collecting all bid prices, the highest bidder wins the item at the price by predetermined mechanisms.

While researchers achieved theoretical equilibria in certain specific auction environments (Vickrey & William, 1961; Jackson & Swinkels, 2005; Athey & Haile, 2002), the field underwent a paradigm shift with the adoption of learning approaches. These methods, including no-regret learning (Feng et al., 2021; Bichler et al., 2021; Kohring et al., 2023; Deng et al., 2022) and multi-agent reinforcement learning (Kolumbus & Nisan, 2022; Banchio & Skrzypacz, 2022), have fostered the development of (approximate) equilibrium strategies by leveraging learning approaches to either mitigate regret or augment expected rewards (Bichler et al., 2022; Badanidiyuru et al., 2021; Nedelec et al., 2021). Nonetheless, these approaches are mainly specifically designed to solve a unique auction setting, such as a fixed number of bidders and/or fixed distributions, and are both time and resource-intensive.

In a related domain, the area of auction design has witnessed the successful application of deep learning techniques (Dütting et al., 2021; 2024), notably attention-based architectures, showcasing their potential for flexible input handling and generalizability (Ivanov et al., 2022). This raises a pivotal question:

*Can a trained deep learning model efficiently compute approximate equilibria for any mechanism only with fine-tuning?*

Motivated by this question and drawing inspiration from the remarkable successes of Transformer models (Vaswani et al., 2017; Ivanov et al., 2022), we introduce Auctionformer, a Transformer model to solve general auction games. Auction-

[†]Work done at Alibaba [‡] Xiang Wang is also affiliated with Institute of Dataspace, Hefei Comprehensive National Science Center [1]University of Science and Technology of China, Hefei, China [2]Alibaba Group, Hangzhou, China. Correspondence to: Ziqian Chen <eric.czq@alibaba-inc.com>, Xue Wang <xue.w@alibaba-inc.com>.

*Proceedings of the 41st International Conference on Machine Learning*, Vienna, Austria. PMLR 235, 2024. 

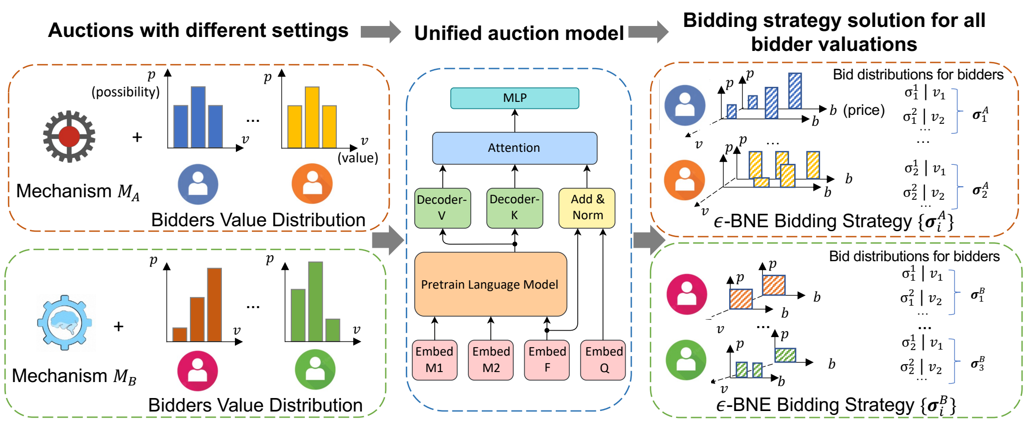


*Figure 1.* The pipeline of the Auctionformer framework.

former uses the pretrained Transformers (e.g., BERT (Devlin et al., 2018) and GPT (Radford et al., 2019; OpenAI, 2023)) as the backbone. We design a flexible tokenization procedure that is able to translate diverse auction games into a standardized series of tokens, thereby encoding the mechanism and its intricate details (e.g., entrance fee) succinctly. Furthermore, we propose to use Nash error which is also known as *Nikaidö and Isoda* function (Nikaidô & Isoda, 1955) as the loss term instead of the commonly used mean squared error (MSE) and cross-entropy error (CE). Nash error loss relaxes the necessity of underlying equilibrium solutions whose acquisition can be time-consuming (Filos-Ratsikas et al., 2021; Chen & Peng, 2023), thereby circumventing the steep demands of constructing auction equilibria datasets with ground truth. When using Nash error, we only need to engage cheap local solutions during the training stage, and the estimated equilibrium is computed with just one forward path when switching to the inference stage. To adapt the model to the auction settings different from the training stage, we provide a few-shot framework. By training a new mechanism identifier token with just a few samples, Auctionformer can be adapted to approximate equilibrium bidding strategies for all cases within the new mechanism. Our contributions can be summarized as follows:

- We propose Auctionformer, a Transformer model, to solve general auction game equilibria. To our knowledge, Auctionformer is the first machine-learning model that is able to solve multiple auction games simultaneously, which enhances efficiency and broadens the scope of learning-based auction solvers.
- We introduce flexible tokenization procedures that facilitate the conversion of a wide array of existing auction games into a unified series of tokens, and then an encoder-decoder architecture is used to solve bidders' strategies. We also use the Nash error as the loss function instead of conventional MSE or CE loss. The underlying equilibrium solution is not required throughout the whole training procedure.
- Our approach is tested through extensive experiments encompassing end-to-end training, pretraining-finetuning, and few-shot learning methodologies. These experimentations underscore the remarkable efficacy of our proposed model, which outperforms existing methods by a significant margin, thereby attesting to its superiority and potential for real-world applications.

*Table 1.* The equilibrium solving ability in auction games. The scalability of different games are leveraged on player number (Num.), mechanism.

| Ability | Classic Solver | No-regret (MARL) | MLPs | Ours |
|---|---|---|---|---|
| Vary Player Num. | ✓ | ✓ | ✗ | ✓ |
| Vary Mechanisms | ✗ | ✓ | ✗ | ✓ |
| Without Retrain | ✓ | ✗ | ✓ | ✓ |
| Few-shot | - | ✗ | ✗ | ✓ |

## 2. Preliminary

We consider the classical single-item auction problem, which is characterized by a set of bidders $\mathcal{N}$ and a mechanism $\mathcal{M}$. The $i$-th bidder in $\mathcal{N}$ is featured by his or her value distribution $F_i$, and bidding strategy $\sigma_i$ that is associated with mechanism $\mathcal{M}$ and number of bidders $n = |\mathcal{N}|$. When entering the auction game, the $i$-th bidder will receive the (interim) value $v_i$ of the target item which is independently sampled from the (ex-ante) value distribution $F_i$, and the $v_i$ is unknown to other bidders. Once a bidder acquires $v_i$, he or she would submit a bid price according to their bidding strategies $b_i$ based on $\sigma_i$. where $\sigma_i$ is formed with a distribution $P(F_i, \mathcal{M}, n, v_i)$ and $b_i$ is sampled accordingly. We use $\sigma_i(v_i)$ as a short-hand notation of $P(F_i, \mathcal{M}, n, v_i)$. Moreover, we denote $\boldsymbol{\sigma}_i$ as the set of the $i$-th bidder's bidding strategies of all value realization:

$$\boldsymbol{\sigma}_i := \{\sigma_i(v_i)|v_i \sim F_i\} \in \mathcal{D}, \tag{1}$$

where $\mathcal{D}$ is the whole strategy space. We denote the set of the strategies of bidders excluding bidder $i$ as $\boldsymbol{\sigma}_{-i} = (\boldsymbol{\sigma}_1, \cdots \boldsymbol{\sigma}_{i-1}, \boldsymbol{\sigma}_{i+1} \cdots \boldsymbol{\sigma}_n)$. In this work, we assume $\{F_1, \cdots, F_n\}$ is known to all bidders but $v_i$ is private information for bidder $i$. This Bayesian setting(Kagel & Levin, 2015) is widely adopted in auction games (Bichler et al., 2022; Banchio & Skrzypacz, 2022; Kohring et al., 2023; Duan et al., 2023a; Wang et al., 2020).

An auction mechanism $\mathcal{M} = \{\boldsymbol{p}, \boldsymbol{g}, \boldsymbol{\phi}\}$ normally includes a payment rule $\boldsymbol{p}$ and an allocation rule $\boldsymbol{g}$, with other mechanism configurations $\boldsymbol{\phi}$ such as entrance-fee or reserve price (Dütting et al., 2021). We use $p_i$ and $g_i$ to represent the payment and allocation functions for bidder $i$. The utility function $u_i$ of the $i$-th bidder is defined as follows:

$$u_i(v_i, b_i, B_{-i}|\mathcal{M}) = v_i g_i(b_i, B_{-i}) - p_i(b_i, B_{-i}),$$

where $B_{-i} = \{b_j | j \neq i\}$ denotes the opponents' bidding price. The bidder's best response $\sigma_i^*(v_i)$ solves the following optimization problem[1] formulated as:

$$\sigma_i^*(v_i) = \arg\max_{\omega_i} \mathbb{E}_{B_{-i} \sim \boldsymbol{\sigma}_{-i}} \left[ u_i(v_i, \omega_i, B_{-i}|\mathcal{M}) \right].$$

For all the bidders, a strategy profile $\boldsymbol{\sigma}^* = \{\boldsymbol{\sigma}_1^*, \cdots, \boldsymbol{\sigma}_n^*\}$ reaches (ex-ante) Bayesian Nash Equilibrium[2] if and only if every bidder's strategy is the best response of others' strategies $\boldsymbol{\sigma}_{-i}^*$, denoted as

$$\boldsymbol{u}_i(\boldsymbol{\sigma}_i, \boldsymbol{\sigma}_{-i}^*) \leq \boldsymbol{u}_i(\boldsymbol{\sigma}_i^*, \boldsymbol{\sigma}_{-i}^*) \quad \forall i \in N, \boldsymbol{\sigma}_i \in \mathcal{D}, \tag{2}$$

where the expected utility $\boldsymbol{u}_i$ is computed as

$$\boldsymbol{u}_i(\boldsymbol{\sigma}_i, \boldsymbol{\sigma}_{-i}) = \mathbb{E}_{(v_i, b_i, B_{-i}) \sim (F_i, \sigma_i, \boldsymbol{\sigma}_{-i})} \left[ u_i(v_i, b_i, B_{-i}|\mathcal{M}) \right].$$

In this work, we are interested in the solution to the approximated problem of (2) as follows:

$$\boldsymbol{u}_i(\boldsymbol{\sigma}_i, \boldsymbol{\sigma}_{-i}^*) \leq \boldsymbol{u}_i(\boldsymbol{\sigma}_i^*, \boldsymbol{\sigma}_{-i}^*) + \epsilon \quad \forall i \in N, \boldsymbol{\sigma}_i \in \mathcal{D}. \tag{3}$$

where $\epsilon$ is a positive tolerance term. The solution to (3) is known as $\epsilon$-approximate (Bayesian) Nash Equilibrium ($\epsilon$-NE or $\epsilon$-BNE) which demonstrates that the maximal utility gain i.e., the exploitability (Lanctot et al., 2017), is no more than $\epsilon$ for all players in the strategy profile $\boldsymbol{\sigma}$.

Existing learning approaches, as summarized in Table 1, have certain limitations when addressing multiple auction game scenarios. Specialized numerical solvers like backward-shooting (Wang et al., 2020; Marshall et al., 1994; Bajari, 2001; Gayle & Richard, 2008) are tailor-made for specific mechanisms. The no-regret learning approach (Feng et al., 2021; Kolumbus & Nisan, 2022; Banchio & Skrzypacz, 2022) treats each bidder as an independent agent. While each agent refines their bidding strategy for a specific mechanism, adapting to new scenarios may require intensive retraining to achieve newly converged strategies. Traditional neural network models, such as MLP, have shown promise in addressing auction games. For example, RegretNet (Dütting et al., 2021; 2024) is built for mechanism optimization problems, and deep Q-learning (Banchio & Skrzypacz, 2022; Feng et al., 2021) is applied to find equilibrium bidding strategies.

Inspired by the Transformer architecture's success (Vaswani et al., 2017) in efficiently handling flexible sequences and the efficiency of a workload-agnostic learned estimator (Wu et al., 2021), we introduce the Auctionformer. This model provides remarkable flexibility in addressing auction games and showcases superior scalability in equilibrium-solving abilities, as emphasized in Table 1.

We use the notation $X = (\mathcal{M}, n, \{F_1, \cdots, F_n\})$ to denote a game with the mechanism $\mathcal{M}$ and $n$ bidders with associated value distributions $\{F_1, \cdots, F_n\}$ . Presuming a function $\mathcal{F}$ that maps input data, such as the mechanism, bidder's value distributions, and realizations, to output strategies, this is the function the Auctionformer seeks to execute. The function's output, $\mathcal{F}(X)$ , yields the (approximate) equilibrium strategies $\hat{\boldsymbol{\sigma}}$ . For each auction game $X_j$ in auction games space $\mathcal{X}$ , the resultant bidding strategies $\hat{\boldsymbol{\sigma}}^j = (\hat{\boldsymbol{\sigma}}_1^j, \cdots, \hat{\boldsymbol{\sigma}}_{n_j}^j)$ allow each bidder in the game to adopt varying bidding strategies $\hat{\sigma}_i^j(v_i^j)$, as described in (1).

In order to further highlight the potential of the proposed model, let's consider the following examples[3] with discrete value realization from value distribution (Uniform and Gaussian distribution) and limit the bidding price upto $b_{max} = 20$.

**Example 1.** First price auction : Bidder 1's value distribution $F_1 = G(10, \frac{10}{3})$ and Bidder 2 and 3 's value distribution is $F_2 = F_3 = G(5, \frac{5}{3})$

**Example 2.** First price auction + Entry Fee=3 : Bidder 1's value distribution $F_1 = G(10, \frac{10}{3})$ , and Bidder 2-4's value distribution is $F_2 = F_3 = F_4 = G(10, \frac{5}{3})$

We benchmark no-regret solver MWU algorithm (Feng et al., 2021), MLP deep model solver MLPNet[4] and the proposed model with the underlying solution from numerical solver backward-shooting (Wang et al., 2020). From the results

[1] There may exist multiple best responses that lead to the same utility, which forms a mixed strategy (Duan et al., 2023a).

[2] In the complete information game, the definition is equivalent to the NE (Kohring et al., 2023).

[3] More example analysis can be found in appendix C.5

[4] We build a MLP network inspired by Dütting et al. (2021; 2024).The model structure and learning details can be found in appendix C.3.

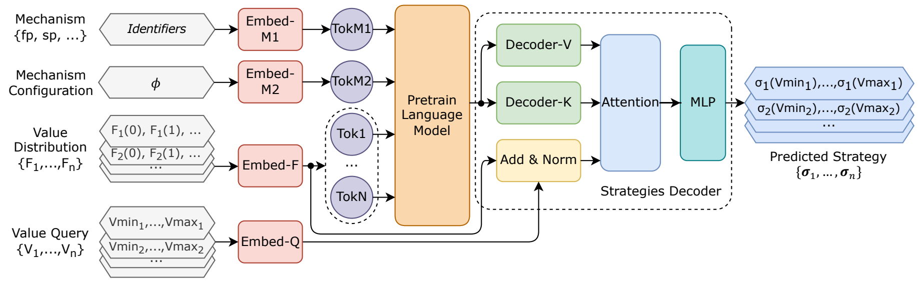


*Figure 2.* The illustration of the Auctionformer architecture. The input value distribution and mechanism names with configuration are firstly tokenized with a mechanism encoder and valuation histogram. The predicted strategies are delivered by an attention-based module that facilitates with decoded language model outputs and an extra value query vector of each bidder.

shown in Figure 7, all of the learning approaches generate a near-optimal solution that is highly close to the numerical solution. Though the MWU algorithm can solve all example games with multi-agent learning, we have to retrain agents for different settings, which is time-consuming. The pretrained MLPNet can directly output the solution of every bidder's bidding strategies within 1 special pretrained mechanism and specific bidder number (e.g., $N = 3$), which has flexibility on any value distribution but not on mechanisms and bidder number. Our proposed (pretrained) Auctionformer, however, has the ability to solve all three examples within 1 forward pass and without re-train.

# 3. Auctionformer Model

The proposed Auctionformer is illustrated in Figure 2. Details on tokenization, encoding, decoding, loss calculation, and few-shot methodology are further discussed in subsequent subsections. For the sake of brevity and to maintain the flow of discussion, we have deferred all proofs and technical derivations to the Appendix.

## 3.1. Tokenization

**Mechanism** The mechanism is characterized by one identifier (e.g., "first price") possibly with extra configurations (e.g., entrance fees, reserve prices), which are mapped to specific tokens via a predefined table and can be easily updated. In practice, identifiers and configurations are processed through an embedding layer to generate their embeddings. This leverages language models' capability to process scalable tokens, where similar tokens yield comparable results.

**Value Distribution** To incorporate bidders' value distributions into the model, we adopt a straightforward approach by inputting the probability distribution function (pdf) and applying a minimal value interpretation for value space discretization, paralleling the setup in Feng et al. (2021). Here, bidder values are allocated within a $1/H$-equally-discretized space, denoted as $\mathcal{V} = \{0, 1/H, 2/H, \cdots, 1\}$, with $H = V_{\max}$ representing the highest granularity of the value space in our learning scenarios. This allows us to transform the pdf of a bidder's value into discrete histograms and create normalized value distribution embeddings for each bidder.

**Discretization Approximation.** The usage of discrete value distributions is common and practical in the auction game field (e.g., Feng et al. 2021; Deng et al. 2022; Kolumbus & Nisan 2022), aligning with typical auctions in real life where bidders bid in monetary units (dollars or cents) rather than float numbers. For simplicity, we only consider a piecewise constant function with equal spacing to approximate the real density or mass function. The estimation error can be characterized via the following theorem.

**Theorem 3.1.** *Let $\mathcal{F}(v)$ be the cumulative distribution function (cdf) of the value distribution of a given bidder. There exists a piecewise constant function $\hat{\mathcal{F}}(v)$ with at most $n$ change points such that*

$$\sup_v \left| \hat{\mathcal{F}}(v) - \mathcal{F}(v) \right| \leq \frac{C}{\sqrt{n}}$$

*holds for some constant $C > 0$.*

In modern large language model settings, the embedding dimension is commonly on the order of $\mathcal{O}(2^{10})$. If we use the token to store the change points information of the piecewise constant function to approximate the density function, it is possible that the difference between the corresponding cdf and real underlying cdf with error up bounded $\mathcal{O}(0.03)$, which is quite small. The efficiency of our method is comparable to the existing literature on continuous games such as RegretNet (Dütting et al., 2021; 2024) and pseudo-gradient approach (Bichler et al., 2021; Kohring et al., 2023), which use piecewise constant empirical distribution as an approximation. Moreover, there are several direct ways to further reduce this error, such as linear or high-order spline approximation, polynomial approximation, and Fourier/Wavelet approximation. All those methods would generate function(s) that can be characterized with a finite number of coefficients and we could directly use them as token embedding.

We then discuss the approximation ability of the discretized bid strategy. A classic single-item auction is composed of the bidders $N$ featured by their value distributions $F$ and the auction mechanism $\mathcal{M} = \{\boldsymbol{p}, \boldsymbol{g}, \phi\}$. In our setting, the value and bid are in the discrete set $\mathcal{V} = \{0, 1/H, 2/H, \dots\}$ and the value distribution $F$ is then discretized to $F^d : \mathcal{V} \to \mathbb{R}_{\geq 0}$, formulating a discrete auction game $\mathcal{G}^d = \{N, F^d, \mathcal{V}, \mathcal{M}\}$. While the corresponding continuous auction game can be represented as $\mathcal{G}^c = \{N, F^c, \mathbb{R}_{\geq 0}, \mathcal{M}\}$, with the value and bid belonging to $\mathbb{R}_{\geq 0}$.

A strategy $\sigma_i^d(v_i^d)$ solved for the discrete game $\mathcal{G}^d$ can be mapped to a strategy of the continuous game $\mathcal{G}^c$, formulatted as: $\sigma_i^c(v_i^c) := \sigma_i^d(\arg\min_{v_i^d \in \mathcal{V}} |v_i^d - v_i^c|)$. Hence, by resolving the discrete strategy profile $\sigma^d$, we can directly approximate the continuous game with $\sigma^c$. Using the following theorem (adapted from Bichler et al. (2022)), the approximation quality can be precisely measured.

**Theorem 3.2.** *Let $\sigma^d$ be an $\epsilon$-BNE of the discrete game $\mathcal{G}^d$, then the corresponding strategy $\sigma^c$ is an $\epsilon + \mathcal{O}(\frac{1}{H})$-BNE of the continuous game $\mathcal{G}^c$.*

Intuitively, a more accurately resolved equilibrium (characterized by a lower $\epsilon$) in the discrete game, and a finer discretization grid will result in a better approximation to the continuous game.

### 3.2. Encoder with Pretrained Transformers

The encoder uses pre-trained Transformers, such as BERT (Devlin et al., 2018) and GPT (Radford et al., 2019)[5] to generate a representation of the joint bidding policy. The mechanism tokens and all bidders' value distribution tokens are fed into the trainable Transformer based language model. Considering the varied input token length, we utilize the model to generate a representation of the joint bidding policy rather than directly output every bidder's bidding strategy, which will be decoded later by the strategies decoder module.

### 3.3. Strategies Decoder

**Attention-based Design.** In Auctionformer, the output joint bidding distribution for a given game is influenced by the valuation distribution of bidders as well as the value realizations. Bidder $i$'s current value $v_i$ and distribution $F_i$ are merged via the value query module, with layer normalization applied post-merger. Utilizing the language model's embedding layer enables a shortcut in the architecture (Figure 2) for faster calculations across the value space $\mathcal{V}'$ and potential strategies via multi-head attention. This setup, processing value histograms, and mechanism encodings, allows for efficient strategy computation based on value queries and distributions to calculate equilibrium strategies. An illustration example is shown in Figure 3.

[5]The GPT in the NLP area is mentioned as a decoder-only Transformer. Here we consider GPT as a Transformer with causal masking and use it to generate the representation of a joint bidding policy. Since we have another module to further use its output to decode the final output, GPT is referred to as an encoder module.

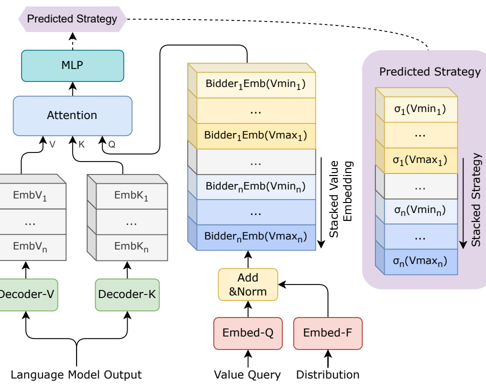


*Figure 3.* The strategies decoder module in Auctionformer, where the predicted strategy is computed through attention layer based on queried value and bidder's distribution.

The decoder ensures the privacy of bidders' value realizations by separating the query ($Q$) from the joint bidding policy's key/value ($K/V$) pairs during decoding. Cross-attention acts as a secure retrieval method to access bidding policy details without disclosing individual valuations, maintaining the confidentiality of value realizations throughout the computation. It is possible to compute a partial bidding strategy by selecting a smaller subset of value queries (e.g., $2, 3$) and setting the values of other opponents to zero. Auctionformer aims to derive each bidder's strategy based on their current value realization, independent of others' current values but conditional on their own value distribution.

Finally, we employ a Multilayer Perceptron (MLP) to generate the final bidding policy. The MLP input, of dimension $\mathbb{R}^{N \times D}$, is transformed into a final bidding strategy dimension of $\mathbb{R}^{N \times B_{max}}$, where the MLP functions as a $D \times B$ matrix coupled with a nonlinear operation, with $B$ representing the bidding range. This ensures that there is no data leakage and that each bidder's strategy is not predicated on the current value distributions of their opponents.

### 3.4. Nash Error Loss Function

Our learning objective aims to minimize $\epsilon$-BNE (Chen & Peng, 2023) and we functional Nash error (Enrich et al., 2020) to build the below loss function:

$$\begin{aligned} &\min_{\mathcal{F}} \mathcal{L}_{\text{Nash}}(\mathcal{F}, X) \\ &= \frac{1}{|\boldsymbol{X}|} \sum_{X_j \in \boldsymbol{X}} \max_{\boldsymbol{\sigma}_i \in \mathcal{D}, i \in N} \left[ \boldsymbol{u}_i^j(\boldsymbol{\sigma}_i^j, \hat{\boldsymbol{\sigma}}_{-i}^j) - \boldsymbol{u}_i^j(\hat{\boldsymbol{\sigma}}_i^j, \hat{\boldsymbol{\sigma}}_{-i}^j) \right] \end{aligned} \tag{4}$$

$$\text{s.t.} \quad [\hat{\boldsymbol{\sigma}}_1^j, \cdots, \hat{\boldsymbol{\sigma}}_N^j] = \mathcal{F}(X_j),$$

where $\boldsymbol{u}_i^j(\boldsymbol{\sigma}_i^j, \hat{\boldsymbol{\sigma}}_{-i}^j)$ is the sampled version of Eq. (3) with fixed $v_i$:

$$\boldsymbol{u}_i^j(\boldsymbol{\sigma}_i^j, \hat{\boldsymbol{\sigma}}_{-i}^j) = \mathbb{E}_{(b_i, B_{-i}) \sim (\sigma_i^j(v_i^j), \hat{\boldsymbol{\sigma}}_{-i}^j)} u_i(v_i^j, b_i, B_{-i} | \mathcal{M}^j).$$

Functional Nash error has two advantages over commonly

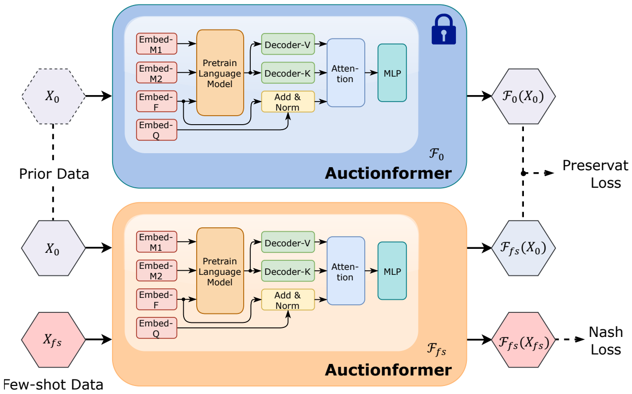


*Figure 4.* The illustration of the proposed few-shot method where the few-shot model is trained both on the Nash loss and a preservation loss between the results of the pre-trained Auctionformer model and the few-shot model on prior data.

used MSE or CE loss. First, this objective directly describes the max exploitability (Lanctot et al., 2017), which measures the distance between predicted approximate equilibrium strategy profile $\hat{\boldsymbol{\sigma}}$ and BNE status. On the other hand, no ground truth or optimal solution is required which facilitates us to address the situation when the optimal solution is hard to obtain.

One may note that maximization over all possible strategies is required in the Nash loss and we may need to solve a possible combinatorial optimization problem. We next show that for standard auction formats, such as first-price and second-price auctions, the closed-form solution exists in Proposition 3.3.

**Proposition 3.3.** *In the context of first-price and second-price auctions, the maximal utility can be directly determined by the following expression:*

$$\max_{\boldsymbol{\sigma}_i \sim \mathcal{D}} \boldsymbol{u}_i^j(\boldsymbol{\sigma}_i^j, \hat{\boldsymbol{\sigma}}_{-i}^j) = \max_{b_i} \sum_{x<b_i} (v_i^j - m^j) \cdot \Pr(\max_{k \neq i} b_k = x | \hat{\boldsymbol{\sigma}}_{-i}^j).$$

Here $m^j$ is equivalent to $b_i$ for first-price auction and $x$ for second-price auction, and $\Pr(\max_{k\neq i} b_k = x|\hat{\boldsymbol{\sigma}}_{-i}^j) = \prod_{k\neq i}\sum_{b_k\leq x}\Pr(b_k|\hat{\boldsymbol{\sigma}}_k^j) - \prod_{k\neq i}\sum_{b_k\leq x-1}\Pr(b_k|\hat{\boldsymbol{\sigma}}_k^j)$. Additionally, this computational approach can be readily adapted to auctions that include an entry fee by simply deducting the entry fee from the utility function. Consequently, by directly computing the maximal utility, the Nash loss can be efficiently calculated and applied to update the model without the need for labeled solutions.

### 3.5. Few-shot Learning

An interesting ability to use the pretrained Transformer is the generalization ability to unseen domains with very few instances, i.e., few-shot learning ability. It allows AuctionFormer to handle various auction settings using a single model, requiring only a few examples for fine-tuning without detailed payment or allocation rules. Inspired by Dreambooth (Ruiz et al., 2023), AuctionFormer uses Nash loss for unsupervised learning and calculates preservation loss with the pretrained model to address the challenge of limited ground-truth in auctions. The process for applying few-shot learning is outlined in Figure 4.

The detailed loss function is described below. The fine-tuned model, denoted as $\mathcal{F}_{fs}$, is derived from the pretrained AuctionFormer model, $\mathcal{F}_0$. The few-shot samples $X_{fs}$ are external to the training domain. Due to the lack of assurance in locating identical value distributions in the pretraining set, we reuse a subset from the pretraining set, $X_0$, most correlated with the few-shot scenarios. For instance, when addressing cases with $N = 15$ bidders, data with similar value distributions and $N = 10$ bidders are utilized.

$$\min_{\mathcal{F}} \mathcal{L}_{\text{Nash}}(\mathcal{F}_{fs}, X_{fs}) + \lambda L_p(\mathcal{F}_0(X_0), \mathcal{F}_{fs}(X_0)), \quad (5)$$

where $\lambda > 0$ and $L_p$ represent the preservation loss, ensuring the model $\mathcal{F}$ retains its original dataset performance.

The preservation loss $L_p$ can be implemented as the discrepancy between the few-shot model's predictions $\mathcal{F}_{fs}(X_0)$ and the original model's predictions $\mathcal{F}_0(X_0)$. In our study, we employ the $L_1$ norm to assess this discrepancy, formulated as: $L_p(\mathcal{F}_0(X_0), \mathcal{F}_{fs}(X_0)) = \|\mathcal{F}_0(X_0) - \mathcal{F}_{fs}(X_0)\|_1$.

## 4. Experiment

### 4.1. Implement details

**Datasets.** We evaluate Auctionformer across four standard mechanisms: First Price single item auction (FP), Second Price single item auction (SP), First Price single item auction with a minimum entry fee (FP+Ent), and Second Price single item auction with a minimum entry fee (SP+Ent). These mechanisms are translated into integer codes, as detailed in Sec 4.2. The entry fee is within the range $(0, 3]$.

We do not restrict the value distribution to be symmetric. The potential asymmetric bidder's value distribution is either drawn from the Uniform $U$ or Gaussian distribution $G$ sets, with the maximum value set at $V_{max} = 20$. For clarity, we differentiate between two distribution scenarios: distributions starting with 0 (e.g., $U[0, 8]$) are labeled as $U_0$ or $G_0$. Distributions that start non-zero (e.g., $U[3, 5]$) are denoted as $U_{-0}$ or $G_{-0}$. The total number of bidders in the training set varies between $[2, 10]$, and results for bidder numbers greater than 10 are assessed in the zero-shot and few-shot capability tests. Given these settings, we randomly generate 100,000 samples for each distribution. Every set of 500 distinct value distribution samples forms the validation set, with the remainder designated for training. The optimal

*Table 2.* The $\mathcal{L}_{\text{Nash}}$ of Auctionformer's solutions on the auction games among different value distributions, mechanisms, and asymmetric player numbers ("Ent" refers to entry fee). The $\mathcal{L}_{\text{Nash}}$ of random strategy across mechanisms is 1.59e-1 for comparison.

| | FP | SP | FP+Ent | SP+Ent |
|---|---|---|---|---|
| | **Player Number = 2∼4** | | | |
| **U+G** | 7.99e-05 | 6.19e-05 | 2.78e-04 | 9.60e-06 |
| $U_0$ | 5.15e-05 | 1.31e-05 | 6.28e-05 | 5.27e-07 |
| $G_0$ | 4.59e-05 | 6.10e-08 | 6.64e-05 | 1.15e-06 |
| $U_{-0}$ | 1.39e-04 | 2.18e-04 | 4.07e-04 | 1.46e-05 |
| $G_{-0}$ | 8.32e-05 | 1.68e-05 | 5.77e-04 | 2.22e-05 |
| | **Player Number = 5∼7** | | | |
| **U+G** | 5.79e-05 | 2.15e-04 | 3.09e-04 | 2.00e-05 |
| $U_0$ | 5.69e-05 | 3.73e-05 | 1.17e-04 | 1.27e-06 |
| $G_0$ | 4.31e-05 | 1.77e-07 | 1.23e-04 | 2.62e-06 |
| $U_{-0}$ | 9.26e-05 | 7.14e-04 | 3.69e-04 | 3.08e-05 |
| $G_{-0}$ | 3.91e-05 | 1.06e-04 | 6.28e-04 | 4.53e-05 |
| | **Player Number = 8∼10** | | | |
| **U+G** | 4.49e-05 | 3.68e-04 | 3.26e-04 | 2.55e-05 |
| $U_0$ | 5.22e-05 | 8.00e-05 | 1.41e-04 | 3.96e-06 |
| $G_0$ | 3.59e-05 | 4.11e-07 | 1.65e-04 | 6.58e-06 |
| $U_{-0}$ | 7.01e-05 | 1.18e-03 | 3.56e-04 | 3.74e-05 |
| $G_{-0}$ | 2.16e-05 | 2.15e-04 | 6.42e-04 | 5.41e-05 |

model from the validation set is retained. For testing, we create an additional $2,000$ samples for each value distribution, distinct from those in the training and validation sets, and adjust the sample mechanism codes and configurations for the various mechanisms. More detailed information is summarized in Section C.2 in Appendix.

For the few-shot capability assessment, the model undergoes updates with a learning rate of $10^{-5}$. The sizes for the training and validation datasets are set at 1,500 and 500, respectively, for every configuration. The testing dataset is consistent with the earlier configuration, providing 2,000 samples for each value distribution.

**Evaulation Metrics.** For results with an existing theoretical solution, we assess the quality of the generated solution by calculating its $L_2$ distance with the theoretical counterparts (Kohring et al., 2023; Bichler et al., 2022). In scenarios lacking a theoretical solution, we employ the metric of max exploitability. This metric gauges how one's strategy deviates when altered to the best response, essentially measuring the distance of a strategy profile $\hat{\boldsymbol{\sigma}}$ from the BNE. To evaluate Auctionformer's performance across the entire test set, we use the averaged Nash error which is also termed max exploitability, i.e., $\mathcal{L}_{\text{Nash}}$ in (4).

It's noteworthy that in benchmark comparisons, all referenced works (Bichler et al., 2022; 2021; Kohring et al., 2023) undergo validation in multiple games constrained within the $[0, 1]$ value space. Consistently, our approach also adheres to this format. We project our value space onto

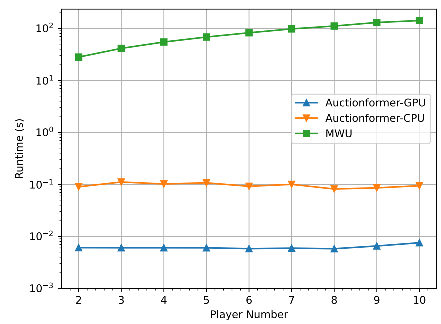


*Figure 5.* The running-time of MWU and Auctionformers in GPU and CPU. The time of MWU is greatly increased with bidder number while Auctionformer shows similar running times across numbers.

a $1/H$-evenly-discretized space (Feng et al., 2021) with $H = V_{max} = 20$.

**Implementation details.** The model is implemented in Pytorch[6]. The embedding module includes a linear layer and a 2-layer MLP with hidden dimension 512 and keeps the decoder layer's input dimension the same as the language model output dimension. We use the pretrained BERT as our default language model and train Auctionformer for all parameters (including BERT) with $10^{-4}$ learning rate[7] and batch size 1024 using Adam optimizer. During 300 training epochs, we half the learning rate every 50 epochs.

### 4.2. Main Results

**Special Case of Two Symmetric Bidders.** To our best knowledge, no existing literature demonstrates a capacity to directly solve multiple auction games with the general asymmetric bidder value distribution and different numbers of bidders at the same time. As such, we position our results against two distinct solutions: the no-regret learning approach MWU, and learning-based solvers like NPGA (Bichler et al., 2021), SODA (Bichler et al., 2022), and the more recent SM (Kohring et al., 2023). For a balanced comparison, we present results for the FP auction involving two bidders with a uniform distribution, as shown in Table 5. It's notable that the results from these cited works are derived from a highly discretized space, typically leading to a reported lower max exploitability (Kohring et al., 2023; Bichler et al., 2022).

**Comparison with No-Regret Algorithms.** No-regret algorithms are prevalent in the auction games domain, with established works confirming their convergence properties (Feng et al., 2021; Kolumbus & Nisan, 2022; Banchio & Skrzypacz, 2022; Feng et al., 2021). These algorithms are particularly prominent in classical asymmetric cases, where theoretical bidding strategies are absent, rendering no-regret

[6]Codes are available at `https://github.com/Hesse73/Auctionformer_codes`

[7]Extended results of fixed BERT are available in Appendix F.3

*Table 3.* Few-shot learning results on new 2000 sample on player number 15 and 20 and test on $U+G$ of different player number games where "Pre." denotes the preservation loss. Few-shot approach manages to transfer equilibrium solving ability on new cases while maintaining the model ability.

| | Few-shot on 15 | | Few-shot on 20 | |
|---|---|---|---|---|
| Player Number | $N \leq 10$ | $N = 15$ | $N \leq 10$ | $N = 20$ |
| Train on $N \leq 10$ | 1.56e-04 | 5.67e-02 | 1.56e-04 | 7.25e-02 |
| +Nash loss | 1.90e-02 | 2.01e-03 | 6.15e-02 | 4.01e-03 |
| + Ours (Nash+Pre.) | 4.01e-04 | 2.94e-03 | 4.96e-04 | 6.82e-03 |

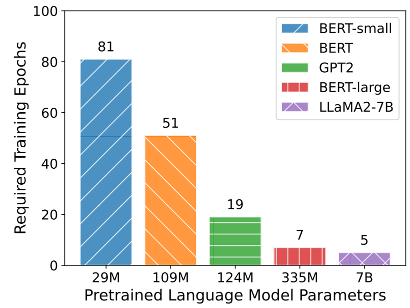


*Figure 6.* The Auctionformer training epochs of diverse pretrained language model until converged to $\mathcal{L}_{\text{Nash}} \leq 10^{-3}$ in the validation set, where a larger pretrained model requires less training epochs in the same degree of convergence.

algorithms with learning agents a go-to solution.

In contrast, we employ the BERT-based Auctionformer, a model capable of resolving auction games efficiently with a single forward pass, eliminating the need for an extensive agent learning process. Our approach not only mirrors the equilibrium bidding strategies akin to no-regret algorithms but also addresses the limitations in 0-near cases as depicted in Figure 7.

Furthermore, in simpler symmetric instances, where theoretical solutions are accessible (e.g., Uniform distribution cases showcased in Figure 12 in Appendix F.5), Auctionformer consistently yields bidding strategies that align closely with theoretical predictions and outperforms no-regret based solutions, even those involving prolonged agent learning periods.

**Scalability.** We executed experiments on asymmetric bidders and summarized the results in Table 2. Approximation errors from typical games indicate that the max exploitability of Nash solutions ranges between 1e-3 and 1e-6, depending on task complexity (Duan et al., 2023a). Despite the inherent complexity of auction games, Auctionformer has shown the ability to produce approximate equilibrium bidding strategies with max exploitability consistently around 1e-4 across different mechanisms and value distributions for bidder numbers within $[2, 10]$. These findings are based on experiments excluding all test samples from the training and validation phases, with each bidder's value distribution following a uniform model. Compared to random strategies, Auctionformer's Nash error is significantly lower (about $10^3$ times), demonstrating its scalability and effectiveness in handling diverse auction games without the error necessarily increasing with more players.

**Few-Shot Learning.** We assess Auctionformer's few-shot learning ability, aiming to extend its application to out-of-domain cases. Specifically, we evaluate its transfer capability to scenarios not encompassed within the training data, such as auctions involving 15 and 20 asymmetric bidders. Aggregate results across all mechanisms and value distributions are showcased in Table 3, where the equilibrium solver's performance is examined both pre and post few-shot learning. As observed from Table 3, while direct fine-tuning on novel auction games can indeed enhance performance, as indicated by the Nash loss, it may compromise Auctionformer's competence in its pre-trained domain. Contrastingly, our proposed few-shot method adeptly navigates novel game domains utilizing limited data, all the while retaining previously acquired knowledge. More detailed few-shot results of mechanism transformation are summarized in Appendix F.6.

### 4.3. Ablation Studies

**Running-time with Increasing Bidder Number.** In the classical equilibrium solvers, the running time is increasing greatly or even exponentially (Bichler et al., 2022) in the growth of the bidder number. We conduct similar experiments from Bichler et al. (2022) to validate whether Auctionformer will face such a limitation. The experiments are run on Intel(R) Xeon(R) Platinum 8163 CPU(s) @ 2.50GHz or a single Nvidia Tesla V100-SXM2 with 16GB memory. From Figure 5, the MWU algorithm runs on CPU and consumes significantly more time on 10 bidders compared to 2 bidders, while Auctionformer performs the same magnitude running time on both CPU and GPU. As a result, our proposed Auctionformer is able to solve auction games with similar time complexity regardless of the magnitude of the bidder number.

**Language model structure.** We also explore the influence of the base language model structure in solving auction games, where BERT-small (Bhargava et al., 2021), BERT, BERT-large, GPT2 (Radford et al., 2019) and LLaMA2-7B (Touvron et al., 2023) are considered. The model parameters along with the training epoch to the convergences are plotted in Figure 6, and a larger model turns to require fewer epochs in the same degree of convergence, which proves the great potential of the large language model in solving equilibria.

## 5. Conclusions

In this study, we introduce Auctionformer, a unified method tailored to address general auction games. We propose a novel tokenization procedure that makes hyper-parameters agnostic to bidder count and auction mechanisms. With its versatile valuation histogram, Auctionformer makes remarkable scalability. We demonstrate Auctionformer can efficiently handle various auction scenarios including those with diverse bidder counts, mechanisms, and value distributions without the need for retraining. Empirical results indicate Auctionformer's superior performance over traditional no-regret learning algorithms, particularly in cases involving a multitude of players and intricate asymmetric bidder value distributions. Notably, its computational efficiency remains constant, irrespective of bidder count, and its adaptability is further validated through few-shot tests.

## Acknowledgements

This research is supported by the National Science and Technology Major Project (2023ZD0121102), National Natural Science Foundation of China (92270114). This work is also supported by Alibaba Group through Alibaba Innovative Research (AIR) Program.

## Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

## References

Arora, S., Hazan, E., and Kale, S. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(6):121–164, 2012. doi: 10.4086/toc.2012.v008a006.

Athey, S. and Haile, P. A. Identification of standard auction models. *Econometrica*, 70(6):2107–2140, 2002. doi: https://doi.org/10.1111/j.1468-0262.2002.00435.x.

Auer, P., Cesa-Bianchi, N., Freund, Y., and Schapire, R. Gambling in a rigged casino: The adversarial multi-armed bandit problem. In *Proceedings of IEEE 36th Annual Foundations of Computer Science*, pp. 322–331, 1995. doi: 10.1109/SFCS.1995.492488.

Badanidiyuru, A., Feng, Z., and Guruganesh, G. Learning to bid in contextual first price auctions, 2021.

Bajari, P. Comparing competition and collusion: a numerical approach. *Economic Theory*, 18:187–205, 2001.

Banchio, M. and Skrzypacz, A. Artificial intelligence and auction design. In *Proceedings of the 23rd ACM Conference on Economics and Computation*, EC '22, pp. 30–31, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450391504. doi: 10.1145/3490486.3538244.

Bhargava, P., Drozd, A., and Rogers, A. Generalization in nli: Ways (not) to go beyond simple heuristics, 2021.

Bichler, M., Fichtl, M., Heidekrüger, S., Kohring, N., and Sutterer, P. Learning equilibria in symmetric auction games using artificial neural networks. *Nature machine intelligence*, 3(8):687–695, 2021.

Bichler, M., Fichtl, M., and Oberlechner, M. Computing bayes nash equilibrium strategies in auction games via simultaneous online dual averaging. *arXiv preprint arXiv:2208.02036*, 2022.

Bigler, J. Rolling out first price auctions to google ad manager partners. https://blog.google/products/admanager/rolling-out-first-price-auctions-google-ad-manager-partners, 2008. Accessed 2020.01.27.

Cai, Y. and Papadimitriou, C. Simultaneous bayesian auctions and computational complexity. In *Proceedings of the Fifteenth ACM Conference on Economics and Computation*, EC '14, pp. 895–910, New York, NY, USA, 2014. Association for Computing Machinery. ISBN 9781450325653. doi: 10.1145/2600057.2602877.

Cesa-Bianchi, N. and Lugosi, G. *Prediction, learning, and games*. Cambridge university press, 2006.

Chen, L., Iellamo, S., Coupechoux, M., and Godlewski, P. An auction framework for spectrum allocation with interference constraint in cognitive radio networks. In *2010 Proceedings IEEE INFOCOM*, pp. 1–9. IEEE, 2010.

Chen, X. and Peng, B. Complexity of equilibria in first-price auctions under general tie-breaking rules. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, STOC 2023, pp. 698–709, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9781450399135. doi: 10.1145/3564246.3585195.

Chen, X., Deng, X., and Teng, S.-H. Settling the complexity of computing two-player nash equilibria. *J. ACM*, 56 (3), may 2009. ISSN 0004-5411. doi: 10.1145/1516512.1516516.

Chen, Y. Programmatic advertising is preparing for the first-price auction era. https://digiday.com/marketing/programmatic-advertising-readying-first-price-auction-era/, 2017. Accessed 2020.1.29.

Deng, X., Hu, X., Lin, T., and Zheng, W. Nash convergence of mean-based learning algorithms in first price auctions. In *Proceedings of the ACM Web Conference 2022*, WWW

’22, pp. 141–150, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450390965. doi: 10.1145/3485447.3512059.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021.

Duan, Z., Huang, W., Zhang, D., Du, Y., Wang, J., Yang, Y., and Deng, X. Is nash equilibrium approximator learnable? In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*, AAMAS ’23, pp. 233–241, Richland, SC, 2023a. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450394321.

Duan, Z., Tang, J., Yin, Y., Feng, Z., Yan, X., Zaheer, M., and Deng, X. A context-integrated transformer-based neural network for auction design, 2023b.

Dütting, P., Feng, Z., Narasimhan, H., Parkes, D. C., and Ravindranath, S. S. Optimal auctions through deep learning. *Commun. ACM*, 64(8):109–116, jul 2021. ISSN 0001-0782. doi: 10.1145/3470442.

Dütting, P., Feng, Z., Narasimhan, H., Parkes, D. C., and Ravindranath, S. S. Optimal auctions through deep learning: Advances in differentiable economics. *J. ACM*, 71 (1), feb 2024. ISSN 0004-5411. doi: 10.1145/3630749. URL https://doi.org/10.1145/3630749.

Enrich, C. D., Jelassi, S., Domingo-Enrich, C., Scieur, D., Mensch, A., and Bruna, J. Extragradient with player sampling for faster nash equilibrium finding, 2020.

Feng, Z., Guruganesh, G., Liaw, C., Mehta, A., and Sethi, A. Convergence analysis of no-regret bidding algorithms in repeated auctions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 5399–5406, 2021.

Filos-Ratsikas, A., Giannakopoulos, Y., Hollender, A., Lazos, P., and Poças, D. On the complexity of equilibrium computation in first-price auctions. In *Proceedings of the 22nd ACM Conference on Economics and Computation*, EC ’21, pp. 454–476, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450385541. doi: 10.1145/3465456.3467627.

Gayle, W.-R. and Richard, J. F. Numerical solutions of asymmetric, first-price, independent private values auctions. *Computational Economics*, 32:245–278, 2008.

Guo, Y., Shi, H., Kumar, A., Grauman, K., Rosing, T., and Feris, R. Spottune: transfer learning through adaptive fine-tuning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4805–4814, 2019.

Hartline, J., Syrgkanis, V., and Tardos, E. No-regret learning in bayesian games. In Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

Interactive Advertising Bureau and PwC. Internet advertising revenue report. https://www.iab.com/wp-content/uploads/2023/04/IAB_PwC_Internet_Advertising_Revenue_Report_2022.pdf, April 2023.

Ivanov, D., Safiulin, I., Filippov, I., and Balabaeva, K. Optimal-er auctions through attention. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=Xa1T165JEhB.

Jackson, M. O. and Swinkels, J. M. Existence of equilibrium in single and double private value auctions1. *Econometrica*, 73(1), 2005.

Kagel, J. H. and Levin, D. *Auctions: A Survey of Experimental Research*. 2015. ISBN 0691139997;9780691139999;.

Kalai, A. and Vempala, S. Efficient algorithms for online decision problems. *Journal of Computer and System Sciences*, 71(3):291–307, 2005. ISSN 0022-0000. doi: https://doi.org/10.1016/j.jcss.2004.10.016. Learning Theory 2003.

Kohring, N., Pieroth, F. R., and Bichler, M. Enabling first-order gradient-based learning for equilibrium computation in markets. In *International Conference on Machine Learning*, pp. 17327–17342. PMLR, 2023.

Kolumbus, Y. and Nisan, N. Auctions between regret-minimizing agents. In *Proceedings of the ACM Web Conference 2022*, WWW ’22, pp. 100–111, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450390965. doi: 10.1145/3485447.3512055.

Krishna, V. *Auction theory*. Academic press, 2009.

Lanctot, M., Zambaldi, V., Gruslys, A., Lazaridou, A., Tuyls, K., Perolat, J., Silver, D., and Graepel, T. A unified game-theoretic approach to multiagent reinforcement learning. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Lei, Q., Ganesh Nagarajan, S., Panageas, I., and wang, x. Last iterate convergence in no-regret learning: constrained min-max optimization for convex-concave landscapes. In Banerjee, A. and Fukumizu, K. (eds.), *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pp. 1441–1449. PMLR, 13–15 Apr 2021.

Louargand, M. A. and McDaniel, J. Price efficiency in the art auction market. *Journal of Cultural Economics*, pp. 53–65, 1991.

Marris, L., Gemp, I., Anthony, T., Tacchetti, A., Liu, S., and Tuyls, K. Turbocharging solution concepts: Solving nes, ces and cces with neural equilibrium solvers. *Advances in Neural Information Processing Systems*, 35:5586–5600, 2022.

Marshall, R. C., Meurer, M. J., Richard, J.-F., and Stromquist, W. Numerical analysis of asymmetric first price auctions. *Games and Economic behavior*, 7(2): 193–220, 1994.

Mayer, C. J. Assessing the performance of real estate auctions. *Real Estate Economics*, 26(1):41–66, 1998.

Milgrom, P. R. and Weber, R. J. A theory of auctions and competitive bidding. *Econometrica*, 50(5):1089–1122, 1982. ISSN 00129682, 14680262.

Motiian, S., Jones, Q., Iranmanesh, S., and Doretto, G. Few-shot adversarial domain adaptation. *Advances in neural information processing systems*, 30, 2017.

Myerson, R. B. Optimal auction design. *Mathematics of Operations Research*, 6(1):58–73, 1981. doi: 10.1287/moor.6.1.58.

Nedelec, T., Baudet, J., Perchet, V., and El Karoui, N. Adversarial learning in revenue-maximizing auctions. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '21, pp. 955–963, Richland, SC, 2021. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450383073.

Nikaidô, H. and Isoda, K. Note on non-cooperative convex games. 1955.

OpenAI. Gpt-4 technical report, 2023.

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

Reny, P. J. and Zamir, S. On the existence of pure strategy monotone equilibria in asymmetric first-price auctions. *Econometrica*, 72(4):1105–1125, 2004. doi: https://doi.org/10.1111/j.1468-0262.2004.00527.x.

Ruiz, N., Li, Y., Jampani, V., Pritch, Y., Rubinstein, M., and Aberman, K. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22500–22510, 2023.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Ferrer, C. C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I., Korenev, A., Koura, P. S., Lachaux, M.-A., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E. M., Subramanian, R., Tan, X. E., Tang, B., Taylor, R., Williams, A., Kuan, J. X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M., Narang, S., Rodriguez, A., Stojnic, R., Edunov, S., and Scialom, T. Llama 2: Open foundation and fine-tuned chat models, 2023.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Vickrey and William. Counterspeculation, auctions, and competitive sealed tenders. *Journal of Finance*, 16(1), 1961.

Wang, Z., Shen, W., and Zuo, S. Bayesian nash equilibrium in first-price auction with discrete value distributions. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '20, pp. 1458–1466, Richland, SC, 2020. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450375184.

Wu, R., Ding, B., Chu, X., Wei, Z., Dai, X., Guan, T., and Zhou, J. Learning to be a statistician: Learned estimator for number of distinct values. *Proc. VLDB Endow.*, 15 (2):272–284, oct 2021. ISSN 2150-8097. doi: 10.14778/3489496.3489508.

# A. Related Work

Our work lies in a wide field of research including equilibrium solvers, learning in auction games as well as deep language models, among which we point out a few of the works.

**Equilibrium solver:** Solving Nash Equilibrium (NE) is a challenging task, which is well-known for its high computational complexity. Even finding NE in a 2-player game is considered PPAD-complete (Chen et al., 2009). In auction games, solving NE or Bayesian Nash Equilibrium (BNE) remains PPAD-complete (Filos-Ratsikas et al., 2021; Chen & Peng, 2023) or even PP-hard in certain scenarios (Cai & Papadimitriou, 2014). But for BNE, there has been promising progress on numerical analysis for solving equilibrium in some special auction cases. For asymmetric auction, the backward-shooting solver (Marshall et al., 1994) is proposed to compute BNE, which is widely adopted by subsequent works with enhanced precision (Bajari, 2001; Gayle & Richard, 2008). More recently, Wang et al. (2020) extended it to the discrete setting of the first-price auction.

**Learning in auction:** Learning-based auction algorithms have also shown great potential in equilibrium solving. No-regret learning, as discussed in Cesa-Bianchi & Lugosi (2006), has established its convergence towards a looser approximation of NE (Hartline et al., 2015). Feng et al. (2021) analyzes the convergence of no-regret algorithms to BNE in some auction game formats including second price or two-bidder first price auction with a uniform prior, but such convergence may fail in some cases such as the first-price auction with fixed values (Deng et al., 2022). Typical no-regret algorithms include Follow-The-Perturbed-Leader (Kalai & Vempala, 2005), Hedge (Auer et al., 1995), and Multiplicative Weights Update (MWU) (Arora et al., 2012), which has been extended in repeated auctions with reinforcement learning agents (Feng et al., 2021; Kolumbus & Nisan, 2022; Banchio & Skrzypacz, 2022). Similar to multi-agent reinforcement learning (MARL), other gradient-based methods such as NPGA (Bichler et al., 2021) and SM (Kohring et al., 2023) are both capable of approximating BNE in various types of auctions, and SODA (Bichler et al., 2022) utilizes dual averaging algorithms to solve discrete auction games with approximate solutions.

**Deep model and few-shot** Previous algorithms focus on solving individual games, and recent advancements have been made to solve games in batches (Duan et al., 2023a; Marris et al., 2022) with data-driven algorithms. In the auction game field, some works use data-driven algorithms with deep model (Dütting et al., 2021; 2024) to tackle the mechanism design problems and the latest work replaces the MLP with Transformer (Duan et al., 2023b) in mechanism design. Yet, no existing work turns to solving multiple auction game equilibria, especially on different mechanisms, with deep models in batches.

The latest huge success on large language models such as GPT-4 (OpenAI, 2023) enlightens the huge potential in natural language model. The notable progress in deep language model is the convention of Transformer (Vaswani et al., 2017), which has been implemented to BERT (Devlin et al., 2018), GPT (Radford et al., 2019; OpenAI, 2023) and even computer vision models (e.g., Dosovitskiy et al. (2021)). The common utilization of these pre-trained models in one specific domain can be transferred to other domains through fine-tuning or few-shot learning (Guo et al., 2019; Motiian et al., 2017), where much fewer samples are accessible in the few-shot tasks.

In this work, we apply a pretrained BERT model as a starting point and prove the great potential to combine deep language model in the auction game-solving field, which can be extended to other auction game mechanisms with the proposed few-shot methodology as well as the utilization of large language model (LLM) in the future work.

**Comparison with Auction Design:** Auction design, while a related field, primarily seeks to create auction mechanisms that optimize objectives like revenue or social welfare. Pioneering this domain, Dütting et al. (2021; 2024) introduced RegretNet, a neural network model designed to predict the optimal auction mechanism (i.e. allocation and payment strategies) for maximizing seller revenue. Subsequent studies, such as those by Ivanov et al. (2022); Duan et al. (2023b), have employed attention-based or transformer architectures in auction design, enhancing model input flexibility and generalization capability. However, it's crucial to differentiate this research area from equilibrium solving, the focal point of our study. Our research aims to calculate various equilibrium profiles based on provided auction mechanisms and bidder specifics. In contrast, auction design tries to derive optimal mechanisms to achieve specific goals. Essentially, our work is committed to solving various predetermined games, while auction design is concerned with designing better rules for the game.

# B. Illustration on solution scheme for Example 1 and 2

See in Fig.7.

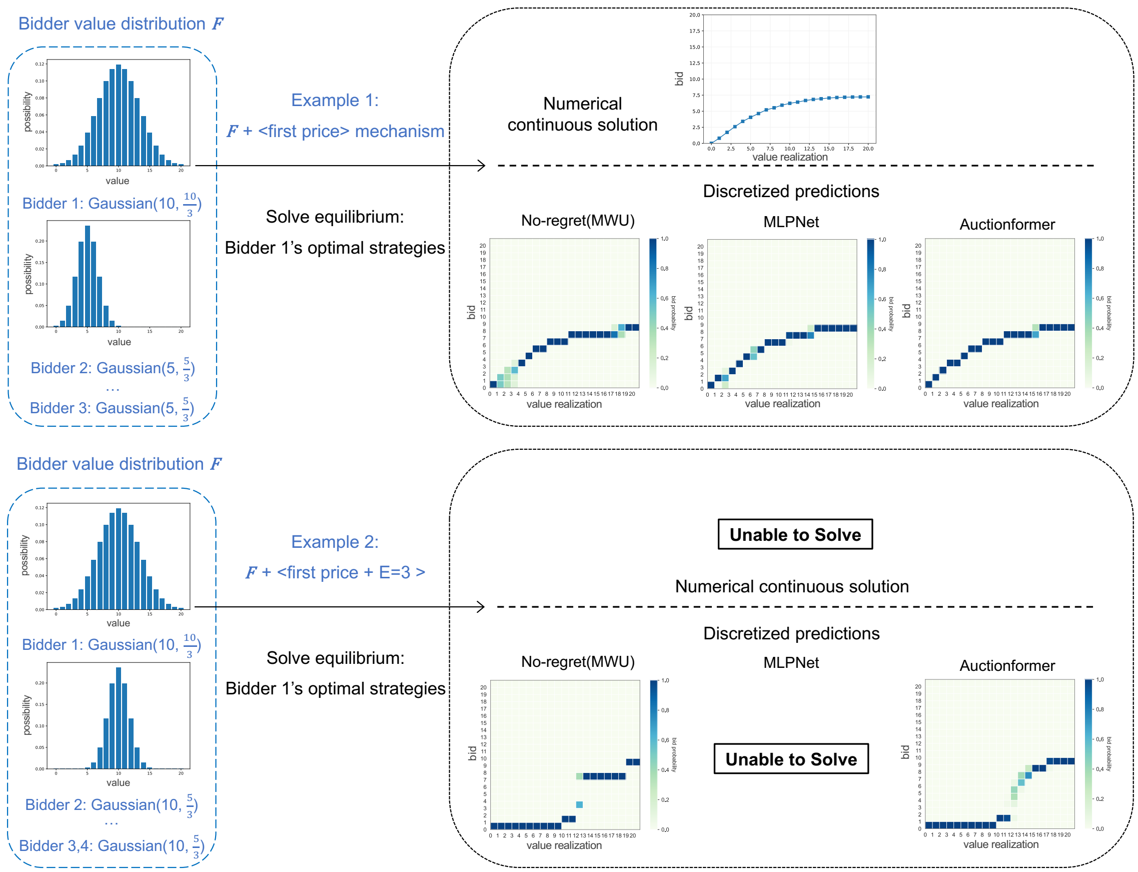


*Figure 7.* The solved bidding strategies of Example 1 & 2 with asymmetric bidders from numerical solver and learning-based algorithms. In Example 1, all algorithms are able to deliver the solution, while in Example 2 are not. All three algorithms deliver a near-optimal solution and MWU disappoints near zero value in Example 1. (Test codes are provided in Appendix.C.1)

# C. Example codes and details

## C.1. Codes

The example codes are attached in the `https://github.com/Hesse73/Auctionformer_example`, which generates examples and validates auction game results with different algorithms. The solved bidding strategies can be derived by running a simple command, where you can easily move to other auction game settings. The trained Auctionformer model checkpoint as well as the MLPNet model checkpoint are both included, and the training codes of MLPNet are also provided. Noted that training codes of Auctionformer have already been provided in the link of Sec.4.1.

## C.2. Dataset Details

Given a specific auction game, characterized by its mechanism and valuation distribution, the Auctionformer model can directly predict a solution. Subsequently, the predicted solution is utilized to guide parameter updates through the Nash error loss function. Therefore, no explicit label is required during the whole training process and we construct the dataset with the auction game information (i.e. auction mechanism and value distribution) without theoretical solutions.

To elaborate, we generate different value realization ranges for each asymmetric bidder, denoted as $[a, b]$, where $a < b < V_{\max}$, to represent their respective value distributions. The game mechanism can be simply iterated during the training process with its mechanism names. The detailed payment rule and allocation rule are applied during the utility function computation.

For the uniform distribution dataset, we can simply generate the value realization via $U[a, b]$, while for the Gaussian distribution dataset, we use a discretized version of Gaussian distribution:

$$P(x) = \begin{cases} \Phi(g(x+0.5)) & ,x = a \\ \Phi(g(x+0.5)) - \Phi(g(x-0.5)) & ,a < x < b \\ 1 - \Phi(g(x-0.5)) & ,x = b \end{cases}$$

Where $\Phi$ is the cumulative density function of standard Gaussian distribution, and $g(x)$ is defined as:

$$g(x) = \frac{6}{b-1} \cdot (x - \frac{a+b}{2}),$$

which can transform a Gaussian (normal) distribution $\mathcal{N}(\frac{a+b}{2}, \frac{b-a}{2})$ to $\mathcal{N}(0, 1)$.

In our main results, we set the value range in $[a, b]$ for the Gaussian distribution and Uniform distribution, where the minimal value realization $a$ and maximal value realization $b$ is selected from $[0, V_{max}]$ and $V_{max}$ is set to 20. We trained our model with 4 datasets: $U_0$, $G_0$, $U_{-0}$, and $G_{-0}$ introduced in Section 4.1. Each dataset contains 100,000 samples. For the fine-tuning experiments, we cut datasets into only 2,000 samples of each distribution.

### C.3. MLPNet

Aside from Auctionformer, we also develop the MLPNet to train in scenarios where the number of bidders is fixed. The learning procedures are described as follows:

**Input:** Each input sample $X_j = (\mathcal{M}^j, F^j) \in \boldsymbol{X}$ from the dataset $\boldsymbol{X}$ covers the mechanism of the auction game $\mathcal{M}^j$ and each bidder $i \in \{1..N\}$'s valuation distribution $F_i^j$ ranged in $[{V_{min}}_i^j, {V_{max}}_i^j]$, where $\forall i, j, 0 \le {V_{min}}_i^j \le {V_{max}}_i^j \le V_{max}$ and $F^j = (F_1^j, F_2^j, \ldots, F_N^j)$.

**Output:** The output of the model $\mathcal{F}$ is the approximate equilibrium strategies $\hat{\boldsymbol{\sigma}}^j = (\hat{\boldsymbol{\sigma}}_1^j, \ldots, \hat{\boldsymbol{\sigma}}_{n_j}^j)$ for all bidders in the game, from which each bidder can adopt different bidding strategies $\hat{\sigma}_i^j(v_i^j)$ according to their valuation as defined in Eq. (1).

**Objective:** The learning objective aims to minimize the Nash loss as described in Eq. (4).

From the perspective of the model architecture, the MLPNet is much simpler compared to the Auctionformer. The illustration of MLPNet architecture is described in Figure 8, where MLPNet only contains 2 embedding layers and one MLP module that transforms the concatenated game information (i.e. embedded mechanism information and value distribution) to the predicted strategy. The mechanism identifiers and the mechanism configuration are utilized to distinguish different auction games and the MLPNet may fail to converge on auction games with diverse mechanisms. Therefore, in the latter section, we train MLPNet on one mechanism and leave the mixed train in the future works. The input/output dimension is strictly determined by the number of bidders. For example, with max value $V_{max} = 20$ and bidder number $N = 3$, the input size of the MLP will be $2 \cdot embed + V_{max} \cdot N$ (where $embed$ is the embedding dimension). With a fixed input dimension, it cannot be adapted to games with a different number of players $N'$.

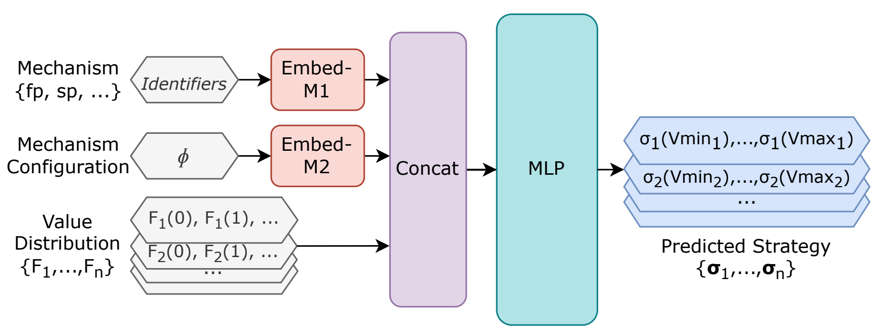


*Figure 8.* The illustration of MLPNet architecture, where the concatenated game information is transformed by 2 embedding layers, and an MLP module is applied to predict bidding strategies with a fixed bidder number.

The MLPNet is trained using three asymmetric bidder scenarios ($N = 3$) across uniform and Gaussian distributions. We generated a comprehensive set of discrete distributions where the minimal value realization is zero and the maximal value realization varies from 1 to 20. This process resulted in a dataset of 8,000 distinct auction games for the 3-player configuration. Additionally, we created another set of 8,000 distributions where the minimal value realization exceeds zero. The datasets were randomly shuffled and partitioned into a training set with 7,500 games for each type of distribution, and a validation set composed of the remaining games. For simplicity, we only consider the first price mechanism when training this MLPNet model.

We set the embedding size $embed = 128$ and the MLP module is constructed of 4 hidden layers, each with 1024 nodes. ReLU activation and batch normalization are applied after each layer. The training process was configured with a learning rate of $1 \times 10^{-4}$, a batch size of 1,024, and was run over 300 epochs and we halved the learning rate every 50 epochs.

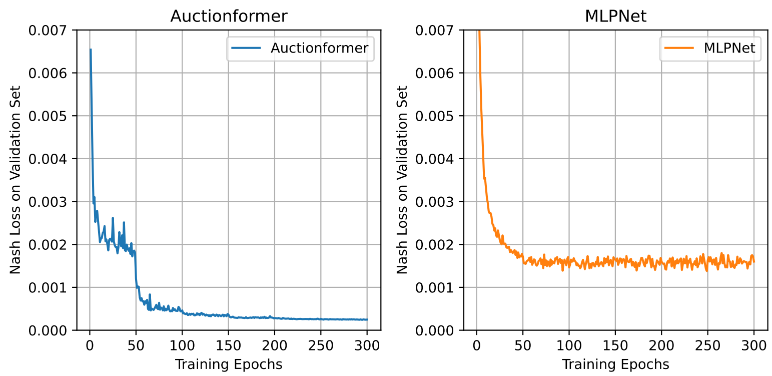


*Figure 9.* The Nash loss on validation set during the whole training process of Auctionformer (on hybrid dataset described in Sec.4.1) and MLPNet (on first price auction games with 3 bidders).

### C.4. MWU and Auctionformer

**MWU** Multiplicative Weights Update(MWU) is a classical no-regret learning algorithm for solving auction games. We implement the MWU algorithm based onArora et al. (2012), where $w_i^t(b,v)$ is regarded as the weight of player $i$'s bidding $b$ at value $v$ and round $t$. In each round $t$, every player receives a private value $v_i^t$ and bids $b_i^t$ by sampling according to the weight $\Pr(b_i^t = b) \propto w_i^t(b, v_i^t)$, and then update their weights based on the auction outcome. The update rule is described below:

$$w_i^{t+1}(b,v) = \begin{cases} w_i^t(b,v) \cdot (1-\epsilon)^{-u_i^t(b,v)/\rho} & ,u_i^t(b,v) < 0 \\ w_i^t(b,v) \cdot (1+\epsilon)^{u_i^t(b,v)/\rho} & ,u_i^t(b,v) \geq 0 \end{cases}$$

where $u_i^t(b,v)$ is the utility of player $i$ at round $t$. In our implements, we set $\epsilon = 0.1$ and $\rho = 1$, and the iteration rounds=100,000.

**Auctionformer** The Auctionformer model is trained on the 4 mechanisms with asymmetric bidders via Nash loss, whose base BERT model is adopted with the first 6 layers, and performances have been depicted in Table 2. We also attach the whole training processing in Figure 9 along with the MLPNet, where the Nash loss is reported on the validation set.

In our implementation, the pretraiend BERT model[8] used as Auctionformer's backbone, along with other pretrained language models (i.e., BERT-small[9], BERT-large[10], GPT2[11] and LLaMA2-7B[12]) depicted in Fig. 6, are employed from the HuggingFace Model Hub. All of these models are publicly available through the Hugging Face community.

Furthermore, the training process of the Auctionformer model leverages the hybrid dataset outlined in Section 4.1. This dataset encompasses a range of auction mechanisms, such as first price and second price auctions, along with varying entrance fees. To effectively integrate these diverse mechanisms into the Auctionformer model, as detailed in Section 3.1, we assign unique identifiers and configurations, denoted as $\phi$, to each mechanism. These identifiers and configurations are

[8]https://huggingface.co/bert-base-uncased
[9]https://huggingface.co/prajjwal1/bert-small
[10]https://huggingface.co/bert-large-uncased
[11]https://huggingface.co/gpt2
[12]https://huggingface.co/meta-llama

then transformed into token embeddings, which are subsequently inputted into the transformer model. The specifics of this mapping process are illustrated in Table 4.

*Table 4.* The mapped values of different mechanisms

| Mechanism | identifier | configuration $\phi$ |
|---|---|---|
| First price | 0 | $\phi = 0$ |
| Second price | 1 | $\phi = 0$ |
| First price with entrance fee $e$ | 2 | $\phi = e$ |
| Second price with entrance fee $e$ | 3 | $\phi = e$ |

### C.5. The Analysis to 3 Example Results

In the previous Sec.2, we use the following examples with various mechanisms and distributions to illustrate different auction games and to compare different methods:

**Example 1.** First price auction : Bidder 1's value distribution $F_1 = G(10, \frac{10}{3})$ and Bidder 2 and 3 's value distribution is $F_2 = F_3 = G(5, \frac{5}{3})$

**Example 2.** First price auction + Entry Fee=3 : Bidder 1's value distribution $F_1 = G(10, \frac{10}{3})$ , and Bidder 2-4's value distribution is $F_2 = F_3 = F_4 = G(10, \frac{5}{3})$

**Example 3.** First price auction : Bidder 1's value distribution is $F_1 = U[0, 20]$ and Bidder 2,3's value distribution is $F_2 = U[0, 10]$

We employed several solution methods, including the numerical BNE solver proposed in Wang et al. (2020), the MWU algorithm (Arora et al., 2012), MLPNet, and Auctionformer, to solve these examples. Note that in example 2, the game involves 4 players, making the application of the 3-player MLPNet infeasible. Additionally, the BNE solver (Wang et al., 2020) is designed for solving first-price auctions without an entry fee, rendering it inadequate for this problem. Consequently, we can only utilize the MWU algorithm (Arora et al., 2012) and our model, Auctionformer, to solve Example 3. The results are depicted in Figure. 10.

In example 1, it's noteworthy that the heatmap of MWU's prediction strategy becomes vague in the lower value range ($v \in [1, 3]$). This is due to the bidder's winning probability being considerably small when their value is low, and this phenomenon gets exacerbated when the rival bidder has a low likelihood of a small value. Consequently, the algorithm struggles to receive sufficient positive feedback to update the weights of different bidding values. As a result, the bidding strategy becomes random and remains unchanged from its initial state. On the other hand, the predicted strategy by MLPNet appears counterintuitive, as there is a discernible possibility of placing a bid of 0 when the value realization is 2. This behavior not only seems irrational but also violates the monotonic bidding rule, a common assumption in auction theory (Myerson, 1981; Reny & Zamir, 2004; Milgrom & Weber, 1982), as it bids 1 for a value of 1 but could bid 0 for a value of 2. In contrast, Auctionformer's prediction maintains alignment with the continuous solution well.

In example 2, the MWU algorithm's solution manifests as a step function. Conversely, Auctionformer's solution adopts mixed strategies and transitions smoothly as the value increases, which provides a better approximation of the equilibrium strategies.

In example 3, although the outlines of the solutions for all four approaches are consistent—ranging from bids of 0 to 8 with values increasing to 20, the solution obtained through MWU suddenly jumps from bid 6 at value 11, and to bid 8 at value 12. In contrast, the results predicted by Auctionformer and MLPNet show better continuity, aligning well with the numerical solution. However, if the bidder number is larger than 3, MLPNet cannot solve these auction games and has to reconstruct its MLP layer dimensions with re-training.

Therefore, from these 3 simple examples, we show the scalability and ability to solve auction equilibria by our proposed Auctionformer.

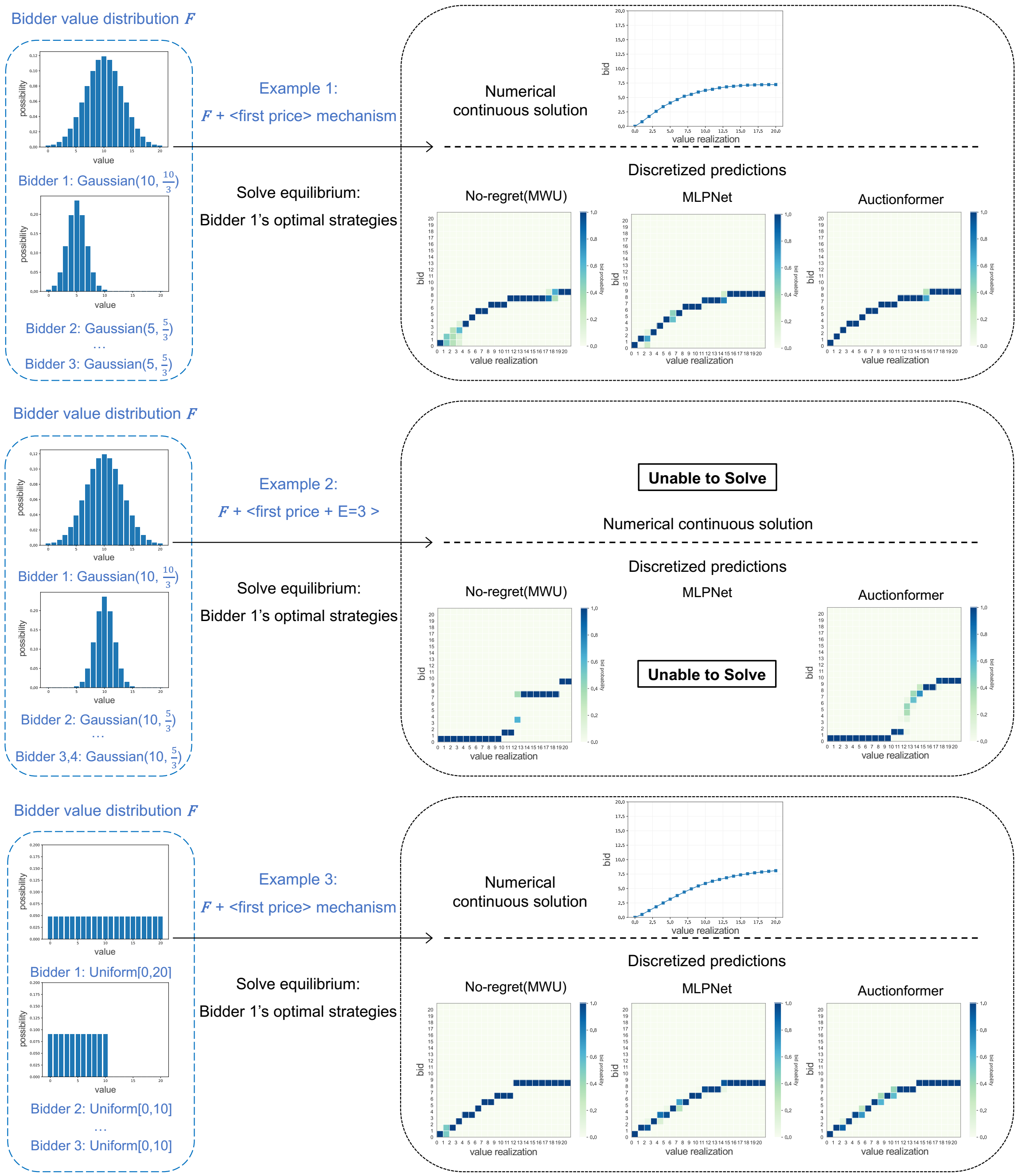


*Figure 10.* The 3 example results are displayed where the left part is the input value distributions of bidders and the right part is the corresponding results of each methodology. Given that the bidding strategy of the BNE-solver (Wang et al., 2020) is continuous, we present the mean bid value for ease of comparison. On the other hand, for the remaining three methods, we illustrate the heatmap demonstrating the bidding strategy at every value realization.

### C.6. Comparison with Other Learning-based Algorithms

As observed in Table 5, Auctionformer outperforms MWU, SODA-32, and SM-1024 in a classical two-bidder auction game scenario. Although NPGA and SM indicate a lower $L_2$ distance as reported in Kohring et al. (2023), necessitating $2^{18}$ valuation samples, Auctionformer emerges superior in efficiency, clocking a faster time consumption of $0.006$s while offering strategy approximation results comparable to NPGA.

*Table 5.* The results for FP of 2 bidders in $U[0,1]$. Both $L_2$ and Time spent $t$ (both smaller is better) are validated by our implement or the reported results extracted from SODA(Bichler et al., 2022) and SM(Kohring et al., 2023). The time spent by the Auctionformer is the inference time. MWU runs 100,000 iterations until converge (Kolumbus & Nisan, 2022) and NPGA/SM runs 2000 iterations .

| | Random | MWU-32 | MWU-64 | SODA-32 | SM-1024 | NPGA-$2^{18}$ | SM-$2^{18}$ | Auctionformer-20 |
|---|---|---|---|---|---|---|---|---|
| $L_2$ | 0.418 | 0.021 | 0.019 | 0.018 | 0.018 | 0.011 | 0.005 | 0.017 |
| Time spent $t$ | - | 27.97s | 31.22s | 0.29s | - | ≈310s | ≈18s | 0.006s |

## D. Extra illustration on Strategies Decoder

We also append more details on the strategies decoder with extra illustrations. As described in Sec.3.3, the vector $Q$ represents the value query embedding vector with dimensions $\mathbb{R}^{N\times D}$. Both vectors $K$ and $V$ are outputs of the model, where $K = V \in \mathbb{R}^{N\times D}$. Here, $N$ symbolizes the maximum number of current bidders, and $D$ is the dimensionality related to the embedding layer and model feature.

When leveraging the multi-head attention module to decode bidders' strategies, we must address an important question: Is there any risk of data (or value) leakage? In essence, we need to ensure that during the entire attention-based decoding computation, no bidder can discern the value realizations of others. We firstly focus on the traditional attention computation, where the multiplication $QK^\top$ is executed on the bidders' query tensor $Q$, and we break it down into independent computations for each of the $N$ bidders:

$$\begin{aligned} QK^T =& [Q_1,\ldots,Q_N]^T \cdot [K_1^T,\ldots,K_D^T] = \begin{bmatrix} Q_1K_1^T & \ldots & Q_1K_D^T \\ \ldots & \ldots & \ldots \\ Q_NK_1^T & \ldots & Q_NK_D^T \end{bmatrix} \\ =& [S_1, S_2, \ldots, S_N]^T \end{aligned} \tag{6}$$

For each bidder $i$, his private value realization forms the weight matrix $S_i$, where none of the weight matrix $S_i$ appears cross-computed according to Eq.6. The attention layer output is

$$\begin{aligned} f(QK^T)V =& [f(S_1),\ldots,f(S_N)]^T \cdot [V_1^T,\ldots,V_D^T] \\ =& \begin{bmatrix} f(S_1)V_1^T & \ldots & f(S_1)V_D^T \\ \ldots & \ldots & \ldots \\ f(S_N)V_1^T & \ldots & f(S_N)V_D^T \end{bmatrix} \end{aligned} \tag{7}$$

where the softmax$(\frac{QK^T}{\sqrt{d_k}})V = f(QK^T)V$ and $V = [V_1, ..V_D]$ is the output of language model described before. Each bidder $i$ independently computes his strategies based on the attention module output $[f(S_i)V_1^T, ..f(S_i)V_D^T]$ with dimension $D$ to the final bidding range $B$ through an MLP, as illustrated in Figure 3.

We then switch to the computation of multi-head attention(Devlin et al., 2018), which is similar to the split of matrix $W = KV$ and combining the result of each split, where bidder's value realization embedding vector $Q$ is computed attention with the same approach in normal attention without cross computation in Eq.8.

$$\begin{aligned} \text{Multi-head}(Q,K,V) =& \text{Concat}(\text{Head}_1, ..\text{Head}_H)W^O \\ \text{Head}_h =& \text{Attention}(Q_h, K_h, V_h) \\ Q_h =& Q * W^h \end{aligned} \tag{8}$$

After the check of no-cross computation in the attention module, we also explain why we should add the bidder's value distribution into the decoder module. If we do not add the bidder $i$'s value distribution $F_i$ into vector $Q$, then the same output $S_A = S_B$ is derived if bidder A's current value realization equals bidder B's value. This may lead to a wrong answer and low performance due to the two bidder can obey completely different value distributions and coincidentally have the same value realization, and their bidding strategies can be completely different intuitively.

# E. Proof

## E.1. Discrete Approximation

*Proof of Theorem 3.1.* As $\hat{\mathcal{F}}$ is a piecewise constant function, we can construct $\hat{\mathcal{F}}$ as one empirical distribution of $\mathcal{F}$. Via Dvoretzky–Kiefer–Wolfowitz inequality, with high probability, the empirical distribution function of $n$ samples would ensure $\sup_v |\hat{\mathcal{F}}(v) - \mathcal{F}(v)| \leq \mathcal{O}(\sqrt{1/n})$ in approximating the cumulative distribution function. Since we have nonzero probability, there must exist $\hat{\mathcal{F}}$ satisfies Theorem 3.1. □

*Proof of Theorem 3.2.* We use the Theorem 1 in Bichler et al. (2022). As described in Sec. 3.1, we equally discretize both the value and bid space into $\mathcal{V} = \{0, 1/H, 2/H, \dots\}$, and we map continuous value $v_i^c$ to the discrete one via $\arg\min_{v_i^d \in \mathcal{V}} |v_i^d - v_i^c|$. Without loss of generality, we assume the value and bid range is normalized to $[0, 1]$, so the coarseness of value space is $\delta_\tau = \max_i \sup_{v_i^c} |v_i^c - v_i^d| = \frac{1}{2H}$. For the mapping of bid, with $\alpha^+(b_i^c)$ being the minimal discrete bid in $\mathcal{V}$ not smaller than $b_i^c$ and $\alpha^-(b_i^c)$ being the maximal bid in $\mathcal{V}$ not greater than $b_i^c$, we have the coarseness of bidding space: $\delta_\alpha = \max_{s\in\{+,-\}} \max_i \sup_{b_i^c} |b_i^c - \alpha^s(b_i^c)| = \frac{1}{H}$.

By incorporating $\delta_\tau = \frac{1}{2H}, \delta_\alpha = \frac{1}{H}$ into Theorem 1 of Bichler et al. (2022), we immediately have the approximation bound $\epsilon + \mathcal{O}(\delta_\tau + \delta_\alpha) = \epsilon + \mathcal{O}(\frac{1}{H})$ in Theorem 3.2. □

## E.2. Loss Function and Computation Details

As described previously, the Nash error loss function we use defined in Eq. 4 enables efficient learning without ground-truth solutions to train our model:

$$\mathcal{L}_{\text{Nash}}(\mathcal{F}, X) = \frac{1}{|\boldsymbol{X}|} \sum_{X_j \in \boldsymbol{X}} \left( \max_{\boldsymbol{\sigma}_i \in \mathcal{D}, i \in N} \left[ \boldsymbol{u}_i^j(\boldsymbol{\sigma}_i^j, \hat{\boldsymbol{\sigma}}_{-i}^j) - \boldsymbol{u}_i^j(\hat{\boldsymbol{\sigma}}_i^j, \hat{\boldsymbol{\sigma}}_{-i}^j) \right] \right)$$
$$\text{s.t.} \quad [\hat{\boldsymbol{\sigma}}_1^j, \cdots, \hat{\boldsymbol{\sigma}}_N^j] = \mathcal{F}(X_j).$$

With $\hat{\boldsymbol{\sigma}}_i^j$ being the model's output and $\boldsymbol{u}_i^j(\cdot)$ defined by the mechanism and value distribution from model's input $X^j$, the main cost of the Nash error comes with the maximization of each bidder $i$'s utility $\boldsymbol{u}_i^j(\boldsymbol{\sigma}_i^j, \hat{\boldsymbol{\sigma}}_{-i}^j)$, or equivalently, the computation of best-response strategy of bidder $i$.

We have demonstrated that the maximal utility can be computed explicitly and directly in Proposition 3.3, obviating the need for supplementary optimization tools. This facilitates a more efficient and streamlined analysis of the model's performance in predicting Nash equilibria, and the proof is as follows:

*Proof of Proposition 3.3.* As defined in Section 2, the utility function given mechanism $\mathcal{M}$'s payment and allocation rule $\mathbf{p}, \mathbf{g}$ is

$$u_i(v_i, b_i, B_{-i}) = v_i g_i(b_i, B_{-i}) - p_i(b_i, B_{-i}).$$

Based on the model's prediction $\hat{\boldsymbol{\sigma}}_i^j$ and mechanism $\mathcal{M}^j$ in auction game $X_j$, we can explicitly compute the expected utility for each bidder $i$, $\boldsymbol{u}_i^j(\boldsymbol{\sigma}_i^j, \hat{\boldsymbol{\sigma}}_{-i}^j)$, given value $v_i^j$ by Eq.(5) as follows:

$$\begin{aligned}
\max_{\boldsymbol{\sigma}_i \sim \mathcal{D}} \boldsymbol{u}_i^j(\boldsymbol{\sigma}_i^j, \hat{\boldsymbol{\sigma}}_{-i}^j) &= \max_{\boldsymbol{\sigma}_i \sim \mathcal{D}} \mathbb{E}_{b_i \sim \sigma_i^j(v_i^j)} \left[ \mathbb{E}_{B_{-i} \sim \hat{\boldsymbol{\sigma}}_{-i}^j} \left[ u_i(v_i^j, b_i, B_{-i} | \mathcal{M}^j) \right] \right] \\
&= \max_{\boldsymbol{\sigma}_i \sim \mathcal{D}} \sum_{b_i} \Pr(b_i | \sigma_i^j(v_i^j)) \cdot \sum_{p,g} (v_i^j g - p) \cdot \Pr(p_i^j(b_i, B_{-i}) = p, g_i^j(b_i, B_{-i}) = g | \hat{\boldsymbol{\sigma}}_{-i}^j) \\
&= \max_{b_i} \sum_{p,g} (v_i^j g - p) \cdot \Pr(p_i^j(b_i, B_{-i}) = p, g_i^j(b_i, B_{-i}) = g | \hat{\boldsymbol{\sigma}}_{-i}^j),
\end{aligned}$$

where $p_i^j(b_i, B_{-i}), g_i^j(b_i, B_{-i})$ are the payment and allocation functions associated with the mechanism $\mathcal{M}^j$.

For the first-price auction mechanism, the allocation rule and payment rule become:

$$g_i^j(b_i, B_{-i}) = \mathbb{I}(b_i > \max_{k\neq i} b_k)$$
$$p_i^j(b_i, B_{-i}) = b_i \cdot \mathbb{I}(b_i > \max_{k\neq i} b_k).$$

which implies

$$\begin{aligned}
&\Pr(p_i^j(b_i, B_{-i}) = p, g_i^j(b_i, B_{-i}) = g|\hat{\boldsymbol{\sigma}}_{-i}^j) \\
=&\Pr(\mathbb{I}(b_i > \max_{k\neq i} b_k) = g, \mathbb{I}(b_i > \max_{k\neq i} b_k) b_i = p|\hat{\boldsymbol{\sigma}}_{-i}^j) \\
=&\sum_x \Pr(\mathbb{I}(b_i > x) = g, \mathbb{I}(b_i > x) b_i = p|\hat{\boldsymbol{\sigma}}_{-i}^j) \cdot \Pr(\max_{k\neq i} b_k = x|\hat{\boldsymbol{\sigma}}_{-i}^j) \\
=&\sum_x \Pr(\mathbb{I}(b_i > x) = g, \mathbb{I}(b_i > x) b_i = p) \cdot \left[\prod_{k\neq i}\sum_{b_k\leq x} \Pr(b_k|\hat{\boldsymbol{\sigma}}_k^j) - \prod_{k\neq i}\sum_{b_k\leq x-1} \Pr(b_k|\hat{\boldsymbol{\sigma}}_k^j)\right],
\end{aligned}$$

where last equality uses the facts that $\Pr(\max_{k\neq i} b_k \leq x|\hat{\boldsymbol{\sigma}}_{-i}^j) = \prod_{k\neq i}\sum_{b_k\leq x}\Pr(b_k|\hat{\boldsymbol{\sigma}}_k^j)$ and $\mathbb{I}(b_i > x) = g, \mathbb{I}(b_i > x)b_i = p$ being independent on $\hat{\boldsymbol{\sigma}}_{-i}^i$.

Therefore the exact expression of $\max_{\boldsymbol{\sigma}_i\sim\mathcal{D}} \boldsymbol{u}_i^j(\boldsymbol{\sigma}_i^j, \hat{\boldsymbol{\sigma}}_{-i}^j)$ becomes:

$$\begin{aligned}
\max_{\boldsymbol{\sigma}_i\sim\mathcal{D}} \boldsymbol{u}_i^j(\boldsymbol{\sigma}_i^j, \hat{\boldsymbol{\sigma}}_{-i}^j) = \max_{b_i}\sum_{p,g}(v_i^j g - p)\cdot &\sum_{x\leq H}\Pr(\mathbb{I}(b_i > x) = g, b_i\cdot\mathbb{I}(b_i > x) = p) \\
&\cdot\left[\prod_{k\neq i}\sum_{b_k\leq x}\Pr(b_k|\hat{\boldsymbol{\sigma}}_k^j) - \prod_{k\neq i}\sum_{b_k\leq x-1}\Pr(b_k|\hat{\boldsymbol{\sigma}}_k^j)\right].
\end{aligned}$$

Since $g\in\{0,1\}, p\leq H$, the equation further simplifies into:

$$\begin{aligned}
\max_{\boldsymbol{\sigma}_i\sim\mathcal{D}} \boldsymbol{u}_i^j(\boldsymbol{\sigma}_i^j, \hat{\boldsymbol{\sigma}}_{-i}^j) = \max_{b_i} &\sum_{p\leq H, g=0} (0-p) \cdot \sum_{x\leq H}\Pr(\mathbb{I}(b_i > x) = 0, b_i\cdot 0 = p) \\
&\qquad\cdot\left[\prod_{k\neq i}\sum_{b_k\leq x}\Pr(b_k|\hat{\boldsymbol{\sigma}}_k^j) - \prod_{k\neq i}\sum_{b_k\leq x-1}\Pr(b_k|\hat{\boldsymbol{\sigma}}_k^j)\right] \\
&+\sum_{p\leq H, g=1}(v_i^j - p) \cdot \sum_{x\leq H}\Pr(\mathbb{I}(b_i > x) = 1, b_i\cdot 1 = p) \\
&\qquad\cdot\left[\prod_{k\neq i}\sum_{b_k\leq x}\Pr(b_k|\hat{\boldsymbol{\sigma}}_k^j) - \prod_{k\neq i}\sum_{b_k\leq x-1}\Pr(b_k|\hat{\boldsymbol{\sigma}}_k^j)\right] \\
= \max_{b_i} \quad 0 + \quad &(v_i^j - b_i)\cdot\sum_{x<b_i}\left[\prod_{k\neq i}\sum_{b_k\leq x}\Pr(b_k|\hat{\boldsymbol{\sigma}}_k^j) - \prod_{k\neq i}\sum_{b_k\leq x-1}\Pr(b_k|\hat{\boldsymbol{\sigma}}_k^j)\right].
\end{aligned}$$

For the second price auction mechanism, the allocation rule and payment rule are defined as:

$$g_i^j(b_i, B_{-i}) = \mathbb{I}(b_i > \max_{k\neq i} b_k)$$
$$p_i^j(b_i, B_{-i}) = \max_{k\neq i} b_k \cdot \mathbb{I}(b_i > \max_{k\neq i} b_k).$$

Similarly, we have the maximal utility simplified into:

$$\max_{\boldsymbol{\sigma}_i\sim\mathcal{D}} \boldsymbol{u}_i^j(\boldsymbol{\sigma}_i^j, \hat{\boldsymbol{\sigma}}_{-i}^j) = \max_{b_i}\sum_{x<b_i}(v_i^j - x)\cdot\left[\prod_{k\neq i}\sum_{b_k\leq x}\Pr(b_k|\hat{\boldsymbol{\sigma}}_k^j) - \prod_{k\neq i}\sum_{b_k\leq x-1}\Pr(b_k|\hat{\boldsymbol{\sigma}}_k^j)\right].$$

□

The best-response $b_i^*$ in RHS can be calculated directly from the model's inputs and outputs, thus the entire computation of the Nash error can be carried out without the need for labeled equilibrium strategies data.

It's also noteworthy that with discretization, scenarios may arise where multiple bidders place the same maximum bid, leading to a tie. In such cases, a tie-breaking mechanism is necessary for the discretized game. Following Bichler et al. (2022), we simply set that no bidder is declared the winner in the event of a tie, i.e. $g_i(b_i, B_{-i}) = \mathbb{I}(b_i > \max_{k \neq i} b_k)$. Though the efficient calculation above is derived using this tie-breaking rule, it has been shown that with alternative tie-breaking rules, such as uniform tie-breaking, the computation of best-response remains tractable (Filos-Ratsikas et al., 2021).

## F. Additional Ablation Experiments

### F.1. Compare with Other No-Regret Learning Methods

In Table. 5, we compared a list of algorithms' abilities to solve a 2-player symmetric first price auction game, including a series of NN-based algorithms and MWU as the no-regret learning algorithm. While it has been established in Feng et al. (2021) that mean-based algorithms—which encompass a broad range of no-regret learning methods—can converge to a Nash equilibrium in symmetric first-price auction settings, the same context as our benchmark in Table 5, it is still crucial to investigate other no-regret learning methods beyond MWU to provide a more comprehensive evaluation.

With regards to this, we also report experiment outcomes for two additional no-regret learning algorithms: Follow The Perturbed Leader (FTPL) and Optimistic Multiplicative Weights Update (OMWU). FTPL is a popular no-regret learning algorithm employed across a variety of applications (Kalai & Vempala, 2005), While OMWU is a modification of MWU noted for its last-iterate convergence characteristic in zero-sum games (Lei et al., 2021).

We implement FTPL by each bidder $i$ bidding according to $b_i^t = \arg\max_b \tilde{u}_i^t(v, b) + p(b)$, where $\tilde{u}_i^t(v, b)$ is the total utility of bidding $b$ at value $v$ before round $t$, and $p(b)$ is a random noise sampled from an exponential distribution $f(x) = \varepsilon e^{-\varepsilon x}, x \geq 0$. As for OMWU, we maintain similar weights to MWU, where $w_i^t(b, v)$ is the weight at round $t$ of bidding $b$ at value $v$, but the update rule is modified as: $w_i^t(b, v) \propto w_i^{t-1}(b, v) e^{-\eta \cdot (\sum_{t'=1}^{t-2} u_i^{t'}(b,v) + 2u_i^{t-1}(b,v))}$. We set $\varepsilon = 0.1, \eta = 0.1$ and all no-regret algorithms runs 100,000 rounds.

*Table 6.* The $L_2$ distance between computed strategies by different methods and the analytic solution, and the running time $t$ (both smaller is better). We set the discretization grid $H$ to either 32 or 64 for no-regret learning algorithms, and maintain a size of 20 for Auctionformer.

| Method | $L_2$ | Time spent $t$ |
|---|---|---|
| Random | 0.418 | - |
| MWU-32 | 0.021 | 27.97 |
| MWU-64 | 0.019 | 31.22 |
| FTPL-32 | 0.026 | 21.08 |
| FTPL-64 | 0.024 | 22.26 |
| OMWU-32 | 0.023 | 32.62 |
| OMWU-64 | 0.017 | 33.33 |
| Auctionformer-20 (ours) | 0.017 | 0.006s |

The empirical results are included in Table. 6. Despite setting a finer discretization for the no-regret learning algorithms (i.e., $H = 32, 64$), which theoretically enhances the approximation quality to the continuous analytical solution, the resulting $L_2$ distances for these methods still can not surpass the solution quality of Auctionformer. Additionally, each no-regret algorithm requires retraining to evolve toward the Nash equilibrium, whereas Auctionformer necessitates only a single inference step once trained.

### F.2. Auctionformer v.s. MLPNet

As outlined in Section C.3, we've developed an MLPNet model specially trained for 3-player FP auction games. Although MLPNet is only capable of solving auction games with a fixed number of bidders, we can still compare the predicted strategies of MLPNet for this particular setting with the results of Auctionformer.

To facilitate this comparison, we extract all the 3-player auction games described in Table. 2 and benchmark these 2 models' predicted equilibrium strategies via Nash loss. The results are presented in Table. 7.

*Table 7.* The $\mathcal{L}_{\text{Nash}}$ of Auctionformer's and MLPNet's solutions on the 3-player FP auction games among different value distributions

| $\mathcal{L}_{\text{Nash}}$ on: | $U+G$ | $U_0$ | $G_0$ | $U_{-0}$ | $G_{-0}$ |
|---|---|---|---|---|---|
| Auctionformer | **7.63e-05** | **4.58e-05** | **4.29e-05** | **1.47e-04** | **9.06e-05** |
| MLPNet | 1.30e-03 | 7.18e-04 | 7.09e-04 | 1.87e-03 | 2.31e-03 |

Despite MLPNet's specialization for three-player first-price auction scenarios, it still yields a higher Nash error (1.3e-03) compared to Auctionformer (7.6e-05). This contrast underscores the superior performance of our method.

## F.3. Disable Parameter Update on Bert

In the previous experiment, we train Auctionformer with the pretrained BERT as the default base language model and train Auctionformer for all parameters, where the ability to solve vast auction games is reported in Table 2. One may raise another question, what if we avoid fine-tuning the language model and train Auctionformer with the rest part of it?

In order to answer the following question, we also report the results where we fixed the pretrain language model Bert and train the Auctionformer in the same way denoted in Sec.4.1. Similarly, we train such Auctionformer in the same 4 mixed mechanism and asymmetric training set and display the results with the same test set in Table 2. The results are displayed in Table. 8

*Table 8.* The $\mathcal{L}_{\text{Nash}}$ of a special trained Auctionformer's solutions on the auction games among different value distributions, mechanisms, and asymmetric player numbers ("Ent" refers to entry fee). The Auctionformer is trained with fixed BERT's parameters where we disable the parameter updating on BERT during the whole training process. The $\mathcal{L}_{\text{Nash}}$ of random strategy is 1.59e-1 for comparison.

| | **Overall** | **Player Number = 2∼4** Sample: 2000/distribution | | | | **Player Number = 5∼7** Sample: 2000/distribution | | | | **Player Number = 8∼10** Sample: 2000/distribution | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Hybrid** | **FP** | **SP** | **FP+Ent** | **SP+Ent** | **FP** | **SP** | **FP+Ent** | **SP+Ent** | **FP** | **SP** | **FP+Ent** | **SP+Ent** |
| **U+G** | 3.17e-04 | 1.71e-04 | 6.50e-05 | 6.46e-04 | 2.89e-05 | 1.14e-04 | 2.32e-04 | 6.51e-04 | 5.00e-05 | 9.20e-05 | 4.07e-04 | 7.19e-04 | 7.97e-05 |
| $U_0$ | 1.40e-04 | 1.30e-04 | 1.33e-05 | 2.78e-04 | 1.98e-05 | 1.09e-04 | 3.82e-05 | 3.12e-04 | 3.14e-05 | 9.35e-05 | 8.34e-05 | 2.73e-04 | 4.61e-05 |
| $G_0$ | 1.70e-04 | 9.89e-05 | 6.10e-08 | 3.16e-04 | 1.55e-05 | 9.24e-05 | 1.77e-07 | 4.34e-04 | 2.62e-05 | 6.65e-05 | 4.10e-07 | 4.34e-04 | 4.64e-05 |
| $U_{-0}$ | 4.48e-04 | 2.89e-04 | 2.28e-04 | 8.70e-04 | 4.69e-05 | 1.63e-04 | 7.62e-04 | 7.08e-04 | 8.06e-05 | 1.42e-04 | 1.28e-03 | 7.89e-04 | 1.39e-04 |
| $G_{-0}$ | 5.10e-04 | 1.68e-04 | 1.91e-05 | 1.12e-03 | 3.33e-05 | 9.26e-05 | 1.26e-04 | 1.15e-03 | 6.19e-05 | 6.54e-05 | 2.65e-04 | 1.38e-03 | 8.76e-05 |

From the Table. 8, we found that the overall Loss in hybrid cases is $2\times$ larger than the Auctionformer with BERT trainable (which is 1.56e-04 averaged over all mechanisms). However, the magnitude of solved auction games approximation is still satisfying, which is similar to other works such as Duan et al. (2023a). Therefore, the scalability of the proposed Auctionformer is also maintained to a degree even if we directly utilize fixed pretrained language model parameters. This inspires us to apply other language models (i.e. GPT4(OpenAI, 2023)) as a feature extractor or interface in the proposed Auctionformer structure to solve complex auction games in the future works, which may avoid huge computation resources on fine-tuning LLM.

## F.4. Pretrained LM v.s. Training from Scratch

While our previous experiments mainly utilized pretrained language models (PLMs) such as BERT, GPT, or LLaMA, the architecture of Auctionformer is indeed compatible with transformers that are not pretrained. Our preference for PLMs is grounded in the presumption that these models may already encapsulate useful knowledge that could benefit the training process for auction equilibrium solving, e.g. better performance or faster convergence.

In light of this, we conducted another ablation study to explore the advantages of using a pretrained language model over initiating training from scratch. For this study, we selected the pretrained LLaMA2-7B model to serve as the LM backbone for the Auctionformer, chosen for its demonstrated rapid convergence, as evidenced in Fig.6. We contrasted the pretrained model's performance with an Auctionformer variant built upon a randomly initialized transformer possessing the same structural specifications as LLaMA2-7B. Our investigation employed two distinct sets of training hyperparameters, featuring varying learning rate decay schedules–'arg1' halves the rate every two epochs, whereas 'arg2' does so every four epochs. In Fig.11, we present the Nash loss recorded for both models under the two different hyperparameter setups on the validation set.

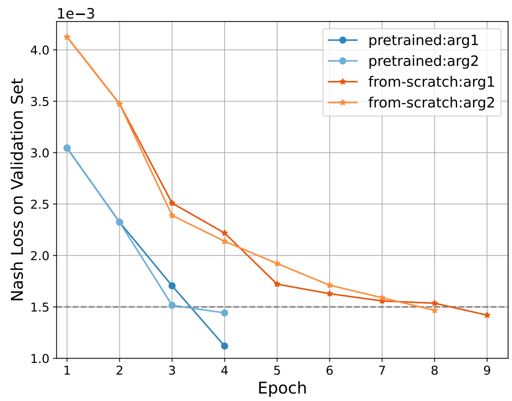


*Figure 11.* Nash loss comparison between the Auctionformer models employing a pretrained LLaMA2-7B model ('pretrained') and a randomly initialized transformer network ('from-scratch'), across two hyperparameter settings ('arg1' and 'arg2') on the validation set.

As depicted in the figure, the Auctionformer powered by the pretrained LLaMA2-7B model showcases a significantly faster convergence rate compared to its from-scratch counterpart, requiring only half the epochs to reduce the validation loss to below 1.5e-3. This obeservation supports our hypothesis that the pretrained language model may possess inherent knowledge that could benefit the training process and validates the rationale of utilizing the pretrained language model in our previous experiments.

### F.5. Zero-shot on Symmetric Cases

We have displayed the Auctionformer performances on vast mechanisms and distributions in Table 3 and find disappointing results in $N = 15$, which the zero-shot ability on the out-of-the-domain cases, including several auction games without explicit solutions. However, when it comes to symmetric cases such as Uniform distribution in the first price or second price, the Auctionformer also adopts a certain ability on the zero-shot scenarios.

We turn to the zero-shot experiments on two classical mechanisms FP and SP with our trained Auctionformer, where the symmetric bidders with Uniform distribution $U_0$ among different bidder numbers. In the symmetric cases, there exist theoretical results on these two special cases(Krishna, 2009) as below.

**Lemma F.1.** *In a first price auction with $n$ players, where each bidder's value $v_i$ is drawn i.i.d. from the uniform distribution $U[0, V]$, bidding $b_i = \frac{n-1}{n} v_i$ constitutes a Bayesian Nash equilibrium.*

**Lemma F.2.** *In a second price auction with $n$ players, where each bidder's value $v_i$ is drawn i.i.d. from the uniform distribution $U[0, V]$, truthful bidding $b_i = v_i$ constitutes a Bayesian Nash equilibrium.*

In the in-domain cases with the maximal bidder number is 10, the averaged bidding strategies of bidders are displayed in Figure 12 compared to MWU. As there exists a theoretical ratio between the bidding price and value realization, we can estimate the averaged bidding strategies of the derived solution from Auctionformer and MWU. The complexity or the difficulty is intuitively increased with the growth of bidder number in auction games and both algorithms are validated in a $1/H$-evenly-discretized space (Feng et al., 2021) ($H = 20$), which may lead to an approximation decay in large bidder number in FP. However, Auctionformer still outperforms MWU in most in domain cases with closer distances to the theoretical results. In the SP, Auctionformer can derive the truthful bidding for every bidder while MWU fails to deliver such optimal bidding strategies.

Then we turn to the out-of-domain cases and the results are recorded in Figure 13, where the maximally involved bidders are up to 15. Similarly, in the SP cases, Auctionformer can still derive the equilibrium solution in the symmetric cases, where bidding the same as value realization is the optimal bidding strategy. In the FP out-of-domain cases, our Auctionfomer

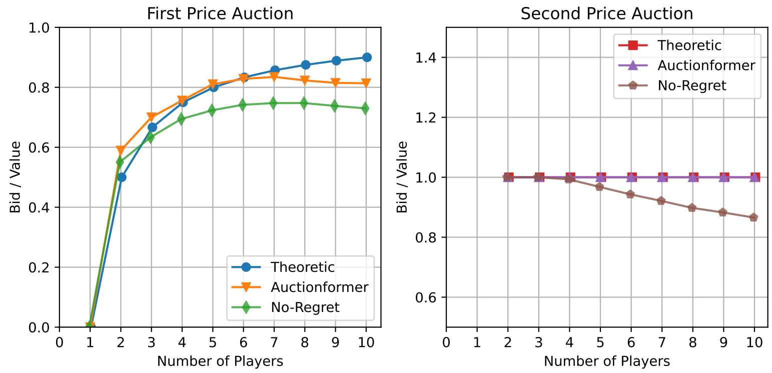


*Figure 12.* Uniform distribution $U[0, 10]$ of different symmetric bidder numbers where Auctionformer is tested in the same domain of the training set. The no-regret results are predicted by the MWU algorithm(Arora et al., 2012).

derives much closer approximation solutions compared to a no-regret algorithm. Even if the Auctionformer is trained on the $N = [2, 10]$, it still manages to deliver an approximate equilibrium solution with the simple symmetric out-of-domain cases and no few-shot learning is required.

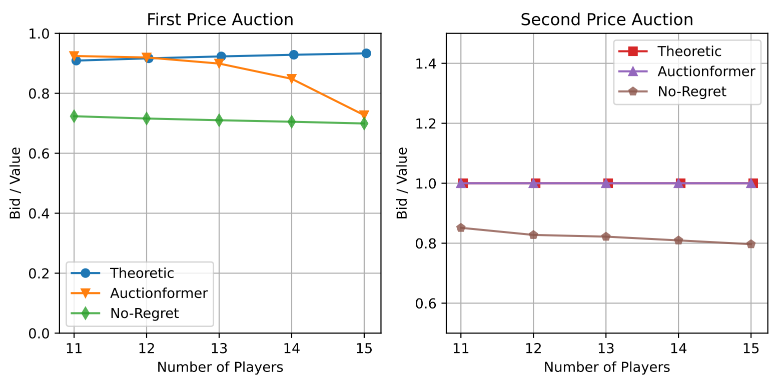


*Figure 13.* Zero-shot results on uniform distribution $U[0, 10]$ of different symmetric bidder numbers where Auctionformer is tested out of the domain of the training set. The no-regret results are predicted by the MWU algorithm(Arora et al., 2012).

### F.6. Few-shot on Mechanism Transformation

We also validate the few-shot ability of mechanism transformation. To assess our model's ability to transfer to different auction mechanisms, we conduct another few-shot learning experiment in which we employ 3 distinct models trained on the first price mechanism under different distributions: $U_0$, $G_0$, and $U_0 + G_0$, and fine-tune them on second price auctions with corresponding distributions. We use the same training settings in the Sec.4.1 for all 3 models.

From the Table 9, we can find that the Auctionformer directly trained in FP cannot derive a satisfying approximation solution on SP, with the magnitude of $\mathcal{L}_{Nash}$ to 1e-2. The direct few-shot on SP mechanism with only $1500$ training samples can enhance the ability of SP with different types of value distributions. However, directly using Nash loss to train on the new mechanism data will decrease the equilibrium solving performances in the previous domain (FP). Our proposed few-shot learning framework with preservation loss can significantly boost the ability of the mechanism transformation from Table 9. Not only the $\mathcal{L}_{Nash}$ in trained mechanism FP is preserved to 1e-4 magnitude, but the solution in SP with $1500$ samples is approximate to the model with only Nash loss.

Noted that no ground-truth or theoretical solution is required and Auctionformer is scalable for any input value distribution. In line with previous few-shot experiments, our proposed self-supervised fine-tuning approach is able to transfer Auctionformer to new auction game domains (including bidder number and mechanism) with a few samples.

*Table 9.* Few-shot learning results on the Auctionformer trained on first price (FP) mechanism to second price (SP) mechanism, where the Nash errors are recorded on $U_0$ and $G_0$ and "Pre" denotes the preservation loss.

| Distribution | Approach | $\mathcal{L}_{Nash}$ on FP | $\mathcal{L}_{Nash}$ on SP |
|---|---|---|---|
| $U_0$ | **Train on FP** | 4.30e-04 | 2.42e-02 |
| | **+ Only Nash loss** | 2.55e-02 | 6.67e-03 |
| | **+ Ours (Nash+Pre.)** | 5.99e-04 | 6.92e-03 |
| $G_0$ | **Train on FP** | 1.35e-04 | 2.06e-02 |
| | **+ Only Nash loss** | 2.79e-02 | 2.70e-03 |
| | **+ Ours (Nash+Pre.)** | 2.75e-04 | 2.81e-03 |
| $U_0 + G_0$ | **Train on FP** | 1.71e-04 | 3.31e-02 |
| | **+ Only Nash loss** | 3.10e-02 | 3.74e-03 |
| | **+ Ours (Nash+Pre.)** | 2.64e-04 | 3.70e-03 |