AuctioNN: A Simulation-Driven Bidding Engine for Large-Scale Cost-Per-Action Optimization

Param Kapur, Cameron Adams, Quinn Behrens, Ernie Zhang
{paramkapur, cadams, qbehrens, irenez}@reed.edu

April 25, 2025

Abstract

AuctioNN is a sandbox framework and decision engine that evaluates whether a *neural-network–guided bidding policy* can reduce system-wide cost-per-action (CPA) while meeting campaign-specific goals and operational constraints. The instrumented simulator streams impression requests one at a time, emulating an external, first-price and second-price ad exchange. We rigorously compare the proposed policy against heuristic baselines using *marginal CPA*, total conversions, and revenue uplift. Results generalize to media owners such as *iHeart* that simultaneously monetize their owned&operated (O&O) inventory and purchase incremental reach on open exchanges. The full architecture, mathematical formulation, and open research questions are documented here.

1 Introduction and Motivation

Large media publishers increasingly extend their reach by *buying* impressions on open ad exchanges. Operating 10^3–10^4 concurrent campaigns in real time raises three entangled challenges:

1. **Win the *right* inventory**—impressions that drive post-click or post-view conversions.
2. **Respect campaign goals**—especially CPA targets stipulated in insertion orders.
3. **Obey operational constraints**—budgets, pacing, and user-level frequency capping.

This work posits that a neural model, combined with explicit budget/frequency bookkeeping, can outperform classical heuristics that ignore conversion likelihood.

2 How an Open Ad Exchange Works

For each page-view or app launch, the exchange issues a *bid request* containing contextual metadata (timestamp, device, geo, etc.). Every buyer has roughly 100 ms to (i) decide whether to participate and (ii) transmit a bid price. The highest bid wins and immediately pays the *clearing price*. In most modern exchanges this is a *first-price auction*; the winner pays its own bid. However, some exchanges use different auction models like *second-price* or programmatic guaranteed (PG) auctions. Any downstream *conversion event* (purchase, signup, ...) is reported asynchronously, often minutes to days later.

3 System Architecture & Environment

AuctioNN treats the exchange as an online stream, delivering impression I_t at discrete time t. For every impression the engine must *(a)* decide whether to bid, *(b)* pick a campaign, and *(c)* set the bid price. Table 1 formalizes the symbols used throughout the paper.

Table 1: Core symbols and state variables.

Symbol / Term	Meaning
I_t	Impression delivered at discrete time t.
features(I_t)	All metadata included in the bid request.
C	Active campaign set.
$c \in C$	One specific campaign.
budget_remaining$[c]$	Dollars left for campaign c.
$p_{\text{conv}}(c, I_t)$	Predicted conversion probability if c wins I_t.
value_per_conv$[c]$	Advertiser-declared value of one conversion.
target_CPA$[c]$	Optional maximum CPA for campaign c.
ad_stock$[\text{user}, c]$	Recent exposure score (models ad fatigue).
score(c, I_t)	Scalar used to select the winning campaign.
bid	Dollar price sent to the exchange.
clearing_price	Amount paid if the bid wins (= bid in first-price).
marginal CPA	ΔSpend/ΔConversions over a window.

3.1 End-to-End Decision Loop

The procedure below executes for *every* incoming impression.

Algorithm 1 AuctioNN per-impression decision loop

Require: incoming impression I_t

Receive I_t; extract features(I_t)

Eligibility gate:

  $C_{\text{elig}} \leftarrow \{c \in C \mid \texttt{budget\_remaining}[c] > 0 \ \wedge \ \texttt{ad\_stock}[\text{user}, c] < \tau\}$

for all $c \in C_{\text{elig}}$ **do**

  $p_{\text{conv}}(c, I_t) \leftarrow f_\theta\big(\text{features}(I_t), \text{embed}(c), \text{ad\_stock}[\text{user}, c]\big)$ ▷ see §4

  $\text{score}(c, I_t) \leftarrow \begin{cases} \dfrac{p_{\text{conv}}}{\text{target\_CPA}[c]}, & \text{if target\_CPA exists,} \\ p_{\text{conv}} \times \text{value\_per\_conv}[c], & \text{otherwise.} \end{cases}$

end for

$c^* \leftarrow \arg\max_c \text{score}(c, I_t)$

expected_value $\leftarrow p_{\text{conv}}(c^*, I_t) \times \text{value\_per\_conv}[c^*]$

bid $\leftarrow \beta \times$ expected_value ▷ $\beta = 1$ initially[1]

Transmit bid; observe win/loss and clearing_price

if win **then**

  $\texttt{budget\_remaining}[c^*] \mathrel{-}=$ clearing_price

  $\text{ad\_stock}[\text{user}, c^*] \mathrel{+}= 1$

end if

Log *(*I_t, c^*, bid, win/loss, time, utility*)* for offline analysis

3.2 Decision Loop Pipeline

The figure below illustrates the decision loop in a block diagram.

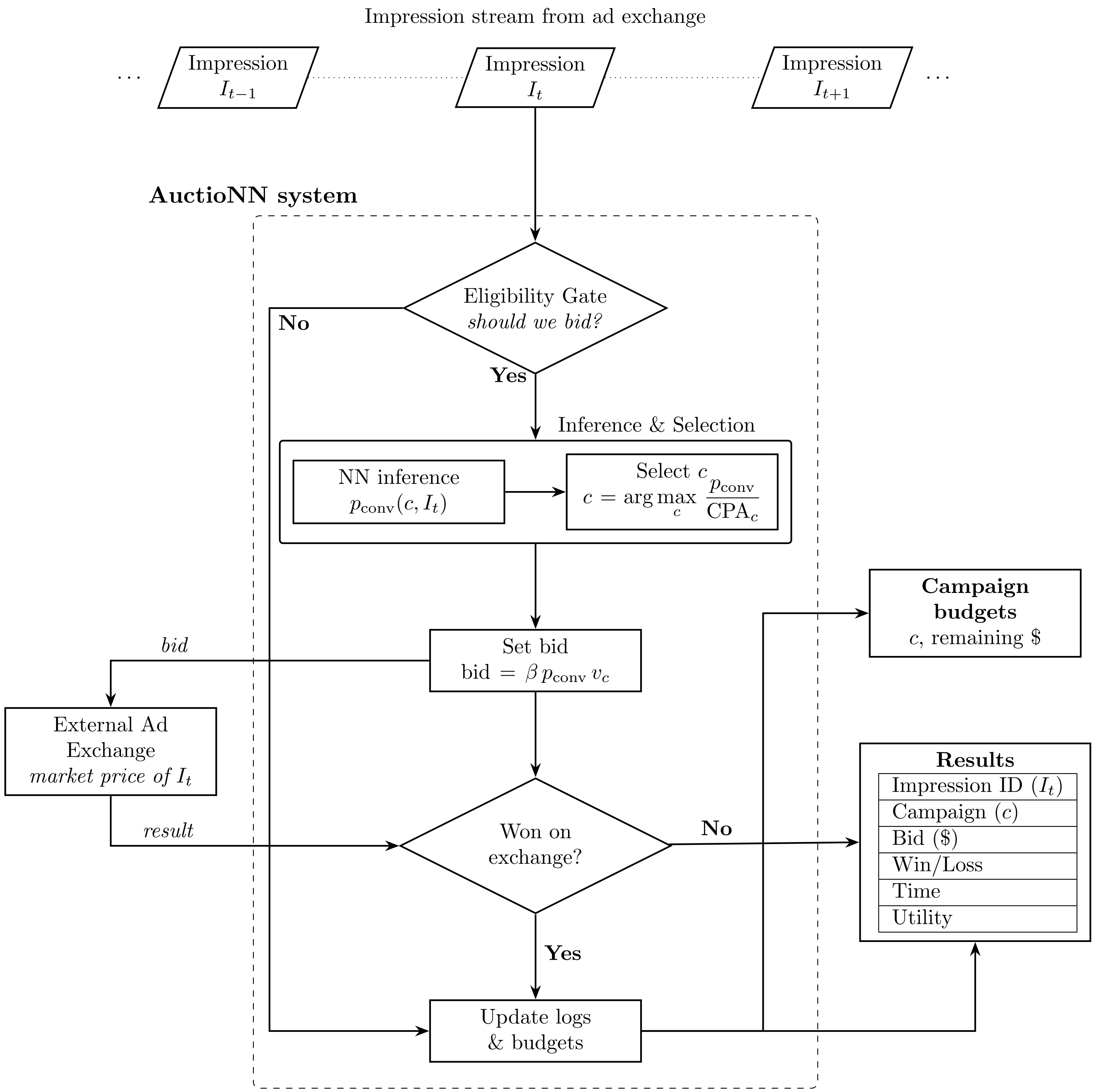

Figure 1: Fully connected block diagram of AuctioNN's *per-impression* decision loop, emphasising the continuous stream of impressions $I_{t-1}, I_t, I_{t+1}, \ldots$. The compound *Inference & Selection* stage first scores every campaign with a neural network and then chooses c via the displayed arg-max rule.

4 Neural Model

Objective. Estimate conversion probability conditioned on both the *impression* and the *campaign*. This is analogous to a contextual click-through-rate model but with a conversion label.

Feature vector.

- Categorical embeddings: device type, operating system, DMA, and campaign ID.
- Temporal features: hour-of-day and day-of-week encoded as sine/cosine pairs.
- Continuous exposure feature: current ad_stock (per-user, per-campaign).

Architecture. Two fully connected layers ($128 \to 64$) with ReLU activations, followed by a sigmoid output; $\approx 55\,\text{k}$ parameters. Exported as TorchScript for ≤ 1 ms inference on an Apple M2 core.

Training. Binary cross-entropy on historical (impression, conversion) pairs. Severe class imbalance ($\mathcal{O}(10^{-3})$ converters) is mitigated via positive-class weighting and mini-batch stratification. The data pipeline materialises a 7-day attribution window (see §7).

5 Evaluation Methodology

We benchmark three policies:

1. **Random**—bid a constant $1 CPM on a random eligible campaign.
2. **Heuristic**—industry default: bid proportional to campaign value, ignoring p_{conv}.
3. **AuctioNN (proposed)**—full decision loop of §3.2.

5.1 Success Metrics

- **Marginal CPA** (ΔSpend/ΔConversions) relative to baseline.
- **Total conversions** aggregated across all campaigns.
- **Revenue uplift** (= advertiser value − media cost).
- **Pacing error**: $|\text{actual spend} - \text{planned spend}|/\text{planned spend} \leq 0.05$.
- **Latency**: distributive $p_{95} \leq 2\,\text{ms}$ per impression on a MacBook M-series laptop.

6 Implementation Notes

7 Open Questions and Future Work

8 Conclusion

AuctioNN provides a controlled environment to test whether modern ML can simultaneously optimise CPA, satisfy campaign obligations, and execute within the harsh latency limits of real-time bidding. The architecture is intentionally minimal yet extensible, enabling rigorous *ablation studies* of every engineering choice—from eligibility gates to neural features to bid shading policies.