ALPHAEXPLOITEM: GOING BEYOND THE NASH EQUILIBRIUM IN POKER BY LEARNING TO EXPLOIT SUBOPTIMAL PLAY

Anonymous authors
Paper under double-blind review

ABSTRACT

Poker is an imperfect information game that has served as a long-standing benchmark for decision-making under uncertainty. To maximize utility beyond the Nash equilibrium, an agent can deviate from equilibrium policies to exploit suboptimal play. We introduce AlphaExploitem, which extends the well-performing RL poker agent AlphaHoldem with a hierarchical transformer encoder that reasons over previously played hands, and we train it to exploit a suboptimal play produced by learned playstyles. We use a style variational autoencoder that learns a continuous latent space of play styles from trajectory data and decodes any sampled latent into a queryable opponent policy. On Leduc, a standard benchmark for poker research, AlphaExploitem extracts $+1.67$ big blinds per hand from the test-set, more than three times what a non-adaptive exact Nash equilibrium extracts from the same set, and roughly five times the league-only baseline. Because the generator learns its style space from general game histories, the same method can be applied on human hand histories.

1 INTRODUCTION

Poker is a family of card games whose incomplete information setting, stochastic elements, and deep strategic complexity have made it a long-standing challenge for artificial intelligence (Rubin & Watson, 2011). State-of-the-art poker agents approximate the Nash Equilibrium (NE): strategies from which no player can improve by unilateral deviation (Nash, 1950). Systems such as DeepStack (Moravčík et al., 2017), Libratus (Brown & Sandholm, 2018), Pluribus (Brown & Sandholm, 2019), and AlphaHoldem (Zhao et al., 2022) have achieved high performance in various poker formats by learning low-exploitability, approximately-NE strategies. However, while NE guarantees a non-negative expected return against any opponent, it does not maximize returns against out-of-equilibrium play. Against suboptimal opponents, substantial additional expected value (EV) can be captured by deviating from the equilibrium to exploit observed flaws in the opponent's strategy.

Competitive poker is rarely played as a single hand, however. A hand is one full deal of cards from shuffling the deck to determining the winner of the wagered money, and sessions may consist of hundreds to thousands of hands between the same opponents. This creates a repeated-interaction structure that proficient players often leverage (Frey et al., 2018), adapting their strategy in response to patterns observed in an opponent's play, whereas NE-approximating agents treat every hand as an independent encounter and discard any information available from the session so far. Prior work on exploiting suboptimal poker opponents has largely relied on hand-crafted summary features of opponent behavior, either conditional strategies based on aggregated game statistics (Billings et al., 1998; 2002; Xu et al., 2021) or recurrent networks consuming pre-computed summary features (Li & Miikkulainen, 2017; 2018a;b). Li et al. (2024) train a causal transformer via algorithm distillation (Laskin et al., 2022), but the method has been criticized for a training technique that does not guarantee or evaluate on human and rational flawed strategies.

In this work, we introduce AlphaExploitem, an actor-critic agent trained through Proximal Policy Optimization (PPO) (Schulman et al., 2017) for which trajectories are generated via K-best-league self-play (Zhao et al., 2022) together with a population of opponents synthesized by a learned generative model of play styles. To exploit weak play, AlphaExploitem extends AlphaHoldem's architec-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

ture with a hierarchical transformer encoder that processes the raw tokenized history of previously played hands within a session. By using separate learned embedding tables for different types of observations, we provide complete tokenized information over the opponent's behavior and the environment's dynamics, so the model itself processes the game data, with no human-built feature aggregators in between.

A question any system that aims to exploit populations of opponents has to answer, is where does the training data come from. Hand-crafting a fixed set of scripted weak policies is the standard answer, but such a set is small, static, difficult to extend, and does not obviously transfer to the variety of styles encountered in real play. Instead of enumerating opponents, we learn them. We train a style variational autoencoder (VAE) on trajectory data and obtain a continuous latent space of play styles, in which every sampled latent decodes into a complete, coherent opponent policy that AlphaExploitem can roll out against and exploit. Hand-crafted policies still play a role, but only as one source of the trajectory data the generator learns from, not as the training opponents themselves. The system is designed with the aim of easy extension to human poker formats, where we can replace the scripted policies with a dataset of human hands histories.

We evaluate on Leduc (Southey et al., 2012), a standard imperfect-information benchmark. AlphaExploitem captures substantially more per-hand EV than the AlphaHoldem-style league-only baseline while also maintaining robust performance against strong policies. We split the network into two parts: one that analyzes the history of past hands, and a second part that focuses on the current hand. Because training draws a fresh latent style every epoch, no evaluation opponent is ever trained against, so the reported numbers measure how well the agent exploits the style space. Memorization of particular opponents cannot explain them. Since human-like play styles are themselves instances of such a space, a generator made diverse enough to cover human play will, by the same mechanism, yield an agent that exploits human opponents in full-scale formats, and the scripted policies behind our training trajectories can be replaced by data sets of human play already available in conventional formats.

2 BACKGROUND

Poker involves private cards, sometimes shared community cards, and rounds of betting with chips. A player wins either by having the best hand at showdown or by making all others fold, and bluffing is an important strategic element. We refer to the player we are interested in as the hero, and to the other competing players as villains. Each hand begins with forced bets from the players, the main utility metric is big blinds per hand (BBs/hand), and at each decision the active player chooses among fold, check, call, and bet or raise. Players are commonly described in terms of stylistic archetypes that summarize their tendencies across many hands. We use these archetypes as the source of the trajectories our generator learns from, and our agent must exploit (Appendix B).

Imperfect information games, equilibrium, and exploitability. We adopt standard extensive-form notation: each player i has a behavioral strategy σ_i mapping their information sets, decision points the player cannot distinguish, to distributions over legal actions, with terminal utility u_i. In poker, an information set is a decision node at which the acting player knows the public history and their own private cards but cannot distinguish histories that differ only in the opponent's private cards. Heads-up poker is the two-player zero-sum special case ($u_1 = -u_2$), which makes equilibrium and exploitability well-defined and, for Leduc, exactly computable. A Nash equilibrium (Nash, 1950) is a profile σ^* from which no player can improve by unilateral deviation, $u_i(\sigma^*) \geq \max_{\sigma'_i} u_i(\sigma'_i, \sigma^*_{-i})$ for all i, and exploitability (Nikaidô & Isoda, 1955) measures how much an opponent can gain by deviating against a given strategy. An NE has zero exploitability and is therefore safe, but it sacrifices the additional utility available against opponents who are not at the equilibrium. Any deviation of an opponent from equilibrium creates positive best-response value, which is the gap an exploitative agent targets. Fixing a stationary opponent further reduces the game to a single-agent partially observable Markov decision process whose transition and observation distributions are opponent-specific, so an agent that conditions on observed opponent patterns is effectively learning a per-opponent transition model.

AlphaHoldem. AlphaHoldem (Zhao et al., 2022) is a competitive end-to-end RL poker agent that reaches CFR-level play (Moravčík et al., 2017) at a fraction of the compute, via two design choices we inherit. It employs self-play training with PPO through a K-best league, a fixed-size pool of

past checkpoints in which iterations that achieve higher ELO ratings (Elo, 1978) survive while low-performing agents are eliminated. It also uses a pseudo-siamese architecture in which two separate networks process the card and the action information of the current hand and fuse only at the policy and value heads. That clean separation of inputs is what allows for the addittion of a cross-hand module as a third component.

3 Related Work

RL for poker. Techniques complementing the systems named above include Fictitious Self-Play (Heinrich et al., 2015), Deep CFR (Brown et al., 2019) and its model-free successor DREAM (Steinberger et al., 2020), which replace tabular regret with deep networks, and ReBeL (Brown et al., 2020) and Student of Games (Schmid et al., 2023), which couple deep RL with search across imperfect-information games more broadly. These systems aim for low exploitability, not for opponent-specific adaptation, and a blueprint or self-play network plays its computed strategy regardless of who sits across the table.

Exploiting suboptimal opponents. Work here can be grouped by how opponent behavior is represented. Early approaches accumulated hand-crafted statistics over the course of play and passed them to conditional decision-making modules, including expert systems and feed-forward networks (Billings et al., 1998; 2002; Hoehn et al., 2005; Xu et al., 2021). More recent work replaced these pipelines with learned sequence models such as recurrent architectures, but still operated on per-hand summaries or cumulative statistics, never on raw interaction histories (Li & Miikkulainen, 2017; 2018a;b). ASHE (Li & Miikkulainen, 2018a;b) is the strongest member of this family, storing session-level memory in a tree indexed by discretized action sequences whose nodes hold aggregated statistics, and feeding those into LSTM estimators. In contrast, Li et al. (2024) process raw hand histories directly with a flat causal transformer over a single stream of cards, actions, and rewards, though their training protocol has been questioned for not clearly ensuring exposure to a diverse set of weak opponents: their opponents come from random policy initializations and from strategies sampled along CFR and PSRO convergence trajectories, and since random policies have no consistency across different hands, while equilibrium-finding trajectories cluster near NE, neither source represents a generator for human-like flaws. Safe Exploitation Search (Liu et al., 2022) solves an abstracted game at each decision whose solution interpolates between equilibrium play and best response to an opponent model, at the cost of a precomputed blueprint and tree expansion at decision time.

Where training opponents come from. What an exploiter trains against has received less attention than how it models opponents. L2E (Wu et al., 2021) considers exploitation in the meta-learning style and shapes the opponent distribution with an adversarial hard-opponent generator and a diversity regularizer, while OMIS (Jing et al., 2024) supervises an actor, an opponent-imitator, and a critic on best-response trajectories against every training opponent, which requires an inner optimization loop per opponent. Specialized opponent populations (single-agent exploiters, population exploiters, and main agents) are also standard in league training more broadly (Vinyals et al., 2019). Our generator differs in that it is fit to trajectories, produces a continuous space in place of a finite roster, and needs neither an inner best-response loop nor access to opponent strategy tables.

LLM-based approaches. PokerGPT (Huang et al., 2024) casts the poker state as natural language prompts, and Suspicion-Agent (Guo et al., 2024) wraps a frozen general model in a parsing and planning pipeline, beating learning-based baselines on Leduc but losing to CFR+. Current benchmarking indicates that language models do not match dedicated architectures, especially against NE (Provost et al., 2026; Zhuang et al., 2025). Lin & Hou (2026) vary access to persistent between-hand memory across poker sessions and find that memory is necessary and sufficient for opponent reasoning to emerge, independent evidence that session-level memory is the right architectural target.

4 AlphaExploitem: Transformer-based Exploitation

AlphaExploitem extends AlphaHoldem to enable cross-hand adaptation without abandoning the techniques that make the base architecture compute-efficient. The design follows three ideas. First, we preserve AlphaHoldem's architecture for current-hand information. Second, we separate responsibilities between streams, so the transformer sees only tokens from completed hands while decisions

within the current hand remain with the original modules, which lets the benefit of cross-hand context be isolated by masking the transformer input at inference. Third, we enhance the K-best league with training against a population of opponents synthesized by a learned generative model of play styles, so the agent sees a continuously sampled variety of exploitable patterns, where earlier work relied on a small fixed set.

4.1 Encoding Poker Games

For each decision, AlphaExploitem assembles three input tensors from the running game state: the cards available to the agent, the action sequence of the current hand, and a structure containing all previously completed hands against the current opponent. For the first two we replicate AlphaHoldem's encoding of card and action information as separate one-hot arrays.

The history stream is a sequence of tokens recording every observable event from completed hands. Each token carries a type tag (agent action, opponent action, private card, community card, opponent card), so the encoder can route token types through separate embedding tables. Within a hand, tokens are laid down in the order events occur: the hero's private card at the deal, community cards as each street is revealed, the interleaved hero and opponent actions of each betting round, and, only at showdown, the opponent's revealed card, so a completed hand spans at most eleven tokens in Leduc. The running history is a fixed buffer of shape `(max_hands, max_tokens_per_hand)` with a write pointer, so the encoder always receives a padded, rectangular tensor. We exclude rewards from the stream since they can be inferred from the actions and cards, and in contrast to prior work we do not aggregate hand information.

4.2 Architecture

Figure 1 summarizes the overall architecture. The card and action tensors are each projected through a fully-connected layer, retaining AlphaHoldem's separation of the two streams.

Hierarchical history transformer. We encode the history stream with a two-level transformer (Vaswani et al., 2017) that mirrors poker's temporal structure, in which tokens group into hands and hands group into a session. A within-hand encoder is applied independently to each completed hand, vectorized across hands: it embeds the hand's tokens through type-specific tables, adds a learned positional embedding, runs self-attention over the tokens, and attention-pools the result to a summary vector h_i (hand summary token). An across-hand transformer then runs over the sequence $\{h_1, \ldots, h_M\}$, each summary tagged with a learned hand-position embedding, and attention-pools them through a learned query into one opponent-context vector. Positional information is therefore explicit at both levels, and the two-stage design keeps attention over a long session tractable, since self-attention operates within each short hand and then over the much shorter sequence of per-hand summaries, never over one flat stream. In contrast to prior work, we move the processing of an opponent's play into the model, away from hand-crafted statistics.

Current-hand exclusion. The transformer is restricted to tokens from hands already completed, and the current hand is handled exclusively by the current-hand encoders. This produces a structural decomposition in which the transformer attends to cross-hand information while the dedicated encoders handle in-hand reasoning, and it prevents the transformer from becoming the primary decision-maker within a single hand. The three encoded streams are finally concatenated, layer-normalized, and passed through a shared hidden network that splits into a softmax actor head and a scalar critic head.

4.3 Generating Opponents: a Style-VAE

Training an exploiter requires a supply of exploitable opponents, and prior work supplies them by hand, scripting a fixed set of weak archetypes. This is laborious to author, static, small relative to the variety of styles encountered in real play, and gives the agent no opponents beyond the ones an expert thought to write down. We replace the scripted set with a generative model of play styles that learns what styles look like from data and can synthesize an unlimited supply of coherent opponents on demand. We instantiate it as a conditional VAE (Kingma & Welling, 2014; Sohn et al., 2015), which can be read as amortized behavioral cloning (Pomerleau, 1991) conditioned on a latent style code, diagrammed in Figure 2.

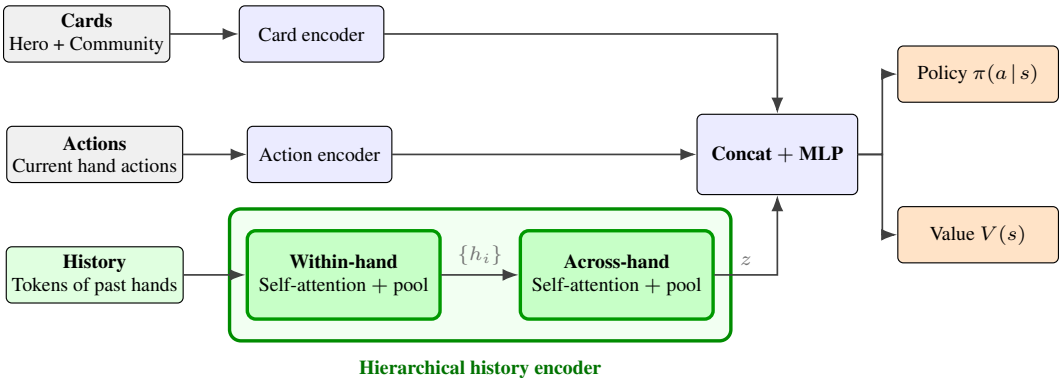

Figure 1: AlphaExploitem architecture (left to right). Three input streams, the cards available to the hero, the current hand's action history, and a tokenized record of all past hands against the same opponent, are fused by a shared MLP that splits into policy and value heads. Our architectural contribution is the hierarchical history encoder (highlighted): a within-hand transformer summarizes each completed past hand to a vector h_i, and an across-hand transformer attends over the per-hand summaries to produce a session-level context z.

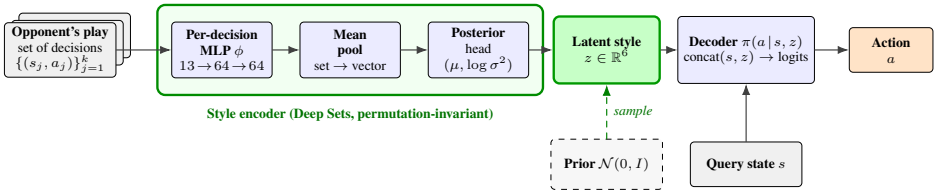

Figure 2: Style-VAE opponent generator (left to right). The generator is a conditional variational autoencoder (Kingma & Welling, 2014; Sohn et al., 2015) that performs amortized behavioral cloning conditioned on a latent style code z. A permutation-invariant set encoder (Deep Sets (Zaheer et al., 2017), highlighted) reads an unordered set of a single player's own-perspective decisions $\{(s_j, a_j)\}$ through a shared per-decision MLP followed by a mean-pool, then projects the pooled summary to the parameters $(\mu, \log \sigma^2)$ of a Gaussian posterior over the 6-dimensional latent style. The decoder is a z-conditioned policy $\pi(a \mid s, z)$ that predicts the next action for a query state s, and does not reconstruct hand histories. Training minimizes a β-VAE objective (Higgins et al., 2017), the behavioral-cloning negative log-likelihood of held-out decisions plus $\beta\,\mathrm{KL}(q(z \mid \cdot) \,\|\, \mathcal{N}(0, I))$, under a context/target split that forces z to encode a transferable style rather than a memorized trajectory. At generation time a fresh style is drawn from the prior $z \sim \mathcal{N}(0, I)$ (dashed) and decoded into a complete, queryable opponent.

Training corpus. The generator learns from the play of a small library of scripted opponents, each a rule-based policy conditioned on hand strength, board texture, and the legal-action mask. For Leduc the library consists of fourteen policies focused on amount aggression, indiscriminate calling, tight play, round-asymmetric aggression, slow-playing, betting-state and board conditioning, card polarization, inverted value betting, and erratic but locally consistent play, with full probability tables in Appendix C. These policies are never opponents the exploiter trains against. They exist only to produce the trajectories the generator learns from, and the generator's task is to interpolate beyond this finite, hand-authored set into a continuous space of styles. Each archetype is played as the hero against a standardized field, comprised of the exact Nash-equilibrium strategy (acting as a robust play anchor point) and a uniform distribution of the archetype library. This also mirrors how real hand histories arise, since human play is overwhelmingly generated by imperfect players facing each other with the occasional strong opponent mixed in. Logging every hero decision over 256 parallel environments for 200 steps yields about 350,000 decisions.

Set encoder. The generator operates on a player's stream of decisions, each pairing an observation with the action taken. The encoder is a permutation-invariant set network in the Deep Sets family (Zaheer et al., 2017): a small per-decision MLP is applied to each decision, the resulting vectors are mean-pooled, and the summary is projected to a six-dimensional latent style, so any number of hands

collapse to one fixed-size style vector. It reads only the acting player's own decisions from their own perspective, since identifying a style from a player's own choices never requires access to the cards hidden from that player (we do not provide full information, including the unseen opponent's cards, to the set encoder).

Decoder as a policy. The decoder concatenates an observation with a latent style code and maps the result to a distribution over the legal actions. It predicts the next action conditioned on the style and does not reconstruct hand histories, which is what makes a sampled latent immediately useful: each latent yields a policy that can be queried at any decision point and rolled out as an opponent, exactly like the scripted policies it replaces. The model minimizes a β-VAE (Higgins et al., 2017) objective under a context/target split, in which the latent is encoded from one slice of a player's decisions and the decoder is scored on a separate slice. A fresh opponent is then produced by drawing $z \sim \mathcal{N}(0, I)$, and sampling many latents yields a population covering the interior of the learned style space, including blends absent from the training data. We never supply the model with strategy tables, so the same procedure can learn the human style space from real hand histories.

4.4 Training

During each training epoch we generate trajectories from two opponent sources: the K-best self-play league and a freshly sampled opponent from the style-VAE. The league keeps the agent strong and hard to beat, while the generated opponents provide the exploitable targets from which it learns to maximize EV on weak opponents. We weight the mixture toward generated opponents, allocating 75% of each epoch's fixed environment budget to them and the remaining 25% to the league, since our main focus in exploitation. Training minibatches are sampled uniformly from the combined pool.

Reinforcement-learning objective. The agent is a shared-trunk actor-critic optimized with standard clipped PPO (Schulman et al., 2017). The per-hand reward is the BBs gained/lost by the hero in a hand. Advantages are estimated with Generalized Advantage Estimation (Schulman et al., 2016) using $\gamma = 0.999$ and $\lambda = 0.95$, bootstrapped from the critic and normalized within each minibatch. The probability ratio is clipped with $\epsilon = 0.1$, the critic is regressed on the discounted returns with a squared-error loss, and an entropy bonus keeps the policy stochastic, which matters because the equilibria of these games are mixed and an unregularized policy tends to collapse onto a deterministic strategy that is in turn trivially exploitable. Finally, we apply early stopping based on KL divergence for each epoch. We use standard PPO in place of AlphaHoldem's Trinal-Clip variant, which exists to tame the heavy-tailed reward variance of No-Limit Hold'em where all-in pots make per-hand outcomes effectively unbounded, whereas Leduc is a limit game with bounded stakes. Relative to AlphaHoldem we inherit the actor-critic formulation, the one-hot card and action representation, and the K-best league, and we change the opponent supply, add the cross-hand transformer, and revert the optimizer to standard PPO.

League self-play. Following AlphaHoldem (Zhao et al., 2022), we train against opponents sampled from a fixed-size league pool that maintains ELO ratings (Elo, 1978) updated from head-to-head outcomes weighted by the winning margin. At checkpoint intervals the current agent plays all league members and replaces the lowest-rated one if its ELO exceeds that member's.

Exposure to generated opponents. At the start of each epoch we draw a fresh latent style from the prior and pair it with the decoder. The latent is held fixed for the duration of the match, so the agent faces one consistent style across an entire session. Because the latent is resampled every epoch, no generated opponent is ever revisited, and over a full run the agent faces thousands of distinct styles, not a fixed and enumerable bank. This removes any opportunity to link responses to particular opponents and forces exploitation strategies that work across the style space itself. We assume these opponents follow a stationary policy within a match.

5 Experiments

AlphaExploitem is evaluated on Leduc against a test set of opponents sampled from the style-VAE's prior. Our results support four claims. (i) The agent exploits opponents drawn from the generator's latent space, at more than three times the rate a non-adaptive exact equilibrium achieves on the same set and roughly five times the rate of the AlphaHoldem-style baseline. (ii) It does so without losing

fundamental strength, converging closer to the equilibrium expectation than that baseline. (iii) The generated opponent population is necessary, since removing it collapses exploitation closer to what a non-adaptive equilibrium already reaches. (iv) The cross-hand transformer is necessary for that exploitation to be safe, since removing it leaves a single unconditional deviation that overfits to the opponent population and is itself more exploitable.

5.1 Experimental setup

AlphaExploitem is implemented in JAX (Bradbury et al., 2018) using Flax and Optax, with the Leduc environment provided by PGX (Koyamada et al., 2024), a JAX-native library of game simulators enabling fully vectorized, GPU-accelerated trajectory generation. Training uses AdamW with gradient clipping and KL-based early stopping. Full architectural and hyperparameter details are in Appendix I.

Training configurations. Every transformer configuration receives a wall-clock budget of eight hours on one NVIDIA A40 and every encoder-free one gets four hours, since the latter complete several times more epochs per hour and the same per-epoch budget of 128 parallel environments times 100 hands. When generated opponents are enabled they receive 75% of these environments and the league's per-member share shrinks to 25%, so no configuration sees more data per epoch than another.

Evaluation protocol. Every 100 epochs the current policy is evaluated in an exploiter mode with the full tokenized session history, and in a non-exploiter mode with the history masked so only the current-hand encoders inform decisions. The test set is a bank of 25 latents drawn once from the prior $\mathcal{N}(0, I)$ and decoded into policies, equal-weighted, and since training resamples a fresh latent every epoch, no evaluation opponent is ever trained against. Play against the equilibrium is measured at the same checkpoints, with the exact full-information-set Leduc Nash equilibrium played as an opponent. The fourteen hand-crafted archetypes the generator was trained on are reported separately in Appendix F. They are never played during training and enter it only indirectly, through the style-VAE's latent space.

Rewards are measured in big blinds per hand. Each seed's checkpoint series is smoothed with a debias-corrected exponential moving average ($\alpha = 0.8$), and curves show the seed-mean with a band giving the 95% confidence interval of that mean. The horizontal axis is environment interactions counted in hands, of which an epoch covers $12{,}800$ for the main configurations. Configurations do not all reach the same number of interactions, since the encoder-free ones complete several times more epochs per hour, so each figure is clipped to the range its plotted configurations share and quoted values are read at that figure's last common checkpoint. Figures comparing different sets of configurations therefore stop at different points, and a level quoted against one comparison may not exactly match the same level quoted against another.

5.2 Exploitation and play against the equilibrium

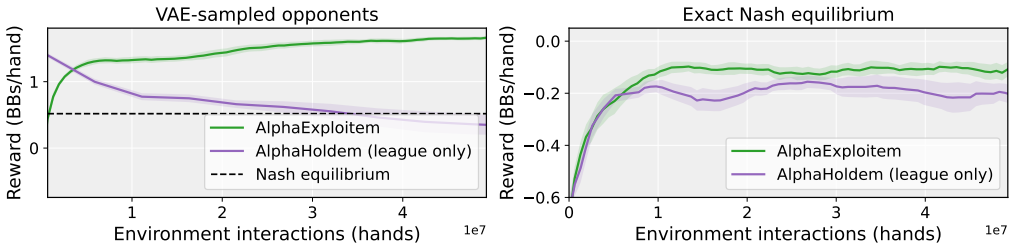

Figure 3: AlphaExploitem against the AlphaHoldem-style league-only baseline. Left: reward against the VAE-sampled test set, dashed line at the reward the exact Nash equilibrium obtains on the same set. Right: reward against the exact Nash-equilibrium policy.

Figure 3 reports both halves of the central claim.

On the test set (left plot), AlphaExploitem climbs monotonically to $+1.670 \pm 0.023$ BBs/hand. The baseline instead peaks very early, at roughly $+1.37$ within the first 10^6 hands, and then declines monotonically to $+0.332 \pm 0.139$ at the matched checkpoint, ending below the equilibrium reference. Pure league self-play drives the policy toward unexploitative equilibrium play, and the early rise is explained by the evolving quality of the league, which permits exploitation while the policies inside it are still flawed during early epochs.

The dashed reference line is the NE's performance against the test sets. AlphaExploitem clears it by more than a factor of three while the baseline ends below it, and since training resamples the latent every epoch, every evaluated latent is effectively held out, so what the agent has learned is a response to the space of styles and not to particular opponents.

Exploitation must not come at the cost of fundamental strength, so the right panel reports reward against the exact equilibrium. Independent policy-gradient self-play of the kind used here has no convergence guarantee to an equilibrium in imperfect-information games, unlike fictitious self-play and regret minimization, which converge in the average strategy (Heinrich et al., 2015; Zinkevich et al., 2007). Both agents approach the equilibrium expectation from below and neither is beaten badly, but AlphaExploitem converges closer, -0.116 ± 0.031 against the baseline's -0.224 ± 0.035 at the matched checkpoint. Training against a continuously refreshed population of exploitable opponents therefore does not soften the agent against strong play. The exploitative curriculum yields a policy that is both far more exploitative and closer to equilibrium than league self-play alone.

5.3 What each component contributes

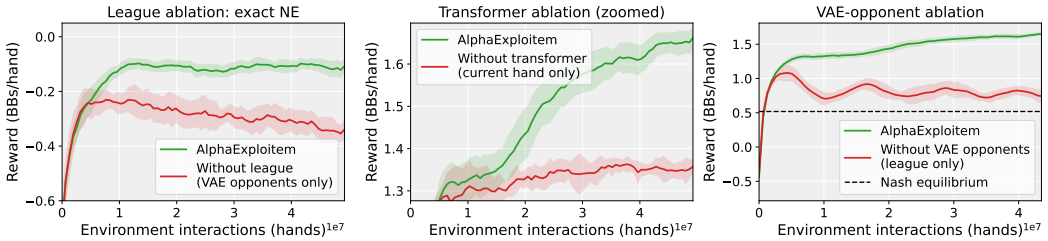

Figure 4: Component ablations against AlphaExploitem: the league removed (left, against the exact Nash equilibrium), the cross-hand transformer removed (middle), and the generated opponents removed (right).

Figure 4 separates the contributions of the three pieces of our approach, in the order league, transformer, generated opponents.

The league (left panel) is meant to act as the provider of robust play in our training pipeline. Removing it raises test-set reward to $+1.774 \pm 0.024$, expected since the entire budget then serves exploitable opponents. The consequence of removing the league is that against the exact equilibrium the league-free agent stalls at -0.344 ± 0.049 and drifts further away as training proceeds, a difference of $+0.227 \pm 0.054$ BBs/hand from AlphaExploitem. The league keeps the policy strong against good opponents at a cost of about 0.13 BBs/hand of exploitation.

The cross-hand transformer (middle panel) improves performance against the test set when coupled with the VAE population. Against the identical agent with both transformer levels removed, a markovian current-hand-only policy, reaches $+1.348 \pm 0.021$, a gap of $+0.301 \pm 0.032$ BBs/hand below AlphaExploitem. The model without the transformer overfits on the training population, trying to learn a general strategy rather than finding exploits, therefore it becomes a more exploitable policy in itself.

The generated opponents (right panel) are essential to our method. Removing them while keeping the transformer collapses exploitation: the agent peaks near $+1.1$ and decays to $+0.728 \pm 0.131$, a gap of $+0.921 \pm 0.132$ BBs/hand, far closer to the level a non-adaptive equilibrium achieves than to AlphaExploitem. The transformer alone cannot learn from a league-only curriculum: exploitable opponents have to be present in training for exploitative behavior to form at all, just as the transformer has to be present for it to form safely.

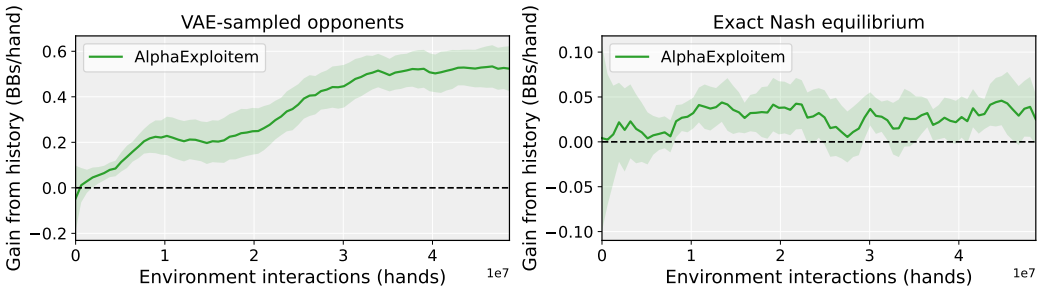

Figure 5: Cross-hand context at decision time, as the per-seed difference between one fixed set of trained AlphaExploitem parameters evaluated with the full session history and with it masked, so that in the masked case the policy acts from current-hand information alone. Left: against the VAE-sampled test set. Right: against the exact Nash equilibrium.

Cross-hand context at decision time (Figure 5). We analyze how much performance does the exploiter extract through the access to the history of the session. The gain from history, is $+0.533 \pm 0.093$ BBs/hand on the test set, still rising at the end of training, and $+0.439 \pm 0.073$ on the archetypes the generator was trained on. The same comparison against the exact equilibrium is positive as well, at $+0.029 \pm 0.033$ BBs/hand (paired, 95% CI), so context helps the agent both to exploit weak opponents and to adjust against solid opponents. AlphaExploitem without history is still a strong exploiter at $+1.116 \pm 0.089$, above both the baseline and the NE reference, the model converges to an slightly overfit (on the VAE curriculum) base-policy and leaves it to the transformer to decide whether to deviate from it.

6 Discussion and Conclusion

Discussion. Leduc was chosen because its size makes exact equilibria computable, unlike No-limit Hold'em and Omaha variants. Our claims are scoped to the distribution the agent is trained to exploit, the generator's prior and the archetypes it was fitted to. We further assume stationary opponents, and the claim that latent diversity yields exploitation of humans is untested on human data.

Conclusion. AlphaExploitem pairs a learned generator of play styles with a hierarchical cross-hand transformer, because each half contributes to the success of the other. The generator supplies opponents worth exploiting, without which the architecture has nothing to learn from. The transformer makes the resulting deviations conditional on the provided game history, without which the agent overfits to its training population, extracting less from that population while becoming prone to exploits due to the stationarity of a non-NE policy. Together they take more than three times the per-hand EV of an exact equilibrium on opponents drawn from the generator, and roughly five times that of the league-only baseline, while converging closer to the equilibrium than that baseline does. Since the generator needs only trajectories, the same method can be applied to open-sourced human hand histories such as PHH[1].

References

Darse Billings, Denis Papp, Jonathan Schaeffer, and Duane Szafron. Opponent modeling in poker. *Aaai/iaai*, 493(499):105, 1998.

Darse Billings, Aaron Davidson, Jonathan Schaeffer, and Duane Szafron. The challenge of poker. *Artificial Intelligence*, 134(1):201–240, 2002. ISSN 0004-3702. doi: https://doi.org/10.1016/S0004-3702(01)00130-8. URL `https://www.sciencedirect.com/science/article/pii/S0004370201001308`.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao

[1] https://zenodo.org/records/13997158

Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/jax-ml/jax.

Noam Brown and Tuomas Sandholm. Superhuman ai for heads-up no-limit poker: Libratus beats top professionals. *Science*, 359(6374):418–424, 2018. doi: 10.1126/science.aao1733. URL https://www.science.org/doi/abs/10.1126/science.aao1733.

Noam Brown and Tuomas Sandholm. Superhuman ai for multiplayer poker. *Science*, 365(6456): 885–890, 2019. doi: 10.1126/science.aay2400. URL https://www.science.org/doi/abs/10.1126/science.aay2400.

Noam Brown, Adam Lerer, Sam Gross, and Tuomas Sandholm. Deep counterfactual regret minimization. In *International conference on machine learning*, pp. 793–802. PMLR, 2019.

Noam Brown, Anton Bakhtin, Adam Lerer, and Qucheng Gong. Combining deep reinforcement learning and search for imperfect-information games. *Advances in neural information processing systems*, 33:17057–17069, 2020.

Arpad E. Elo. *The Rating of Chess Players, Past and Present*. Arco Pub., 1978.

Seth Frey, Dominic K. Albino, and Paul L. Williams. Synergistic information processing encrypts strategic reasoning in poker. *Cognitive Science*, 42(5):1457–1476, 2018. doi: https://doi.org/10.1111/cogs.12632. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/cogs.12632.

Jiaxian Guo, Bo Yang, Paul Yoo, Bill Yuchen Lin, Yusuke Iwasawa, and Yutaka Matsuo. Suspicion-Agent: Playing imperfect information games with theory of mind aware GPT-4, 2024. URL https://arxiv.org/abs/2309.17277.

Johannes Heinrich, Marc Lanctot, and David Silver. Fictitious self-play in extensive-form games. In *International conference on machine learning*, pp. 805–813. PMLR, 2015.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *5th International Conference on Learning Representations (ICLR)*, 2017. URL https://openreview.net/forum?id=Sy2fzU9gl.

Bret Hoehn, Finnegan Southey, Robert C Holte, and Valeriy Bulitko. Effective short-term opponent exploitation in simplified poker. In *AAAI*, volume 5, pp. 783–788, 2005.

Chenghao Huang, Yanbo Cao, Yinlong Wen, Tao Zhou, and Yanru Zhang. PokerGPT: An end-to-end lightweight solver for multi-player Texas Hold'em via large language model. *arXiv preprint arXiv:2401.06781*, 2024.

Yuheng Jing, Bingyun Liu, Kai Li, Yifan Zang, Haobo Fu, Qiang Fu, Junliang Xing, and Jian Cheng. Opponent modeling with in-context search. *Advances in Neural Information Processing Systems*, 37:61549–61591, 2024.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations (ICLR)*, 2014. URL https://arxiv.org/abs/1312.6114.

Sotetsu Koyamada, Shinri Okano, Soichiro Nishimori, Yu Murata, Keigo Habara, Haruka Kita, and Shin Ishii. Pgx: Hardware-accelerated parallel game simulators for reinforcement learning, 2024. URL https://arxiv.org/abs/2303.17503.

Michael Laskin, Luyu Wang, Junhyuk Oh, Emilio Parisotto, Stephen Spencer, Richie Steigerwald, DJ Strouse, Steven Hansen, Angelos Filos, Ethan Brooks, Maxime Gazeau, Himanshu Sahni, Satinder Singh, and Volodymyr Mnih. In-context reinforcement learning with algorithm distillation, 2022. URL https://arxiv.org/abs/2210.14215.

Shuxin Li, Chang Yang, Youzhi Zhang, Pengdeng Li, Xinrun Wang, Xiao Huang, Hau Chan, and Bo An. In-context exploiter for extensive-form games, 2024. URL https://arxiv.org/abs/2408.05575.

Xun Li and Risto Miikkulainen. Evolving adaptive LSTM poker players for effective opponent exploitation. In *AAAI Workshops*, 2017.

Xun Li and Risto Miikkulainen. Dynamic adaptation and opponent exploitation in computer poker. In *AAAI-18 Workshop for Imperfect Information Games*, New Orleans, LA, USA, 2018a. URL http://www.cs.utexas.edu/users/ai-lab?li:aaaiws2018.

Xun Li and Risto Miikkulainen. Opponent modeling and exploitation in poker using evolved recurrent neural networks. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 189–196, 2018b.

Hsieh-Ting Lin and Tsung-Yu Hou. Readable minds: Emergent theory-of-mind-like behavior in LLM poker agents. *arXiv preprint arXiv:2604.04157*, 2026.

Mingyang Liu, Chengjie Wu, Qihan Liu, Yansen Jing, Jun Yang, Pingzhong Tang, and Chongjie Zhang. Safe opponent-exploitation subgame refinement. *Advances in Neural Information Processing Systems*, 35:27610–27622, 2022.

Matej Moravčík, Martin Schmid, Neil Burch, Viliam Lisý, Dustin Morrill, Nolan Bard, Trevor Davis, Kevin Waugh, Michael Johanson, and Michael Bowling. Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*, 356(6337):508–513, 2017. doi: 10.1126/science.aam6960. URL https://www.science.org/doi/abs/10.1126/science.aam6960.

John F. Nash. Equilibrium points in ¡i¿n¡/i¿-person games. *Proceedings of the National Academy of Sciences*, 36(1):48–49, 1950. doi: 10.1073/pnas.36.1.48. URL https://www.pnas.org/doi/abs/10.1073/pnas.36.1.48.

Hukukane Nikaidô and Kazuo Isoda. Note on non-cooperative convex games. *Pacific Journal of Mathematics*, 1955.

Yaniv Oren, Viliam Vadocz, Matthijs T. J. Spaan, and Wendelin Böhmer. Epistemic monte carlo tree search. In *The Thirteenth International Conference on Learning Representations (ICLR)*, pp. 75835–75861, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/bcbd670951a6dede2123961de19e5ea3-Paper-Conference.pdf.

Dean A. Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural Computation*, 3(1):88–97, 1991. doi: 10.1162/neco.1991.3.1.88.

Marc-Antoine Provost, Nejc Ilenic, Christopher Solinas, and Philippe Beardsell. GTO wizard benchmark. *arXiv preprint arXiv:2603.23660*, 2026.

Jonathan Rubin and Ian Watson. Computer poker: A review. *Artificial Intelligence*, 175(5):958–987, 2011. ISSN 0004-3702. doi: https://doi.org/10.1016/j.artint.2010.12.005. URL https://www.sciencedirect.com/science/article/pii/S0004370211000191. Special Review Issue.

Martin Schmid, Matej Moravčík, Neil Burch, Rudolf Kadlec, Josh Davidson, Kevin Waugh, Nolan Bard, Finbarr Timbers, Marc Lanctot, G Zacharias Holland, et al. Student of games: A unified learning algorithm for both perfect and imperfect information games. *Science Advances*, 9(46): eadg3256, 2023.

John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *4th International Conference on Learning Representations, ICLR 2016*, 2016. URL http://arxiv.org/abs/1506.02438.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. URL http://arxiv.org/abs/1707.06347.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 28, 2015.

Finnegan Southey, Michael P Bowling, Bryce Larson, Carmelo Piccione, Neil Burch, Darse Billings, and Chris Rayner. Bayes' bluff: Opponent modelling in poker. *arXiv preprint arXiv:1207.1411*, 2012.

Eric Steinberger, Adam Lerer, and Noam Brown. DREAM: Deep regret minimization with advantage baselines and model-free learning. *arXiv preprint arXiv:2006.10410*, 2020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL http://arxiv.org/abs/1706.03762.

Oriol Vinyals, Igor Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H. Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan Horgan, Manuel Kroiss, Ivo Danihelka, Aja Huang, Laurent Sifre, Trevor Cai, John P. Agapiou, Max Jaderberg, Alexander S. Vezhnevets, Rémi Leblond, Tobias Pohlen, Valentin Dalibard, David Budden, Yury Sulsky, James Molloy, Tom L. Paine, Caglar Gulcehre, Ziyu Wang, Tobias Pfaff, Yuhuai Wu, Roman Ring, Dani Yogatama, Dario Wünsch, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy Lillicrap, Koray Kavukcuoglu, Demis Hassabis, Chris Apps, and David Silver. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019. doi: 10.1038/s41586-019-1724-z. URL https://doi.org/10.1038/s41586-019-1724-z.

Zhe Wu, Kai Li, Enmin Zhao, Hang Xu, Meng Zhang, Haobo Fu, Bo An, and Junliang Xing. L2e: Learning to exploit your opponent, 2021. URL https://arxiv.org/abs/2102.09381.

Jiahui Xu, Jing Chen, and Shaofei Chen. Efficient opponent exploitation in no-limit texas hold'em poker: A neuroevolutionary method combined with reinforcement learning. *Electronics*, 10(17), 2021. ISSN 2079-9292. doi: 10.3390/electronics10172087. URL https://www.mdpi.com/2079-9292/10/17/2087.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabás Póczos, Ruslan Salakhutdinov, and Alexander J. Smola. Deep sets. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017. URL https://arxiv.org/abs/1703.06114.

Enmin Zhao, Renye Yan, Jinqiu Li, Kai Li, and Junliang Xing. Alphaholdem: High-performance artificial intelligence for heads-up no-limit poker via end-to-end reinforcement learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(4):4689–4697, Jun. 2022. doi: 10.1609/aaai.v36i4.20394. URL https://ojs.aaai.org/index.php/AAAI/article/view/20394.

Richard Zhuang, Akshat Gupta, Richard Yang, Aniket Rahane, Zhengyu Li, and Gopala Anumanchipalli. PokerBench: Training large language models to become professional poker players. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 26175–26182, 2025.

Martin Zinkevich, Michael Johanson, Michael Bowling, and Carmelo Piccione. Regret minimization in games with incomplete information. *Advances in neural information processing systems*, 20, 2007.

A Limitations and Assumptions

Assumptions. All evaluation opponents present stationary policies. The model is exposed to non-stationary opponents during league play, since trajectories are generated against the past iterations of the exploiter agent, but we do not integrate evaluation metrics for non-stationary conditions. Each sampled style is held fixed for the duration of a match. We additionally assume heads-up play with an unlimited supply of money and a fixed number of hands per match.

Finite time horizon. AlphaExploitem cannot play indefinitely without losing information about old hands, since the history buffer has a fixed size. A recurrent state updated by the latent dimensions of the cross-hand transformer would be a possible solution.

No search. The agent reacts to the opponent in a single forward pass. The strongest equilibrium systems combine learning with decision-time re-solving (Moravčík et al., 2017; Brown & Sandholm, 2018; Brown et al., 2020), and a next step is to construct the current subgame with the opponent's range biased by the transformer's read and solve it, using ReBeL's Public Belief States as a way to convert the partial information setting to a perfect information one. The encoder's read of the opponent is itself a learned model estimated from a handful of hands, so committing to it on too little evidence is a real risk. A search that propagates the epistemic uncertainty of learned models through the tree, as Epistemic MCTS does for value and dynamics models (Oren et al., 2025), would let the agent hedge toward equilibrium while the history of played hands is short.

Computational efficiency. The training algorithm inherits the parallelizability of the transformer (Vaswani et al., 2017) architecture and the data inefficiency of PPO (Schulman et al., 2017). Multi-GPU training is required for larger contexts.

B Player Types

Real-world poker styles are conventionally placed on two axes: how often a player enters a hand (tight versus loose) and how often they raise versus call (passive versus aggressive). The four resulting quadrants name the archetypes used throughout the poker community: tight-aggressive (TAG, the strong style), loose-aggressive (LAG, profitable but high-variance), tight-passive (the nit or rock, under-bluffs and over-folds), and loose-passive (the calling station, which calls down indiscriminately). Except for the TAG, which is balanced play that an NE-approximating agent already handles, each of these styles corresponds to a stationary mixed policy with predictable, exploitable structure, which is the kind of suboptimal play our agent is trained to detect and exploit. We instantiate the exploitable archetypes as scripted policies with explicit action-probability tables (Appendix C) and use them as the source of the generator's training trajectories, never as training opponents for the agent itself.

C Opponent Specifications

Each archetype maps the current decision to a probability vector over (call/check, raise/bet, fold), conditioned on at most four features: the private card, whether the private card pairs the community card, whether the flop has been dealt, and whether a bet is pending, since fold is only legal in that case. Cards are ranked Jack $<$ Queen $<$ King, and "pair" means the private card matches the community card.

- **Calling Station**: calls with probability 1 at every decision.
- **Maniac**: $(0.05, 0.95, 0)$ at every decision.
- **Folder**: $(0.05, 0, 0.95)$ at every decision.
- **Uniform Random**: uniform over the legal actions.
- **Nit**: pre-flop King $(0.3, 0.7, 0)$, Queen $(0.8, 0, 0.2)$, Jack $(0, 0, 1)$. Post-flop pair $(0.3, 0.7, 0)$, unpaired King $(0.7, 0.1, 0.2)$, otherwise $(0, 0, 1)$.
- **Rock**: pre-flop Jack $(0.3, 0, 0.7)$, otherwise $(0.95, 0, 0.05)$. Post-flop pair $(0.8, 0.15, 0.05)$, unpaired King $(0.6, 0, 0.4)$, otherwise $(0.1, 0, 0.9)$.
- **Loose-Aggressive**: pre-flop $(0.25, 0.7, 0.05)$. Post-flop pair $(0.05, 0.95, 0)$, otherwise $(0.3, 0.55, 0.15)$.
- **Aggressive Fish**: pre-flop $(0.1, 0.9, 0)$. Post-flop pair $(0.05, 0.95, 0)$, otherwise $(0.1, 0.05, 0.85)$.
- **Trapper**: pre-flop $(0.9, 0.05, 0.05)$. Post-flop pair $(0.7, 0.3, 0)$, unpaired King $(0.6, 0.1, 0.3)$, otherwise $(0.1, 0, 0.9)$.
- **Check-Bluffer**: pre-flop Jack $(0.6, 0, 0.4)$, Queen $(0.85, 0.05, 0.1)$, King $(0.3, 0.7, 0)$. Post-flop pair $(0.1, 0.9, 0)$, unpaired and facing a bet $(0.1, 0, 0.9)$, unpaired with no bet pending $(0, 1, 0)$, which is a guaranteed bluff bet.
- **Board Reader**: pre-flop $(0.6, 0.3, 0.1)$. Post-flop conditioned on the community card: Jack $(0.1, 0.8, 0.1)$, Queen $(0.5, 0.35, 0.15)$, King $(0.85, 0.05, 0.1)$.

- **King Bully**: on both streets, King $(0.05, 0.9, 0.05)$, otherwise $(0.8, 0.05, 0.15)$.
- **Pair Bluffer**: pre-flop $(0.4, 0.5, 0.1)$. Post-flop pair $(0.85, 0.1, 0.05)$, otherwise $(0.1, 0.8, 0.1)$.
- **Chaotic**: at each decision, commits uniformly at random to one pure mode, always-call, always-raise, or always-fold.

D Nash Equilibrium Reference Values

Our reference Nash strategy for Leduc is a tabulated equilibrium with one row per information set, computed via counterfactual regret minimization to convergence. Reference values are computed by playing this exact strategy for 100,000 hands against each opponent, seat-averaged, with the big blind equal to the ante. As a sanity check, the NE playing itself yields $+0.009$ BBs/hand, consistent with the zero expectation.

The resulting set-averaged NE rewards are $+0.518$ BBs/hand on the VAE-sampled test set of 25 prior draws and $+0.476$ on the fourteen archetypes used to train the generator. The test-set value is the dashed reference line in Figures 3 and 4, and the archetype value is the dashed line in Figure 6.

Across the 25 test-set opponents the NE reward has a standard deviation of 0.180 and a range of $+0.176$ to $+0.782$, so the set spans a wide range of exploitability. The exact NE earns $+0.805$ against Board Reader, $+0.714$ against Uniform Random, and $+0.700$ against Chaotic at the top, and $+0.219$ against Aggressive Fish and $+0.077$ against Nit at the bottom, since cautious opponents leave little room to extract value.

E Cross-hand Context at Decision Time

Architecture, training, and parameters are identical by construction between the two evaluations differenced in Figure 5. The only difference is whether the transformer sees the prior-hand tokens. On the test set the gap emerges early, around 5×10^6 hands, and grows steadily to $+0.533 \pm 0.093$ BBs/hand. Against the exact equilibrium it is briefly negative before turning over at around 10^7 hands and settling at $+0.029 \pm 0.033$, so the two rise together and do not trade off. Because nothing in training forces the same parameters to behave as a safe equilibrium approximator when context is removed, the masked level is simply what this trained policy produces from card and current-hand action observations alone, and the gap should be read as the decision-time contribution of context under this specific training regime.

F Results on the Hand-crafted Archetypes

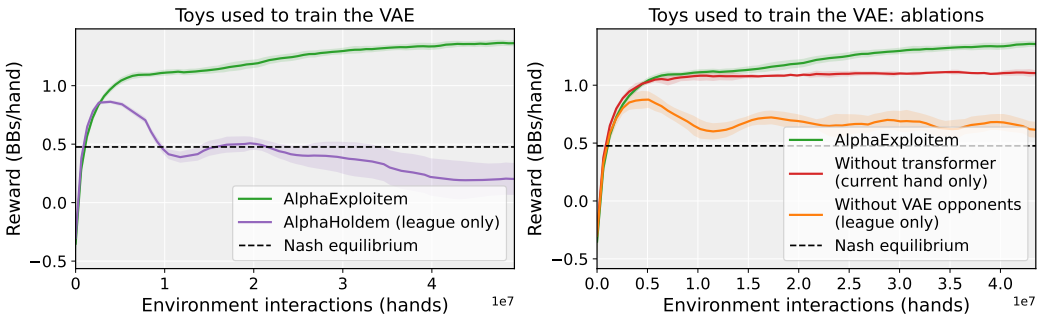

Figure 6: The fourteen archetypes that produced the generator's training trajectories, which the agent never plays during training and which enter training only indirectly through the generator's latent space. Left: AlphaExploitem against the AlphaHoldem-style baseline. Right: the transformer and generated-opponent ablations. The dashed line is the exact NE reward against this set (Appendix D). Seed-mean with 95% CI bands, $n = 16$.

At matched interaction counts AlphaExploitem reaches $+1.361 \pm 0.029$ BBs/hand against the generator's training archetypes while the baseline reaches $+0.195 \pm 0.132$. The exact NE reference on this set is $+0.476$, so AlphaExploitem clears it by a factor of nearly three while the baseline ends well below it. These fourteen policies are never played during training and reach the agent only through the latent space the generator distilled them into, so the result measures whether exploiting draws from the generator's prior transfers back to the styles that prior was built from.

The ablations (right panel) provide similar effects as they did on the test set. The transformer's training-time advantage over the transformer-free model is $+0.245 \pm 0.031$ BBs/hand, and removing the generated opponents costs $+0.757 \pm 0.078$.

Evaluating the same parameters without history on this set (not plotted) gives a gain from history of $+0.439 \pm 0.073$ BBs/hand, against $+0.533 \pm 0.093$ on the test set.

G League-to-Generated Budget Split

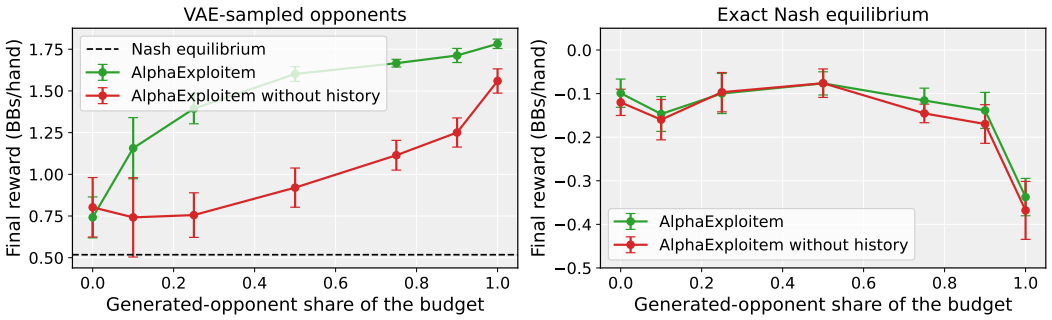

Figure 7: Share of the per-epoch environment budget given to generated opponents, with the remainder going to the league. Left: final reward on the VAE-sampled test set, with and without the session history, dashed line at the exact NE reward on this set. Right: final reward against the exact Nash equilibrium. The endpoints 0.0 and 1.0 are the league-only and generated-only configurations of the main ablation. Mean with 95% CI bands.

The main configuration gives generated opponents 75% of the replay-buffer space. Changing the mixture allows us to understand how each of the two components impacts the model's performance. Exploitation rises with it, from $+0.742 \pm 0.123$ BBs/hand at no generated opponents to $+1.782 \pm 0.028$ at no league. Play against the exact equilibrium is flat across almost the whole range, and collapses only when the league has a share below 25%. Shifting environments toward generated opponents therefore buys exploitation, as long as the league retains some share.

The gain from cross-hand context behaves differently, peaking at intermediate mixtures at $+0.681 \pm 0.145$ for a 50% share and falling to $+0.223 \pm 0.054$ at 100%. Heavier exposure to generated opponents makes the policy without history stronger by itself, slightly overfit, so the measurable contribution of context shrinks.

H Model Size

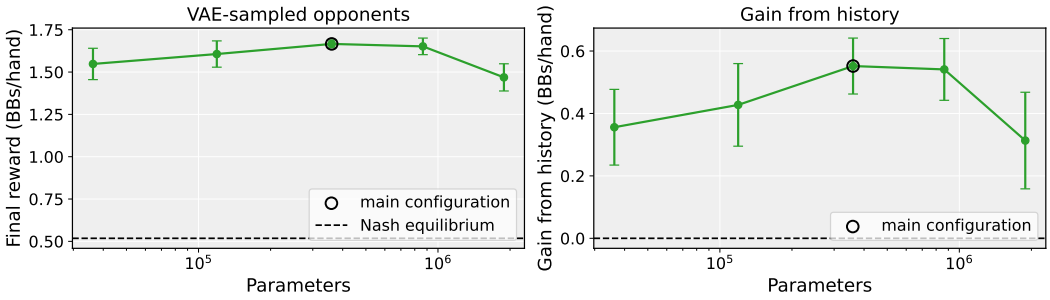

Figure 8: Model size at a fixed wall-clock budget. Trunk width, embedding dimension and encoder depth vary together. The horizontal axis is the resulting parameter count. Left: final reward on the VAE-sampled test set, dashed line at the exact NE reward on this set. Right: the gain from cross-hand context. The main configuration is circled. Mean with 95% CI bands.

Across a $52\times$ range of parameter counts, from 36K to 1.9M, both exploitation and the gain from context peak at or near the main configuration of 360K parameters. Reward rises from $+1.548 \pm 0.093$ BBs/hand at the smallest model to $+1.666 \pm 0.023$ at the main one, is unchanged at $+1.652 \pm 0.049$ for the next size up, and falls to $+1.469 \pm 0.080$ at the largest. The gain from context follows the same shape, from $+0.356 \pm 0.121$ at the smallest through $+0.552 \pm 0.090$ at the main configuration to $+0.313 \pm 0.155$ at the largest.

I Implementation and Hyperparameters

All experiments are implemented in JAX using Flax and Optax, with trajectory generation fully vectorized on a single GPU. Every transformer configuration of the main ablation was trained for a fixed wall-clock budget of eight hours on one NVIDIA A40 (46 GB) and every encoder-free one for four hours.

Table 1: Training hyperparameters, shared across the main-ablation configurations. Only the flags controlling the transformer, the league, and the generated opponents differ between groups.

Optimization (PPO)	
Optimizer	AdamW, learning rate 10^{-4}
Clipping parameter ϵ	0.1
Value-loss coefficient	0.05
Entropy coefficient	0.05
Discount γ and GAE λ	0.999 and 0.95
Updates per epoch	4 steps $\times$ 4 minibatches of 64 samples
KL early-stopping threshold	≈ 0.03
Numerical precision	bfloat16 activations, float32 parameters
Data generation	
Parallel environments	128 (16 per league member $\times$ league size 8)
Match length (hands per epoch per env)	100
Evaluation cadence	every 100 epochs
Epoch target and stopping criterion	30,000 epochs
Opponent sources	
League	K-best pool of size 8, ELO-gated replacement
Generated opponents	style-VAE decoder (latent dimension 6), one fresh $z \sim \mathcal{N}(0, I)$ per epoch, fixed within the epoch
Environment budget split	75% generated, 25% league. League-only and generated-only configurations receive the full 128-environment budget
Evaluation	
Test set	25 prior draws, 16 environments per latent
Hand-crafted sets	32 environments per opponent
Exact-NE evaluation	512 matches of 100 hands per checkpoint

The transformer-free configurations remove both transformer levels entirely, which reduces the parameter count by roughly an order of magnitude while leaving all remaining dimensions unchanged.

AI USE STATEMENT

In this work, we used generative AI tools as a coding and writing assistant. Concretely, AI assistance was used in three places. First, for implementation: parts of the training, evaluation and analysis code were written or refactored with AI assistance. Second, for the analysis and figures: the scripts that aggregate the logged experiment data into the reported statistics and the figures in this paper were produced with AI assistance. Third, for the manuscript: portions of the text were drafted and edited with AI assistance.

All reported numbers are computed by this code from logged experiment data. No experimental result, figure or citation in this paper was generated by a language model.

We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI. All generated plots match their correspondents in the wandb plots generated by the logged data.