

One Recipe, Many Experts: An Efficient Path to Mastering the Pokémon TCG

Sebastian Lashmet and Abhyuday · September 13, 2026

Exported from the supplied Kaggle webpage and attached figures.

TL;DR

- Approach. One small network reads every card from its rules, so the same design can learn to pilot any deck. Each deck's expert starts by imitating strong ladder players, then improves in a league of expert opponents that keep beating each other. At play time, the game engine checks what a network should not have to guess.
- Result. Top 2.3% of 6,807 teams and 57.8% wins over 2,244 final ladder games, above what our rating predicts against every level of opponent and nearly every deck. Our final agent beat our best pure self-play agent 58-42, and once the league exists, a new deck expert costs about 12 GPU-hours.
- What's next. The same recipe applies to any deck. Our limits are overall strength and one kind of slow, defensive opponent, both of which point to the same next steps: longer training, longer-range planning and a more flexible deck.

1. What makes this game hard for a program

Hidden cards are uncertainty, not bluffing. An opponent's hand is guessable less than 3% of the time, but 70% of ladder decklists are exact copies of known lists, so what an opponent holds can be sampled from those lists. With little deception to model, we skipped game-theoretic solvers and instead sample likely hidden cards and reason over those samples.

The value of a move shows up late. A turn is a chain of many small choices, and whether they were good often only becomes clear after the opponent responds.

The field. Early in the competition most teams played hand-written rules; by the end, the strongest teams were mostly learned agents. Even at the top, results depended as much on which deck was chosen and how well it was piloted as on the method. So we worked on both: the deck, and a pilot that can learn any deck.

2. The deck: choose what a bot can pilot

Sebastian, our TCG expert, set the philosophy: bots play best with straightforward decks where "everything is built off the concept of the prize race", and the goal is to "distill this game into as much solitaire as you possibly can".

We chose Alakazam, whose attack, Powerful Hand, places two damage counters (20 damage) on the opponent for every card in our hand. That turns most decisions into counting: draw, keep cards, and attack when the hand is big enough. It is also a single-prize attacker that can knock out the game's big two-prize Pokémon, so it wins the race for prizes. Its one natural enemy is anything that blocks damage counters, such as Mist Energy.

Our 60-card Alakazam list, grouped by job.

Attackers 11 4x Abra 4x Kadabra 3x Alakazam	Draw 8 4x Dunsparce 3x Dudunsparce 1x Fezandipiti ex	Find cards 20 4x Dawn 4x Hilda 4x Poké Pad 4x Buddy-Buddy Poffin 4x Rare Candy	Disrupt 4 2x Xerosic's Machinations 2x Boss's Orders
Recover 5 3x Night Stretcher 1x Lana's Aid 1x Sacred Ash	Beat walls 2 2x Enhanced Hammer	Protect bench 3 3x Battle Cage	Energy 7 4x Telepath Psychic Energy 2x Basic Psychic Energy 1x Enriching Energy

Key card	Job	How the bot uses it
Abra, Kadabra, Alakazam (4-4-3)	attackers	Kadabra and Alakazam draw cards as they evolve; attack once the hand covers the knockout
Dudunsparce (3)	draw engine	draws three and shuffles itself away, growing the hand without clogging the bench
Xerosic's Machinations (2)	disruption	shrinks the opponent's hand, which decides the Alakazam mirror
Enhanced Hammer (2)	wall answer	strips the Mist or Rock Fighting Energy that blocks our damage counters
Battle Cage (3)	protection	stops damage counters landing on our fragile benched Abra and Kadabra
Night Stretcher (3), Sacred Ash (1)	recovery	puts cards back so a hand-hungry deck does not run out

We only changed the list when the same trained network, playing both sides, measured the change as an improvement:

Change	Result	Kept?
+2 Xerosic's Machinations, -2 Enhanced Hammer	mirror match 69-31 over 1,200 games	yes
+1 Battle Cage, -1 Boss's Orders	+22 points vs Mega Froslass, mirror unchanged (60 games per matchup)	yes
Genesect + Lucky Helmet + 4 Battle Cage (anti-spread package)	mirror dropped to 31% over 240 games	no
Dedenne card-recycling loop for wall decks	wall matchup 82% early, 16% after training, winning only by decking opponents out	no

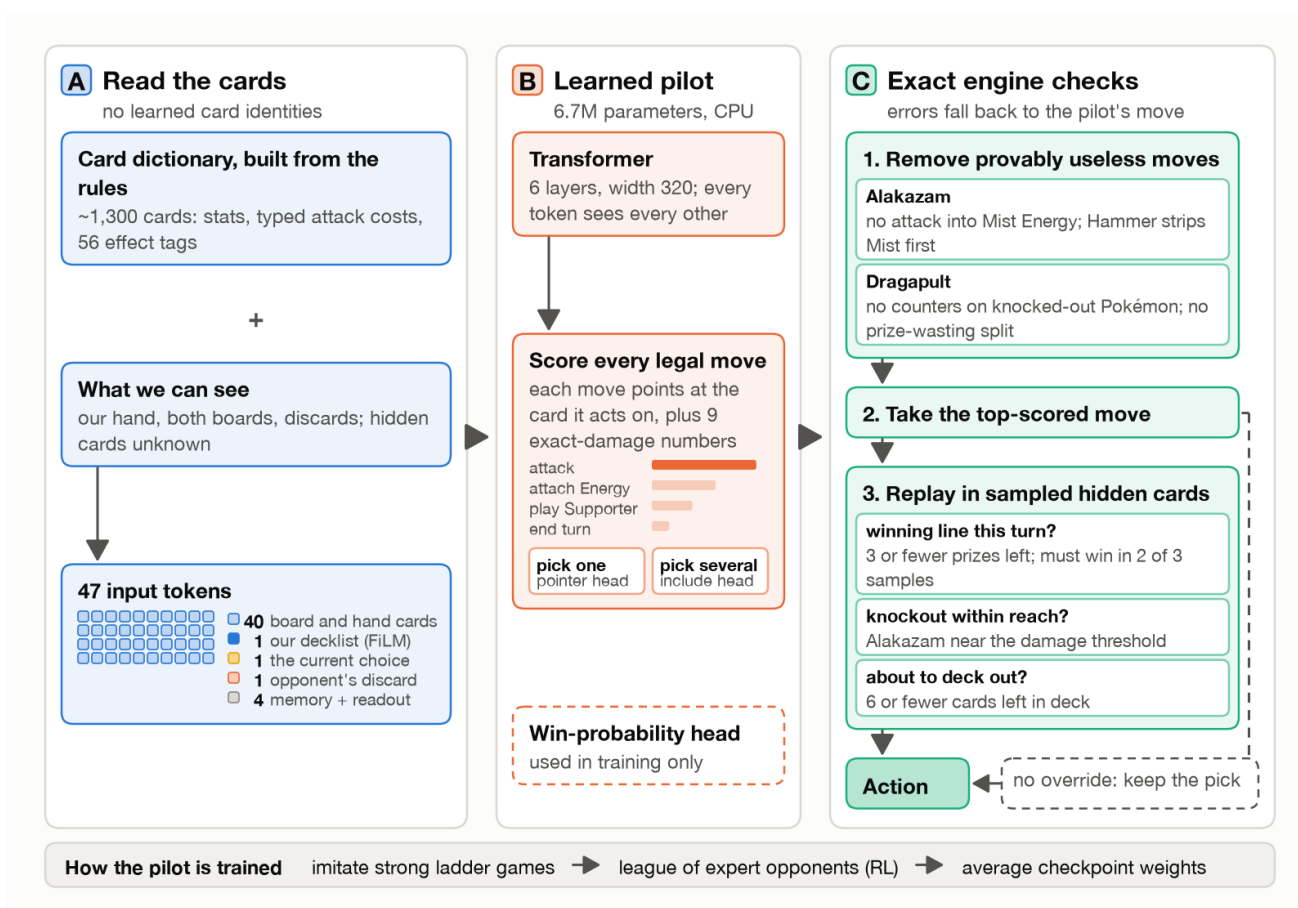
The pattern: cards that sharpen our own plan, like Xerosic in the mirror, paid off, while tech aimed at other decks made the bot worse. We believe two things were at play: every situational card is a new decision the pilot has to learn to time, and in a deck whose damage is its hand size, each slot that does not draw or protect the plan weakens the main attack.

3. The pilot: read cards from their rules

Point at what you act on. Our first agents could not tell apart moves like "attach Energy to this Abra" and "attach Energy to that Abra", because moves were described by card name; 48% of decisions contained options that looked identical. Describing each move by the board piece it acts on fixed this, and let the network score whatever legal moves the game offers.

Describe cards by their rules, not their names. Learning a separate identity for every card only works for cards seen often in training: after 82 million training steps, only 46 of about 1,300 card identities had changed at all. Instead, each card is described by what its rules do: HP, attack costs, damage and effects. A new card is just a new description; our agent played a card it had never trained with in 80% of its games.

How one decision is made.



The network is small (6.7M parameters) and makes a decision in tens of milliseconds on a CPU.

4. The league: opponents that force each other to evolve

Self-play alone hits a wall. Pure self-play, with no human data, produced a solid agent. But a bot that only ever plays one deck only ever meets badly played versions of every other deck, because its opponents are copies of itself; ours won 70% against weaker players but only 17% of 23 games against players rated 950 or higher.

So we built a league of experts. The league is a pool of strong pilots for the decks that matter, each training to beat the others. When one improves, the others have to adapt, so the whole league keeps getting stronger. We trained pilots for

about 15 decks, six of them strong enough to seriously test our main agent, and sampled each opponent by how common its deck is on the ladder and how often it beats the learner.

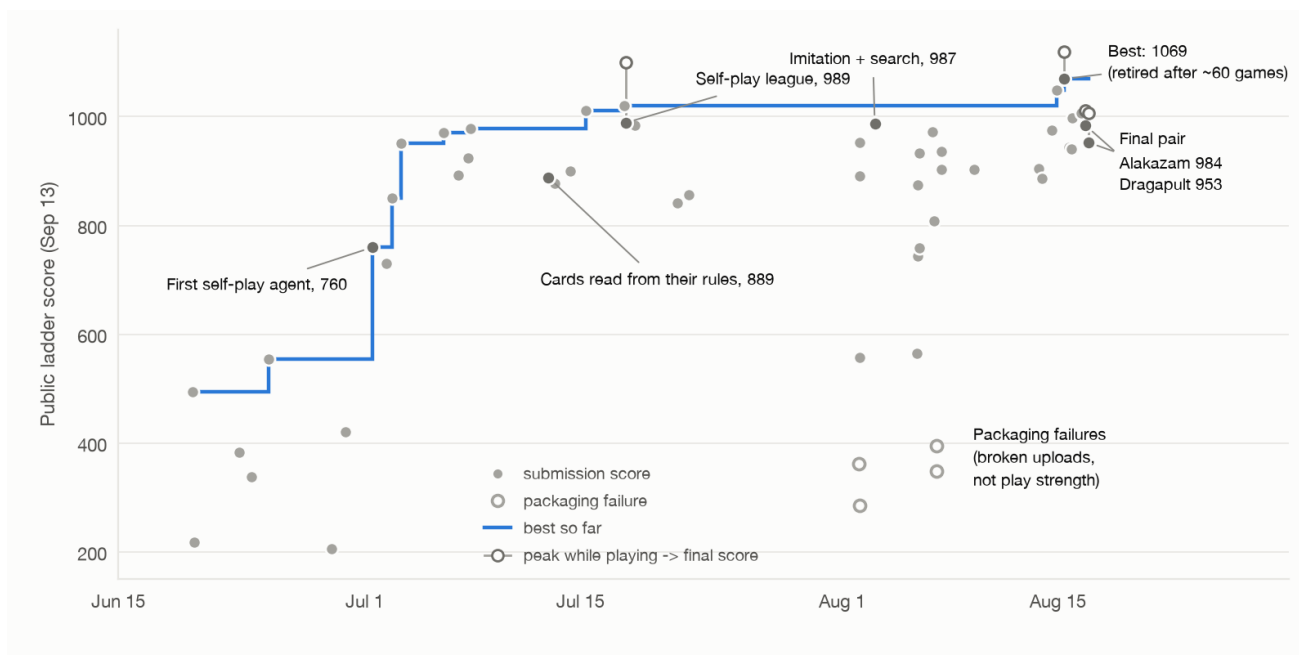
Human games are a head start, not the teacher. A new pilot first imitates strong ladder players of its deck. On its own that pilot is weaker than our best self-play agent, but it starts about 21 points ahead of learning from scratch with the same compute. Reinforcement learning inside the league (improving from its own wins and losses) then takes over, deliberately gently, so it refines what was imitated instead of overwriting it. Finally, we average the weights of checkpoints from one run, which keeps skills that different training phases trade away. The league idea follows AlphaStar and checkpoint averaging follows "model soups"; what is new is that rule-described cards let one recipe serve any deck.

Stage	What it does	Cost (one L4 GPU)	What it bought
Imitation	copy strong ladder players, weighted toward the target deck	~3 GPU-hours	~21 points over starting from scratch
League training	play the league and learn from results, one phase at a time	2-3 GPU-hours per phase	Mega Lucario pilot 19% -> 63% vs our best RL agent after three phases (240 games)
Checkpoint averaging	average the weights of checkpoints from one run	minutes	at least as good as the better checkpoint in every matchup
New deck (Dragapult), given the league	imitation + two league phases	~12 GPU-hours	56% vs our main Alakazam agent (200 games)
Alakazam line, all-in	including teacher decks and side experiments	~160 GPU-hours	final submission
Whole two-month program		~870 GPU-hours (~\$1,000)	

Ladder scores are relative to a field that kept getting stronger, so we compare eras head to head:

Agent vs our July pure self-play agent	Win rate	Games
Imitation only	31%	704
Imitation + league training (Alakazam / Mega Lucario)	52% / 75%	68 / 60
Final Alakazam agent	58%	100

Ladder score of every submission. Ratings are relative to a field that keeps improving, so scores drift: hollow markers show each labelled agent's peak while it was still playing.



Dragapult: a sparring partner. In August, Dragapult rose on the ladder, which gave us plenty of strong games to imitate, and it beats Alakazam. That made it exactly the opponent our league needed, so we trained one quickly; it also became our second submission.

Share of ladder decks over time.

5. Search: what we learned, and why we shipped without it

With the imitation-trained pilot, search helped, and where it judged a move mattered most. Each candidate move is played out in the same sampled hidden cards, so luck cancels out; this sampling is standard for imperfect-information games, and our finding is about where to judge. Judging at the end of our turn was a big step up but every variant stalled there, judging mid-turn did worse than that, and letting the pilot play out the opponent's reply first added another 10 points.

How moves were chosen (imitation-trained pilot)	Win rate vs July self-play agent	Games
Pilot alone	38%	60
Search, judged at the end of our turn	55%	210
Search, judged after the opponent's reply	65%	300

Sampling hidden cards, and where a move is judged.

SAMPLING WHAT IS HIDDEN

Guess the opponent's list

70% of ladder lists are exact copies of known lists

Sample hidden cards

same samples for every candidate move, so luck cancels

Average over samples

more samples barely changed decisions

WHEN TO JUDGE A MOVE

how much the position's value moves



Our turn

Opponent's reply

Our next turn

Inside our turn

worse

than judging at the end of the turn

At end of our turn

55%

big step up, then stalls

After opponent's reply

65%

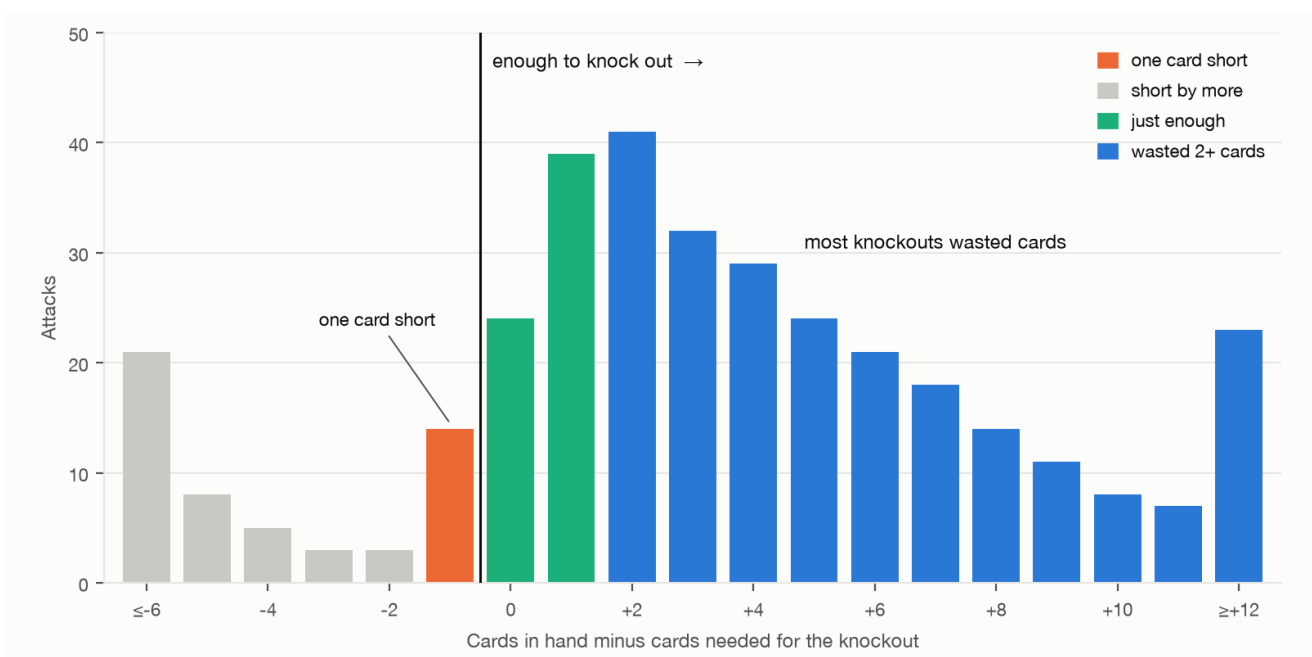
+10 points more

Win rate against our July self-play agent. The imitation-trained pilot alone scored 38%.

After league training, the pilot had learned that lookahead itself. Wrapping the same search around our strongest league-trained network cut its win rate against our search champion from 52% to 38%. Replaying real decisions with perfect information showed an RL-trained pilot already chose the best move two times in three, leaving little for any ranker to add. So we shipped the trained pilot without search.

What remained were counting mistakes, and the engine can count exactly. Sebastian's game reviews found most big mistakes came from miscounting the hand: in ladder replays, 14 attacks fell exactly one card short of a knockout, and 78% of knockouts wasted two or more cards that could have been played first.

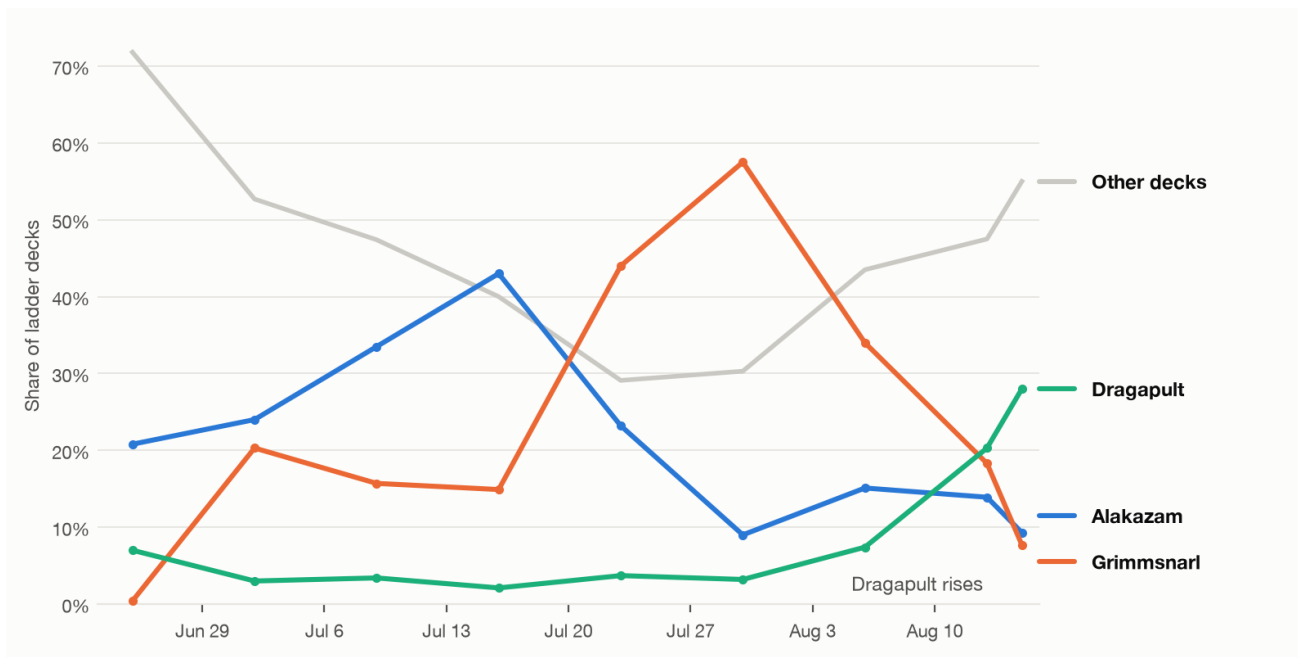
Cards in hand at attack minus cards needed for the knockout.



We gave the network the count directly, and added exact engine checks for things a network should never guess: take a winning line when one exists, avoid decking out, and for Dragapult, split its spread damage so no prize is wasted.

Dragapult had put all six damage counters on a single target 84% of the time; it now places them optimally on every one of 3,452 recorded attacks. Two lessons kept these checks honest. A rule must be right for every possible next draw: a rule stopping the bot from fetching a "dead" Alakazam cut those picks from 31 to 0, but we removed it, because with Rare Candy still in the deck that Alakazam is often live a draw later. And blocking a bad move is not enough: our veto cut zero-damage attacks into walls from 213 to 0, but the bot then often passed instead, and deck-outs in those games rose from 50% to 64%.

6. What we ruled out

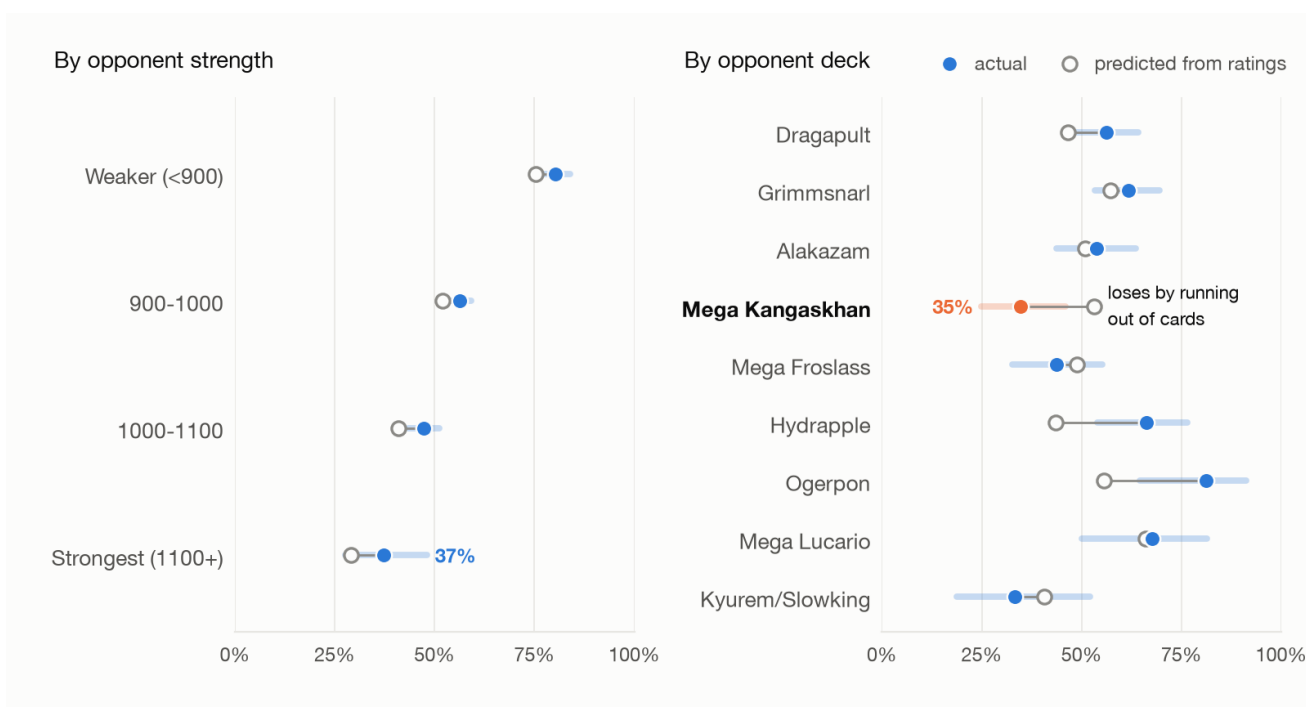


We tried	Result	What we did instead
Search during training (AlphaZero-style)	a strong network weakened: 43% -> 30% against its own frozen copy	train without search, check with the engine at play time
Training against every deck at once	the bot forgot its own deck: own-deck strength fell to 21%	a league of expert pilots
A learned identity for each card	only 46 of ~1,300 cards ever learned one	describe cards by their rules
A value network to choose moves	right on only 40% of 705 close decisions, though good at judging positions	the pilot chooses, the engine checks
Bigger networks and more human games first	4x the size: no change over 1,000 games	fix the recipe, then scale the network
Copying search decisions into the pilot	the pilot got weaker: 45% -> 28% as we copied more	let the league teach lookahead
Rewarding damage dealt	12 points weaker over 240 games	reward prizes and wins
Popular human tech cards	mirror dropped to 31%	keep the list simple
Trusting wins against our own practice opponents	54% at home, 0 of 5 on the ladder	measure against opponents the agent never trained with

7. Robustness: strong across the board, with one clear exception

Over the final two weeks of ladder games, facing 24 different opponent decks with none above 19% of games, our agents won more often than their rating predicts against every level of opponent, including the strongest: against opponents rated 1000-1100 they won 47% where 41% was predicted. Against nearly every common deck the same held once opponent strength is accounted for, and win rate was steady over time: across five consecutive 200-game blocks it stayed between 55% and 61%. Like the rest of the field, our agents do better going first (63%) than second (46%), and going second they play right at their rating's expectation.

Actual win rate vs rating prediction, both final agents pooled.



The one real exception is Mega Kangaskhan with Mist Energy, which blocks Alakazam's damage counters. Our Alakazam agent won 39% where 55% was predicted, while other Alakazam players beat the same lists 73% of the time. The list already holds the answers, Enhanced Hammer to strip the Energy and backup attackers that deal normal damage, but our pilot too often passes instead, and three quarters of our losses there were deck-outs. That plan spreads over several turns, exactly the kind of rare, long-range line that learning from game outcomes struggles to discover.

So the pilot is not exploitable by opponent strength or by most strategies; its limits are overall strength and this one class of slow defensive deck. It is also reliable: not one of 744 checked games was lost to the time limit.

Measurement. The engine cannot replay a game identically, so every comparison is noisy: 240 games only reliably detect a 9-point difference. We set sample sizes before reading results and measured progress against opponents the agent never trained with. We also opened a public playground where anyone could play our bots: Abhyuday never won in more than ten games, and Sebastian won one of two.

8. How this extends

- A more flexible deck. The recipe is deck-independent; on a deck with fewer inherent bad matchups, the same pilot quality would translate into better results.
- One pilot for every deck. The network already conditions on the decklist; training one pilot across the league's decks is the natural next step.

- Longer-range planning. Targeted league opponents and engine checks that look several turns ahead are the direct fix for wall decks, and a longer thinking budget in Round 2 would let them go further.

Team: Abhyuday (agent, training, infrastructure) and Sebastian (TCG expert: deck strategy and game reviews). Pokémon and all character names are trademarks of Nintendo, Creatures and GAME FREAK; all figures are our own.