PokéLLMon Trainer: LLM Model Distillation

Christopher Nalty
Oregon State University
naltyc@oregonstate.edu

Sasha Rosenthal
Oregon State University
rosenths@oregonstate.edu

Abstract

Pokémon battling is a complex game that can be used to benchmark an agents ability to deal with complex situations. Pokémon provides many challenges that will be essential to overcome for generalized AI systems including hidden information and implicit long term planning. Currently to our knowledge, the only language based system to play Pokémon battles [15] uses prompting methods to improve performance, such as In-context learning and self-consistency [23]. These methods can only improve an agents ability so far without being able to instill knowledge about the specific domain into the network. We propose PokéLLMon Trainer, a method for model distillation on Pokémon battles. Our solution improves small agent's ability to select valid moves and gives a slight improvement on performance against human opponents.

1. Introduction

As Artificial Intelligence (AI) agents become more advanced, finding ways to benchmark their capabilities on complex tasks compared to humans is important to measuring progress of the field. We use Pokémon battles as an environment for such a benchmark. Pokémon battles present significant challenges for AI agents, including the need to navigate hidden information, requiring domain-specific knowledge, and the formulation of long-term strategies.

PokéLLMon [15] introduced the first Large Language Model (LLM) agent capable of near human parity performance using GPT-4 [18] and multiple methods to deal with the aforementioned difficulties. However, these methods can only improve performance so far without fine-tuning a model. Without such fine-tuning the agent has limited capability to reason about future turns and can still suffer from LLM hallucinations. Their methods also create computational inefficiencies due to repeating large portions of their prompt at each time step.

We propose model distillation as a first step towards training a LLM agent to outperform humans in Pokémon battles. Our method uses a 7-billion parameter Llama model [21] to collect game data, which is then used to distill a 1.1-billion parameter model Tiny-Llama [24]. Our method has the following contributions:

- We provide a way for the agent to gain knowledge about the environment dynamics purely through examples. This gives our agent the ability to surpass the teacher model.
- We increase the efficiency of inference by removing roughly 10% of each turn's prompt.
- We greatly improve the student model's ability to select valid moves.

2. Background

2.1. Pokémon Battles

Species: As of the writing of this paper there are currently 1025 species of Pokémon [7], each with a unique combination of typing, stats, abilities, and move pools.

Stats: Stats are several parameters relating to an individual Pokémon that can affect turn order, move success, and damage output during a battle. There are six main stats: Hit Points (HP): determines how much damage a Pokémon can take before it faints; Attack (Atk): partly determines how much damage a Pokémon deals when using a physical move; Defense (Def): partly determines how much damage a Pokémon takes when being hit by a physical move; Special Attack (SpA): partly determines how much damage a Pokémon deals when using a special move; Special Defense (SpD): partly determines how much damage a Pokémon takes when being hit by a special move; Speed (Spe): determines the order that Pokémon act in battle within a moves priority group [10].

Abilities: Abilities are passive effects that Pokémon can have that can affect battles. For example, *Quagsire*'s ability "Unaware" makes it ignore an opponent's stat changes during damage and accuracy calculations, both when it takes it an attack or attacks. Some Pokémon have multiple ability options to choose from, each with differing effects. These abilities also may not always be helpful, some abilities can be hindering to a Pokémon [2].

Type: Each Pokémon can have up to two types, which define which type moves it has advantages and weaknesses against, as well as are immune to. If a Pokémon has two types the total advantage is determined is by multiplying the advantage multipliers together for each type. However, if either type is immune the Pokémon will be immune no matter how the other type would be affected (ex: a ground move used against a steel/flying type) [13]. Additionally, if a Pokémon uses a move that shares a type with the Pokémon, the move gets a 1.5x damage multiplier [4]. Some abilities may further increase this multiplier.

Moves: Each Pokémon can learn up to four moves from a pool of learnable moves for that Pokémon out of the 934 possible Pokémon moves [6]. These moves have three general categories, physical moves, special moves, and status moves. Physical and special moves deal damage based on the users Attack [8] or Special Attack [9] respectively, as well as the targets Defense or Special Defense. Status moves don't directly cause damage, and includes moves which do things such as change the weather, inflict status conditions, raise or lower the stats of a Pokémon, or heal a Pokémon, etc [12].

Status conditions: Status conditions are a variety of conditions which affect a Pokémon's ability to battle. There are two main types, non-volatile status conditions which last until healed or removed by other effects, and volatile status conditions which last while a Pokémon is active during a battle [11].

Non-volatile affects are a smaller category of conditions which includes burn, freeze, paralysis, poison and badly poisoned, and sleep. Burn, poison, and badly poisoned deal varying damage over time, 1/16th, 1/8th, and 1/16th increasing per turn respectively. Burn and paralysis both cause stat drops, burn to Attack and paralysis to speed. Freeze, sleep, and paralysis can all cause a Pokémon to be unable to move, with freeze having a 20% chance for the Pokémon to thaw each turn, sleep lasting one to three turns before the Pokémon wakes up, and paralysis being a 25% chance for a Pokémon to be unable to move each turn.

Volatile status conditions are a much broader category of conditions which includes dozens of conditions, some of the more common of which will be covered here: Taunt, Confusion, and Encore. Taunt is a condition which causes a Pokémon to be unable to use status moves for 3 turns. Confusion is a condition which sometimes causes a Pokémon to attack itself instead of using the selected move for two to four turns [3]. Encore is a condition which forces a Pokémon to use the same move it used prior to being encored for 3 turns [5].

2.2. Pokémon Showdown

Pokémon Showdown is an online Pokémon battle simulator which provides a well featured simulation with a web-based GUI for human players, as well as web APIs to interact with programmatically. We extend PokéLLMon [15] and PokéEnv's [20] battle environment to add our agents. This battle environment tracks the state of the game received from Pokémon Showdown and translates it to text. Our LLM agent then uses this text and gives an output of the next action to perform, which is then sent to the server.

Battles are comprised of two players who each start with six Pokémon. In our case for random battles each player is given six random Pokémon from a pool each with random move sets, abilities and items. Both players pick moves simultaneously before they are executed in turn based on each player's current Pokémon's speed. Players can also choose to switch Pokémon, which will be executed before the opponent's move. Any information about the opponent's team and moves is hidden until that Pokémon or move is used by the opponent. The goal of the battle is to reduce all of your opponent's Pokémon to 0 hit points. A Pokémon with 0 hit points can no longer be used, and once all of a player's Pokémon reach 0 hit points that player loses.

2.3. Large Language Models

Large Language Models (LLMs) encompass mainly Transformer [22] based architectures that perform language modelling. It has been shown by [19] that these models perform strong zero-shot generalization on a multitude of language based tasks. They show that this generalization capability increases with the size of the model. Additionally, [1] shows that the performance of these models can be greatly improved by fine-tuning on the target domain. In this work we attempt to leverage both of these facts by collecting zero-shot data with a 7-billion parameter version of Llama2 [21] and using it to fine-tune Tiny-Llama [24], a 1.1-billion parameter model.

2.4. PokéLLMon [15]

The previous work we build off of is PokéLLMon [15]. In this work they introduce the first LLM capable of playing Pokémon battles at human near human parity. They achieve this by adopting three techniques to improve pretrained LLM performance. First, they employ In-Context Reinforcement Learning, where the agent is given feed back about its previous moves when selecting a next move. Second, to deal with LLM hallucination they use Retrieval-Augmented Generation [17] to give the agent information about Pokémon including types, stats and move effectiveness. This allows the agent to make informed decisions without needing to have perfect prior knowledge of the game dynamics. Finally, they use Self-consistency [23] which generates multiple candidate outputs by sampling predicted tokens, and chooses the output generated the most times. We continue to use all three of these methods on our trained models.

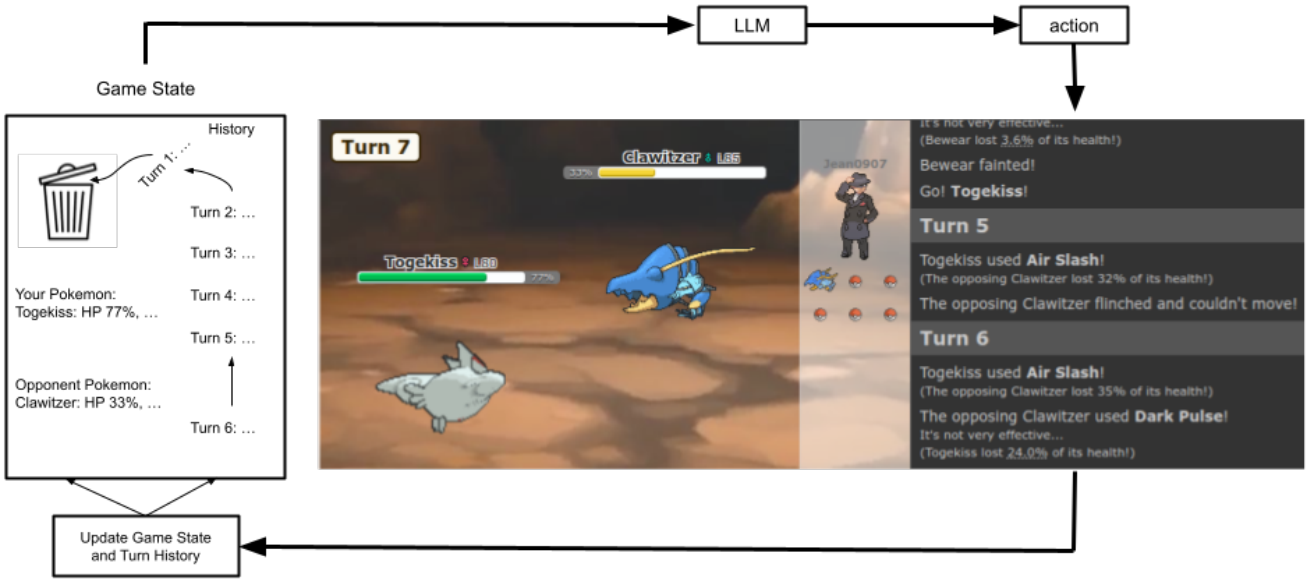

Figure 1. PokéLLMon framework that allows an LLM agent to interact with a human. Each turn the information is sent to our system, which updates the turn history and status of each player Pokémon. The game state is then given to the agent which selects an action. This action is then sent back to Pokémon Showdown.

3. Methods: PokéLLMon Trainer

We employ a framework akin to that described in PokéLLMon [15], engaging our agents in 8th Generation Random Battles on the Pokémon Showdown ladder. Agents are provided with both the turn history and the current state of the game to inform future decision-making. We adopt similar methodologies for In-Context Reinforcement Learning, Retrieval-Augmented Generation [17], and Self-Consistency [23] with k=3. Notably, untrained agents are introduced to Pokémon battles through an informational prompt—an advantage not extended to their trained counterparts.

This high-level framework facilitates the collection of prompts and responses from large-model engagements against human opponents on the Pokémon Showdown ladder. The accumulated data serves as a foundation for fine-tuning a smaller model, thereby distilling knowledge concerning the output format and the typical progression of Pokémon battles. Our method allows the agent to learn the dynamics of Pokémon battles, and increase its rate of generating valid moves.

3.1. Data Collection

To collect data we select a 7-billion parameter variant of Llama2 [21] as the teacher. By LLM standards this model is quite small, however it is the largest model we can reasonably run with our hardware restrictions. Despite its size, this model consistently generates valid moves, and achieves a slightly higher win rate than the baseline random agent. We collect prompts from 1,091 ladder games, which in total provides 23,323 prompts for use in training and testing. Each turn the model is given a prompt and produces an output, both of which are saved for training and testing. If the teacher agent chooses an invalid move the prompt and output are not saved, and the agent chooses the highest damaging move available.

3.2. Model Distillation

We propose a method of distilling a larger model's ability into a small model. A 1.1-billion parameter language model, TinyLLama [24], is used as the student. This model has the same architecture and tokenizer as our teacher model allowing us to isolate size as a variable.

We create training examples from the data recorded by the teacher model. First, the introduction prompt about Pokémon battles is removed to improve model efficiency. Removing this reduces each prompt by nearly 200 tokens, about 10% of the entire prompt. We posit that this prompt is unnecessary after fine-tuning due to the model seeing historical turns and their outcomes. Once the introduction prompt is removed, the remaining prompt is combined with the teacher's output and an end of sequence token is added to create a training example.

The student model is then trained in a causal language modelling mode where the model is given the prompt and sequentially predicts each next token. The model's parameters are then trained by using Cross-Entropy Loss on each

Agent	Perplexity	Valid %
Distilled (Ours)	1.0377	80.7%
Tiny-Llama [24]	4.6229	8.87%

Table 1. Validation Results on the test set of recorded turns taken by the teacher model. The distilled model significantly outperforms the base model on both perplexity and valid moves.

predicted token. The optimizer used is 8-bit Adam [14] with weight decay. An initial learning rate of 2e-5 is used with a linear annealing, and a weight decay of 0.01. The model is trained for a maximum of five epochs, which takes a run time of approximately ten hours.

Through this process, we aim to imbue the TinyLLama [24] model with the practical knowledge and decision-making prowess of its larger counterpart, thereby enhancing its move selection accuracy and overall performance in Pokémon battles.

4. Experiments and Results

To test our method we perform multiple experiments including validation on a holdout set comprised of 15% of the data collected from the teacher, and ladder results for each model. For validation we record perplexity [16] and move validity of Tiny-Llama [24] before and after fine-tuning. Additionally we test the teacher, student (both trained an untrained), and an agent that selects random moves created in [20]. We show that fine-tuning small LLMs on a larger models output can significantly increase the number of valid moves output by the model, and instill an understanding in Pokémon that allows the small model to surpass the performance of the large model.

4.1. Validation Performance

Our first comparison is to measure statistics about the models performance on a holdout set of data collected from Llama2's [21] ladder games and the models own ladder games. We measure the perplexity [16] of the model's outputs on the holdout set, a common metric for natural language tasks. We also measure the percentage of valid outputs by the model while playing its own ladder games. A valid move is one that does not necessarily match the output label, but is an output that is 1) in the json format and 2) a move that the current agent's Pokémon knows or a Pokémon the agent can switch too.

We find that prior to fine-tuning the model is almost completely unable to predict the move chosen or even produce a valid move, with it failing to output valid moves over 90% of the time and has high perplexity. After fine-tuning for five epochs we see a significant increase in performance for both of these metrics. Perplexity decreases from 4.6229 down to 1.0377, and the percentage of valid moves increases over nine times from 8.87% to 80.7%. We

Agent	Win Rate	Adj. Win	Score
Llama-7b [21]	13.52%	10.92%	-3.61
Distilled (Ours)	17.7%	14.1%	-3.27
Tiny-Llama [24]	16.67%	10.11%	-3.89
Random [20]	10.10%	10.10%	-4.09

Table 2. The results of each model on the ladder against human players. The adjusted win rate is the win rate after wins where the opponent disconnects while ahead are removed. Score is calculated by calculating (Opponent Pokémon knocked out - Agent Pokémon left) at the end of battle. For all metrics a larger number indicates better performance.

believe that this gap may be caused by the limited amount of training data collected. Even though we played over 1,000 games to collect data this likely does not give us coverage of all Pokémon available in random battles, or all moves available in this generation. Additionally, smaller models may have trouble selecting the specific information from the long context of the prompts that surpass 2,000 tokens in length. These prompts also contain move names and Pokémon that are controlled by the opponent, which could overlap in some cases.

4.2. Ladder Performance

Comparative to previous works our models are significantly less powerful, however we still find it useful to set a baseline of performance for battling real humans. We compare the performance of the teacher model (Llama2-7b-chat-hf [21]), the student model (Tiny-Llama [24]) before fine-tuning, our distilled student model, and a random agent. We compare the agents performance on the Pokémon Showdown ladder with three metrics: Win Rate, Adjusted Win Rate, and Score. To calculate win rate we remove any games won by the agent where the opponent disconnected before taking any actions. Adjusted Win Rate is calculated by additionally removing any games won where the opponent disconnected with more Pokémon unfainted than the agent. We note that it is not necessarily the case that the agent would lose such a game. However, we find evidence in our recorded replays that many of these games the opponent disconnects with a clear advantage. Finally, we report the average score of the agent's games. The score of a game is calculated by taking the number of Pokémon the agent has unfainted and subtracting the number of Pokémon the opponent has unfainted at the end of battle. This can give a better idea about how close agent is to winning. The more positive a score is the larger the margin of a win, and the more negative a score is the larger the margin of loss.

To gather results the teacher model played 1,091 games on the ladder, which were also used for training. When this agent chooses an invalid move it takes the most damaging move it has available. Each other agent was tested on the ladder for approximately 100 games. To better highlight the

raw abilities of these agents they select a move at random when their generated output is invalid.

The Tiny-Llama model [24], prior to fine-tuning, achieves a notably high raw win rate. Yet, its performance, when evaluated against adjusted win rates and scores, aligns more closely with that of a random agent—attributable to over 90% of its outputs being invalid, necessitating substitution with random actions. Remarkably, the distilled model surpasses the teacher across all ladder metrics, a testament to its ability to assimilate and retain knowledge from turn histories and retrieval-augmented prompts. This contrast underscores the distilled agent's advantage: it internalizes and applies battle strategies across encounters, leveraging insights inaccessible to the teacher, which operates on a turn-by-turn basis without cumulative learning.

5. Conclusion

We present the first method to train LLMs to play Pokémon Battles by using model distillation. We show that this greatly increases a small models ability to select valid moves, and even increases its ability against human opponents to greater than the original model. We see a nine-fold increase in the percentage of valid moves, and a 4% win rate increase. We also increase model efficiency with training by removing the need for an engineered introduction prompt that is repeated every turn.

There are still many challenges to be overcome in future work. The agent is still unable to pick valid moves for every turn and it is yet to be seen if the performance increase remains when distilling between much larger models. Here we the lay the groundwork to show such experiments may be plausible and leave the potential for further improvement through similar means with human data.

References

[1] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.

[2] Bulbapedia. Ability. `https://bulbapedia.bulbagarden.net/wiki/Ability`, 2024. Accessed: 2024-03-22.

[3] Bulbapedia. Confusion (status condition). `https://bulbapedia.bulbagarden.net/wiki/Confusion_(status_condition)`, 2024. Accessed: 2024-03-22.

[4] Bulbapedia. Damage. `https://bulbapedia.bulbagarden.net/wiki/Damage`, 2024. Accessed: 2024-03-22.

[5] Bulbapedia. Encore (move). `https://bulbapedia.bulbagarden.net/wiki/Encore_(move)`, 2024. Accessed: 2024-03-22.

[6] Bulbapedia. List of moves. `https://bulbapedia.bulbagarden.net/wiki/List_of_moves`, 2024. Accessed: 2024-03-22.

[7] Bulbapedia. List of pokémon by national pokédex number. `https://bulbapedia.bulbagarden.net/wiki/List_of_Pok%C3%A9mon_by_National_Pok%C3%A9dex_number`, 2024. Accessed: 2024-03-22.

[8] Bulbapedia. Physical move. `https://bulbapedia.bulbagarden.net/wiki/Physical_move`, 2024. Accessed: 2024-03-22.

[9] Bulbapedia. Special move. `https://bulbapedia.bulbagarden.net/wiki/Special_move`, 2024. Accessed: 2024-03-22.

[10] Bulbapedia. Stat. `https://bulbapedia.bulbagarden.net/wiki/Stat#List_of_stats`, 2024. Accessed: 2024-03-22.

[11] Bulbapedia. Status condition. `https://bulbapedia.bulbagarden.net/wiki/Status_condition`, 2024. Accessed: 2024-03-22.

[12] Bulbapedia. Status move. `https://bulbapedia.bulbagarden.net/wiki/Status_move`, 2024. Accessed: 2024-03-22.

[13] Bulbapedia. Type. `https://bulbapedia.bulbagarden.net/wiki/Type`, 2024. Accessed: 2024-03-22.

[14] T. Dettmers, M. Lewis, S. Shleifer, and L. Zettlemoyer. 8-bit optimizers via block-wise quantization, 2022.

[15] S. Hu, T. Huang, and L. Liu. Pokellmon: A human-parity agent for pokemon battles with large language models, 2024.

[16] F. Jelinek, R. L. Mercer, L. R. Bahl, and J. K. Baker. Perplexity—a measure of the difficulty of speech recognition tasks. *The Journal of the Acoustical Society of America*, 62(S1):S63–S63, 08 2005.

[17] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. tau Yih, T. Rocktäschel, S. Riedel, and D. Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks, 2021.

[18] OpenAI. Conversational support from chatgpt. `https://www.openai.com/`, 2024. Assistance provided on drafting and coding for academic work.

[19] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners. 2019.

[20] H. Sahovic. Poke-env: pokemon ai in python.

[21] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. Bikel, L. Blecher, C. C. Ferrer, M. Chen, G. Cucurull, D. Esiobu, J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. Hartshorn, S. Hosseini, R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. Kloumann, A. Korenev, P. S. Koura, M.-A. Lachaux,

T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E. M. Smith, R. Subramanian, X. E. Tan, B. Tang, R. Taylor, A. Williams, J. X. Kuan, P. Xu, Z. Yan, I. Zarov, Y. Zhang, A. Fan, M. Kambadur, S. Narang, A. Rodriguez, R. Stojnic, S. Edunov, and T. Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023.

[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[23] X. Wang, J. Wei, D. Schuurmans, Q. Le, E. Chi, S. Narang, A. Chowdhery, and D. Zhou. Self-consistency improves chain of thought reasoning in language models, 2023.

[24] P. Zhang, G. Zeng, T. Wang, and W. Lu. Tinyllama: An open-source small language model, 2024.