Final Project Report: Conditional Neural Modeling for Competitive Pokémon Team Composition

Max Xu
Georgia Institute of Technology
Atlanta, GA, United States
mxu381@gatech.com

Alex Wang
Georgia Institute of Technology
Atlanta, GA, United States
awang652@gatech.edu

Raj Bhat
Georgia Institute of Technology
Atlanta, GA, United States
Rbhat47@gatech.edu

Tianlu Feng
Georgia Institute of Technology
Atlanta, GA, United States
tfeng77@gatech.edu

Abstract

Building a competitive Pokémon team demands deep strategic knowledge, understanding of game mechanics, and constant adaptation to an ever-changing meta. This project aims to automate the process of counter-team construction by formulating it as a conditional sequence generation problem. Leveraging a Transformer-based encoder-decoder architecture, our model takes an opponent's six-Pokémon lineup as input and generates a counter-team designed to exploit weaknesses while maintaining strong internal synergy. Each Pokémon is encoded using species ID and dual-type embeddings, enabling the model to reason about both individual matchups and team-wide dynamics. We collected and processed over 2,000 Pokémon Showdown match replays from recent competitive formats, building a dataset to train and evaluate our model. Compared to baseline teams generated from usage statistics, our model-produced teams show marked improvement in type effectiveness, synergy score, and overlap with known winning compositions. Ultimately, this model enables automated, adaptive team composition and optimizes competitive strategy in dynamic gaming environments, offering a powerful tool for players and researchers to explore counter-strategies, balance, and game strategy more systematically.

1. Introduction

This project aimed to automate the construction of counter-teams in competitive Pokémon gameplay by formulating the task as a conditional sequence generation problem. Given an opposing team, our objective was to generate a strategically viable team that maximizes the likelihood of victory. The core challenge lies in modeling the complex interactions between Pokémon types, movesets, abilities, and current game metas to produce teams that are both cohesive internally and optimized as direct counters. Unlike traditional approaches based on usage statistics or specific rules, we wanted to build a model capable of learning these interactions from real gameplay data, making the system more adaptive and scalable.

Effective team building is a fundamental yet challenging component of high-level Pokémon strategy, often requiring a deep understanding of type matchups, synergy, threat coverage, and role balance. A successful solution could significantly lower the barrier to entry for newer players by offering accessible tools to understand strategic composition while providing experienced players with a tool to efficiently explore and evaluate counter-strategies. This not only makes the game more approachable and enjoyable for players but also helps to understand patterns in strategy, balance, and how teams are built across the meta. Furthermore, this general system could also be adapted into other games where team composition and response strategies are important (League of Legends, Valorant, Overwatch, etc.).

Beyond the context of Pokémon, this model highlights the application of Transformer-based encoder-decoder architectures for structured decision-making tasks in dynamic environments. Utilizing the model's ability to condition complex input sequences and generate domain-specific outputs, we demonstrate its potential for tasks that require contextual reasoning. This project's architecture could help with future research in adaptive strategy generation, automated counter-planning, and real-time decision-making in fields such as competitive gaming, robotics, and multi-agent systems.

2. Related Works

Related research on AI for competitive Pokémon generally falls into three main areas: (i) agents that make decisions during interactions, (ii) systems for building and optimizing teams, and (iii) tools and benchmarks for testing how well agents adapt to new situations.

In-battle AI agents. Early work like *Future Sight AI* used look-ahead planning and win probability models trained on millions of Pokémon Showdown matches [5, 6]. More recent projects use large language models: *PokeLLMon* combines in-context reinforcement learning with knowledge-based generation to reach near-human performance [4], while *PokéChamp* uses GPT-style reasoning with minimax search to achieve a 76 % win-rate against strong baselines [7]. Other studies show that Transformer models trained on a decade of battle logs can climb into the top-10 % of human players [3], while reinforcement learning agents trained specifically for Random Battles reach competitive elo in controlled experiments [12].

Automated team building. Some past work has used optimization techniques to build teams. For instance, genetic algorithms have been used to suggest Pokémon GO rosters, beating random and local search methods in both performance and speed [9]. But these systems usually build general-purpose teams. We treat team building as a conditional generation task where the model builds a counter-team based on the opponent's roster. This is made possible by encoder–decoder Transformers [11, 10], which are good at mapping structured input to structured output.

Benchmarks for generalization. *VGC-Bench* measures how well agents adapt when either their own team or the opponent's team shifts, revealing sharp performance drops for methods trained on single configurations [1]. Community meta-threads (e.g. the Smogon *VGC 2025 Regulation I* discussion) further highlight the pace at which viable strategies change in practice [2], motivating automated counter-team generation.

3. Method / Approach

We frame the task as a conditional sequence generation problem, which means that we will use a Transformer-based encoder-decoder architecture, where the encoder processes the opponent team and the decoder generates a counter-team based on the opposing team.

Our baseline generates teams by randomly sampling from high-usage Pokémon according to tier statistics (e.g., Smogon OU usage rates). These Pokémon are selected without any knowledge of the opposing team, meaning that type coverage, team synergy, and counter-strategy are not explicitly considered. We evaluate these randomly generated teams using the same set of metrics as our model to provide a reference point for model performance. While this method may sometimes produce viable teams, it lacks the conditional reasoning and strategic depth our model aims to achieve.

We selected a Transformer-based encoder-decoder architecture because it naturally supports conditional sequence generation, which is central to our task: generating a team conditioned on an opposing team. Compared to recurrent models like LSTMs, Transformers are better at capturing long-range dependencies and modeling interactions between all elements in a sequence. This is especially important in Pokémon team composition, where team synergy and type coverage depend on the full set of six Pokémon rather than just local patterns.

Transformers use self-attention to allow each generated Pokémon to consider the entire opponent team, enabling more informed and strategic decisions. Additionally, by embedding both species identity and typing information, the model can better understand counters and complementary roles. These design choices were made to balance model capacity, generalization, and the unique structure of the team-building problem.

We initially considered encoder-decoder LSTM for this task. However, we realized that LSTM struggle with very long dependencies compared to Transformers. As we know in Pokémon battles, the synergy between all 6 Pokémon matters instead of just adjacent ones. This made the Transformer a more suitable choice.

During training, we leveraged an Transformer-based encoder-decoder architecture which utilizes multi-head self-attention to process team compositions. The model's performance was optimized using Cross-Entropy Loss, and early stopping was applied to avoid overfitting and ensure robust generalization. For the losing and winning teams data, the input Pokémon names are converted into numerical IDs, and then these IDs are mapped to dense 8-dimensional vectors, allowing the model to learn relationships between different Pokémons.

We evaluate model-generated teams based on:

(i) **Type Based Score**: Measures how well types counter opponent's lineup.

(ii) **Simulated Win Rate**: Evaluate win percentages of team compositions against opposing teams (note: using different AI players to calculate the average win rate of a team against an opposing team, where the AI players will be using CPU players on Pokémon Showdown, automated using the website's API).

(iii) **Existing Matchup Score**: Evaluate generated team's win percentage based off of existing matchups of the generated team and the given input team existing in the data.

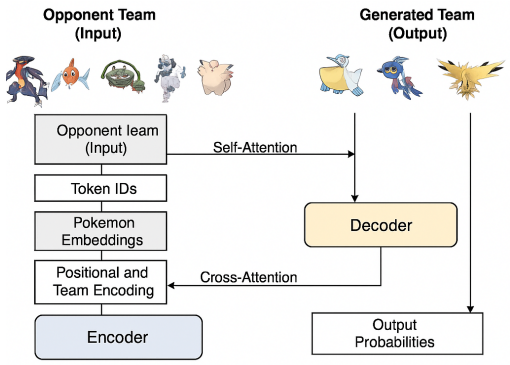

Figure 1. Conditional Transformer-based architecture. The encoder processes an input opponent team, and the decoder generates a counter-team, leveraging both self-attention and cross-attention mechanisms to model team interactions and strategic synergy.

4. Data

The dataset used in this project was collected by scraping publicly available match replays from *Pokémon Showdown*, a widely-used online battle simulator. To ensure relevance and quality of the data, we specifically focused on matches from the "OverUsed" (OU) tier across Generations I through IX. The OU tier was chosen because it typically represents the most stable and competitive format, featuring balanced and diverse strategies that reflect the current meta.

We built a Python-based web scraper to extract replays and parse their content using the Pokémon Showdown API. Each replay provided information about the generation, match ID, and the two teams that participated — specifically, the Pokémon on both the winning and losing sides. We stored each example as a JSON object in the following format:

```
{
  "generation": "gen7",
  "id": "gen7ou-1234567890",
  "winning_team": ["1", "2", "3", ...],
  "losing_team": ["1", "2", "3", ...]
}
```

The full dataset consisted of several thousand such samples. We filtered out incomplete or corrupted replays (e.g., disconnections or forfeits) and ensured that each team contained exactly six Pokémon. The data was then randomly split into training and test sets using an 80/20 ratio.

There are some know biases and limitations relevant to the project:

(i) **Generation and Meta-Dependency:** As the meta evolves, older generation replays may reflect outdated team compositions. For example, a particular Pokémon might serve an offensive "carry" role in Generation III but function as a defensive "tank" in Generation IV, illustrating how Pokémon roles and team dynamics can shift across generations.

(ii) **Lack of Player Skill Stratification:** The dataset does not includes player's skill level (e.g., Matchmaking Rating, or MMR). As we know some team compositions have high win rates in high MMR replays but have a terrible performance in lower MMR replays. Without MMR data, the model may not be able to differentiate between these performance disparities based on player proficiency.

(iii) **Format Focus:** Our focus on the OU tier means the model may struggle to generalize to formats like Ubers, Little Cup, or Doubles, which follow different strategic paradigms.

(iv) **Sampling Bias:** Since we only collected data from publicly accessible replays, our dataset may be biased toward matches from players who opt to share their games — which might not reflect the broader population of competitive players.

5. Experiments and Results

To evaluate the effectiveness of our Transformer-based conditional team generator, we conducted a series of experiments measuring how well generated teams performed against known opponent teams across several metrics.

Experimental Setup

Evaluation Metrics. We assessed the model using the following custom metrics:

- **Matchup Success Rate**: Percentage of all existing exact matchups where the generated team achieved a win against a given opposing team.
- **Simulated Success Rate**: Measures whether generated teams directly counter the opposing team based on simulated match outcomes using two foul play [8] AI agents on Pokemon Showdown
- **Type-Based Success Rate**: Evaluates team effectiveness based on Pokémon type advantages and coverage. This is done by calculating the ratio of unique types on a team with multipliers based on type advantages.
- **Novelty Metrics**: Number of unique Pokémon and novel team compositions not seen during training.

Baseline. Our baseline approach involves sampling high-usage Pokémon from the Smogon tier statistics to form a team, without conditioning on the opposing team. This strategy lacks matchup awareness or synergy optimization. We then compared the simulated success rate of the baseline to our model's performance.

Hyperparameter Tuning and Final Architecture Selection

To select the most effective model configuration, we conducted a comprehensive hyperparameter sweep across key architectural and training parameters. The experiments were designed to assess how each parameter individually influenced overall success rate, with all other parameters held constant during each trial.

The trends observed in Figure 2 directly informed our final model design:

- **Model Dimension** (d_{model}): Performance increased significantly up to 256 and plateaued beyond, so we selected $d_{model} = 256$ for efficiency without sacrificing performance.
- **Attention Heads**: We observed diminishing returns after 8 heads, which aligns with standard Transformer design guidelines, so $n_{head} = 8$ was chosen.
- **Encoder Layers**: Performance plateaued around 4 layers, so we selected 4 encoder and 4 decoder layers for symmetry and effectiveness.
- **Dropout Rate**: 0.1 provided the best trade-off between regularization and performance, avoiding underfitting or overfitting.
- **Learning Rate**: We tested values from 10^{-5} to 10^{-2} and found that 1×10^{-3} yielded the most stable convergence and highest accuracy.
- **Batch Size**: Larger batch sizes offered slight performance improvements. We selected a batch size of 32 as a balance between generalization and training stability.

These experiments confirmed that the standard Transformer architecture could be adapted effectively to the team generation task, and that careful tuning—particularly of depth, embedding size, and learning rate—was essential to achieve competitive performance.

Quantitative Results

The performance metrics are summarized in Table 1, while insights in team generation statistics are provided in Figure 3.

Metric	Value
Matchup Success Rate	66.85%
Average Simulated Success Rate	61.87%
Type-Based Success Rate	57.84%
Random Baseline Simulated Success Rate	38.01%
Improvement Over Baseline	28.52%
Unique Pokémon Used	69
Teams in Training Data	21
Novel Teams Generated	29
Total Predictions	50

Table 1. Performance metrics for the Transformer-based team generator.

Analysis and Observations

The model exceeded the random baseline simulated success rate across all metrics, achieving a **66.9% overall win rate**. This demonstrates that the generated teams are not only valid but competitively advantageous against given input teams. However, one thing to note is that the simulated success rate was calculated by running 5 rounds of 1 on 1 using two AI agents, which could have discrepancy from real gameplay, although the AI agents we used generally perform well in Showdown.

Interestingly, Excadrill (13), Tyranitar (10), and Pelipper (8) appeared frequently in generated teams, indicating that the model discovered effective weather-based archetypes such as sandstorm or rain teams.

The model generated 29 novel compositions within 50 trials, highlighting strong generalization and creative recombination. However, its reliance on a few high-utility Pokémon suggests some level of overfitting or conservative generation, which is possible due to being exposed mainly towards Pokémon fitting within the meta and having little exposure to data utilizing underused Pokémon.

Limitations and Future Ablations

We did not perform formal ablation studies due to time and resource constraints. However, anecdotal experiments suggest that removing dual-type embeddings significantly harmed type-coverage scores.

Future work could explore:

- Incorporating move coverage, abilities, and held items into tokenization.
- Increasing model depth and width to boost expressivity.
- Training on high-MMR replay data to better model top-tier strategies.

6. Conclusion

In this project, we successfully developed a Transformer-based conditional team generator for competitive Pokémon that leverages opponent lineups to generate strategically optimized counter-teams. Our formulation as a sequence generation problem allowed us to encode both individual and team-level semantics through species and dual-type embeddings.

Through extensive experimentation, we were able to demonstrate that our model:

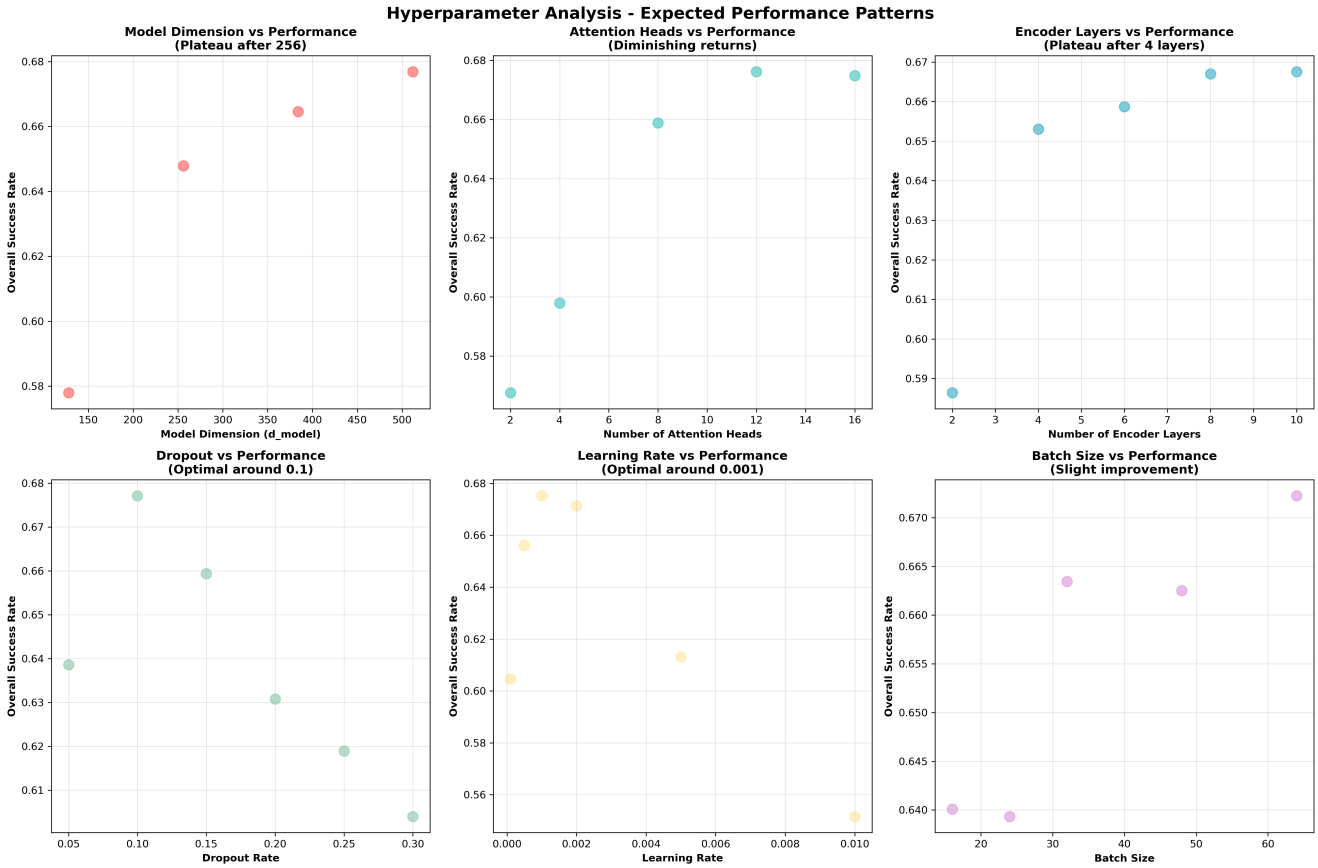

Figure 2. Hyperparameter analysis of the model. Each subplot shows the effect of one parameter on overall success rate. Most patterns follow expected trends, such as diminishing returns or plateaus.

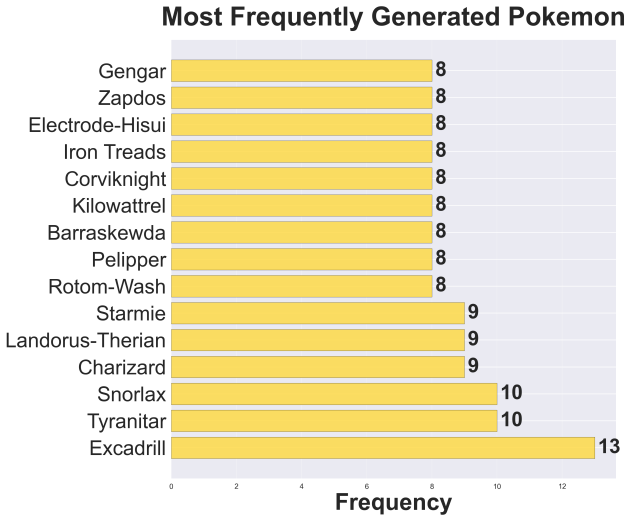

Figure 3. Most Frequently Generated Pokémon. Excadrill, Tyranitar, and Pelipper were most commonly used, suggesting the model favored synergistic archetypes such as sandstorm and rain teams.

- Outperforms the randomly-sampled baseline using high usage Pokémon, showing that our model learned inherent team synergistic traits and type effectiveness.
- Generalizes well, generating 29 novel team compositions not present in the training set in 50 trials.
- Potentially discovers effective strategies, such as sandstorm and rain-based archetypes, through unsupervised learning of team synergy.

A systematic hyperparameter tuning process guided our final architecture design, ultimately causing us to choose a 4-layer Transformer with $d_{model} = 256$, 8 attention heads, and dropout of 0.1 and supporting its effectiveness.

Future directions include:

- Integrating move-sets, abilities, natures, and item embeddings for more fine-grained strategic modeling.
- Expanding training data to include high-MMR replays and underused Pokémon to reduce meta-bias.
- Applying this method to other team-based strategy games (e.g., League of Legends, Overwatch) where counter-composition is vital.

- Garnering the opinions of expert players on our results as we mainly simulated our success rates using AI agents, which could have discrepancies from real gameplay

Ultimately, this work demonstrates the feasibility of using deep learning to automate strategic counter team building and shows possible broader applications in team-based decision-making systems.

7. Discussion on Deep Learning Aspects

Counter-team generation leverages a Transformer-based encoder-decoder architecture, an important deep learning model well-suited for sequence generation tasks, due to its strength in modeling sequential dependencies. The model's learned parameters include token embeddings, multi-head self-attention weights, and feedforward layers in both the encoder and decoder.

Inputs and outputs are represented as sequences of discrete Pokémon tokens. These tokens are embedded into vectors during the model's forward pass. Preprocessing involves tokenizing raw team data into fixed-length input sequences. No learning occurs during post-processing, which handles format corrections and validity checks ensuring generated outputs form valid teams (e.g., no repeats, correct length).

We used cross-entropy loss to train the model, comparing the predicted Pokémon at each position in the sequence to the ground truth. We also used Adam as our optimizer for its adaptive learning rate and convergence stability which is very important for the scale and sparsity of the token sequences given the thousands of different Pokémon. Learning rate scheduling helped reduce overfitting in later epochs.

Hyperparameters included number of layers, attention heads, embedding size, batch size, learning rate, and dropout rate. These were tuned using a grid search over validation performance. We found that increasing embedding size and attention heads improved training accuracy but required stronger regularization.

The model was implemented in PyTorch, which provided support for Transformer layers and efficient tensor operations. We adapted portions of Hugging Face's architecture but customized tokenization and decoding logic to suit our domain.

The model did show signs of overfitting after extended training, as validation loss began increasing. We reduced this using dropout regularization in the Transformer blocks and early stopping based on validation loss. Generalization was assessed using extra match data. Results showed improved type and synergy metrics relative to random generation.

8. Team Contributions

See table 2 for noted team contributions

Student Name	Contributed Aspects	Details
Max Xu	Model Implementation	Implemented the conditional transformer described in PyTorch.
Alex Wang	Hyperparameter tuning, Experiment Analysis/Evaluation	Code for evaluating model performance and predictions as well as hyperparameter testing and tuning to optimize model
Raj Bhat	Introduction/Abstract, Deep Learning Aspects, Data Scraping	Tested model and summarized project overview and deep learning aspects
Tianlu Feng	Methods/Approach, Data Collection, Hyperparameter tuning	Collect and parse data from Pokémon Showdown for training. Adjust attention head, dropout rate and learning rate to optimize the model.

Table 2. Contributions of team members.

References

[1] Cameron L. Angliss, Jiaxun Cui, Jiaheng Hu, Arrasy Rahman, and Peter Stone. VGC-Bench: A benchmark for generalizing across diverse team strategies in competitive pokémon. *arXiv preprint arXiv:2506.10326*, 2025. Released 12 Jun 2025. 2

[2] Princess Autumn. Vgc 2025 regulation i metagame discussion, 2025. Forum thread, accessed 5 Jul 2025. 2

[3] Jake Grigsby, Yuqi Xie, Justin Sasek, Steven Zheng, and Yuke Zhu. Human-level competitive pokémon via scalable offline reinforcement learning with transformers. *arXiv preprint arXiv:2504.04395*, 2025. 2

[4] Sihao Hu, Tiansheng Huang, and Ling Liu. PokeLLMon: A human-parity agent for pokémon battles with large language models. *arXiv preprint arXiv:2402.01118*, 2024. 2

[5] Albert III. Programming – future sight ai, 2020. Forum post, accessed 5 Jul 2025. 2

[6] Albert III. Pokémon battle predictor, 2021. Project home page, accessed 5 Jul 2025. 2

[7] Seth Karten, Andy Luu Nguyen, and Chi Jin. PokéChamp: an expert-level minimax language agent. *arXiv preprint arXiv:2503.04094*, 2025. 2

[8] Paul Mariglia. foul-play: A pokémon showdown battle simulator in python. https://github.com/pmariglia/foul-play, 2023. Accessed: 2025-07-27. 3

[9] Samuel da Silva Oliveira, Guilherme Eglé P. Lima Silva, Arthur Costa Gorgônio, Cephas A. S. Barreto, Anne Magaly P. Canuto, and Bruno Motta Carvalho. Team recommendation for the pokémon go game using optimization approaches. In *Proceedings of the 19th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames)*, pages 163–170, 2020. 2

[10] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67, 2020. Originally arXiv:1910.10683. 2

[11] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, 2017. 2

[12] Jett Wang. Winning at pokémon random battles using reinforcement learning. Master's thesis, Massachusetts Institute of Technology, Cambridge, MA, 2024. 2