Guess, Group, Reward: Solving Connections with Policy Learning over Latent Assignments

Iris Xia
Department of Computer Science
Stanford University
ixxia@stanford.edu

Justin Wei
Department of Computer Science
Stanford University
jwei31@stanford.edu

Linda Tong
Department of Computer Science
Stanford University
lkt@stanford.edu

Abstract

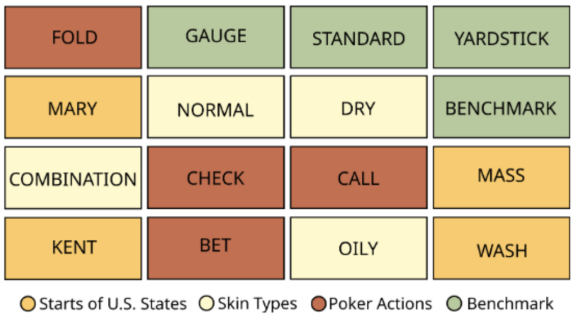
We explore different ways to model and solve the *New York Times* Connections game, a word puzzle in which the player is given 16 words and needs to form 4 groups of 4 words, with each group following a theme. This game presents an interesting challenge in combining a natural language task with learning from environment feedback, and we find that even top-performing large language models fall short of human performance. We collected 710 puzzles from the *New York Times* archive and compared three strategies: a zero-shot K-means clustering of semantic embeddings, a deep Q-learning agent trained in a custom POMDP with sparse rewards, and an RLVR-style fine-tuning of a pre-trained language model using binary puzzle-completion rewards. We find that K-means is able to capture broad semantic meanings but fails on nuanced, context-dependent categories, that classical deep Q-learning struggles with sparse rewards and long-term planning (evidenced by the 0% accuracy), and that RLVR harnesses LLM priors and sparse binary rewards to discover abstract groupings effectively, greatly outperforming the other methods.

1 Introduction

We propose a reinforcement learning agent for solving New York Times Connections, a word puzzle that requires grouping 16 words into four different categories. On each turn, the player selects four words they believe form a group. If the selection is incorrect, one of four lives is lost. Correct groupings are removed from the grid, and the game continues until all groups are found or the player runs out of lives. The difficulty lies not just in identifying semantic similarity, but also in navigating a wide range of associations—including idioms, cultural references, and categorical quirks. An example of a Connections game can be seen in Figure 1 below. In this board, FOLD, CHECK, CALL, and BET are all poker actions, while MARY, KENT, WASH, and MASS are abbreviatons of U.S. state names.

The agent learns to assign words to groups based on semantic similarity and past observations of the game state. We explore two reinforcement learning approaches to tackle this task. First, we frame the puzzle as a partially observable Markov decision process and train a deep Q-learning agent to form correct groups through interaction with the environment. This method treats grouping as a sequential decision-making problem, using dense word embeddings and game feedback to learn a Q-function over possible actions.

Stanford CS224R 2025 Final Report



(a) Sample instance of *Connections* showing a grid with various terms; each is marked to indicate its category.

Second, we adopt a Reinforcement Learning with Verifiable Reward (RLVR) approach, fine-tuning a pre-trained large language model to generate full puzzle solutions using only sparse binary rewards—specifically, whether the predicted groupings exactly match the correct ones. This setup mirrors recent advances in fine-tuning LLMs with structured feedback and encourages the model to learn abstract grouping strategies that generalize beyond surface-level similarity.

While the NYT Connections puzzle is a word game on the surface, it offers a rich testbed for evaluating reasoning, categorization, and pattern recognition—skills that are foundational to many real-world NLP and AI applications. The game's format mirrors challenges in clustering, weak supervision, and few-shot generalization, making it a surprisingly powerful lens for studying model behavior under ambiguity and limited context.

2 Related Work

Limitations of Natural Language Models: Despite recent advances in prompting and instruction tuning, large language models continue to struggle with tasks that require deliberate reasoning, particularly when intuitive patterns are misleading. Lopez et al. (2025)'s NYT-CONNECTIONS benchmark, which evaluates LLMs on the Connections puzzle, reveals deep limitations in their ability to resist "System 1" heuristics and engage in "System 2" reasoning. Even top-performing models like GPT-4 and Claude 3.5 fall short of human performance by nearly 30% on average, with minimal gains from advanced prompting techniques such as Chain-of-Thought and Self-Consistency. This performance gap motivates the exploration of alternative methods, such as reinforcement learning over latent structures, which may better capture the underlying semantics of categorization tasks under feedback.

Integrating MDPs with Language Models: The area of integrating reinforcement learning techniques with language models is fairly new and developing, but a couple of papers have shown promising results. Li et al. (2022) introduce the Pre-Trained Language Models for Interactive Decision-Making framework, which encodes the inputs to a policy (including observations, goals, and history) as a sequence of embeddings that are then passed into a policy network that is initialized by a pre-trained language model and fine-tuned to predict actions. Another example is Ma et al. (2025), who introduce Exploring with LLMs (ExploRLLM), where large language models guide the exploration phase in RL. Both studies show promising results on benchmark tasks.

Improving Reasoning with RL: After speaking with Chelsea, we wanted to adapt the 1-shot RLVR approach Wang et al. (2025), where we fine-tune a base model like Qwen using sparse binary rewards for solving full puzzles correctly. By selecting a single, well-chosen training example and applying a policy-gradient-based RL algorithm, Wang et al. (2025) were able to boost a 1.5 B-parameter model's accuracy on the MATH500 benchmark from 36.0% to 73.6%. The 1-shot approach holds across multiple base models (e.g., Qwen2.5-Math-7B, Llama3.2-3B-Instruct) and RL algorithms (GRPO, PPO), yielding substantial relative improvements in each case. This study showed promising results for fine-tuning our LLM-based puzzle-grouping policy to be more data efficient and improve cross-domain generalization.

3 Data

To build our dataset, we scraped New York Times Connections puzzles from the website word.tips using a Selenium scraper. This site archives historical puzzles by month and presents the solution groupings. Our scraper programmatically expanded each month's accordion section, parsed the HTML elements for the 16 words and their associated categories, and validated that each puzzle contained exactly 4 groups of 4 words. We extracted 710 puzzles using this approach.

4 Methods

4.1 K-means Baseline

As a non-learning baseline, we cluster the 16 words in each puzzle using K-Means on OpenAI embeddings. We use the `text-embedding-3-large` model to generate 3072-dimensional embeddings for each unique word across the dataset. These embeddings are cached and reused to avoid repeated API calls. For each puzzle, we apply K-Means clustering with $k = 4$ clusters and then assign exactly 4 words to each group based on proximity to cluster centers, using a greedy matching heuristic that respects the group size constraint. This baseline does not leverage any game feedback or puzzle-specific supervision, making it a zero-shot semantic clustering method.

4.2 Deep Q-learning

For the next approach, we attempted to leverage the gamified nature of the task to train a reinforcement learning agent using deep Q-learning. We first modeled the game as a partially observed Markov decision process. The game state consists of a list of word embeddings for the words in the puzzle (using the same embeddings as in the K-means baseline), a bit mask indicating words that have been successfully grouped, the number of lives left, and the true groupings. Actions (i.e. guesses at groupings) are modeled as a 4-tuple of indices indicating the 4 words in the guess. The observation space includes the word embeddings, updated bit mask, the number of lives left, and a categorical game feedback that contains 4 options: illegal move, incorrect answer, almost correct answer (meaning 3 out of the 4 words guessed belong to the same group), and correct answer. The reward structure is such that the agent receives a reward of 2.0 for a correct grouping (so a maximum of 8.0 can be earned for a perfectly solved puzzle), receives a neutral reward of 0.0 for any almost correct answers, and is penalized with a reward of -1.0 for any incorrect answers.

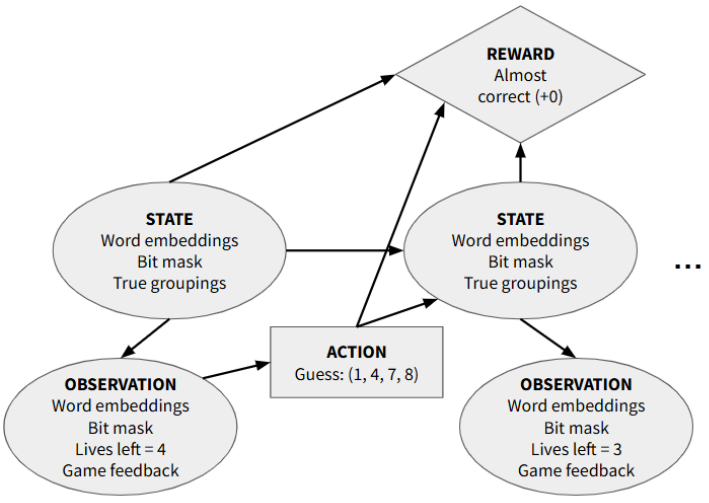


Figure 1: POMDP formulation of the task

The model architecture consists of 3 learnable fully connected multi-layer neural networks, each with parametrizable depth.

- **State Encoder**: converts the word embeddings and a normalized version of the other components in the observation vector to an embedding vector.
- **Action Encoder**: concatenates the embeddings of the four selected words and calculates an embedding vector.
- **Q Network**: concatenates the embedding vectors generated by the state and action encoders to output an estimated Q-value for the (state, action) tuple.

4.3 Reinforcement Learning with Verifiable Reward (RLVR)-Style Fine-Tuning

For the third approach, we fine-tune the `Qwen/Qwen2.5-1.5B-Instruct` model using the Group-wise Regularized Policy Optimization (GRPO) algorithm Shao et al. (2024). GRPO is a RL method that updates the policy by balancing reward-seeking behavior with stability and exploration. Specifically, the GRPO objective includes three components:

$$\mathcal{L}_{\text{GRPO}} = \mathcal{L}_{\text{PG}} + \beta \cdot D_{\text{KL}}(\pi_\theta \,||\, \pi_{\text{ref}}) + \alpha \cdot H(\pi_\theta)$$

where:

- $\mathcal{L}_{\text{PG}}$ is the policy gradient loss, computed using group-normalized advantages to stabilize updates when multiple responses are sampled from a single prompt.
- $D_{\text{KL}}(\pi_\theta \,||\, \pi_{\text{ref}})$ is a KL-divergence penalty that ensures the fine-tuned policy π_θ remains close to the reference policy π_{ref}.
- $H(\pi_\theta)$ is an entropy bonus that promotes output diversity.

In our setup, each training instance consists of a 16-word puzzle. For each instance, we sample multiple completions and compute a reward for each based on how many predicted words overlap with the ground-truth solution. We use the following prompt for completions, and we parse the model output with regexes to evaluate accuracy.

Prompt Template

You are solving a NYT Connections puzzle. You have 16 words that need to be grouped into 4 categories of 4 words each.
Words: {WORD1, WORD2, ..., WORD16}
Find the 4 groups of 4 related words. For each group, provide: 1. The 4 words in the group 2. A brief description of what connects them
Format your answer like this:
Group 1: [WORD1, WORD2, WORD3, WORD4] - Description
Group 2: [WORD1, WORD2, WORD3, WORD4] - Description
Group 3: [WORD1, WORD2, WORD3, WORD4] - Description
Group 4: [WORD1, WORD2, WORD3, WORD4] - Description
Answer:

For each completion, our reward function grants +10 points for each exactly correct group, +2 points per correct word in the right group (partial credit), -1 point for entirely incorrect groupings, and we normalize the resulting score to fall roughly within $[-1, 1]$.

5 Experiments

5.1 K-means Baseline

We evaluated performance using the Adjusted Rand Index (ARI), which calculates the proportion of word pairs that are assigned to the same or different clusters in both the predicted and true groupings, adjusted for chance-level grouping, to measure clustering quality and counted perfect group matches.

5.2 Deep Q-learning

For the deep Q-learning model, we ran a total of 500 episodes on 250 training puzzles and evaluated on the same 100 puzzles as in the K-means baseline. At the beginning of each episode, a random puzzle is chosen from the 250 training puzzles available, and we chose to run for twice the number of episodes so that the agent has the opportunity to see the same puzzle multiple times and potentially improve with multiple iterations on the same puzzle (note that the agent doesn't ever receive feedback on what the true groupings are, which is a deviation from real-life play). We computed a mean squared error loss between the predicted Q-values and target Q-values, and performed batch updates with a batch size of 32. We used the Adam optimizer with a learning rate of 10^{-4}. Lastly, we ran a simple hyperparameter sweep and used a state encoder with a depth of 2, an action encoder with a depth of 2, and a Q network with a depth of 10. We also used a standard epsilon-greedy method with an epsilon of 0.2 and a gamma of 0.9.

Our preliminary results had fairly low accuracy, and we observed that the model was often getting almost correct answers (3 out of 4 of the guesses were in the same group), but very few fully correct answers. As a result, we explored several ways to push the agent to find fully correct groups rather than just settling with almost correct answers.

Reward Shaping: We experimented with several different reward structures before settling on the one described above. Initially, we gave the agent a +0.25 reward for every correct pair in the group, such that a fully correct guess would get a reward of +1.5. With this reward structure, we observed many partially correct answers but still not many fully correct groups. We then experimented with penalizing for any incorrect answers and having a neutral reward for almost correct answers. We also increased the reward for a fully correct answer to +2.0 to further incentivize fully correct answers.

Hindsight Experience Replay: We also experimented with Hindsight Experience Replay (HER) as a strategy to deal with the sparse reward structure and to help the agent learn more from the training episodes. HER, as discussed in Andrychowicz et al. (2018) and in this course, is a strategy that relabels failed transitions as successful, pretending that the state we reached was the goal state. This allows the agent to receive some additional rewards, which gives the agent more signals to learn from. For our implementation of HER, we provided a reward of +1.0 for any predicted groups that have semantic similarity to one of the true groups. We used cosine similarity as the metric and imposed a threshold of 0.85, such that if a predicted group had a cosine similarity of at least 0.85 with one of the true groups, the agent would receive the additional reward. This incentivizes the agent to explore groupings that could lead to success.

For the deep Q-learning model, we used the same evaluation metrics as K-means, except instead of calculating the Mean ARI score, we instead compute the average reward per episode. this is because this model gets feedback from subsequent guesses rather than making four clusters at once, and so it's more appropriate to evaluate the rewards collected at each step.

5.3 RLVR

We use the `trl` library's `GRPOTrainer` with the following hyperparameters: learning rate of 3×10^{-6}, batch size of 2 per device, 3 training epochs, temperature 0.7, $\beta = 0.1$, and `loss_type="dr_grpo"`. We adopt the `dr_grpo` loss variation, which normalizes by a fixed constant L (e.g., the max generation length) instead of sequence length, in order to avoid reward bias introduced by varying sequence lengths:

$$\mathcal{L}_{\text{DrGRPO}}(\theta) = -\frac{1}{LG} \sum_{i=1}^{G} \sum_{t=1}^{|o_i|} \ell_{i,t}$$

where each $\ell_{i,t}$ is defined as:

$$\ell_{i,t} = \frac{\pi_\theta(o_{i,t} \mid q, o_{i,<t})}{\text{stop\_grad}[\pi_\theta(o_{i,t} \mid q, o_{i,<t})]} \hat{A}_{i,t} - \beta D_{\text{KL}}[\pi_\theta \,\|\, \pi_{\text{ref}}]$$

In our setup, each training instance consists of a 16-word puzzle and a gold solution partitioning them into 4 semantically coherent groups. We generated prompts that instructed the model to output

its reasoning followed by 4 labeled groups in a standardized format. The training set contained 600 examples, and the test set 100 examples.

We perform training on mixed precision (bfloat16) with gradient checkpointing enabled on one A100 GPU. We evaluate the finetuned model by parsing the model's output and measuring the accuracy of the extracted groupings.

6 Results

6.1 Quantitative Evaluation

Table 1: K-Means Baseline Results on 100 NYT Connections Puzzles

Metric	**Value**
Total Puzzles Evaluated	100
Perfect Puzzles	4 (4.0%)
Mean ARI Score	0.284 ± 0.229
Mean Perfect Groups per Puzzle	0.54
Perfect Groups Distribution:	
0 perfect groups	62 puzzles
1 perfect group	30 puzzles
2 perfect groups	4 puzzles
4 perfect groups	4 puzzles

The k-means baseline achieved modest performance with only 4% of puzzles solved perfectly. The mean ARI score of 0.284 indicates clustering performance significantly above random chance (ARI = 0) but far from perfect clustering (ARI = 1). Most puzzles (62%) had zero perfect groups, while 30% achieved exactly one perfect group match.

Table 2: Deep Q-Learning Results on 100 NYT Connections Puzzles

Metric	**Value**
Total Puzzles Evaluated	100
Perfect Puzzles	0 (0.0%)
Mean Reward per Puzzle	-2.62
Mean Perfect Groups per Puzzle	0.07
Mean Almost Correct Groups per Puzzle	1.24

For the deep Q-learning model, we see that the model actually performs worse than K-means, with no puzzles fully solved. There were a total of 7 correct groups found across all puzzles, amounting to an average of 0.07 correct groups per puzzle. However, we do see that each puzzle had on average 1.24 almost correct groups, which is significant. This means that the agent was often able to detect words that belong in the same grouping, but wasn't able to identify all words in the grouping. We also see that the mean reward per puzzle is -2.62, which amounts to making 2.62 fully incorrect guesses per puzzle.

For the RLVR fine-tuned model, we observe a marked improvement over both the deep Q-learning baseline and unsupervised clustering methods like K-means. Out of 50 puzzles, the model was able to fully solve 28 (56%) of them, indicating its ability to reason over structured prompts and output coherent groupings. On average, the model identified 2.44 perfect groups per puzzle, suggesting that even in cases where it failed to solve the entire puzzle, it was often able to detect multiple correct groupings. The Adjusted Rand Index (ARI) averaged 0.638 ± 0.451, reflecting high alignment between the predicted and ground-truth clusters. Notably, 29 of the puzzles resulted in zero perfect groups, showing that while the model is capable of high performance, there is still significant variance in its behavior—likely due to the open-ended nature of natural language prompts and the challenge of reward estimation in one-shot settings.

Table 3: RLVR Results on 50 NYT Connections Puzzles

Metric	Value
Total Puzzles Evaluated	50
Perfect Puzzles	56%
Mean ARI Score	0.638 ± 0.451
Mean Perfect Groups per Puzzle	2.44
0 Perfect Groups	29
1 Perfect Group	1
2 Perfect Groups	2
4 Perfect Groups	28

6.2 Qualitative Analysis

6.2.1 K-Means Baseline

The baseline demonstrates that semantic similarity alone is insufficient for solving Connections puzzles. While the model successfully clusters semantically related words (e.g., account-related terms), it fails on categories requiring contextual knowledge or cultural references, such as "Words Before BED" or "Detectives of Kid-Lit." This suggests that more sophisticated approaches incorporating contextual reasoning will be necessary for improved performance. Clustering also fails to account for the natural strategy humans use when solving Connections: identifying the most semantically obvious groups first to reduce ambiguity. By treating all groupings as equally likely, clustering misses the opportunity to lock in high-confidence sets early, shrink the combinatorial search space, and use easy categories as anchors for reasoning through harder, more abstract groupings.

Sample Output - Puzzle 2 (ARI Score: 0.375, Perfect Groups: 1/4)

True Groups:

1. ACCOUNT BOOK: [LEDGER, LOG, RECORD, REGISTER]
2. SEEN IN A BARN: [BALE, HORSE, PITCHFORK, TROUGH]
3. DETECTIVES OF KID-LIT: [BROWN, DREW, HARDY, HOLMES]
4. WORDS BEFORE "BED": [CANOPY, DAY, MURPHY, WATER]

K-Means Predictions:

1. Cluster 0: [DREW, BROWN, DAY, CANOPY]
2. Cluster 1: [RECORD, LOG, REGISTER, LEDGER] ✓
3. Cluster 2: [TROUGH, WATER, PITCHFORK, HORSE]
4. Cluster 3: [HOLMES, MURPHY, HARDY, BALE]

Figure 2: Example k-means clustering output showing one perfect group match (Cluster 1). The model succeeds when semantic similarity aligns with the true grouping but struggles with contextual or cultural knowledge-based categories.

6.2.2 Deep Q-learning

As discussed above, the deep Q-learning model was good at grouping together words that belonged to the same category, but wasn't able to detect all four words in the group. From examples of almost correct guesses, we see that the deep Q-learning agent was able to leverage semantic similarities between words to put them into groups, but struggled to put those guesses in the context of other words on the grid that also needed to be grouped together.

As an example, for one puzzle, the true groupings include "LITTLE WOMEN SISTERS", which are [AMY, BETH, JO, MEG] and "LIL ____ RAPPERS", which are [BABY, JON, KIM, WAYNE]. The agent makes the nearly correct prediction [BETH, JO, MEG, JON], thus correctly grouping together literary first names, but either missing the context of the *Little Women* sisters being grouped together or not looking ahead to finding the other groups.

This presents one of the main issues of this approach, in that our reward structure doesn't incentivize long-term planning; the agent prioritizes short-term rewards by trying to find nearly correct groups, but doesn't learn to make groupings that result in high similarities for all four groups. Perhaps instead of formulating the environment similar to realistically playing the game (i.e. making one guess at a time), this method would work better with each action being a full solution with the four groups, as in the K-means model. The issue with context is a difficult one; the Connections game is tricky because the puzzles include words that have multiple meanings or could belong in multiple groups, and this is hard to detect using simple word embeddings. A more successful model could use more sophisticated natural language techniques to detect more nuanced meanings of the words in the puzzle, and feed that into the observations of our environment.

6.2.3 RLVR-Style Fine-Tuning

Overall, the RLVR approach yields significantly more promising results than both the deep Q-learning baseline and traditional clustering methods. We attribute this improvement to the use of a larger language model with access to broader world knowledge, which is crucial for solving Connections puzzles. Unlike K-means or Q-learning, which rely primarily on local token-level similarity or memorized reward feedback, the RLVR model can reason more holistically about thematic and cultural associations—skills central to the task. This suggests that the puzzle-solving process benefits not just from semantic clustering but also from contextual understanding and commonsense reasoning, which are better captured through reinforcement fine-tuning of a pretrained language model.

Beyond quantitative metrics, we also observe that the RLVR model often produces groupings with strong, interpretable rationales, even in cases where the exact group membership is slightly off. For example, in one completion, the model grouped [ACCOUNT, CHRONICLE, DESCRIPTION, STORY] under a common theme of "writing and record-keeping"—a rationale that clearly aligns with the intended semantic connection. Similarly, it identified that [GUM, LATEX, RESIN, SAP] all relate to natural rubber materials, demonstrating an ability to reason about underlying material properties. Even when the model misclassifies certain items—such as grouping [BALL-IN-CUP, CORN DOG, COTTON SWAB, LOLLIPOP] as "snacks or food items"—it still surfaces a coherent and contextually plausible explanation. This suggests that the model isn't merely memorizing surface-level similarities, but instead leveraging its pretraining to generate grounded, real-world associations, a critical capability for solving Connections-style puzzles.

7 Discussion

Table 4: Performance Comparison

Method	Perfect Solve Rate (%)	Average Groups Matched
Baseline K-Means	4	0.54
Deep Q-Learning	0	0.07
RLVR-Style Fine-Tuning	56	2.44

From our results, we see that the RLVR-Style Fine-Tuning model performed quite well, beating out the baseline K-means model and our RL Deep Q-learning model. This tells us that sparse environmental feedback alone is not enough to overcome the lack of context.

Each of the models had different limitations, but one of the overarching themes was the difficulty in learning connections between the words in the puzzle without a wide context. Many of the words can have multiple meanings and group themes could relate to popular culture, slang, and idioms that weren't always captured in the word embeddings we used for the first two models. Another limitation was the availability of data and compute. We ran our models on AWS with GPUs, but we still didn't have enough compute to run as many experiments as we would have liked.

8 Conclusions and Future Work

In this project, we explored three different methods for solving the *New York Times* Connections game, a challenging task that combines elements of natural language processing with deep reinforcement learning.

After trying baseline methods with K-means clustering and deep Q-learning, we found that the RLVR method worked the best, leveraging LLM priors to find the best groupings. Our experiments also open up many avenues for future work. For the deep Q-learning model, we found that the agent's strategy of always taking the four words with the highest semantic similarity didn't generalize well to having to make four clusters, and so changing the problem formulation to make four guesses at a time might yield more promising results.

An exciting direction for future work is to adopt the one-shot RLVR framework by selecting a single highly informative example to supervise training. Following Wang et al. (2025) we can compute a historical variance score over training examples by first running GRPO training for E epochs, then ranking examples by the variance of their per-epoch training accuracy. Sorting these variances yields a permutation π over the dataset, where π_1 denotes the example with highest variance. We can use this ranking to select a single example (e.g., π_1), then duplicate it to fill the training batch (e.g., 128 copies), enabling RLVR updates using only that one trajectory. The model is then finetuned via GRPO using this single repeated example. Prior work shows that certain high-variance examples lead to surprisingly strong generalization across many benchmarks, and performance may saturate with just one or two examples. This experiment could help determine whether reasoning in the Connections task can be bootstrapped from a single, well-chosen trajectory. Moreover, given that prior work showed one-shot RLVR improves generalization across mathematical reasoning benchmarks like MATH500 and AIME 2025, we are particularly interested in whether training on a single Connections puzzle can similarly enhance the model's broader reasoning ability.

Ultimately, this task and our results are not only interesting for the many players of the game, who may be excited about building something to beat the game, but also hold important connections to real-world NLP and AI applications involving challenges of categorization, pattern recognition, and reasoning between words with multiple definitions and contexts.

9 Team Contributions

We collaborated through all steps of the project, and all worked on the poster and the final report.

- **Linda:** Data processing, baseline K-means model, and RLVR fine-tuning (training and evaluation).
- **Iris:** DQN training, evaluation, reward shaping, implementing HER, and miscellaneous utility scripts.
- **Justin:** DQN environment configuration, DQN architecture, reward shaping, and miscellaneous utility scripts.

Changes from Proposal Our original plan was that Linda would take the lead on gathering data and identifying benchmarks, Iris would take point on the baseline model and pre-trained natural language models, and Justin would lead the EM model. We decided to forgo the EM model and focus on the other two approaches instead. Since Linda was most interested in RLVR and had some relevant background, she was in charge of the implementation and testing. Iris and Justin divided up the deep Q-learning experiment, with Justin focusing more on the environment setup and model architecture and Iris focusing more on setting up the trainer and evaluation mechanisms.

References

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, Pieter Abbeel, and Wojciech Zaremba. 2018. Hindsight Experience Replay. arXiv:1707.01495 [cs.LG] https://arxiv.org/abs/1707.01495

Shuang Li, Xavier Puig, Chris Paxton, Yilun Du, Clinton Wang, Linxi Fan, Tao Chen, De-An Huang, Ekin Akyürek, Anima Anandkumar, Jacob Andreas, Igor Mordatch, Antonio Tor-

ralba, and Yuke Zhu. 2022. Pre-Trained Language Models for Interactive Decision-Making. arXiv:2202.01771 [cs.LG] https://arxiv.org/abs/2202.01771

Angel Yahir Loredo Lopez, Tyler McDonald, and Ali Emami. 2025. NYT-Connections: A Deceptively Simple Text Classification Task that Stumps System-1 Thinkers. arXiv:2412.01621 [cs.CL] https://arxiv.org/abs/2412.01621

Runyu Ma, Jelle Luijkx, Zlatan Ajanovic, and Jens Kober. 2025. ExploRLLM: Guiding Exploration in Reinforcement Learning with Large Language Models. arXiv:2403.09583 [cs.RO] https://arxiv.org/abs/2403.09583

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. arXiv:2402.03300 [cs.CL] https://arxiv.org/abs/2402.03300

Yiping Wang, Qing Yang, Zhiyuan Zeng, Liliang Ren, Lucas Liu, Baolin Peng, Hao Cheng, Xuehai He, Kuan Wang, Jianfeng Gao, Weizhu Chen, Shuohang Wang, Simon Shaolei Du, and Yelong Shen. 2025. Reinforcement Learning for Reasoning in Large Language Models with One Training Example. arXiv:2504.20571 [cs.LG] https://arxiv.org/abs/2504.20571