Solving NYT Connections using Language Models

Taraj Shah tarajsha@usc.edu
Nupoor Dode dode@usc.edu
Neelam Somai somai@usc.edu
Harshavardhan Alimi harshava@usc.edu
Xusheng Feng xushengf@usc.edu

Abstract

This project explores computational approaches for solving the New York Times (NYT) Connections puzzle, a word-categorization challenge where players group 16 words into 4 thematic clusters. Using a dataset of 500+ puzzles, we evaluate models' performance on this task, ranging from traditional clustering algorithms to large language models (LLMs). Initial experiments using K-means clustering with word embeddings produced suboptimal results, underscoring the limitations of static word representations. Advanced approaches such as prompting the LLaMA-3.1-70b (Dubey et al., 2024) LLM showcased significantly better performance achieving a success rate of 25.5% compared to K-means' 1.5%. We explored the T5 model and conducted experiments to evaluate the impact of incorporating priors into T5 model (Raffel et al., 2019), which resulted in a reduction in accuracy from 15.5% to 13%. To investigate the cause of this decline, we performed further error analysis to identify the underlying factors.

1 Introduction

This project investigates computational strategies for solving the New York Times Connections puzzle, a daily word categorization challenge. In this puzzle, players group sixteen words into four thematic categories, each containing four words. The four categories are color-coded to indicate increasing levels of difficulty: yellow for the most straightforward themes, followed by green, blue, and finally purple for the most complex connections. To solve a Connections puzzle, players need to grasp both the literal meanings of the 16 words and the nuances of their contextual usage.

Our objective is to evaluate how effectively language modeling algorithms can solve this puzzle and to explore whether incorporating puzzle-specific prior knowledge improves their performance. We begin by establishing a baseline using a constrained K-Means model on PCA-reduced word2vec embeddings. However, the static nature of word embeddings resulted in poor performance, underscoring their limitations in capturing contextual meaning and polysemy.

To address these shortcomings, we shifted our focus to Large Language Models (LLMs), leveraging their contextual understanding and ability to handle polysemy. Prompting the LLaMA-3.1-70b (Dubey et al., 2024) LLM yields significant improvements over the K-Means baseline but still falls short of human-level performance, establishing the task as inherently challenging for existing models.

To better understand the nature of the problem, we analyze a dataset of 500+ puzzles, identifying recurring patterns and thematic categories using tools such as NLTK and CMUdict. Building on these insights, we incorporate domain-specific prior knowledge into a fine-tuned T5 model (Raffel et al., 2019), reframing the task as text generation. Initial experiments using category labels as priors yield performance comparable to LLMs, demonstrating the potential of prior knowledge integration.

However, subsequent results using domain-specific priors were less promising compared to those using only category labels. Through error analysis, we observed that priors consuming a significant portion of the training space led to overfitting, causing them to dominate the predictions across all labels during testing. In contrast, priors that were infrequent in the training data failed to appear in the test set predictions. This imbalance in the training data negatively affected the inference results, limiting the effectiveness of incorporating domain-specific priors.

2 Related Work

In our project, we use WordNet, a lexical database by Miller et al. (1990), to explore connections between English words through their semantic relationships, such as synonymy and hypernymy. This foundational tool aids in categorizing words for the New York Times Connections puzzle, enhancing our understanding of word associations, and improving word embedding models.

We build on recent research, including Samadarshi et al. (2024), which evaluated the reasoning abilities of large language models (LLMs) such as Claude 3.5 Sonnet and GPT-4 (Achiam et al., 2023) on Connections puzzles. Their findings indicated that while LLMs could partially solve these puzzles, they significantly lagged behind expert human players, with Claude 3.5 achieving only 18% perfect solutions.

Earlier studies by Allaway and McKeown (2021) highlight that LLMs excel in character-based puzzles but struggle with abstract tasks that require associative thinking. Our approach investigates whether integrating domain-specific knowledge within word embeddings can enhance Connections puzzle performance, aiming to bridge the reasoning gaps of LLMs through enriched embeddings.

Saha et al. (2024) explores the use of Large Language Models (LLMs) for solving crossword puzzles. The authors develop a search algorithm that enables LLMs to solve complete crossword grids, achieving 93% accuracy on New York Times Crosswords puzzles. Their results challenge previous findings that LLMs lag behind human experts, suggesting a narrower performance gap. This work is closely related to our project, as both the Crossword Puzzle and the Connections puzzle require advanced language understanding, reasoning, and world knowledge. Similar to the research in this paper, we aim to formulate methods to improve previously observed performance of Language models in solving word games, in our case incorporating priors to solve New York Times Connections.

The T5 (Text-to-Text Transfer Transformer) model, introduced by Raffel et al. 2019, provides another perspective on leveraging pre-trained language models for tasks requiring reasoning and semantic understanding. It reframes every NLP task as a text-to-text problem, offering flexibility in handling diverse tasks, including reasoning-based ones. Although T5 (Raffel et al., 2019) has achieved state-of-the-art results on various benchmarks, its performance on associative reasoning tasks, such as the Connections puzzle, remains an area of exploration. Studies such as Agarwal et al. 2023 demonstrated that fine-tuning T5 (Raffel et al., 2019) with domain-specific data can improve its performance in specialized tasks. However, its reliance on large-scale pretraining may limit its ability to generalize to puzzles that require deep semantic connections unless fine-tuned with carefully curated datasets enriched with external knowledge such as WordNet (Miller et al., 1990).

3 Methodology

3.1 Dataset

We gather our dataset from an external API [1], covering puzzles from June 2023 to October 2024. The dataset consists of 503 puzzles, each containing 16 words grouped into 4 thematic categories. Each group includes a label (e.g., "WET WEATHER," "NBA TEAMS"), a level indicating the difficulty, with level 0 being the easiest and level 3 being the most difficult, and a list of members that are semantically related words. For example, one group labeled "WET WEATHER" contains the words "HAIL," "RAIN," "SLEET," and "SNOW," while another labeled "NBA TEAMS" includes "BUCKS," "HEAT," "JAZZ," and "NETS." Each puzzle also comes with a unique identifier and a creation date. The dataset structure can be seen in Appendix A. After excluding emoji-based puzzles, we retained 499 puzzles and split them into train, validation, and test sets using an 80-10-10 split. This structured dataset, with clear labels and word groupings, forms the basis for evaluating models on tasks involving word associations and semantic reasoning.

3.1.1 Data Composition

We used NLTK's direct POS tagger to perform word-level POS tagging and retrieved the most common synset sense for each word from WordNet. Next, we performed a group-level POS intersection, identifying the number of groups with common POS tags across all four words in each category. Initially, 294 words had no entries in WordNet (Miller et al., 1990), and using only the most common sense resulted in 485 groups without intersections. By considering all possible POS tags from WordNet, we improved the intersection

[1] raw.githubusercontent.com/Eyefyre/NYT-Connections-Answers/main/connections.json

POS	Word-level		Category Level Intersection	
	NLTK tagger	**WordNet**	**NLTK tagger**	**WordNet**
Noun	5761	5646	1150	1110
Adjective	288	188	0	0
Adverb	82	73	0	0
Verb	159	233	0	1
Other	94	0	3	0
None	0	294	443	485

Table 1: POS Tagging Results

Category	count
Synonym	840
Hypernym	533
Contextual Relation	1039
Phonology	4
Homophones	13
Collocation	146
Rhyming	13
Cultural References	23
Pop-culture References	19
Miscellaneous	418

Table 2: Semantic relation categories

count to 322, effectively grouping words using less common senses (see Appendix D), showing the effective grouping of words using less common senses.

3.1.2 Data Categories

Based on the words' functions, forms, and meanings, we categorized the dataset into three main groups: Semantic Association, Word Forms, and Word Forms + Meanings.

3.1.3 Semantic Association

Most NYT Connection puzzles employ on semantic associations. We categorized the data into several semantic relation categories, see Appendix D.
Synonymy: Words sharing overlapping synsets were identified using WordNet and grouped together
Hypernymy: Words sharing common hypernyms (excluding the five most common ones) were identified using WordNet and grouped together
Contextual Relation: We used WordNet similarity scores, grouping together words when the vector similarity exceeded a predefined threshold.

3.1.4 Word Forms

This category focuses on the phonological and morphological aspects of the words in the puzzle:
Phonology: Words sharing same silent letters were identified using category labels
Homophones: We used CMUdict to identify homophones and analyzed their relationships.
Rhyming: Words with similar rhyming patterns were grouped using pronunciation data from CMUdict.

3.1.5 Word Form + Meaning

Here, words are grouped based on their collocational and cultural relevance:
Collocation: Phrases in the category labels that commonly co-occur with a shared term often in a fill-in-the-blank style format were grouped together.
Cultural References: We identified words related to specific cultural contexts (e.g., horror movies). In our analysis, we specifically focused on movie references by searching for the term movie within the category labels
Pop-culture References: Slang terms with similar meanings were grouped together by searching for slang terms within the category labels.

3.1.6 Miscellaneous

We explored additional relations that did not fit into the previously identified classifications by looking into the category labels for such groups. Some notable relations identified were as follows:
Character Modifications: Identified semantic relationships established by adding or removing characters from words
Acronyms: Identifying relationships through acronyms.
App-Related Terms: Terms related to app functionalities (e.g., Microsoft fonts, dating app actions).
Common Origins: Words derived from common linguistic origins (e.g., Latin or Greek).

Cultural or Regional Words: Words related to specific cultures or regions (e.g., Chinese or Mexican references).

3.1.7 Data Exploration Results

The dataset analysis results summarized in Tables 1 and 4 revealed key insights into the distribution and challenges of the puzzle data. Nouns were the most frequent POS, appearing 5761 times, while adjectives and adverbs were less common. Notably, 294 words had no POS tags assigned in WordNet (Miller et al., 1990), indicating challenges in POS mapping. By considering all possible POS tags, the intersection of WordNet senses improved, reducing uncategorized groups from 485 to 322. Semantic associations, particularly Synonymy and Contextual Relations, were most prevalent, while categories like Phonology, Homophones, and Rhyming occurred less frequently. These insights highlight the importance of semantic relationships in solving the Connections puzzle and guide our efforts to enhance Language Model performance in such tasks.

3.2 Solving task using K-means clustering

3.2.1 Data Preprocessing

We initiated our analysis using the adarshsng/googlenewsvectors dataset, which comprises word2vec embeddings for 3 million words, pre-trained on a 3 billion word corpus from Google News [2]. These pre-trained embeddings will facilitate the clustering of words in a single Connections puzzle based on their semantic similarity.

To reduce the computational load associated with 300-dimensional vectors, we applied Principal Component Analysis (PCA), reducing each word embedding to 2 dimensions. This dimensionality reduction aids visualization (Appendix B) and optimizes clustering performance for the puzzle.

3.2.2 Experiment Setup

With the PCA-reduced word2vec embeddings ready, we applied the constrained K-means clustering algorithm to predict solutions for each puzzle in our test dataset. This algorithm ensures that each cluster contains exactly four words.

Clustering was performed using a similarity matrix created by calculating the cosine similarity between the 16 PCA-reduced embeddings. The model outputs a 4x4 matrix of words, where each row represents a cluster of four closely related words identified for grouping.

3.3 Solving task using LLAMA

3.3.1 Data Preprocessing

To prepare the data, we first process the puzzle data received from the API, constructing an array of size 16, where each element represents a single word. Each array thus corresponds to a "16-word connections puzzle," forming part of the input instruction for the LLama model. We access the Llama-3.1-70b model through the Groq API to leverage its large language model capabilities for word clustering.

3.3.2 Experiment Setup

We used prompting technique as outlined in Samadarshi et al. (2024), the prompt is available in Appendix C, we applied to our test dataset of 16 words, guiding the model to generate predictions in the form of a 4x4 matrix. Each row of this matrix represents a cluster, with each cluster containing four words that the model associates based on latent semantic or categorical relationships. This clustering approach provides insight into the model's interpretative capabilities for word association tasks in structured puzzles.

3.4 Solving task using T5 Model

3.4.1 Data Preprocessing

In this experiment, we begin incorporating priors into the models to evaluate their effect on performance. To leverage T5 model (Raffel et al., 2019), we transform our task into a text generation problem by constructing structured input-output pairs for each puzzle in the dataset. We design two versions of the task: one without priors and another with priors (see Appendices E and F for examples). In the without priors version (Appendix E), the input instructs the model to cluster the 16 words into 4 groups with proper reasoning but does not provide any hints regarding relationships or group categories. The model must deduce these connections independently. In the with priors version (Appendix F), the input explicitly provides possible relationships, such as hypernyms, homophones, semantic similarity, or shared connections to pop culture, among others. This additional prior knowledge serves as a guide, enabling the model to focus on specific linguistic or semantic clues during the clustering task. The expected output for both versions includes the four solution groups, along with

[2] https://www.kaggle.com/datasets/adarshsng/googlenewsvectors

a corresponding label or reasoning for each group. By comparing performance on both tasks, we aim to analyze the impact of integrating prior knowledge into the training data and its effectiveness in improving prediction accuracy during the testing phase.

3.4.2 Experiment Setup

We fine-tuned the T5 model (Miller et al., 1990) on the training data in both setups (with and without priors) to produce the desired output, then evaluated its performance on the test data. To compare performance, we tested three variants of the T5 model: T5-small (Raffel et al., 2020), T5-large (Raffel et al., 2020), and T5-Flan (Chung et al., 2024).

Fine-tuning was conducted with a learning rate of 1e-4 using the AdamW optimizer, where the weight decay was set to 0.01 and the adam epsilon to 1e-6 to ensure stable convergence. Training was conducted over 100 epochs for the T5-Small model and 40 epochs for the T5-Large and T5-Flan models.

During inference, we employed beam search with num_beams = 5 to consider multiple candidate outputs. To reduce repetitive predictions, we applied a repetition penalty of 5.0, with a length penalty of 1.0 to balance the output length. Early stopping was enabled to terminate decoding once a complete solution was found. Additionally, we incorporated sampling techniques by setting do_sample = True, with a temperature of 0.8 to encourage diversity and top-p (nucleus sampling) at 0.9 to focus on the most probable tokens.

These hyperparameter choices ensured a balance between output diversity, fluency, and accuracy

3.5 Evaluation

To evaluate prediction accuracy, we used the Success Rate and Jaccard similarity metrics to compare our generated clusters against the actual solutions

We define the two metrics in the context of our project as follows:

Jaccard Similarity: A measure of similarity between the actual and predicted groups, averaged across the best matches calculated as:

$$Jacc = \frac{|Actual\_grp \cap Predicted\_grp|}{|Actual\_grp \cup Predicted\_grp|} \quad (1)$$

This score, ranging from 0 to 1, measures the overlap between predicted and actual groups, serving as a partial accuracy metric that rewards predictions with some correct words, even if the group is not entirely accurate.

Jaccard similarity in this evaluation is calculated using an optimal one-to-one matching between predicted and ground truth clusters. First, a 4x4 similarity matrix is constructed, where each entry represents the Jaccard similarity score between an actual (ground truth) group and a predicted group. To determine which clusters to evaluate against each other, the algorithm iteratively selects the maximum score from the matrix, corresponding to the best match between one predicted group and one ground truth group. Once a match is chosen, the associated row (ground truth) and column (predicted) are removed from further consideration. This process continues until all clusters are paired, ensuring each predicted group is matched to at most one actual group. The selected Jaccard scores are then averaged to provide the final similarity measure.

Success Rate: Proportion of correctly predicted groups, calculated as:

$$SuccessRate = \frac{No.of ExactMatches}{4} \quad (2)$$

This score measures how many of the 4 groups the model predicted perfectly, serving as a strict accuracy metric that awards points only for fully correct group predictions.

We assess the overall model performance by averaging the success rate and Jaccard similarity on a test dataset of 50 random puzzles and found the following results in Table 3.

3.6 Results

The baseline K-means clustering model achieved a Jaccard Similarity of 0.3733 and a Success Rate of 1.5%, highlighting its limited ability to identify relationships between words. Interestingly, the T5-Small model (Raffel et al., 2020) underperformed compared to K-means, likely due to its smaller size. This hypothesis is supported by the trend observed across the T5 models, where performance improved as model size increased. The T5-Flan model (Chung et al., 2024) delivered the best results among the T5 variants, achieving a Jaccard Similarity of 0.4960, comparable to the state-of-the-art LLaMA-3.1-70b model (Dubey et al., 2024). While LLaMA-3.1-70b had a higher Success Rate (25.5%), the T5-Flan model (Chung et al., 2024) excelled in Jaccard Similarity, reflecting a balance between partial and exact group predictions.

Model	Jaccard Similarity	Success Rate
K-means clustering	0.3733	0.015
T5-Small	0.3438	0.01
T5-Large	0.4650	0.10
T5-Flan	**0.4960**	0.155
T5-Flan(All priors)	0.4745	0.11
LLama 3.1-70b	0.4641	**0.255**

Table 3: Comparison of results produces by LLama, K-means clustering and T5 model.

Priors	Jaccard Similarity	Success Rate
Synonym	0.4495	0.11
Hypernym	0.4597	0.11
Contextual Relation	0.4989	0.135
Phonology	0.4932	0.155
Homophones	0.4762	0.14
Collocation	0.4900	0.13
Rhyming	0.4856	0.16
Cultural References	**0.5221**	0.16
Pop-culture References	0.5218	**0.165**
No priors	0.4960	0.155
ALL priors	0.4745	0.11

Table 4: Ablation study comparing the performance of the model with no priors, all priors, or only specific priors, alongside reason labels.

3.6.1 Ablation Study

We expected the results to improve with the incorporation of priors; however, surprisingly, the performance degraded. To understand this, we conducted an ablation study with several setups for T5 fine-tuning.

First, we fine-tuned the T5 model while freezing all the weights except for the final layer weights. In this experiment, we observed that even with many epochs, the model failed to produce the expected output. Additionally, when the outputs were generated, they were neither grouped into four distinct categories nor consistently placed within the correct group, with some repetition occurring within the same group. This suggests that fine-tuning only the final layer is insufficient, highlighting the importance of fine-tuning all layers in the network. Fine-tuning all layers allows the model to adjust its internal representations and capture more complex relationships, improving its overall performance.

Second, we fine-tuned the model by incorporating the priors individually, one at a time, instead of using all priors in the output. This experiment revealed that very frequent priors, such as hypernyms, synonyms, and contextual references, often appeared in the output, even when the cluster was not intended to belong to the prior category. This behavior can be attributed to the heavy occurrence of these priors in the training data, which caused confusion between the prior sentence and the output format. Instead of generating reasoning, the model mistakenly treated the prior information as part of the output, indicating that the model was not fully understanding the significance of these priors.

On the other hand, including less frequent priors performed better than having no priors at all, as shown in Table 4. This can be attributed to the model recognizing them as part of the reasoning process rather than the output structure, although these priors were still underrepresented in the final output since they were not predicted during generation.

In conclusion, the study suggests that we need to restructure the dataset to balance the frequency of priors. This would involve increasing the representation of less frequent priors to ensure they are adequately captured, while reducing the occurrence of more frequent priors to prevent them from being treated as part of the output structure.

4 Future Work

Moving forward, the importance of generating more data cannot be overstated, as our current dataset consists of only 500 puzzles, which limits the model's ability to generalize effectively. To address this, we plan to generate synthetic data to augment our training set, thereby enhancing the model's robustness. Furthermore, we recognize the need to balance the representation of priors in the dataset. By increasing the presence of less frequent priors and reducing the occurrence of more frequent priors, we can better capture nuanced semantic relationships and avoid over-reliance on dominant priors. We also plan to explore reinforcement learning approaches or auxiliary losses to ensure that priors are treated as reasoning cues and not part of the output.

Additionally, we aim to integrate external encyclopedic datasets and knowledge graphs to provide richer contextual understanding, which will aid in improving the model's reasoning capabilities. We also intend to explore ensemble methods that combine large language models (LLMs) with heuristic algorithms, enabling more sophisticated reasoning and enhancing overall performance. Through these efforts, we aim to refine the model's ability to make more accurate associations, ultimately improving its performance on the NYT Connections puzzle.

Another key observation from the current model's outputs is its tendency to place words in multiple groups or assign random words not part of the original 16 to a group. This disrupts the clustering process and results in incorrect outputs. To address this, we plan to explore techniques to better handle repetitions and random words in the model's output. This will be coupled with tweaking the loss function to heavily penalize these kinds of errors, encouraging the model to produce more accurate and logically consistent groupings. Through these efforts, we aim to refine the model's ability to make more accurate associations and ultimately improve its performance on the NYT Connections puzzle.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Ankush Agarwal, Sakharam Gawade, Amar Prakash Azad, and Pushpak Bhattacharyya. 2023. Kitlm: Domain-specific knowledge integration into language models for question answering. *Preprint*, arXiv:2308.03638.

Emily Allaway and Kathleen McKeown. 2021. A unified feature representation for lexical connotations. *Preprint*, arXiv:2006.00635.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. J. Miller. 1990. Introduction to WordNet: an on-line lexical database. *International Journal of Lexicography*, 3(4):235–244.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *Preprint*, arXiv:1910.10683.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Soumadeep Saha, Sutanoya Chakraborty, Saptarshi Saha, and Utpal Garain. 2024. Language models are crossword solvers. *Preprint*, arXiv:2406.09043.

Prisha Samadarshi, Mariam Mustafa, Anushka Kulkarni, Raven Rothkopf, Tuhin Chakrabarty, and Smaranda Muresan. 2024. Connecting the dots: Evaluating abstract reasoning capabilities of llms using the new york times connections word game. *Preprint*, arXiv:2406.11012.

5 Appendices

A Dataset Structure

Figure 1 shows the structure.

B Visualization of PCA-reduced embedding vectors

A scatter plot of the reduced embeddings for a sample puzzle, as seen in Figure 2 shows no obvious clustering patterns among words that belong to the same solution group, likely due to the embeddings' general-purpose nature, which does not capture task-specific group memberships.

C Prompt

Find groups of four items that share something in common.
Category Examples:
FISH: Bass, Flounder, Salmon, Trout
FIRE ___: Ant, Drill, Island, Opal
Categories will always be more specific than '5-LETTER-WORDS', 'NAMES', or 'VERBS.'
Example 1:
Words: ['DART', 'HEM', 'PLEAT', 'SEAM', 'CAN', 'CURE', 'DRY', 'FREEZE', 'BITE',

‘EDGE’, ‘PUNCH’, ‘SPICE’, ‘CONDO’, ‘HAW’, ‘HERO’, ‘LOO’]
Groupings:
1. Things to sew: [‘DART’, ‘HEM’, ‘PLEAT’, ‘SEAM’]
2. Ways to preserve food: [‘CAN’, ‘CURE’, ‘DRY’, ‘FREEZE’]
3. Sharp quality: [‘BITE’, ‘EDGE’, ‘PUNCH’, ‘SPICE’]
4. Birds minus last letter: [‘CONDO’, ‘HAW’, ‘HERO’, ‘LOO’]
Example 2:
Words: [’COLLECTIVE’, ‘COMMON’, ‘JOINT’, ‘MUTUAL’, ‘CLEAR’, ‘DRAIN’, ‘EMPTY’, ‘FLUSH’, ‘CIGARETTE’, ‘PENCIL’, ‘TICKET’, ‘TOE’, ‘AMERICAN’, ‘FEVER’, ‘LUCID’, ‘PIPE’]
Groupings:
1. Shared: [‘COLLECTIVE’, ‘COMMON’, ‘JOINT’, ‘MUTUAL’]
2. Rid of contents: [‘CLEAR’, ‘DRAIN’, ‘EMPTY’, ‘FLUSH’]
3. Associated with “stub”: [‘CIGARETTE’,‘PENCIL’, ‘TICKET’, ‘TOE’]
4. __ Dream: [‘AMERICAN’, ‘FEVER’, ‘LUCID’, ‘PIPE’])
Example 3:
Words: [‘HANGAR’, ‘RUNWAY’, ‘TARMAC’, ‘TERMINAL’, ‘ACTION’, ‘CLAIM’, ‘COMPLAINT’, ‘LAWSUIT’, ‘BEANBAG’, ‘CLUB’, ‘RING’, ‘TORCH’, ‘FOXGLOVE’, ‘GUMSHOE’, ‘TURNCOAT’, ‘WINDSOCK’]
Groupings:
1. Parts of an airport: [‘HANGAR’, ‘RUNWAY’, ‘TARMAC’, ‘TERMINAL’]
2. Legal terms: [‘ACTION’, ‘CLAIM’, ‘COMPLAINT’, ‘LAWSUIT’]
3. Things a juggler juggles: [‘BEANBAG’,‘CLUB’, ‘RING’, ‘TORCH’]
4. Words ending in clothing: [‘FOXGLOVE’, ‘GUMSHOE’, ‘TURNCOAT’, ‘WIND SOCK’]

Categories share commonalities:
- There are 4 categories of 4 words each
- Every word will be in only 1 category
- One word will never be in two categories
- As the category number increases, the connections between the words and their category become more obscure. Category 1 is the most easy and intuitive and Category 4 is the hardest
- There may be a red herrings (words that seems to belong together but actually are in separate categories)
- Category 4 often contains compound words with a common prefix or suffix word
- A few other common categories include word and letter patterns, pop culture clues (such as music and movie titles) and fill-in-the-blank phrases You will be given a new example (Example 4) with today’s list of words.

Give your final answer following the structure below [[word1, word2, word3, word4],[word5, word6, word7, word8],[word9, word10, word11, word12],[word13, word14, word15, word16]]

Remember that the same word cannot be repeated across multiple categories, and you need to output 4 categories with 4 distinct words each.
Also do not make up words not in the list. This is the most important rule. Please obey

Today’s list of words are:

# Taxanomies	Count
0	418
1	261
2	384
3	516
4	16
5	1

Table 5: The table provides information on how each sample can be categorized into different categories.

POS	WordNet (All senses)
(noun)	828
(noun, verb)	269
(verb)	102
(others)	75
(None)	322

Table 6: Effective grouping of words using less common senses

D Table

Table 5 and Table 6 provides more details.

E Input-Output Example for T5 Model without priors

Input:-
Instruction:
Cluster the following 16 words into 4 groups with

no repetition. Provide proper reasoning for the clustering, specifying if they fall under hypernyms, homophones, or other linguistic relationships. Words are: [words 1-16]
Output:- example
Group1: article, essay, paper, report
Reason1: The words are writing assignments
Group2: bubble gum, carnation, eraser, flamingo
Reason2: The words are things that are pink
Group3: foxtrot, hotel, victor, yankee
Reason3: The words are nato phonetic alphabet
Group4: deputy, derek jeter, pencil, silver medal
Reason4: The words are associated with 2

F Input-Output Example for T5 Model with priors

Input:
Instruction:
You are tasked with clustering the following 16 words into 4 distinct groups. Each word must belong to only one group, and no repetition of words is allowed. The clustering should be based on specific linguistic or semantic relationships. For each group, provide:
The cluster name or the relationship type (e.g., hypernyms, homophones, synonyms, etc.).
The reasoning behind why the words in the group belong together.

When clustering, consider the following possible relationships among the words:
Words that have hierarchy relationship (a broader category they fall under).
Words that have a shared homophone. (sound the same but have different meanings/spellings).
Words that often appear with common preceding or succeeding words.
Words that serve as similar fillers in blanks (e.g., in a sentence or phrase).
Words with the same rhyming pattern.
Words that are semantic modifications of others (e.g., plural/singular, prefixes, antonyms).
Words with the same silent letters in their spelling.
Words with connections to movies or pop culture.
Words that are slang terms for the same or similar concepts.
Words that are synonyms (mean the same or nearly the same).
Words that exhibit semantic vector similarity (close in meaning in a computational model).
Words are: [words 1-16]

Output: example
Group1: cooler, lantern, sleeping bag, tent
The words have hierarchy relationship.
The words exhibit semantic vector similarity.
Reason1: The words are camping supplies
Group2: barb, dig, diss, jab
The words are synonyms.
Reason2: The words are insult
Group3: good, impossible, nothing, warren
Reason3: The words are ___ burger
Teh words are similar fillers for the blanks.
Grou4: gnat, gnaw, gnocchi, gnome
The words have same silent letters.
Reason4: The words are silent "g"
All possible priors:- The words have hierarchy relationship.
The words have a shared homophone.
The words follow common predecessors.
The words are followed by common successors.
The words are similar fillers for the blanks.
The words have same rhyming pattern.
The words are plus or minus modifications of other semantics.
The words have same silent letters.
The words have a connection to movies.
The words are slang terms for a common word.
The words are synonyms.
The words exhibit semantic vector similarity.

G Group Contributions

G.1 Ideas

1. Solving the NYT Connections puzzle using LLMs – Nupoor, Neelam
2. Dataset identification – Xusheng, Taraj
3. Creating prompts for Llama – Taraj, Harshavardhan
4. Structuring input and output for T5 fine-tuning – Harshavardhan, Nupoor
5. Steps for conducting the ablation study – Xusheng, Neelam, Taraj, Nupoor
6. Jaccard similarity and success rate design – Neelam, Xusheng

G.2 Data Analysis

1. Performing POS tagging using WordNet and NLTK – Neelam
2. Using WordNet for hypernyms, synonyms, and contextual relationships – Harshavardhan
3. Using CMUdict to analyze word forms, including pronunciations – Nupoor
4. Exploring additional analyses, such as automatic

identification of slang and identification of silent letters using modules. (Note: These methods did not work as expected – despite having one cluster, only 2-3 words from it were identified, so these results were not included in the report.) – Taraj
5. Using labels to identify the remaining priors – Xusheng
6. Creating pipelined data for Llama and K-means – Nupoor, Neelam
7. Creating a pipelined structured dataset for T5 – Harshavardhan, Taraj

G.3 LLAMA

1. Creation of the Groq API and coding for the entire baseline model – Harshavardhan
2. Jaccard similarity and success rate metrics implementation – Taraj PCA and K-means:
3. Loading GoogleNews-vectors-negative300 to encode all the words in the dataset into word embeddings and record all unrecognized words – Xusheng
4. Applying K-means clustering algorithm to cluster 16 words into 4 groups based on the cosine similarity of word embeddings – Nupoor
5. Using PCA to address the curse of dimensionality and analyze how this affects clustering. This analysis helps decide how many top components to remove from the embeddings, making the remaining components more discriminative – Neelam
6. PPA Implementation (though not used later) – Xusheng

G.4 T5

1. Initial experimental setup for T5 – Nupoor
2. Implementing the model on CARC and setting up training – Harshavardhan
3. Handling training issues, such as CUDA memory errors, and working on new techniques to manage memory constraints – Taraj, Harshavardhan
4. Implementing error analysis techniques, such as using only priors and freezing layers – Neelam

G.5 Code Structuring

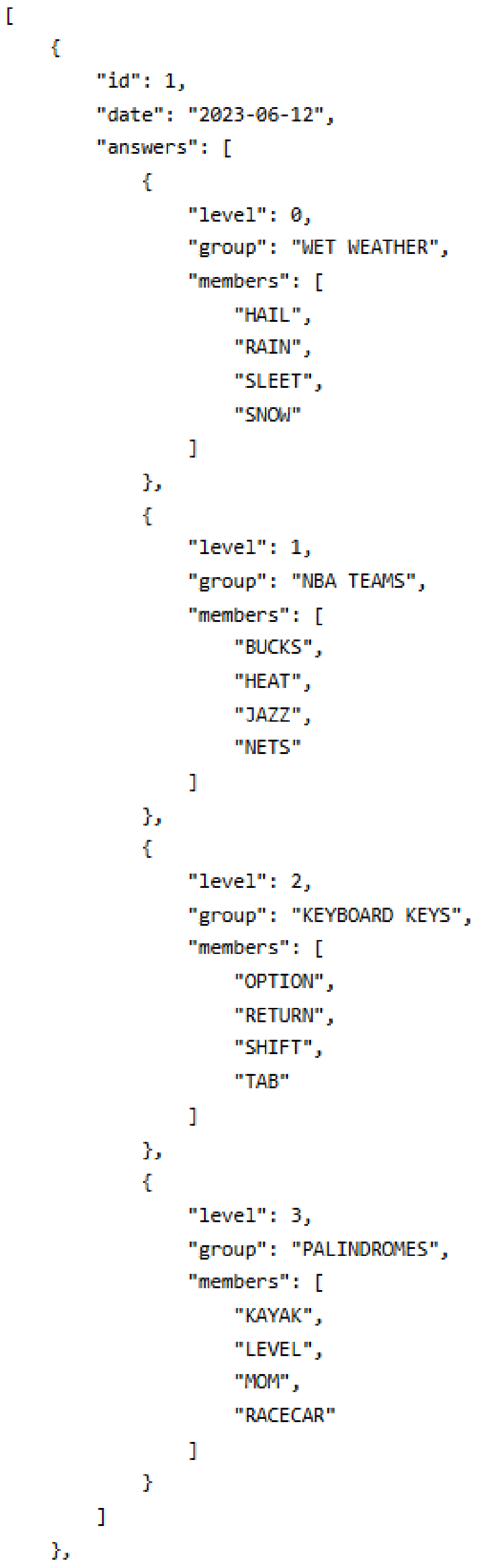
Organizing code into a single file for improved structure and manageability – Xusheng

G.6 Slides, Reports, and Paper Reading

All team members

G.7 Brainstorming Ideas

Brihi



```
[
    {
        "id": 1,
        "date": "2023-06-12",
        "answers": [
            {
                "level": 0,
                "group": "WET WEATHER",
                "members": [
                    "HAIL",
                    "RAIN",
                    "SLEET",
                    "SNOW"
                ]
            },
            {
                "level": 1,
                "group": "NBA TEAMS",
                "members": [
                    "BUCKS",
                    "HEAT",
                    "JAZZ",
                    "NETS"
                ]
            },
            {
                "level": 2,
                "group": "KEYBOARD KEYS",
                "members": [
                    "OPTION",
                    "RETURN",
                    "SHIFT",
                    "TAB"
                ]
            },
            {
                "level": 3,
                "group": "PALINDROMES",
                "members": [
                    "KAYAK",
                    "LEVEL",
                    "MOM",
                    "RACECAR"
                ]
            }
        ]
    },
```

Figure 1

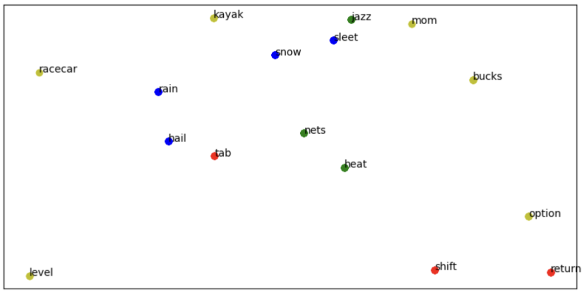


Figure 2: Embedding values for the 16 words in the Connections puzzle visualized on a scatter plot.