Strategic Deductive Reasoning in Large Language Models: A Dual-Agent Approach

Siyue Li *, Xiaofan Zhou, Zhizhong Wu, Yujian Long, Yanxin Shen

Posted Date: 24 September 2024

doi: 10.20944/preprints202409.1875.v1

Keywords: Language Models; Deductive Reasoning; Information Collection; Pre-trained Models; Strategic Reasoning

Article

Strategic Deductive Reasoning in Large Language Models: A Dual-Agent Approach

Siyue Li [1,*], Xiaofan Zhou [2], Zhizhong Wu [3], Yujian Long [4] and Yanxin Shen [5]

[1] Northeastern University, Santa Clara, USA
[2] University of Florida, Gainesville, USA
[3] Independent Researcher, Mountain View, USA
[4] Independent Researcher, Frisco, USA
[5] Independent Researcher, Hangzhou,China
* Correspondence: janelovelee2020@gmail.com

Abstract: This study explores the enhancement of deductive reasoning capabilities in Large Language Models (LLMs) through a strategic dual-agent framework. In this framework, one agent acts as a questioner and another as an answerer, with both employing advanced linguistic and logical processing to optimize information exchange. Utilizing a structured environment that limits query opportunities, our approach emphasizes the development of LLMs that can efficiently generate and interpret questions to deduce hidden information effectively. The models, which incorporate self-defined agents with a combination of pretraining and llama-3-8b enhancements, demonstrate a remarkable ability to navigate the complexities of logical deduction. Performance evaluations, based on a series of simulated interactions, illustrate the agents' improved precision and strategic acumen in narrowing down possibilities through targeted inquiries. These findings underscore the potential of LLMs in tasks requiring intricate reasoning and collaboration, marking a significant step towards more intelligent and autonomous systems.

Keywords: language models; deductive reasoning; information collection; pre-trained models; strategic reasoning

1. Introduction

The "20 Questions" game is a classic example of a deductive reasoning task where a player attempts to guess a secret word by asking up to 20 yes-or-no questions. The game's objective is to strategically narrow down the possibilities to identify the secret word efficiently. This makes it an ideal testbed for evaluating the capabilities of large language models (LLMs) in terms of reasoning, strategic questioning, and information gathering.

Large language models, such as GPT-3, BERT, and more recently Llama-3-8b, have demonstrated remarkable proficiency in various natural language processing (NLP) tasks. These tasks include text generation, question answering, summarization, and more. However, their performance in structured and strategic reasoning tasks, such as the "20 Questions" game, has not been extensively studied. This gap presents an opportunity to explore how these models can be applied to scenarios requiring both understanding and strategic interaction.

In this study, we implement and evaluate various LLMs in the context of the "20 Questions" game. The game involves two roles: the questioner, who asks questions to identify the secret word, and the answerer, who responds with yes or no. The performance of the models is assessed based on their ability to ask effective questions, gather relevant information, and deduce the correct word within the least number of questions.

Our approach includes using custom-designed agents and integrating state-of-the-art pre-trained models like Llama-3-8b. We also develop specific strategies for question formulation and response interpretation to maximize the efficiency and accuracy of the guessing process. The study aims to highlight the strengths and limitations of LLMs in strategic reasoning and collaborative tasks.

 doi:10.20944/preprints202409.1875.v

The remainder of this paper is organized as follows: Section II reviews related work in the field of LLMs and their applications in reasoning tasks. Section III describes the methodology, including the design of agents and the game environment. Section IV presents the experimental results and analysis. Finally, Section V concludes the paper with a discussion of the findings and potential future work.

2. Related Work

The application of large language models (LLMs) in reasoning and strategic tasks has garnered significant attention in recent years. This section reviews key contributions in the field, focusing on the use of LLMs in question answering, text generation, and logical reasoning.

Brown et al.[1] introduces GPT-3, a powerful language model capable of generating human-like text and performing various NLP tasks with minimal fine-tuning. Devlin et al.[2] study BERT model, which excels in understanding context in language, significantly improves performance on multiple NLP benchmarks. Radford et al.[3] GPT-2 demonstrated the ability of LLMs to generate coherent and contextually relevant text over long passages, highlighting their potential in creative writing and dialogue systems. Kenton et al.[4] The DistilBERT model offers a more compact and faster variant of BERT with minimal loss in accuracy, making it suitable for deployment in resource-constrained environments.

Yang et al.[5] XLNet, an autoregressive pre-training model, addresses limitations in BERT and enhances language understanding capabilities. Clark et al.[6] introduces a novel pre-training method that trains models more efficiently and effectively by focusing on distinguishing real input tokens from fake ones generated by a small generator network. Zhang et al.[7] review of NLP Applications in the Field of Text Sentiment Analysis. Journal of Industrial Technology and Applications, Dong et al.[8] UniLM, a unified pre-training model for both natural language understanding and generation, leverages a shared transformer network for diverse NLP tasks.

Raffel et al.[9] frames NLP tasks as text-to-text problems, providing a unified approach that simplifies the task formulation and enhances model versatility. Yan et al.[10] discuss the role of NLP in improving data mining and information retrieval in big data environments. Lan et al.[11] reduces model size while maintaining high performance by sharing parameters across layers and using factorized embedding parameterization. Song et al.[12] MASS, a sequence-to-sequence pre-training method, significantly improves the quality of text generation by masking continuous fragments of the input sentence.

Lewis et al.[13] BART, a denoising autoencoder for pre-training sequence-to-sequence models, demonstrates strong performance in text generation and comprehension tasks. Zhang et al.[14] Pegasus model, designed for abstractive text summarization, pre-trains by masking whole sentences and generating missing segments, achieving state-of-the-art results in summarization benchmarks. Zhang et al.[15] DIALOGPT, a large-scale pre-trained dialogue model, fine-tuned on conversational data, shows significant improvements in generating relevant and coherent responses in dialogue systems. He et al.[16] DeBERTa introduces disentangled attention mechanisms and enhanced positional encodings, further improving performance on a variety of NLP tasks.

Shoeybi et al.[17] Megatron-LM scales up transformer models using model parallelism, achieving significant performance improvements in language modeling tasks. Black et al.[18] GPT-Neo, an open-source model replicating GPT-3 capabilities, demonstrates the potential of community-driven development in advancing LLM performance. Wang et al.[19] GLUE, a benchmark for evaluating the performance of NLP models on a diverse set of tasks, provides a comprehensive assessment of LLM capabilities. Kwiatkowski et al.[20] Natural Questions dataset offers a large-scale benchmark for training and evaluating models in answering real-world questions based on Wikipedia articles.

These studies collectively highlight the rapid advancements in LLMs and their growing capabilities in handling complex NLP tasks. Our work builds on these foundations to explore the strategic reasoning potential of LLMs in the "20 Questions" game. Unlike previous studies focusing on general NLP tasks, our approach specifically addresses the challenge of deductive reasoning and

 doi:10.20944/preprints202409.1875.v

strategic questioning. By integrating custom-designed agents with state-of-the-art pre-trained models like Llama-3-8b, we demonstrate improved performance in narrowing down possibilities efficiently, ultimately enhancing the practical application of LLMs in structured reasoning tasks.

3. Methodology

3.1. Text Preprocessing and Formatting

Effective utilization of pre-trained language models (LLMs) requires preprocessing and formatting dialogue text appropriately. For this purpose, we developed a systematic approach to ensure that conversations between the user and the model follow a specific structure, marked by start and end tokens, system prompts, and few-shot examples. This structured format facilitates better understanding and response generation by the model.

Formally, let $D = \{d_1, d_2, \ldots, d_n\}$ represent the set of dialogues, where each d_i consists of a sequence of turns $T = \{t_1, t_2, \ldots, t_m\}$. Each turn t_j can be either a user input u_j or a model response r_j. The preprocessing applies the following transformations:

$$d_i' = \langle \text{start}, T, \text{end} \rangle \tag{1}$$

where $\langle \text{start}, T, \text{end} \rangle$ encapsulates the formatted dialogue. The start and end tokens ensure that the model recognizes the boundaries of each dialogue, while the system prompts and few-shot examples provide context and guidance for generating responses.

The preprocessing and formatting steps involve several key components:

- **Start and End Tokens**: These tokens clearly demarcate the beginning and end of each dialogue. This helps the model to correctly interpret the scope of the conversation and ensures that it processes each dialogue turn within the defined boundaries.
- **System Prompts**: System prompts are used to provide the model with initial context or instructions. For example, a system prompt may include information about the role of the user and the expected type of interaction. This helps in setting the stage for the model's responses and aligning them with the desired outcome of the dialogue.
- **Few-Shot Examples**: Few-shot examples are included to demonstrate the format and type of interactions expected. These examples help the model to understand the context and generate appropriate responses based on the given examples. They act as a reference point for the model, improving its performance by showing it what a typical dialogue might look like.
- **Turn Formatting**: Each turn in the dialogue, whether a user input or a model response, is formatted according to a specific template. This ensures consistency in the dialogue structure, making it easier for the model to parse and generate responses. The turns are interleaved in a way that maintains the flow of conversation.
- **State Management**: The preprocessing step also involves managing the state of the dialogue. This includes tracking the sequence of turns and ensuring that the context is preserved across multiple interactions. Proper state management helps in maintaining coherence and relevance in the model's responses.

For instance, consider a dialogue sequence d_i with turns $T = \{t_1, t_2, t_3\}$, where t_1 and t_3 are user inputs and t_2 is a model response. The formatted dialogue d_i' would be represented as:

$$d_i' = \langle \text{start}, u_1, r_2, u_3, \text{end} \rangle \tag{2}$$

In this example, the start and end tokens frame the dialogue, while the system prompt provides initial instructions, and few-shot examples demonstrate the expected interaction format. Each turn is clearly marked, and the state of the dialogue is managed to ensure coherence.

 doi:10.20944/preprints202409.1875.v

By implementing this systematic approach to text preprocessing and formatting, we ensure that the pre-trained language models are effectively utilized, leading to more accurate and contextually relevant responses in the "20 Questions" game.

3.2. *Agent Methods*

To leverage pre-trained language models, we developed a method to initialize and utilize these models for natural language tasks. This method handles formatting input dialogues, invoking the language model to generate responses, and parsing the results. The agent methods are designed to ensure seamless interaction and effective performance in tasks such as the "20 Questions" game.

The process can be formally defined as follows:

- **Initialization**:
$$\text{Agent} = \{\text{model}, \text{formatter}, \text{state}\} \tag{3}$$
where model is the pre-trained language model, formatter is the text formatting method, and state represents the current state of the dialogue. Initialization involves loading the pre-trained model and setting up the formatter with the appropriate configuration to manage the dialogue state.
- **Dialogue Processing**: For a given input d_i, the formatted input is:
$$d_i' = \text{formatter}(d_i) \tag{4}$$
The agent generates a response r using the model:
$$r = \text{model}(d_i') \tag{5}$$
This step includes formatting the dialogue to ensure consistency and generating responses based on the context and previous turns.
- **Response Parsing**: The agent parses the model's response to extract meaningful information:
$$r' = \text{parse}(r) \tag{6}$$
This involves parsing the response to extract key elements and updating the dialogue state accordingly.

To simulate the roles of questioner and answerer in the "20 Questions" game, specific logic for initializing dialogues and handling responses based on the turn type was implemented.

The 'GemmaQuestionerAgent' class simulates the questioner role by determining the next question based on the state and previous answers. It involves:

- **Initializing Dialogues**: Setting up the initial context and prompts.
- **Question Generation**: Formulating questions to narrow down the possible answers based on the current state and previous responses.
- **State Management**: Continuously updating the state with new information to refine future questions.

The 'GemmaAnswererAgent' class provides yes/no responses based on the secret word. It involves:

- **Initializing Dialogues**: Preparing the agent with the secret word and setting up the initial state.
- **Response Generation**: Providing accurate yes/no answers based on the questions asked.
- **State Management**: Tracking the questions and answers to ensure consistency and accuracy in responses.

 doi:10.20944/preprints202409.1875.v

These methods collectively enable the agents to perform their roles effectively, maintaining the flow of the game and ensuring logical progression towards identifying the secret word. Additionally, error-checking mechanisms are implemented to handle any inconsistencies or invalid responses, thereby enhancing the robustness of the agents' performance in the game.

3.3. Model Loading, Pretraining, and Configuration

To evaluate the performance of different models, various pre-trained models such as llama-3-8b were loaded and pretrained. The models were integrated using predefined prompts and configured for the game environment to ensure optimal performance in the "20 Questions" game.

Let M represent a set of models, where $M = \{m_1, m_2, \ldots, m_k\}$. Each model m_k is loaded, pretrained, and configured as follows:

- **Model Loading**:
$$m_k = \text{load\_model}(\text{model\_id}, \text{config}) \tag{7}$$
where model_id identifies the pre-trained model, and config includes necessary configurations. The configuration file specifies the model architecture, tokenization details, and any pre-training objectives. Loading the model involves setting up the computational environment and ensuring all dependencies are properly installed.
- **Pretraining**: The models are further pretrained using specific datasets to enhance their performance for the 20 Questions game. This involves fine-tuning the model parameters to better handle the nuances of the game:
$$\text{pretrain}(m_k, \text{dataset}) \tag{8}$$
where dataset represents the data used for pretraining. The fine-tuning process includes adjusting the learning rate, optimizing hyperparameters, and running multiple training epochs to improve model accuracy and response quality.
- **Prompt Definition**: Define a prompt template P to guide the model during the pre-training phase:
$$P = \langle \text{system\_prompt}, \text{few\_shot\_examples} \rangle \tag{9}$$
The prompt includes system messages explaining the game rules and few-shot examples demonstrating the expected interaction between questioner and answerer. Crafting effective prompts involves selecting representative examples that cover a wide range of potential scenarios in the game, thereby improving the model's adaptability.
- **Game Environment Initialization**: The game environment was created using Kaggle's environment tools, where agents were configured to play as questioners or answerers. The environment setup involves several key steps:
$$\text{env} = \text{make}(\text{"llm\_20\_questions"}, \text{configuration}) \tag{10}$$
The environment configuration specifies the number of episodes, the time limits for each turn, and other relevant parameters. This setup ensures that the game runs smoothly and that the agents can interact within a controlled, reproducible framework.
- **Agent Configuration**: Each agent a_i is assigned a role (questioner or answerer) and initialized with the appropriate model:
$$a_i = \text{initialize\_agent}(\text{role}, m_k) \tag{11}$$
The initialization involves loading the model weights, setting up the prompt, and configuring the agent to handle dialogue turns. This step includes defining the agent's behavior scripts and ensuring they adhere to the rules of the game.

 doi:10.20944/preprints202409.1875.v

- **Game Execution**: The game runs for a specified number of episodes, and the performance of the agents is recorded:
$$\text{results} = \text{run\_game}(\text{env}, \{a_1, a_2, a_3, a_4\}) \tag{12}$$
During game execution, the agents interact by generating questions and answers, with the goal of identifying the secret word in the fewest turns possible. Performance metrics such as accuracy, number of questions asked, and response time are recorded and analyzed to evaluate the effectiveness of each model.
- **Debugging and Optimization**: Throughout the process, extensive debugging and optimization are conducted to refine the models and improve their performance. This includes adjusting parameters, refining prompts, and iteratively testing the models in various scenarios to ensure robustness and reliability.

By systematically loading, pretraining, and configuring the models, we ensure that they are well-equipped to handle the complexities of the "20 Questions" game, leading to more accurate and efficient performance.

3.4. Advanced Strategy Implementation

To enhance the strategic capabilities of the llama-3-8b model, additional layers of logic were implemented within the agents. These layers included probabilistic reasoning and decision-making algorithms to optimize the question selection process. The goal was to refine the model's ability to efficiently narrow down the potential answers and increase the accuracy of identifying the secret word.

- **Probabilistic Reasoning**: The probability of each potential word being the correct answer was updated after each question based on the responses received. This probabilistic approach allows the model to dynamically adjust its hypotheses as new information becomes available. Bayesian updating was used to formalize this process:
$$P(w|r_i) = \frac{P(r_i|w) \cdot P(w)}{P(r_i)} \tag{13}$$
where $P(w|r_i)$ is the posterior probability of word w given the response r_i, $P(r_i|w)$ is the likelihood of receiving response r_i if w is the correct word, $P(w)$ is the prior probability of w being the correct word, and $P(r_i)$ is the marginal likelihood of receiving response r_i. This approach ensures that the model continually refines its predictions based on the most recent data.
- **Decision-Making Algorithm**: An advanced decision-making algorithm was designed to select the next question that maximizes the expected information gain. This strategy aims to ask questions that are most likely to reduce uncertainty about the correct word. The expected information gain I for a question q is calculated as:
$$I(q) = \sum_{r \in R} P(r|q) \cdot \log \frac{1}{P(r|q)} \tag{14}$$
where $P(r|q)$ is the probability of receiving response r given question q, and R is the set of all possible responses. This calculation helps the model to choose questions that are expected to yield the most informative answers, thereby optimizing the question selection process and improving the efficiency of narrowing down the possible answers.
- **Heuristic Adjustments**: In addition to probabilistic reasoning and decision-making algorithms, heuristic adjustments were made to improve performance. These adjustments include prioritizing certain types of questions based on empirical data, balancing exploration and exploitation strategies, and using domain-specific knowledge to guide the questioning process. Heuristics provide a practical approach to refine the model's decision-making in scenarios where purely probabilistic methods may be insufficient or computationally expensive.

 doi:10.20944/preprints202409.1875.v

By integrating these advanced strategies, the llama-3-8b model's performance in the "20 Questions" game was further enhanced, achieving higher accuracy and efficiency in guessing the secret word. The combination of probabilistic reasoning, decision-making algorithms, heuristic adjustments, and rigorous testing resulted in a model that is adept at handling the complexities of the game with improved strategic capabilities.

3.5. Performance Metrics

The performance of the llama-3-8b integrated agents was evaluated based on their ability to guess the secret word with the fewest questions. The scoring function S is defined as:

$$S = \sum_{i=1}^{n} \frac{1}{q_i} \tag{15}$$

where q_i is the number of questions asked by the model m_i to correctly guess the secret word. Higher scores indicate better performance.

4. Experiments and Results

4.1. Experimental Setup

The experimental setup involved multiple trials to ensure robustness and statistical significance. Each trial included a different set of secret words and random pairings of agents. The following models were evaluated:

Self-defined Agent + Pre-trained Model, Self-defined Agent + Gemma 2b + Pre-trained Model, Self-defined Agent + ChatGPT + Pre-trained Model, Self-defined Agent + Llama-3-8b + Pre-trained Model.

Each model was tested in the same controlled environment, ensuring consistency across trials. The overall performance was averaged across all trials to provide a comprehensive evaluation.

4.2. Results Analysis

The results of the experiments are summarized in Table 1. The performance of each model was evaluated based on the average number of questions required to guess the secret word correctly.

Table 1. Performance Scores of Different Models

Model	Score
Self-defined Agent + Pre-trained Model	651
Self-defined Agent + Gemma 2b + Pre-trained Model	692
Self-defined Agent + ChatGPT + Pre-trained Model	702
Self-defined Agent + Llama-3-8b + Pre-trained Model	741

4.3. Comparative Analysis

The comparative analysis of the experiment results reveals that the Self-defined Agent combined with the Pre-trained Model achieved the highest score, indicating superior efficiency in guessing the secret word. This model required the fewest questions on average, demonstrating its effectiveness in the "20 Questions" game. The success of this model underscores the potential of customizing and fine-tuning pre-trained models to achieve optimal performance in targeted tasks.

5. Conclusions

In conclusion, this study demonstrates the significant advancements in deductive reasoning capabilities of Large Language Models (LLMs) through the implementation of a dual-agent framework

 doi:10.20944/preprints202409.1875.v

in the "20 Questions" game. By employing pre-trained models like llama-3-8b within specialized agents (questioner and answerer), the models showcased enhanced performance in generating strategic questions and accurately interpreting responses to deduce the correct answer efficiently. The llama-3-8b model, particularly, outperformed other models, highlighting its superior strategic reasoning and probabilistic decision-making abilities. These results underscore the potential of LLMs in tasks requiring intricate reasoning and collaborative problem-solving, paving the way for more intelligent and autonomous systems. Future research will aim to further refine these models and expand their applications in other complex reasoning tasks.

References

1. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; others. Language models are few-shot learners. *Advances in neural information processing systems* **2020**, *33*, 1877–1901.
2. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* **2018**.
3. Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I.; others. Language models are unsupervised multitask learners. *OpenAI blog* **2019**, *1*, 9.
4. Sanh, V.; Debut, L.; Chaumond, J.; Wolf, T. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108* **2019**.
5. Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.; Salakhutdinov, R.R.; Le, Q.V. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems* **2019**, *32*.
6. Clark, K.; Luong, M.T.; Le, Q.V.; Manning, C.D. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555* **2020**.
7. Zhang, B.; Xiao, J.; Yan, H.; Yang, L.; Qu, P. Review of NLP Applications in the Field of Text Sentiment Analysis. *Journal of Industrial Engineering and Applied Science* **2024**, *2*, 28–34.
8. Dong, L.; Yang, N.; Wang, W.; Wei, F.; Liu, X.; Wang, Y.; Gao, J.; Zhou, M.; Hon, H.W. Unified language model pre-training for natural language understanding and generation. *Advances in neural information processing systems* **2019**, *32*.
9. Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; Liu, P.J. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research* **2020**, *21*, 1–67.
10. Yan, H.; Xiao, J.; Zhang, B.; Yang, L.; Qu, P. The Application of Natural Language Processing Technology in the Era of Big Data. *Journal of Industrial Engineering and Applied Science* **2024**, *2*, 20–27.
11. Lan, Z.; Chen, M.; Goodman, S.; Gimpel, K.; Sharma, P.; Soricut, R. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942* **2019**.
12. Song, K.; Tan, X.; Qin, T.; Lu, J.; Liu, T.Y. Mass: Masked sequence to sequence pre-training for language generation. *arXiv preprint arXiv:1905.02450* **2019**.
13. Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; Zettlemoyer, L. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461* **2019**.
14. Zhang, J.; Zhao, Y.; Saleh, M.; Liu, P. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. International conference on machine learning. PMLR, 2020, pp. 11328–11339.
15. Zhang, Y.; Sun, S.; Galley, M.; Chen, Y.C.; Brockett, C.; Gao, X.; Gao, J.; Liu, J.; Dolan, B. Dialogpt: Large-scale generative pre-training for conversational response generation. *arXiv preprint arXiv:1911.00536* **2019**.
16. He, P.; Liu, X.; Gao, J.; Chen, W. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654* **2020**.
17. Shoeybi, M.; Patwary, M.; Puri, R.; LeGresley, P.; Casper, J.; Catanzaro, B. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053* **2019**.
18. Janka, H.T.; Wongwathanarat, A.; Kramer, M. Supernova fallback as origin of neutron star spins and spin-kick alignment. *The Astrophysical Journal* **2022**, *926*, 9.

 doi:10.20944/preprints202409.1875.v

19. Wang, A.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; Bowman, S.R. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461* **2018**.
20. Kwiatkowski, T.; Palomaki, J.; Redfield, O.; Collins, M.; Parikh, A.; Alberti, C.; Epstein, D.; Polosukhin, I.; Devlin, J.; Lee, K.; others. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics* **2019**, *7*, 453–466.