university of groningen

faculty of arts

This is the Title of my Thesis
And this is an optional subtitle

Nathalie de Palm

Master thesis
Information Science
Nathalie de Palm
S4295897
June 29, 2025

ABSTRACT

This is what Kenneth K. Landes says about abstracts: "The abstract is of utmost importance for it is read by 10 to 500 times more people than hear or read the entire article. It should not be a mere recital of the subjects covered, replete with such expressions as *is discussed* and *is described*. It should be a condensation and concentration of the essential qualities of the paper.". In other words, it should comprise the goal of the thesis, describe the problems, the method used to solve them, the obtained results, and what consequences the results have (conclusions). Try to fit the abstract on one page (remember: it is an abstract, not an essay).

You may also find the detailed instructions for writing abstracts by the Nature journal helpful.[1]

[1] https://cbs.umn.edu/sites/cbs.umn.edu/files/public/downloads/Annotated_Nature_abstract.pdf

CONTENTS

PREFACE

The preface usually contains information that has nothing to do with the contents of the thesis. It has a personal character, and is written in the first person. It could contain personal circumstances, the context in which the thesis has been written, a description of the assignment, information about the author, and relations to other reports or theses. It is also the place to thank people that helped you, including your well-deserved supervisor!

1 INTRODUCTION

Multiple Choice question (MCQ) is a very popular form of assessment in which respondents are asked to select the best possible answer out of a set of choices (Singh Bhatia et al., 2013; Majumder and Saha, 2014). Furthermore, the Needle In The Haystack approach has become more popular when handling large amounts of contexts by Large Language Models (Nelson et al., 2024; Wang, 2025).

In this thesis, we combine these research areas into the following question:

How does giving Support influence the performance of Large Language Models when answering Multiple Choice Questions?

This question can be split into the following four subquestions, which will be answered in the upcoming chapters:

1. *How does feeding prompting relevant support affect the accuracy of Large Language Models?*
2. *How does feeding irrelevant information affect the accuracy of Large Language Models and what happens when it gets increased with irrelevant information from two and twenty questions?*
3. *How does changing the positioning of the Support affect the accuracy of Large Language Models?*
4. *How does the positioning of the answers affect the accuracy of Large Language Models?*

First, we will present relevant literature and research in Chapter 2. This is followed by the methodology that was used in Chapter 3. This chapter is split into four section, where we describe the data used in the first section, the Large Language Models used in the second, the experiments performed in the third, and how these experiments will be evaluated in the fourth. In Chapter 4, the results will be presented and discussed. Finally, in Chapter 5, the conclusion is presented and directions for future work are discussed.

2 | RELATED WORK

As this chapter covers multiple angles of the study, we split the related work that has been done into three sections. In these sections, we discuss the robustness of Large Language Models, the Needle In The Haystack, and the creation of Multiple Choice Questions.

2.1 LANGUAGE MODEL ROBUSTNESS

In this section, we discuss how well Large Language Models perform and how their robustness is affected after changed circumstances.

2.1.1 Evaluation on Language Models Predictions

Methods on many Multiple Choice Question Asnwering tasks involves pecialized models, extensive per-task engineering, and individualized tuning (Robinson et al., 2022) with different benchmarks on how these models are evaluated (Papineni et al., 2002; Lin, 2004; Zhang* et al., 2020; Sellam et al., 2020; Zheng et al., 2023b; Chen et al., 2021). In this section we discuss whether these methods perform well enough and how certain benchmarks do not give an accurate evaluation of the Large Language Models.

Robinson et al. (2022) argue that Large Language Models fall short when answering Multiple Choice Questions because dominant methods used at the time conflate probabilities of sentences with probabilities of correct answers and call to use what is known as Multiple Choice Prompting. They also show that not all Large Language Models are equally skilled when answering Multiple Choice Questions on their introduced way of prompting and show that the models most capable on this approach can individually approach or beat SOTA on most of the considered tasks using this method.

While previous work used *cloze prompting*, where a question is passed to a Large Language Model and the candidate answers are each independently scored by the models, with the one with the highest probability thus being the answer selected by the model, Robinson et al. (2022) introduce Multiple Choice Prompting. This consists of a question and its symbol-enumerated candidate answers being passed to a Large Language Model in a single prompt, with the prompt being structured so that the Large Language Model should only predict a single token.

The results show that the Multiple Choice Prompts elicit much more accurate responses than the cloze prompts, while also noting that the position of the correct answer has an influence on the Large Language Models' performance, which we will discuss in section 2.1.2.

Tsvilodub et al. (2024) and Robinson et al. (2022) explore the robustness of Large Language Models when answering Multiple Choice Questions. Tsvilodub et al. (2024) determined a Large Language Model's answer choice by using several methods and Zhang et al. (2024) validates the argument made by Robinson et al. (2022) by generating a data set from incorrect predictions on GSM8K (Cobbe et al., 2021) from open source models. The questions from GSM8K are turned into Multiple Choice Questions. The same models are then tested to see if the performance increases on the Multiple Choice Questions compared to the standard ones. Just as Robinson et al. (2022), Zhang et al. (2024) find that while Large Language Models understand the format of Multiple Choice Questions, they do have a bias towards certain options. Furthermore, their results show that as the number of options increases in the Multiple Choice Questions, the correctness distribution gap is reduced.

While Tsvilodub et al. (2024) suggests that the performance of Large Language Models are not robust to choice orders, Zhang et al. (2024) differ in opinion. Tsvilodub et al. (2024) observe that while human accuracy is consistently high, the performance of models and methods varies considerably and that accuracy and likelihood of data are fairly correlated, recommending to not use rating approaches and calling for attention to variability in performance assesment. Zhang et al. (2024) explore how much that is the case on the created data set by constructing different sets of four-way Multiple Choice problems where both the choice of the three distractors and the order of the four options are randomized and repeated, with their results of this experiment showing that the variation on one model's performance is quite small compared with the inner-model difference.

2.1.2 Answer Positioning

As mentioned in section 2.1.1, the position that the answers were in while prompting the Large Language Models has an influence on the performance (Robinson et al., 2022; Zhang et al., 2024). This suggests that models are sensitive to the order of options, which can impact their performance (Wang et al., 2024; Wei et al., 2024; Zheng et al., 2023c). In this section, we discuss how these models are affected and what could be the influence.

Pezeshkpour and Hruschka (2023) investigate the sensitivity of Large Language Models to the order of options, aiming to observe to what extent the Large Language Models exhibit sensitivity to the order of Multiple Choice Questions, what factors contribute to the sensitivity, and how the robustness of these models can be improved. They conduct experiments using different models and analyzed their predictions and whether they changed upon reordering the answers. To quantify this sensitivity, Pezeshkpour and Hruschka (2023) calculate the sensitivity gap, which is the difference between the maximum and minimum performance of a model using an oracle ordering. This is done on zero-shot and few-shot experiments.

The results show that Large Language Models not only exhibit sensitivity to the order of the options, but that this sensitivity also diminishes slightly when demonstrations are integrated into the few-shot setting under specific circumstances. Furthermore, Pezeshkpour and Hruschka (2023) observe that the issue arises from the Large Language Models positional bias, particularly in uncertain circumstances, but despite validation provided, they suggest that deeper comprehension is needed.

In contrast to Pezeshkpour and Hruschka (2023), Zheng et al. (2023a) argue that the cause of selection bias is less from Large Language Models' position bias, but from the models token bias instead, which is measured based on the balance of recalls of different option IDS and the use of the standard deviation of recalls and Gupta et al. (2024) observe whether shuffling the order of the answer label contents, leaving the order of the labels the same affects the measurement of accuracy and they define a new metric that quantifies the effect of chance and suggest the importance to take it into consideration.

The preliminary results of Zheng et al. (2023a) show that selection bias is prevalent across Large Language Models and varies with model families and sizes. Furthermore, selection bias within the same model displays a moderate similarity across different domains and in context examples can reduce but may meanwhile alter selection bias. To observe whether it is token bias or position bias, Zheng et al.

(2023a) experiment by shuffling and removing the option IDs, which resulted in little change for the shuffling of IDs and a notable reduction in selection bias for the removing experiment. Though the selection bias is reduced, Zheng et al. (2023a) suggest that removing the option IDs is not a practical method as it also degrades the model performance.

Gupta et al. (2024)'s results show that models fail to select the correct answer for the questions that it initially selected correctly, with more inconsistencies found in the smaller models. Furthermore, they analyzed the performance drop and discovered that the models struggled most with problem-solving data sets.

Based on the experiments done by Robinson et al. (2022), this thesis will use the same method when prompting the Large Language Models and take into consideration the fact that the position of the answers has an influence on the performance of these models by running parallel experiments similar to what is done by Gupta et al. (2024), which will be further explained in section 3.3.

2.2 NEEDLE IN THE HAYSTACK

The Needle In The Haystack phenomenon in Large Language Models came to be as a manner to help these Language Models retain longer context that was included in the prompt (Nelson et al., 2024; Bianchi et al., 2025). Recently, multiple mechanisms have been suggested to address the issues with long-context modeling (Munkhdalai et al., 2024; Beltagy et al., 2020; Dai et al., 2019; Bulatov et al., 2022). In this section we examine what impacts the performance of Large Language Models and the effect that the length of context has.

Nelson et al. (2024) introduce a method for writing long contexts to an external associative memory, with read/write memory operations that use a prefix or subset of the written text to ease retrieval when reading, show that this method is able to perform log-context retrieval tasks, and demonstrate the feasability of scaling to longer contexts without increasing the GPU memory space footprint. They conducted two experiments to evaluate this methods long-context recall.

With the first experiment, the the context is divided into sentences, which are written in the memory, with the exception of the final prompt which is fed into the decoder, which resulted in a small number of memory slots being used. The second experiment followed (Kamradt, 2023), of which the results showed that baseline models struggle with shorter contexts and larger ones maintained a strong recall.

Wang (2025) demonstrate that memory-based responses significantly impact the performance of Large Language Models and identify that the accuracy drop is related to a shorter thinking process. To do this, they evaluate the models byconcentrating on questions where models can answer correctly when supported by the provided documents, but fail to do so when answering directly without relying on them. The results show that although there is a noticeable difference in accuracy between models, the decrease in accuracy with increasing context lengths is negligible. This, however, does not adequately reflect the impact of context length on the performance of different models, as after filtering a clear decline in accuracy for each model as context increases is observed.

Wang (2025) further investigate the intersection of the models' filtered questions and find that the accuracy at the intersection follows a similar decreasing trend with context length, which leads to exploring three potential factors. They first examine the impact of the placement of supporting documents within the context on accuracy decline. The results of this, however, reveal that the accuracy fluctuates slightly with changes in the placement and exhibits no clear trend, suggesting that the accuracy degradation is independent of the placement. With the next experiment, Wang (2025) vary the position of the second document, increasing the distance between them. This also shows no significant accuracy trend and suggests that the accuracy decline is not related to the distance between the documents. Lastly, Wang (2025) instruct the models to provide their thinking processes tokens, which resulted in a strong correlation with its response accuracy, which suggests that it is possible that longer contexts shorten the thinking process of the models, leading to incomplete or incorrect information retrieval.

Bianchi et al. (2025) examine how gold context size affects long-context performance in Large Language Models by adapting three benchmarks. They observe that models are more sensitive to the placement of smaller gold spans, with the accuracy declining when relevant content appears later in the context window, while larger gold-contexts are more resiliant to positional variation. Furthermore, they find that this problem is amplifies in domain-specific reasoning, that domain-specific tasks exacerbate the challenges relative to general-knowledge tasks, and that this trend persists as the number of distractor documents increases.

The findings by Nelson et al. (2024); Wang (2025); Bianchi et al. (2025) are imperative to this thesis as we feed the Large Language Models support from the Multiple Choice Questions. As the experiment done by Wang (2025), we will examine whether the position of the support affects the performance of the models, though our experiment differs in the type and amount of context given. Furthermore, we also examine the models performance based on the amount of context given, similar to the experiment of Bianchi et al. (2025).

2.3 CREATING MULTIPLE CHOICE QUESTIONS

Developing Multiple Choice Questions has been an occurring research problem (Singh Bhatia et al., 2013) and requires domain expertise (Majumder and Saha, 2014). More specifically, the construction of large, high quality data sets has been one of the main drivers of progress in Natural Language Processing, which lead to multiple ways that Multiple Choice Questions have been created (Singh Bhatia et al., 2013; Majumder and Saha, 2014; Welbl et al., 2017; Manakul et al., 2023).

Singh Bhatia et al. (2013) and Majumder and Saha (2014) use similar, yet slightly different approaches to develop Multiple Choice Questions. Singh Bhatia et al. (2013) develop an automatic Multiple Choice Question generator, which does not require any ontology or WordNet. Instead, their technique makes use of the web, specifically Wikipedia, as the source of information, making it able to easily transfer it to any domain or language.

To select the sentences and generate the questions, Singh Bhatia et al. (2013) work in four phases. First, they collect available Multiple Choice Questions from different sources, which then get turned into sentences. Then, patterns are extracted from these sentences and are used to find similar ones within the Wikipedia domain. These potential new sentences are identified and tagged with the Named Entity Recognition system, where the corresponding entity is selected as the correct answer. Lastly, the sentence is transformed into a question, where the correct answer gets replaced by a proper wh-word. Majumder and Saha (2014) propose a parse-tree matching based approach. Based on the computation of parse tree similarity of a target sentence with a set of reference sentences, they find the most common structures by taking a number of existing Multiple Choice Questions, converting them into into assertive sentences, and then parsing them. Similarly to Singh Bhatia et al.

(2013), they also use the Named Entity Recognition system to check the structure of the created sentences.

As the quality of distractors play an important role in Multiple Choice Questions, Singh Bhatia et al. (2013) develop a framework where all categories are handled by a similar approach, but the attribute set representing the category differs. To generate the distractors, information is extracted from Wikipedia's tabular or structured data or the first sentence of the content. Then, Wikipedia is searched for a list of related candidates from the same category. In case of more difficult sentences, more information is extracted. The used distractors are then randomly picked unless the more difficult sentences need otherwise, then the distractors get picked by preference.

While both Singh Bhatia et al. (2013) and Majumder and Saha (2014) use Wikipedia to extract structures to create sentences, Majumder and Saha (2014) does not turn them into Multiple Choice Questions, thus only providing possible data that can be used to create them. Singh Bhatia et al. (2013) on the other hand continued with their method to create Multiple Choice Questions and tested these with defined parameters. These parameters are then used by five people to evaluate the Multiple Choice Questions, including distractors. There they observed that the system was generating good quality distractors. Furthermore, the model performed well when generating questions, only struggling with ones that are lengthy, compound, or complex sentences.

While the results of Singh Bhatia et al. (2013) are pretty well, the score of the model did suffer when it came to longer or more complex sentences. As we will be using Science-based questions, this method would thus struggle to create the complex sentences that would get used by the Large Language Models. It would be interesting, however, to see how well these methods would apply on the Science-based data to create Multiple Choice Questions to compare with other work.

Welbl et al. (2017) introduce a method for mitigating the difficulties of crowd-sourcing QA data, focusing on Multiple Choice Questions, creating a new data set that translates improvements in more targeted domains instead of broader ones (Bowman et al., 2015; Hermann et al., 2015; Rajpurkar et al., 2016; Hewlett et al., 2016; Bajaj et al., 2016; Manakul et al., 2023).

The method used to create the data set involves a two-step process, where they start by presenting the workers a set of candidate passages and asking to choose one and ask a question about it. Once a passage was chosen, it would not be reused to formulate more questions about it.

After the questions were formulated, another worker takes this question and produces three distractors, aided by a model to predict good distractors. The distractors had to comply with certain characteristics. These characteristics being that distractors had to be dramatically consistent, be consistent with respect to abstract properties, and be consistent with the semantic context of the question. This data set is also set up that each question has the passage separate so that systems can be tested on their own background knowledge when answering the question (Welbl et al., 2017).

Compared to the methods Singh Bhatia et al. (2013) and Majumder and Saha (2014) used, Welbl et al. (2017)'s method was more hands-on. While they extracted information to create questions from the web, Welbl et al. (2017) took a different approach and focused on a specific data type and category. And while Singh Bhatia et al. (2013)'s approach included models to create the sentences, questions, answer, and distractors, only then to be evaluated by humans, Welbl et al. (2017) included human involvement during the entire process, which led to a more consistent and correct data set.

3 METHOD

As mentioned before, we will be exploring what the influence is of different types and amounts of support on the performance of Large Language Models. To explore this influence, we set up multiple experiments with different cases of which the support changes. In this section, we will discuss the data and Large Language Models used for the experiments, the experiments that will be conducted, the evaluation method that will be used for the results.

3.1 DATA

For our experiments, we use the train and evaluation data sets from the SciQ data set. This data set contains 13,679 crowdsourced science exam questions about Physics, Chemistry and Biology, among others , and is split as 11.679, 1000, and 1000 between the train, evaluation, and test data sets Welbl et al. (2017). The questions are in multiple-choice format with 4 answer options each, with the majority of the questions having an additional paragraph with supporting evidence for the correct answer of a varying length. Figure 2 shows the distribution of length for each supporting paragraph. A paragraph is considered small when it has less than 100 tokens, medium when it has between 100 and 500 tokens, and long when it has more than 500 tokens.

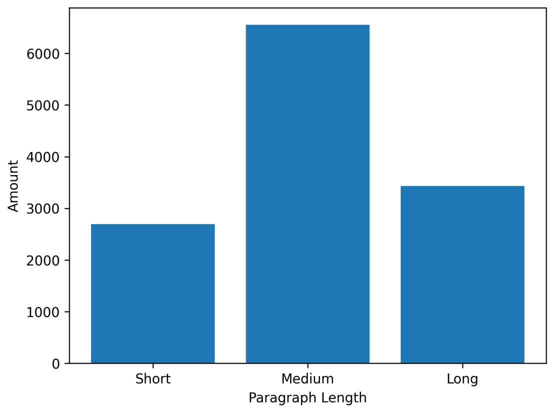

Figure 1: Histogram of the amount of short, medium, and long support paragraphs

This is done by presenting a set of candidate passages to a crowd worker and letting them choose one of the passages and ask a question about it. These passages turn into the support of each question, and include the correct answer to the questions. Then another worker takes the question and answer generated and produces three distractors, aided by a model trained to predict good distractors. This results in a multiple choice science question, a passage, a correct answer, and a set of distractors.

The data sets are directly loaded in the Notebook. Here, we combine the train and evaluation datasets into one dataframe, which will be used for our experiments, no further pre-processing is done on the data. During each experiment, the result of the model is saved in the data frame. After the experiments are done, the dataframe gets saved as a CSV file. As the experiments take a significant amount of time and one of them can only be done after the initial evaluation, the experiments are split into three and thus result in three separate CSV files of the dataframe.

3.2 LANGUAGE MODELS

In this section we will discuss the Language Models that have been used to run the experiments. Furthermore, we will briefly discuss how the models run the experiments.

Model Information

Qwen2 is the new series of Qwen Large Language Models. For Qwen2, the developers released a number of base Language Models and instruction-tuned Language Models ranging from 0.5 to 72 billion parameters, including a Mixture-of-Experts model. Compared to the at the time state-of-the-art opensource Language Models, Qwen2 has generally surpassed most opensource models and demonstrated competitiveness against proprietary models across a series of benchmarks targeting for language understanding, language generation, multilingual capability, coding, mathematics, reasoning, etcetera.

This model series includes decoder Language Models of different model sizes, of which they release the base Language Model and the aligned chat model for each size. During our experiments, we will be using the Qwen2-0.5B-Instruct and Qwen2.5-3B-Instruct models, with the former being a smaller (and earlier) model ad the latter being a bigger (and more recent) model.

The Qwen2.5-3B-Instruct model is a later series of the Qwen Large Language Models Team (2024b). For Qwen2.5, the developers released a number of base Language Models and instruction-tuned Language Models ranging from 0.5 to 72 billion parameters, just as the previously mentioned Qwen2, but brings the following improvements as well:

- It has significantly more knowledge and has greatly improved capabilities in coding and mathematics.
- There are significant improvements in instruction following, generating long texts, understanding structured data, and generating structured outputs.
- It is more resilient to the diversity of system prompts.
- It has long-context Support up to 128K tokens and can generate up to 8K tokens.
- It has Multilingual support for over 29.

Experimentation

After the data is done processing, each model loops over the experiments and rows of the data frame. In this loop there are if statements to determine the type of experiment that is taking place for both the correct answer positioning and amount of support given. Then, each question (with its support), will be given to the model, and an answer will be given by it. This answer is then saved in the data frame. After the experiments are done, the dataframe gets saved as a csv file. The code used and the resulting data sets can be found on the Github[1] page.

[1] https://github.com/NathalieDeP/Masters_Thesis

3.3 EXPERIMENTS

Since Large Language Models are known to be affected by choice order as discussed in section 2.1.2, we run each type of experiment twice, where in the initial run the correct answer is A and in the next run it is D. This is done to explore how much the position of the correct answer has an influence on the performance of the models as has been discussed in section 2.1.2.

Furthermore, all experiments receive an instruction that consists of the following: **"You are a QA system that only answers with a singular letter as an answer"**. The structure of the question is **"What type of organism is commonly used in preparation of foods such as cheese and yogurt? A viruses, B protozoa, C gymnosperms, D mesophilic organisms"**. This structure is used to give more accurate results as discussed in section 2.1.1.

As the models usually answer in the format of **"A viruses"**, we split the first character and thus only save the letter of the answer that the models gave. In case the model does not answer in the way it has been prompted, the default is set that the model will answer the question incorrectly. Though it is important to note that these cases are incredibly rare and are negligible ($< 1\%$) This is done to avoid any issues during the evaluation process.

We run experiments where the model receives no support, with support, with relevant and irrelevant support, with different amounts of irrelevant support, and difficult support. Figure **??** shows the pipeline that the experiments follow.

3.3.1 Initial Baseline

The initial baseline is based on the assumption that someone has a 25% chance of guessing the answer correctly. This means that the models would get 25% of the questions correct and have an accuracy of 0.25. Therefore, with the following experiments, we aim to score higher than this.

3.3.2 No Support

With this experiment, the models only receive the question and answer choices as content. This is done to see how well the models perform without any additional help on the set questions. This is experiment is also considered a second baseline, as this is the first experiment where the models come into play. These results will thus be the initial results of how well the models perform.

3.3.3 Original Question Support

Here, the models not only receive the base information, but also get the corresponding support of the question as the prompt to the model. In this case, the content would look as shown in figure 3. Furthermore, figure 4 shows what the multiple choice question with the answers would look like. This information should give the models a better chance of answering the question correctly.

3.3.4 Additional Irrelevant Support

To see whether the models can differentiate between directly related and unrelated support (Bianchi et al., 2025), more support is given during this experiment. In this case, we give the models support of the question itself, and some not relevant questions. This way, the models have to proccess the multiple supports provided and answer the question as related work has shown in section 2.2. With this, we expect

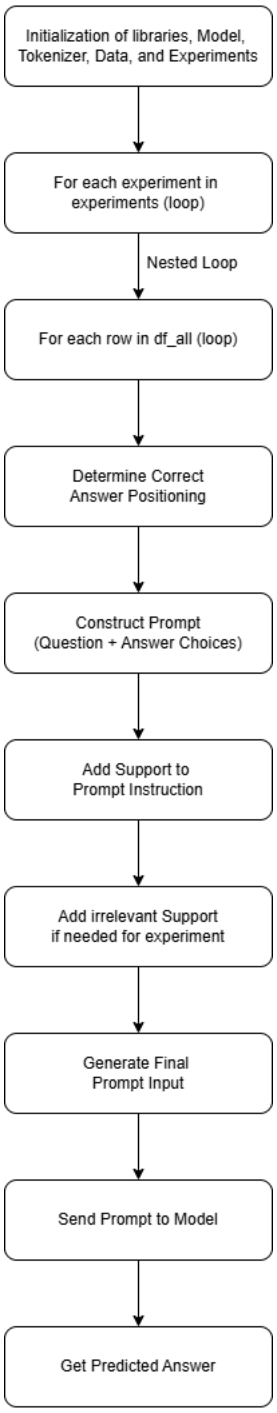
Initialization of libraries, Model, Tokenizer, Data, and Experiments
For each experiment in experiments (loop)
Nested Loop
For each row in df_all (loop)
Determine Correct Answer Positioning
Construct Prompt (Question + Answer Choices)
Add Support to Prompt Instruction
Add irrelevant Support if needed for experiment
Generate Final Prompt Input
Send Prompt to Model
Get Predicted Answer

Figure 2: Flowchart of the model pipeline.

that the models' performance will slightly decrease compared to the experiment mentioned in section 3.3.3.

This experiment was run twice, once where the models received support of the two questions and another where the models received support of the twenty questions. By running the experiment another time with more irrelevant support, we can examine whether the amount of irrelevant support also has an influence on the performance of the models.

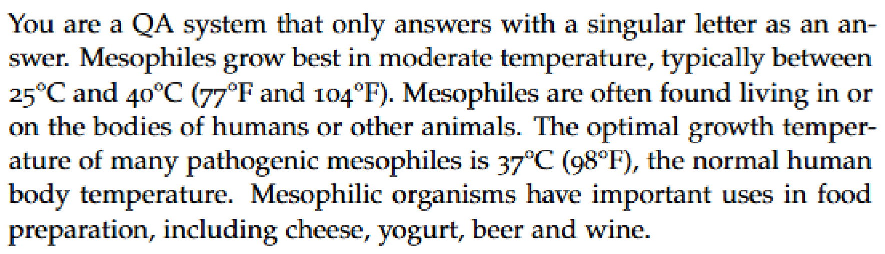
You are a QA system that only answers with a singular letter as an answer. Mesophiles grow best in moderate temperature, typically between 25°C and 40°C (77°F and 104°F). Mesophiles are often found living in or on the bodies of humans or other animals. The optimal growth temperature of many pathogenic mesophiles is 37°C (98°F), the normal human body temperature. Mesophilic organisms have important uses in food preparation, including cheese, yogurt, beer and wine.

Figure 3: Example of the support a question has.

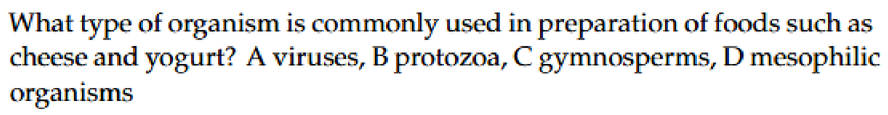
What type of organism is commonly used in preparation of foods such as cheese and yogurt? A viruses, B protozoa, C gymnosperms, D mesophilic organisms

Figure 4: Example of a multiple choice question with its answers.

3.3.5 Support Positioning

Here we explore whether the position has an influence on the models' performance. These experiments thus are a continuation of the previously mentioned *Additional Irrelevant Support*, specifically the experiment where the models receive two cases of irrelevant supports. In the mentioned experiment, the support of the corresponding question is the first one that the models see, followed by the other two. With these experiments, we place the corresponding question support in the middle between the other two supports, and after the other two supports, respectively. With this experiment, we expect there to be minor differences between the results as a model should not, in theory, produce different results though the placement has been slightly altered.

3.3.6 Support Difficulty

With this experiment we explore whether giving the model support consisting of more difficult questions yield into a decrease of the Language Model's performance. This experiment is built on the first three experiments, where we have taken the instances of which the models have answered correctly during the Original Question Support experiment, but answered incorrectly during the experiment where the model also received two cases of irrelevant support. These were gathered and the first fifty have been used. The structure of the given support is exactly the same as in the mentioned experiment, where the models first receive the corresponding question support, and then the other fifty supports.

3.4 EVALUATION

The evaluation of each experiment is done by calculating the accuracy. We use the accuracy to test the model's performance as it would be similar to the models taking a test that consists of multiple choice questions, which then gets graded. Then, these scores are compared to examine and discuss what the influence of different support strategies is on the performance of the Language Models. Table 1 shows an overview of the evaluations made. In addition to these experiments, we will also observe the difference in accuracy based on the positioning of the support.

Experiment	**Compared With**
No Support	Original Question Support
No Support	Additional Irrelevant Support
Original Question Support	Additional Irrelevant Support (2)
Original Question Support	Additional Irrelevant Support (20)
Additional Irrelevant Support (2)	Additional Irrelevant Support (20)
Support Difficulty	Original Question Support
Support Difficulty	Additional Irrelevant Support (2)
Support Difficulty	Additional Irrelevant Support (20)
Correct Answer is "A"	Correct Answer is "D"
0.5B-Instruct	Qwen2.5-3B-Instruct

Table 1: Overview of all experiments that will be compared and discussed.

4 RESULTS AND DISCUSSION

Table 3 shows the accuracy for each experiment.

Experiment	Qwen2-0.5B-Instruct		Qwen2.5-3B-Instruct	
	A	D	A	D
Baseline	0.250	0.250	0.250	0.250
No Support	0.234	0.647	0.907	0.765
Original Question Support	0.386	0.772	0.961	0.954
Additional Irrelevant Support (2 - Front)	0.412	0.780	0.958	0.965
Additional Irrelevant Support (2 - Middle)	0.361	0.766	0.950	0.966
Additional Irrelevant Support (2 - End)	0.352	0.778	0.951	0.968
Additional Irrelevant Support (20)	0.459	0.771	0.944	0.962
Difficult Support	0.731	0.237	0.951	0.921

Table 2: Accuracy of All Experiments

4.1 BASELINE

As mentioned before, the initial baseline was set at an accuracy of 0,25 as this would be the same as the chance of someone guessing an answer correctly. While conducting the experiments, we expected all of them to score higher than this margin. Table 3 shows that Qwen2.5-3B-Instruct scored significantly higher than 0,25. Qwen2-0.5B-Instruct on the other hand, had two experiments where the model scored lower than the baseline, these being the No support with correct answer A and Difficult Support with correct answer D experiments.

When compared to the baseline of the Language Models without any support, we can see that most have an accuracy of at least double, which is mostly expected. The No Support experiment with the correct answer being A by Qwen2-0.5B-Instruct having a lower accuracy than the set baseline is surprising as this would mean that the model performed worse than chance.

4.2 NO SUPPORT

This section will discuss the No Support experiments compared to others. Mainly we will compare it to Original Question Support and Additional Irrelevant Support. We also consider the No Support experiments as the baseline when discussing these results.

4.2.1 Original Question Support

Table 3 shows the effect of giving the Language Models the corresponding support of the question on their performance. Qwen2-0.5B-Instruct saw an increase in accuracy from 0,234 to 0,386 when the correct answer was A and an increase from 0,647 to 0,772 when the correct answer was D. Furthermore, Qwen2.5-3B-Instruct saw an increase from 0,907 to 0,961 when the correct answer was A and an increase from 0,765 to 0,954 when the correct answer was D. This confirms that giving the Language Models the corresponding support increases their performance on the multiple choice questions.

An explanation for this is that the model without any support has to answer the questions based on the data that it was trained upon. In section 3.2 we discussed both Qwen models being trained on (insert data), which is not necessarily fine-tuned on Science-based data. Giving the models support that is directly tied to the question thus should, and has, improved the overall accuracy of both Language Models.

4.2.2 Additional Irrelevant Support

Again, there is an improvement in all the Language Models' scores in table 3 when the models are given support. In the case of Qwen2-0.5B-Instruct, when comparing the No Support baseline to the Additional Irrelevant Support experiments, specifically the ones where it received two cases of irrelevant support, there was an increase from 0,234 to 0,412 and 0,647 to 0,780, with the correct answers being A and D, respectively. When it comes to Qwen2.5-3B-Instruct, there was an increase from 0,907 to 0,958 and 0,765 to 0,965, with the correct answers being A and D.

When comparing the No Support Baseline to the Additional Irrelevant Support experiments where the irrelevant support consisted of the twenty cases of irrelevant support, for Qwen2-0.5B-Instruct we see an increase from 0,234 to 0,459 and from 0,647 to 0,771, with the correct answers being A and D, respectively. When it comes to Qwen2.5-3B-Instruct, we also see an increase. With these experiments, we see an increase from 0,907 to 0,944 and 0,765 to 0,962.

The increase in performance in both experiments using more support is also expected. As mentioned previously, giving the models support that is more directly tied to the topic that the multiple choice question is based on should increase the performance.

4.3 SUPPORT INCREASE

While we have now established that giving support to the model will likely increase its performance, it is imperative to know whether there is a limit to the amount of support a model can handle when answering multiple choice questions. Based on the amount of support the model is now getting, we would expect the performance to decrease as it should have to navigate through and select the corresponding support that matches the possible answers to the multiple choice question (Bianchi et al., 2025).

When examining table 3, we see that Qwen2-0.5B-Instruct has an increase in performance from 0,382 to 0,412 and 0,772 to 0,780 if the model received support from the corresponding- and irrelevant two questions. Qwen2.5-3B-Instruct shows both an increase and a decrease depending on the position of the correct answer. In this model's case, there is a decrease from 0,961 to 0,958 when the correct answer is A and an increase from 0,954 to 0,965 when the correct answer is D.

Increasing the support to include irrelevant support from twenty other questions, leads to an increase from 0,386 to 0,459 when the correct answer is A and a slight decrease from 0,772 to 0,771 when the correct answer is D for Qwen2-0.5B-Instruct. Qwen2.5-3B-Instruct also shows an increase and decrease in performance here. Just as the experiment that included irrelevant support of two other questions, the model show a decrease in its performance from 0,961 to 0,944 when the correct answer is A. The increase for the experiment where the correct answer is D, is from 0,954 to 0,962.

Examining these results, we see that there are three cases where the models have a decrease in its performance when provided with more support, which we expected there to be more of. This does show that while the performance in most experiments increased, there were cases where the models initially answered a question correctly and later answered it incorrectly when they received even more support. It is, however, not as frequent as it would be expected.

4.4 SUPPORT AMOUNT

Table 3 also showed that there are differences between the experiments where the extra support received was from irrelevant two questions and from the irrelevant twenty questions. With these experiments, we also expected the performance of the models to decrease as the models would have to sift through even more support to get to the corresponding one to answer the question correctly (Bianchi et al., 2025).

Qwen2-0.5B-Instruct shows a slight increase from 0,412 to 0,459 and a slight decrease from 0,772 to 0,771 when the correct answers are A and D, respectively. Qwen2.5-3B-Instruct shows a decrease in both cases, with a decrease from 0.958 to 0,944 and 0,965 to 0,962.

Other than the single experiment that showed an increase in its performance, these results are as expected. As mentioned before, since the model has to manage much more support, it has a more difficult time finding and making use of the support that corresponds with the question to give the correct answer.

4.5 SUPPORT DIFFICULTY

With this experiment, we compare the results to the experiments where the model only received the corresponding support, where it also received support from two irrelevant questions, and where it also received support from the twenty irrelevant questions. With this, expect the models' performance to decrease as well.

Table 3 shows that this is the case for all experiments except one. Qwen2-0.5B-Instruct, surprisingly has an increase in performance when the correct answer is A, of which the performance jumps from 0,386 to 0,731 when compared to the experiment of which the model received the support of the corresponding question. When the correct answer is D, there is a significant decrease from 0,772 to 0,237. Qwen2.5-3B-Instruct, on the other hand, shows a decrease in both cases, with a decrease from 0,961 to 0,951 and 0,954 to 0,921.

When compared to the experiment of which the models also received the support of two irrelevant questions, we see the same phenomenon where the performance of Qwen2-0.5B-Instruct has an increase in performance from 0,412 to 0,731 when the correct answer is A. When the correct answer is D, we see a decrease from 0,780 to 0,237. Qwen2.5-3B-Instruct once again shows a decrease in both cases. Compared to the mentioned experiment, there is a decrease from 0,958 to 0,951 and a decrease from 0,965 to 0,921.

Lastly, when compared to the experiment where the models also received the support of the irrelevant twenty questions, Qwen2-0.5B-Instruct again shows an increase. In this case, the increase went from 0,459 to 0,731 when the correct answer was A. There is a decrease when the correct answer is D, with the performance of the model going from 0,771 to 0,237. Qwen2.5-3B-Instruct in this case also has an increase in performance when the correct answer is A, with the performance increasing from 0,944 to 0,951. When the correct answer is D, this model once more shows a decrease in performance going from 0,962 to 0,921.

Comparing these experiments shows a mixed bag of results. On one hand, Qwen2.5-3B-Instruct shows an almost consistent trend of the more difficult support having a negative influence on the models' performance. On the other hand, Qwen2-0.5B-Instruct shows a mostly positive influence on the model's performance, which is not expected. These differences could potentially be chalked up to model and or the positioning of the correct answer or the support difficulty having a random influence on the performance of the models.

Another reasoning for these differences is the combination of Qwen2-0.5B-Instruct being a smaller model and the amount of more difficult irrelevant support given. To explore whether this is the case, we set up an additional experiment where instead of fifty cases of difficult irrelevant support is given, it is lowered to two and twenty similar to as our experiment in section 3.3.4. The results of this experiment can be found in table 3.

Token Length	Qwen2-0.5B-Instruct		Qwen2.5-3B-Instruct	
	A	D	A	D
Original Question Support	0.	0.	0.	0.
Additional Irrelevant Support (2 - Front)	0.	0.	0.	0.
Additional Irrelevant Support (20)	0.	0.	0.	0.
Difficult Support (2 - Front)	0.	0.	0.	0.
Difficult Support (20)	0.	0.	0.	0.
Difficult Support (50)	0.	0.	0.	0.

Table 3: Accuracy of the LLMs based on the types of Support

4.6 SUPPORT POSITIONING

To check whether or not the positioning has an influence on the models performance, we changed the position of the support front to the middle and end on the *Additional Irrelevant Support* experiment that takes two cases irrelevant support. With these experiments we expect the model to perform similarly each time as done in previous work (Wang, 2025).

Table 3 shows that there is a > 0.05 performance decrease when the support of the corresponding question is put at the beginning compared to the middle and end when the correct answer is A for Qwen2-0.5B-Instruct. When the correct answer is D, the performance is more similar, only having a 0.01 performance difference between the support positioning. The results for Qwen2.5-3B-Instruct are similar for all experiments, having a 0.01 performance difference between the support positioning.

While the result for Qwen2-0.5B-Instruct when the correct answer is A is surprising, most of the experiments yield a similar performance as expected. This could potentially be because Qwen2-0.5B-Instruct is a smaller model, and might put importance on the first support that it receives.

4.7 CORRECT ANSWER POSITIONING

The position of the correct answer when giving the model the question, should not yield different results on the experiments. However, as seen in the previous sections, the results differ when the positioning of the correct answer is changed from A to D.

This issue is more noticeable in the results of Qwen2-0.5B-Instruct, which at times gave significantly different results in the same experiment. With all experiments, there is a stark difference between the results where the correct answer was A compared to where the correct answer was D. In most cases, one could even argue that the performance doubles when changing the correct answer from A to D. The results of Qwen2.5-3B-Instruct, while not the same, did yield similar results between the experiments where the positioning of the correct answer was changed. We can see that in this models case, excluding the No Support, there is a < 0.05 difference between the results.

In contrast to what was mentioned in section 4.6, Qwen2-0.5B-Instruct could potentially prioritize the last answer it receives.

4.8 LANGUAGE MODEL COMPARISON

With the expectation that Qwen2.5-3B-Instruct would outperform Qwen2-0.5B-Instruct, we saw that it became clear how much better Qwen2.5-3B-Instruct performed on the set of experiments. While Qwen2-0.5B-Instruct struggles to even reach an accuracy score of 0,80, Qwen2.5-3B-Instruct achieves it with ease.

As mentioned in section 3.2, Qwen2.5-3B-Instruct is a larger model than Qwen2-0.5B-Instruct and has significantly more knowledge, improved instruction following, and is more resiliant to the diversity of system prompts. This base knowledge could have given Qwen2.5-3B-Instruct an advantage over Qwen2-0.5B-Instruct when it came to the experiment where the models did not receive any support. Furthermore, the system prompts is where the model received the support for each experiment, so this specifically could have had a significant influence on its performance.

4.9 SUPPORT LENGTH

An additional experiment was done to explore whether the length of the support of a question has an effect on the accuracy of the Large Language Models (Bianchi et al., 2025). We extracted 1000 Multiple Choice Questions each, where the support was less than 100 tokens, between 100 and 500 tokens, and more than 500 tokens. As with the previous experiments, we promoted the models with these questions and their support of which the results can be found in table 4.

Token Length	Qwen2-0.5B-Instruct		Qwen2.5-3B-Instruct	
	A	D	A	D
TL < 100	0.070	0.710	0.090	0.853
100 < TL < 500	0.034	0.791	0.007	0.985
TL > 500	0.042	0.778	0.005	0.983

Table 4: Accuracy of the LLMs based on the Token Length

The results of table 4 show that the length of the support paragraph affects the performance of the Large Language Models. With both models there is a significant increase in performance when the token length is larger than 100 and the correct answer is "D". Between the token length being 100 < TL < 500 and TL > 500, there is also a 0,013 difference in performance for Qwen2-0.5B-Instruct, while Qwen2.5-3B-Instruct's performance stays almost the exact same. When the Token Length is less than 100, however, the model does not perform as well. This could mean that the shorter paragraphs do not always provide corresponding support for the questions while the longer ones do.

In section 3.1, figure 2 showed that most of the Multiple Choice Questions consist of a support paragraph that is between 100 and 500 tokens. Furthermore, a third of it also consists of paragraph lengths that are over 500 tokens, which yielded similar results as shown in table 4. This suggests that the performance of these Language Models is affected by the length of the support paragraph, which coincided with previous work (Bianchi et al., 2025).

Additionally, the ideal number of tokens a support has for the model to perform optimally is between 110 and 500 tokens.

Both models did not perform well when the correct answer was "A", having an accuracy of < 0.1. While the models not performing well with "A" is not as surprising, the stark difference between the experiments where the correct answer is "A" and "D"is. Furthermore, Qwen2.5-3B-Instruct performs worse than Qwen2-0.5B-Instruct when the Token Length is larger than 100, which is also surprising considering that Qwen2-0.5B-Instruct is a smaller model. In contrast to when the correct answer is "D", the results are also flipped for what type of Token Length results in the best performance. When the correct answer is "A", the Language Models perform the best when the Token Length is smaller than 100.

5 CONCLUSION AND FUTURE WORK

In this thesis, we examined how giving the Language Models Support influences their performance. More specifically, we set out to see how different amounts, giving irrelevant information, Support positioning, and the positioning of the answers affect the performance of these models.

Overall, we found that giving the Language Models Support improved the overall performance of the models, with Qwen2.5-3B-Instruct outperforming Qwen2-0.5B-Instruct. Excluding *Support Difficulty* experiment, as it is separate from the others, all experiments performed better than the set baseline and the *No Support* baseline. When adding irrelevant information to the Support, there are mixed results depending on the model and positioning of the correct answer. The amount of irrelevant information added to the Support shows a decrease in the Language Models' performance in most cases. When taking fifty cases where the Language Models have gotten the answer to the Multiple Choice Question incorrect, there is a mostly decrease in the performance.

We also found that the positioning of the given Support plays a smaller role in the performance of the Language Models, while the positioning of the answers plays a major one. **CONTINUEEE**

Additional experiments were done to observe whether there is a limit on the amount of support a Language Model can process by lowering the amount from fifty to two and twenty cases. This resulted in .... **CONTINUEEE**

Another experiment was ran to see whether the Token Length of the Support paragraphs affected the performance of the Language Model. The results in table 4 showed that Language Models perform the best when the Token Length is between 100 and 500, followed closely by the Token Length being over 500. When looking at the data, we suggest that the performance of Language Models might thus be affected by the Token Lengths, as suggested in previous works (Bianchi et al., 2025). Both Language Models also perform poorly when the correct answer is "A".

Future work...............

more support difficulty tests additional tests positioning correct answer more test on seeing limit or where it degrades

BIBLIOGRAPHY

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. 2016. Ms marco: A human generated machine reading comprehension dataset.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer.

Owen Bianchi, Mathew J. Koretsky, Maya Willey, Chelsea X. Alvarado, Tanay Nayak, Adi Asija, Nicole Kuznetsov, Mike A. Nalls, Faraz Faghri, and Daniel Khashabi. 2025. Lost in the haystack: Smaller needles are more difficult for llms to find.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Aydar Bulatov, Yuri Kuratov, and Mikhail S. Burtsev. 2022. Recurrent memory transformer.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating large language models trained on code.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context.

Vipul Gupta, David Pantoja, Candace Ross, Adina Williams, and Megan Ung. 2024. Changing answer order can decrease mmlu accuracy.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Daniel Hewlett, Alexandre Lacoste, Llion Jones, Illia Polosukhin, Andrew Fandrianto, Jay Han, Matthew Kelcey, and David Berthelot. 2016. Wikireading: A novel large-scale language understanding task over wikipedia.

Greg Kamradt. 2023. Needle in a haystack-pressure testing llms. *Github Repository*, page 28.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Mukta Majumder and Sujan Kumar Saha. 2014. Automatic selection of informative sentences: The sentences that can generate multiple choice questions. *Knowledge Management amp; E-Learning: An International Journal*, page 377–391.

Potsawee Manakul, Adian Liusie, and Mark J. F. Gales. 2023. Mqag: Multiple-choice question answering and generation for assessing information consistency in summarization.

Tsendsuren Munkhdalai, Manaal Faruqui, and Siddharth Gopal. 2024. Leave no context behind: Efficient infinite context transformers with infini-attention.

Elliot Nelson, Georgios Kollias, Payel Das, Subhajit Chaudhury, and Soham Dan. 2024. Needle in the haystack for memory based large language models.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation.

Pouya Pezeshkpour and Estevam Hruschka. 2023. Large language models sensitivity to the order of options in multiple-choice questions.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Joshua Robinson, Christopher Michael Rytting, and David Wingate. 2022. Leveraging large language models for multiple choice question answering.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online. Association for Computational Linguistics.

Arjun Singh Bhatia, Manas Kirti, and Sujan Kumar Saha. 2013. Automatic generation of multiple choice questions using wikipedia. In *Pattern Recognition and Machine Intelligence*, pages 733–738, Berlin, Heidelberg. Springer Berlin Heidelberg.

Qwen Team. 2024a. Qwen2 technical report.

Qwen Team. 2024b. Qwen2.5: A party of foundation models.

Polina Tsvilodub, Hening Wang, Sharon Grosch, and Michael Franke. 2024. Predictions from language models for multiple-choice tasks are not robust under variation of scoring methods.

Haochun Wang, Sendong Zhao, Zewen Qiang, Nuwa Xi, Bing Qin, and Ting Liu. 2024. Llms may perform mcqa by selecting the least incorrect option.

Yidong Wang. 2025. Reasoning on multiple needles in a haystack.

Sheng-Lun Wei, Cheng-Kuang Wu, Hen-Hsen Huang, and Hsin-Hsi Chen. 2024. Unveiling selection biases: Exploring order and token sensitivity in large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 5598–5621, Bangkok, Thailand. Association for Computational Linguistics.

Johannes Welbl, Nelson F. Liu, and Matt Gardner. 2017. Crowdsourcing multiple choice science questions. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 94–106, Copenhagen, Denmark. Association for Computational Linguistics.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Ziyin Zhang, Zhaokun Jiang, Lizhen Xu, Hongkun Hao, and Rui Wang. 2024. Multiple-choice questions are efficient and robust llm evaluators.

Chujie Zheng, Hao Zhou, Fandong Meng, Jie Zhou, and Minlie Huang. 2023a. Large language models are not robust multiple choice selectors.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E Gonzalez, and Ion Stoica. 2023b. Judging llm-as-a-judge with mt-bench and chatbot arena. In *Advances in Neural Information Processing Systems*, volume 36, pages 46595–46623. Curran Associates, Inc.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023c. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.