Different Hallucinations calls for Different Solutions – A Categorisation of LLM Transcription Mistakes

Nemi Pelgrom[1,*], Håkan Grahn[2]

[1]Department of Computer Science and Media Technology, Linnaeus University
[2]Department of Computer Science, Blekinge Institute of Technology

Abstract

This paper presents a contribution to better interpretation of the results we get from GenAI models, more specifically, better interpretation of the mistakes that they make. We have conducted an analysis of 644 (from GPT-4o) + 4858 (from ARIA) mistakes made by two models on a key-value extraction task, and found that they may be categorised into three mutually exclusive groups. These groups are; **i** problems identifying the requested information, **p** problems presenting the correct information, and **s** skewed training data. These categories could be used to indicate which action a user could take to reduce the number of mistakes. Further, we have found a strong correlation between the suggested categories and the Ratcliff/Obershelp pattern recognition score between the generated result and the expected result; all faulty results containing minor mistakes are more than 60% similar to the expected result. Only mistakes based on lack of identifying what was requested had less than 60% similarity to the expected result.

Keywords

Generative AI, LLM, Verification, Document analysis

1. Introduction

While there are many papers detailing the accuracy of large language models' (LLMs') knowledge of particular topics, or forms of reasoning, we are looking closer at the different ways that LLMs are making mistakes, often called "hallucination". While hallucinations are mentioned much in media and AI research, the focus is mainly on avoiding them. One dimensional accuracy measurements are giving some indication of how well models are presenting results at particular tasks. The lack of discussion on what form the mistakes are taking in most of those papers are leaving us with little *insight* into how one might reach higher accuracy results. For example, shall we do better prompting, use better models, change the line of questioning, how one might avoid the wrong results [1, 2], and what might be the edge cases that were not easy to classify as right or wrong [3, 4, 5]?

This experiment had the aim of identifying patterns in the mistakes made by GenAI models with vision capabilities, to better interpret if a model is the right fit for a certain task. This is relevant to the currently emerging paradigm, where there are GenAI models of varying formats

SAIS2025: Swedish AI Society Workshop 2025, 16-17 June 2025, Halmstad, Sweden.
*Corresponding author.
nemi.pelgrom@lnu.se (N. Pelgrom); hakan.grahn@bth.se (H. Grahn)
https://lnu.se/personal/nemi.pelgrom/ (N. Pelgrom); https://grahn.cse.bth.se/ (H. Grahn)
0009-0004-0150-665X (N. Pelgrom); 0000-0001-9947-1088 (H. Grahn)

and aims, and the main question is no longer what model structure is the best one, but rather "What model is the best for my particular task".

We have conducted previous experiments on transcription tasks which had indicated possibilities to systematically categorise the mistakes made [6, 7] (while hallucinations is currently used to refer to a wide range of mistakes made by all GenAI models [8], we find this terminology to be unhelpful, and will mainly use the term mistake to refer to the responses that are faulty). This paper is a continuation of that work, towards systematically identifying the different mistakes that these models make, so that we may better identify the best path forward when we get lower-than-expected results from a GenAI model. It is not currently possible to automatically identify whether the model or the prompt, or something else in the requested task is responsible for unsatisfying results, when such occur. We contribute towards making such an identification possible by here reporting our methods and results for identifying categories of mistakes that are likely to have different sources.

The experiment is using the two multimodal generative models GPT-4o and ARIA to complete a transcription task on a dataset of 3000 images. This provides us with an environment where it is immediately clear if the model is understanding the requested task, and where it is easy to automatically separate the correct answers from the incorrect ones, as opposed to pure text questions which often require some qualitative interpretation. Further, we are requesting transcriptions of both numbers and of natural language strings separately, allowing us to identify possible differences in how they are interpreted by the models.

2. Background and Related work

The development of vision augmented LLM models has gone fast. The first ones were made easily available just a year ago [9]. Despite this, there is such wide interest in using them that there have been several significant developments since then. There are now many options for vision-LLMs [10], including several open-source ones [11, 12]. Open-source models are extra interesting for tasks which require careful data-handling. For example, medical sciences are one of the main driving forces in AI enhanced image interpretation tasks [13], and many of the images gathered for medical studies contain personally identifiable information, making them difficult to handle without breaking confidentiality laws. When it is possible to run all AI tasks locally, such issues disappear. This progress also builds on decades of research in optical character recognition (OCR), which has long aimed to automate transcription tasks and other kinds of information extraction from images. While traditional OCR methods had limitations in dealing with complex layouts or degraded text, vision-capable models have significantly improved the accuracy and versatility of these systems, expanding their usefulness in domains that demand both precision and context understanding [7, 14, 15].

Vision models generally have a three-part structure; one for interpreting the image, one for interpreting text, and one that combines the end result in a fitting way [16, 17]. There are other versions suggested as well [18, 19]. Many of these models are possible to fine-tune to be better at your desired task. There are several different ways this may be done. Regular fine-tuning, where the whole model is re-trained with the added dataset. LoRA (Low Rank Adaption) [20, 21] fine-tuning, where the new training-data is added in between layers rather than making any

changes to the pre-trained parts of the model. And there are also good results from adding the new data to the beginning of the model [22].

There are also other ways to add additional training-data to a pre-trained model. These include Retrieval Augmented Generation (RAG) [23], larger context windows [24], and creating pipelines which ads the relevant information to the model at the right time [25].

A recent addition to the set of GenAI architecture is mixture-of-experts solutions. These aim at minimising the effort used for any particular task, by allowing for separating the models into several parts, where some parts may be ignored when they are deemed irrelevant for the intended task [26]. For our experiment, we are using a model based on this kind of architecture; ARIA [27], the best GenAI model with vision-to-text capabilities *available to run locally* at the beginning of this project. To contrast it, we are using OpenAIs model GPT4-o, the best *available* model with vision-to-text capabilities at the time.

With these great progresses, there are still some shortcomings [28, 29]. Hallucinations have become the standard word to use for mistakes made by Generative AI models [30, 31], and there are both many ways for the models to make mistakes [32, 33, 34], and for us researchers to judge or estimate what should be counted as a mistake [35, 36]. This is the area in which our paper will contribute. There have been some in-depth qualitative studies done on the hallucinations of these models [37, 8], discussing how the content and factuality of text relates to generated text [28, 38, 39], and of course the many that are stating the presence of mistakes simply by presenting the accuracy results of some experiment [40]. There have not been much exploration of what can be learned quantitatively from the mistakes themselves. While simple accuracy estimations allow for fast comparison between several ways to solve the same task, they give little insight into how any of the ways may be improved. The emergence of prompt engineering as a role in itself [41, 42, 43] shows that it is possible to reach significantly different accuracy results depending not only on the information provided to the model of what task it should complete, but also how the information is presented to the model. It would be valuable to be able to identify whether lower accuracy results from using a model is due to the limitations of the model, or to the limitations of the prompt used.

3. Methodology

This section contains all the details of the conducted experiment, the results are presented in the next section.

3.1. Dataset

The dataset consists of 3000 images of real receipts collected from a wide range of purchases in Sweden. The images are scans or photos containing both full view of receipts, and in most cases some additional background such as hands, tables, and knees. Most of the receipts have wrinkles and are not fully flat, which makes them harder to read, and keeps them representative of receipt scans that are uploaded to receipt-reading services.

These images provide a complex task of identifying the correct information requested in the prompt, extracting it correctly, and then presenting it in the way specified by the prompt. If

any one of these steps go wrong, the end result will be faulty. This makes it very impressive that some of these models are able to reach high accuracy results [7, 19].

So from this raw dataset, we created the data used in our experiment; we ran each image through each of the two models, and created separate JSON files containing the response of each reading, for each of the model. The same prompt was used for both models. GPT-4o was accessed through an API call, and ARIA was run locally on a Nvidia A100 GPU. These models were chosen for being the best available, and respectively the best available to run locally, in regards to OCR tasks [44], at the time of our experiments.

This gave us three datasets for our comparisons: 3000 JSON files with GPT-4o transcriptions, 3000 JSON files with ARIA transcriptions, and 3000 JSON files containing the key for each image.

We made python code that compared a selection of keys from each file: date, total amount to pay, VAT amount, company name, and organisation number. We chose these keys to include in this experiment because they are present in most of the images, so they are more representative of a standard receipt than for example ”tips” which are only present on a small minority of the images. When we originally included more of these keys in the comparison, it meant we had to do much more data cleaning. We decided a smaller dataset is preferable to a more complex cleaning step, to ensure the replicability of our results.

3.2. Prompt

The prompt that was used was developed by Fortnox AB, with support from Microsoft. It is extracting most of the information that could be of book-keeping interest from each receipt. While this means that a large amount of keys have been extracted from each image, we choose to do our analysis on only 5 keys, so that we only looked at what is available on the majority of the images. Including keys that exist on fewer of the images would require more time spent on the data cleaning, without increasing the amount of useful mistakes in a proportional way.

We included TotalAmount as a representation of a free-format number. We included date, and VAT as a representations of a number to be extracted in a specific format. OrganisationNumber as a representation of longer number strings (which are known to be difficult for GenAI models [6]), and finally MerchantName to represent text. We included this variety of keys, since it is known that there are patterns of different mistakes made for different kinds of information.

```
Your task is to extract EXACT information from receipts.
Extract the total amount to pay (totalt, totalbelopp, att betala).
Extract the total vat (moms, momsbelopp, total moms).
Extract one or more vat details (moms).
Vat rates can only be 25% | 12% | 6% | 0%. Use 0% if no vat is found.
Extract the currency.
Extract the date (format yyyy-MM-dd). Use "" if not found.
Extract payment method (betalningsmetod).
Extract tip if found.
Extract bonuses or discounts if found.
Extract the supplier name. Use "" if not found.
Extract the supplier organisation number (organisationsnummer / orgnummer). Use "" if
↪ not found.
```

```
Extract all items (articles).

Your final output must satisfy the following typescript schema:

Valid VAT rates
type VatRate = "0%" | "12%" | "25%" | "6%";

Valid payment methods
type PaymentMethod = "" | "card" | "gift card" | "mobile" | "swish";

Valid currency types
type Currency = "SEK" | "DKK" | "NOK" | "EUR" | "USD" | "GBP" | "JPY" | "AUD" | "CHF"
 ↪  | "CAD" | "CNY" | "SGD" | "KRW" | "PLN" | "INR" | "HUF" | "other";

Item interface
interface Item {
    totalAmount: number;
    quantity: number;
    price: number;
    name: string;
}

VAT detail interface
interface VatDetail {
    amount: number;
    rate: VatRate;
}

Main receipt interface
interface Receipt {
    date: string;
    bonus: number;
    vat: number;
    currency: Currency;
    merchantName: string;
    totalAmount: number;
    orgno: string;
    paymentMethod: PaymentMethod;
    tip: number;
    vatDetails: VatDetail[];
    items: Item[];
}

Return the response in a JSON-format that satisfies the above typescript type for
 ↪  Receipt.
Only use the types from above nothing else.
Only output pure json.
Do your best, think step by step.
```

3.3. Data Cleaning

We have two datasets of 15000 comparisons (the amount of files multiplied with the amount of keys we chose for the comparison) in the format of excel sheets.

For each of them we conduct the following steps:

1. We remove all rows where our transcriptions match the keys.
2. We remove all rows where the key has no entry.
3. This left us with 644 (GPT-4o), and 4858 (ARIA) results that did not match their respective keys.

An additional 488 rows were removed during the annotation process. These were comparisons that were not identifiable earlier in the process as irrelevant to our study, but which could have skewed our results if they have been left in the final dataset. All of these rows were ones where we deemed it inaccurate to count the transcription as wrong, despite it not matching the key. Here are some examples of these instances:

- The key stating ”Company Name”, and the model’s result is stating ”Company Name AB”. Both answers are present on the receipt and may therefore be seen as correct.
- The key stating ”179.90” and the transcription stating ”180”, and the image stating that the total amount is 180, but that the card was charged 179.90. Both answers are therefore present on the receipt as a total amount, and may be seen as correct.

Only these false positives were removed.

3.4. Annotation

Once the set of transcriptions was ready, we annotated each row with one of the following:

- **i**: the mistake is not correctly identifying what information is requested
- **s**: the mistake is skewed data, presenting a word or number that is clearly not a misreading of what is present on the image, but instead a word or number from the training data that the model interprets as equivalent
- **p**: when the correct information is identified, and mostly correctly presented, we have annotated with p for presentation, where the mistake is a presentation problem. Wrong spellings are included here. Reading i as 1, and 8 as 3, are included here.

Category **s** could be understood as relating to the category ”confabulations” as introduced in [39]. They are not identical since that concept is focused on the meaning content of text, and we are focusing on the characters, but both concepts aim to find wrong answers that may appear right when there is no key available. No row was left unannotated, and the categories are mutually exclusive so that no row could be annotated into more than one category.

3.5. Ratcliff/Obershelp pattern, Jaccard simularity and Levenshtein distance

Once all rows were annotated, we had noticed a pattern of mistakes in the **p** category being close to the correct replies with only a difference of one or two symbols (e.g. misspellings or

reading a 7 as a 1). This is partly per definition, but indicated a possibility of automatically identifying which category a particular mistake belongs to. Based on this, we ran the document through Python code that added a Ratcliff/Obershelp pattern calculation [45], a Jaccard similarity percentage [46], and a Levenshtein distance [47] for each row. We found that there was a strong correlation between the Ratcliff/Obershelp similarity of a response to its key, and what mistake category it had been annotated as. While it is not surprising that we found correlations, since part of the definition of the category **p** that the result is similar to the correct one, it was unexpected that there was a consistent amount of spelling mistakes, rather than an evenly distributed one.

Levenshtein distance, however, did not show any useful correlation with the categories. While it correctly identifies long strings as being close to the correct answer when there are only spelling mistakes, it does not account for length of the compared strings, which makes fully wrong replies score equally good as slightly wrong ones, when the compared strings are short.

Jaccard similarity had some predictive value; lower similarity correlates with category **i**, and higher with category **p**, however there is no cut-off point between them, and the distribution is broad. Further, Jaccard similarity is based on only the characters that are present, which means that it calculates two strings as equal, if they contain the same characters, even if the characters are not in the same order. This means that it equates 660 with 600, which is not ideal.

Ratcliff/Obershelp pattern matching, also called Gestalt pattern matching, is accounting for the characters present, the lengths of the strings, and the order of the characters in the strings. This makes it a useful algorithm for identifying what kind of mistake a generative model has made, by simply calculating the pattern matching score. And such an estimate may be performed automatically, making it possible to identify the character of the mistakes a model makes for a task without any further manual or otherwise qualitative interpretation of faulty responses from models. When we sorted the rows according to their similarity score, we found a very strong cut-off point, where there are no instances of **i** mistakes above it, and no instances of **p** mistakes below it. This correlation will be shown in detail in the results section. While this is here shown to be true in the context of our experiment, we have not yet tested the possibilities of generalising this to other application areas. It is possible that this correlation holds true for other tasks where it is possible to systematically identify one unique correct answer in relation to a prompt. The category **s** (hallucinations according to the most common usage of it in literature on Chatbots; ”fictional or erroneous information” [8]) had no cut-off point, but the majority of the mistakes in this category had a high Ratcliff/Obershelp similarity score.

4. Results and Discussion

We present two primary findings from this experiment. **First**, we identified a consistent and practical categorization scheme for all transcription mistakes made by GenAI models with vision-to-text capabilities, in scenarios where a structured key is available for comparison. These categories—**i** (identification), **p** (presentation), and **s** (skewed training data)—are broadly applicable across all types of errors observed in our experiments, which involved thousands of receipt images.

As shown in Table 1, both ARIA and GPT-4o models produced mistakes that fell across all three categories, and the distributions of these mistakes are not uniform. ARIA made significantly more total mistakes (4,858 vs. 644), and the majority of its mistakes (4,124) were classified as identification mistakes. GPT-4o, by contrast, exhibited a more balanced distribution, with 403 presentation, and 138 identification mistakes. This uneven distribution suggests that these errors are not random and may reflect inherent tendencies in how each model handles uncertainty or incomplete information. While it is well known that GPT-4o avoid giving blank answers to the degree of producing factual contradictions [48, 49, 50, 24], such tendencies are not necessary for all GenAI models, which might be the explanation for this difference in this case.

Table 1
Number of mistakes identified in each category for each model.

Dataset	Model	Total data-points	Total mistakes	i	p	s
Fortnox 3000	ARIA	15000	4858	4124	605	129
Fortnox 3000	GPT-4o	15000	644	138	403	103

This disparity highlights that these errors are not random; rather, they reflect consistent patterns tied to model behaviour. Table 2 quantifies the difference, using GPT-4o as a baseline: ARIA made 2,888% more identification mistakes, 50% more presentation mistakes, and 25% more skewed data mistakes. The striking increase in **i** mistakes from ARIA suggests a conservative extraction approach—opting to leave fields blank when uncertain.

Table 2
Variance in the number of mistakes in each category between different models.

Category	GPT-4o	ARIA	Variance (%)
i	138	4124	+2888%
p	403	605	+50%
s	103	129	+25%

Field-level analysis provides more insight. Table 3 breaks down the number and type of mistakes by data key. ARIA's highest concentration of identification errors occurred with the Merchant Name and Organisation Number fields—1,619 and 1,847 respectively. For these, the model frequently failed to return any value, even when the information was clearly visibly present in the image. Table 4 supports this: ARIA's empty response rate was 91% for Merchant Name and 98% for Organisation Number, versus 40% and 9% for GPT-4o, respectively. These high omission rates indicate ARIA's tendency to skip uncertain fields entirely, possibly due to a narrower confidence threshold or the model being overwhelmed by the amount of information that was requested of it [51, 52].

The significant difference in identifying Organisation Numbers may also be explained by the structural properties of receipts and the training-data of the models. For example, Swedish Organisation Numbers are often present but not explicitly labelled as such in the receipts in our dataset. Models not specifically trained to recognize these patterns, such as ARIA, are more prone to fail on identification, especially if they rely on keyword cues. GPT-4o, by contrast,

appears more likely to attempt a response even when uncertain, which explains its higher relative rate of presentation mistakes, where the correct value is approximated but not matched exactly to the key.

Table 3
Distribution of mistakes across keys and models. All categories of mistakes were found in all of the subcategories that were analysed.

Category	GPT-4o			ARIA		
	i	p	s	i	p	s
Date	6	75	2	10	271	110
VAT	35	45	4	522	81	2
Merchant Name	48	37	2	1619	122	9
Total Amount	25	13	1	126	92	3
Org No	24	233	94	1847	39	5

Table 4
How many of all mistakes were empty responses, and how many were Category **s**.

Category	Empty Values				Category **s**			
	GPT-4o		ARIA		GPT-4o		ARIA	
	Amount	%	Amount	%	Amount	%	Amount	%
Date	6	7%	10	3%	2	2%	110	28%
VAT	10	12%	113	19%	4	5%	2	0.3%
Merchant Name	35	40%	1597	91%	2	2%	9	1%
Total Amount	5	13%	14	6%	1	3%	3	1%
Org No	17	9%	1845	98%	94	27%	5	0.3%

Additionally, the Merchant Name field revealed another nuance. According to Table 5, this field was the most affected by cleaning in both model outputs. GPT-4o had a 21% retention rate post-cleaning, while ARIA retained 86%. This difference is partly explained by ARIAs tendency to give blank answers. But it brings our attention to another issue with automatic accuracy estimations; instances where there are multiple correct answers. All of the results that were removed in this data-cleaning had the issue of having multiple accurate results. They could have been included in our experiment if the keys were of a format that allowed several answers to be understood as correct.

The second major finding is the strong correlation between Ratcliff/Obershelp similarity scores and error category. Table 6 shows that all identification mistakes **i** had similarity scores below 60%, while presentation mistakes **p** had scores of 60% or higher. Skewed data mistakes **s** showed a mixed pattern: most were above 60%, but a small percentage (10% for GPT-4o, 5% for ARIA) fell between 35–60%. This suggests that Ratcliff/Obershelp similarity can be used as a heuristic for error classification, high similarity likely indicates **p** or **s** errors, while low similarity indicates **i** errors. The exact threshold may vary depending on text length and structure of the information that is extracted, but the trend holds across both models and all keys that we tested.

Further analysis of skewed data errors reveals model-specific behaviour. ARIA exhibits a high rate of s errors in the Date field (28%), suggesting a tendency to hallucinate plausible

Table 5
Data loss during the cleaning process for each key, for each model.

Category	Original	Cleaned	% of Original
GPT-4o			
Date	122	83	68%
VAT	95	84	88%
Merchant Name	408	87	21%
Total Amount	89	39	44%
Organisation No	419	351	84%
ARIA			
Date	423	391	92%
VAT	674	604	90%
Merchant Name	2054	1760	86%
Total Amount	257	221	86%
Organisation No	1891	1891	100%

Table 6
Distribution of items by Ratcliff/Obershelp Similarity threshold (60% cutoff).

Category	$< 60\%$	$< 60\%$ (%)	$\geq 60\%$	$\geq 60\%$ (%)	Number of
GPT-4o					
i	138	100%	0	0%	138
p	0	0%	403	100%	403
s	18	17%	85	83%	119
ARIA					
i	4124	100%	0	0%	4124
p	0	0%	605	100%	605
s	7	5%	122	95%	129

but incorrect values when uncertain. In contrast, GPT-4o shows more skewed mistakes in the VAT field (5%), but overall keeps skewed data rates low across most keys (Table 4). These patterns reinforce the idea that ARIA is conservative—risking omissions—while GPT-4o aims for completeness, sometimes at the expense of accuracy.

By combining our categorization with Ratcliff/Obershelp similarity metrics, we provide a replicable framework for analysing transcription performance.

5. Conclusion

We made an experiment focused on identifying categories of mistakes made by GenAI models, with the intention of finding patterns that could help clarify what models may or may not be useful for particular tasks, further than a simple accuracy estimate does. We found two useful patterns; that there are three systematically identifiable categories of mistakes that reoccur across different models, and that there is a simple way to automatically sort mistakes into at least two of these categories. This paper reports the details of how the experiment was conducted, and we suggest that further research should be done on the possibilities of generalising these

findings, and on identifying the reasons for why these mistakes occur.

Acknowledgments

We want to thank Fortnox AB for providing the dataset, prompt, and financial support for this project, and we also thank the reviewers for their helpful contributions.

Declaration on Generative AI

During the preparation of this work, the author(s) used GPT-4o for drafting the structure of the paper. After using these tool(s)/service(s), the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

References

[1] S. Tonmoy, S. Zaman, V. Jain, A. Rani, V. Rawte, A. Chadha, A. Das, A comprehensive survey of hallucination mitigation techniques in large language models, arXiv preprint arXiv:2401.01313 6 (2024).

[2] M. Peychev, M. Müller, M. Fischer, M. Vechev, Automated classification of model errors on imagenet, Advances in Neural Information Processing Systems 36 (2023) 36826–36885.

[3] C. Thomson, E. Reiter, B. Sundararajan, Evaluating factual accuracy in complex data-to-text, Comput. Speech Lang. 80 (2023). URL: https://doi.org/10.1016/j.csl.2023.101482. doi:10.1016/j.csl.2023.101482.

[4] A. Dutta, S. Krishnan, N. Kwatra, R. Ramjee, Accuracy is not all you need, arXiv preprint arXiv:2407.09141 (2024).

[5] B. Goodrich, V. Rao, P. J. Liu, M. Saleh, Assessing the factual accuracy of generated text, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '19, ACM, 2019, p. 166–175. URL: http://dx.doi.org/10.1145/3292500.3330955. doi:10.1145/3292500.3330955.

[6] N. Pelgrom, J. Hangelbäck, M. Ericsson, J. Nordqvist, H. Grahn, Hallucinations and training-data bias: Results from two number transcription experiments using gpt models, in: International Conference on Computational Science and Computational Intelligence, Springer, 2025, pp. 59–69.

[7] N. Pelgrom, M. Ericsson, H. Grahn, J. Nordqvist, J. Hagelbäck, Chatgpt as a combined ocr and key-value extractor, in: 2025 IEEE 10th International Conference on Big Data Analytics (ICBDA), 2025. Accepted, to appear.

[8] N. Maleki, B. Padmanabhan, K. Dutta, Ai hallucinations: a misnomer worth clarifying, in: 2024 IEEE conference on artificial intelligence (CAI), IEEE, 2024, pp. 133–138.

[9] OpenAI, Gpt-4v(ision) system card, https://openai.com/index/gpt-4v-system-card/, 2024. Accessed: 2024-10-14.

[10] W. Wang, Z. Chen, X. Chen, J. Wu, X. Zhu, G. Zeng, P. Luo, T. Lu, J. Zhou, Y. Qiao, et al., Visionllm: Large language model is also an open-ended decoder for vision-centric tasks, Advances in Neural Information Processing Systems 36 (2024).

[11] H. Liu, C. Li, Q. Wu, Y. J. Lee, Visual instruction tuning, Advances in neural information processing systems 36 (2024).
[12] Z. Peng, W. Wang, L. Dong, Y. Hao, S. Huang, S. Ma, F. Wei, Kosmos-2: Grounding multimodal large language models to the world, 2023. URL: https://arxiv.org/abs/2306.14824. arXiv:2306.14824.
[13] S. Zhang, D. Metaxas, On the challenges and perspectives of foundation models for medical image analysis, Medical image analysis 91 (2024) 102996.
[14] S. Kim, J. Baudru, W. Ryckbosch, H. Bersini, V. Ginis, Early evidence of how llms outperform traditional systems on ocr/htr tasks for historical records, 2025. URL: https://arxiv.org/abs/2501.11623. arXiv:2501.11623.
[15] S. Chen, X. Guo, Y. Li, T. Zhang, M. Lin, D. Kuang, Y. Zhang, L. Ming, F. Zhang, Y. Wang, J. Xu, Z. Zhou, W. Chen, Ocean-ocr: Towards general ocr application via a vision-language model, 2025. URL: https://arxiv.org/abs/2501.15558. arXiv:2501.15558.
[16] L. H. Li, M. Yatskar, D. Yin, C.-J. Hsieh, K.-W. Chang, Visualbert: A simple and performant baseline for vision and language, arXiv preprint arXiv:1908.03557 (2019).
[17] A. Masry, J. A. Rodriguez, T. Zhang, S. Wang, C. Wang, A. Feizi, A. K. Suresh, A. Puri, X. Jian, P.-A. Noël, S. T. Madhusudhan, M. Pedersoli, B. Liu, N. Chapados, Y. Bengio, E. Hoque, C. Pal, I. H. Laradji, D. Vazquez, P. Taslakian, S. Gella, S. Rajeswar, Alignvlm: Bridging vision and language latent spaces for multimodal understanding, 2025. URL: https://arxiv.org/abs/2502.01341. arXiv:2502.01341.
[18] W. Kwon, Z. Li, S. Zhuang, Y. Sheng, L. Zheng, C. H. Yu, J. E. Gonzalez, H. Zhang, I. Stoica, Efficient memory management for large language model serving with pagedattention, 2023. URL: https://arxiv.org/abs/2309.06180. arXiv:2309.06180.
[19] M. Faysse, H. Sibille, T. Wu, B. Omrani, G. Viaud, C. Hudelot, P. Colombo, Colpali: Efficient document retrieval with vision language models, 2024. URL: https://arxiv.org/abs/2407.01449. arXiv:2407.01449.
[20] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, et al., Lora: Low-rank adaptation of large language models., ICLR 1 (2022) 3.
[21] Y. Mao, Y. Ge, Y. Fan, W. Xu, Y. Mi, Z. Hu, Y. Gao, A survey on lora of large language models, Frontiers of Computer Science 19 (2025) 197605.
[22] R. Zhang, J. Han, C. Liu, P. Gao, A. Zhou, X. Hu, S. Yan, P. Lu, H. Li, Y. Qiao, Llama-adapter: Efficient fine-tuning of language models with zero-init attention, 2024. URL: https://arxiv.org/abs/2303.16199. arXiv:2303.16199.
[23] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, et al., Retrieval-augmented generation for knowledge-intensive nlp tasks, Advances in Neural Information Processing Systems 33 (2020) 9459–9474.
[24] S. Chen, S. Wong, L. Chen, Y. Tian, Extending context window of large language models via positional interpolation, arXiv preprint arXiv:2306.15595 (2023).
[25] A. Scius-Bertrand, A. Fakhari, L. Vögtlin, D. R. Cabral, A. Fischer, Are layout analysis and ocr still useful for document information extraction using foundation models?, in: International Conference on Document Analysis and Recognition, Springer, 2024, pp. 175–191.
[26] N. Ding, Y. Qin, G. Yang, F. Wei, Z. Yang, Y. Su, S. Hu, Y. Chen, C.-M. Chan, W. Chen, et al., Parameter-efficient fine-tuning of large-scale pre-trained language models, Nature

Machine Intelligence 5 (2023) 220–235.
[27] D. Li, Y. Liu, H. Wu, Y. Wang, Z. Shen, B. Qu, X. Niu, G. Wang, B. Chen, J. Li, Aria: An open multimodal native mixture-of-experts model, arXiv preprint arXiv:2410.05993 (2024).
[28] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, P. Fung, Survey of hallucination in natural language generation, ACM computing surveys 55 (2023) 1–38.
[29] P. P. Liang, A. Zadeh, L.-P. Morency, Foundations & trends in multimodal machine learning: Principles, challenges, and open questions, ACM Computing Surveys 56 (2024) 1–42.
[30] N. Maleki, B. Padmanabhan, K. Dutta, Ai hallucinations: A misnomer worth clarifying, in: 2024 IEEE Conference on Artificial Intelligence (CAI), 2024, pp. 133–138. doi:10.1109/CAI59869.2024.00033.
[31] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin, T. Liu, A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions, ACM Trans. Inf. Syst. 43 (2025). URL: https://doi.org/10.1145/3703155. doi:10.1145/3703155.
[32] P. Koehn, R. Knowles, Six challenges for neural machine translation, ACL 2017 (2017) 28.
[33] S. A. Athaluri, S. V. Manthena, V. K. M. Kesapragada, V. Yarlagadda, T. Dave, R. T. S. Duddumpudi, Exploring the boundaries of reality: investigating the phenomenon of artificial intelligence hallucination in scientific writing through chatgpt references, Cureus 15 (2023).
[34] J. Gravel, M. D'Amours-Gravel, E. Osmanlliu, Learning to fake it: limited responses and fabricated references provided by chatgpt for medical questions, Mayo Clinic Proceedings: Digital Health 1 (2023) 226–234.
[35] S. Banerjee, A. Agarwal, S. Singla, Llms will always hallucinate, and we need to live with this, arXiv e-prints (2024) arXiv–2409.
[36] S. Barros, I think, therefore i hallucinate: Minds, machines, and the art of being wrong, arXiv e-prints (2025) arXiv–2503.
[37] J. Li, J. Chen, R. Ren, X. Cheng, W. X. Zhao, J.-Y. Nie, J.-R. Wen, The dawn after the dark: An empirical study on factuality hallucination in large language models, in: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2024, pp. 10879–10899.
[38] Y. Bang, Z. Ji, A. Schelten, A. Hartshorn, T. Fowler, C. Zhang, N. Cancedda, P. Fung, Hallulens: Llm hallucination benchmark, arXiv preprint arXiv:2504.17550 (2025).
[39] S. Farquhar, J. Kossen, L. Kuhn, Y. Gal, Detecting hallucinations in large language models using semantic entropy, Nature 630 (2024) 625–630.
[40] F. Liu, Y. Liu, L. Shi, H. Huang, R. Wang, Z. Yang, L. Zhang, Z. Li, Y. Ma, Exploring and evaluating hallucinations in llm-powered code generation, arXiv preprint arXiv:2404.00971 (2024).
[41] L. Giray, Prompt engineering with chatgpt: a guide for academic writers, Annals of biomedical engineering 51 (2023) 2629–2633.
[42] J. White, Q. Fu, S. Hays, M. Sandborn, C. Olea, H. Gilbert, A. Elnashar, J. Spencer-Smith, D. C. Schmidt, A prompt pattern catalog to enhance prompt engineering with chatgpt, arXiv preprint arXiv:2302.11382 (2023).
[43] P. Sahoo, A. K. Singh, S. Saha, V. Jain, S. Mondal, A. Chadha, A systematic survey of prompt engineering in large language models: Techniques and applications, arXiv preprint

arXiv:2402.07927 (2024).
[44] H. Duan, J. Yang, Y. Qiao, X. Fang, L. Chen, Y. Liu, X. Dong, Y. Zang, P. Zhang, J. Wang, et al., Vlmevalkit: An open-source toolkit for evaluating large multi-modality models, in: Proceedings of the 32nd ACM International Conference on Multimedia, 2024, pp. 11198–11201.
[45] J. W. Ratcliff, D. E. Metzener, et al., Pattern matching: The gestalt approach, Dr. Dobb's Journal 13 (1988) 46.
[46] P. Jaccard, Nouvelles recherches sur la distribution florale, Bull. Soc. Vaud. Sci. Nat. 44 (1908) 223–270.
[47] V. I. Levenshtein, et al., Binary codes capable of correcting deletions, insertions, and reversals, in: Soviet physics doklady, volume 10, Soviet Union, 1966, pp. 707–710.
[48] A. Payandeh, D. Pluth, J. Hosier, X. Xiao, V. K. Gurbani, How susceptible are llms to logical fallacies?, 2023. URL: https://arxiv.org/abs/2308.09853. arXiv:2308.09853.
[49] R. Zhu, Z. Ma, J. Wu, J. Gao, J. Wang, D. Lin, C. He, Utilize the flow before stepping into the same river twice: Certainty represented knowledge flow for refusal-aware instruction tuning, 2024. URL: https://arxiv.org/abs/2410.06913. arXiv:2410.06913.
[50] X. Zhao, J. Yu, Z. Liu, J. Wang, D. Li, Y. Chen, B. Hu, M. Zhang, Medico: Towards hallucination detection and correction with multi-source evidence fusion, 2024. URL: https://arxiv.org/abs/2410.10408. arXiv:2410.10408.
[51] R. M. French, Catastrophic forgetting in connectionist networks, Trends in cognitive sciences 3 (1999) 128–135.
[52] Y. Luo, Z. Yang, F. Meng, Y. Li, J. Zhou, Y. Zhang, An empirical study of catastrophic forgetting in large language models during continual fine-tuning, 2025. URL: https://arxiv.org/abs/2308.08747. arXiv:2308.08747.