Artificial Intelligence and Law
https://doi.org/10.1007/s10506-024-09411-z

ORIGINAL RESEARCH

Applicability of large language models and generative models for legal case judgement summarization

Aniket Deroy[1] · Kripabandhu Ghosh[2] · Saptarshi Ghosh[1]

Accepted: 3 July 2024

Abstract

Automatic summarization of legal case judgements, which are known to be long and complex, has traditionally been tried via extractive summarization models. In recent years, generative models including abstractive summarization models and Large language models (LLMs) have gained huge popularity. In this paper, we explore the applicability of such models for legal case judgement summarization. We applied various domain-specific abstractive summarization models and general-domain LLMs as well as extractive summarization models over two sets of legal case judgements – from the United Kingdom (UK) Supreme Court and the Indian Supreme Court – and evaluated the quality of the generated summaries. We also perform experiments on a third dataset of legal documents of a different type – Government reports from the United States. Results show that abstractive summarization models and LLMs generally perform better than the extractive methods as per traditional metrics for evaluating summary quality. However, detailed investigation shows the presence of inconsistencies and hallucinations in the outputs of the generative models, and we explore ways to reduce the hallucinations and inconsistencies in the summaries. Overall, the investigation suggests that further improvements are needed to enhance the reliability of abstractive models and LLMs for legal case judgement summarization. At present, a human-in-the-loop technique is more suitable for performing manual checks to identify inconsistencies in the generated summaries.

Keywords Legal judgement summarization · Abstractive summarization · Large language models · Prompting · Hallucinations

✉ Aniket Deroy
roydanik18@kgpian.iitkgp.ac.in

Kripabandhu Ghosh
kripa.ghosh@gmail.com

Saptarshi Ghosh
saptarshi.ghosh@gmail.com

[1] Computer Science and Engineering, IIT Kharagpur, Kharagpur, West Bengal 721302, India

[2] Computational and Data Sciences, IISER Kolkata, Kolkata, West Bengal 741246, India

1 Introduction

Summarizing legal case judgements is a practical and challenging task, given the complicated and lengthy nature of the legal judgements. Typically, legal judgements are manually summarized by legal practitioners; in fact, many legal information systems offer case summaries or headnotes written by legal practitioners. To reduce the pressure on humans, there have been several attempts towards automatic summarization of legal judgements. Traditionally, extractive summarization methods – which extract important sentences/parts from the input document – have been used to summarize legal case judgements (Deroy et al. 2023; Bhattacharya et al. 2021; Polsley et al. 2016; Zhong et al. 2019; Deroy et al. 2024). But recently there has been increasing interest in using *abstractive* summarization models because these models are considered to generate more 'natural' and 'coherent' summaries. Hence there are some recent works that train abstractive summarization models on (legal document, summary) pairs (Shukla et al. 2022; Feijo and Moreira 2023). Legal domain-specific versions of pre-trained abstractive summarization models (e.g., Longformer (Beltagy et al. 2020), Pegasus (Zhang et al. 2020)) have also been released, such as Legal-LED[1] and Legal-Pegasus.[2]

Additionally, the recent times have seen the advent of general-domain Large language models (LLMs) which can be used for a wide variety of NLP tasks including summarization, translation, question answering, and so on Teubner et al. (2023). These LLMs have the advantage that they can be applied for summarization of any document, including legal documents, without further training. In fact, LLMs have been used recently for news document summarization (Zhang et al. 2023). So the question arises how well the pre-trained abstractive summarization models and LLMs (like ChatGPT and Davinci) can perform in the task of legal case judgement summarization. We attempt to answer this question in this paper.

In this work, we apply various legal domain-specific abstractive summarization models (such as Legal-LED and Legal-Pegasus) as well as general-domain LLMs on two datasets of legal case judgements and their summaries – (1) the UK-Abs dataset (a dataset of United Kingdom case judgements), and (2) IN-Abs dataset (a dataset of Indian case judgements). For comparison, we also apply extractive summarization models on the same datasets. We also perform experiments on a third dataset from the legal domain – (3) the GOVREPORT dataset (consisting of long reports published by the U.S. Government Accountability Office). We compute a large number of summary quality metrics for all the models, including traditional metrics such as ROUGE (Lin 2004), METEOR (Banerjee and Lavie 2005) and BERTScore (Zhang et al. 2019) (that match model-generated summaries with gold standard summaries) as well as metrics for quantifying the consistency of summaries with respect to the original document such as SummaC (Laban et al. 2022), NEPrec (Deroy et al. 2023), and Numprec (Deroy et al. 2023).

[1] https://huggingface.co/nsi319/legal-led-base-16384.

[2] https://huggingface.co/nsi319/legal-pegasus.

Our results show that abstractive summarization models and LLMs usually perform better than the extractive summarization methods in terms of the conventional summary quality metrics (see Sect. 6). However, summaries generated by the abstractive models and LLMs often contain inconsistencies and hallucinations (Ji et al. 2023). We manually analyze the summaries generated by these models to report examples of such hallucinations and inconsistencies in the summaries (see Sect. 7). We also explore ways of reducing such inconsistencies in the generated summaries, by domain-specific fine-tuning of abstractive summarization models, suitable prompting of LLMs and a semantic similarity-based approach. We also provide a human evaluation (by Law students) of summaries generated by some of the models. Our contributions in this work are as follows:

(1) We experiment with general-domain LLMs (and several different prompts), legal domain-specific abstractive summarization models, and extractive summarization methods for legal case judgement summarization. To our knowledge, this is the first systematic comparison of all these families of summarization models for this practically important problem.
(2) We show several examples of hallucinations and inconsistencies in the summaries generated by the abstractive summarization models and LLMs (see Sect. 7).
(3) We explore several different ways to reduce hallucinations and inconsistencies in legal case summaries (see Sect. 8). In particular, we develop a simple semantic similarity-based approach that effectively reduces hallucinations while also improving the quality of the generated summaries.
(4) We perform a human evaluation of the summaries generated by the best performing summarization models based on several human evaluation metrics (see Sect. 10).

2 Related work

Legal case Judgement and Long Document Summarization Several extractive and abstractive summarization methods have been used for summarizing legal case judgments (Bhattacharya et al. 2019; Shukla et al. 2022; Feijo and Moreira 2023; Nigam et al. 2023; Nigam and Deroy 2023). In particular, recent works have applied abstractive summarization models like Legal-Pegasus, Legal-LED, and BART to Indian and UK court judgements (Shukla et al. 2022). To this end, supervised extractive and abstractive models have been trained / fine-tuned on (legal judgement, summary) pairs.

Unsupervised extractive models have also been developed for this purpose, such as CaseSummarizer (Polsley et al. 2016), DelSumm (Bhattacharya et al. 2021), and so on. Graphical models have also been used for the purpose of legal case judgement summarization (Saravanan et al. 2006).

LLMs for summarization and other tasks in different domains With the recent popularity of LLMs, they have been applied for text summarization as well. A work on news document summarization (Zhang et al. 2023) tried to investigate the performance of multiple LLMs based on the type of prompts and LLM size. The study found that prompting techniques play a critical role in determining the quality of the summaries. A recent work Ahmed and Devanbu (2023) on code summarization using GPT used zero-shot and few-shot capabilities of LLMs for project-specific code summarization. A work on book summarization (Chang et al. 2023) tried to develop novel methods to summarize long books using foundational models like GPT-4 and Turbo-GPT-3.5. They divide the entire book into fragments and then follow a hierarchical strategy of divide and conquer to summarize the entire corpus of a long book. LLMs have also been used for comment summarization of videos and topic matching for videos (Dimarco 2023). Summarization of long meetings has also been attempted using LLMs like GPT-4 and GPT-3.5 and encoder-decoder models like BART (Schneider and Turchi 2023). There is a work Maity et al. (2024a) on distractor generation in MCQs where a multi-stage prompting approach delivers better results than a single-stage prompting approach. There are several works on question generation (Maity et al. 2023, 2024b, c) for school level subjects using LLMs. There are several works on classification (Deroy and Maity 2021), summarization Deroy et al. (2023b), low resource language learning (Maity et al. 2024d), bias detection in texts (Deroy and Maity 2023), etc which uses LLMs.

However, to our knowledge, LLMs have not been much tried for legal document summarization. We attempt to fill this gap in the present paper.

Hallucinations, Inconsistencies and Unfaithfulness in Large language models and Extractive Summarization models Hallucinations are a serious pitfall of LLMs and other natural language generation models (including abstractive text summarization models) as they may generate content which is non-factual and sometimes irrelevant to the context (Ji et al. 2023). Particularly, in domains such as medical/healthcare and legal, hallucinations can lead to serious outcomes. For instance, a recent work Ahmad et al. (2023) focused on the use of LLMs in the medical domain where there is a reluctance of professionals to use LLMs due to hallucinations and generation of inconsistent content. So this work focused on strategies and techniques to mitigate hallucinations in LLMs so that these models can be suitably used by medical professionals without any issues. Note that unfaithfulness has also been observed in *extractive* summaries (Zhang et al. 2023). This study categorized five types of broad unfaithfulness issues that can arise in extractive summaries, extending beyond mere non-entailment. These issues include incorrect co-reference, incomplete co-reference, incorrect discourse, incomplete discourse, and other forms of misleading information.

In the present work, we point out several examples of hallucinations by LLMs and other abstractive summarization models while summarizing case judgements, and then explore ways to reduce such hallucinations.

Segmentation for long document summarization Legal documents are mostly long, while most LLMs have a limit on the size of the input/prompt that can be given at once. So, in this work, we divide long legal documents into chunks where every chunk is semantically related, and then summarizing one chunk at a time. Such chunking or segmentation-based methods for summarization of long documents have been used in prior works. For instance, semantic segmentation was used in Moro and Ragazzi (2022) to summarize long legal documents in low-resource settings. They partitioned a long input into semantically coherent chunks, allowing transformers to summarize very long documents without truncation by summarizing each chunk and then concatenating the results. Again, a multi-stage summarization framework for long dialogues and documents was developed in Zhang et al. (2022) where the source data was broken into chunks so that it does not exceed the token limit of the summarization models. Moro and Ragazzi (2023) developed a novel align-then-abstract representation learning model for low resource summarization, where they jointly trained a summarizer and a segmenter by maximizing the alignment between the chunk-segment pairs in the output from text segmentation. In another work, Moro et al. (2023) developed another model for long document summarization, where the model processes lengthy inputs by dividing them into multiple text fragments, allowing it to store and compare the current chunk with previous ones. This approach lets the model understand the full context of a document while using a fixed amount of GPU memory.

Present work as an extension of our prior work The present work is a much extended version of our prior work Deroy et al. (2023). The present work extends the work in Deroy et al. (2023) in the following ways: (1) The work Deroy et al. (2023) considered only one dataset of Indian legal documents, while the present work adds extensive experiments on two new datasets of UK legal documents and the GOVREPORT dataset. (2) The present work explores many variations of the LLMs –ChatGPT, Davinci, Llama2-70b and GPT-4-Turbo – which were not explored in Deroy et al. (2023). Also some new extractive summarization methods like PACSUM (Zheng and Lapata 2019) and HipoRank (Dong et al. 2021) are applied in this work. (3) The present work additionally explores different ways of reducing hallucination/inconsistency in the summaries generated by abstractive summarization models and LLMs. We also present a novel semantic similarity-based approach for reducing hallucinations. (4) The present work includes a human evaluation of the summaries generated by different methods.

3 Datasets for legal judgement summarization

For most of the experiments in this paper, we use two datasets of court case judgements and their gold standard summaries, that are described below.

The IN-Abs dataset, obtained from the prior work Shukla et al. (2022), consists of Indian Supreme Court Case Judgements collected from the website of the Legal Information Institute of India.[3] It consists of 7,130 (legal judgement, summary) pairs. The gold standard / reference summaries present in this dataset are the "headnotes" written by legal experts recruited by the Legal Information Institute Of India.[4] As an example, one of the documents used in this dataset can be seen at https://indiankanoon.org/doc/1801104/. The page consists of some meta-data (e.g., names of the petitioner, the respondent, the judges, etc.), the "HEADNOTE", and

3 http://www.liiofindia.org/in/cases/cen/INSC/.

4 We use the terms 'reference summaries' and 'gold standard summaries' interchangeably in this paper.

Table 1 Dataset statistics for IN-Abs dataset

	Nos. of documents	Average nos. of words per document	Average nos. of words per gold-standard summary
Train set	7030	4368.49	839.75
Test set	100	4782.71	932.01

To get the average number of words per document (or summary), we first calculate the number of words in every document (summary). Then we add up the number of words from all documents (summaries), and divide the total word-count by the number of documents (summaries)

Table 2 Dataset statistics for UK-Abs dataset

	Nos. of documents	Average nos. of words per document	Average nos. of words per gold-standard summary
Train set	693	12812.01	1097.38
Test set	100	14476.49	1095.49

The average number of words per document or summary is computed as stated in Table 1

the "JUDGMENT". The dataset was created by collecting such pages, and extracting the headnote part and the judgement part. The headnote part serves as the gold standard summary of the judgement.

The dataset is split into a training set consisting of 7030 (legal judgement, summary) pairs and the test set consisting of 100 (legal judgement, summary) pairs. Table 1 shows the dataset statistics for the IN-Abs dataset.

Following Grusky et al. (2018); Shen et al. (2022), the coverage and density of the legal judgements with respect to the gold-standard summaries are 0.35 and 1.67 respectively for this dataset.

The UK-Abs dataset, which is also obtained from Shukla et al. (2022), consists of a collection of 793 legal case documents from the United Kingdom's Supreme Court website.[5] Along with the judgements, the website also provides official press summaries for legal cases, which are considered as the reference (gold standard) summaries. As an example, one of the cases in this dataset is available at https://www.supremecourt.uk/cases/docs/uksc-2015-0063-judgment.pdf and its press summary is available at https://www.supremecourt.uk/cases/docs/uksc-2015-0063-press-summary.pdf. The dataset was created by extracting the text from such PDF documents.

Out of the 793 (legal document, summary) pairs, 100 pairs were used as the test set and 693 pairs were used as the training dataset. Table 2 shows the dataset statistics for the UK-Abs dataset. Following Grusky et al. (2018); Shen et al. (2022), the coverage and density of the legal judgements with respect to the gold-standard summaries in this dataset are 0.23 and 1.53 respectively.

[5] https://www.supremecourt.uk/decided-cases/.

4 Summarization models

In this study, we explore a variety of summarization models, categorizing them into three main categories – (1) extractive summarization models, (2) general purpose LLMs applied as summarizers, and (3) legal domain-specific abstractive summarization models. This section describes all the models in detail.

4.1 Extractive summarization models

We try 5 different extractive summarization methods. Three of these methods – CaseSummarizer (Polsley et al. 2016), BertSum Liu (2019), SummaRunner (Nallapati et al. 2017) – were observed to perform well for legal case judgement summarization in the prior work (Deroy et al. 2021). Additionally, we apply two recent methods PACSUM (Zheng and Lapata 2019) and HipoRank (Dong et al. 2021).

CaseSummarizer (Polsley et al. 2016) – This unsupervised method ranks sentences based on a TF-IDF matrix which is created using the corpus of legal judgements. CaseSummarizer adjusts sentence scores based on dates, entities, and closeness to section headings. The implementation of this model has been taken from https://github.com/Law-AI/summarization/tree/aacl/extractive/CaseSummarizer.

BertSum (Liu 2019) - BertSum is a supervised summarization model which is based upon a variant of the BERT model. The sentences which are present in the gold standard summary are considered extremely important by the summarization algorithms. Hence BertSum is trained to perform a binary classification task of labeling sentences of the source document as summary-worthy or not. The implementation of this model has been used from-https://github.com/nlpyang/BertSum.

For the IN-Abs dataset, we use the training dataset of IN-Abs of 7030 (legal document, summary) pairs to train the BertSum model. For the UK-Abs dataset, we use the training dataset of UK-Abs of 693 (legal document, summary) pairs to train the BertSum model. Since the gold standard summaries in IN-Abs and UK-Abs are abstractive in nature, we take a sentence from a gold-standard summary and compare it with all sentences in the corresponding source judgement. Based on Moro et al. (2023), we take the 3 sentences with highest Rouge-2 F1 from the main judgement to be used as a part of the reference pseudo-extractive summary for training the model. We perform this step for every sentence in the abstractive gold-standard summary and we take a union of all the corresponding sentences obtained from the main judgement to be labelled as '1' for the binary classification task. All other sentences in the main judgment are labelled as '0' to train the BertSum model.

SummaRunner (Nallapati et al. 2017) – SummaRunner is a supervised summarization model which is based on the Recurrent Neural Network architecture. The implementation of this model has been used from https://github.com/hpzhao/SummaRuNNer. Similar to BertSum (described above), this supervised model is also trained to perform a binary classification task of labeling sentences of the source document as summary-worthy or not. We train the SummaRunner model in exactly the same way over the IN-Abs and UK-Abs training sets, as described above for BertSum.

PACSUM (Zheng and Lapata 2019) - PACSUM (Position-Augmented Centrality based Summarization) is an extractive summarization method that enhances BERT by integrating sentence position information. It uses BERT to generate contextual embeddings, incorporates sentence positions into the scoring mechanism, constructs a similarity graph with position-adjusted weights, ranks sentences based on these weights, and then selects top-ranked sentences to form the summary. This approach results in more relevant and informative summaries by combining BERT's contextual understanding with position-aware scoring. The github link for the code implementation is taken from-https://github.com/mswellhao/PacSum

HiPoRank (Dong et al. 2021) - HiPorank (Hierarchical and Positional Ranking model) is an extractive summarization algorithm that improves summary quality by considering hierarchical structure and positional importance. It organizes documents into sections, paragraphs, and sentences to understand context, prioritizes sentences at the beginning or end of sections/paragraphs, and represents sentences as graph nodes with edge weights adjusted for hierarchy and position. The algorithm scores and ranks sentences using this data and selects top-ranked sentences to form a coherent, comprehensive summary, resulting in more relevant and coherent summaries. The implementation of this model has been taken from -https://github.com/mirandrom/HipoRank.

4.2 General purpose LLMs

We try out the following Large language models (LLMs). All the LLMs take as input a 'prompt' and generate text as a 'response'. Specifically for the summarization task, the prompt consists of (i) the text to be summarized, which we refer to as `<text to summarize>` and (ii) an 'instruction' that tells the model that the input text has to be summarized. Note that the LLMs have a limit on the maximum number of tokens possible in a prompt+response, as stated below.

Text Davinci-003 is an advanced transformer-based LLM with 175 billion parameters. Though detailed information on the exact sources of the training data are not publicly available, it is known that the model has been trained on a massive text dataset using a combination of supervised and unsupervised learning methods. Text-Davinci-003 has a maximum prompt+response length of 4096 tokens.

Turbo-GPT-3.5 (popularly known as ChatGPT) is a cost-effective LLM, said to have approximately 154 billion parameters, which is close to Text-Davinci-003 in terms of performance. This LLM has been trained on a diverse range of text data, including web pages, books, scientific articles, and other sources of human-written text including chats, using a combination of supervised and reinforcement learning methods. Turbo-GPT-3.5 has two different variations -(i) the original version with maximum prompt+response length of 4096 tokens and (ii) a version with a longer prompt+response length of 16,384 tokens, popularly called 'chatgpt-16k'.

GPT-4 Turbo is one of the most powerful LLMs available today. The model can handle both text and images. It has a long context length of 128K due to which we can feed long documents into these models. The model has advanced reasoning and broader general knowledge capabilities.

Llama2-70b is an open-source generative pretrained and fine-tuned text model with 70 billion parameters. It has a relatively long context length of 4096 tokens.

We try the Davinci, GPT-3.5 and GPT-4 models using the OpenAI API.[6] The Llama2-70b model implementation is obtained from Huggingface.[7] All the LLMs were used in 'zero-shot' mode, without any in-context learning or fine-tuning.

4.2.1 Variations of Text-Davinci-003

We try different prompts with the model, leading to the following variations:

(i) **Davinci-summ** For this variant, the prompt is "`<text to summarize>` Summarize the document in `<YY>` words" where *YY* is a number representing the target length of the output summary in number of words. How the value of *YY* is decided is explained later in the section.

(ii) **Davinci-tldr** For this variant, the prompt is "`<text to summarize> Tl;Dr`". We first pass the document followed by the "Tl;Dr" identifier which is an identifier for summarization.

(iii) **Davinci-explicit** For this variant, we use a more explicit prompt – "Your task is to summarize the following document in at most `<YY>` words. The document to be summarized is given within <>. Document to summarize - `<text to summarize>`".

(iv) **Davinci-hybrid** LLMs like Text-Davinci-003 have constraints on the length of prompt+response, which is 4096 tokens at most. So the idea behind this extractive-abstractive hybrid summarization technique is that an extractive summarization method is first used to filter out some key information present in the main judgement. In the second step, the key information is provided to the LLM to summarize the key information as an abstractive summary.

Since CaseSummarizer performed the best amongst the extractive summarization methods (as will be seen later in the paper), we choose the CaseSummarizer method to generate the extractive summary in the first step. First, we create an extractive summary of 1500 words (approx 2000 tokens) or less using the CaseSummarizer method. The sentences chosen for inclusion in the extractive summary are placed in the order in which they are present in the source document. Next we create an abstractive summary from this extractive summary using the prompt "`<text to summarize>` Summarize the document in `<YY>` words" where YY is a number that represents the target length of the output summary in number of words.

4.2.2 Variations of Turbo-Gpt-3.5 (ChatGPT)

Similar to what we tried for Davinci, we try different variations of the ChatGPT model:

6 https://platform.openai.com/docs/api-reference/completions.

7 https://huggingface.co/meta-llama/Llama-2-70b-hf.

(i) **Chatgpt-summ** For this variant, the prompt is "`<text to summarize>` Summarize the document in `<YY>` words" where **YY** is a number representing the target length of the output summary in number of words. How the value **YY** is decided is explained later in the section.

(ii) **Chatgpt-tldr** For this variant, the prompt is "`<text to summarize> Tl;Dr`". We first pass the "Tl;Dr" identifier followed by the text to be summarized.

(iii) **Chatgpt-explicit** For this variant, the prompt is "Your task is to summarize the following document in at most `<YY>` words. The document to be summarized is given within <>. Document to summarize - `<text to summarize>`".

(iv) **Chatgpt-hybrid** The original ChatGPT model also has a constraint on the maximum prompt+response length which is 4096 tokens. So the idea behind this extractive-abstractive hybrid summarization technique is to first use an extractive summarization method to filter out the key information present in the main judgement, and then to generate an abstractive summary of the key information using the LLM. As before, we use CaseSummarizer to generate the extractive summary of at most 1500 words. All sentences in the extractive summary are placed in the order in which they appear in the document. Then we create an abstractive (target) summary from this extractive summary using the prompt "`<text to summarize>` Summarize the document in `<YY>` words" where YY is a number that represents the target length of the output summary in number of words.

(v) **chatgpt-16k-long** Here we use the chatgpt-16k variant which has an input+response length of 16k tokens. This variant can be fed with longer inputs, hence it is natural to try out this variant for summarization of long legal documents. For this variant, we use the prompt "Summarize the document in `<YY>` words: `<text to summarize>`" (the prompt is the same as in Chatgpt-summ) where *YY* is a number representing the target length of the output summary in number of words.

4.2.3 Variations of GPT-4-Turbo (gpt4)

The various GPT4 variations we experiment with are:-

(i) **gpt4-summ** For this variant, the prompt is "`<text to summarize>` Summarize the document in `<YY>` words" where *YY* is a number representing the target length of the output summary in number of words. How the value *YY* is decided is explained later in the section.

(ii) **gpt4-tldr** For this variant, the prompt is "`<text to summarize>` Tl;Dr". We first pass the "Tl;Dr" identifier followed by the text to be summarized.

(iii) **gpt4-explicit** For this variant, the prompt is "Your task is to summarize the following document in at most `<YY>` words. The document to be summarized is given within <>. Document to summarize - `<text to summarize>`".

(iv) **gpt4-hybrid** In this extractive-abstractive hybrid summarization technique, we first use an extractive summarization method to filter out the key information present in the main judgement, and then to generate an abstractive summary of the key information using the LLM. As before, we use CaseSummarizer to generate the extractive summary of at most 1500 words. All sentences in the extractive summary are placed in the order in which they appear in the document. Then we create an abstractive (target) summary from this extractive summary using the prompt "`<text to summarize>` Summarize the document in `<YY>` words" where *YY* is a number that represents the target length of the output summary in number of words.

4.2.4 Variations of Llama2-70b

Similar to the variations for the other LLMs, we try the following variations of Llama2-70b:-

(i) **llama-summ** For this variant, the prompt is "`<text to summarize>` Summarize the document in `<YY>` words" where *YY* is a number representing the target length of the output summary in number of words. How the value *YY* is decided is explained later in the section.

(ii) **llama-tldr** For this variant, the prompt is "`<text to summarize>` Tl;Dr". We first pass the "Tl;Dr" identifier followed by the text to be summarized.

(iii) **llama-explicit** For this variant, the prompt is "Your task is to summarize the following document in at most `<YY>` words. The document to be summarized is given within <>. Document to summarize - `<text to summarize>`".

(iv) **llama-hybrid** In this extractive-abstractive hybrid summarization technique we first use an extractive summarization method (CaseSummarizer) to filter out the key information present in the main judgement, and then to generate an abstractive summary of the key information using the LLM. As before, we use CaseSummarizer to generate the extractive summary of at most 1500 words. All sentences in the extractive summary are placed in the order in which they appear in the document. Then we create an abstractive (target) summary from this extractive summary using the prompt "`<text to summarize>` Summarize the document in `<YY>` words" where *YY* is a number that represents the target length of the output summary in number of words.

4.3 Legal domain-specific abstractive summarization models

We tried two different legal domain-specific abstractive summarization models namely Legal-Pegasus (abbreviated as LegPegasus) and Legal-LED (abbreviated as LegLED).

LegPegasus Pegasus (formally, `google/pegasus-cnn_dailymail`) is a general-purpose abstractive summarization model developed by Google. This model was fine-tuned on the 'sec-litigation-releases' dataset – consisting of 2,700 litigation releases and complaints related to civil lawsuits in various courts in the United States of America (USA) along with their summaries – to develop the LegPegasus model designed specifically for abstractive summarization in the legal domain. Leg-Pegasus can be accessed at https://huggingface.co/nsi319/legal-pegasus and has a maximum input token length of 1024 tokens.

LegLED This model is based on the Longformer architecture, a transformer-based neural network architecture designed to process long sequences of text efficiently. The LegLED model has also been fine-tuned on the same 'sec-litigation-releases' dataset consisting of 2700 (legal document, summary) pairs related to civil lawsuits in various courts in the USA. It is also specifically designed for summarization in the legal domain. The LegLED model can be accessed at https://huggingface.co/nsi319/legal-led-base-16384. The model has a maximum input token length of 16,384.

4.4 Chunking of long legal documents

As stated earlier, Text-Davinci-003 and Turbo-Gpt-3.5 have a token limit of maximum 4,096 for (prompt+generated text). One token is approximately $\frac{3}{4}$ words, i.e., 1000 tokens correspond to around 750 words.[8] Also, legal domain-specific abstractive summarization models like LegPegasus have a maximum input token length of 1,024.

Recall from Sect. 3 that the average length of UK case judgements in the UK-Abs test dataset is 14,476 words and that of the Indian case judgements in the IN-Abs test dataset is 4,782 words. Thus, these legal case judgements are often much longer than what can be input into the summarization models/LLMs at once, and hence we have to follow a divide-and-conquer approach with the long legal documents. Our strategy involves chunking long legal documents into smaller segments or chunks of at most K words (where K can be 1,024 or 2,048 or higher), and each chunk is passed individually into the summarization models to obtain the output summary. Then the summaries generated for all the chunks (of a given document) are appended together in the order in which the chunks appear in the main document, to form the final output summary for the legal judgement. For legal documents with a length of lesser than 1024 words, the summary is obtained at once by passing the entire document through the summarization models.

For the ChatGPT and Davinci models, we experiment with $K = 1{,}024$ and 2,048. For the chatgpt-16k model, we divide a legal judgement into longer chunks of length $K = 8192$ words (10,922 tokens approx). Note that almost all documents in the IN-Abs test set and a large majority of documents in the UK-Abs test set are actually shorter than 8,192 words; hence the summary is obtained at once by passing the entire document through the chatgpt-16k model.

Deciding the target summary length of a chunk When text is input to an LLM such as ChatGPT, we also need to specify a hyperparameter namely 'max tokens' to specify the maximum desired length of the summary to be generated (in terms of number of words).

[8] Detailed explanation about tokens available at https://help.openai.com/en/articles/4936856-what-are-tokens-and-how-to-count-them.

Let $|Doc|$ be the length of an input document, and let $|Exp|$ be the length of its reference summary written by human experts. Assume that the document is split into a series of chunks, each of size K words. Then the target summary length to be generated by the model for each model is $K \times |Exp|/|Doc|$ words. In other words, we use the LLMs / summarization models to summarize every chunk considering the same compression ratio as for the whole document and the reference summary.

There is an inherent limitation in this simple method. In reality, all parts of the document are not equally important, hence different chunks should ideally be allocated different lengths in the summary. In contrast, this method allocates the same length in the summary for all chunks. However, there is no simple way of knowing the relative importance of different chunks in a legal case judgement; hence we follow the simple method of assuming the same compression ratio for every chunk.

5 Performance metrics

We compare the quality of summaries generated by the different methods along two aspects – (1) their match with the gold standard or reference summaries, and (2) their consistency with the input documents.

5.1 Match with gold-standard summaries

Here we measure the match of an algorithm-generated summary for a document with the gold standard / reference summary of the same document. The following metrics are used to measure the performance of the summarization models.

ROUGE The term 'ROUGE' stands for "Recall-Oriented Understudy for Gisting Evaluation" (Lin 2004). ROUGE is a family of metrics used for the automatic evaluation of machine-generated text, particularly in the context of text summarization. Here we measure the Rouge-score between the expert-written summaries and model-generated summaries. Specifically, we calculate Rouge-2 precision, recall and F1 scores, which evaluate the *bigram match*, and Rouge-L precision, recall, F1 scores which measure the Longest Common Subsequence-based match between the model-generated summaries and the gold standard summaries.

METEOR (Banerjee and Lavie 2005) is a metric used for the automatic evaluation of machine translation and summarization output. It was designed to address some limitations of other metrics like BLEU by incorporating explicit word order information and considering synonyms and stemming. This metric measures the unigram overlap between expert-written summaries and model-generated summaries.

BERTSCORE (Zhang et al. 2019) is a popular metric for evaluating the quality of machine-generated text, especially in the context of natural language processing (NLP) tasks such as text summarization, machine translation, and question answering. Unlike traditional evaluation metrics like ROUGE, which rely on exact matches or n-gram overlaps, BERTScore takes into account the *semantic similarity* between the reference (ground truth) summary and the model generated summary.

To implement these metrics, we employ the SummEval package available at https://github.com/Yale-LILY/SummEval, which is a well-known toolkit used for the evaluation of summarization.

5.2 Metrics for consistency of summaries

Now we discuss three metrics used to assess the consistency of model-generated summaries. These metrics compare a model-generated summary with the original document and estimate how consistent the summary is with the document. All these metrics give a score in the range [0, 1]; the higher the score, the more consistent is the summary.

SummaC (Laban et al. 2022) – This metric utilizes Natural Language Inferencing (NLI) to determine the logical relationship between sentences. The NLI task involves determining the relationship between two sentences. One of the sentences is considered as a 'hypothesis' and the other sentence is considered as a 'premise'. Typically, a NLI model will give a score representing how likely the hypothesis sentence is to logically follow from the premise sentence.

Given a (document, summary) pair, the SummaC metric (Laban et al. 2022) computes NLI scores for each sentence in the model-generated summary, indicating the likelihood that the sentence logically follows from the sentences in the original document. Lower NLI scores for a particular sentence s in the summary suggest a higher mismatch between this sentence and the document, and hence the potential presence of hallucinated information. The NLI scores of individual sentences in the summary are combined to give a single SummaC score for the given (document, summary) pair. A higher SummaC score indicates greater consistency between the model-generated summary and the original document. We use the standard implementation of SummaC available at https://github.com/tingofurro/summac.

NumPrec - Numbers play a significant role in legal case judgements, such as dates, statute identifiers, monetary values, etc. The NumPrec metric measures the fraction of numbers present in the model-generated summary that also appear in the source document. We rely on the standard Python library for number identification.

NEPrec - Named Entities are important for legal case judgements, and changes in entities like a person's name or organization's name can result in information loss and potential misrepresentation. The NEPrec metric measures the fraction of named entities present in the model-generated summary that also exists in the original document. We use the Spacy toolkit[9] to detect named entities in original documents as well as the summaries. It is worth noting that the accuracy of the NEPrec metric depends on the accuracy of the toolkit used to identify named entities.

[9] https://spacy.io/.

Table 3 Summarization performances of ChatGPT, Davinci, and Llama for different chunk sizes on the IN-Abs dataset

Model	R2-P	R2-R	R2-F1	RL-P	RL-R	RL-F1	ME	BS
Chatgpt								
Chatgpt-summ (1024)	0.196	**0.173**	**0.181**	0.236	**0.208**	**0.218**	**0.196**	**0.627**
Chatgpt-summ (2048)	0.187	0.151	0.163	0.208	0.191	0.200	0.185	0.617
Chatgpt-16k-long (8192)	**0.231**	0.127	0.149	**0.319**	0.176	0.206	0.147	0.615
Davinci								
Davinci-summ (1024)	**0.220**	**0.179**	**0.195**	**0.251**	**0.205**	**0.223**	0.191	0.624
Davinci-summ (2048)	0.190	0.167	0.174	0.223	0.187	0.192	**0.218**	**0.627**
Llama								
Llama-summ (1024)	**0.186**	**0.144**	**0.163**	**0.219**	**0.187**	**0.192**	0.152	**0.621**
Llama-summ (2048)	0.180	0.140	0.158	0.213	0.180	0.182	**0.166**	0.600

Chunk sizes are stated within parentheses. The same prompt is used for all models. Best value of each metric for each family of models is in bold font

6 Results

In this section, we study the performance of various summarization models on the two datasets.

6.1 Selecting the chunk size for different LLMs

First we compare the performances of Chatgpt, Davinci, and Llama with different chunk sizes, to check which chunk size gives the best performance. We perform these experiments on the IN-Abs dataset.

Table 3 shows the ROUGE, METEOR, BERTScore metrics for chatgpt-summ, chatgpt-16k-long, davinci-summ, and llama-summ, using different chunk sizes. The same prompt is used for all the different variations of Chatgpt, Davinci, and Llama, as stated in Sect. 4.2. The highest value for every metric is shown in blue-bold.

For all the three LLMs (chatgpt, davinci and Llama), the best results for most metrics are obtained with chunk-size 1024. This is possibly because the LLMs find it difficult to capture the context when the chunk sizes become too large, thus leading to better summaries with chunk size 1024. However, we preferred not to use even shorter chunks, since there is a possibility of redundancy and loss of continuity/coherence if the document is broken into too many chunks. Hence, we perform all subsequent experiments with chatgpt, davinci, and Llama considering a chunk size of 1024 words.

For chatgpt-16k, we will continue to use longer chunks of 8192 words. For GPT-4 Turbo, we feed the entire document into the model since GPT-4 Turbo has a very long context length of 128K.

Table 4 Summarization performances of general-purpose LLMs on the UK-Abs dataset for different prompts

Model	R2-P	R2-R	R2-F1	RL-P	RL-R	RL-F1	ME	BS
Llama2-70b(chunk size 1024)								
llama-tldr	0.156	0.121	0.136	0.186	0.159	0.168	0.184	0.605
llama-summ	0.178	0.126	0.148	0.234	0.174	0.186	0.196	0.625
llama-explicit	0.165	0.129	0.148	0.206	0.173	0.181	0.200	0.629
llama-hybrid	0.210	0.062	0.104	0.320	0.104	0.137	0.121	0.619
GPT-4 Turbo								
gpt4-tldr	0.172	0.134	0.149	0.203	0.179	0.187	0.208	0.629
gpt4-summ	0.193	0.146	0.159	0.235	0.182	0.198	0.207	0.631
gpt4-explicit	0.184	0.148	0.168	0.228	0.179	0.195	0.209	0.643
gpt4-hybrid	0.220	0.064	0.107	**0.327**	0.108	0.148	0.125	0.626
Chatgpt (chunk size 1024)								
chatgpt-tldr	0.152	0.151	0.151	0.198	0.196	0.196	0.216	0.634
chatgpt-summ	0.188	0.160	0.170	0.227	0.194	0.205	0.214	0.636
chatgpt-explicit	0.164	0.163	0.163	0.214	**0.210**	0.211	0.218	0.640
chatgpt-hybrid	0.214	0.086	0.113	0.307	0.126	0.174	0.116	0.626
Davinci (chunk size 1024)								
davinci-tldr	**0.222**	0.143	0.169	0.260	0.166	0.197	0.207	0.641
davinci-summ	0.196	**0.181**	**0.187**	0.223	0.206	**0.213**	**0.219**	0.638
davinci-explicit	0.175	0.159	0.166	0.214	0.195	0.203	0.209	**0.643**
davinci-hybrid	0.172	0.144	0.154	0.295	0.113	0.154	0.121	0.602
Chatgpt-16k (chunk size 8192)								
chatgpt-16k-long	0.153	0.144	0.148	0.213	0.199	0.205	0.183	0.619

The value shown in underline is the best value in every summarization family for every metric. The value shown in bold is the highest value for every metric

6.2 Summarization results on UK-Abs dataset

We first compare the performances of LLM-based summarization using different prompts (that were stated in Sect. 4.2), and select the best LLM variants for summarization of this dataset. Then we compare the best LLM variants with other extractive and abstractive summarization models.

Comparing LLM-based summarization variants Table 4 shows the ROUGE, METEOR, and BERTScore metric for the general-purpose LLMs on the UK-Abs dataset. Among the Chatgpt variants (all of which are run considering a chunk length of 1024 words, as stated earlier), the Rouge-2 F1 score for chatgpt-summ is higher than the other chatgpt variants. The chatgpt-explicit variation achieves the highest score amongst all chatgpt variations for most metrics, including Rouge-2 recall, Rouge-L Recall, Rouge-L F1, METEOR and BERTScore. Among the Davinci variants, Davinci-summ achieves the highest scores for most of the metrics. Among the gpt4 variants, gpt4-explicit achieves the highest scores for

Table 5 Comparing performances of the best performing general-purpose LLM variants, legal domain-specific abstractive models, and extractive summarization models on the UK-Abs dataset

Model	R2-P	R2-R	R2-F1	RL-P	RL-R	RL-F1	ME	BS
General domain LLMs								
gpt4-explicit	0.184	0.146	0.168	0.228*	0.179	0.195	0.206	0.623
chatgpt-explicit	0.164	0.163	0.163	0.214	**0.210***	0.211*	0.218	**0.640**
llama-explicit	0.165	0.129	0.148	0.206	0.173	0.181	0.200	0.629
davinci-summ	0.196	**0.181**	**0.187**	0.223*	0.206*	**0.213***	**0.219**	0.638
Legal domain-specific abstractive models								
LegPegasus	**0.249***	0.085	0.101	**0.248***	0.113	0.136	0.195	0.635
LegLED	0.108	0.107	0.107	0.151	0.149	0.150	0.173	0.601
Extractive models								
HipoRank	0.161	0.147	0.155	0.165	0.140	0.161	0.204	0.602
PACSUM	0.152	0.140	0.150	0.158	0.138	0.141	0.202	0.606
SummaRunner	0.148	0.144	0.146	0.153	0.149	0.151	0.209	0.617
CaseSummarizer	0.173	0.164	0.168	0.174	0.165	0.169	0.204	0.635
BertSum	0.158	0.156	0.157	0.158	0.155	0.156	0.217	0.624

The value shown in underline is the best value in every summarization family for every metric. The value shown in bold is the highest value for every metric. Entries with an asterisk (*) indicate a value that is statistically significantly higher in terms of the student T-test at a 95% confidence interval than the best value achieved by an extractive summarization model for the same metric

most of the metrics. Among the llama variants, llama-explicit achieves the highest scores for most of the metrics.

We observe that the 'hybrid' variants have higher Rouge-2 precision and Rouge-L precision across all the model variants. This suggests that the summaries generated by the 'hybrid' variants contains many of the same sequences of words (in the same order) as the reference / gold standard summaries. Whereas, the 'explicit' and 'summ' variants have higher Rouge-2 recall and Rouge-L recall across all the model variants, suggesting that the summaries include a significant portion of the content of the reference summaries.

An interesting observation is that the summaries created by GPT-4 Turbo are generally shorter in length, compared to the summaries created by Turbo-GPT-3.5 (Chatgpt). This is why the summaries created by Chatgpt have higher matches with the gold standard summaries, thereby having higher scores for the Recall and FI metrics. However, the summaries generated by GPT-4 achieve higher Rouge-Precision scores in most cases.

We now compare some of the best performing LLM variants with the other types of summarization models.

Comparing general-domain LLMs with domain-specific abstractive summarizers and extractive models Table 5 shows the ROUGE scores, METEOR, and BERTScore metrics for some of the best-performing general-domain LLM versions, legal domain-specific abstractive models and extractive summarization models on the UK-Abs dataset.

Table 6 Consistency metrics for summaries generated by various models on UK-Abs dataset

Model	SummaC	NEPrec	NumPrec
Llama2-70b			
llama-tldr	0.568	0.917	0.925
llama-summ	0.558	0.906	0.921
llama-hybrid	0.604	0.887	0.947
llama-explicit	0.606	0.875	0.936
GPT4-Turbo			
gpt4-tldr	0.595	**0.956**	0.983
gpt4-summ	0.586	0.937	0.955
gpt4-hybrid	0.641	0.927	**0.984**
gpt4-explicit	0.636	0.927	0.979
Chatgpt			
chatgpt-tldr	0.588	0.934	0.965
chatgpt-summ	0.570	0.921	0.933
chatgpt-hybrid	0.630	0.902	0.963
chatgpt-explicit	0.622	0.900	0.952
Davinci			
davinci-tldr	0.623	0.833	0.912
davinci-summ	0.634	0.926	0.977
davinci-hybrid	0.624	0.903	0.956
davinci-explicit	**0.677**	0.887	0.963
Chatgpt-16k			
chatgpt-16k-long	0.644	**0.956**	0.981
Legal domain-specific abstractive models			
LegLED	0.628	0.874	0.914
LegPegasus	0.649	0.907	0.959

The value shown in underline is the best value in every summarization family for every metric. The value shown in bold is the overall highest value for every metric

Amongst the extractive summarization models, CaseSummarizer has performed better than other extractive methods across most metrics. Among the legal domain-specific abstractive models, LegLED achieves slightly higher ROUGE-2 and ROUGE-L Recall and F1 scores, while LegPegasus achieves higher METEOR and BERTScore values, and ROUGE-2 and ROUGE-L Precision scores.

Across all methods, the highest Rouge-2 recall, Rouge-2 F1 score, Rouge-L F1 score, and the highest METEOR score are achieved by davinci-summ. We see that the general-domain LLMs (davinci-summ in particular) perform better than the extractive models for the UK-Abs dataset.

The highest Rouge-2 Precision and Rouge-L Precision are obtained for LegPegasus, whereas LegLED achieves higher Rouge-2 Recall and Rouge-L Recall scores. We observe that the summaries generated by LegPegasus are generally shorter than the summaries generated by LegLED for the UK-Abs dataset. Specifically, the average length of summaries generated by LegPegasus is 820.77 words, while that of LegLED is 890.63 words. This difference in length of the summaries plays a role in higher ROUGE precision scores for LegPegasus and higher ROUGE Recall scores for LegLED.

Consistency of generated summaries Table 6 shows the consistency metrics for general-purpose LLMs and the domain-specific abstractive models on the UK-Abs dataset. The value shown in blue is the best value in every summarization family for every metric. The value shown in blue-bold is the highest value for every metric. We found these metric values to be 1.0 for the extractive summaries, hence we do not show them in the table.[10]

Among the chatgpt variants, chatgpt-tldr gets the highest NEPrec and NumPrec scores, while chatgpt-hybrid gets the highest SummaC score. Among the gpt4 variants, gpt4-tldr gets the highest NEPrec scores, while gpt4-hybrid gets the highest SummaC and NumPrec scores. Among the llama variants, llama-tldr gets the highest NEPrec scores, while llama-hybrid gets the highest NumPrec score and llama-explicit gets the highest SummaC score. Whereas, among the davinci variants, davinci-summ gets the highest NEPrec and NumPrec scores, and the highest SummaC score is obtained by Davinci-explicit (which is the overall highest score for SummaC). The overall highest values for NEPrec and Numprec scores are achieved by GPT4-Turbo (NEPrec jointly highest with chatgpt-16k-long).

From Table 6, it is evident that there are instances of hallucination / inconsistency in the summaries generated by LLMs and other generative summarization models. We will see some actual examples of such inconsistencies in the next section.

6.3 Summarization results on IN-Abs dataset

We now study the performances of various summarization models on the IN-Abs dataset. As we did in the previous section for the UK-Abs dataset, here also we first compare the performances of LLM-based summarization using different prompts (that were stated in Sect. 4.2), and select the best LLM variants for summarization of this dataset. Then we compare the best LLM variants with other extractive and abstractive summarization models.

Comparing LLM-based summarization variants Table 7 compares the performance of general-domain LLMs on the IN-Abs dataset, for different prompts. The best value for a metric within a particular family of summarization models is shown in blue, and the overall best value for every metric is shown in blue-bold.

Amongst the chatgpt models, chatgpt-summ achieves the best value for most metrics, including METEOR and BERTScore, although chatgpt-explicit achieves the best ROUGE-L Recall and F1 scores. Amongst the gpt4 models, gpt4-summ achieves the best value for most metrics, including METEOR and BERTScore, although gpt4-explicit achieves the best ROUGE-L Recall and F1 scores. Among the llama models, different variants get the highest score for different metrics.

[10] Note that a recent study has shown that even extractive summaries can have different types of inconsistencies (Zhang et al. 2023). However, the metrics we applied gave value 1.0 for the extractive summaries.

Table 7 ROUGE, METEOR, and BERTScore metrics for general-domain LLMs on the IN-Abs dataset, for different prompts

Model	R2-P	R2-R	R2-F1	RL-P	RL-R	RL-F1	ME	BS
Llama2-70b(chunk size 1024)								
llama-tldr	0.243	0.113	0.148	0.307	0.147	0.184	0.142	0.594
llama-summ	0.194	0.147	0.162	0.241	0.181	0.185	0.173	0.593
llama-explicit	0.209	0.141	0.157	0.262	0.182	0.206	0.158	0.591
llama-hybrid	0.161	0.122	0.131	0.246	0.147	0.165	0.129	0.581
GPT4-Turbo								
gpt4-tldr	**0.256**	0.126	0.159	0.316	0.162	0.207	0.159	0.602
gpt4-summ	0.216	0.158	0.174	0.253	0.192	0.206	0.183	0.618
gpt4-explicit	0.216	0.156	0.173	0.283	0.209	0.224	0.172	0.604
gpt4-hybrid	0.183	0.127	0.141	0.263	0.155	0.175	0.149	0.601
Chatgpt (chunk size 1024)								
chatgpt-tldr	0.238	0.140	0.173	0.298	0.175	0.204	0.167	0.609
chatgpt-summ	0.199	0.177	0.186	0.232	0.209	0.214	**0.193**	**0.624**
chatgpt-explicit	0.205	0.174	0.175	0.267	**0.224**	**0.235**	0.186	0.605
chatgpt-hybrid	0.195	0.134	0.144	0.257	0.173	0.188	0.155	0.616
Davinci (chunk size 1024)								
davinci-tldr	0.233	0.125	0.156	0.284	0.152	0.190	0.141	0.604
davinci-summ	0.220	**0.179**	**0.195**	0.251	0.205	0.223	0.191	0.624
davinci-explicit	0.169	0.125	0.140	0.245	0.188	0.209	0.170	0.616
davinci-hybrid	0.201	0.121	0.145	0.236	0.180	0.196	0.168	0.612
Chatgpt-16k (chunk size 8192)								
chatgpt-16k-long	0.231	0.127	0.149	**0.319**	0.176	0.206	0.147	0.615

The values shown in underline are the best values in every summarization family for every metric. The values shown in bold are the highest values for every metric across all methods

Among the davinci models, davinci-summ achieves the highest values for most metrics.

We observe that the 'tldr' variants have higher Rouge-2 precision and Rouge-L precision across all the model variants. Whereas, the 'summ' and 'explicit' variants have higher Rouge-2 and Rouge-L recall and F1 scores across all the model variants.

An interesting observation is that the summaries produced by GPT-4 Turbo are shorter than those generated by Turbo-GPT-3.5 (Chatgpt). As a result, the summaries from Chatgpt exhibit higher ROUGE Recall and F1 scores, whereas the GPT-4 summaries achieve higher ROUGE precision scores.

We now compare some of the best performing LLM variants with the other types of summarization models.

Comparing general-domain LLMs with domain-specific abstractive summarizers and extractive models Table 8 shows the ROUGE, METEOR, BERTScore metrics for best performing general-domain LLMs, abstractive and extractive summarization models on the IN-Abs dataset.

Table 8 ROUGE, METEOR, BERTScore metrics for best performing general-domain LLMs, abstractive and extractive summarization models on the IN-Abs dataset

Model	R2-P	R2-R	R2-F1	RL-P	RL-R	RL-F1	ME	BS
General domain LLMs								
gpt4-summ	0.216	0.153	0.174	<u>0.253</u>	0.192	0.206	0.183	0.618
chatgpt-summ	0.199	0.177	0.186	0.232	**0.209**	0.214	<u>0.193</u>	**0.624**
llama-summ	0.194	0.147	0.162	0.241	0.181	0.185	0.173	0.593
davinci-summ	<u>0.220</u>	<u>0.179</u>	<u>0.195</u>	0.251	0.205	**0.223**	0.191	**0.624**
Legal domain-specific abstractive models								
LegPegasus	<u>0.196</u>	<u>0.120</u>	<u>0.133</u>	**0.263***	<u>0.154</u>	<u>0.172</u>	<u>0.194</u>	<u>0.594</u>
LegLED	0.111	0.107	0.108	0.150	0.146	0.147	0.142	0.590
Extractive models								
PACSUM	0.228	0.207	0.216	0.229	0.185	0.202	0.200	0.603
HipoRank	0.221	0.204	0.207	0.214	0.184	0.194	0.194	0.602
SummaRunner	0.227	0.210	0.218	0.198	0.182	0.189	**0.203**	<u>0.621</u>
CaseSummarizer	**0.251**	**0.226**	**0.238**	<u>0.231</u>	<u>0.208</u>	<u>0.219</u>	0.194	0.614
BertSum	0.248	0.218	0.232	0.226	0.196	0.209	0.202	0.615

The value shown in underline is the best value in every summarization family for every metric. The value shown in bold is the highest value for every metric. Entries with an asterisk (*) indicate a value that is statistically significantly higher in terms of the student T-test at a 95% confidence interval than the best value achieved by an extractive summarization model

Among the extractive models, CaseSummarizer achieves the highest metric values for most metrics, especially the ROUGE-based metrics. In fact, for most ROUGE-based metrics, CaseSummarizer achieves the overall highest values for all metrics. Among the legal domain-specific abstractive models, LegPegasus performs better than LegLED across all metrics. Among the LLMs, chatgpt-summ achieves higher ROUGE-L Recall and BertScore values than the extractive models. Thus for the IN-Abs dataset, it is seen that the abstractive models (including LLMs) perform at par with the best extractive models.

Consistency of generated summaries Table 9 shows the consistency metrics for summaries generated by all abstractive models on the IN-Abs dataset. The value shown in blue is the best value in every summarization family for every metric. The value shown in blue-bold is the overall highest value for every metric. We found these metric values to be 1.0 for the extractive summaries, hence we do not show them in the table.

Among the chatgpt variants, chatgpt-hybrid gets the highest SummaC and NEPrec scores, while chatgpt-explicit gets the highest NumPrec score. Among the gpt4 variants, gpt4-hybrid gets the highest SummaC scores, while gpt4-explicit gets the highest NumPrec score (which is the overall highest score for NumPrec) and gpt4-summ gets the highest NEPrec scores. Among the llama2-70b variants, llama-hybrid gets the highest SummaC and NEPrec scores, while llama-explicit gets the highest NumPrec score. Among the davinci variants, davinci-explicit gets the highest SummaC (highest overall) and NumPrec scores, and the highest NEPrec score is obtained by davinci-hybrid.

Table 9 Consistency metrics for different abstractive summarization models on IN-Abs dataset

Model	SummaC	NEPrec	NumPrec
Llama2-70b			
llama-tldr	0.559	0.846	0.927
llama-summ	0.558	0.883	0.921
llama-hybrid	0.605	0.904	0.904
llama-explicit	0.579	0.884	0.928
GPT4-Turbo			
gpt4-tldr	0.581	0.905	0.966
gpt4-summ	0.596	0.942	0.978
gpt4-hybrid	0.652	0.927	0.973
gpt4-explicit	0.631	0.909	**0.986**
Chatgpt			
chatgpt-tldr	0.570	0.851	0.943
chatgpt-summ	0.573	0.901	0.956
chatgpt-hybrid	0.635	0.907	0.963
chatgpt-explicit	0.608	0.886	0.974
Davinci			
davinci-summ	0.635	0.895	0.932
davinci-tldr	0.608	0.833	0.912
davinci-explicit	**0.694**	0.831	0.957
davinci-hybrid	0.622	0.904	0.956
Chatgpt-16k			
chatgpt-16k-long	0.633	**0.944**	0.975
Legal Domain-specific abstractive models			
LegPegasus	0.633	0.842	0.948
LegLED	0.656	0.719	0.819

The value shown in underline is the best value in every summarization family for every metric. The value shown in bold is the highest value for every metric

Again, it is evident that there are instances of hallucination / inconsistency in the summaries generated by LLMs and other generative summarization models. We will see some actual examples of such inconsistencies in the next section.

7 Examples of inconsistencies and hallucinations in abstractive summaries

In the context of generative models such as LLMs and abstractive summarizers, *hallucination* refers to the generation of text that is not based on real or accurate information but *appears to be* coherent and contextually relevant (Ji et al. 2023). These models generate responses/outputs by predicting the next word or sequence of words based on patterns learned from vast amounts of training data. Occasionally, these models can generate responses that are factually incorrect, nonsensical, or entirely made up, resembling hallucinations in human perception.

From the relatively low values of the SummaC metric that we observed in the previous section, it appears that the LLMs and legal abstractive summarizers are likely to hallucinate parts of the summary. To check if this is the case, we manually observed several (document, summary) pairs in the two datasets and found some inconsistencies / hallucinations in almost all of the summaries that we manually checked. In particular, we observed those sentences that obtained relatively low SummaC scores, and those sentences that contained numbers and named entities that could *not* be matched with the original documents (while computing NERPrec and NumPrec). We also observed the relevant parts in the main documents to understand the errors/inconsistencies.

We present some examples of errors/inconsistencies in this section. Tables 10 and 11 show examples of such errors in the summaries of two Indian Supreme Court cases, while Tables 12 and 13 show examples of errors in summaries of two cases of the UK Supreme Court. In all tables, the summarization model that committed the error is stated, an extract from the summary showing the error is given, and the error is explained briefly.

In Table 10, the first example shows two wrong monetary values that are stated in the summary generated by chatgpt-summ, but are *not* present in the main judgement. The second example shows a hallucinated statute included in the summary generated by chatgpt-summ, that is not present in the main judgement. Other examples show incomplete lines in the summary, and names of persons wrongly stated in the summary. Table 11 also shows examples where the name of the judge is stated wrongly in the summary generated by LegLED, the name of an important statute is omitted from the summary generated by chatgpt-summ, and where LegLED has hallucinated the name of a court in the USA while summarizing a judgement of the Indian Supreme Court.

Tables 12 and 13 show examples from two UK Supreme Court case judgement summaries. In Table 12, a wrong monetary value in mentioned in the summary in the first example (actually, Davinci-summ possibly confused the amount "$2.5m" with "Clause 4.5", both of which appear in the judgement, and wrongly generated "4.5m" as the amount). The second example shows a statute that is included in the summary ("Section 17(a) of Securities Act of 1933") but which is absent in the main judgement. So this information is hallucinated. Table 13 also shows examples of hallucinated country names and court names, as well as instances where a statute name is only incompletely stated in the summary.

We noticed one strange type of error particularly in summaries generated by LegLED – even when the model is summarizing Indian or UK case judgements, names of U.S. Courts and names of U.S. statutes come up in the summaries, which are not at all related to the input document (several examples are given in the tables). Such hallucinations are probably due to the fact that LegLED has been trained on US legal document-summary pairs, and the model has a tendency of generating US court / statute names that it has seen during training.

Along with the more serious errors that we have shown – such as names and monetary values being reported wrongly, court names and statute names being

Table 10 Examples of errors in abstractive summaries generated by different summarization models for the Indian Court judgement available at https://indiankanoon.org/doc/1801104/

id	Model	Extract from summary showing error	Explanation of error
1	chatgpt-summ	... *27,000* in currency notes on the pretext that he wanted to pay the balance of *Rs 26,500* due to the accused under the agreement alleged to have been entered into between them..	There is no mention of Rs. 26,500 or Rs. 27,000 in the main document. Both these amounts are hallucinated. The amount stated in the original document is Rs. 29,500
2	chatgpt-summ	On these materials, the learned Magistrate held that the appellant had committed an offence under *Section 387 of the Indian Penal Code* and convicted him accordingly	Section 387 of the Indian Penal Code is not mentioned in the main document. This Section 387 is hallucinated
3	davinci-tldr	The Supreme Court accepted the argument raised on behalf of the	Incomplete line
4	LegLED	On February 1, *2019*, the Honorable *Chandrasekhar A. Lama* of the *Presidency of the United Kingdom* entered a final judgment against W.H. King.....	The name of the judge according to the original judgement is “Aiyar, N. Chandrasekhara” which is written wrongly in the summary. The judge clearly does not belong to the Presidency of United Kingdom, since this is a case in the Indian Supreme Court. Also, the year 2019 does not occur in the judgement

The errors in the summaries are marked in italic, and explained in the last column

Table 11 Examples of errors in abstractive summaries generated by different summarization models for the Indian Court judgement available at https://indiankanoon.org/doc/302141/

id	Model	Extract from summary showing error	Explanation of error
1	LegLED	On March 15, 2019, the *Honorable M.L. Mehta* of the *U.S. District Court* for the District of Ludhiana entered a final judgment against the defendants in the *Ludhiana, Ludhiana, Ludhiana, Ludhiana, and Ludhiana* Revision Petition No. 17 R/55 of 1955	There is an advocate named “K. L. Mehta” but no judge named “M. L. Mehta” in the original judgement. The word “Ludhiana” has been repeated multiple times. Also there is no mention of U.S. District Court in the judgement
2	chatgpt-summ	... as their application for the allotment of Raikot land was rejected by the Deputy Custodian General on the ground that his jurisdiction to revise the order has been taken away by virtue of the provisions of, *(44 of 1954)* and the notification issued thereunder on March 24, 1955	The original document states “Displaced Persons (Compensation and Rehabilitation) Act, 1954 (44 of 1954)” but the name of the Act is not mentioned in the summary; a critical detail is omitted
3	LegLED	According to the *SEC’s complaint filed in the U.S. District Court for the Southern District of New York*, *from March 24, 1955 to March 24, 1955*, the Raikot lands were allotted to the appellants.....	The summary talks about “U.S. District Court, SEC”, and “Southern District of New York” which are completely unrelated to this case, since the case is in the Indian Supreme Court. Also “from March 24, 1955 to March 24,1955” is meaningless

The errors in the summaries are marked in italic, and explained in the last column

Table 12 Examples of errors in abstractive summaries generated by different summarization models for the UK Court judgement available at https://www.supremecourt.uk/cases/docs/uksc-2011-0196-judgment.pdf

id	Model	Extract from summary showing error	Explanation of error
1	Davinci-summ	The Supreme Court has allowed an appeal by a woman against the Serious Organised Crime Agency (SOCA) over the agency's attempt to recover *£ 4.5m* from her	There is no mention of £4.5m in the main document. Rather the main document mentions "US $2.5m". Rather there is a mention of Clause 4.5 in the settlement deed. The model has confused between the clause 4.5 and the amount $2.5m
2	LegLED	The Honorable Lord Neuberger has agreed to the entry of a final judgment that permanently enjoins him from future violations of the antifraud provisions of *Section 17(a) of the Securities Act of 1933 ("Securities Act")*...	The Section 17(a) of the Securities Act of 1933 is hallucinated. No such Act is present in the main judgement. This is, in fact, an US Act whereas the input case is of the UK Supreme Court

The errors in the summaries are marked in italic, and explained in the last column

hallucinated, incomplete statute names – the abstractive summarizers and LLMs also frequently commit smaller errors such as two words being merged or two sentences being merged (a sentence beginning without the previous one ending).

8 Exploring ways to reduce inconsistencies in abstractive summaries

From the previous section, it is evident that hallucinations / inconsistencies are observed in the abstractive summaries of legal judgements. We now explore three different strategies for reducing hallucinations in abstractive summaries, one for abstractive summarizers (that allow fine-tuning) and two methods for LLMs (for which fine-tuning is not possible or prohibitively expensive).

8.1 Domain-specific fine-tuning of abstractive models

As stated earlier, the abstractive summarization models LegPegasus and LegLED are originally fine-tuned over legal (case judgement, summary) pairs from courts in the USA. These models allow further fine-tuning. Hence we check if domain-specific fine-tuning (i.e., fine-tuning over data from the target domain) of these models helps in improving consistency of the generated summaries.

To make these models more suitable for summarizing UK legal documents, we further fine-tune them on the UK-Abs training dataset containing 693 UK case judgements and their corresponding reference summaries. These models fine-tuned on the UK-Abs training dataset are referred to as **LegPegasus-UK** and **LegLED-UK**. Similary, to make these models more suitable for Indian case judgements, LegLED and LegPegasus models are further finetuned on the IN-Abs training

Table 13 Examples of errors in abstractive summaries generated by different summarization models for the UK Court judgement available at https://www.supremecourt.uk/cases/docs/uksc-2015-0063-judgment.pdf

id	Model	Extract from summary showing error	Explanation of error
1	Chatgpt-tldr	The closest any international instrument has come to provide for such a general immunity is article 11.2(b) of the *United Nations Convention on Jur The* European Court of Human Rights has only considered article 11 in cases.....	The phrase marked in red (actually a statute name) is incomplete and has been merged with the next sentence. The entire phrase as stated in the original judgement is "United Nations Convention on Jurisdictional Immunities of States and their Property (2004)"
2	Davinci-summ	The case was brought by two women, one Moroccan and *one Tunisian*, who sued the embassies of Sudan and Libya respectively....	The original judgement does not have the country 'Tunisia' anywhere in the judgement. This information is hallucinated. The judgement talks about two women both of whom are from Morocco
3	LegLED	The *U.S. District Court for the Western District of New York today entered a final judgement in favor of the Kingdom of Saudi Arabia* in a case involving claims to state immunity	The U.S. District Court is nowhere stated in the original legal judgement. So this entire information is hallucinated

The errors in the summaries are marked in italic and explained in the last column

Table 14 Comparing the performance of abstractive summarization models over the IN-Abs test set, before and after domain-specific fine-tuning

Model	R2-P	R2-R	R2-F1	RL-P	RL-R	RL-F1	ME	BS
LegPegasus	0.196	0.120	0.133	0.263	0.154	0.172	0.194	0.594
LegPegasus-IN	**0.264**	0.243	0.251	**0.281**	0.262	0.269	0.196	**0.624**
LegLED	0.111	0.107	0.108	0.150	0.146	0.147	0.142	0.590
LegLED-IN	0.260	**0.253**	**0.255**	0.276	**0.269**	**0.271**	**0.226**	**0.624**

The higher value for every metric is shown in underline, and the overall highest value for every metric is in bold

Table 15 Consistency metrics for abstractive summarization models over the IN-Abs dataset, before and after domain-specific fine-tuning

Model	SummaC	NEPrec	NumPrec
LegPegasus	0.633	0.842	0.948
LegPegasus-IN	0.736	**0.854**	**0.995**
LegLED	0.656	0.719	0.819
LegLED-IN	**0.855**	0.827	0.976

The higher value for every metric is shown in underline, and the overall highest value for every metric is in bold

Table 16 Comparing the performance of abstractive summarization models over the UK-Abs test set, before and after domain-specific fine-tuning

Model	R2-P	R2-R	R2-F1	RL-P	RL-R	RL-F1	ME	BS
LegPegasus	**0.249**	0.085	0.101	**0.248**	0.113	0.136	0.195	0.635
LegPegasus-UK	0.183	**0.173**	**0.178**	0.217	**0.205**	**0.211**	0.200	**0.640**
LegLED	0.108	0.107	0.107	0.151	0.149	0.150	0.173	0.601
LegLED-UK	0.164	0.155	0.159	0.194	0.182	0.188	**0.208**	0.637

The higher value for every metric is shown in underline, and the overall highest value for every metric is in bold

Table 17 Consistency metrics for legal domain-specific abstractive models for UK-Abs dataset before and after domain-specific fine-tuning

Model	SummaC	NEPrec	NumPrec
LegPegasus	0.649	**0.907**	0.959
LegPegasus-UK	0.708	**0.907**	**0.992**
LegLED	0.628	0.874	0.914
LegLED-UK	**0.835**	0.882	0.989

The higher value for every metric is shown in underline, and the overall highest value for every metric is in bold

dataset consisting of 7030 (case judgement, summary) pairs. We refer to these models fine-tuned over Indian data as **LegLED-IN** and **LegPegasus-IN**.

Table 14 shows the ROUGE, METEOR, and BERTScore for legal domain-specific abstractive models on the IN-Abs dataset before (LegPegasus and LegLED) and after (LegPegasus-IN and LegLED-IN) using fine-tuning. Table 15 shows the consistency metrics over IN-Abs for the same scenarios. It is seen that both the match of the generated summaries with reference summaries (Table 14) as well as the consistency of generated summaries (Table 15) improve after fine-tuning over data from the target domain.

Similarly, Tables 16 and 17 compare the performance of the abstractive summarization models on the UK-Abs dataset, before and after fine-tuning over target domain data. We again see that performance as well as consistency of the generated summaries improves massively upon fine-tuning over the target domain data.

Recall that when LegLED was used to summarize Indian or UK judgements, there were several instances of US court and statute names being hallucinated in the summaries (e.g., the examples stated in Tables 12 and 13). Importantly, we did *not* observe such instances in the summaries generated by LegLED-IN or LegLED-UK. While the summaries generated by LegLED-IN or LegLED-UK also contain some issues such as words or sentences being merged together, we did not observe such gross hallucinations in the summaries generated by these fine-tuned versions.

Thus, for both datasets IN-Abs and UK-Abs, it is seen that fine-tuning over the target domain data helps to improve the quality as well as consistency of the abstractive summaries. If data in the target domain is available, it seems an effective approach to utilize such data for fine-tuning abstractive summarizers.

8.2 Suitable prompting of LLMs

Unlike the abstractive summarization models described previously, LLMs are very difficult/expensive to fine-tune. Hence, avoiding hallucinations in LLMs is much more challenging and is, in fact, an open question (Huang et al. 2023).

In this work, we explore a very simple approach to reducing hallucinations in LLMs – we include in the prompt an instruction to avoid hallucinations / inconsistencies. Also, since we observed the presence of incomplete sentences in the generated summaries, we included an instruction to output complete sentences only. Specifically, we add the following text in the prompt – "*Output complete sentences and not half sentences. Do not have hallucinations and inconsistencies in your summary.*"

We selected those variations of the LLMs that performed the best on the two datasets (e.g., chatgpt-explicit, davinci-summ, and chatgpt-16k-long for the UK-Abs dataset), and created new variations by adding the above-mentioned text in the prompt. For instance, we got a new variation from chatgpt-explicit, which we call **Chatgpt-reduce-hallucination** (abbreviated as **chatgpt-RH**, by using the prompt: "Your task is to summarize the following document in at most `<YY>` words. Output complete sentences and not half sentences. Do not have hallucinations and inconsistencies in your summary. The document to be summarized is given within <>. Document to summarize - `<text to summarize>`". Similarly, we got the variations **Davinci-reduce-hallucination** (abbreviated as **davinci-RH**) from davinci-summ, and **chatgpt-16k-RH-hallucination**

Table 18 ROUGE, METEOR, BERTScore metrics for general-domain LLMs, before and after we use prompting to reduce hallucinations, over the IN-Abs dataset

Model	R2-P	R2-R	R2-F1	RL-P	RL-R	RL-F1	ME	BS
Chatgpt (Chunk size 1024)								
chatgpt-summ	0.199	0.177	0.186	0.232	**0.209**	0.214	0.193	0.624
chatgpt-RH	0.151	0.116	0.128	0.225	0.169	0.188	**0.197**	0.540
Davinci (Chunk size 1024)								
davinci-summ	0.220	**0.179**	**0.195**	0.251	0.205	**0.223**	0.191	0.624
davinci-RH	0.161	0.146	0.151	0.205	0.140	0.167	0.187	**0.630**
Chatgpt-16k (Chunk size 8192)								
chatgpt-16k-long	**0.231**	0.127	0.149	**0.319**	0.176	0.206	0.147	0.615
chatgpt-16k-RH	0.143	0.106	0.118	0.216	0.159	0.178	0.176	0.568

The higher value for every metric is shown in underline, and the overall highest value for each metric is shown in bold

Table 19 Consistency metrics for general-domain LLMs before and after we use prompting to reduce hallucinations, over the IN-Abs dataset

Model	SummaC	NEPrec	NumPrec
Chatgpt (Chunk size 1024)			
chatgpt-summ	0.573	0.901	0.956
chatgpt-RH	0.600	0.917	0.979
Davinci (Chunk size 1024)			
davinci-summ	0.635	0.895	0.932
davinci-RH	0.645	0.922	0.964
Chatgpt-16k (Chunk size 8192)			
chatgpt-16k-long	0.633	0.944	0.975
chatgpt-16k-RH	**0.658**	**0.955**	**0.989**

The higher value for every metric is shown in underline, and the overall highest value for every metric is in bold

Table 20 ROUGE, METEOR, BERTScore metrics for general-domain LLMs before and after we use prompting to reduce hallucinations, over the UK-Abs dataset

Model	R2-P	R2-R	R2-F1	RL-P	RL-R	RL-F1	ME	BS
Chatgpt (Chunk size 1024)								
chatgpt-explicit	0.164	0.163	0.163	0.214	**0.210***	0.211	0.218	**0.640**
chatgpt-RH	0.169	0.125	0.140	**0.245**	0.188	0.209	0.170	0.616
Davinci (Chunk size 1024)								
davinci-summ	**0.196**	**0.181**	**0.187**	0.223	0.206	**0.213**	**0.219**	0.638
davinci-RH	0.186	0.114	0.131	0.206	0.188	0.194	0.176	0.617
Chatgpt-16k (Chunk size 8192)								
chatgpt-16k-long	0.153	0.144	0.148	0.213	0.199	0.205	0.183	0.619
chatgpt-16k-RH	0.159	0.126	0.135	0.215	0.139	0.160	0.155	0.588

The higher value for every metric is shown in underline, and the overall highest value for every metric is in bold

Table 21 Consistency metrics for general-domain LLMs before and after we use prompting to reduce hallucinations, over the UK-Abs dataset

Model	SummaC	NEPrec	NumPrec
Chatgpt (Chunk size 1024)			
chatgpt-explicit	0.622	0.900	0.952
chatgpt-RH	<u>0.634</u>	<u>0.947</u>	<u>0.981</u>
Davinci (Chunk size 1024)			
davinci-summ	0.634	0.926	0.977
davinci-RH	<u>0.635</u>	<u>0.932</u>	<u>0.984</u>
Chatgpt-16k (Chunk size 8192)			
chatgpt-16k-long	0.644	0.956	0.981
chatgpt-16k-RH	**0.654**	**0.967**	**0.988**

The higher value for every metric is shown in underline, and the overall highest value for every metric is in bold

(abbreviated as **chatgpt-16k-RH**) from chatgpt-16k-long, by including the explicit instructions for reducing hallucinations and outputing complete sentences.

Analysing the effect of the Semantic similarity based approach to reduce hallucination We now compare the quality of summaries generated by the SS variations of the LLMs with that of the summaries generated by the variations that achieved the best result in Sect. 6 (e.g., chatgpt-summ with chatgpt-SS, davinci-summ with davinci-SS, gpt4-summ with gpt4-SS).

For the IN-Abs dataset, Table 18 shows the ROUGE, METEOR, and BERTScore metrics for the different variations of the LLMs while Table 19 shows the consistency metrics for the different variations. Similarly, for the UK-Abs dataset, Table 20 shows the ROUGE, METEOR, and BERTScore metrics while Table 21 shows the consistency metrics for the different versions of the LLMs. Across both datasets, we find that the RH versions score *lower* than the 'explicit' or 'summ' versions of the LLMs, according to most ROUGE-based, METEOR and BertScore metrics. Whereas, the RH versions score higher according to the consistency metrics (SummaC, NEPrec, NumPrec).

We manually examined several summaries generated by the RH variants and the corresponding summaries by the 'summ', 'long' and 'explicit' variants, to understand the reasons behind the fall in the summary quality of the RH variants. We observed that the summaries generated by the RH-variants have lesser amount of key information, such as named entities (names of persons, organizations, etc.) in the case document, as compared to those generated by the other variants. For instance, the average number of named entities in the summaries generated by chatgpt-RH is 13.9, while that in the summaries generated by chatgpt-summ is 16.5. The summaries generated by the RH-variant have lesser amounts of unigram, bigram, longest common subsequence, and semantic similarity matches with the gold standard summaries as compared to the 'summ' variant. Since there is lesser amount of key information contained in the summaries generated by the RH variants, the ROUGE, METEOR, and BERTScore of these summaries are lower than those for the other variants.

These observations imply that, while it is possible to reduce hallucinations / inconsistencies in the generated summaries by explicit prompting, there may be an associated reduction in the quality of the summary. A future direction of research would be to devise strategies for reducing hallucinations and inconsistencies in LLM-generated legal summaries, while maintaining the information quality of the summaries.

8.3 Semantic similarity based approach for reducing hallucinations

We now describe a novel approach that uses semantic similarity between entities to reduce hallucinations and inconsistencies in summaries. Given a legal judgement j and its summary s generated by an AI model, the approach consists of the following steps:- (i) Consider all the named entities and numbers present in the legal judgement j. Let the set of named entities and numbers present in j be V_j (the set of named entities and numbers can be detected by any standard named entity recognizer; in particular, we use the popular Python Spacy library). (ii) Consider all the named entities and numbers present in the summary s generated by the AI model. Let the set of named entities and numbers present in the summary be defined as V_s. (iii) Now define a set $V_r = V_s - V_j$. V_r is the set of all those named entities / numbers that are present in the generated summary but are *not* present in the original judgement (hallucinated entities). (iv) For every element in the set V_r, consider an embedding/representation from any pre-trained language model (specifically, we used the bert-base-uncased embeddings). Also, for every element in the set V_j, consider the same embedding. (v) Calculate the cosine similarity between the embeddings of every element in V_r and every element in V_j. (vi) Replace every element in the set V_r with that element in V_j which has the highest cosine similarity with this element, based on the embeddings (numbers to be replaced by numbers only). Basically, we replace every hallucinated named entity or number (in the generated summary) with the most semantically similar entity that is present in the input document.

The chatgpt variant which uses this semantic similarity based approach to reduce hallucinations is denoted as chatgpt-SS. Similarly, we denote the Davinci and chatgpt-16k variant which uses the semantic similarity based approach to reduce hallucinations as davinci-SS and chatgpt-16k-SS respectively.

Table 22 shows examples of two summary extracts with hallucinated named entities / numbers replaced by the correct entities using the semantic similarity based method. The first example shows the wrong monetary amount ‘Rs. 26,500’ replaced by ‘Rs. 29,500’. The second example shows that a wrong judge name is replaced by the correct judge name using the semantic similarity based method.

Analysing the effect of semantic similarity to reduce hallucination We now compare the quality of summaries generated by the SS variations of the LLMs with that of the summaries generated by the variations that achieved the best result in Sect. 6 (e.g., chatgpt-summ with chatgpt-SS, davinci-summ with davinci-SS, chatgpt-16k-long with chatgpt-16k-SS). For the IN-Abs dataset, Table 23 shows the ROUGE, METEOR, and BERTScore metrics for the different variations of the LLMs while Table 24 shows the consistency metrics for the different variations. Similarly, for

Table 22 Examples of errors in abstractive summaries generated by different summarization models for the Indian Court judgement available at https://indiankanoon.org/doc/1801104/

id	Model	Extract from summary showing hallucinated named entity / number	Extract from corrected summary
1	davinci-summ	The prosecution asserts that the accused initially demanded Rs. 30,000 later reduced to Rs. *26,500* for granting the complainant vacant possession of a flat	The prosecution asserts that the accused initially demanded Rs. 30,000 later reduced to Rs. *29,500* for granting the complainant vacant possession of a flat
2	chatgpt-summ	On February 1, 2020, the Honorable *Chandrasekhar A. Lama* of India entered a final judgment......	On February 1, 2020, the Honorable *Aiyar, N. Chandrasekhara* of India entered a final judgment......

The third column shows the summary extract with the hallucinated named entity / number in red-colored font. The fourth column shows the same summary extract corrected by the semantic similarity based method, with the corrected named entity / number in italic

Table 23 ROUGE, METEOR, BERTScore metrics for general-domain LLMs, on using semantic similarity-based approach to reduce hallucinations, over the IN-Abs dataset

Model	R2-P	R2-R	R2-F1	RL-P	RL-R	RL-F1	ME	BS
Chatgpt (Chunk size 1024)								
chatgpt-summ	0.199	0.177	0.186	0.232	0.209	0.214	0.193	0.624
chatgpt-SS	0.206	0.184	0.192	0.239	**0.227**	0.230	**0.197**	**0.636**
Davinci (Chunk size 1024)								
davinci-summ	0.220	0.179	0.195	0.251	0.205	0.223	0.191	0.624
davinci-SS	0.216	**0.185**	**0.198**	0.242	0.213	0.225	0.196	0.633
Chatgpt-16k (Chunk size 8192)								
chatgpt-16k-long	0.231	0.137	0.149	0.319	0.176	0.206	0.154	0.615
chatgpt-16k-SS	**0.242**	0.132	0.155	**0.329**	0.182	**0.234**	0.147	0.624

The higher value for every metric is shown in underline, and the overall highest value for each metric is shown in bold

Table 24 Consistency metrics for general-domain LLMs on using semantic similarity based approach to reduce hallucinations, over the IN-Abs dataset

Model	SummaC	NEPrec	NumPrec
Chatgpt (Chunk size 1024)			
chatgpt-summ	0.573	0.901	0.956
chatgpt-SS	0.605	0.929	0.984
Davinci (Chunk size 1024)			
davinci-summ	0.635	0.895	0.932
davinci-SS	0.647	0.932	0.972
Chatgpt-16k (Chunk size 8192)			
chatgpt-16k-long	0.633	0.944	0.975
chatgpt-16k-SS	**0.659**	**0.958**	**0.993**

The higher value for every metric is shown in underline, and the overall highest value for every metric is in bold

Table 25 ROUGE, METEOR, BERTScore metrics for general-domain LLMs on using semantic similarity based approach to reduce hallucinations, over the UK-Abs dataset

Model	R2-P	R2-R	R2-F1	RL-P	RL-R	RL-F1	ME	BS
Chatgpt (Chunk size 1024)								
chatgpt-explicit	0.164	0.163	0.163	0.214	0.210*	0.211	0.218	0.640
chatgpt-SS	0.166	0.173	0.167	0.223	**0.215***	0.218	0.228	**0.653**
Davinci (Chunk size 1024)								
davinci-summ	0.196	0.181	0.187	0.223	0.206	0.213	0.219	0.643
davinci-SS	**0.209**	**0.194**	**0.192**	**0.236**	0.212	**0.220**	**0.231**	0.638
Chatgpt-16k (Chunk size 8192)								
chatgpt-16k-long	0.153	0.146	0.148	0.213	0.199	0.205	0.183	0.619
chatgpt-16k-SS	0.164	0.145	0.158	0.226	0.201	0.209	0.194	0.625

The higher value for every metric is shown in underline, and the overall highest value for every metric is in bold

Table 26 Consistency metrics for general-domain LLMs on using semantic similarity based approach to reduce hallucinations, over the UK-Abs dataset

Model	SummaC	NEPrec	NumPrec
Chatgpt (Chunk size 1024)			
chatgpt-explicit	0.622	0.900	0.952
chatgpt-SS	<u>0.635</u>	<u>0.957</u>	<u>0.986</u>
Davinci (Chunk size 1024)			
davinci-summ	0.634	0.926	0.977
davinci-SS	<u>0.636</u>	<u>0.937</u>	<u>0.988</u>
Chatgpt-16k (Chunk size 8192)			
chatgpt-16k-long	0.644	0.956	0.981
chatgpt-16k-SS	**0.658**	**0.969**	**0.992**

The higher value for every metric is shown in underline, and the overall highest value for every metric is in bold

the UK-Abs dataset, Table 25 shows the ROUGE, METEOR, and BERTScore metrics while Table 26 shows the consistency metrics for the different versions of the LLMs. For both datasets, we observe an interesting trend. The SS variations not only achieve higher scores for the consistency metrics (e.g., chatgpt-SS has higher SummaC, NEPrec and NumPrec scores than chatgpt-summ in Table 24), but also mostly score higher in terms of the ROUGE, METEOR and BERTScore metrics, as compared to the other variations (e.g., chatgpt-SS has higher scores than chatgpt-summ as per most metrics in Table 23).

These observations imply that, there is a reduction in hallucinations / inconsistencies in the generated summaries by this approach, and there is also an improvement in terms of results of the automated metrics. Hence, the semantic similarity-based approach is quite effective in improving the quality of generated summaries. However, we have noticed that all hallucinations / inconsistencies cannot be corrected by this approach; hence the correction of hallucination/inconsistency in abstractive summaries remains an open area of research.

9 Summarization of other types of legal documents

Till now, we have applied all summarization models over two datasets of legal case judgements. In this section, we check how well the models perform over other types of legal documents. For this, we use the GOVREPORT dataset (Huang et al. 2021).

9.1 GOVREPORT dataset

The GOVREPORT dataset, obtained from Huang et al. (2021), comprises of 19,466 detailed reports from the U.S. Government Accountability Office (GAO) and the Congressional Research Service (CRS). These reports, prepared in response to congressional requests, span a broad range of national policy topics and include human-written summaries. During data collection, some boiler-plate parts were removed from the crawled files, and the section and paragraph structure of the documents and

Table 27 Dataset statistics for GOVREPORT dataset

	Nos. of documents	Average nos. of words per document	Average nos. of words per gold-standard summary
Train set	17519	9407.17	556.75
Test set	973	9409.45	540.48
Validation set	974	9417.19	545.04

To get the average number of words per document (or summary), we first calculate the number of words in every document (summary). Then we add up the number of words from all documents (summaries), and divide the total word-count by the number of documents (summaries)

Table 28 ROUGE, METEOR, BERTScore metrics for best performing general-domain LLMs, abstractive and extractive summarization models on the GOVREPORT dataset

Model	R2-P	R2-R	R2-F1	RL-P	RL-R	RL-F1	ME	BS
General domain LLMs								
gpt4-summ	0.386	0.169	0.205	0.453	0.292	0.326	0.205	0.627
chatgpt-summ	<u>0.395</u>	**0.172**	**0.221**	<u>0.461</u>	**0.302**	**0.346**	<u>0.215</u>	**0.629**
llama-summ	0.369	0.148	0.179	0.433	0.283	0.317	0.202	0.602
Legal domain-specific abstractive models								
LegPegasus-GR	**0.408**	<u>0.149</u>	<u>0.193</u>	**0.481**	<u>0.215</u>	<u>0.304</u>	**0.217**	<u>0.618</u>
LegLED-GR	0.382	0.147	0.184	0.460	0.204	0.298	0.205	0.608
Extractive models								
CaseSummarizer	0.351	0.126	0.183	0.431	0.188	0.296	0.201	0.604

The value shown in underline is the best value in every summarization family for every metric. The value shown in bold is the highest value for every metric

summaries were retained. The dataset features 12,228 GAO reports and 7,238 CRS reports. On average, GAO reports have 6.9 sections, while CRS reports have 4.6 sections. The dataset is organized into training, validation, and test sets based on the publication dates, resulting in 17,519 training samples, 974 validation samples, and 973 test samples. Table 27 shows the dataset statistics for the GOVREPORT dataset. Following Grusky et al. (2018); Shen et al. (2022), the coverage and density of the legal judgements with respect to the gold-standard summaries is 0.16 and 1.40 respectively for this dataset.

In this work, the summarization models are evaluated over a randomly sampled 100 document-summary pairs from the test set of GOVREPORT. The supervised summarization models are trained over the training set of GOVREPORT.

9.2 Summarization results on GOVREPORT dataset

We now study the performances of summarization models on the GOVREPORT dataset. To this end, we report the results of some of the best performing LLMs and

Table 29 Consistency metrics for different abstractive summarization models on GOVREPORT dataset

Model	SummaC	NEPrec	NumPrec
General domain LLMs			
gpt4-summ	<u>0.599</u>	**0.952**	<u>0.984</u>
chatgpt-summ	0.579	0.927	0.946
llama-summ	0.564	0.872	0.873
Legal Domain-specific abstractive models			
LegPegasus-GR	**0.617**	<u>0.935</u>	**0.986**
LegLED-GR	0.603	0.904	0.973

The value shown in underline is the best value in every summarization family for every metric. The value shown in bold is the highest value for every metric

summarization models (as observed in the previous sections of this paper). From the GPT-4 Turbo variants, we report the results of the best performing variant gpt4-summ. Similarly, from the Llama2-70b and Chatgpt variants, we report the results of llama-summ and chatgpt-summ. We also report the results of LegPegasus and LegLED fine-tuned on the training dataset of GOVREPORT; we denote these models as LegPegasus-GR and LegLED-GR. We also report the results of best performing extractive summarization model namely CaseSummarizer.

Comparing general-domain LLMs with domain-specific abstractive summarizers and extractive models Table 28 shows the ROUGE, METEOR, BERTScore metrics for the aforementioned summarization models on the GOVREPORT dataset. Among the legal domain-specific abstractive models, LegPegasus-GR performs the best. Among the General domain LLMs, chatgpt-summ performs the best for most automated metrics. We see that both LLMs and legal domain-specific abstractive models perform much better than the extractive CaseSummarizer over this dataset. This is somewhat expected, since CaseSummarizer is specifically meant for summarizing legal case judgements and not other types of legal documents.

In general, we see higher metric values, especially for the ROUGE scores, for GOVREPORT than what we obtained for the IN-Abs and UK-Abs datasets. These trends seem to indicate that summarizing legal case judgement reports is a more complicated task than summarizing the Government reports in the GOVREPORT dataset.

Consistency of generated summaries Table 29 shows the consistency metrics for summaries generated by different abstractive models on the GOVREPORT dataset. The value shown in blue is the best value in every summarization family for every metric. The value shown in blue-bold is the overall highest value for every metric. gpt4-summ gets the highest NEPrec score, while LegPegasus-GR achieves the highest SummaC and NumPrec scores.

10 Human evaluation of summaries

In this final section, we perform a human evaluation of the summaries generated by some of the best-performing models. For this, we consulted three senior Law students from the Rajiv Gandhi School on Intellectual Property Law, a reputed Law school in India. Due to their limited availability, we considered only the five best-performing summarization models for each dataset. Specifically, we considered the best-performing model from every summarization family.

For the IN-Abs dataset, we considered chatgpt-summ (the best-performing model amongst Chatgpt-1024 models), davinci-summ (the best-performing model amongst Davinci-1024 models), chatgpt-16k-long (the best-performing model amongst Chatgpt-16k models), CaseSummarizer (the best-performing extractive model), and LegLED-IN (the best-performing model amongst legal domain-specific abstractive summarization models). For the UK-Abs dataset, we considered chatgpt-explicit (the best-performing model amongst Chatgpt-1024 models), davinci-summ (the best-performing model amongst Davinci-1024 models), chatgpt-16k-long (the best-performing model amongst Chatgpt-16k models), CaseSummarizer (the best-performing extractive model), and LegPegasus-UK (the best-performing model amongst legal domain-specific abstractive models). For the GOVREPORT dataset, we considered for human evaluation chatgpt-summ, llama-summ, gpt4-summ, LegPegasus-GR, and CaseSummarizer.

Metrics for human evaluation We used the following four metrics for the human evaluation of the summaries:-

- *Informativeness* This metric measures how much relevant information the summary contains from the source document. The annotators were asked to assess if the summary captures the essential points of the original document. A higher value of informativeness means a higher-quality summary.
- *Redundancy* This metric measures how much information is repeated in the summary. Redundancy can negatively impact the quality of a summary. A *lower* value of redundancy means a higher-quality summary.
- *Factuality* This metric determines if the information presented in the summary is factually accurate. The annotators were asked to compare the summary content with the source document to assess accuracy. A higher value of factuality means a higher-quality summary.
- *Coherence* This metric assesses if the sentences in the summary are logically connected and form a coherent and understandable narrative. A higher value of Coherence means a higher-quality summary.

Method of human evaluation From each dataset, we considered 25 randomly selected documents, and their summaries generated by the 5 models stated above. We asked the human annotators to give a score between 1 and 5 to every model-generated summary, for each of the above metrics. Each annotator was asked to evaluate the summaries independently, i.e., without discussing with the others. Also, the annotators were *not* told which summary was generated by which

Table 30 Human evaluation scores on UK-Abs dataset calculated by averaging the scores provided by three annotators for 25 summaries for every model

Model	Informativeness	Redundancy	Factuality	Coherence
chatgpt-explicit	3.34	1.65	3.68	3.26
davinci-summ	**3.67**	1.53	3.73	3.22
chatgpt-16k-long	3.01	**1.28**	**4.06**	**3.67**
CaseSummarizer	3.09	1.54	3.62	3.13
LegPegasus-UK	3.53	1.61	3.90	3.44

The highest scores for Informativeness, Factuality, and Coherence, and the lowest score for Redundancy are shown in bold

Table 31 Human evaluation scores on IN-Abs dataset calculated by averaging the scores provided by three annotators for 25 summaries for every model

Model	Informativeness	Redundancy	Factuality	Coherence
chatgpt-summ	3.34	1.66	3.46	3.26
davinci-summ	3.46	1.60	3.49	3.48
chatgpt-16k-long	2.98	**1.30**	**4.20**	**3.89**
CaseSummarizer	3.54	1.74	3.64	3.52
LegLED-IN	**3.61**	1.68	3.85	3.61

The highest scores for Informativeness, Factuality, and Coherence, and the lowest score for Redundancy are shown in bold

Table 32 Human evaluation scores on GOVREPORT dataset calculated by averaging the scores provided by three annotators for 25 summaries for every model

Model	Informativeness	Redundancy	Factuality	Coherence
chatgpt-summ	3.25	1.60	3.43	3.26
llama-summ	3.16	1.56	3.40	3.04
gpt4-summ	2.94	**1.35**	**3.89**	**3.94**
CaseSummarizer	3.10	1.61	3.26	3.10
LegPegasus-GR	**3.56**	1.62	3.56	3.47

The highest scores for Informativeness, Factuality, and Coherence, and the lowest score for Redundancy are shown in bold

model, in order to ensure an unbiased evaluation. Through this evaluation, a particular summarization model got 75 scores (scores provided to model-generated summaries for 25 documents, by 3 annotators) for every metric, and then we take the mean/average of the 75 scores. We report these average scores for each metric, separately for the three datasets.

Inter-annotator agreement We measure the inter-annotator agreement scores in terms of Fleiss Kappa over the scores given by the human annotators, over all the datasets combined. The Fleiss Kappa for Informativeness, Redundancy, Factuality and Coherence come out to be 0.54, 0.65, 0.42, and 0.56 respectively. The Fleiss Kappa scores for Informativeness, Factuality, and Coherence represent moderate agreement while the Fleiss Kappa score for Redundancy represents substantial agreement (Landis and Koch 1977).

Results of human evaluation Table 30 shows the human evaluation scores for the UK-Abs dataset, Table 31 for the IN-Abs dataset, and Table 32 shows the human evaluation scores for the GOVREPORT dataset. For IN-Abs and UK-Abs datasets, the summaries generated by chatgpt-16k-long have been given the best scores by the annotators, in terms of Redundancy, Factuality and Coherence. In terms of Informativeness, davinci-summ has the highest score for UK-Abs while LegLED-IN gets the highest score for IN-Abs. For GOVREPORT dataset, gpt4-summ has the best scores in terms of redundancy, factuality, and coherence, while LegPegasus-GR has the highest informativeness.

These scores reflect the trade-off in breaking up a long legal document into chunks and independently summarizing every chunk (which is the approach for every other summarization model, other than chatgpt-16k-long, and gpt4-summ). The chunking approach potentially leads to redundancy (e.g., if the same information comes into the summary from multiple chunks) and lack of coherence at the chunk-boundaries. In contrast, chatgpt-16k-long (with a chunk size of 8,192) and GPT4 can accommodate most case judgements and government reports as a whole, and hence can generate summaries with lesser redundancy and higher coherence. On the other hand, informativeness of the summaries are better with the chunking strategy, since more information from each chunk can be accommodated in the summaries. This is possibly why davinci-summ, LegLED-IN and LegPegasus-GR are given the highest scores for Informativeness by the Law students.

11 Concluding discussion

To our knowledge, this is the first work that systematically compares the performances of three different families of models for the practical and challenging task of legal case judgement summarization – (1) traditional extractive summarization models, (2) domain-specific abstractive summarization models (Legal-Pegasus and Legal-LED), and (3) general domain LLMs such as ChatGPT, Davinci, Llama2-70b and GPT-4-Turbo. We conduct comprehensive experiments over three datasets from the Indian, UK Supreme Courts and GOVREPORT dataset. Our experiments lead to the following insights.

- Abstractive models and LLMs generally outperform extractive models in terms of both quantitative metrics as well as human evaluation. In particular, LLMs like Text-Davinci-003, Turbo-GPT-3.5, Llama-70b and GPT4 perform well even without specific legal document training. However, generative models often contain hallucinations and inconsistencies in the generated summaries.

- Fine-tuning abstractive models with the target domain data, if available, helps in reducing hallucinations/inconsistencies as well as improves the quality of the summaries (as seen on both UK-Abs and IN-Abs datasets).
- Suitable prompting of LLMs can help in reducing hallucinations/inconsistencies in the generated summaries. But there may be an associated reduction of the summary quality. An important future direction of research would be to reduce inconsistencies in LLM-generated summaries while maintaining the summary quality.
- We discussed a semantic similarity-based approach to reduce hallucinations in the summaries generated by the LLMs. However, complex errors, such as confusion between names or numbers are difficult to detect or prevent completely.
- The extreme length of legal case judgements is a domain-specific challenge. There is a trade-off associated with the approach of breaking these long documents into chunks. As seen from the human evaluation, and as already demonstrated in Moro and Ragazzi (2022), chunking leads to better information quality in the summaries, but also leads to redundancy and lack of coherence.

Based on these results, we conclude that, for complex domains like law, LLMs and pre-trained abstractive summarization models are not ready yet for fully automatic deployment. A human-in-the-loop approach, where a legal expert monitors the generated summaries, may be more appropriate. Furthermore, better methods are needed to detect complex errors in abstractive summaries. We plan to explore these directions in the future.

The present work has certain limitations. We have used a basic segmentation technique to manage lengthy documents, which has certain limitations. First, the token-level segmentation strategy can cause the last sentence of a chunk to end abruptly, potentially affecting readability. Second, within a single chunk, sentences might address different topics, which can reduce the coherence of the chunks. While this chunking aspect is not the main focus of the present study, it is important to acknowledge that the quality of the chunks significantly influences the final findings and conclusions of the work. An important future direction of research is to explore better segmentation strategies for handling long legal documents.

Acknowledgements The authors acknowledge the anonymous reviewers whose comments greatly helped to improve the paper, and the Law students who helped with the human evaluation of the summaries. The work is partially supported by the Technology Innovation Hub (TIH) on AI for Interdisciplinary Cyber-Physical Systems (AI4ICPS) set up by IIT Kharagpur under the aegis of DST, Government of India (GoI).

References

Ahmad M, Yaramic I, Roy TD (2023) Creating trustworthy llms: dealing with hallucinations in healthcare ai

Ahmed T, Devanbu P (2023) Few-shot training llms for project-specific code-summarization. In: Proceedings of the 37th IEEE/ACM international conference on automated software engineering

Banerjee S, Lavie A (2005) Meteor: an automatic metric for mt evaluation with improved correlation with human judgments. In: Proceedings of the ACL workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization, pp 65–72

Beltagy I, Peters ME, Cohan A (2020) Longformer: the long-document transformer. arXiv preprint arXiv:2004.05150

Bhattacharya P, Hiware K, Rajgaria S, Pochhi N, Ghosh K, Ghosh S (2019) A comparative study of summarization algorithms applied to legal case judgments. In: Proceedings of european conference on information retrieval (ECIR), pp. 413–428

Bhattacharya P, Poddar S, Rudra K, Ghosh K, Ghosh S (2021) Incorporating domain knowledge for extractive summarization of legal case documents. In: Proceedings of international conference on artificial intelligence and law (ICAIL), pp 22–31

Chang Y, Lo K, Goyal T, Iyyer M (2023) Booookscore: a systematic exploration of book-length summarization in the era of llms. arXiv preprint arXiv:2310.00785

Deroy A, Maity S (2021) Multi-label classification of covid-tweets using large language models

Deroy A, Maity S (2023) Questioning biases in case judgment summaries: legal datasets or large language models? arXiv preprint arXiv:2312.00554

Deroy A, Bhattacharya P, Ghosh K, Ghosh S (2021) An analytical study of algorithmic and expert summaries of legal cases. In: Professional of international conference on legal knowledge and information systems (JURIX), pp 90–99

Deroy A, Ghosh K, Ghosh S (2023) Ensemble methods for improving extractive summarization of legal case judgements. Artif Intell Law 32:231–289

Deroy A, Ghosh K, Ghosh S (2023a) How ready are pre-trained abstractive models and llms for legal case judgement summarization? In: Proceedings of the international workshop on artificial intelligence and intelligent assistance for legal professionals in the digital workplace (LegalAIIA)

Deroy A, Maity S, Ghosh S (2023b) Prompted zero-shot multi-label classification of factual incorrectness in machine-generated aummaries. In: FIRE pp. 734–746

Deroy A, Bailung NK, Ghosh K, Ghosh S, Chakraborty A (2024) Artificial intelligence (AI) in legal data mining. arXiv:2405.14707

Dimarco MH (2023) Llm-based comment summarization and topic matching for videos

Dong Y, Mircea A, Cheung JCK (2021) Discourse-aware unsupervised summarization for long scientific documents. In: Proceedings of the 16th conference of the european chapter of the association for computational linguistics: main volume, pp 1089–1102

Feijo DDV, Moreira VP (2023) Improving abstractive summarization of legal rulings through textual entailment. Artif Intell Law 31(1):91–113

Grusky M, Naaman M, Artzi Y (2018) Newsroom: a dataset of 1.3 million summaries with diverse extractive strategies. In: Proceedings of the 2018 conference of the north american chapter of the association for computational linguistics: human language technologies, Volume 1 (Long Papers), pp 708–719

Huang L, Cao S, Parulian N, Ji H, Wang L (2021) Efficient attentions for long document summarization. In: Proceedings of the 2021 conference of the north american chapter of the association for computational linguistics: human language technologies, pp 1419–1436

Huang L, Yu W, Ma W, Zhong W, Feng Z, Wang H, Chen Q, Peng W, Feng X, Qin B, Liu T (2023) A survey on hallucination in large language models: principles. Challenges, and Open Questions, Taxonomy

Ji Z, Lee N, Frieske R, Yu T, Su D, Xu Y, Ishii E, Bang YJ, Madotto A, Fung P (2023) Survey of hallucination in natural language generation. ACM Comput Surv 55(12):1–38

Laban P, Schnabel T, Bennett PN, Hearst MA (2022) SummaC: re-visiting NLI-based models for inconsistency detection in summarization. Trans Assoc Comput Linguist 10:163–177

Landis JR, Koch GG (1977) The measurement of observer agreement for categorical data. Biometrics pp 159–174

Lin C-Y (2004) ROUGE: a package for automatic evaluation of summaries. In: Text summarization branches out, pp 74–81

Liu Y (2019) Fine-tune bert for extractive summarization. arXiv preprint arXiv:1903.10318

Maity S, Deroy A, Sarkar S (2023) Harnessing the power of prompt-based techniques for generating school-level questions using large language models. FIRE 2023:30

Maity S, Deroy A, Sarkar S (2024a) A novel multi-stage prompting approach for language agnostic mcq generation using gpt. In: European conference on information retrieval, pp 268–277

Maity S, Deroy A, Sarkar S (2024b) Exploring the capabilities of prompted large language models in educational and assessment applications. arXiv preprint arXiv:2405.11579

Maity S, Deroy A, Sarkar S (2024c) How effective is gpt-4 turbo in generating school-level questions from textbooks based on bloom's revised taxonomy?

Maity S, Deroy A, Sarkar S (2024d) How ready are generative pre-trained large language models for explaining bengali grammatical errors? arXiv preprint arXiv:2406.00039

Moro G, Ragazzi L (2022) Semantic self-segmentation for abstractive summarization of long documents in low-resource regimes. In: Proceedings of the AAAI conference on artificial intelligence, vol. 36, pp 11085–11093

Moro G, Ragazzi L (2023) Align-then-abstract representation learning for low-resource summarization. Neurocomputing 548:126356

Moro G, Ragazzi L, Valgimigli L, Frisoni G, Sartori C, Marfia G (2023) Efficient memory-enhanced transformer for long-document summarization in low-resource regimes. Sensors 23(7):3542

Moro G, Ragazzi L, Valgimigli L et al (2023) Graph-based abstractive summarization of extracted essential knowledge for low-resource scenarios. In: 26th European conference on artificial intelligence, vol. 372, pp 1747–1754

Nallapati R, Zhai F, Zhou B (2017) Summarunner: a recurrent neural network based sequence model for extractive summarization of documents. In: Proceedings of the AAAI conference on artificial intelligence, vol. 31, pp 3075–3081

Nigam SK, Deroy A (2023) Fact-based court judgment prediction. arXiv preprint arXiv:2311.13350

Nigam SK, Deroy A, Shallum N, Mishra AK, Roy A, Mishra SK, Bhattacharya A, Ghosh S, Ghosh K (2023) Nonet at semeval-2023 task 6: methodologies for legal evaluation. In: Proceedings of the 17th international workshop on semantic evaluation (SemEval-2023), pp 1293–1303

Polsley S, Jhunjhunwala P, Huang R (2016) CaseSummarizer: a system for automated summarization of legal texts. In: Proceedings of COLING 2016, the 26th international conference on computational linguistics: system demonstrations, pp 258–262

Saravanan M, Ravindran B, Raman S (2006) Improving legal document summarization using graphical models. Front Artif Intell Appl 152:51

Schneider F, Turchi M (2023) Team zoom@ automin 2023: utilizing topic segmentation and llm data augmentation for long-form meeting summarization. In: Proceedings of the 16th international natural language generation conference: generation challenges, pp 101–107

Shen Z, Lo K, Yu L, Dahlberg N, Schlanger M, Downey D (2022) Multi-lexsum: real-world summaries of civil rights lawsuits at multiple granularities. Adv Neural Inf Process Syst 35:13158–13173

Shukla A, Bhattacharya P, Poddar S, Mukherjee R, Ghosh K, Goyal P, Ghosh S (2022) Legal case document summarization: Extractive and abstractive methods and their evaluation. In: Proceedings of the conference of the asia-pacific chapter of the association for computational linguistics and the international joint conference on natural language processing (Volume 1: Long Papers), pp 1048–1064

Teubner T, Flath CM, Weinhardt C, Aalst W, Hinz O (2023) Welcome to the era of chatgpt et al the prospects of large language models. Bus Inf Syst Eng 65(2):95–101

Zhang T, Kishore V, Wu F, Weinberger KQ, Artzi Y (2019) Bertscore: evaluating text generation with bert. arXiv preprint arXiv:1904.09675

Zhang J, Zhao Y, Saleh M, Liu P (2020) Pegasus: pre-training with extracted gap-sentences for abstractive summarization. In: International conference on machine learning, pp 11328–11339. PMLR

Zhang Y, Ni A, Mao Z, Wu CH, Zhu C, Deb B, Awadallah A, Radev D, Zhang R (2022) Summ^n: a multi-stage summarization framework for long input dialogues and documents. In: Proceedings of the 60th annual meeting of the association for computational linguistics (Volume 1: Long Papers), pp 1592–1604

Zhang T, Ladhak F, Durmus E, Liang P, McKeown K, Hashimoto TB (2023) Benchmarking large language models for news summarization. arXiv preprint arXiv:2301.13848

Zhang S, Wan D, Bansal M (2023) Extractive is not faithful: an investigation of broad unfaithfulness problems in extractive summarization. In: Proceedings of the 61st annual meeting of the association for computational linguistics (Volume 1: Long Papers), pp 2153–2174

Zheng H, Lapata M (2019) Sentence centrality revisited for unsupervised summarization. In: Korhonen A, Traum D, Màrquez L (eds.) Proceedings of the 57th annual meeting of the association for computational linguistics, pp 6236–6247

Zhong L, Zhong Z, Zhao Z, Wang S, Ashley KD, Grabmair M (2019) Automatic summarization of legal decisions using iterative masking of predictive sentences. In: Proceedings of the seventeenth international conference on artificial intelligence and law (ICAIL), pp 163–172