A Transformer-based Approach for Abstractive Summarization of Requirements from Obligations in Software Engineering Contracts

Chirag Jain
Data and Decision Sciences
TCS Research
Pune, India
chirag.rjain3@tcs.com

Preethu Rose Anish
Data and Decision Sciences
TCS Research
Pune, India
preethu.rose@tcs.com

Amrita Singh
Data and Decision Sciences
TCS Research
Pune, India
s.amrita3@tcs.com

Smita Ghaisas
Data and Decision Sciences
TCS Research
Pune, India
smita.ghaisas@tcs.com

2023 IEEE 31st International Requirements Engineering Conference (RE) | 979-8-3503-2689-5/23/$31.00 ©2023 IEEE | DOI: 10.1109/RE57278.2023.00025

***Abstract*—Software Engineering (SE) contracts are a valuable source of software requirements. Seed requirements derived from SE contracts can provide a starting point to the Requirements Engineering (RE) phase. To extract such a seed however, a correct interpretation of contracts text is crucial. A major challenge with contracts text interpretation is that the text is lengthy, convoluted, and it incorporates a complex Legalese. If a summary of the high-level requirements from obligations present in SE contracts is available to the requirement analysts in a language that is comprehensible to them, they can use this seed requirements knowledge to ask the right questions to the stakeholders. In this paper, we propose an approach for summarizing the requirements present in obligations in a language comprehensible to requirement analysts. We use the principles of Prompt Engineering to prompt GPT-3 to generate summaries for training Natural Language Generation (NLG) models for generating SE-specific summaries. Experiments using NLG models such as BART, GPT-2, T5, and Pegasus indicate that Pegasus generates the most accurate summaries with the highest ROUGE score as compared to other models.**

Keywords—Software Engineering Contracts, Software Requirements, Requirements Engineering, Abstractive Summarization, Large Language Models, Prompt Engineering

I. Introduction

A Software Engineering (SE) contract is an agreement between the customer and the SE partner (henceforth referred to as vendor) about mutual expectations and responsibilities. They serve as a valuable source in eliciting the software requirements of the customers [1]. Close to 40% of obligations mentioned in contracts are a source of software requirements [2].

The Requirements Engineering (RE) phase plays a very important role in the entire process of Software Development Life Cycle (SDLC) [3]. Typically, the process of requirements elicitation begins after the contractual agreement is signed by signatories from the customer and vendor side. Requirements knowledge derived from SE contracts can serve as a starting point for requirements elicitation and give a jump-start to the RE phase of SDLC. However, contracts are confidential documents and only the top management of an organization have a direct access to them. To use contracts as a starting point for eliciting requirements, the obligations pertinent to software requirements should be extracted from contracts and presented to the requirement analysts as a seed requirements knowledge which they can use to ask the right questions to the stakeholders. However, contracts text is lengthy, convoluted, incorporates a complex Legalese and is meant to be comprehended by legal experts, contract managers and finance department of an organization. For a requirement analyst to comprehend the requirements extracted from contracts documents, the requirements text should include only the information that is relevant to derive high-level SE requirements. One way to achieve this is by providing a summary of the high-level requirements from obligations present in SE contracts to the requirement analysts in a language that is comprehensible to them. Typically, requirement analysts study the customers' business domain broadly and try to extract as much information about the business from different sources available to them so that their RE sessions with customers are productive. The knowledge seed derived from a signed contract can further assist them in asking more specific questions aligned with the commitments mentioned in the contract. For example, a contracts text summary on security related obligations will assist the requirement analyst in asking questions to understand specific details pertaining to Security requirements such as geography-specific standards that must be adhered to while implementing safeguards. The preferred extent of this adherence can be disambiguated as opposed to being left open for interpretation of what is implied by vague terms generally found in contracts such as "reasonable". Customers' requirements for maintaining current certification as per the specific standards can also be elaborated and negotiated. If this is done early in the SE cycle, the software development activities can be aligned more precisely with the expectations of customers.

The length, complexity, and the specialized language of contracts text place a high cognitive burden on the human readers and increase the time required to understand and process the contractual information. Automated text summarization is a process of creating a condensed version of the body of text while retaining critical information elements and removing uncritical ones [4]. In recent years, many transformer-based deep learning models have reported state-of-the-art performance on the task of text summarization of general English language corpus. However, to the best of our knowledge, summarization of the high-level requirements from obligations present in SE contracts in a language that is comprehensible to requirement analysts has not yet been reported.

There are two main types of text summarization – Abstractive and Extractive [5]. While abstractive summarization uses natural language techniques to interpret and understand the important aspects of a text and generates a more "human" friendly summary using new or simplified words and/or phrases, an extractive summarization technique simply ranks all the sentences according to the understanding and relevance of the text and presents the most important sentences without creating new or simplified words or phrases. Since the goal of this work is to generate summaries in a language that is comprehensible to requirement analysts, we focus on generating abstractive summaries. We propose an abstractive contracts text summarization approach wherein we use the principles of Prompt Engineering (PE) [6,7] to provide prompts to GPT-3 [8] for generating summaries that are used to train several Natural Language Generation (NLG) models. Writing summaries manually for training the models is a time-consuming and an effort-intensive task. We therefore employ GPT-3 as an aid to generate summaries to be used for training the NLG models. Since we are dealing with contracts which are highly confidential organizational data, we completely obfuscate the training data to be fed to GPT-3. The GPT-3 generated training summaries are then manually refined (the details on refinement are presented in Section III. B. 3) by two senior requirement analysts and further used in training the NLG models. We would like to highlight that for an industry scale real-life application of this work, we cannot each time feed obfuscated contracts text to GPT-3 and unobfuscate the output (summaries) back for use by requirement analysts of a given project. We therefore used GPT-3 only as an aid in generation of training data that is used to train state-of-the-art NLG models to do the final summarization task. The final summarization is done on actual contracts data using Bidirectional Auto-Regressive Transformers (BART), Generative Pre-trained Transformer 2 (GPT-2), Text-to-Text Transfer Transformer (T5) and Pegasus. The model weights for these models are available and they operate in vendor's own infrastructure. Therefore, unlike with GPT-3, when BART, GPT-2, T5 and Pegasus are used for generating summaries, there are no concerns around data storage in external environments like GPT-3 and we do not need to obfuscate the input (contracts paragraphs) and then unobfuscate the output (summaries).

We conduct a range of experiments using state-of-the-art NLG techniques such as BART and GPT-2 and abstractive text-summarization technique namely T5 and Pegasus to generate abstractive summaries of the software requirements employing a language that is comprehensible to requirement analysts. The results from the experiments indicate that Pegasus generates the most accurate summaries with the highest Recall-Oriented Understudy for Gisting Evaluation (ROUGE) score as compared to other models. Even though evaluation metrics such as ROUGE score is useful in quantitative evaluation of automated summarization of text; they fail to capture semantic and contextual nuances of the text, are sensitive to text length and fail to evaluate coherence and readability of the summary. We therefore additionally performed human evaluation of the generated summaries. Manual evaluation of the generated summaries by requirement analysts on parameters – Correctness, Coherence, Completeness, Readability and Conciseness also ended up in ranking the Pegasus generated summaries highest when compared to the other models.

The rest of the paper is organized as follows: Section II covers related work. In Section III, the experiments to automate the summarization of requirement types from obligations present in SE contracts are explained. Section IV discusses threats to validity, Section V presents discussion in the light of continually emerging Large Language Models (LLMs). Section VI concludes the paper and provides direction for future work. In Section VII we discuss our stand on Data Availability to support the Open Science Policy.

II. RELATED WORK

In recent years, many transformer-based deep learning models reported state-of-the-art performance on the task of text summarization. However, most of them used general English corpus dataset [for example 9, 10, 11, 12]. Owing to the peculiarity and complexity of legal text which brings in unique challenges for text summarization, we focus only on summarization of legal dataset while discussing the related work. Significant research on summarization of legal text has been reported in the literature utilizing both extractive and abstractive summarization methods. We divide the related work into two subsections – extractive summarization of legal text and abstractive summarization of legal text.

A. Extractive Summarization of Legal Text

We found a lot of work on legal text summarization using extractive summarization method [13-39]. We summarize some of the notable work here. In [33], the authors observed that legal contracts are often written in dense and complex language that can be difficult for non-legal professionals to understand, leading to misunderstanding and disputes. To mitigate this issue, the authors proposed an initial dataset of legal text snippets paired with summaries written in plain English and manually verified the qualities of these summaries. The authors discovered that unsupervised extractive summarization methods such as TextRank, KLSum, Lead-1, Lead-K, and Random-K, did not perform well on this task due to level of abstraction and style difference between plain and legal English. The authors concluded the paper by hoping that their work will further encourage development of resource for simplification and transformation of style of legal language. In [34], the authors employed extractive summarization approaches for summarization of commercial contracts, and their performance was compared with human-written summaries. The outcome revealed that the LSA-based summarizer yielded the highest ROUGE score compared to other extractive techniques such as TF-IDF Summarization, TextRank, LexRank, and KL-Sum. In [35], the authors automated the creation of summaries for contracts documents using the extractive summarization approach. They introduced a software tool that incorporated Context-Free Grammar rules to identify significant clauses in the contracts. An assessment of the tool showed a 79.2% accuracy rate in recognizing and summarizing critical clauses. In [36], the author introduced a domain-specific extractive summarization model for summarizing privacy contracts. A CNN model was utilized to identify risk statements from the contract. Subsequently, a summarization model was constructed utilizing a risk-focused and coverage-focused content selection mechanism. The model's performance was compared to baseline approaches such as TextRank, KLSum, Lead-k, and Random. The outcome demonstrated that the proposed extractive-based summarization model outperformed the baseline methods. In [37], the authors developed an AI-assisted contract authoring through clause recommendation tool. The authors presented a pipeline to predict relevant clause types and generate appropriate content based on the contract content. However, their approach did not work well on lengthy contracts and contracts by nature are lengthy. In [38], an extractive summarization system was developed to create a summary of the key obligations, entitlement, and prohibitions in a contract. The system comprised of two modules: a content categorizer and an importance ranker. The effectiveness of the system was assessed by comparing it against several text rank baselines.

As discussed in Introduction section, extraction-based approach for text summarization only extracts crucial words, sentences, and paragraphs to generate a summary, leading to limitations in the coherence of the generated summary [39]. As opposed to extractive summarization, the abstractive approach is deemed superior as it yields more concise, informative, and coherent summaries that capture the essence of the source text and resembles human interpretation. Thus, in our work we used abstractive summarization of requirements from obligations present in SE contracts so that they are comprehensible to requirement analysts.

B. Abstractive Summarization of Legal Text

We found limited reported work that adopted abstractive summarization technique to generate summaries for legal text [40-47]. In [45], a thorough investigation was conducted on legal document summarization and the dataset used in such scenarios. The study also analyzed the quality of summarization with and without reinforcement learning. The finding indicated that the Rational Augmented Model (RAG) with deep reinforcement learning achieved superior performance for legal text summarization, with an F-score of 75 %. In case of text summarization, evaluation metric such as F-score fail to consider the sequential arrangement of words in summaries. Therefore, relying solely on F-score as an indicator of the quality of the generated summaries is not completely justified. In order to assess the effectiveness of generated summaries for a real-world application, it is imperative to include human experts to validate the generated summaries. In [46], the authors examined the types of automated text summarization techniques and their application in different fields such as legal contract analysis. A systematic study on abstractive text summarization was conducted, which emulates the human cognitive process of summarizing text. The authors analyzed the different techniques, challenges, opportunities, and current state-of-the-art in abstractive summarization. In [47], the authors performed abstractive summarization of construction contracts using three transformer-based models, namely BART, Pegasus, and Distilbart and the evaluation revealed that the Distilbart model outperformed the other two in terms of information correctness, information completeness, and human readability. The authors relied only on manual evaluation of the generated summaries and the evaluation did not include other critical parameters such as structure, coherence and conciseness, which we included as a part of manual verification. Also, our goal is to generate summaries in a way that legal text is comprehensive to requirement analysts.

In the field of legal text summarization, most of the reported work has used extractive summarization technique, with limited endeavours towards producing abstractive summaries. This potentially is due to the extensive natural language processing required for producing abstractive summaries [39]. Our work involves generation of abstractive summaries of software requirements from obligations present in SE contracts in a language that is comprehensible to requirement analysts so that they can take them as seed requirements knowledge and detail them further.

III. Abstractive Summarization of software requirements from Obligations Present in SE Contracts

A. Dataset Details

The dataset comprised of 251 expired SE contracts from 13 diverse application domains including healthcare, automotive, finance, banking, pharmaceuticals, telecom, technology, clothing retail, supermarket, agriculture, ecommerce, manufacturing, logistics and supply chain management. All these contracts belonged to the same large vendor organization and were intended to be executed by distributed teams for multinational companies operating worldwide. As per the contract governance team in the vendor organization, each contract document had between 100 to 500 pages. The contract governance team in the vendor organization provided us with a labeled dataset in excel format with columns ‘Document name’, ‘Contract Statement’ and ‘Statement Type’. The ‘Statement Type’ indicated whether each statement belonged to one or more requirement types (namely *Information Security, Data Privacy, Improvement and Innovation, Vendor IP Licensed, Export Law, Third Party IP Licensed* and *Standards)* or belonged to *Others* (*Others* included non-obligation statements and peripheral obligation statements such as those pertinent to sub-contracting, human resource related processes etc. that are not directly relevant for software development. The statements belonging to *Others* were excluded from our study). We received a total of 57,200 contractual statements from the vendor organization. Table I presents these ‘Statement Types’ along with instance count in the dataset.

TABLE I: DISTRIBUTION OF STATEMENT TYPES

Statement Type	Count
Information Security	3994
Data Privacy	918
Improvement and Innovation	188
Vendor IP Licensed	157
Export Law	113
Third Party IP Licensed	105
Standards	91
Others	51634

B. Automated Summarization of Software Requirements

In this section, we discuss the various transformer-based techniques that we experimented with for the summarization of software requirements from obligations present in SE contracts. Figure 1 depicts the summarization approach.

1) Dataset Preparation: As mentioned in section III A, we received a total of 57200 contractual statements from the contracts governance team in the vendor organization. These 57200 statements underwent a filtering process based on 'Statement Type'. Subsequently all obligation statements with Others as 'Statement Type' were removed from the dataset. Following this filtration process, the count of contractual statements was reduced to 5666. These statements were preprocessed using standard preprocessing steps such as null value removal and lemmatization. After preprocessing, we had a total of 5587 statements. The statements were then divided into train-test split with 4831 statements in training and 756 in testing. The statements were grouped on the basis of the requirement type and document i.e. all the requirements in one contract document belonging to the same requirement type formed one group. Following this approach for grouping, 264 paragraphs were obtained for training and 60 paragraphs were obtained for testing.

2) State-of-the-art Natural Language Generation Techniques employed in Summarization: We experimented with Pegasus, T5, GPT-2 and BART for generating abstrative summaries for contractual requirements from obligations present in SE contracts. Next, we briefly explain each of these techniques and the method adopted to fine-tune them for our purpose.

PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization (PEGASUS) [48] is a transformer-based sequence-to-sequence model with an encoder-decoder architecture (where encoder processes the input text, and the decoder generates the output summary) designed with gap-sentences generation as a pre-training objective to optimize fine-tuning performance specifically for abstractive text summarization. In pre-training Pegasus, relevant sentences are masked out from paragraphs and used as a separate output and the model is trained to generate these missing sentences. This gap-sentence generation [11] technique makes the model learn how to summarize long paragraphs by identifying relevant sentences and encourages the model to comprehend semantic relationships between sentences. Pegasus outperforms other models (which randomly selects sentences) by selecting relevant sentences from the paragraphs through Gap-sentence generation method. We have used Pegasus-Large model which is pretrained on C4 [49] (Colossal Clean Crawled Corpus) that is a cleaned version of web crawl corpus and Huge News dataset.

T5: Text-to-Text Transfer Transformer model (T5) [50] is a state-of-the-art NLG method based on transformer architecture that uses transfer learning. We have used T5-large model which is pretrained on a large data corpus (Colossal Clean Crawled Corpus) and finetuned for many text-to-text downstream tasks such as translation, question-answering and summarization. T5 is an encoder-decoder based transformer which uses multi-head mechanism for encoding input sequence and generating output summaries. T5 model uses sequence-to-sequence architecture for summarization and it consists of encoder and decoder, where the input sequence is tokenized into sequence of words and then fed into encoder layer consisting of self-attention layers and feed forward neural networks. This encoder layer generates a fixed length vector representation of the sequence of words as output and the decoder takes encoder output and generates sequence of output tokens one at a time by using an autoregressive approach. This [11] self-attention layer in encoder computes the entire input over the attention distribution and generates encoded representation for each input token. The encoded input sequence is passed through the decoder, and it generates output based on previous output and the encoded input sequence. It also uses embedding matrix shared by input/output that allows the model to fill it with weights from the output of the final layer.

BART: Bidirectional and Auto Regressive Transformers (BART) [51], is a Seq2Seq transformer model based on the architecture of GPT and BERT [11]. BART architecture consists of encoder-decoder framework wherein, bi-directional encoder (captures contextual information from both direction in input text) encodes input paragraphs and generates hidden states which are then passed to autoregressive decoder to generate the summary. Encoder and decoder in this model

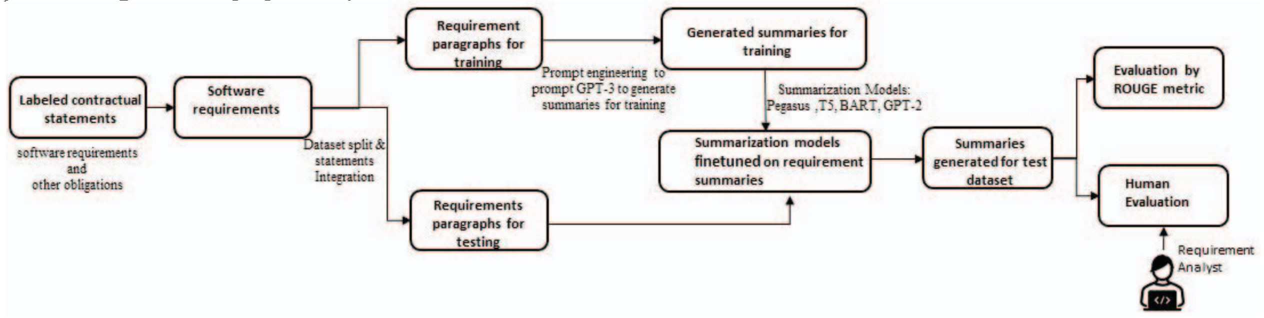
Labeled contractual statements
software requirements and other obligations
Software requirements
Dataset split & statements Integration
Requirement paragraphs for training
Requirements paragraphs for testing
Prompt engineering to prompt GPT-3 to generate summaries for training
Generated summaries for training
Summarization Models: Pegasus ,T5, BART, GPT-2
Summarization models finetuned on requirement summaries
Summaries generated for test dataset
Evaluation by ROUGE metric
Human Evaluation
Requirement Analyst

Fig. 1. Summarizing software requirements from obligations present in SE contracts

consist of multi-head attention layers and feed-forward layers which are necessary for generating relevant summaries. BART uses a denoising autoencoder pre-training objective, wherein the model is trained to improve its ability to handle noisy input. This denoising objective improves the performance of the model on downstream tasks such as text summarization. We have used Facebook/BART-Large-CNN model which is pretrained on English language text.

GPT-2: Generative Pretrained Transformer 2 (GPT-2) [52] is pretrained on large corpus of English data to predict subsequent word in a sequence. It uses unsupervised pretraining as training objective wherein the model is trained on a large corpus of text without any specific downstream task in mind. This enables the model to learn text representation patterns and dependencies in the text which can further be finetuned on specific downstream tasks such as summarization. The architecture of this model consists of several layers of self-attention and feed-forward neural network layers which can attend to specific parts of the input text and further generate relevant summary. It uses decoder-only architecture where the input text is tokenized and fed into the model that predicts next word in the sequence based on previous word of the input text and generates the summary. We have used GPT-2 Large model which is pretrained on a very large corpus of English data.

3) Fine-tuning: We first attempted summarization of requirements from SE contracts without finetuning the four model on SE contracts dataset. Without finetuning, all the four models returned nearly the original text as summary. They neither reduced the text size while generating the summaries nor simplified them to remove the legal jargons. These pretrained models have a good understanding of data patterns as they are trained on a large corpus of diverse text such as C4 corpus (which includes some legal terminologies). However, they may not perform well for summarization of legal text without finetuning as they lack knowledge of legal domain in general and contracts in particular. Therefore, we decided to finetune the models for summarization task on SE contracts dataset so that they learn the vocabulary of SE contracts and produce more accurate summaries relevant to SE stakeholders.

All the four models were finetuned for the downstream task of summarization of requirements from obligations present in SE contracts dataset. For fine-tuning the models for summarization, we required summaries of the contractual requirement paragraphs. For this, we took the 264 contractual requirement paragraphs that were segregated for training. Manually creating summaries for 264 large paragraphs of contractual text is an effort intensive and time-consuming task with dependence on experts. To ease this task, we used GPT-3 prompts as an aid in generating the summaries for training. The quality of the input prompt determines the quality of the generated summaries. Designing effective prompts increases the likelihood that the model will return a response that is both favorable and contextual. To achieve this, we used the principles of Prompt Engineering to generate contextual prompts. Prompt Engineering (PE) is a Natural Language Processing (NLP) concept of developing and refining prompts that make large language models yield desirable or useful results [6, 7]. In our study, we leveraged PE to orchestrate prompts that are capable of generating summaries that are as close as possible to summaries generated by human experts. We followed the key principles of PE [53] such as providing relevant context, ensuring specificity and adopting a methodical approach to arrive at the most precise prompt. Following these key principles, we designed seven different prompts each specific to the seven software requirement types mentioned in Table 1. We then refined the prompts iteratively through manual verification of the output generated by each prompt. This approach helped us to tailor the prompts to specific software requirement type so that the prompt takes into cognizance the requirement type while generating the summaries. We used mathematical notations [54] to design these prompts and provided them as input to GPT-3 for generating summaries for training the NLG models. Table II summarizes the mathematical notations that we have used in designing the prompts.

TABLE II. MATHEMATICAL NOTATIONS FOR PE

Terminology	Notation	Description
Input	x or p	Input paragraph to be summarized
Output	Y	Output summary generated
Prompting function	y = f (p | c)	y is a function of p given c, where c is a condition
Substitute	x = p	Replacing value of x by p

Next, we present an example <Original Paragraph (obfuscated)> belonging to the requirement type – '*Standards'*. The examples are obfuscated (particularly the client name and the name of the standards and years) due to confidentiality agreement signed by the vendor organization and the customer. In Table III, we present an example of refining prompts iteratively to arrive at a summary for '*Standards'* requirement type which is closest to the one created by human experts.

<Original Paragraph (obfuscated)>: "*Supplier will maintain and enforce physical and logical security procedures with respect to its access and maintenance of <client_name> Systems for which Supplier is responsible and any information or data on <client_name> Systems to which Supplier has access where such security procedures are (i) at least equal to industry standards for such types of locations such as <standard_1>, (ii) in accordance with reasonable <client_name> security requirements as notified to Supplier and (iii) which provide reasonably appropriate technical and organizational safeguards against accidental or unlawful destruction, loss, alteration or unauthorized access to or disclosure of information or data on <client_name> Systems to which Supplier has access and <client_name> hereby confirms that such reasonably appropriate technical and organizational safeguards implemented by Supplier provides an appropriate level of protection of information or data upon review of a security assessment as requested by <client_name>. Supplier will maintain current certification (i.e., <standard_2>, <standard_3>, <standard_4>, and <standard_5>) and shall maintain such certification for the term of this* Agreement."

As is evident from Table III, we iteratively refined the prompts to bring the generated summaries as close as possible to the summary written by human experts. The ***First Prompt*** that we created was a generic prompt that generated a summary for <Original Paragraph (obfuscated)> without taking into cognizance the target audience (requirement analysts in this case) who is going to consume the output. As a result, the summary generated by GPT-3 on the basis of the ***First Prompt*** retained the convoluted and complex legalese inherent to the contracts text. Taking this as a feedback, we refined the ***First Prompt*** and created a ***Second Prompt*** to include knowledge about the target audience. The summary generated by GPT-3 based on the ***Second Prompt*** was therefore more comprehensible to requirement analysts as it simplified the legal jargons. However, the ***Second Prompt*** failed to teach GPT-3 the terms or phrases in the original paragraph that GPT-3 needs to give attention to. We therefore refined the ***Second Prompt*** and created a ***Third Prompt*** that brought in the knowledge of the requirement type – '*Standards'*. This helped GPT-3 to generate prompts that were closet to the ones that human experts would create. As is evident from Table III, the summary generated after the third prompt seems longer. However, it is information rich and includes all the required details pertinent to the requirement type. The GPT-3 generated summaries were manually reviewed by two senior requirement analysts (20 plus years of experience) from the vendor organization and modified in case they felt that the summaries needed further tuning. As per the two senior requirement analysts, the GPT-3 generated summaries served as a useful aid. They needed to do some refinements, but the automation considerably eased their task of manual summarization. The refinements included correcting distortions in the GPT-3 generated summaries such as correcting the names of sections, laws and standards. The requirement analyst further noticed that GPT-3 generated summaries sometimes contained redundant and repetitive information and over-simplified names of sections, laws and standards. For example, instead of *Export Control Law*, the output showed *Applicable Law*. The generated summaries also sometimes omitted important information. For example, for a paragraph that detailed contractual obligations for data transfer, the GPT-3 generated summary correctly picked up the data transfer needs, but details of preferred mechanisms mentioned in the paragraph were not picked up. Including these details for training the NLG models (such as T5, Pegasus, BART and GPT-2 employed in our final summarization experiments) was important to derive meaningful summaries for requirement analysts. As per the two senior requirement analysts, the refinement activity took around 1 to 2 minutes per summary as opposed to 12 to 15 minutes for each summary if done completely manually.

Next, we present an example of the final summary generated for <Original Paragraph (obfuscated)>.

Final summary (obfuscated) after manual refinements by senior requirement analysts - *The supplier will maintain physical and logical security procedures for management and safeguard of client's systems and data which they can access. These procedures must comply with industry standards (<standard_1>) and client's security requirements and must also provide technical safeguards to prevent accidental or unlawful destruction, loss, modification or unauthorized access or disclosure of client's information or data. The supplier will provide an appropriate level of protection, reviewing a security assessment requested by the client and shall also maintain current certifications (i.e., <standard_2>, <standard_3>, <standard_4>, and <standard_5>) for the term of the agreement*.

Once all the summaries for training the NLG models were generated, they were fed into the four NLG models for fine-tuning them for the summarization of high-level requirements from obligations present in SE contracts. We performed hyperparameter tuning to optimize the performance of the models. We identified the optimal hyperparameters for each

TABLE III. EXAMPLE OF REFINING PROMPTS USING PRINCLIPLES OF PROMPT ENGINEERING

Method	**Prompt + <Original Paragraph (obfuscated)>**	**GPT-3 Output**

First Prompt (Generic or foundation prompt)	p = <Original Paragraph (obfuscated)>
	c = Concise summary
	f(x) = Could you please summarize the paragraph [x]?
	x = p
	y = f (p | c)
	Evaluate y
		"Supplier will maintain and enforce physical and logical security procedures with respect to <client_name> Systems, information, and data that meets industry standards and <client_name> security requirements. Supplier will maintain current certifications and maintain them for the duration of the agreement."

Second Prompt (Requirement analysts specific prompt)	p = <Original Paragraph (obfuscated)>
	c = Concise and in context of requirement analysts
	f(x) = Could you please summarize the paragraph [x]?
	x = p
	y = f (p | c)
	Evaluate y
		"Supplier will maintain and enforce physical and logical security procedures with respect to its access and maintenance of <client_name> Systems, in accordance with reasonable <client_name> security requirements and standards. Supplier must maintain current certification for the term of this Agreement and provide an appropriate level of protection of information or data upon review of a security assessment as requested by <client_name>."

Third Prompt (requirement type – 'Standards' specific prompt)	p = <Original Paragraph (obfuscated)>
	c = Concise and 'Standards' requirement types ('Standard' includes requirements from obligation in SE contracts that deal with various audit and industry standards that needs to be complied with.)
	f(x) = Could you please summarize the paragraph [x]?
	x = p
	y = f (p | c)
	Evaluate y
		Supplier must maintain and enforce physical and logical security procedures with respect to access and maintenance of <client_name> Systems, and any information or data on such systems, that are equal to or exceed industry standards (e.g. *<standard_1>*), in accordance with reasonable <client_name> security requirements and provide reasonably appropriate technical and organizational safeguards against accidental or unlawful destruction, loss, alteration or unauthorized access or disclosure. Supplier must also maintain current certification (e.g. *<standard_2>, <standard_3>, <standard_4>, and <standard_5>*) for the duration of the agreement.

model, with learning rate having the most significant impact on performance. Table IV presents optimal hyperparameters with their values used for finetuning the four models. We would like to highlight that all the four models performed the best with the parameter values mentioned in Table IV.

C. Evaluation and Results

As an example, we first present an original software requirement paragraph (obfuscated), and in Table V, we provide its ground truth summary, along with the model-generated summaries for each of the four transformer-based model.

Original Paragraph (obfuscated): *The Service Provider must submit to the <client_name> a transition plan for migrating to <standard_1> within 3 months of the revised standard being published by the International Organization for Standardization. the Quality Management System must meet the requirements of <standard_2> "Quality Management Systems Requirements". The Service Provider must submit to the <Client_name> a transition plan for migrating to <standard_1>.*

TABLE IV. HYPERPARAMETERS UTILIZED FOR FINETUNING

Hyperparameters	Value
learning_rate	3e-4
save_steps	100
num_train_epochs	8
train_batch_size & eval_batch_size	4

We evaluated the summaries generated by the four models using (1) automated evaluation metrics for summarization and (2) human evaluation. Although automated evaluation metrics are useful in quantitative evaluation, they are not able to capture the semantic and contextual nuances of text, are sensitive to text length and fail to evaluate the coherence and readability of the generated summaries. Human evaluation takes into consideration several facets of summarization such as its relevance, coherence, fluency and overall readability. However, human evaluation is time consuming and may introduce subjective biases. We evaluated the generated summaries using (1) automated metrics and (2) human evaluation, for comprehensively evaluating the summaries generated by the four transformer-based models. Such an evaluation can offer informative and balanced assessment of the quality of the generated summaries. Next, we present details on evaluation of summaries using automated metrics and human evaluation.

1) ***Evalution using Automated Metrics:*** Some of the commonly used metrics for evaluation of generated summaries are - ROUGE [55], BLEU [56] and METEOR [57,58]. BLEU and METEOR are designed to be primarily used for evaluation of machine-translation tasks. BLEU calculates similarity based on precision of only n-grams. METEOR is similar to BLEU but calculates similarity based on f-score and may not perform well for summaries as its 'concept similarity component' considers fluency and grammatical meaning of the overlapping words by using synonymy and paraphrase detection. ROUGE on the other hand, concentrates on content overlap. It calculates not only n-gram overlap but also the longest common subsequence that captures more content and is more content-specific. Since our goal *is* to generate summaries that could be understood by requirement analysts and not to check for grammatical mistakes and fluency, we considered ROUGE metrics for autom*a*ted evaluation of the generated summaries.

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) is a metric to evaluate the quality of the generated summaries with respect to ground truth summaries. In the realm of abstractive summarization, ROUGE metric is used as a preliminary measure of how well the generated summary aligns with the ground truth summary. ROUGE has several variants - ROUGE-1, ROUGE-2, ROUGE-L.

TABLE V. EXAMPLE GROUND TRUTH SUMMARY AND THE SUMMARY GENERATED BY THE FOUR MODELS

Ground Truth Summary (obfuscated)	Model	Generated Summary (obfuscated)
The Service Provider must submit a transition plan to the <Client_name> for migrating to <*standard_1*> within 3 months of the revised standard being published by the International Organization for Standardization. The Quality Management System must meet the requirements of <*standard_2*> "Quality Management Systems Requirements".	Pegasus (finetuned)	The Service Provider must submit to the <Client_name> a transition plan for migrating to <*standard_1*> within 3 months of the revised standard being published by the International Organization for Standardization. The Quality Management System must meet the requirements of <*standard_2*> "Quality Management Systems Requirements".
	T5 (finetuned)	The Service Provider must submit a transition plan for migrating to <*standard_1*> within 3 months of the revised standard being published by the International Organization for Standardization. The Quality Management System must meet the requirements of <*standard_2*> "Quality Management Systems Requirements".
	BART (finetuned)	The Service Provider must submit a transition plan to the Client within 3 months of the revised standard being published by the International Organization for Standardization. The Quality Management System must meet the requirements of <*standard_2*> "Quality Management Systems Requirements".
	GPT-2 (finetuned)	The Service Provider must submit to the <Client_name> a transition plan for migrating to <*standard_1*> within 3 months of the revised standard being published by the International Organization for Standardization. The Quality Management System must meet the requirements of <*standard_2*> "Quality Management Systems Requirements". The Service Provider must submit to the <Client_name> a transition plan for migrating to <*standard_1*>

ROUGE-1 measures unigram overlap between the model generated summaries and the ground truth summaries.

ROUGE-2 measures bigram overlap between the model generated summaries and the ground truth summaries.

ROUGE-L measures the longest common subsequence overlap between the model generated summaries and the ground truth summaries.

As each variant of ROUGE measures a different aspect of the generated summary, we used all three variants of ROUGE in evaluating the summaries. Each variant of ROUGE is important for capturing content overlap at various levels of specificity such as at phrase level or at sentence level or at longest common subsequence level. Using all 3 variants particularly helps in case where the generated summaries differ from the ground truth summaries in word order but captures the same content. ROUGE is calculated in terms of the metrics of Recall, Precision and F1-score. Following is the method for calculating these metrics in the context of ROUGE.

Recall (R) in case of ROUGE is the measure of the summary captured with respect to the model generated summary.

$$\boldsymbol{R} = \frac{(\boldsymbol{No.\ of\ overlapping\ words})}{(\boldsymbol{Total\ no.\ of\ words\ in\ ground\ truth\ summary})}$$

Precision (P) in ROUGE measures the relevancy of the model generated summary.

$$\boldsymbol{P} = \frac{(\boldsymbol{No.of\ overlapping\ words})}{(\boldsymbol{Total\ no.\ of\ words\ in\ generated\ summary})}$$

F1-score is the harmonic mean of Recall and Precision.

$$\boldsymbol{F1 - score} = \boldsymbol{2} * \frac{(\boldsymbol{Precision} * \boldsymbol{Recall})}{(\boldsymbol{Precision} + \boldsymbol{Recall})}$$

We evaluated the performance of the four models on the 60 requirements paragraphs segregated for testing. The ground truth for the test set was obtained by leveraging PE (similar to the way we generated summaries for finetuning process). Table VI presents ROUGE score of the four models on the test dataset of 60 paragraphs of software requirements. As is evident from Table VI, Pegasus outperformed the other transformer-based models with respect to ROUGE score.

TABLE VI. SUMMARY EVALUATION USING ROUGE METRICS FOR THE FOUR MODELS

Model	**ROUGE SCORE**		
	ROUGE-1	***ROUGE-2***	***ROUGE-L***
Pegasus (finetuned)	**0.67**	**0.51**	**0.61**
T5 (finetuned)	0.65	0.48	0.58
GPT-2 (finetuned)	0.62	0.47	0.57
Bart (finetuned)	0.59	0.43	0.52

2) Human Evaluation: To strengthen the evaluation study, in addition to the automated evaluation using ROUGE, we asked 10 requirements analysts (different from those who participated in refining the ground truth) from the vendor organization to manually evaluate a small sample of the generated summaries. Each requirement analyst had eight plus years of experience in their respective business domain, interacting with customers and gathering requirements for various large projects from across the globe. Each requirement analyst was given a different original software requirement paragraph from obligations present in a SE contract from our dataset and four corresponding model-generated summaries. We masked the identity of the models from the requirement analysts. They were then asked to rate each generated summary on a scale of 1 to 5 (with higher ratings indicating better performance) by comparing it to the original software requirement paragraph, based on five standard summary evaluation factors - Correctness, Coherence, Completeness, Readability and Conciseness [47]. We chose these factors because they allow human evaluators to verify that the generated summaries are not only accurate and comprehensive but also well written and easy to understand. We gave the following definitions of the factors to the requirement analysts.

Correctness: Correctness pertains to the extent to which the generated summary represents the factual information from the original text.

Coherence: Coherence checks if the generated summary presents a logically organized structure and sequence of ideas that are interconnected.

Completeness: Completeness refers to the extent to which the model generated summary retains the pertinent and crucial information from the original text.
Readability: Readability refers to how easily the generated summary can be understood.

Conciseness: Conciseness indicates whether the generated summary effectively conveys important information with minimal wording by avoiding unnecessary and redundant information.

We averaged the ratings from all 10 requirement analysts for each of the five factors, based on the four models. The results from the study are presented in Table VII. As is evident from Table VII, manual evaluation of the generated summaries by requirement analysts ranked Pegasus the best among the four models.

TABLE VII. FACTOR DRIVEN HUMAN EVALUATION OF MODEL PERFORMANCE

Models	**Pegasus**	**T5**	**BART**	**GPT-2**
Correctness	**4.46**	4.38	4.15	4.3
Coherence	**4.53**	4.46	3.92	3.84
Completeness	**4.46**	4.38	3.92	4.38
Readability	**4.69**	4.53	4	3.53
Conciseness	**3.76**	**3.76**	3.23	2.15

At this point, we have not conducted a systematic investigation of how much effort and time the requirement analysts would need for summarizing the requirements from contracts. In fact, the requirement analysts had not attempted to use contracts as a source for requirements since contracts are confidential documents and only the top management of their departments had a direct access to them. They did note however, that a summary of "SE -specific expectations" can be helpful as a starting point for requirements elicitation of projects.

IV. Threats to Validity

The first potential threat to validity is the universality of the SE contracts dataset that we used for summarization. We acknowledge that our dataset may not capture the complete scope of language and terminology used across different contracts from all the domains. However, since our dataset contains 251 real world contracts documents from 13 different application domains, we expect it to be a reasonable representation of the software requirements from obligations present in SE contracts.

The second threat to validity is the use of GPT-3 to generate summaries for training NLG models and for creating ground truth. As observed by the senior requirement analysts in the vendor organization, the GPT-3 generated summaries are not always accurate and therefore cannot be directly used to fine-tune the model effectively. To mitigate this threat, we used GPT-3 as only an aid to generate the training and ground truth summaries. Two senior requirement analysts carefully reviewed the GPT-3 generated summaries and manually refined them to make them more accurate.

The third threat to validity pertains to the use of ROUGE to evaluate model generated summaries in an automated way. ROUGE primarily measures content overlap and may not adequately capture semantic and contextual nuances of text and may fail to evaluate coherence and readability of the summary. To mitigate this threat, we additionally conducted human evaluation wherein we involved ten requirement analysts to manually evaluate the generated summaries by considering factors such Correctness, Coherence, Completeness, Readability and Conciseness.

The fourth threat is about human evaluation of the generated summaries. Human evaluation is known to be biased to some extent. To mitigate this threat to the extent possible, we asked ten experienced requirement analysts from the vendor organization to evaluate a sample of the generated summaries based on certain standard factor. We masked the identity of the model that generated the summary. This led to an unbiased evaluation of the summaries.

The fifth threat to validity is the use of GPT-2 instead of GPT-3 as one of the four models for testing. We used GPT-2 as the weights of GPT-3 are not yet made publicly available. Further, unlike GPT-2, finetuning GPT-3 is a challenging task as it is a relatively very large and complex model. Finetuning GPT-3 on custom dataset therefore requires significant amount of time and computational resources.

V. Discussion in the Light of Continually Emerging Large Language Models

The recent swirl of releases of new variants of generative AI have brought a storm of hype and fright. Large Language Models (LLMs) such as GPT-3, ChatGPT and the recently released GPT-4 appear to be a gamechanger in almost every sector. These LLMs are an accelerant that operate at such a scale and speed that they can appear to do whatever one prompts them to do. The more we use them and feed them data and questions, the faster and better they learn to predict desirable outcomes.

However, at this stage, these LLMs cannot be used without human in the loop. This is to make sure that the output generated from them are accurate, reliable, efficient and does no harm. Additionally, in their current form, these LLMs pose some major practical risks especially when using them to solve real-life industrial problems and feeding them highly confidential data such as SE contracts. Data privacy has been the biggest concern for corporates around the world who are looking to use GPT-3 or its current subsequent variants for creating niche domain-specific applications. Even though there are hints on introducing Data Privacy Agreement (DPA) and Memorandum of Understanding (MoU) for corporate accounts, we are still not at a stage wherein there is clarity on how practical and reliable these would be. Further, the access to GPT-3 is currently given in the form of an open-ended API which allows the users to provide training data to GPT-3 in the form of prompts. GPT-3 uses these input prompts to come up with the appropriate output. For individual accounts, the GPT-3 generally stores the training data as a part of its continuous learning feature that makes the model better on the go. This feature creates a hitch around using GPT-3 for use cases involving highly confidential data.

Given the above-mentioned concerns, we used GPT-3 only as an aid to generate training data with an intent to ease the effort of requirement analysts in creating manual summaries for training. The training data fed to GPT-3 was completely anonymized to remove any reference to customer identity, their proprietary processes or standards. We did not use GPT-3 or ChatGPT as the final model, as we cannot feed them unobfuscated sentences for generating summaries. Alternatively, we used state-of-the-art NLG models such as Pegasus, GPT-2, T5 and BART to do the final summarization task on real contracts data as the model weights for these models are available and there are no concerns around storing of data by these models.

VI. Conclusion anD Future Work

In this paper, we proposed a contracts text summarization approach for summarizing the requirements from obligations present in SE contracts employing a language that is comprehensible to requirement analysts. We used the principles of PE to prompt GPT-3 to generate training and ground truth summaries for training NLG models on contract text summarization task. Through experiments, we found that Pegasus outperformed other NLG models in terms of ROUGE score. Additionally, we conducted manual evaluation of the generated summaries by requirement analysts which corroborated the findings from ROUGE score.

Although the proposed approach shows promising results, there are several areas that can be explored for future research. We plan to expand our training dataset further that may result in improved accuracy. We also plan to use Prompt based-learning instead of fine-tuning to train the NLG models for contracts text summarization. This will eliminate the need of labeled datasets. We plan to further engineer our prompts to make them more SE-specific so that the generated summaries are closer to SE requirements. As we write this paper, GPT-4 [59] is released. We will explore GPT-4 and its impact of SE-specific summarization task.

Another interesting research direction is to conduct empirical studies involving requirement analysts to evaluate the usefulness of our approach. We plan to evaluate the savings in terms of effort and time that such an approach would bring in when compared to manual summarization.

VII. Data Availability

The replication package of this work includes (a) the code, (b) the prompts. These artifacts are made available publicly and can be accessed via https://zenodo.org/record/8026646. The SE contracts dataset used for training and testing the models cannot be made publicly available as it includes confidential proprietary contracts document signed between the vendor organization and their different customers and carries Non-Disclosure Agreements (NDAs). The legal department of the vendor organization prohibits sharing of this confidential data.

References

[1] Nekvi, R., Ferrari, R., Berenbach, B. and Madhavji, N.H., 2011, August. Towards a compliance meta-model for system requirements in contractual projects. In *2011 Fourth International Workshop on Requirements Engineering and Law* (pp. 74-77). IEEE.

[2] Sainani, A., Anish, P.R., Joshi, V. and Ghaisas, S., 2020, August. Extracting and classifying requirements from software engineering contracts. In *2020 IEEE 28th international requirements engineering conference (RE)* (pp. 147-157). IEEE).

[3] Atas, M., Samer, R. and Felfernig, A., 2018, December. Automated Identification of Type-Specific Dependencies between Requirements. In 2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI) (pp. 688-695). IEEE

[4] Abualigah L. Bashabsheh M. Alabool Q. H. and Shehab M. Text summarization: a brief review. *Recent Advances in NLP: The Case of Arabic Language*, 1-15, 2020

[5] Chen, Y., Ma, Y., Mao, X. and Li, Q., 2019. Multi-task learning for abstractive and extractive summarization. *Data Science and Engineering*, *4*, pp.14-23.

[6] Oppenlaender, Jonas. (2022). Prompt Engineering for Text-Based Generative Art.

[7] Reynolds, Laria & McDonell, Kyle. (2021). Prompt Programming for Large Language Models: Beyond the Few-Shot Paradigm.

[8] Sorensen, Taylor & Robinson, Joshua & Rytting, Christopher & Shaw, Alexander & Rogers, Kyle & Delorey, Alexia & Khalil, Mahmoud & Fulda, Nancy & Wingate, David. (2022). An Information-theoretic Approach to Prompt Engineering Without Ground Truth Labels.

[9] Völske, Michael, Martin Potthast, Shahbaz Syed, and Benno Stein. "Tl; dr: Mining reddit to learn automatic summarization." In *Proceedings of the Workshop on New Frontiers in Summarization*, pp. 59-63. 2017.

[10] Singh, Rajeev Kumar, Sonia Khetarpaul, Rohan Gorantla, and Sai Giridhar Allada. "SHEG: summarization and headline generation of news articles using deep learning." *Neural Computing and Applications* 33 (2021): 3251-3265.

[11] Gupta, Anushka, Diksha Chugh, and Rahul Katarya. "Automated news summarization using transformers." In *Sustainable Advanced Computing: Select Proceedings of ICSAC 2021*, pp. 249-259. Singapore: Springer Singapore, 2022.

[12] Song, Shengli, Haitao Huang, and Tongxiao Ruan. "Abstractive text summarization using LSTM-CNN based deep learning." *Multimedia Tools and Applications* 78 (2019): 857-875.

[13] Pandya, Varun. "Automatic text summarization of legal cases: A hybrid approach." *arXiv preprint arXiv:1908.09119* (2019).

[14] Polsley, Seth, Pooja Jhunjhunwala, and Ruihong Huang. "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. 2016.

[15] Bhattacharya, Paheli, Soham Poddar, Koustav Rudra, Kripabandhu Ghosh, and Saptarshi Ghosh. "Incorporating domain knowledge for extractive summarization of legal case documents." In *Proceedings of the eighteenth international conference on artificial intelligence and law*, pp. 22-31. 2021.

[16] Nguyen, Duy-Hung, Bao-Sinh Nguyen, Nguyen Viet Dung Nghiem, Dung Tien Le, Mim Amina Khatun, Minh-Tien Nguyen, and Hung Le. "Robust deep reinforcement learning for extractive legal summarization." In *Neural Information Processing: 28th International Conference, ICONIP 2021, Sanur, Bali, Indonesia, December 8–12, 2021, Proceedings, Part VI 28*, pp. 597-604. Springer International Publishing, 2021.

[17] Shukla, Abhay, Paheli Bhattacharya, Soham Poddar, Rajdeep Mukherjee, Kripabandhu Ghosh, Pawan Goyal, and Saptarshi Ghosh. "Legal case document summarization: Extractive and abstractive methods and their evaluation." *arXiv preprint arXiv:2210.07544* (2022).

[18] Schweighofer, E. "Improving Legal Case Summarization Using Document-Specific Catchphrases." In *Legal Knowledge and Information Systems: JURIX 2021: The Thirty-fourth Annual Conference, Vilnius, Lithuania, 8-10 December 2021*, vol. 346, p. 76. IOS Press, 2022.

[19] Bansal, Neha, Arun Sharma, and R. K. Singh. "Fuzzy AHP approach for legal judgement summarization." *Journal of Management Analytics* 6, no. 3 (2019): 323-340.

[20] Parikh, Vedant, Vidit Mathur, Parth Mehta, Namita Mittal, and Prasenjit Majumder. "Lawsum: A weakly supervised approach for indian legal document summarization." *arXiv preprint arXiv:2110.01188* (2021).

[21] Begum, Naimoonisa, and Ankur Goyal. "Analysis of Legal Case Document Automated Summarizer." In *2021 6th International Conference on Signal Processing, Computing and Control (ISPCC)*, pp. 533-538. IEEE, 2021.

[22] Deroy, Aniket, Kripabandhu Ghosh, and Saptarshi Ghosh. "Ensemble methods for improving extractive summarization of legal case judgements." *Artificial Intelligence and Law* (2023): 1-59.

[23] Jain, Deepali, Malaya Dutta Borah, and Anupam Biswas. "Automatic summarization of legal bills: A comparative analysis of classical extractive approaches." In *2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, pp. 394-400. IEEE, 2021.

[24] Satwick Gupta, M. N., NL Siva Narayana, V. Sai Charan, Kunam Balaram Reddy, Malaya Dutta Borah, and Deepali Jain. "Extractive Summarization of Indian Legal Documents." In *Edge Analytics: Select Proceedings of 26th International Conference—ADCOM 2020*, pp. 629-638. Singapore: Springer Singapore, 2022.

[25] M Viswanathan, Daleesha, and R. Rajesh. "Automatic Legal Judgment Summarization Using Graph." (2020).

[26] Jain, Deepali, Malaya Dutta Borah, and Anupam Biswas. "CAWESumm: A Contextual and Anonymous Walk Embedding Based Extractive Summarization of Legal Bills." In *Proceedings of the 18th International Conference on Natural Language Processing (ICON)*, pp. 414-422. 2021.

[27] Agarwal, Abhishek, Shanshan Xu, and Matthias Grabmair. "Extractive Summarization of Legal Decisions using Multi-task Learning and Maximal Marginal Relevance." *arXiv preprint arXiv:2210.12437* (2022).

[28] Zhong, Yang, and Diane Litman. "Computing and Exploiting Document Structure to Improve Unsupervised Extractive Summarization of Legal Case Decisions." *arXiv preprint arXiv:2211.03229* (2022).

[29] Anand, Deepa, and Rupali Wagh. "Effective deep learning approaches for summarization of legal texts." *Journal of King Saud University-Computer and Information Sciences* 34, no. 5 (2022): 2141-2150.

[30] Jain, Deepali, Malaya Dutta Borah, and Anupam Biswas. "Bayesian Optimization based Score Fusion of Linguistic Approaches for Improving Legal Document Summarization." *Knowledge-Based Systems* 264 (2023): 110336.

[31] Kanapala, Ambedkar, Srikanth Jannu, and Rajendra Pamula. "Passage-based text summarization for legal information retrieval." *Arabian Journal for Science and Engineering* 44 (2019): 9159-9169.

[32] Jain, Deepali, Malaya Dutta Borah, and Anupam Biswas. "A sentence is known by the company it keeps: Improving Legal Document Summarization Using Deep Clustering." *Artificial Intelligence and Law* (2023): 1-36.

[33] Manor, Laura, and Junyi Jessy Li. "Plain English summarization of contracts." *arXiv preprint arXiv:1906.00424* (2019).

[34] Balachandar, Keshav, Anam Saatvik Reddy, A. Shahina, and Nayeemulla Khan. "Summarization of commercial contracts." (2021).

[35] Kubeka, Sibusiso, and Abejide Ade-Ibijola. "Automatic Comprehension and Summarisation of Legal Contracts." *contract* 9 (2021): 10.

[36] Keymanesh, Moniba. "Adapative Summarization for Low-resource Domains and Algorithmic Fairness." PhD diss., The Ohio State University, 2022.

[37] Aggarwal, Vinay, Aparna Garimella, Balaji Vasan Srinivasan, and Rajiv Jain. "CLAUSEREC: A Clause Recommendation Framework for AI-aided Contract Authoring." *arXiv preprint arXiv:2110.15794* (2021)..

[38] Sancheti, Abhilasha, Aparna Garimella, Balaji Vasan Srinivasan, and Rachel Rudinger. "What to Read in a Contract? Party-Specific Summarization of Important Obligations, Entitlements, and Prohibitions in Legal Documents." *arXiv preprint arXiv:2212.09825* (2022).

[39] Widyassari, Adhika Pramita, Supriadi Rustad, Guruh Fajar Shidik, Edi Noersasongko, Abdul Syukur, and Affandy Affandy. "Review of automatic text summarization techniques & methods." *Journal of King Saud University-Computer and Information Sciences* (2020).

[40] Venkataramanan, Kannan, Sandeep Bhupatiraju, and Daniel Li Chen. "Automated Legal Information Retrieval and Summarization."

[41] Ghosh, Satyajit, Mousumi Dutta, and Tanaya Das. "Indian Legal Text Summarization: A Text Normalisation-based Approach." *arXiv preprint arXiv:2206.06238* (2022).

[42] Salaün, Olivier, Aurore Troussel, Sylvain Longhais, Hannes Westermann, Philippe Langlais, and Karim Benyekhlef. "Conditional Abstractive Summarization of Court Decisions for Laymen and Insights from Human Evaluation." In *Legal Knowledge and Information Systems*, pp. 123-132. IOS Press, 2022.

[43] Shen, Zejiang, Kyle Lo, Lauren Yu, Nathan Dahlberg, Margo Schlanger, and Doug Downey. "Multi-LexSum: Real-World Summaries of Civil Rights Lawsuits at Multiple Granularities." *arXiv preprint arXiv:2206.10883* (2022).

[44] Feijo, Diego de Vargas, and Viviane P. Moreira. "Improving abstractive summarization of legal rulings through textual entailment." *Artificial intelligence and law* 31, no. 1 (2023): 91-113.

[45] Shukla, Bharti, Sonam Gupta, Arun Kumar Yadav, and Divakar Yadav. "Text Summarization of Legal Documents Using Reinforcement Learning: A Study." In *Intelligent Sustainable Systems: Proceedings of ICISS 2022*, pp. 403-414. Singapore: Springer Nature Singapore, 2022.

[46] Ramya, R. S., M. Shahina Parveen, Savitha Hiremath, Isha Pugalia, S. H. Manjula, and K. R. Venugopal. "A Survey on Automatic Text Summarization and its Techniques." *International Journal of Intelligent Systems and Applications in Engineering* 11, no. 1s (2023): 63-71.

[47] Xue, X., Y. Hou, and J. Zhang. "Automated Construction Contract Summarization Using Natural Language Processing and Deep Learning." In *ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction*, vol. 39, pp. 459-466. IAARC Publications, 2022.

[48] Zhang J, Zhao Y, Saleh M, Liu PJ (2019) PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization. CoRR abs/1912.08777.

[49] Dodge, J., Sap, M., Marasović, A., Agnew, W., Ilharco, G., Groeneveld, D., Mitchell, M. and Gardner, M., 2021. Documenting large webtext corpora: A case study on the colossal clean crawled corpus. arXiv preprint arXiv:2104.08758.

[50] Raffel C, Shazeer N, Roberts A, Lee K, Narang S, Matena M, Zhou Y, Li W, Liu PJ (2019) Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer.

[51] Lewis, Mike & Liu, Yinhan & Goyal, Naman & Ghazvininejad, Marjan & Mohamed, Abdelrahman & Levy, Omer & Stoyanov, Veselin & Zettlemoyer, Luke. (2020). BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. 7871-7880. 10.18653/v1/2020.acl-main.703.

[52] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D. and Sutskever, I., 2019. Language models are unsupervised multitask learners. OpenAI blog, 1(8), p.9.

[53] https://medium.com/@AkhilPillai1992/the-three-principles-of-prompt-engineering-train-your-ai-to-be-a-pro-98d16194ae40 last accessed on 14-03-2023.

[54] Liu, Pengfei & Yuan, Weizhe & Fu, Jinlan & Jiang, Zhengbao & Hayashi, Hiroaki & Neubig, Graham. (2022). Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. ACM Computing Surveys. 55. 10.1145/3560815.

[55] Lin, Chin-Yew. (2004). ROUGE: A Package for Automatic Evaluation of summaries. Proceedings of the ACL Workshop: Text Summarization Braches Out 2004. 10.

[56] Papineni, Kishore & Roukos, Salim & Ward, Todd & Zhu, Wei Jing. (2002). BLEU: a Method for Automatic Evaluation of Machine Translation. 10.3115/1073083.1073135.

[57] Lavie, Alon and Abhaya Agarwal. "METEOR: An Automatic Metric for MT Evaluation with High Levels of Correlation with Human Judgments." WMT@ACL (2007).

[58] Lavie, Alon and Michael J. Denkowski. "The Meteor metric for automatic evaluation of machine translation." Machine Translation 23 (2009): 105-115.

[59] GPT-4 (openai.com) last accessed on 16-03-2023