Toward Regulatory Compliance: A few-shot Learning Approach to Extract Processing Activities

Pragyan K C[1], Rambod Ghandiparsi[1], Rocky Slavin[1], Sepideh Ghanavati[2], Travis Breaux[3], Mitra Bokaei Hosseini[1]
[1]University of Texas at San Antonio, San Antonio, TX, USA, [2]University of Maine, Orono, ME, USA,
[3]Carnegie Mellon University, Pittsburgh, PA, USA
[pragyan.kc, rambod.ghandiparsi, rocky.slavin, mitra.bokaeihosseini]@utsa.edu, sepideh.ghanavati@maine.edu, breaux@cs.cmu.edu

***Abstract*—The widespread use of mobile applications has driven the growth of the industry, with companies relying heavily on user data for services like targeted advertising and personalized offerings. In this context, privacy regulations such as the General Data Protection Regulation (GDPR) play a crucial role. One of the GDPR requirements is the maintenance of a Record of Processing Activities (RoPA) by companies. RoPA encompasses various details, including the description of data processing activities, their purposes, types of data involved, and other relevant external entities. Small app-developing companies face challenges in meeting such compliance requirements due to resource limitations and tight timelines. To aid these developers and prevent fines, we propose a method to generate segments of RoPA from user-authored usage scenarios using large language models (LLMs). Our method employs few-shot learning with GPT-3.5 Turbo to summarize usage scenarios and generate RoPA segments. We evaluate different factors that can affect few-shot learning performance consistency for our summarization task, including the number of examples in few-shot learning prompts, repetition, and order permutation of examples in the prompts. Our findings highlight the significant influence of the number of examples in prompts on summarization F1 scores, while demonstrating negligible variability in F1 scores across multiple prompt repetitions. Our prompts achieve successful summarization of processing activities with an average 70% ROUGE-L F1 score. Finally, we discuss avenues for improving results through manual evaluation of the generated summaries.**

***Index Terms*—Record of Processing Activities (RoPA), Large Language Models, GDPR Compliance**

I. Introduction

The widespread adoption and use of mobile applications (apps) by users have fueled the expansion of the industry across various domains and categories. In this contemporary era, app-developing companies offer services and products that heavily rely on extensive user data. The analysis and utilization of personal data have become significant drivers of growth, particularly in areas such as targeted advertising and personalized services [1], [2]. To ensure that such data processing aligns with the fundamental rights and freedoms of individuals, privacy regulations such as the General Data Protection Regulation (GDPR) require companies to maintain a record of processing activities (RoPA) [3]. RoPA includes various requirements (see Section II) regarding the details of data processing activities, which involve information such as the name of the data processing activity, its description, purposes, types of data involved, and other relevant entities [4].

Companies with fewer than 250 employees are exempt from these record-keeping obligations if the processing is unlikely to present a risk to the rights of the data subject, no special categories of data are processed, or if the processing occurs only occasionally, as stipulated in the GDPR - Article 30(5) [3]. However, in practice, this exemption is seldom applicable as the term "only occasional" is ambiguous, and companies process users' data regularly. Failure to keep RoPA or provide a comprehensive index to authorities can result in fines up to €10.000.000 or 2% of the total worldwide annual turnover under the GDPR - Article 83(4)(a) [3]. For example, in November 2020, Vodafone Italia was fined for €12.3 million because of a vast range of GDPR violations [5]. In March 2021, Vodafone Spain was issued €8.15 million by the Spanish DPA. Vodafone could have prevented these fines by regularly auditing its data processing activities, establishing data processing agreements with contractors, and maintaining clear records of all relationships with third-party data processors.

Small app-developing companies (i.e., those that are significantly smaller than 250 personnel) not only struggle with these compliance requirements but also face resource limitations, have tight development timelines to compete in the market, and have limited access to legal or privacy experts to make important privacy decisions [6]–[8]. Further, such companies frequently overlook creating in-depth documentation during app development processes [9], [10]. Recent studies show that developers, especially in small or medium-sized companies, mainly consider privacy concepts as an afterthought and are not generally familiar enough with those concepts [7], [8], [11]–[20]. In addition, studies show that many developers are only familiar with a limited number of privacy-by-design approaches [21]. A recent survey reviewed the RoPA practices of 30 public organizations, where only 7 (23%) of the RoPA practices contained sufficient detail for the processing activities and their purposes [22]. Therefore, there is a significant challenge for small app-developing companies to create and maintain RoPA to comply with the GDPR's processing activities requirements and avoid potential legal repercussions.

The current research domain predominantly focuses on for-

mally representing RoPA using knowledge bases and semantic models to enhance automatic accountability checking and querying [23], [24]. This trend highlights the lack of emphasis on the actual creation of RoPA. Huth et al. suggested leveraging Enterprise Architecture (EA) for RoPA creation [4]. The presence of EA signifies organizational maturity, supported by adequate resources and personnel capable of sustaining EA development and maintenance. This scenario contrasts sharply with small app-developing companies, which often operate with limited personnel and resources, leading to a lack of comprehensive documentation [9], [10].

To assist such developers in creating RoPA, we propose a framework to generate segments of RoPA for existing mobile apps. To address the lack of documentation in small app-developing companies, the framework takes advantage of usage scenarios provided by users as the primary source of processing activities. These scenarios can be elicited from end users through simple interactions with the app. This approach makes such resources accessible and attractive, particularly for smaller app development companies that lack the resources for more complex elicitation techniques.

Empirical research has shown that users *spontaneously* provide explanatory scenarios where they describe the app inefficiencies and desired features alongside the processing activities [25], [26]. Therefore, we propose an *extractive summarization method* specifically for the purpose of filtering and extracting the processing activities from the whole scenario text, which also entails app deficiencies and desired features. We further design experiments to investigate how well large language models (LLMs) can assist with extractive summarization to generate segments of RoPA.

The contributions of this paper are: (1) a 50-scenario corpus with summarized processing activities; (2) an empirical evaluation of utilizing the GPT-3.5 Turbo as an instance of LLMs for extractive summarization; (3) an evaluation of different parameters affecting the LLM instance in extractive summarization task; (4) a manual evaluation of summaries extracted using the LLM instance.

The remainder of this paper is organized as follows: Section II presents background & related work; Sections III & IV entail the approach & experiment designs; Sections V & VI entail results & discussion; Sections VII & VIII contain threats to validity & conclusion.

II. Background and Related Work

A. Record of Processing Activities (RoPA)

To comply with GDPR requirements [3] in maintaining records of processing activities, companies are obligated to furnish records of: (1) the data controller's name and contact; (2) the processing purposes (e.g., the processing of contact data of suppliers for order management); (3) a description of the categories of data subjects (e.g., customers, suppliers, and employees), personal data (e.g., health data), and recipients; (4) the latest deadlines for the cancellation of the different categories of data; (5) a description of the technical and organizational security measures [4], [27].

Huth et al. demonstrated how existing Enterprise Architecture (EA) can be enhanced with the requisite information to create and sustain a RoPA. EA encompasses a cohesive framework of principles, methodologies, and models utilized in designing and implementing an enterprise's organizational structure, business processes, information systems, and infrastructure [28]. Owning and maintaining EA models signifies a level of maturity within an organization, with ample resources and personnel capable of contributing to the development and maintenance of EA. However, our research targets small app-developing companies with limited personnel and resources, often lacking comprehensive documentation and EA models.

Presently, the majority of RoPA are manually created and maintained, typically presented informally in Word documents or Excel files, and often shared with the public in their original formats or as PDFs [4], [23]. To address this informality in presentation, Martinez et al. introduced a RoPA knowledge graph incorporating both legal requirements and practical insights from the privacy and data protection community [23]. This knowledge graph offers companies a formal means of presenting their RoPA, facilitating stakeholders such as legal entities and customers in reading, evaluating, and querying the information contained within RoPA. Ryan and Brennan developed a common semantic model to represent a machine-readable RoPA, which can be used in automated accountability systems [24]. While these works are focused on RoPA representation, we focus on creating segments of RoPA tailored for small app-developing companies that lack comprehensive documentation throughout their app development processes.

B. Legal Compliance in RE

Data protection officers (DPOs) and privacy engineers are pivotal experts in privacy requirements engineering, particularly for companies aiming to adhere to legal regulations [29]. However, small app-developing companies may find it challenging to obtain the necessary knowledge to demonstrate compliance with regulatory requirements. Therefore, researchers have proposed methods and tools aimed at automating the creation of artifacts and materials required for legal compliance [30]–[34]. For instance, Herwanto et al. proposed a tool to help development teams automatically identify assets and privacy-related entities (such as data subject, processing, and personal data entities) from user stories [29] and generate data flow diagrams [35]. Ghanavati et al. developed a Legal-URN framework for extracting legal requirements and establishing and evaluating compliance [36], [37].

C. Scenarios

Scenarios are detailed descriptions of system behaviors, often depicted as sequences of steps, typically from a user's viewpoint [38]–[41]. Concrete scenarios are essential to an understanding of the operational concept of a system [25], [42]. They encapsulate the behavior patterns of an existing system and serve to enhance comprehension of work practices and business processes [41]. Further, these scenarios represent the aggregation of interactions, each of which is a short

sequence of goal-oriented activities to achieve [43]. In a recent study, Huang et al. proposed a framework for developers to measure the privacy risk associated with the information users provide to the app using user-authored scenarios [44]. We follow their survey design to collect usage scenarios describing processing activities and steps user take to accomplish their specific goals.

D. Extractive Summarization

Extractive summarization extracts text segments from an input text, capturing essential information related to the *key concepts* mentioned in the input text [45], [46]. Unlike abstractive summarization, which rephrases the information, extractive summarization preserves the original text, ensuring factual accuracy and clarity [47]. This characteristic makes it particularly suitable for factual text like our work, where preserving precise details about processing activities is crucial.

Jadhav et al. introduced SWAP-NET, a sequence-to-sequence model for extractive summarization [48]. SWAP-NET is designed to identify salient sentences and keywords within a document, combining them to create an extractive summary. Similarly, Nallapati et al. developed a method using an RNN-based binary classifier to determine whether a sentence should be included in the summary [49]. This method relies on the sentence's content, its significance within the document, its novelty compared to previously selected sentences, and additional positional features. Expanding on these approaches, Yadav et al. proposed a textual graph-based technique [50], where document sentences form the nodes of a graph, with edges representing associations between sentences. The summary is generated based on the sentence weight and the average weight of the textual graph. Diverging from the neural network and graph-based methodologies, Mishra et al. approached summarization as a question-answering task [51]. They leveraged LLMs to generate pseudo-labels for dialogue, which were then used to fine-tune a chat summarization model, effectively transferring knowledge from a large LLM to a more specialized, smaller model. In contrast to these works, our method conceptualizes extractive summarization as an event-argument extraction task. By leveraging LLMs, we generate summaries of scenarios that are both more precise and less cluttered with irrelevant information.

E. Large Language Models

In recent years, the NLP community has witnessed substantial changes due to the introduction of LLMs. In the RE community, Fantechi et al. [52] evaluated the ChatGPT's ability to identify ambiguity in requirement text and compared its performance with QuARS, a traditional rule-based NLP tool [53]. Ruan et al. incorporated ChatGPT-based zero-shot learning to extract requirement models from requirement texts and compose them using predefined rules [54]. Gorer et al. used prompt engineering to generate business domain descriptions, linear scripts, and conversation pieces focused on certain types of mistakes in requirements elicitation interviews [55].

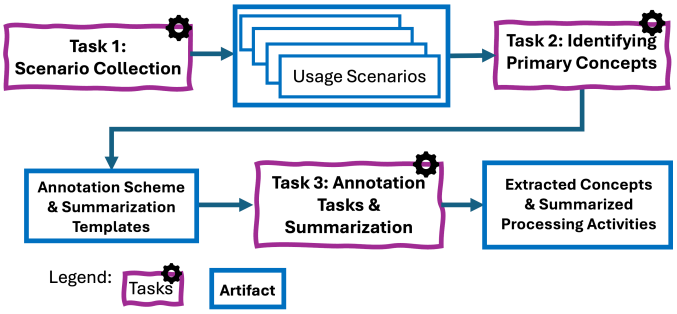

Fig. 1. The Overview of Extracting Processing Activities

III. Approach

We aim to generate segments of RoPA from usage scenarios provided by users as the primary data source entailing processing activities. The process of collecting scenarios and extracting processing activity summaries using extractive summarization is shown in Figure 1.

In Task 1, we first publish a survey to collect usage scenarios. In Task 2, given the usage scenarios as input, we identify the primary concepts that should be described in the processing activity summary. We employ a grounded analysis method to identify key concepts and design two *artifacts* accordingly: (a) an annotation scheme; and (b) controlled-natural language templates to represent the summaries. In Task 3, we annotate scenarios using the annotation scheme and construct the summaries using the templates.

A. Task 1: Scenario Collection

To construct a corpus of usage scenarios, we follow the survey design method from Huang et al. [44]. To this end, we publish a survey and invite mobile app end-users to upload a screenshot from a mobile app, describing a usage scenario for the screenshot in at least 150 words, and answering some questions. The survey targets apps from various domains. Survey participants are first asked to select a mobile app they use frequently. Next, participants select a specific screen in the app and take a screenshot. Participants are instructed to select screens with the following properties: (a) emphasis on the core functionality of the apps; (b) not the app's homepage; (c) not the app's login page; and (d) not the app's settings page. We also provide the participants with examples of ideal and bad screenshots as further guidance in the instructions. To submit the screenshot from mobile phones, participants scan a QR code that navigates them to a web page where they upload, redact any personal information, and submit the screenshot. Fourth, the submitted screenshot is loaded to the main survey page, where participants are asked to write a usage scenario of at least 150 words that contains: (a) a description of the goal the user wants to achieve through the screen; (b) steps that they take to get to the screen on the app; and (c) the steps they take to achieve the goal once at the screen. The participants are also provided with an example of a scenario in the instructions. Figure 2 illustrates an example of a scenario.

I am trying to sign up for this calorie counting app. I am in the registration process and it is asking me for my personal information. I would want to maintain my current healthy weight and track my calories through out the day. I do not want to lose or gain weight through the use of this app. My interactions with this app would be to monitor it as I go through out the day and track how many calories I am consuming each day. The app is asking for my age, it also asks for my country of residence. I am not sure why it is asking for that and my zipcode as well. I feel somewhat uncomfortable providing my zip code because I don't know why the app needs that or what it will do with that information. I might use this app or might not.

Fig. 2. Example Scenario

The survey was published on the AMT platform. Eligible participants were required to have completed at least 5,000 Human Intelligence Tasks (HITs), possess an approval rating exceeding 97%, and reside in the US. Upon completion of the survey, workers received a compensation of $4.00. The survey yielded a total of **50** usage scenarios.

B. Task 2: Identifying Primary Concepts

Usage scenarios may articulate actual or desired behaviors, indicating the mood of the scenario sentences [25], [38], [56]. For instance, in the scenario depicted in Figure 2, the user expresses uncertainty about the app's need for a zip code. This expression conveys the user's desire for transparency and underscores the importance for app developers to incorporate a transparency requirement. Users' desires and preferences trigger change management & maintenance activities. However, when constructing RoPA, our focus lies in identifying and extracting behaviors that represent the actual interactions between the system, users, and other entities. Thus, we propose an extractive summarization method tailored to filter processing activities from scenario texts, which may also include app deficiencies and desired features.

To construct an extractive summarization method, we first need to identify the primary concepts involved in the actual behaviors stated in scenarios. For this reason, the first and the last authors conducted a manual grounded analysis of five randomly selected scenarios obtained in Task 1. Our grounded analysis yielded the following key concepts within the scenarios: (1) categories of actions performed; (2) actors (user, app, or external entities); (3) personal data types; (4) purposes of the actions; (5) recipients of data types (i.e., external entities); and (6) the User Interface (UI) elements that users interact with. Furthermore, we analyzed the action verbs present in the scenarios. As outlined in Task 1, users were instructed to provide the goal they aim to achieve, a list of steps they take to accomplish that goal, and data types used or provided during their interaction with the app (e.g., health data, financial data). This structural approach led to the identification of three distinct categories of actions within the scenarios:

- *Goal Actions*: Verbs or verb phrases that articulate the user's overarching goal to be achieved through the selected UI screen.
- *Step Actions*: Verbs or verb phrases that detail the user's interaction with a specific UI component (e.g., button).
- *Data Practice (DP) Actions*: Verbs or verb phrases that specify the events or actions related to the collection, usage, transfer, and retention of *data types*.

Based on this analysis, we designed three controlled natural language (NL) templates that summarize sentences involving each category of actions. It is worth noting that a sentence in the scenario may encompass multiple action verbs from different categories. The use of these templates enables us to differentiate the concepts (e.g., data types, purposes, UI components, and actors) associated with each action verb and construct a controlled NL sentence that represents the action verb along with its corresponding concepts. This structured approach facilitates a concrete representation of the diverse actions and their associated concepts within the scenarios.

Goal Template	User <goal-action> <data-type-args> <purpose-args> <external-entity-args>.
Step Template	<actor> <step-action> <UI-args> <purpose-args> <external-entity-args>.
DP Template	<actor> <DP-action> <data-type-args> <UI-args> <purpose-args> <external-entity-args>.

Fig. 3. Controlled NL Templates for Summarization

Figure 3 lists the templates for three categories of actions. For *goal actions*, the template is composed of the token "user" that represents the actor, the goal action verb presented in a third person singular format, followed by three placeholders for data-type, purposes, and external-entity concept arguments (i.e., ⟨data-type-args⟩ ⟨purpose-args⟩ ⟨external-entity-args⟩). The template sentence for *step actions* is composed of a placeholder token (i.e., ⟨actor⟩) for the actor, the step action verb presented in a third person singular format, followed by three placeholders for the UI component, purposes, and external-entity concepts (i.e., ⟨UI-args⟩ ⟨purpose-args⟩ ⟨external-entity-args⟩). The template sentence for *DP actions* is composed of a placeholder token (i.e., ⟨actor⟩) for the actor, the DP action verb presented in a third person singular format, followed by three placeholders for data-type, UI component, purposes, and external-entity concepts (i.e., ⟨data-type-args⟩ ⟨UI-args⟩ ⟨purpose-args⟩ ⟨external-entity-args⟩).

Goal: User <tgr>signs up</tgr>.
DP: App <tgr>asks</tgr> personal information in the registration process.
DP: User <tgr>tracks</tgr> calories.
DP: App <tgr>asks</tgr> age.
DP: App <tgr>asks</tgr> country of residence.
DP: App <tgr>asks</tgr> zipcode.
DP: User <tgr>provides</tgr> zip code.

Fig. 4. Filled Templates for the Example Scenario

Figure 4 shows the summarized version of our example scenario, with filled templates for *Goal* & *DP*, where the placeholder tokens are replaced by the arguments whenever possible. To specify the desired action for summarization, a special token ⟨tgr⟩ is employed surrounding the action verbs. If there are multiple arguments for the same slot (e.g., ⟨data-type-args⟩), we connect the arguments with the token "and".

C. Task 3: Annotation & Summarization

To generate processing activity summaries for all 50 scenarios, we developed an annotation tool and published it on AMT. The coding frame is illustrated in Figure 5 and is based on the primary concepts identified in Task 2. This coding frame contains the following codes: *action verb or verb phrase*, *data type*, *purposes of the actions*, *external entities as recipients of data types*, and *UI component*. The annotators are instructed to first highlight an action verb in a sentence and then identify two attributes for the action: the action category (i.e., goal, step, or data practice (DP)) and the actor (i.e., user, app, or external entity). Next, the annotators are instructed to identify the corresponding concepts for the annotated action verb, such as data types, purposes, external entities, and UI components.

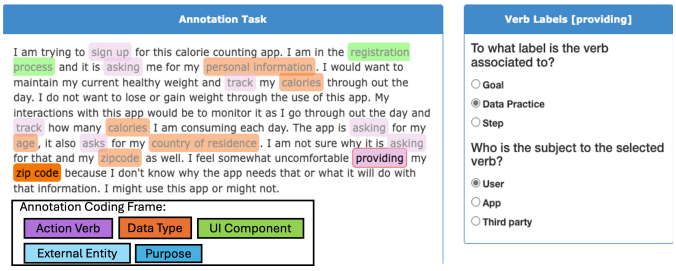

Fig. 5. Annotation Coding Frame

Among 50 scenarios, we generated a HIT per scenario. Two annotators (the first & last authors) were assigned to the HITs such that each HIT would be annotated by two respondents. An annotator typically took about 9.65 minutes to complete one HIT. To facilitate reconciliation between the two annotators, we use tokenization and present each scenario as a sequence of tokens that are annotated with specific labels.

We annotated the 50 scenarios in four rounds. In round one, both authors coded a random sample of 10 scenarios using the coding frame, which yielded in Cohen Kappa of 0.50 [57], which is a moderate agreement [58]. The authors next discussed disagreements and identified heuristics to clarify boundary and edge cases. In round two, the authors coded a new random sample of 10 new scenarios using the new heuristics, yielding in Cohen Kappa of 0.68. The authors reconvened, examined disagreements, and developed the following heuristics:

- H1: Step action verbs involve UI components rather than specifying data types.
- H2: Goal action verbs are introduced by a verb phrase, such as "I want to$\cdots$" or "The app allows me to$\cdots$", indicating a clear goal.
- H3: Users may describe app-wide steps or data practices rather than specific screen-related actions. Therefore, those action verbs cannot be considered as pre/post-conditions of the main goal for the scenario.
- H4: Extra information detailing data types, purposes, and external entities (e.g., pronouns and articles) is not annotated. Only the core/head of a noun phrase is selected, excluding additional modifiers.
- H5: Scenarios with lists of elements separated by commas, conjunctions/disjunctions are annotated individually.

In round three, the authors re-coded the second sample using the heuristics to reach a Kappa of 0.74. In round 4, the authors coded a new random sample of 10 scenarios using the heuristics and coding frame to yield a Kappa of 0.95 (almost perfect agreement) [58]. Due to this high level of agreement, the remaining scenarios were then coded by the last author using the coding frame and accompanying heuristics. To this point, we have a total of 400 action verbs from our reconciliation of 50 annotated scenarios. Of these, 64 are linked to goal, 83 to step, and 253 to DP.

IV. Experiment Design

We investigate how LLMs can extract summaries of processing activities from scenarios automatically. We explore the effects of the number of training examples in few-shot learning, repetition, and ordering of the examples on LLMs performance for extractive summarization. From the 50 scenarios in the corpus, we identify sentences that contain goal, step, or DP action verbs, yielding three datasets containing 64, 83, and 253 sentences, respectively. We partition each dataset into training, validation, and testing sets using a ratio of 60:20:20. Using the training, validation, and testing sets for all three datasets (i.e., goal, step, and DP), we design two experiments to address the following research questions.

RQ1. How can the number of examples in few-shot learning change the performance of extractive summarization?
RQ2. How consistent are LLMs when repeating the same experiment in terms of providing identical results?
RQ3. How crucial is the order sensitivity in few-shot learning?
RQ4. To what extent can LLMs effectively extract RoPA concepts and construct summaries from usage scenarios?

To design our experiments, we utilize GPT-3.5 Turbo as an instance of LLMs. We design Experiment 1 to investigate the effect of the number of examples in few-shot learning on the validation set. We further evaluate the effect of repetition on the results generated by GPT-3.5 Turbo. Building on the optimal number of examples identified in Experiment 1, we formulate Experiment 2 to investigate the order & arrangement of examples within the prompt. Using the results from both experiments, we configure a few-shot learning model and evaluate it on the testing set for each dataset.

A. Experiment 1

We aim to explore the significance of the number of examples in few-shot learning prompts and the resulting variation when the experiment is repeated. For each dataset, we initiate this experiment by randomly selecting 10 examples from the training set. Following this, we structure prompts in a specific format to guide the Chat Completion API of GPT-3.5 Turbo as shown in Figure 6. This format allows us to change the number of examples from zero to 10 to create *11 unique prompts*. Additionally, to gauge response consistency, each prompt was repeated 10 times (10 different GPT API calls). Note that we consider the zero-shot prompt as the *baseline*

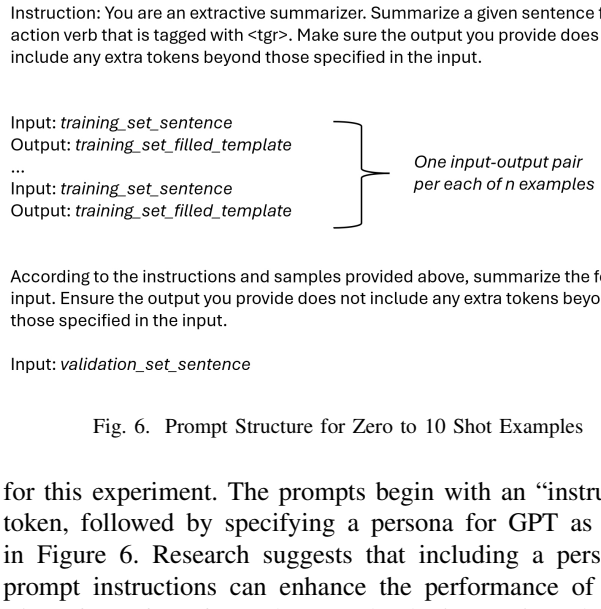

Fig. 6. Prompt Structure for Zero to 10 Shot Examples

for this experiment. The prompts begin with an "instruction" token, followed by specifying a persona for GPT as shown in Figure 6. Research suggests that including a persona in prompt instructions can enhance the performance of LLMs [59], [60], [61], [62]. Subsequently, the instructions detail the summarization task, concluding with a constraint to ensure that the summary output only contains tokens from the input. Examples are then provided based on the number of shots used in the experiments. Additionally, an excerpt is provided for GPT to reference the initial instruction in generating an output for the given input [62], [59]. It's worth noting that altering and assessing this prompt's structure can offer insights into its impact on results. However, in this paper, we narrow our focus to only exploring how variations in the number of examples and the order of examples influence the result.

Evaluation is conducted using six different metrics, including ROUGE-1, ROGUE-2, ROUGE-L, ROGUE-S, METEOR, and BERTScore, on the testing set. ROUGE-1 focuses on individual word overlap, ROUGE-2 considers adjacent word pairs, while ROUGE-L assesses the longest common subsequence, allowing for word reordering [63]. ROUGE-S incorporates skip-bigrams, introducing flexibility in word order for sentence-level coherence [64]. METEOR extends evaluation beyond exact word matches, considering synonymy and stemming for a more nuanced semantic assessment [65]. Lastly, BERTScore captures the detailed semantic similarities between the provided reference and prediction [66].

B. Experiment 2

Our objective is to determine if organizing examples in a specific order in prompts improves the overall effectiveness of few-shot learning models. To achieve this, we explore all order permutations of examples (i.e., number of shots) derived from Experiment 1 for each dataset (i.e., goal, step, and DP). Using the order permutations and the number of examples derived from Experiment 1, we create unique prompts and evaluate each prompt on the validation sets of each dataset. Similar to Experiment 1, we use GPT-3.5 Turbo and evaluate each prompt on the validation sets. The prompts in this experiment have a similar structure as Figure 6.

V. Evaluation & Results

Experiment 1 Results: Table I displays the mean F1-scores for six evaluation metrics across different numbers of examples (i.e., zero to 10). The means are computed based on 10 repetitions of each unique prompt. Across all three datasets, there is a noticeable improvement in performance from zero-shot (i.e., baseline) to one-shot-learning for all metrics. ROUGE-L is a more restrictive metric that prioritizes the recall of content units shared between the input text and the output summary. It highlights the significance of capturing the reference text's overall content and meaning. Therefore, we opt for the ROUGE-L score for detailed analysis. Figure 7 illustrates the box plots representing the mean Rouge-L F1 scores for the number of examples changing from zero to 10, along with the variance for each shot. Upon examination to address **RQ1**, we note that the variance remains insignificant for all repetitions of each unique prompt.

To address **RQ2** and determine the optimal number of examples for few-shot learning, we calculate the cumulative mean of Rouge-L F1 score for zero to 10 shots. Additionally, we calculate the standard error for each cumulative mean. We expect that the standard error decreases as the number of examples increases. Figure 8 presents a line plot depicting the relationship between the standard error and the number of examples. As expected, an increase in the number of examples results in a decrease in the standard error. To promote reproducibility, we establish an acceptable standard error threshold of 0.05. This threshold may vary depending on researchers' available resources to augment the number of examples in few-shot learning. The dashed reference line in Figure 8 represents this 0.05 acceptable error threshold. Considering this level of error, we select seven, nine, and six examples for the goal, step, and DP datasets. Our results can be found online [1].

Experiment 2 Results: Figure 9 illustrates the distribution of ROUGE-L F1 scores for the order permutations of 7!, 9!, and 6! for the goal, step, and DP validation sets. Each box plot illustrates minimal variability in the distribution of F1 scores, as indicated by the size of the box and whiskers. To address **RQ3**, we report the variance for the goal, step, and DP validation sets as 0.00, 0.06, and 0.00, respectively.

To optimize the permutation process and minimize execution time, we leverage the "ThreadPoolExecutor" library for efficient task management across multiple threads concurrently. This strategy ensures that tasks are assigned to idle threads as they are completed, maximizing parallelism and speeding up the overall process. Additionally, we distribute the permutation workload across multiple files, enabling parallel execution within each file and across different files simultaneously. We utilized a high-performance virtual machine equipped with 80 CPU cores and 128GB of memory, further enhancing computational capabilities. The execution time for goal and DP permutations was approximately 24 hours, while for the step validation set, it extended slightly beyond 48 hours.

[1] https://tinyurl.com/DataForESPRE2024

TABLE I
PERFORMANCE METRICS FOR DIFFERENT NUMBER OF EXAMPLES

	ROUGE-1			ROUGE-2			ROUGE-L			ROUGE-S			METEOR			BERTScore		
Shots	Goal	Step	DP	Goal	Step	DP	Goal	Step	DP	Goal	Step	DP	Goal	Step	DP	Goal	Step	DP
0	0.26	0.22	0.22	0.10	0.12	0.10	0.25	0.21	0.22	0.09	0.11	0.09	0.4	0.45	0.31	0.89	0.89	0.88
1	0.62	0.62	0.53	0.47	0.44	0.35	0.62	0.61	0.53	0.43	0.42	0.32	0.68	0.75	0.66	0.95	0.95	0.94
2	0.63	0.68	0.54	0.49	0.58	0.36	0.63	0.68	0.54	0.45	0.54	0.32	0.68	0.79	0.63	0.95	0.96	0.94
3	0.63	0,71	0.58	0.47	0.60	0.41	0.63	0.71	0.57	0.43	0.58	0.38	0.68	0.79	0.68	0.95	0.96	0.95
4	0.65	0.72	0.56	0.47	0.63	0.37	0.64	0.72	0.55	0.43	0.59	0.34	0.69	0.79	0.67	0.95	0.96	0.94
5	0.67	0.71	0.61	0.49	0.63	0.45	0.67	0.71	0.60	0.45	0.59	0.41	0.73	0.78	0.70	0.95	0.96	0.95
6	0.71	0.72	0.63	0.53	0.64	0.48	0.71	0.72	0.63	0.48	0.60	0.45	0.77	0.79	0.74	0.96	0.96	0.95
7	0.71	0.72	0.64	0.51	0.63	0.49	0.71	0.72	0.63	0.46	0.61	0.45	0.75	0.79	0.73	0.96	0.96	0.95
8	0.69	0.71	0.63	0.51	0.62	0.47	0.69	0.71	0.63	0.47	0.59	0.44	0.73	0.80	0.73	0.96	0.96	0.95
9	0.70	0.70	0.62	0.52	0.61	0.47	0.70	0.70	0.62	0.48	0.58	0.44	0.72	0.78	0.71	0.96	0.96	0.95
10	0.74	0.70	0.64	0.58	0.61	0.48	0.74	0.70	0.63	0.54	0.57	0.44	0.75	0.77	0.73	0.96	0.96	0.95

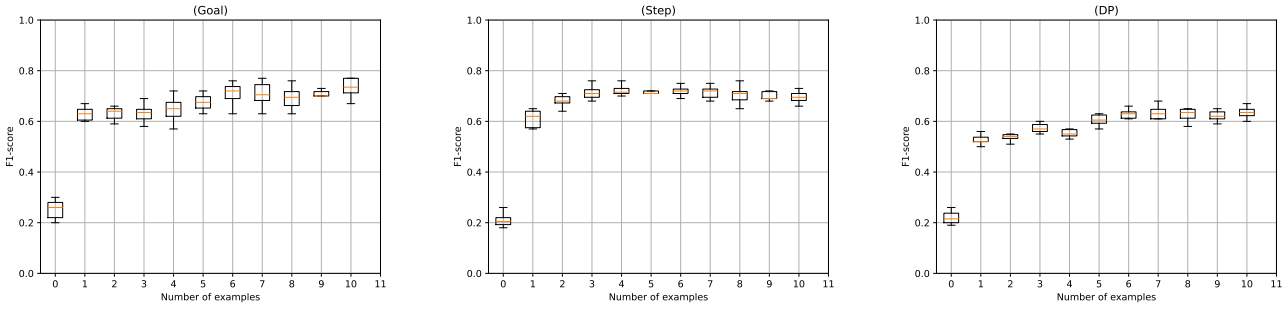

Fig. 7. Repetition for Goal, Step, and DP

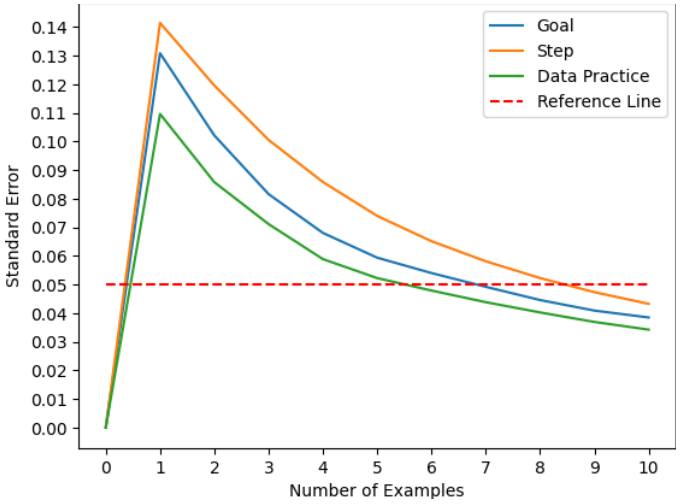

Fig. 8. Standard Error Line Plot

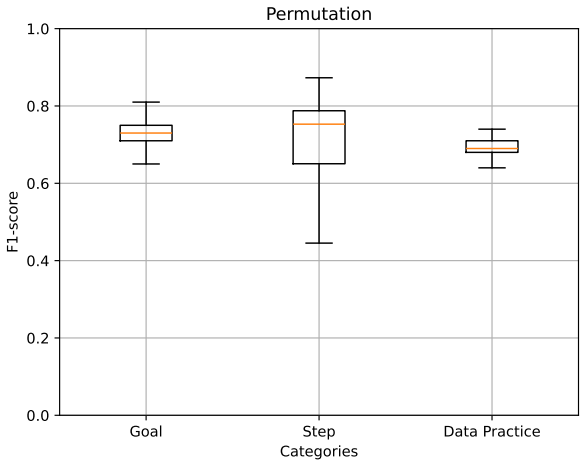

Fig. 9. Permutations Performed on Each Dataset

The cumulative cost of completing these API calls for the permutations amounted to roughly $3,500.

Testing Set Results: Experiment 1 & 2 results reveal that the number of examples within a prompt significantly influences performance, while repetition and order permutations have a minor impact on consistency & performance. Consequently, we leverage these findings and use one prompt per dataset, comprising seven, nine, and six examples for the goal, step, and DP datasets, respectively. We assess each test set using these prompts and GPT-3.5 Turbo to address **RQ4**. Table II presents the results for the testing sets.

TABLE II
RESULTS ON TESTING SET

Metric	ROUGE-1	ROUGE-2	ROUGE-L	ROGUE-S	METEOR	BERTScore
Goal	0.75	0.48	0.75	0.36	0.80	0.95
Step	0.75	0.63	0.76	0.49	0.85	0.95
Data Practice	0.62	0.42	0.61	0.30	0.69	0.94

VI. DISCUSSION

We proceed to examine the research questions in light of our findings. **RQ1** delves into how varying the number of examples in few-shot learning impacts the performance of extractive summarization. We ascertain the optimal number

of shots required by analyzing the accumulative mean and standard error of ROUGE-L scores. Further, we set zero-shot learning as our baseline for Experiment 1. We observe that zero-shot learning performs poorly for this specific task across all datasets. Therefore, it is crucial to provide examples for such domain-specific tasks.

RQ2 evaluates the reliability of GPT-3.5 Turbo by examining its consistency in producing identical results when the same experiment is repeated. Through repetition of each shot 10 times, our analysis presents a minimal variance, approaching zero. This observation underscores the consistency exhibited by GPT-3.5 Turbo across repeated iterations, affirming its ability to generate consistent outputs.

RQ3 investigates the sensitivity of order in few-shot learning through permutation analysis. Using optimal example counts of seven, nine, and six for goal, step, and DP, respectively, we generate permutations of 7!, 9!, and 6! for each dataset. Subsequently, these permutations are evaluated on a validation set. Upon comparing the ROUGE-L F1 scores across all permutations, we observe minimal variance. This leads us to conclude that the order of examples in GPT-3.5 Turbo has a negligible impact on few-shot learning outcomes.

To address **RQ4** and evaluate the effectiveness of GPT-3.5 Turbo in extracting RoPA concepts and generating summaries, we employ the optimal number of examples to craft three distinct prompts, which are then applied to the testing sets. Our prompts achieve successful summarization of processing activities, yielding an average ROUGE-L F1 score of 70%. To further understand the metric scores, we expand our exploration with a manual qualitative study. In this study, both the first and last authors compared the ground truth summary templates with the summaries generated by GPT-3.5 Turbo for each sentence in the three testing sets. This comparison resulted in the development of a coding scheme comprising six code categories: (1) additional modifiers and adjectives; (2) incorrect action verb or subject; (3) missing data type; (4) missing purpose; (5) missing UI Component; and (6) summary contains more than two verbs. After completing the coding exercise, both annotators reconciled any disagreements and analyzed the results. From the analysis of 81 sentences in the testing sets across all three datasets, the following ratios were observed for the coding categories 1-6: 29/81, 20/81, 2/81, 6/81, 1/81, and 6/81. Coding category (1) exhibits the highest ratio, indicating instances where GPT-3.5 Turbo extracted additional tokens from the sentence compared to human extractions in the ground truth. For example, GPT-3.5 Turbo generates "User gets regular promotions offered," for a testing set sentence, "If I opt in, I would probably be able to ⟨tgr⟩get⟨/tgr⟩ regular promotions offered to me," with the ground truth summary template as "User gets promotions". Despite these disparities, we note that the summaries remain constrained to the sentence tokens as instructed in the prompts.

VII. Threats to Validity

Internal Validity concerns whether the inferences drawn from the experiments are valid. LLMs are generative models; thus, the responses generated by repeating the same prompt may vary. We assess the consistency of F1 scores by repeating each prompt 10 times. Our findings indicate that the variance in F1 scores across different prompts is insignificant. We also conducted a manual evaluation study and identified various reasons why the generated summary did not align with the summary templates. This insight can inform modifications to prompts for the summarization task. Additionally, the survey participants and the quality of the scenarios can affect the validity of the generated RoPA segments. Finally, further evaluation is required to analyze the usability & effectiveness of our framework in real-world industry examples.

External Validity concerns the extent to which the results generalize beyond the experimental setting. While we exclusively analyzed GPT-3.5 Turbo as an instance of LLMs, further experiments are necessary to compare our findings with other LLMs, such as LLaMA and T5. In addition, GPT-3.5 Turbo undergoes updates in every few months. These updates can also significantly alter the model output over time. Apart from different LLMs and their updates, the use of fine-tuning instead of few-shot learning may yield different results. Further, the generated summaries rely on an input sentence where the action verb is tagged with a specific token. However, further analysis and NLP models are required to identify and label action verbs in the scenario sentences. In this work, we only focus on the effect of the number of examples in few-shot learning and their order. However, further experiments are required to evaluate the effect of prompt structure and instruction on the results.

Reliability indicates that the researchers' approach is consistent across different researchers and different projects [67]. To ensure the reliability of our research, we measure the inter-coder agreement using Cohen Kappa [57] for the scenario labeling task.

VIII. Conclusion & Future work

In this paper, we propose a framework to generate segments of RoPA, which is required for GDPR compliance. Our framework is tailored to support small app-developing companies in their efforts to comply with GDPR and avoid regulatory fines. The framework utilizes usage scenarios that contain processing activities. We demonstrate the efficacy of few-shot learning with GPT-3.5 Turbo for generating summaries of processing activities given usage scenarios. Further analysis is needed to evaluate the fine-tuning strategy, utilization of other open-source LLMs, such as LLaMA and T5, and changes to the prompt structure. Our framework currently lacks an initial phase for identifying and labeling processing activity action verbs within the usage scenarios. As a result, we plan to extend the framework in the future to incorporate models such as Named Entity Recognition (NER) to identify and label these action verbs accurately. Moreover, we intend to conduct empirical studies to evaluate the effectiveness and practical applicability of our framework in extracting RoPA details.

REFERENCES

[1] X. Wang, X. Qin, M. B. Hosseini, R. Slavin, T. D. Breaux, and J. Niu, "Guileak: Tracing privacy policy claims on user input data for android applications," in *Proceedings of the 40th International Conference on Software Engineering*, 2018, pp. 37–47.

[2] R. Slavin, X. Wang, M. B. Hosseini, J. Hester, R. Krishnan, J. Bhatia, T. D. Breaux, and J. Niu, "Toward a framework for detecting privacy policy violations in android application code," in *Proceedings of the 38th International Conference on Software Engineering*, 2016, pp. 25–36.

[3] European Parliament and Council of the European Union, "General data protection regulation (GDPR)," pp. 1–88, May 2016, originally published on April 27, 2016. [Online]. Available: https://gdpr-info.eu/

[4] D. Huth, A. Tanakol, and F. Matthes, "Using enterprise architecture models for creating the record of processing activities (art. 30 gdpr)," in *2019 IEEE 23rd EDOC*. IEEE, 2019, pp. 98–104.

[5] "Biggest gdpr fines," https://www.tessian.com/blog/biggest-gdpr-fines-2020/.

[6] R. Balebako and L. Cranor, "Improving app privacy: Nudging app developers to protect user privacy," *IEEE Security & Privacy*, vol. 12, no. 4, pp. 55–58, 2014.

[7] R. Balebako, A. Marsh, J. Lin, J. I. Hong, and L. F. Cranor, "The privacy and security behaviors of smartphone app developers," 2014.

[8] S. E. Noura Alomar and and J. L. Fischer, "Developers say the darnedest things: Privacy compliance processes followed by developers of child-directed apps," *Proceedings on Privacy Enhancing Technologies*, vol. 2022, no. 4, 2022.

[9] G. Wagenaar, S. Overbeek, G. Lucassen, S. Brinkkemper, and K. Schneider, "Working software over comprehensive documentation–rationales of agile teams for artefacts usage," *Journal of software engineering research and development*, vol. 6, pp. 1–23, 2018.

[10] A. Hess, P. Diebold, and N. Seyff, "Towards requirements communication and documentation guidelines for agile teams," in *2017 ieee 25th international requirements engineering conference workshops (rew)*. IEEE, 2017, pp. 415–418.

[11] K. Bednar, S. Spiekermann, and M. Langheinrich, "Engineering privacy by design: Are engineers ready to live up to the challenge?" *The Information Society*, vol. 35, no. 3, pp. 122–142, 2019.

[12] I. Hadar, T. Hasson, O. Ayalon, E. Toch, M. Birnhack, S. Sherman, and A. Balissa, "Privacy by designers: Software developers' privacy mindset," *Journal of Empirical Software Engineering*, vol. 23, no. 1, p. 259–289, Feb. 2018.

[13] A. Ekambaranathan, J. Zhao, and M. Van Kleek, ""money makes the world go around": Identifying barriers to better privacy in children's apps from developers' perspectives," in *Proceedings of the 2021 CHI*, 2021, pp. 1–15.

[14] S. Spiekermann, J. Korunovska, and M. Langheinrich, "Inside the organization: Why privacy and security engineering is a challenge for engineers," *Proceedings of the IEEE*, vol. 107, no. 3, pp. 600–615, 2018.

[15] S. Spiekermann-Hoff, J. Korunovska, and M. Langheinrich, "Understanding engineers' drivers and impediments for ethical system development: The case of privacy and security engineering," 2018.

[16] A. Dalela, S. Giallorenzo, O. Kulyk, J. Mauro, and E. Paja, "A mixed-method study on security and privacy practices in danish companies," *ArXiv*, vol. abs/2104.04030, 2021.

[17] M. Tahaei and K. Vaniea, ""developers are responsible": What ad networks tell developers about privacy," in *Extended Abstracts of the 2021 CHI*, 2021, pp. 1–11.

[18] M. Green and M. Smith, "Developers are not the enemy!: The need for usable security apis," *IEEE Security & Privacy*, vol. 14, pp. 40–46, 2016.

[19] M. Tahaei, A. Jenkins, K. Vaniea, and M. Wolters, ""i don't know too much about it": On the security mindsets of computer science students," in *Socio-Technical Aspects in Security and Trust*, T. Groß and T. Tryfonas, Eds. Springer International Publishing, 2021.

[20] M. Prybylo, S. Haghighi, S. T. Peddinti, and S. Ghanavati, "Evaluating privacy perceptions, experience, and behavior of software development teams," 2024.

[21] M. Tahaei, T. Li, and K. Vaniea, "Understanding privacy-related advice on stack overflow," *Proceedings on PETs*, vol. 2022, no. 2, pp. 114–131, 2022. [Online]. Available: https://doi.org/10.2478/popets-2022-0038

[22] "Castlebridge register of processing activities (2020)," https://castlebridge.ie/registers-of-processing-activities-research/.

[23] M. M. Martínez González, M. L. Alvite Díez, P. Casanovas, N. Casellas, D. Sanz, A. Aparicio de la Fuente *et al.*, "Ontoropa deliverable 1. state of the art and ambition." 2021.

[24] P. Ryan and R. Brennan, "Support for enhanced gdpr accountability with the common semantic model for ropa (csm-ropa)," *SN Computer Science*, vol. 3, no. 3, p. 224, 2022.

[25] A. I. Antón, W. M. McCracken, and C. Potts, "Goal decomposition and scenario analysis in business process reengineering," in *Advanced Information Systems Engineering*. Springer, 1994, pp. 94–104.

[26] M. Lubars, C. Potts, and C. Richter, "Developing initial ooa models," in *Proceedings of 1993 15th International Conference on Software Engineering*. IEEE, 1993, pp. 255–264.

[27] P. Voigt and A. Von dem Bussche, "The eu general data protection regulation (gdpr)," *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, vol. 10, no. 3152676, pp. 10–5555, 2017.

[28] M. Lankhorst *et al.*, *Enterprise architecture at work*. Springer, 2009, vol. 352.

[29] G. B. Herwanto, G. Quirchmayr, and A. M. Tjoa, "A named entity recognition based approach for privacy requirements engineering," in *2021 IEEE 29th RE Workshops (REW)*. IEEE, 2021, pp. 406–411.

[30] A. Sleimi, N. Sannier, M. Sabetzadeh, L. Briand, and J. Dann, "Automated extraction of semantic legal metadata using natural language processing," *2018 IEEE 26th RE*, pp. 124–135, 2018.

[31] A. Sleimi, M. Ceci, M. Sabetzadeh, L. C. Briand, and J. Dann, "Automated recommendation of templates for legal requirements," in *2020 IEEE 28th RE*, 2020, pp. 158–168.

[32] O. Amaral CEJAS, S. Abualhaija, D. Torre, M. Sabetzadeh, and L. Briand, "Ai-enabled automation for completeness checking of privacy policies," *IEEE Transactions on Software Engineering*, pp. 1–1, 2021.

[33] O. Amaral, S. Abualhaija, M. Sabetzadeh, and L. Briand, "A model-based conceptualization of requirements for compliance checking of data processing against gdpr," in *2021 IEEE 29th RE Workshops (REW)*, 2021, pp. 16–20.

[34] S. Ezzini, S. Abualhaija, C. Arora, M. Sabetzadeh, and L. Briand, "Maana: An automated tool for domain-specific handling of ambiguity," in *2021 IEEE/ACM 43rd ICSE-Companion*, 2021, pp. 188–189.

[35] G. B. Herwanto, G. Quirchmayr, and A. M. Tjoa, "Privacystory: Tool support for extracting privacy requirements from user stories," in *2022 IEEE 30th RE*. IEEE, 2022, pp. 264–265.

[36] S. Ghanavati, "Legal-URN framework for legal compliance of business processes," Ph.D. dissertation, University of Ottawa, Ottawa, Canada, 2013.

[37] S. Ghanavati, A. Rifaut, E. Dubois, and D. Amyot, "Goal-oriented compliance with multiple regulations," in *2014 IEEE 22nd RE*, 2014, pp. 73–82.

[38] A. I. Antón and C. Potts, "A representational framework for scenarios of system use," *Requirements Engineering*, vol. 3, pp. 219–241, 1998.

[39] A. Sutcliffe, "Scenario-based requirements analysis," *Requirements engineering*, vol. 3, pp. 48–65, 1998.

[40] ——, "Scenario-based requirements engineering," in *Proceedings. 11th IEEE RE, 2003.* IEEE, 2003, pp. 320–329.

[41] K. Weidenhaupt, K. Pohl, M. Jarke, and P. Haumer, "Scenarios in system development: current practice," *IEEE software*, vol. 15, no. 2, pp. 34–45, 1998.

[42] M. Lubars, C. Potts, and C. Richter, "A review of the state of the practice in requirements modeling," in *IEEE RE*. IEEE, 1993, pp. 2–14.

[43] C. Potts, K. Takahashi, and A. I. Anton, "Inquiry-based requirements analysis," *IEEE software*, vol. 11, no. 2, pp. 21–32, 1994.

[44] T. Huang, V. Kaulagi, M. B. Hosseini, and T. Breaux, "Mobile application privacy risk assessments from user-authored scenarios," in *2023 IEEE 31st RE*. IEEE, 2023, pp. 17–28.

[45] E. Filatova and V. Hatzivassiloglou, "Event-based extractive summarization," 2004.

[46] W.-T. Hsu, C.-K. Lin, M.-Y. Lee, K. Min, J. Tang, and M. Sun, "A unified model for extractive and abstractive summarization using inconsistency loss," in *ACL*, 2018, pp. 132–141.

[47] H. Lin and V. Ng, "Abstractive summarization: A survey of the state of the art," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 9815–9822.

[48] A. Jadhav and V. Rajan, "Extractive summarization with swap-net: Sentences and words from alternating pointer networks," in *ACL 2018-56th*, vol. 1. Association for Computational Linguistics (ACL), 2018, pp. 142–151.

[49] R. Nallapati, F. Zhai, and B. Zhou, "Summarunner: A recurrent neural network based sequence model for extractive summarization of documents," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 31, no. 1, 2017.

[50] A. K. Yadav, Ranvijay, R. S. Yadav, and A. K. Maurya, "Graph-based extractive text summarization based on single document," *Multimedia Tools and Applications*, vol. 83, no. 7, pp. 18 987–19 013, 2024.

[51] N. Mishra, G. Sahu, I. Calixto, A. Abu-Hanna, and I. Laradji, "Llm aided semi-supervision for efficient extractive dialog summarization," in *Findings of the Association for Computational Linguistics: EMNLP 2023*, 2023, pp. 10 002–10 009.

[52] A. Fantechi, S. Gnesi, and L. Semini, "Rule-based nlp vs chatgpt in ambiguity detection, a preliminary study," 2023.

[53] G. Lami, S. Gnesi, F. Fabbrini, M. Fusani, and G. Trentanni, "An automatic tool for the analysis of natural language requirements," *Informe técnico, CNR Information Science and Technology Institute, Pisa, Italia, Setiembre*, 2004.

[54] K. Ruan, X. Chen, and Z. Jin, "Requirements modeling aided by chatgpt: An experience in embedded systems," in *2023 IEEE 31st RE (REW)*. IEEE, 2023, pp. 170–177.

[55] B. Görer and F. B. Aydemir, "Generating requirements elicitation interview scripts with large language models," in *2023 IEEE 31st RE (REW)*. IEEE, 2023, pp. 44–51.

[56] M. Jackson, *Software Requirements & Specifications: a lexicon of practice, principles and prejudices*. ACM Press/Addison-Wesley Publishing Co., 1995.

[57] J. Cohen, "A coefficient of agreement for nominal scales," *Educational and psychological measurement*, vol. 20, no. 1, pp. 37–46, 1960.

[58] J. R. Landis and G. G. Koch, "The measurement of observer agreement for categorical data," *biometrics*, pp. 159–174, 1977.

[59] B. Chen, Z. Zhang, N. Langrené, and S. Zhu, "Unleashing the potential of prompt engineering in large language models: a comprehensive review," *arXiv preprint arXiv:2310.14735*, 2023.

[60] F. Yu, L. Quartey, and F. Schilder, "Exploring the effectiveness of prompt engineering for legal reasoning tasks," in *Findings of the Association for Computational Linguistics: ACL 2023*, 2023, pp. 13 582–13 596.

[61] C. Olea, H. Tucker, J. Phelan, C. Pattison, S. Zhang, M. Lieb, D. Schmidt, and J. White, "Evaluating persona prompting for question answering tasks."

[62] J. White, Q. Fu, S. Hays, M. Sandborn, C. Olea, H. Gilbert, A. Elnashar, J. Spencer-Smith, and D. C. Schmidt, "A prompt pattern catalog to enhance prompt engineering with chatgpt," *arXiv preprint arXiv:2302.11382*, 2023.

[63] M. Akter, N. Bansal, and S. K. Karmaker, "Revisiting automatic evaluation of extractive summarization task: Can we do better than rouge?" in *ACL 2022*, 2022, pp. 1547–1560.

[64] H. Nanba and M. Okumura, "An automatic method for summary evaluation using multiple evaluation results by a manual method," in *COLING/ACL*, 2006, pp. 603–610.

[65] S. Banerjee and A. Lavie, "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*, 2005, pp. 65–72.

[66] A. Soleimani, C. Monz, and M. Worring, "Nonfacts: Nonfactual summary generation for factuality evaluation in document summarization," in *ACL 2023*, 2023, pp. 6405–6419.

[67] G. R. Gibbs, *Analyzing qualitative data*. Sage, 2018, vol. 6.