Leveraging LLMs for Knowledge Engineering from Technical Manuals: A Case Study in the Medical Prosthesis Manufacturing Domain

Francesca Incitti, Andrea Salfinger, Lauro Snidaro
Department of Mathematics, Computer Science and Physics
University of Udine
Udine, Italy
{francesca.incitti | andrea.salfinger | lauro.snidaro}@uniud.it

Sri Challapalli
Lima Corporate S.p.A.
Via Nazionale, 52,
33038 Villanova UD, Italy
sri.challapalli@limacorporate.com

***Abstract*—Ontologies are nowadays widely used to organize information across specific domains, being effective due to their hierarchical structure and the ability to explicitly represent relationships between concepts. *Knowledge engineering*, like compiling companies' vast bodies of knowledge into these structures, however, still represents a time-consuming, largely manually performed process, esp. with significant amounts of knowledge often only recorded within unstructured text documents. Since the recently introduced Large Language Models (LLMs) excel on text summarization, this raises the question whether these could be exploited within dedicated knowledge fusion architectures to assist human knowledge engineers by automatically suggesting relevant classes, instances and relations extracted from textual corpora. We therefore propose a novel approach that leverages the taxonomic structure of a partially defined ontology to prompt LLMs for hierarchical knowledge organization. Unlike conventional methods that rely solely on static ontologies, our methodology dynamically generates prompts based on the ontology's existing class taxonomy, prompting the LLM to generate responses that extract supplementary information from unstructured documents. It thus introduces the concept of using ontologies as scaffolds for guiding LLMs, in order to realize a mutual interplay between structured ontological knowledge and the soft fusion capabilities of LLMs. We evaluate our proposed algorithm on a real-world case study, performing a knowledge fusion task on heterogeneous technical documentation from a medical prosthesis manufacturer.**

***Keywords*—Large Language Models, Knowledge Engineering, Ontology Population, Soft Fusion, Natural Language Processing**

I. Introduction

Motivation. In today's data-rich, quickly evolving corporate world it is becoming increasingly important to organize information efficiently and in a comprehensive and interoperable way. One of the major subjects that has been tackling this issue is *Knowledge Engineering*: Throughout the past years different technologies have been employed to understand how to better represent information, especially when this is coming from either specific or heterogeneous domains [1], [2]. On this matter, ontologies represent a mature solution to organize information across domains and potential "departmental silos", to create a unified, consistent formalization of company's entire *body of knowledge.* Their strength lies in their representation, structuring the data by modeling entities and linking various pieces of information; they also allow to create hierarchies of concepts and to perform inferences on the data depending on the rules and axioms given in the modeling step. However, the actual *Knowledge Engineering* (KE), like compiling a company's vast bodies of knowledge to an ontology, often still represents a *time-consuming* and *labor-intensive* process. Since significant amounts of a company's knowledge are often not yet available in structured formats like databases, but scattered across unstructured text documents (e.g., technical documentation or manuals), typically human knowledge engineers need to perform the *knowledge fusion* process of collating knowledge across vast corpora of unstructured text documents to manually identify and specify the required classes, instances and their relationships. Thus, the recently introduced *Large Language Models*' (LLMs') advanced text summarization and text computation capabilities [3] suggest novel opportunities for supporting this knowledge fusion process, inducing the following question that we study in this work: Could LLMs be leveraged within dedicated knowledge extraction architectures to support human knowledge engineers by automatically proposing relevant classes, instances and relationships extracted from these textual corpora?

Challenges. While the potential of LLMs for KE has already been recognized [4], [5], the focus so far has been on regarding an LLM as a *Knowledge Base* (KB) in itself for extracting its *common-sense knowledge* acquired during training, as in [6], not on (i) how to leverage LLMs for extracting structured *domain-specific knowledge* from text documents included in the prompt. Such company-specific text documents might utilize custom verbiage, abbreviations and terminology, leaving it unclear (ii) whether existing general-purpose LLMs could successfully generalize to this highly domain-specific language. Moreover, LLMs' well-known deficiencies like (iii) their occasional generation of factually incorrect information, known as *hallucinations* [5], and (iv) the observed variance of produced outputs with respect to slight variations of the input prompt [4], [6], render it an open question whether and how LLMs could be adopted for such structured *knowledge extraction* tasks in highly domain-specific settings.

Objectives. We thus empirically examine these questions on a real-world knowledge fusion case study from a medical prosthesis manufacturing company: We contribute (i) a dedicated *knowledge extraction architecture* that leverages LLMs as

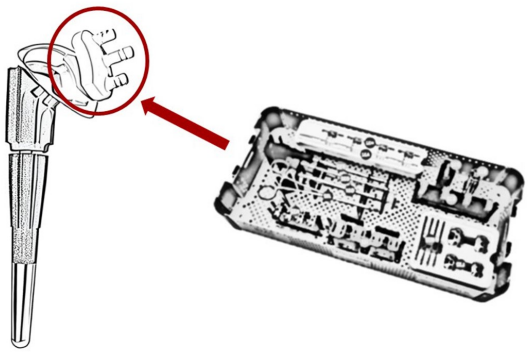

Figure 1: Example of a specific shoulder prosthesis, which consists of multiple individual parts each associated with a dedicated *instrument tray* for inserting this part during the orthopedic surgery.

generic *Information Extraction* (IE) components, and present (ii) *empirical insights* on the LLMs' consistency and performance on such structured knowledge extraction tasks on real-world company documentation. We focus on an orthopaedic prosthesis setting, for which a basic ontology structure has been created, which we try to expand and populate by using Natural Language Processing (NLP) technologies. Specifically, we use GPT-based models to retrieve information from unstructured text and try to extract structured information useful for the ontology population step.

Structure of the Paper. Section II provides the context for this research and this specific case study. Section III presents some studies that have been taken into account while researching the general tackled topics. Our proposed pipeline is explained in Section IV, followed by the discussion of the experiments in Section V and our drawn conclusions in Section VI.

II. Background

Our case study originates from a project developing an ontology to create the unified data structure across different branches of a medical prosthesis manufacturing company, to fuse the knowledge scattered across different departments. Such prosthetic gear businesses provide hospitals all over the world with their products, mainly devices designed to be used in surgeries to restore motion capabilities in patients that are in need, such as knee or shoulder prostheses. These companies not only provide hospitals the actual prosthesis, but also the instruments needed for the surgeries, including general tools like saws, screwdrivers, guides for drilling or retractors, but also trial components. Thus, capturing and modeling the knowledge of such a prosthetic gear portfolio represents a complex endeavor: First of all, a prosthesis is configured of multiple individual parts (with different parts being usable in different prostheses and prosthesis configurations), see Fig. 1. When implanting the prosthesis during the surgery, these individual parts need to be assembled in a certain manner, with each part requiring specific sets of instruments and surgical steps to perform. Each individual prosthesis part thus has a dedicated *instrument tray* associated, comprising the array of instruments needed for its insertion during the surgery. Thus, as typical for such manufacturing domains offering a broad range of products in highly configurable variations and product lines, while it is relatively straight-forward to define the *upper-level* ontology classes, which already have been modeled (i.e., the general domain concepts), modeling the entire range of knowledge on the product palette in their various configuration options represents a considerable effort due to the sheer amount of sub-classes and specific relations to capture. Moreover, knowledge like relations between different implant parts and their associated instruments are typically available in unstructured format, esp. in the form of technical documentation. For example, These companies also usually supply hospitals detailed documentation explaining how the surgery needs to be performed: these documents represent an official form of information for understanding how and for which prosthesis parts instruments need to be used.

To speed up the ontology creation, we would thus require a means to extract suggestions on the relevant instrument classes required for the different prosthesis parts from such textual documents, and present them to the human knowledge engineer for verification, framing the case study problem investigated in the following.

III. Related Work

(Semi-)Automating the labor-intensive KE process by automatically mining knowledge from text corpora with Natural Language Processing (NLP), or more specifically, dedicated *Information Extraction* (IE) approaches, as required for our problem setting illustrated in section II, represents a long-standing problem and active avenue in KE research. However, we note that established Information Extraction (IE) techniques are of limited applicability for our use case: On the one hand, established rule-based IE systems, with hand-crafted IE rules, will be too brittle for capturing for example our complex *prosthesis part – instrument* relations, requiring advanced text understanding and summarization capabilities. On the other hand, creating dedicated human-annotated training datasets for training or fine-tuning a custom domain-specific machine learning-based IE model would be infeasible given the limited amount of documentation available and their relatively infrequent updates, which would equate or even surpass the efforts for creating a labeled training dataset with the efforts of directly modeling them in the ontology. Thus, gains in such *low-resource settings* could only be expected if existing pre-trained language models (LMs) could be applied to this task "out-of-the-box", without any prior fine-tuning or other customization. This seems to have come in reach with the significantly advanced text understanding and "in-situ generalization" capabilities of the recently introduced Large Language Models (LLMs) [7], [8], inducing the question of whether and how these could be adopted as *generic IE components* when provided with appropriate textual instructions on the types of information to extract, i.e., so-called *prompts* [9]. The huge potential impact of LLMs for advancing KE already has been broadly recognized in the KE field [4], [5], leading even to the recommendation of critically re-examining established IE pipelines for KE [4]. Most related to our current problem, a concrete algorithm for *automated ontology construction* with one of the currently most prominent LLMs, OpenAI's

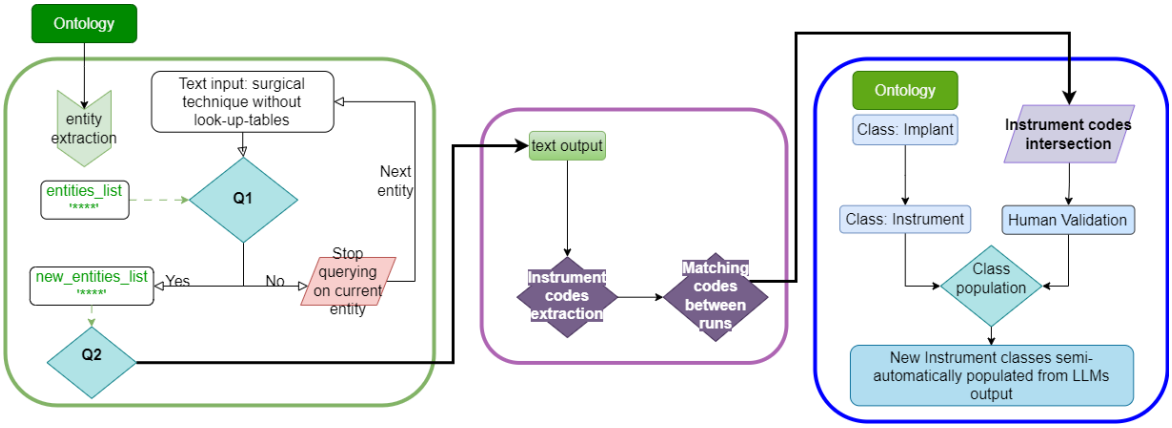

Figure 2: Processing pipeline to query LLMs for information extraction.

GPT [10], has already been proposed in [6], which we thus will base upon for developing our ontology construction approach. However, in-line with the current predominant focus on exploiting LLMs for KE by regarding an LLM as a KB in itself [11]–[14], the approach proposed by Funk et al. only extracts *common-sense knowledge* from the LLM acquired during its training. Conversely, for our given problem setting we investigate whether this approach can be extended to extracting knowledge from *domain-specific input texts* in terms of company-specific documentation included in the prompt. In the following section, we thus will carve out the specific requirements arising from this problem setting, and thereupon introduce our algorithmic solution.

IV. Approach

As clarified in the previous sections, our required knowledge extraction approach should exploit the already existing upper-level ontology and expand it by prompting LLMs to extract the taxonomy of its sub-classes and their relations with existing ontology classes from the available textual documentation.

However, employing LLMs as systematic *IE components* within such an automated ontology population framework entails the following requirements, which clearly differ from their currently predominant deployment paradigm within chatbots generating free-form natural language responses with constant human user interaction and immediate supervision:

- LLMs should generate *accurate* results, i.e., only return (the correct) information as provided in the input documentation. While this requirement appears obvious, given the well-known issue that LLMs occasionally tend to generate factually wrong responses, known as *hallucinations* [5], we thus need to examine whether this might pose a problem for the envisioned type of IE application. Hence, we also do not aim for a fully-automated ontology population pipeline, but include a human verification step, i.e., in terms of the established JDL Data Fusion Model, implement JDL Level 5 – *user refinement* [15] to incorporate the human knowledge engineer into the knowledge fusion process. Nonetheless, the objective is to ease the human knowledge engineer's work load – thus, we require that the extracted knowledge should be as accurate as possible to achieve this goal and not introduce much revision effort.
- For leveraging LLMs in such structured knowledge extraction tasks, these need to be capable of producing *consistent* results, i.e., the same prompts should lead to the same responses. Given that LLMs' (output token sampling) behavior is inherently non-deterministic (even with sampling temperature = 0)[1], we thus esp. need to assess whether the extracted knowledge is *consistent* across multiple responses to the same prompt, to reliably serve our knowledge extraction deployment scenario.
- Naturally, adopting LLMs with specifically devised prompts as "generic IE" modules requires that the LLM consistently produces well-formed structured output formats usable for the computational down-stream components, like the ontology population module.

Thus, our algorithmic architecture and evaluation case study need to specifically focus on these requirements, to provide empirical insights on LLMs' potential for adoption in such structured knowledge extraction settings.

Algorithmic Outline. Fig. 2 and Algorithm 1 outline our devised *knowledge extraction* algorithm, which leverages GPT within its individual knowledge extraction steps. Our approach builds upon the algorithm proposed in [6], but expands it to our different knowledge extraction problem: Whereas Funk et al. regarded the LLM as KB in itself, we also supply the input document, the technical manual M, in the prompt (automatically parametrized with the current class P_i of interest), to rather utilize the LLM as "generic IE" component prompted to extract information on our classes of interest from this supplied document. Given our already specified taxonomic class hierarchy under seed class P_0, we aim to extract its related classes under seed class I_0, and consequently expand I_0 accordingly by adding the identified sub-classes I_j and the mentioned relations between the already defined sub-classes P_i under P_0 and the newly added sub-classes I_j under I_0. Given our problem setting from section II, P_0 thus could refer to the taxonomic section of the already

[1]As specified in OpenAI's current API reference (v. 15/03/2024), sampling is currently non-deterministic: "Parameter `seed`: This feature is in Beta. If specified, our system will make a best effort to sample deterministically, such that repeated requests with the same seed and parameters should return the same result. *Determinism is not guaranteed*, [...]" https://platform.openai.com/docs/api-reference/chat/create#chat-create-seed

Algorithm 1: Our proposed semi-automated knowledge extraction algorithm.

Input: Seed classes: P_0, I_0, Technical Manual M; // e.g., P_0 = Shoulder Prosthesis, I_0 = Instrument

1 $P \leftarrow$ children of P_0, i.e., $\{P_i | P_i \sqsubseteq P_0\}$; // query all children of the P_0 seed class (transitively)

2 **foreach** $P_i \in P$ **do**

3 $a_1 \leftarrow$ prompt LLM whether P_i is mentioned in document M; // Q_1 - *Existence Question*

4 **if** a_1 *is* yes **then**

5 $I \leftarrow$ prompt LLM to list and describe the set of I_j mentioned for P_i; // Q_2 - *IE Question*

6 **foreach** $I_i \in I$ **do**

7 $h_1 \leftarrow$ ask human modeler to verify that I_j is related to P_i; // Human Verification

8 **if** h_1 *is* yes; // Ontology Extension

9 **then**

10 insert I_j in ontology as subclass of I_0

11 add relation $r(I_j, P_i)$ to ontology

12 **else**

13 **else**

specified *Shoulder Prosthesis* parts, as illustrated in Fig. 3, whereas I_0 could denote the generic *Instrument* class which we will need to expand with the concrete instruments required for assembling these shoulder prosthesis parts, as explained in full detail in the technical manuals on how to perform shoulder joint replacement surgeries. Thus, lines 1-2 and 8-12 in Algorithm 1 conform to conventional, automated, computational ontology manipulation steps. However, we also observe a few processing modules which appear "uncommon" for such automated ontology construction workflows: In lines 3 and 5, LLMs are employed as automatically prompted modules for extracting and summarizing the information of interest in structured format from the unstructured input document M, which is accepted or refuted in line 7 by including a dedicated *human verification step*, to ensure the accuracy of extracted information before the automated ontology population proceeds in lines 8-12. We next will explain the rationale behind these steps in greater detail, guided by our case study on the instruments extraction problem for shoulder prosthesis parts.

For this approach we exploited a partially defined ontology that already describes the domain of the prosthetic business taken into consideration. The ontology structure has been modeled with Protégé [16] from scratch to properly represent this specific orthopaedic domain: Certain aspects of this ontology have been defined through expert consultation and the use of available data in a conventional way. Some portions, on the other hand, are more difficult to capture due to the explicit lack of structured information, for which we examine our proposed semi-automatic population and ontology expansion method. The latest version of the upper-ontology is made of 59 classes and 26 relations. For our case study we have focused on the two classes and relation depicted in Fig. 3, the prosthetic *instrument* section. Thus, the upper-level class "Shoulder prosthesis" already exists, together with its sub-classes connected with a *is_a_type_of* relation, see Fig. 3. This portion of the ontology would generate, for instance, the axioms *Stem is a type of Shoulder Prosthesis*, i.e., in the formal notation of Description Logic (DL): Stem $\sqsubseteq$ Shoulder Prosthesis.

Starting from this already modeled upper-level section of the ontology, we want to expand it and populate it. Specifically, we already created an *Instrument* class, however, it is still missing its concrete subclasses. Our goal is thus to (semi)automatically

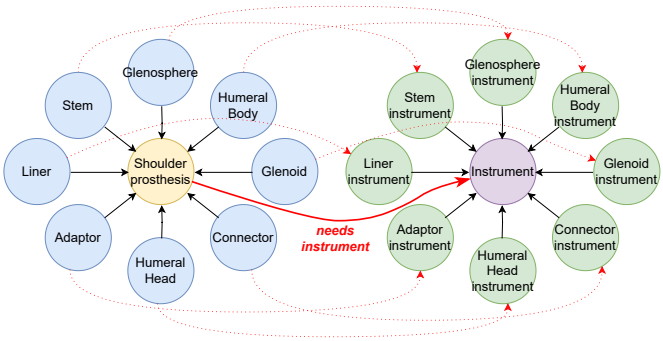

Figure 3: Upper-level class of the Shoulder Prosthesis and its subclasses linked by the relation *"is_a_type_of"* (left). The upper classes Shoulder Prosthesis and Instruments both exist in the ontology, but while the blue subclasses are already specified, we aim to extract and populate the – yet unspecified – subclasses of Instrument (depicted in green) and their relations (red arrows) with our algorithm.

populate the new Instrument subclasses as shown in Figure 3 based on extracting the relevant knowledge from the available technical manuals in PDF format.

Why 2 Prompts? One of the most imminent questions arising from Algorithm 1 may be why our algorithm prompts the LLM in a two-step approach. Our two developed prompts aim to (i) understand which type of prosthesis is present in the document served as input, by querying a yes/no question in line 3 asking whether document M is talking about concept P_i (denoted as *Q1 - Existence Question* in Algorithm 1). Only based on a positive response it proceeds with (ii) the actual IE query in line 5 (denoted as *Q2 - IE Question* in Algorithm 1), rather than directly prompting the LLM with Q2 to extract all classes I_j related to P_i. In the following, we will explicate the rationale behind this design decision and our prompt engineering.

A. Prompt Engineering on Q1 – Existence Question

The first task while processing the input text has been to clearly filter types of prosthesis that are correctly present in the document and which ones are instead missing. The ontology section below the *Shoulder Prosthesis* class lists the full shoulder catalogue, so it is imperative to only select the correct type of orthopaedic gear that is indeed present in the analyzed text. We introduced this step based on our initial experiments, which backed the existing findings reported in literature that LLMs generally do not clearly state if they are unable to answer a specific question due to a lack of information. They rather tend to generate some plausibly sounding answer if the requested information is actually not mentioned in the text provided for summarization. Analogously to the findings and algorithmic solution presented in [6], we thus aimed to prevent such undesired responses by first asking a simple yes/no question whether the supplied document actually contains information on the class of interest (Q1). Only if this has been confirmed in Q1, we proceed with the actual knowledge extraction on the class of interest in Q2. This prevents receiving an answer set for a directly asked Q2 for a prosthesis part that is actually not even mentioned in the document, for which it is unclear how and where the LLM has derived this answer set from. Note that while we were always instructing the model to only consider the text supplied in the input for answering our questions, we frequently observed that it tried answering questions based on its training knowledge if the information in the text was insufficient to answer the question, rather than stating that the relevant information was not present in the input text. We decided to prompt the LLM instead of applying string matching heuristics to identify the mentioning of our target concepts in our input document M, in order to leverage the LLM's richer semantic text understanding capabilities and to systematically analyze the consistency of these responses. The chosen first question is **Q1 = "Does this document talk about **** for shoulder surgery? Answer with Yes or No."**, where instead of the **** we iteratively provided the models the list of subclasses P_i of the Shoulder Prosthesis class. After querying the model with Q1, we saved the answers and only stored the subclasses for which the model output a positive answer: these subclasses are the only ones that are going to be queried in the second question.

B. Prompt Engineering on Q2 – Information Extraction

For the second question (Q2), the prompt engineering phase has been more complex: while in Q1 we only requested a binary response (yes/no), in this case it is necessary for the model to generate answers from scratch. To define Q2, multiple questions have been attempted, some of them are listed below:

- Does this document talk about instruments related to **** implant? Keep in mind that trials are considered instruments. Answer with yes or no.
- Which type of instruments are related to the **** implant? Return their code and description in a table format
- List the instruments found in the text (description and code) referring to **** implant.
- List the instruments referring to **** implant. Return only a list of description and codes.

Our final solution is the following: **Q2 = "List the instruments referring to **** implant. Return a csv format with names and codes of instruments on two columns and divide each row with a ;."** Something also to be noted is that we took extra care into assuring that the model did not provide additional information, meaning that the only answer that we accepted is a *Yes* or *No* for the fist prompt, and a list of instruments for the second prompt. Even when paying close attention to these precautions, the models often still provided additional information for Q2. Once outputs from the models have been retrieved, we proceed into filtering and processing the different answers. Our approach proposes now to confront the results with a human validation, and only then new instrument classes are created corresponding to the existing prosthesis classes in the ontology: The population step can now take place with the instruments retrieved from LLMs.

V. Experiments and Discussion

In the following, we provide some quantitative analysis of the results obtained in our case study. For this first case study, we investigated this approach on one document M, the most comprehensive one that explains how most of the shoulder prostheses are used.

In our experiments, we prompted two different GPT-based models:

- ChatPDF's API[2]. ChatPDF represents a web service and API for answering questions on any PDF document served as input. ChatPDF invokes GPT-3.5, with parameters set for IE.
- GPT 4-1106-preview [10]. This is one of the latest available versions of OpenAI and improves the latest 3.5 version. Other versions of GPT were meant to be tested, but unfortunately these have shown limitations regarding the input document lengths: in particular, these models do not allow prompt lengths longer than 16K, 8K or 4K tokens, depending on the version of the model. This has been a setback for the experiments, as the surgical instruction document chosen is quite long, significantly exceeding the previously listed limitations. Thus, to more broadly examine the general performance trends of GPT-based models on this knowledge extraction task, we also chose to include the results obtained with ChatPDF in our comparisons. ChatPDF analyzes the input text not in its entirety, but instead performs an automatic pre-filtering that selects only the paragraphs that it considers most relevant to the given question. Thus, only these filtered paragraphs will be passed on in its calls to GPT-3.5. While this allows us to include another GPT-based model in our analysis to examine the performance of GPT-based models, we thus need to stress that ChatPDF's results are not fully comparable to our focused analysis with GPT-4, for the following reasons: First of all, we need to be aware that we cannot fully evaluate the recall obtained with ChatPDF, since for each question ChatPDF opaquely filters the parts of the content to be passed on to GPT-3.5, which thus can only generate its answers based on these chunks of text – leading to a potential loss of information since instruments and prosthesis are cited in the text multiple times in different parts of the document, which

[2]https://www.chatpdf.com/docs/api/backend

might not have been selected in ChatPDF's pre-filtering step. Something else to be noted is the control of the parameters of the requests sent to GPT, esp. with respect to the temperature setting for the selected models. While we could explicitly set the output token sampling *temperature* parameter[3] for GPT-4, ChatPDF does not allow to explicitly set these parameters. We set GPT-4's temperature to 0.2 in all tests, forcing the model to consider only the more probable output tokens, while ChatPDF remained with its original parameters (not disclosed). We also experimented with setting the temperature to 0 for our GPT-4-based tests to only consider the most probable output tokens, but interestingly, often received completely wrong responses in this cases, representing an issue that we plan to further investigate in future work by explicitly analyzing the log probabilities of the output tokens.

Document Preprocessing. The analyzed document is 84 pages long and structured as follows: Firstly, there is a small description of the full prosthesis configuration (5 pages), then most of the technical manual consists of textual instructions on how the surgery needs to take place with some supporting images (54 pages). The final pages list the section of the product catalogue containing the instruments and prosthesis referred to by the instructions in the document (25 pages). Companies usually assign a alphanumeric code to these products to allow for their unique identification in the text and facilitate their referencing as they frequently have lengthy names. This means that the actual names of the instruments and prostheses are hardly ever mentioned in the usage instructions; instead, they are referred to by their codes. According to the structure of the document, we have decided to make the models analyze not the full manual, but only the first descriptive pages and the operative instructions – the first 59 pages – removing the last section that deals with the product catalogue. This choice has been taken based on observations made during the first testing stage: The models had a tendency to make up responses pertaining to the instruments and the prosthesis themselves, picking up randomly the product codes at the end of the document. As an example, consider the following scenario from the first test phase: When the model was asked about instruments to be used for a type of prosthesis that was not actually contained in the document it still provided a list of instruments. This list however contained wrong information: Instruments not present in the document were listed and wrongly associated the codes of instruments that instead were present in the document. This happened only when the entire document was passed to the models without removing its final product catalogue part. Upon cross-checking, we observed this mismatch between the instrument's names and the extracted code. Thus, we refrained from giving the models access to all data to stop them from coming up with their own solutions, and consequentially creating false answers.

Consistency of Extracted Information. While experimenting with the models, we realized the need to specifically investigate their consistency: Something noticeable with these kinds of generative models is that whenever asked the same question, with the same input data, it is very unlikely that the models answer twice in the same way due to their non-deterministic sampling of the output tokens, representing one of the specific issues that we will dissect next by retrieving and comparing $n = 5$ responses for each prompt from each model. We have chosen this limited number of sampled responses for this first feasibility study to fine-tune our experimental setup, while also considering the cost constraints from using OpenAI's API. For future work, based on our findings derived from these initial results, we aim to expand our experimental setup by increasing the number of sampled responses and types of extracted information.

[3]https://platform.openai.com/docs/api-reference/chat/create#chat-create-temperature, last accessed on 15-03-2024.

A. Analysis Q1 – Existence Question

For a total of 28 subclasses of the Shoulder Prosthesis class, we noticed consistent answers for both models, as shown in Table I. To check for the correctness of the results from Q1, we previously detected how many prosthesis types are present in the document analyzed by the models out of the complete list of children of the Shoulder Prosthesis class. ChatPDF detected 25 correct outputs out of 28. The three remaining have been falsely found by the model, and these false positives will go through the Q2 step even though they are not present in the input document. The same experiment has been repeated for the second model (GPT-4). Again in Table I we notice less consistency than before, in fact the first response (1) is slightly different that the following ones (2-5). In the first response we get 24 correct results out of 28, meaning that the model made one additional mistake compared to the previous model. Nonetheless, the next responses (2-5) perform overall better, result in in 26 correct answers out of 28. Overall, we can state that for the first step (Q1) results are impressive: 89% accuracy for ChatPDF and 90% accuracy on average for GPT 4.11-06.

Table I: Experiment results for Q1

	ChatPDF (GPT 3.5)	GPT 4.11-06	
Response	1-5	1	2-5
True Positives (TP)	15	14	15
False Negatives (FN)	-	1	-
False Positives (FP)	3	3	2
True Negatives (TN)	10	10	11

B. Analysis Q2 – Information Extraction

As one can imagine, this second step has been more difficult to evaluate, since the answers we received from both models have not been consistent, differently than what happened with Q1. As mentioned in the previous sections, one issue for this step has been preventing the model to add information from its own training knowledge and just remain coherent to the content of the document served as input. Once we settled with a satisfactory Q2, we again queried 5 responses from ChatPDF and GPT-4, respectively, for each prosthesis type. As explained earlier, Q2 has been asked to the model only upon entities that received a positive answer in the previous Q1 step. This means that we have queried the models on different prostheses, dependent on the outputs from the previous step. As we can see from Table I, we queried:

- for ChatPDF: 15 true positives and 3 false positives. This means that we still queried the model upon 3 wrongly-output prosthesis.

	ChatPDF (GPT-3.5) Results						GPT-4 Results					
Prosthesis Part	Corr.	Extr.	GT	Prec.	Rec.	Cons.	Corr.	Extr.	GT	Prec.	Rec.	Cons.
3 ** Glenoid	4	4	17	1	0.24	0.24	9	9	17	1	0.53	0.82
Anatomic Adaptor	4	4	33	1	0.12	0.21	6	8	33	0.75	0.18	0.73
A** Humeral Body	5	5	34	1	0.15	0.29	22	31	34	0.71	0.65	0.61
Cemented Glenoid	1	1	22	1	0.05	0.25	8	8	22	1	0.36	0.62
Glenosphere	5	5	36	1	0.14	0.38	6	10	36	0.60	0.17	0.37
Glenosphere Connector	1	1	36	1	0.03	0.06	5	6	36	0.83	0.14	0.67
Humeral Head	2	2	35	1	0.06	0.18	5	10	35	0.50	0.14	0.15
M** Glenoid	13	13	38	1	0.34	0.93	14	14	38	1	0.37	0.56
M** Liner	0	0	23	0	0	0	1	1	23	1	0.04	0.06
R** Humeral Body	1	1	35	1	0.03	0.05	12	13	35	0.92	0.34	0.30
Stem	5	7	51	0.71	0.10	0.33	8	17	51	0.47	0.16	0.31
Connector (x)	–	–	–	–	–	–	3	3	36	1	0.08	0.33
Liner (x)	–	–	–	–	–	–	0	1	34	0	0	0.33
Peg (x)	–	–	–	–	–	–	0	6	36	0	0	0.46
Screw (x)	–	–	–	–	–	–	2	2	38	1	0.05	0.17
Summary	Σ : **35**	Σ :**43**	Σ :**360**	**0.88**	**0.11**	**0.26**	Σ : **101**	Σ :**139**	Σ :**360**	**0.72**	**0.2**	**0.43**

Table II: Outputs evaluation, based on comparing the intersecting sets of extracted instruments. References: "*Corr.*" stands for the correct number of instrument extracted for each prosthesis, "*Extr.*" is the total number of the instruments detected by the model for each prosthesis, "*GT.*" is the ground truth supplied by Lima, "*Prec.*" is the precision value for the outputs, "*Rec.*" stands for the recall value and finally "*Cons.*" represent the consistency of the outputs of the models for each prosthesis. Tha last row "Summary" shows a recap of the overall obtained results: the cells with the Σ symbol refer to the sum (total) of all the detected instruments, while the other columns are summarized with the average value (mean).

• for GPT-4 we selected the outputs from responses (2-5). We choose this because the first response's results were not later repeated, so we kept querying on the outputs that the model was more consistent on. As before, we queried the Q2 upon 15 true positives, but this time we have only 2 false positives. Even if just minor, GPT-4 shows some improvements compared to the previous model, being able to detect correctly one more type of prosthesis.

C. Codes extraction

We decided to exploit the codes of the instruments as their unique identifiers, to assess how many of them consistently appeared in each response. We quantify the consistency in models' returned instrument code sets by computing its overlap: i.e., we compute the size of the intersection of all five returned instrument code sets (representing the consistently returned results), and divide this by the size of the union of all returned instrument codes across all responses, as shown in the *Consistency* columns in Tab. II (0 = not a single code overlapping across all responses, thus zero consistency, 1 = all responses return the same set of codes, i.e., full consistency). Taking advantage of the instrument codes allowed us to avoid having to resort to regular expression-based string matching techniques: While the names of the instruments may appear differently, their codes are unique and remain the same for the entire document. For this reason, we again extracted the $n = 5$ sets of instrument codes R_1 - R_5 for each prosthesis type, and computed their set intersection, $R_{\text{intersect}} = R_1 \cap R_2 \cap R_3 \cap R_4 \cap R_5$. We regard this intersection of the different sampled responses' answers as the most reliable extraction of instruments required to operate with the specific type of prosthesis.

To check the accuracy of the results we consulted a domain expert to provide us with a ground truth mapping. Based on the document that we have used for the experiments, the expert created a mapping between the prosthesis and their related instruments. In this case the expert has drawn on all the information reported in the technical manual, but also on his expert knowledge as a person who works daily with these prostheses and these instruments. We thus subsequently also filtered these lists for the codes that were indeed mentioned in the pre-processed document supplied in the prompts. The comparison with the different experiments and this expert-ground truth is shown in Tab. II, which shows for each prosthesis type: how many correct instruments have been extracted by the models ("Corr") and how many were extracted in total ("Extr"), the number of ground truth entities ("GT"), precision ("Prec") and recall ("Rec"): The precision states how accurate our predictions overall are, and the recall measures the amount of ground truth entities that have been indeed extracted by the models. To make some overall comparison we can clearly see a high precision for the ChatPDF extraction, but fewer codes have been overall detected (which might be attributed to the previously mentioned text-pre-filtering performed by ChatPDF). On the other hand, GPT-4 seems to extract an higher number of instruments, with still a significant precision. Both models fail to extract all the correct instruments, as shown by the recall. Something to be noted is that Table II presents some empty rows: This happened because during the experiments we noticed that some responses output, for specific prosthesis, a total of 0 instruments detected. This was especially true for ChatPDF, particularly for prostheses that could be recognised as instruments by non-experts: Prostheses

for which this happened are marked with "*(x)*" in Table II. An example of this occurring is the *Screw class*: It may seem evident that a screw is an instrument rather than a prosthesis. However, this contradicts the domain logic of a prosthesis manufacturer, as any element that will be permanently implanted in the patient is always considered and processed as a prosthesis, regardless of their function.

VI. Conclusions and Future Work

In this work, we focused on the problem of how LLMs could be leveraged to speed up KE by exploiting a company's existing textual documents, driven by the question whether LLMs might serve as generic IE components such in domain-specific low-resource settings to avoid the costly creation or fine-tuning of domain-specific IE models. We contributed (i) a knowledge extraction algorithm, which exploits a partially modeled ontology to dynamically prompt LLMs to extract suggestions on yet missing classes and relations from text corpora for semi-automated ontology population, and (ii) empirical insights from its application in a real-world case study from a prosthetic gear company. While this first case study naturally is of limited scope to still allow for manual validation of the retrieved results, we found that the *consistency* and *precision* achieved with these generic models on complex real-world technical documentation appears indeed promising, esp. if the obtained structured answers are averaged across several sampled responses. However, our results indicate that *recall* might be a more problematic issue. This does not appear to be exclusive to our experiments and domain, but represents a finding which we recently corroborated also in a different domain [17], warranting a more systematic investigation in future work. Even if this contribution presents a first analysis of the feasibility of exploiting LLMs for KE, it has shown both potential benefits and limitations, highlighting the need for further studies and algorithm development. To tackle the problematic recall, we aim to curate more comprehensive human-annotated ground truth datasets with our domain experts in future work, which will allow us to analyze the situations in which LLMs tend to struggle more systematically. We might also consider Retrieval-Augmented-Generation (RAG) to enable for the analysis of more comprehensive text corpora. In conclusion, our initial findings suggest that state-of-the-art LLMs might indeed offer potential as *supportive* tools in our envisioned semi-automated KE setting, following the key vision of Human-Centered AI [18], if their users are clearly aware of their limitations.

Acknowledgments

This work was funded by the European Union – NextGenerationEU, National Recovery and Resilience Plan (NRRP) M4C2 Inv. 3.3 D.M. 352/2022. The views and opinions expressed are solely those of the authors and do not necessarily reflect those of the European Union, nor can the European Union be held responsible for them. This research was funded in whole, or in part, by the Austrian Science Fund (FWF) [Grant J4678-N]. We also thank Lima Corporate S.p.A. for supporting this research. This research has been funded by the Ministero delle Imprese e del Made in Italy through a grant named Accordi per l'Innovazione, for the project titled "SHIAB". We would like to further thank Giacomo Pozza and Diego Faelli, both from Lima Corporate S.p.A., for their valuable time dedicated to this research.

References

[1] K. J. Cios and L. A. Kurgan, "Trends in data mining and knowledge discovery," in *Advanced techniques in knowledge discovery and data mining*. Springer, 2005, pp. 1–26.

[2] M. J. Vlaanderen, "Automated knowledge acquisition for expert systems: An overview," 1990.

[3] F. Incitti, F. Urli, and L. Snidaro, "Beyond word embeddings: A survey," *Information Fusion*, vol. 89, pp. 418–436, 2023.

[4] J. Z. Pan *et al.*, "Large Language Models and Knowledge Graphs: Opportunities and Challenges: 38 pages," *Transactions on Graph Data and Knowledge (TGDK)*, vol. 1, 2023.

[5] Z. Ji *et al.*, "Survey of Hallucination in Natural Language Generation," *ACM Computing Surveys*, vol. 55, no. 12, pp. 1–38, 2023.

[6] M. Funk *et al.*, "Towards Ontology Construction with Language Models," in *Joint proceedings of the 1st workshop on Knowledge Base Construction from Pre-Trained Language Models (KBC-LM) and the 2nd challenge on Language Models for Knowledge Base Construction (LM-KBC) co-located with the 22nd International Semantic Web Conference (ISWC 2023), Athens, Greece, Nov. 6, 2023*, ser. CEUR Workshop Proceedings, vol. 3577, 2023.

[7] T. Brown *et al.*, "Language Models are Few-Shot Learners," in *Advances in Neural Information Processing Systems*, H. Larochelle *et al.*, Eds., vol. 33. Curran Associates, Inc, 2020, pp. 1877–1901. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf

[8] S. Chan *et al.*, "Data Distributional Properties Drive Emergent In-Context Learning in Transformers," in *Advances in Neural Information Processing Systems*, S. Koyejo *et al.*, Eds., vol. 35. Curran Associates, Inc, 2022, pp. 18 878–18 891. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2022/file/77c6ccacfd9962e2307fc64680fc5ace-Paper-Conference.pdf

[9] S. Bach *et al.*, "Promptsource: An integrated development environment and repository for natural language prompts," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. ACL, 2022, pp. 93–104.

[10] OpenAI *et al.*, "GPT-4 Technical Report." [Online]. Available: https://arxiv.org/abs/2303.08774

[11] *Joint proceedings of the 1st workshop on Knowledge Base Construction from Pre-Trained Language Models (KBC-LM) and the 2nd challenge on Language Models for Knowledge Base Construction (LM-KBC) co-located with the 22nd Intl. Semantic Web Conference (ISWC 2023), Athens, Greece, Nov. 6, 2023*, ser. CEUR Workshop Proceedings. CEUR-WS.org, 2023.

[12] B. Veseli *et al.*, "Evaluating the knowledge base completion potential of GPT," in *Findings of the Association for Computational Linguistics: EMNLP 2023*, H. Bouamor, J. Pino, and K. Bali, Eds. Stroudsburg, PA, USA: Association for Computational Linguistics, 2023, pp. 6432–6443. [Online]. Available: https://aclanthology.org/2023.findings-emnlp.426/

[13] D. Alivanistos *et al.*, "Prompting as probing: Using language models for knowledge base construction." [Online]. Available: http://arxiv.org/pdf/2208.11057.pdf

[14] B. P. Allen, L. Stork, and P. Groth, "Knowledge engineering using large language models: 19 pages," *Transactions on Graph Data and Knowledge (TGDK)*, vol. 1, 2023.

[15] E. P. Blasch and S. Plano, "Level 5: user refinement to aid the fusion process," *Proc. SPIE*, vol. 5099, 2003. [Online]. Available: http://dx.doi.org/10.1117/12.486899

[16] M. A. Musen, "The protégé project: a look back and a look forward," *AI Matters*, vol. 1, no. 4, pp. 4–12, 2015. [Online]. Available: https://doi.org/10.1145/2757001.2757003

[17] A. Salfinger and L. Snidaro, "Probing the consistency of situational information extraction with large language models: A case study on crisis computing," in *2024 IEEE Conference on Cognitive and Computational Aspects of Situation Management (CogSIMA)*, Montreal, Canada, May 2024.

[18] B. Shneiderman, "Human-centered artificial intelligence: Reliable, safe & trustworthy," *International Journal of Human–Computer Interaction*, vol. 36, no. 6, pp. 495–504, 2020.