Integration of Semantic Wiki Technologies with Large Language Models as a Technological Foundation for Experience Acquisition from Natural Language Documents*

Igor Sinitsyn[1,†], Julia Rogushina[1,2,†], Kostiantyn Yurchenko[1,†], Yurii Bova[1,*,†]

[1] *Institute of Software Systems, National Academy of Sciences of Ukraine, Glushkov Pr. 44, 03680 Kyiv, Ukraine,*
[2] *Institute for Digitalisation of Education, National Academy of Educational Sciences of Ukraine, Berlynskoho 9, 04060 Kyiv, Ukraine*

Abstract

The modern era of digital transformation accelerates the data accumulation across all areas of human and societal activity and causes the emergence of a large number of heterogeneous electronic documents with natural language information of varying structuring degrees. One of the main challenges becomes the efficiency of intelligent analysis of large volumes of unstructured data contained in electronic documents. The purpose of such analysis is to obtain information required by users to satisfy their informational needs and to transform the acquired information into a form that ensures its convenient and effective use. But such analysis needs in tools that transform data to knowledge and allow the use of experience reflected in the processed data, complementing it with existing knowledge sources and supporting collaboration with experts. This situation motivates our research aimed at knowledge acquisition from documents where the main part of the content consists of natural language fragments and semi-formalized semantic relations between them. We integrate semantic technologies that allow explicit application of formalized domain knowledge, with elements of generative artificial intelligence as a tool for linguistic analysis, to develop models and methods that enable control over the correctness of the acquired knowledge and the rules for its interpretation for generating recommendations based on experience. The use of large language models in combination with the knowledge management allows increasing the reliability of analysis results and provides a conceptual framework for explaining the ways in which these results are obtained and improved (based on elements of the task ontology model). Such integration of two substantially different directions of artificial intelligence ensures the application of verified knowledge structures for generating complex information objects. As a result, the user receives explanations of the ways of result construction that explicitly track and indicate utilized particular elements of formalized content and logical rules. Such explanations become an important component of trust in the system (especially in critical subject domains).

Keywords

Large Language Models, Semantic Wiki, ontological analysis, natural language documents, knowledge acquisition

1. Introduction

The digital transformation of society as a whole and individual areas of its activity has led to the emergence of a large number of various electronic documents that contain natural language information about valuable human experience. These documents are focused on human perception, rather than machine processing, and contain information of different levels of structuring and formalizing complicated its analysis and acquisition of practically useful knowledge. We consider

* *Applied Information Systems and Technologies in the Digital Society (AISTDS-2025), October 01, 2025, Kyiv, Ukraine*

[2]* Corresponding author.
[†] These authors contributed equally.
ips@nas.gov.ua (I.Sinitsyn); ladamandraka2010@gmail.com (J.Rogushina); urchikak8@gmail.com (K.Yurchenko); bova1997@gmail.com (Yu.Bova)
0000-0002-4120-0784 (I.Sinitsyn); 0000-0001-7958-2557 (J.Rogushina); 0000-0003-3150-0027 (K.Yurchenko); 0009-0008-9797-5213 (Yu.Bova)

the class of tasks as *experience acquisition from documents* (EAD), where the main part of the content consists of natural language text together with relations of varying formalization between fragments of that text.

Examples of EAD tasks include:

- construction of structured specifications of selected objects (information systems, organizations, etc.) for the purposes of certification and attestation based on analysis of examples, normative documents and standards;
- building recommendations for further actions based on a generalization of experience examples described in the provided documents (reports, test results) in accordance with the rules and requirements of the subject area;
- intelligent processing of document workflows in research and educational institutions that combine domain specific knowledge with state and international standards for information representation (for example, FAIR principles for open science), thereby enabling automated generation of analytical reports and RDs in forms defined by templates and examples [1].

Within the context of semantic analysis of natural language (NL) texts, solution of these task emerges a need for some intelligent software system that transforms input documents (case descriptions, normative acts, user requests) into resulting artifacts (structured responses, formalized instructions, semantic templates) by processing of their semantics. The complexity of such EAD tasks is caused not only by the necessity to process large volumes of unstructured NL information but also by the absence of explicitly defined requirements for the target documents and of formal rules for their construction. In most cases, the required knowledge is obtained by analysis of examples of resulting documents (and the available set of such examples is typically insufficient to derive unambiguous solutions) and by the normative documentation (rules, standards, resolutions, etc.) and pertinent sources that the user provides or that are available in the open information space.

This transformation is not trivial because it requires:

- identification of relevant fragments in the input NL texts;
- acquisition of rules that define that model transformation of input information into results from examples;
- construction of semantic correspondences between fragments of example documents and rules;
- application of these rules to user data;
- assessment of the quality and relevance of the resulting document.

2. Specificity of experience acquisition from natural language documents

Why do we combine such rather different tasks into a single group, and why do we attempt to build a generalized platform for their solving? The main criterion that unites these examples of EAD is their result-oriented nature, that is, the choice of analysis methods and procedures for processing input information largely depends on the user's requirements regarding the resulting information object created by this processing. These requirements determine what specific information needs to be extracted from the available input documents.

From the perspective of information analysis, the entire set of the above-mentioned tasks of NL-document processing can be regarded as a task of transforming the knowledge. This knowledge is contained in a heterogeneous set of documents into a *complex information object* (CIO) [2], which consists of content elements of the input documents (either directly or after appropriate transformations). The structure of this CIO is defined by examples of documents provided by the

user and by the domain regulatory documents (they are also provided by the user or available on the Web). The CIO consists of a set of simpler *information objects* (IOs) that are connected by various relations. The values of individual elements of this CIO are determined on the basis of the analysis of the properties of IOs that are described in the input data, and according to the rules that can be extracted from other documents.

In many cases a secondary subtask arises: transformation of the CIO into a resulting document (RD) where the information from the CIO is represented in a certain formalized representation in accordance with the subject domain requirements (for example, text formatting according to specific rules).

A mandatory element of EAD is experience – the presence of examples of solution construction that link proposed RDs with one or several input documents (IDs). Such examples describe the precedents corresponded to this RD. NL-descriptions of the knowledge transformation rules contained in the input regulatory documents, as well as the characteristics of objects in precedent descriptions, may be fuzzy and allow for ambiguous interpretation. Thus, to solve the EAD task, we need both for linguistic analysis tools (with corresponding knowledge bases) and for semantic analysis tools, which, based on the formalized knowledge about the domain, generate the characteristics of the resulting CIOs.

The specificity of the proposed approach is the integration of methods of traductive learning from examples ("from particular to particular") with methods of deductive inference of new knowledge from existing one (using rules extracted from the domain regulatory documentation): if the generalization of the available examples allows for the generation of several different RDs, then domain knowledge, represented in various forms, can be used to resolve such ambiguity — explicitly (external knowledge bases, ontologies, thesauri) or implicitly (regulatory NL-documents, instructions, standards, etc.). Ambiguous interpretation of the provided information may be caused either by ambiguity of NL or by an insufficient set of input data. Both problems can be solved by involving external knowledge bases (KBs) that are pertinent to the domain and user task.

First step of analysis is identification the general structure of the RD (its main elements, their order and form of representation) and formal fixation of this structure (using templates, ontological models, metadata schemas, etc.). Second step is identification to the sources of information for each element of this structure and detection the rules for transforming these sources into the values of the RD structural components.

Moreover, it is advisable to select from the corpus of available documents those facts and statements that are necessary to construct the CIO of a given type; that is, it is necessary to analyse a large volume of information in order to determine the relevant and actual rules.

Let us distinguish the main types of information processed in EAD:

- $ID = \{id_i\}, i = \overline{1,p}$ is a non-empty set of ID;
- $RD = \{rd_i\}, i = \overline{1,k}$ is a set RD examples, such that $\forall rd_i \in RD, i = \overline{1,k},$ at least one precedent exists in the IDs $\exists id_j \in ID, j = \overline{1,k},$ corresponding to each RD, $F(\{id_m \in ID, m = \overline{1,k_{rd_i}}\}) = rd_i \in ID$;
- $ND = \{nd_i\}, i = \overline{0,q}$ is a collection of normative documents, instructions and external KBs bases that can contain both structured and unstructured knowledge: $ND = ND_{struct} \cup ND_{non\_struct}$;
- EAD thesaurus $SP = \{sp_i\}, i = \overline{1,w}$ is a nonempty set of semantic properties of IDs and RDs used for content structuring;

- $VSP = \{vsp_i\}, i = \overline{1, w_{sp_i}}, i = \overline{1, w}$ is a nonempty set of admissible values for semantic properties of IDs and RDs;
- an ontological model of EAD constructed on the basis of the semantic properties from SP and the value space properties from VSP.

At the upper level of abstraction, all input documents from ID are classifiable according to information type the system can obtain from them during analysis for generation of the resulting document:

- profile descriptions of domain IOs, which explicitly or implicitly define parameters of objects of various types, their relations and admissible values, and which may characterize their semantics or intended roles in the system — these are sources of property sets;
- Descriptions of specific instances of IOs of various types (this information may be provided separately or be included in the evaluated precedents) as asource of property values;
- Descriptions of classified precedents (individual domain events that associate sets of parameter values for different object sets with a specific decision, assessment, or action);
- Descriptions of domain knowledge concerning the (normative documents, resolutions, standards, instructions) that can be used as the sources of transformation rules for converting task conditions and user requirements into a result object.

The results produced by the system can also be divided into several classes depending on their impact:

- search for precedents and analogues;
- specification of the current situation or object, based on integration of facts and rules extracted from various domain documents;
- forecast regarding the assessment of a certain situation, the consequences of actions, or the functioning of a specified object based on the analysis of precedents and the application of general rules derived from regulations;
- recommendation is a set of proposals on how it is expedient to modify the properties of the situation or object in order to increase the probability of a positive forecast or decrease the probability of a negative one.

This classification requires dividing semantic properties of objects used in semantic annotation of documents into the following groups:

- *invariant* (the value is fixed, static and does not change over time), for example, the name of organization or unit, the capacity of a device, or the date of an event;
- *dynamic, independent* of user actions (values are change over time, but the user cannot directly influence these changes), for example, weather conditions or equipment parameters;
- *dynamic, dependent* (the user can explicitly change values through actions), for example, equipment operating parameters, the location of available personnel, or the sequence of available actions.

Recommendations have to include only advices concerning changes to properties of the third type, and only under the condition that such changes can be influenced by the user who receives the recommendations. Therefore, it is advisable to describe these characteristics of semantic properties in more detail, explaining who or what exactly can influence their values and in what way.

From a formal point of view, EAD requires execution of the following main stages of analysis:

- definition of the set of vsp_i for each input document from ID and resulting document from RD;
- construction of a function F for transforming an arbitrary set vsp_i for an ID into a set vsp_i for an RD;
- transformation of the set vsp_i for an RD into the RD itself in the required format.

Therefore, we can distinguish the following main stages for solving tasks of this type:

- analysis of the structure and semantics of the RDs;
- analysis of the structure and semantics of precedents corresponding to the proposed RDs;
- identification of rules and dependencies between precedents and RDs;
- extraction from normative documents of the subject domain of rules that allow resolution of ambiguities in dependencies between precedents and the RDs;
- construction of an ontological model of the subject domain, which contains the components of the CIO, the relations among them, and constraints on possible ways of combining them.

Manual execution of such analysis requires a great deal of time and the involvement of experts with appropriate knowledge in the chosen domain. Moreover, if the information environment has dynamic nature then generated results become outdated even before their preparation is complete — for example, by use of out-of-date standards or documents that have lost validity, without accounting for newly available domain rules.

These reasons define the need in creation automated tools for processing NL documents that can quickly and reliably perform analysis of their content at the knowledge level. They have to ensure representation of analysis results in a form that meets user requirements (that is, to generate complex-structured NL documents with user-defined set of elements).

The current level of development of *Generative Artificial Intelligence* (GenAI), in particular *Large Language Models* (LLMs), meets many requirements of such tasks, but it leaves open the issues of reliability of the obtained results and of the possibility of explaining the reasoning paths that led to them. When processing large volume documents, it is insufficient merely to indicate that certain sources were used; instead, it is necessary to explicitly specify which particular content elements were interpreted to create particular elements of the resulting document.

Now information environment saturated with unstructured NL texts needs to build systems capable of performing semantic analysis of such data for the purposes of structuring, interpretation, and integration into formalized knowledge models. The use of large language models (LLMs) opens new possibilities for automated processing of such texts, but it also raises a number of challenges related to ensuring the reliability, consistency, and interpretability of results. One of the key problems is the traductive nature of knowledge generation by LLMs — that is, the transition from individual examples to new individual cases without formal generalization. Such specifics create a risk of constructing semantic representations that do not correspond to user expectations or to regulatory requirements. The absence of explicit formalization of knowledge in input data complicates the verifying process and requires the creation of intermediate semantic layers that can serve as an interface between the LLMs and the evaluation system. In addition, we have to take into account that many LLMs process English texts much more effectively than Ukrainian or multilingual documents.

3. Analysis of research on the use of LLMs for document analysis

Recent advances in machine learning, especially deep learning, have expanded the capabilities of artificial intelligence (AI), enabling high quality image and speech recognition, processing of NL

documents, etc. The growth of Big Data technology and the increase in computational power provided by graphics processors have further accelerated research and practical application of AI .

LLMs are machine learning models based on neural networks and trained on large data repositories for tasks related to NL analysis. The effectiveness of LLMs depends on the criteria for training data selection. Each NL word in LLM is represented as a point in a multidimensional space corresponding to a fixed length vector, which encodes the semantic properties of the word and its relations with other NL words. Semantically close concepts are represented by nearby points with similar vectors, and this similarity underlies subsequent text analysis. LLMs are designed to process information and to make decisions or forecasts based on the data provided to them; therefore, their performance depends not only on processing algorithms but also on the volume and quality of the training data. The training corpus has to be sufficiently large to allow the model to acquire an adequate vocabulary and to understand the meanings and semantic relations among concepts represented by words and phrases. However, the use of excessively large volumes of information irrelevant to the LLM tasks may reduce model quality, especially if the texts contain unreliable, tendentious, contradictory or outdated information.

In the development of LLMs that show not only excellence in linguistic analysis but also include a spectrum of generation and optimization tasks, several generations that differ in performance and depth of analysis are distinguished. LLMs are developed in many countries and by major corporations (such as Microsoft, OpenAI, and Google), and many models are released as open source. Currently, fourth generation LLMs are characterized by:

- improved text generation capabilities for producing more precise answers to user queries;
- discovery of complex semantic relations between words and improved context comprehension;
- more advanced personalization of operational parameters.

Various metrics are used to evaluate the effectiveness and quality of language models [3], both for general text processing tasks and for specialized tasks such as software generation, chatbot construction, mathematical problem solving, logical inference, and “human in the loop” interaction. The most widely used general metrics include:

- *accuracy* defined as a proportion of correct model responses on a test dataset: Exact match, Quasi exact match, F1 score, ROUGEs score [4];
- *calibration* is a possibility of additional adjustments: Expected calibration error, Area under the curve [5];
- *fairness* is an absence of false or fabricated statements and references: Demographic parity difference, Equalized odds difference [6];
- *robustness* is a resistance to small changes in object parameters: Attack success rate, Performance drop rate [7].

In addition, important parameters are *execution time* – how quickly the model processes requests, and the *amount of computing resources* required for the operation of the LLM.

It should be noted that among the most successful LLMs, it is difficult to give preference to any one of them by all these criteria, because each new version of these systems offers certain advantages over others. At the same time, software implementations differ in hardware requirements, speed and tuning to certain NL. In addition, some LLMs are more focused on certain functions (such as translation, generation of diagrams, images, program code, etc.), and it makes the choice of LLM for a specific task a complex multi-criteria task that requires both theoretical analysis of parameters and practical verification.

Fine tuning of LLMs is a set of methods for adapting pretrained base models to specific domains, tasks or behavioral objectives. The need for fine tuning arises if a particular task requires additional

knowledge, specialized terminology or the use of NL that has not been sufficiently studied. The choice of method depends on the available data, computational budget, privacy constraints, deployment requirements (latency, memory), and alignment/safety criteria. Now a lot of different approaches to fine tuning are distinguished.

Domain Adaptive Pretraining (DAP) is performed on large corpora of unlabeled texts from a specific domain in order to reduce the gap between the general pretrained model and the NL distribution in the target domain (for example, medical articles, legal documents, scientific publications) and to improve performance. However, it may cause "catastrophic drift" of the model general properties and requires significant resources and high quality filtering of the text corpus [8];

Task specific supervised fine tuning is aimed at optimizing model weights on labeled input–output pairs, provided that high quality labels are available for the target task and that most model parameters can be updated. However, it requires substantial memory and computational resources [9];

Instruction tuning is performed on datasets of "instruction–response example" pairs to improve the model ability to follow NL instructions from users. The quality of results depends on the diversity and quality of instruction pairs [10].

Reinforcement Learning from Human Feedback (RLHF) and preference-based optimization align the model with human judgments of usefulness and safety and reduce undesirable outputs. However, they require multistage processing, including collection of human preferences comparing pairs of responses, training a model on these preferences, and optimizing the base model to maximize reward [11].

Parameter efficient fine tuning (PEFT) keeps the base model largely fixed, while adaptation is achieved by adding or updating small subsets of parameters. Examples include Prefix/Prompt tuning (optimization of contextual vectors or prefixes instead of model weights) and LoRA (low rank adaptation method that inserts trainable matrices into weight updates).

Hybrid pipelines combine supervised instruction fine tuning (SFT), RLHF for final alignment, and RAG. They are most often applied in the construction of chatbots and fact sensitive systems.

Continual learning supports model updates based on a stream of new data but requires complex mechanisms for controlling forgetting and for retaining representative examples.

The choice between finetuning of open model via API and finetuning of a local copy of the model is determined by two groups of factors: requirements for privacy, regulation, and data control; and technical organizational constraints such as computational resources, costs, and the ability to store information. For many situations, typical approaches are developed, such as SFT, instruction tuning, RLHF, PEFT, RAG, domain pretraining, and examples of their implementation.

Some EAD tasks that need in LLMs require a high level of security and confidentiality, and therefore the analyzed documents cannot be sent to LLMs accessible via the Web. For this purpose, local versions of LLMs which prevent data leakage into the public domain can be used, [12]. However, we have to take into account that such LLMs have significantly lower performance and produce less reliable results with a high number of "hallucinations," meaning that the quality of information analysis is considerably lower (although this strongly depends on the size of the model and its tuning for a specific task and NL).

Finetuning scenarios for open LLMs (through an API provider such as OpenAI, Azure OpenAI, Anthropic, etc.) include supervised fine tuning (on a small set of labeled pairs), alignment to modify style or response policy, functional integration, and special APIs (for example, fine tuning or demonstrations via API). The limitations and risks of using open models are high cost, lack of privacy, and lack of access to internal model weights.

LLMs are deployed locally (self hosted) or in a private cloud environment if it is necessary to keep sensitive data local, to guarantee compliance with legislation and protection of personal data (for example, medical data); if it is necessary to support many different adaptations for each task; if training is required under an organization's own response policy and internal expert preferences (for example, in a financial institution); or if combining an LLM with a local KB (vector storege) for factuality and instant updating (for example, a research institution integrates an LLM with a local

repository of publications from internal sources). If data are subject to restrictions, local fine tuning is almost mandatory.

Thus, analysis of research and implementation examples shows that open API models are convenient for rapid, low cost adaptations, tasks with less privacy requirements and prototyping, while local models are justified if full control over data is required, if complex domain adaptations are needed, or if applications are in regulated subject domains. Local finetuning of LLMs is applied to adapt the model to a narrow subject domain; to personalize behavior within an organization; to improve fact checking and justification of responses through local knowledge (RAG); and to ensure privacy and regulatory compliance. Initial parameters for local fine tuning are determined by such factors as model size (from 7B to 70B+ parameters), available computational infrastructure (number of GPUs and their memory), and the volume and quality of the domain corpus. Metrics for effectiveness evaluating of local finetuning are divided into: automatic (task oriented and retrieval oriented); expert assessments of usefulness, truthfulness, style, and safety; comparisons with results without fine tuning (ablation studies); and monitoring after deployment.

Retrieval Augmented Generation (RAG) is a paradigm that combines external information retrieval systems with language models, where LLM text generation is iriented on context retrieved from an indexed document corpus at inference time. The purpose of RAG is to compensate for the limitations of the parametric memory of LLMs (static knowledge, "knowledge cutoff") and thereby improve accuracy, factuality, and currency of responses, reducing the frequency of "hallucinations" (fabricated factual statements) in text generation. In its classical implementation, RAG consists of three components: a retriever (query encoder), a document corpus with a ranking mechanism, and a generator (conditional LM) that receives retrieved passages and generates the result [13].

It should be noted that due to the ambiguity of natural language, interpretation of some NL content fragments of documents requires reconciliation, additional verification, and consultation with subject domain experts. Therefore, information processing in such a system has to consist of several stages, with the possibility of returning to previous stages to make clarifications. To ensure flexibility and adaptability of such a system, it is advisable to divide its functions into separate modules and to organize communication among these modules according to the principles of service oriented programming: modules exchange data according to clearly defined rules and protocols, and changes in one module do not require reconfiguration of others (the need for changes may be caused by requirements for scalability, security, quality and speed of computations, or changes in conditions of software use).

4. Review of Research on the Use of LLMs for Semantic Annotation

We propose to combine the use of LLMs with semantic annotation of documents using subject domain terminology: it allows to identify meaningful elements of documents for further processing and simplifies the process of controlling the procedure for creating resulting documents for domain experts.

At present, the problem of LLM combining with semantic markup is only weakly represented in scientific publications, although some works address individual aspects of this task. For example, in [14] the authors investigate the problem of the automated creation of semistructured documents in the field of public administration, where templates do not always cover the full variability of structure. The system consists of a set of agents, and each one is responsible for a separate phase of document generation. A role-based instruction for the LLM is represented as a prompt part and applies semantic information about documents for markup construction. The system adapts templates to real cases, reducing the need for manual intervention.

Some researchers [15] consider the task of semantic markup of documents that change during editing. The system automatically updates the markup when the text is changed, using LLM to recognize semantic relations without explicit tagging.

In [16] the automation of literature review generation using LLMs is examined. The system annotates articles of selected subject domain according to proposed scheme. This approach

overcomes the limitations of human processing of large volumes of scientific information. LLM integrates data from more than 1000 articles. The quality of automatic reviews is not inferior to manual ones, and the hallucination rate is reduced to less than 0.5% with 95% confidence.

Another study of similar orientation [17] describes an open tool for automated construction of literature reviews that uses LLM to extract relevant information and to assess correspondence of results to the user query.

Thus, the conducted review shows that, despite the existence of research in related directions (semantic parsing, lexically constrained decoding, Wikidata data import, etc.), the scientific literature does not yet explicitly describe methods and models for applying LLMs to semantic markup generation.

5. Basic Requirements for the Technological Platform of Experience Acquicition from Documents

According to the purpose and specificity of the task, the technological platform for EAD has to support the following functions:

- accumulation and storage of different types of documents (IDs, RDs, normative documents, etc.), both NL and multimedia;
- creation and storage of a subject domain KB schema (ontological model or another type of formal knowledge representation), with support for its integration both with external ontologies and with accumulated content;
- structuring of content and generation of semantic annotation of documents using the subject domain KB schema, storage of these metadata, and provision of navigation within the content based on this annotation;
- tools for searching and formalizing rules that allow linking structural elements of RDs with information obtained from structured IDs;
- tools for explaining the construction of resulting CIOs, which provide access both to the information sources used (specific IDs and their structural elements identified by semantic annotation tools) and to the rules applied for their transformation (these may be rules extracted from normative documents or results of traductive inference based on analysis of classified IDs);
- tools for providing users with resulting information, ensuring both the form of RD presentation and the selection of information relevant to the needs of specific users.

The choice of models, methods, and specific software implementation for each of these functions depends on the domain specifics of the and the user informational needs. However, this set defines the core requirements for unified technological platform capable to adaptation to specific tasks. All defined functions are related to the creation, representation and analysis of knowledge, and it thus necessitates a unified KM system for information exchange between individual modules.

The tasks of EAD imply a certain sequence of stages for the creation and application of knowledge; however, if the environment or user needs change, these processing stages are repeated.

A preliminary stage in EAD document analysis is the creation of a thesaurus that contains a set of domain-specific semantic properties that can be used for semantic markup of documents. This thesaurus is built iteratively using such means as knowledge import from external ontologies (if such ontologies are available), linguistic analysis of standards and normative domain documents, and typical examples of Ids and RDs, as well as through consultations with experts.

At the next stage this thesaurus serves as a basis for semantic markup of IDs. It is important to note that semantic markup should only extract those content elements that can be useful for RD generation, rather than all existing structural and content elements of the document. A full linguistic

analysis can produce a highly complex ontological structure even for a single paragraph of NL text, but such level of detail is unnecessary for most specific tasks. If the thesaurus is supplemented or modified during the task solving, semantic markup of IDs is repeated, but only for the new or changed thesaurus elements. This approach is aimed at reducing computational costs and ensuring significantly faster document analysis.

Then it is necessary to determine the semantic relations between the thesaurus concepts and elements of document content: text fragments are transformed into the values of corresponding semantic properties and the relations between them are defined. For such linguistic analysis, it is advisable to employ LLM that h receives the thesaurus as a part of the prompt. and the documents requiring markup as input data. This stage provides semantically structured documents that can be processed for constructing the template with RD structure and populating this template with values.

All these stages should also be controlled explicitly by matching the LLM-generated template with available examples, and comparing the generated values with user requirements. Domain experts should be involved in the system training, identifying errors or deficiencies in the generated RDs. Following this, a prompt should be constructed that queries the LLM for the sources and rules that led to incorrect results, using the concepts and relations from the thesaurus. Analyzing these responses allows for system improvement and also addresses the issue of trust in the results generated by generative artificial intelligence.

At different stages of EAD either a single LLM or multiple LLMs can be used. The choice depends on the complexity of analysis, security and data protection requirements, the target NK and the task specifics.

6. Problem Definition

The EAD task requires the use of linguistic analysis tools to determine the semantics and structure of NL documents that record the experience gained in a certain domain, and the semantic transformation of the results of this analysis into the information objects that users need.

These documents accumulate the experience of mankind from various fields of knowledge, but obtaining this experience for various EAD tasks in a clear, usable form requires quite complex analysis and the use of external knowledge sources. Traditional machine learning methods based on logical inference are not designed to process such large amounts of information, and GenAI tools do not guarantee the correctness of the obtained results and do not provide a sufficiently complete explanation of the ways to obtain them.

We propose to separate EAD task stages of transforming a set of the natural language documents into complex information object and to fix the intermediate results of each of them using formalized semantic markup. Depending on the domain requirements for expressive power of knowledge representation, various formal models can be used for this purpose (from Semantic MediaWiki markup to OWL and RDF). In addition, many domains have generally accepted standards for metadata representation and tools that implement these standards, which can also be used for knowledge structuring.

The presence of an intermediate formalized representation of document semantics allows to check whether the LLM correctly interprets knowledge from NL documents, what domain concepts it associates with text fragments, what values based on these fragments it chooses and how exactly it determines the relations between these concepts.

In this research we propose an intelligent technological platform oriented on solving of various EAD tasks, analyze its functionality and consider solving of the various practical problems on its base.

7. “Linza” Technological Platform

Let us consider an example of a tool for solving EAD tasks. The technological platform “Linza” is designed for the intelligent processing of unstructured documents, enabling the transformation of

heterogeneous sets of NL documents with complex structures into resulting structured CIOs according to a branched set of user requirements [18]. It is oriented to perform EAD tasks of various complexity and volume. This functionality is achieved through the combination of semantic technologies, which allow the system to understand the content of information, with LLMs that provide linguistic analysis of document content and NL text generation. The main purpose of the system is to transform heterogeneous, often unstructured documents (raw data, semi-structured description of precedents and text cases, well-structured domain standards, instructions and regulations, etc.) into an integrated knowledge base that allows the generation of clear, structured and comprehensible conclusions and recommendations.

"Linza" system combines Semantic MediaWiki technologies and ontology-based knowledge formalization with GenAI tools for analyzing NL documents. Wiki-platform makes visible, understandable and editable the new domain knowledge acquired by LLMs from documents. Such integration allows aggregation the LLM power with the reliability and explainability of semantic analysis results.

"Linza" provides the following functional capabilities:

- Automation of routine tasks for processing and structuring IDs;
- Generation of recommendations based on examples and analysis of semanically marked documents;
- Availability of explanations for the proposed recommendations and multi-level inspection: unlike the "black box" of LLMs, the system allows evaluation of intermediate data interpretation results through semantic markup;
- Dynamic adaptation to changes in input data and modifications of user requirements.

The thesaurus used in "Linza" is a dictisonary of the domain basic terms and definitions that allows the system to interpret information unambiguously. It is based on the domain ontological model that defines a complete, structured set of concepts and relations within a chosen domain. This ontology can be integrated with the ontological model of "Linza", but it should be noted that here we consider two different ontologies: "Linza" ontology is relatively static, whereas domain ontologies are specific to each EAD task and can differ significantly in structure and volume. Such ontology-based approach ensures the formalization of knowledge and provides the foundation for applying machine learning methods, effective knowledge management, and providing clear and well-justified explanations for all recommendations.

In accordance with the goals for transforming documents into CIO we identify the main functional modules of "Linza" and determine the requirements for these modules, and only after this stage we can select technologies for their implementation and interaction mechanisms.

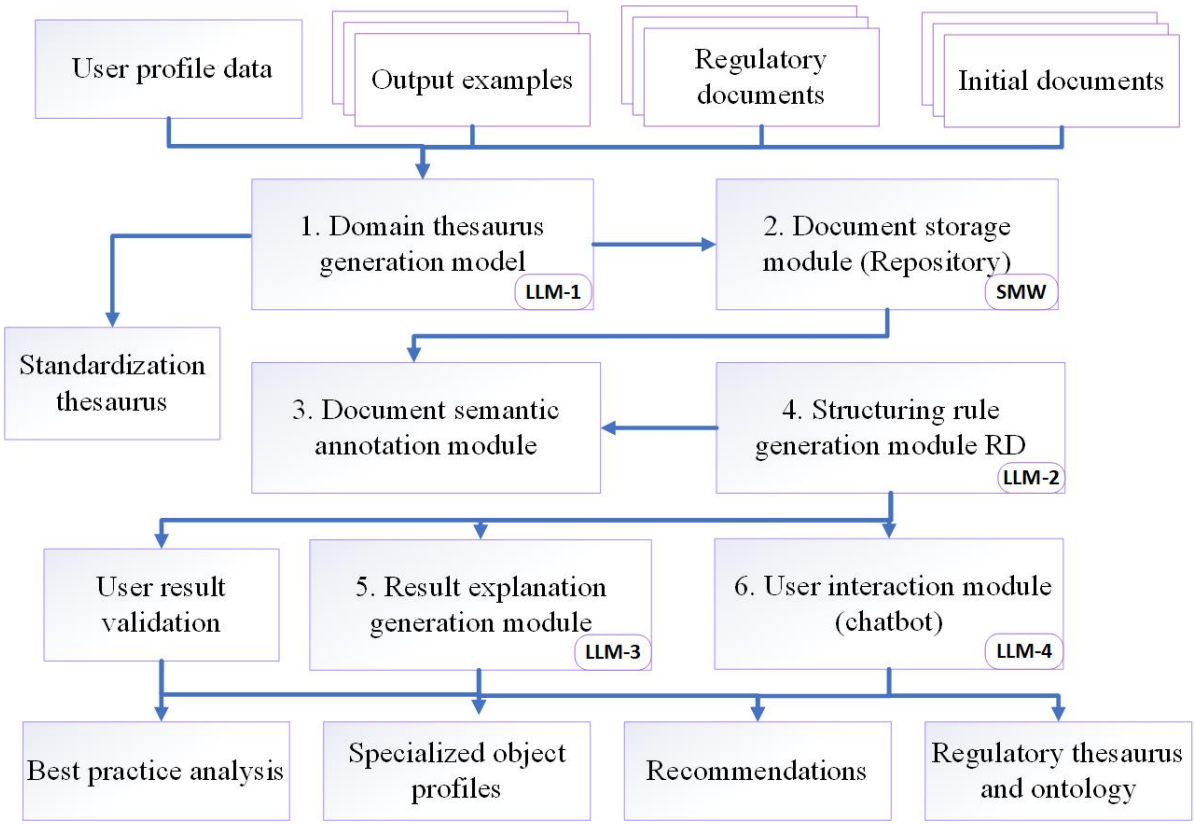

Figure 1: Functional modules of the "Linza" technological environment.

"Linza" contains six main functional modules (Fig. 1):

- Domain thesaurus generation module.
- Document storage module (repository).
- Document semantic markup module.
- Rule generation module for building RD.
- Result explanation module.
- User interaction module (chatbot).

In addition, "Linza" provides an *operational environment* that coordinates all document processing workflows from collection and analysis of data to transformation and verification, supporting secure and reliable data exchange between system modules responsible for document ingestion and preprocessing, semantic markup, knowledge modeling, response generation via LLM, and information integration. This operational environment also manages authorized user access. Sensitive user data remain under control and are not transmitted to external LLM, which is critical from an information security perspective. The system is further protected by access roles, comprehensive logging of all actions, and secure HTTPS connections, preventing unauthorized modifications or deletions. "Linza" implements a "human-in-the-loop" concept, ensuring full transparency and control. A human expert can supervise and validate each processing stage and its results, accessing all relevant information while not viewing documents of other users. This concept not only increases trust in automatically generated documents but also ensures their factual accuracy, which is essential for responsible decision-making.

The *domain thesaurus generation module* provides tools for constructing a consistent vocabulary that defines relations between base domain classes, their instances and instance properties. This formalized description based on the domain ontology enables all development participants to

consistently interpret data semantics and information exchange. Depending on the task, this thesaurus is: generated from the class subset of relevant ontology; constructed by document analysis (using LLM or other analytical tools); manually created by experts. In practice developers combines all three methods with iterative enrichment based on deeper domain research and user feedback. The module outputs an organized set of concepts (NL words and phrases) usable as semantic markup tags and components of document metadata schemas. It can automatically extract semantic properties from NL text (e.g., from the descriptions of precedents or from the normative documents) that can be used for ontology construction, semantic markup or LLM prompt generation. Here, a semantic property is a formalized knowledge fragment describing entity relations, characteristics or contextual dependencies.

The repository module supports storage of domain information (IDs, RDs, regularities, etc. and their metadata) at all stages of semantic structuring, analysis and conversion into knowledge elements for the resulting CIO. The main form of record keeping is semantic markup similar to Semantic MediaWiki. This syntax allows Semantic MediaWiki tools to support search and navigation in semantically annotated documents, though knowledge for multi-arity properties have to be described explicitly. Depending on task specifics, other metadata types can also be created and stored. Property values can include data (text, numbers, other constants) or links to other objects. To represent complex domain concepts, in addition to Semantic MediaWiki constructs such as [[property::value]], more complex constructs such as [[property::$value_1$+$value_2$+…+$value_n$]] can be used.

The document semantic markup module transforms NL documents into structured ones by linking content elements to thesaurus concepts. LLM are employed to extract relevant fragments. Annotated documents can be stored as a semantic wiki resource, enabling search, navigation, and data integration, and facilitating expert evaluation and LLM refinement [19].

The rule generation module identifies semantic relations between marked elements of IDs and RDs using knowledge from provided documentation and formalized domain sources. LLM and, if necessary, more advanced data analysis and ML tools are employed.

The result explanation module provides users access to applied rules and clearly shows which information from ID is used to create each RD element and what rules are used for transformation of ID elements into RD ones.

The chatbot offers a NL interface for personalized user interaction with the system. This chatbot allows the user to process the previous history of interaction with the system instead of having to enter their information every time.

LLMs serve as the primary linguistic analysis tool in “Linza”. Different LLMs that differ in function, power, and security can be utilize for various modules. It should be taken into account that many EAD tasks at different stages of analysis need to process protected data (secret information, personal data, etc.), and local LLMs can be used for this. Choice of the most appropriate LLM is based not only on its power, scope and orientation to the processing of a particular NL, but also on the context volume that this LLM can process. For many "Linza" tasks the last parameter is be the most critical.

Therefore, the following solution is envisaged: at first, much more powerful open LLMs that process “training” data examples are used for creation the semantic markup rules that allow linking NL fragments of input documents with semantic properties (and later with CIO structural elements). Open LLMs allow initial construction of the domain thesaurus and the subsequent definition of rules for recognizing semantic property values in text.

Then these rules are proposed as a prompt for local LLM that process secret data. This approach is based on the assumption that IDs that require protected processing have a similar structure and use a significantly limited subset of NL. This step involves a local LLM receiving these transformation rules along with the set of documents that require protection. Once semantic markup is performed, its results are stored in the repository – both the marked document itself and its semantic metadata separately.

The last stage involves analyzing CIO examples and identifying their primary structural elements. Depending on the task's security requirements for the examples, this analysis can be performed by:

- an open LLM (most efficient);
- a local LLM (most secure);
- a domain expert (highest quality).

In practice, any combination of these approaches is advisable.

The next stage is the most complex: it requires determining the information sources for values of the CIO structural elements (the values of semantic properties in IDs that meet specific criteria) and the algorithms used to compute these resulting values for RD elements. Such algorithms can range from simple operations (for example, "the number of items spent per day" is calculated as the sum of "number of items spent" across all documents where the "date" value meets the required criterion) to more complex computations. However, for many tasks, such simple analysis and document retrieval (based on a selected set of properties) are insufficient, and the provided set of CIO examples allows for multiple interpretations of such rules. In such cases, the transformation rules have to be obtained either from domain regulatory documents or directly from experts.

8. Ecosystem of Experience Extraction from Document

One consequence of approach proposed in this research that divides the EAD task into a set of relatively independent subtasks, is the need to integrate modules of these subtasks and the activities of their development teams. This process requires a clear and unambiguous understanding of the basic terminology, formalization of the functions of individual modules and the roles of all project participants. The results of the work of some modules provides input data for others, but functions of modules can be changed, expanded or transferred to other modules, therefore we need to formalize the development ecosystem itself.

Ecosystem approach to software development is currently used to create complex information systems and to develop various applications on a single software platform. The ecosystem concept serves as a powerful tool for formalizing the fundamental connections (both informational and material) within a complex system between its subjects and the resources that they jointly utilize in their activities, and the outputs of these activities that they exchange. This approach is used for a more precise modeling of systems where subjects collaboratively use certain resources and compete for access to them, and where the outputs of one actor's activities can be regarded as resources for others.

In contemporary scientific discourse, the term "ecosystem" is widely used not only in its literal sense as a description of the cohabitation of a set of living organisms (ecological entities) within a shared environment. It is considered also as a metaphor for modeling various types of systems characterized by a fixed set of actors, objects, and interaction types during collaborative activity (e.g., educational processes, software systems) [20]. Each ecosystem subject has its own set of interests and goals, and their realization is linked to the activities of other actors and to the ecosystem as a whole.

Software ecosystems (SECO) are used to model the process of designing and programming complex information systems that include heterogeneous components developed by both internal and external participants but leveraging a common software platform. The SECO concept enables the formalization of the actions of software developers and users [21]. At a high level of abstraction, such ecosystems describe the relations between programs and services that arise during their creation and usage. Existing SECOs differ significantly in terms of complexity, technological platforms and operating systems, users, domains of application, etc.

In 2003, SECO was defined as a set of software products developed together within a single technological environment. Later, this definition was expanded to include other elements such as a shared market for software tools, services and applications, as well as mechanisms for interaction

among developers to exchange information, resources and artifacts. More specific SECO definitions also consider aspects of software engineering, along with technical, social, and economic relations among ecosystem actors. Furthermore, now, due to the spread of GenAI, the question arises of how to classify LLMs – as biotic or abiotic components of the ecosystem. On one hand, they are products and resources of software developers; but on the other hand, they themselves generate software code and produce other resources.

Thus, we can consider SECO is a system that describes the interactions of a non-empty set of participants resulting in a set of software solutions and services, that work on a shared technological platform. It is advisable to record formally the current requirements for the system as a whole and for all its actors and objects, based on knowledge representation methods that ensure unambiguous interpretation. One such method is ontological analysis, which is widely used today for formalizing distributed knowledge across various domains [22]. An ontological model of SECO allows for precise and unambiguous description of all key components of ecosystem, as well as for setting priorities in development efforts for individual modules, allocating tasks, and defining a work plan. We propose the ontological model of the "Linza" ecosystem distinguishes biotic (subject) (Fig. 2) and abiotic (objects) components (Fig. 3) and defines the semantic relations between instances of the corresponding classes using object properties and data properties (Fig. 4).

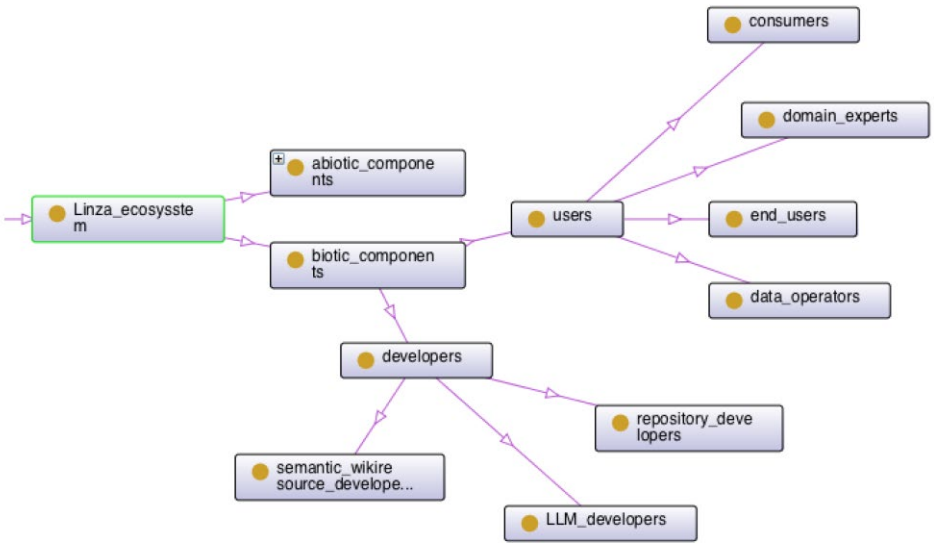

Figure 2: Biotic components of the ontological model of the "Linza" ecosystem

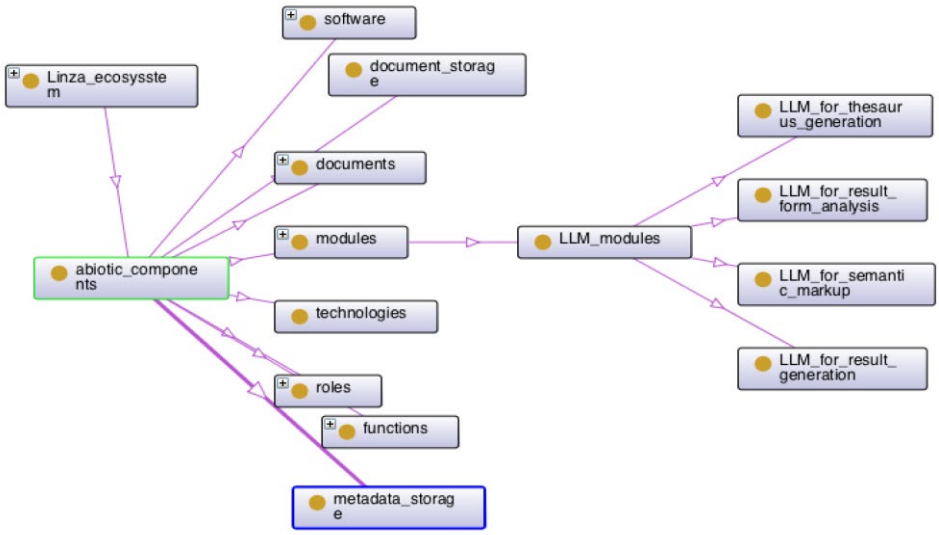

Figure 3: Abiotic components of the ontological model of the "Linza" ecosystem.

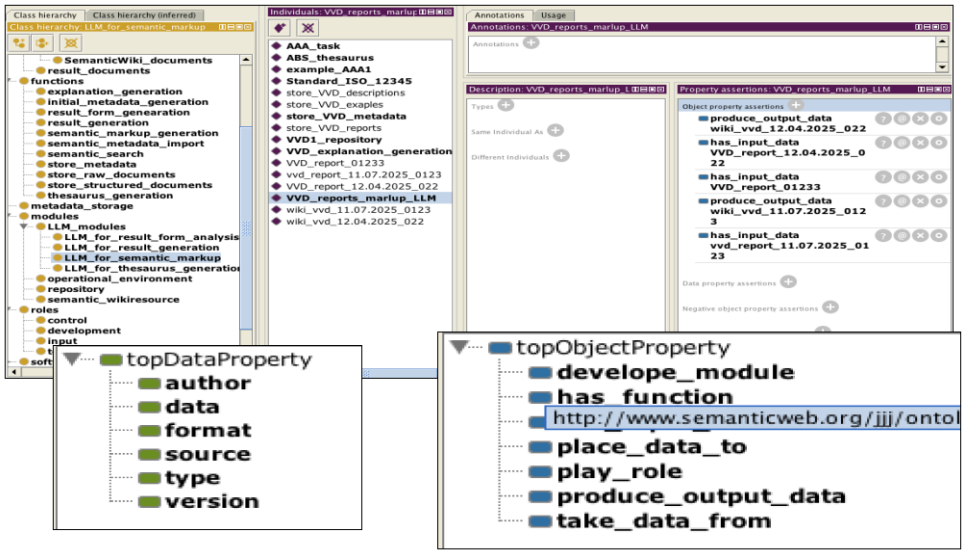

Figure 4: Class “LLM for semantic markup” instance of the "Linza" ecosystem.

This model is applied to turn a generalized description of the multi-level architecture of "Linza" into a clear plan for its development and operation, clearly define the functions of individual modules and the roles of developers and users.

9. Examples of the “Linza” Platform Use to Solve Practical Tasks

Let us consider several examples of adapting the “Linza” technological platform to solve applied EAD tasks of various levels of complexity.

"Linza" is result-oriented system in the sense that the basis for input information analysis is not, in the first place, the explicit user descriptions or formal rules (from instructions and regulations), but rather example-based analysis. Thus, system involves not only *domain deductive* reasoning but *domain traductive* reasoning (from particular to particular) also. Such approach leads to the

generation of RDs that do not meet user requirements (because traductive interpretation can be ambiguous). This ambiguity causes the need for iterative refinement of generated RDs and further retraining of used LLMs. As a result, the mere use of even powerful LLMs is insufficient, because the key challenge lies in evaluating the obtained results. Therefore, we use intermediate formalized representations of semantics recognized by LLMs from primary documents. The processing itself is performed in several stages. Depending on the task, certain processing stages may be omitted (e.g., thesaurus construction for formalized domain knowledge), or, conversely, be repeated or performed by different tools depending on the complexity of the input and output data.

Intelligent Assistant for a PhD Student. The task is to select a personalized set of learning objects (LOs) such as lectures, textbooks, online courses that correspond to the doctoral program requirements but extend it in accordance with the chosen research topic. LO selection takes into account the scientific interests of both the PhD student and his/her scientific adviser, additional domain information such as the student's ability to perceive information (knowledge of foreign languages, motivation, previous academic performance), as well as experience of other PhD Students**.** The main challenge lies in the fact that it is difficult to formalize an up-to-date domain of interest in scientific research because the model of it is highly specialized and cannot directly reuse an existing ontology without modification.

However, the task is simplified by the following factors:

- almost all documents are openly accessible (except for a small subset of personal data that are not mandatory for generating recommendations);
- the volume of processed documents and the number of system users are relatively small;
- there are existing standards for profiling students and instructors; and the requirements for resulting documents are not overly complex.

IDs in this task are: PhD student profiles, supervisor profiles, descriptions of academic courses, unstructured descriptions of educational resources, precedents (information about previous PhD students and their outcomes); regulatory documents concerning PhD education and student requirements.

RDs ara ordered sets of LOs.

The domain thesaurus generation module for this task is relatively complex: it processes domain-standard student profiling information and LOs metadata, as well as applies elements of the domain ontology and documents from the institution's website.

The document storage module (repository) does not require complex access control procedures and can be implemented using a semantically extended wiki resource (Semantic MediaWiki). Most documents do not require separation between the original view and semantic markup; therefore, all information can be represented as sets of wiki pages.

The semantic markup module ensures generation of LO metadata in domain terms. Information about users is entered immediately using markup or is exported from the institutional website.

The rule generation module for RD constructing acquires rules from regulatory documents (e.g., regarding the use of new resources), while the LLM leverages the domain ontology to determine the degree of semantic similarity between these resources and the dissertation topic.

The explanation module provides users with parameter values and sets of keywords used to match LOs pertinent to their needs.

The chatbot explains the process of learning, retrieves relevant regulatory documents and proposes precedent examples using the scheme: "A PhD student with parameter values similar to user who previously used resource X had a successful results".

This is a relatively simple example of "Linza" usage that not only provides a serviceable practical outcome but also enables comparison of the effectiveness of different open LLMs at various stages of processing .

Constructing an Information System Security Profile. This example of using "Linza" enables the creation of a document used for the certification of information systems based on a set of example

pairs “User Information – Resulting Document” and a large number of up-to-date standards. Compared to the previous case, this example is significantly more complex due to the large volume of regulatory documents and the need to construct RDs with a complex formalized structure. However, regulatory documents can be analyzed by open LLMs that supply more quick and high-quality processing of information then local ones, while rules for RD constructing are quite strictly formalized and use a limited subset of NL. Therefore, the need for LLM retraining is unlikely. Significant problem deals with the generation of the domain thesaurus that has to contain a complete set of parameters influencing the RD composition. The relations between domain conceprs are too complex to be represented by the expressive means of Semantic MediaWiki. Therefore, this, information source becomes only one component of the repository. A more sophisticated metadata schema and storage mechanism should therefore be considered. Moreover, transformation rules from IDs to RDs documents are also complex and not reducible to a set of production rules. Therefore, the format for their storing and representation to users requires additional instrumental tools (for example, by means of the ontology editor Protégé).

Retrieval and Analysis of Precedents in the Accumulated Experience of the Armed Forces of Ukraine
During combat operations, military units accumulate valuable, critically important experience, but it is stored in an unstructured manner that causes difficulties to promptly transform it into clear, verifiable recommendations for other units to improve battlefield effectiveness. Therefore, this information from combat units requires semantic interpretation and unification with regulatory documents to provide analytical support for identifying, formalizing and generalizing current shifts in best practices for operational-tactical and operational-strategic decision-making under combat conditions, enabling their integration into military training. This EAD task is the most complex among all analyzed cases. It requires analyzing a significantly larger number of input documents (reports on real events, after-action reviews from multiple units), considering their relevance, as well as domain information on the adversary, directives of the General Staff of the Armed Forces of Ukraine, combat statutes, NATO standards, etc.). These documents follow certain rules but are prepared in a far less formalized manner and can contain fuzzy, incomplete and incorrect data (misspellings, various types of abbreviations, ambiguous acronyms, etc.). Moreover, this information is secret, meaning access to it is restricted, that only local LLMs can be used for its analysis, which reduces quality of analysis. The domain thesaurus is developed from both such IDs and regulatory sources (such as capability codes and capability requirements in NATO defense planning, characteristics of equipment and units). External knowledge sources (e.g., information about geographical objects and weather conditions) can also be required. Situations and relations described in the reports have a complex structure, requiring significant effort for alignment and determination of semantic similarity. Moreover, generating of recommendations and search for precedents have to take into account individual characteristics of the end users to provide them only relevant information. Additional requirements for LLMs arise at various stages of processing according to the functions of the modules and the requirements for information security. Therefore, retraining of different local LLMs tailored to the tasks of specific modules is advisable. There are additional requirements for explaining the results of the analysis and for their unambiguous interpretation caused by critical consequences of the low-quality outcomes. Therefore, the explanation module and chat-bot should be clearly integrated with the domain thesaurus.

Declaration on Generative AI

During the preparation of this work, the authors used AI programs Chat GPT 4o and Copilot for grammar and spelling check. After using this services, the authors reviewed and edited the content as needed and take full responsibility for the publication’s content.

Conclusion

Our research is aimed at integration achievements in the field of knowledge modeling and analysis with modern directions of AI development to solve the task of the experience acquisition and use. The work proposes the integration of GenAI with modern semantic wiki technologies that provides a combination of the LLM power in linguistic processing of NL documents of with the guarantee and explainability of results of knowledge management systems. We consider the advantages and challenges of the proposed approach demonstrates by the intelligent technological platform "Linza" that provides tools for acquiring and implementing experience from heterogeneous NL documents. The main AI elements used in "Linza" modules are: ontological modeling of domain knowledge that provides unified terminology system an thesaurus of domain used for content markup of document, LLM prompt engineering and recommendation formulation; Semantic markup LLM-based means that structure NL content with thesaurus concepts for transforming the set of documents into the knowledge base; Semantic MediaWiki provide the technological base for storage, semantic search, navigation and verification of structured documents; Intelligent user assistant (a chatbot based on an LLM and the constructed knowledge base) enables NL dialogue for precedent retrieval and pattern discovery, and also acts as a tool for refining real user needs and improving the thesaurus; Machine learning algorithms for constructing decision trees and recognizing situations based on training datasets, classifying experience, discovering causal relations, and forming sequences of successful actions.

To reduce the risk of “hallucinations” (erroneous generation caused by insufficient training or overfitting of the neural model to a particular aspect) and to increase accuracy, LLMs are integrated with semantic knowledge bases and ontologies. Thanks to semantic wikis, LLM operation is no longer a “black box”: the user can control the reasoning process of the system and assess the correctness of content interpretation.

References

[1] J. Rogushina, I. Sinitsyn, Application of semantic technologies in the digital transformation of the National Academy of Sciences of Ukraine. In: Proceedings of the 1st Workshop “Software Engineering and Semantic Technologies” (SEST 2025), CEUR Workshop Proceedings, Kyiv, Ukraine, May 13-14, Vol. 4053 (2025), 108–122. URL: https://ceur-ws.org/Vol-4053/paper7.pdf.

[2] J. V. Rogushina, Ontological Approach in the Smart Data Paradigm as a Basis for Open Data Semantic Markup, In: Proceedings of the 7th International Conference on Computational Linguistics and Intelligent Systems. Volume III: Intelligent Systems Workshop, Kharkiv, Ukraine, April 20-21, 2023, volume 3403 of CEUR Workshop Proceedings, CEUR-WS.org, 2023, pp. 12–27. URL: https://ceur-ws.org/Vol-3403/paper2.pdf.

[3] Y. Chang, X. Wang, J. Wang, Y. Wu, L. Yang, K. Zhu, X. Xie, A survey on evaluation of large language models. ACM Transactions on Intelligent Systems and Technology, 15(3) (2024) 1–45. DOI: 10.1145/3641289.

[4] C.-Y. Lin, ROUGE: A package for automatic evaluation of summaries. In: Text Summarization Branches Out, Association for Computational Linguistics (2004), 74–81. DOI: 10.3115/1073445.1073465.

[5] Y. Geifman, R. El-Yaniv, Selective classification for deep neural networks. Advances in Neural Information Processing Systems, 30 (2017). URL:https://proceedings.neurips.cc/paper_files/paper/2017/file/4a8423d5e91fda00bb7e46540e2b0cf1-Paper.pdf.

[6] M. Hardt, E. Price, N. Srebro, Equality of opportunity in supervised learning, In: Proceedings of 30th Conference “Neural Information Processing Systems” (NIPS 2016), Barcelona, Spain, 29 (2016), pp.1-9. URL: https://proceedings.neurips.cc/paper/2016/file/9d2682367c3935defcb1f9e247a97c0d-Paper.pdf.

[7] K. Zhu, J. Wang, J. Zhou, Z. Wang, H. Chen, Y. Wang, L. Yang, W. Ye, N. Z. Gong, Y. Zhang et al., PromptBench: Towards evaluating the robustness of large language models on adversarial prompts. (2023). URL: arXiv:2306.04528 .
[8] S. Gururangan, A. Marasović, S. Swayamdipta, K. Lo, I. Beltagy, D. Downey, N. A. Smith, Don't stop pretraining: Adapt language models to domains and tasks (2020). URL: arXiv:2004.10964
[9] J. Howard, S. Ruder, Universal language model fine-tuning for text classification, (2018). URL: https://arxiv.org/pdf/1801.06146.
[10] J. Wei, M. Bosma, V. Y. Zhao et al., Finetuned language models are zero-shot learners, (2021). URL: https://arxiv.org/pdf/2110.08207.
[11] S. Hatgis-Kessell, W. B. Knox, S. Booth, S. Niekum, P. Stone, Influencing humans to conform to preference models for RLHF, (2025). URL: https://arxiv.org/pdf/2501.06416.
[12] B. E., Perron, H., Luan, B. G., Victor, O.,Hiltz-Perron, J. Ryan, Moving beyond ChatGPT: Local large language models (LLMs) and the secure analysis of confidential unstructured text data in social work research, Research on Social Work Practice, 35(6) (2025) 695-710.
[13] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, … S. Riedel, Retrieval-augmented generation for knowledge-intensive NLP tasks. In: Proceedings of Neural Information Processing Systems 33 (NeurIPS 2020), Curran Associates Inc., 2020. URL: https://proceedings.neurips.cc/paper_files/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html.
[14] E., Musumeci, M., Brienza, V., Suriani, D.,Nardi, & D. D. Bloisi, (2024). LLM based multi-agent generation of semi-structured documents from semantic templates in the public administration domain. In: International Conference on Human-Computer Interaction, pp. 98-117. Cham: Springer Nature Switzerland. URL: https://arxiv.org/pdf/2402.14871.
[15] E. Misback, Z. Tatlock, S. L. Tanimoto, Magic Markup: Maintaining document-external markup with an LLM. In: Companion Proceedings of the 8th International Conference on the Art, Science, and Engineering of Programming, ACM, 2024, pp. 22–30. DOI: 10.1145/3640537.3640559
[16] S. Wu, X. Ma, D. Luo, L. Li, X. Shi, X. Chang, J. Gong, A survey on large language model for recommendation, (2023). URL: https://arxiv.org/pdf/2409.16694.
[17] C. Y. Haryanto, A framework for legal information retrieval using semantic search and fine-tuned large language models, International Journal of Computer Science and Network Security (IJCSNS), 24(4) (2024) 161–168.
[18] I. P. Sinitsyn, J. V. Rogushina, K. Y. Yurchenko, Integration of large language models with semantic processing tools as an instrument for knowledge digitization. Problems in Programming, 2 (2025) 63–76. URL: https://pp.isofts.kiev.ua/index.php/ojs1/article/download/838/889.
[19] J. Rogushina, I. Grishanova, Ontological methods and tools for semantic extension of the MediaWiki technology, In: Proceedings of the 12th International Scientific and Practical Conference of Programming (UkrPROG 2020), CEUR Workshop Proceedings, Kyiv, Ukraine, May 13-14, Vol. 2866, 61-73. http://ceur-ws.org/Vol-2866/ceur_61-73Rogushina6.pdf.
[20] W. Geary et al., A guide to ecosystem models and their environmental applications. Nature Ecology & Evolution, 4(11) (2020) 1459–1471. DOI: 10.1038/s41559-020-01298-8.
[21] K. Manikas, K. M. Hansen, Software ecosystems. A systematic literature review, Journal of Systems and Software, 86(5) (2013) 1294–1306. DOI: 10.1016/j.jss.2012.12.026.
[22] T. R. Gruber, What is an ontology? Stanford Knowledge Systems Laboratory Technical Report KSL-92-01 (1992). URL: http://www-ksl.stanford.edu/kst/what-is-an-ontology.html.