MAKING MEDICAL EXPERTS FIT4NER: TRANSFORMING DOMAIN KNOWLEDGE THROUGH MACHINE LEARNING-BASED NAMED ENTITY RECOGNITION

Florian Freund, Philippe Tamla, Frederik Wilde and Matthias Hemmje

Faculty of Mathematics and Computer Science, University of Hagen, 58097 Hagen, Germany

ABSTRACT

This study presents a comprehensive survey examining the criteria used by Machine Learning (ML) experts in selecting and comparing Names Entity Recognition (NER) frameworks. The survey revealed that while performance is a key criterion, expert opinions vary significantly, highlighting the need for a flexible system that considers various criteria alongside performance. Based on the survey results, a system was developed using the structured Nunamaker methodology to assist medical experts in both comparing NER frameworks and training ML-based NER models. The prototype, including its user interfaces, was qualitatively evaluated using the Cognitive Walkthrough method. The paper concludes with a summary and an outlook on future research.

KEYWORDS

Natural Language Processing, Named Entity Recognition, Machine Learning, Cloud Computing, Medical Expert Systems, Clinical Decision Support

1. INTRODUCTION AND MOTIVATION

Medical experts possess essential domain knowledge that plays a crucial role in the evidence based development of **Clinical Practice Guidelines (CPGs)** [1]. CPGs serve as standardized recommendations that assist physicians in making well-informed decisions for the optimal treatment of their patients, ultimately minimizing patient risk [2]. For example, CPGs provide specific protocols for managing chronic diseases like diabetes, outlining recommended blood glucose targets, medication choices, and lifestyle modifications. The development of CPGs relies heavily on extensive sources of information, often presented in large volumes and natural language, such as clinical study reports, case studies, and scientific literature [1]. For instance, a clinical study on a new cancer treatment might contain thousands of pages detailing patient responses, adverse effects, and long-term survival rates. Medical experts must sift through these massive datasets to extract relevant findings, making the process time-consuming and prone to information overload [3]. **Named Entity Recognition (NER)**, a subfield of **Natural Language Processing (NLP)**, helps convert unstructured text into structured data, making knowledge extraction more efficient [4]. For example, an NER model can identify and categorize critical entities such as drug names (e.g., "Metformin"), diseases (e.g., "Type 2 Diabetes"), and medical procedures (e.g., "MRI scan") from clinical documents. Recent advancements in NER have significantly improved accuracy by leveraging **Artificial Intelligence (AI)** techniques, including **Machine Learning (ML)**, Deep Learning, and **Large Language Models (LLMs)** [4]. However, achieving high performance in specialized domains like medicine requires the development of

domain-specific NER models tailored to recognize intricate medical terminology and abbreviations (e.g., "CABG" for Coronary Artery Bypass Graft) [1], [4]. The creation of ML-based NER models is complex, requiring expert guidance throughout the process to ensure accuracy and reliability [1]. With the increasing adoption of ML-based NER, research activities and the availability of both local and cloud-based NER tools and frameworks have expanded significantly [5]. For example, platforms like spaCy, SciSpacy, and Google's AutoML offer pre-trained and customizable NER models for biomedical text processing. As a result, domain experts often struggle to keep up with rapid developments, compare various NER tools, and determine the most suitable ones for their specific needs, such as extracting patient symptoms from **Electronic Health Records (EHRs)** or analyzing drug interactions in pharmacovigilance studies [6].

Building on the recognition that ML-based NER is a vital technology for medical experts, this research is driven by several key projects. The **RecomRatio** project [7], initiated by the University of Bielefeld in 2018, aims to assist medical professionals in therapy decision-making by extracting arguments for or against specific medical treatments from the medical literature using ML-based NER and the spaCy framework [8]. The extracted data is stored in a knowledge database to support clinical decision-making. Following RecomRatio's outcomes, the **Artificial Intelligence for Hospitals, Healthcare & Humanity (AI4H3)** project focuses on enhancing the transparency and explainability of medical decisions through AI [9]. It proposes a layered architecture with a central hub, the "KlinGard Smart Medical Knowledge Harvesting Hub", which serves as a registration point for AI modules applicable in natural language text analysis. This hub architecture facilitates the decentralized integration of heterogeneous data and AI modules, enabling the efficient integration of information sources, including medical literature, electronic health records, and healthcare social media data, all critical for **Clinical Decision Support (CDS)**. The **Cloud-based Information Extraction (CIE)** project, building on the AI4H3 hub architecture, provides cloud-based resources such as computing power and storage for the automated extraction of natural language texts using ML techniques [10]. These resources support end-to-end NER pipelines in a cloud environment and optimize resource allocation. **Framework-Independent Toolkit for Named Entity Recognition (FIT4NER)**, another project within the AI4H3 context, aims to empower medical experts to utilize various AI-based text analysis techniques, including NER, for efficient **Information Retrieval (IR)** in developing CPGs [6]. It employs the **Content and Knowledge Management Ecosystem Portal (KM-EP)** [11], co-developed by the Faculty of Mathematics and Computer Science at FernUniversität Hagen [12] and the FTK e.V. Research Institute for Telecommunications and Cooperation [13], to facilitate knowledge management across multiple domains. KM-EP manages documents such as medical research findings and clinical studies, serving as an evidence base for the development of CPGs [6]. To enhance IR, KM-EP offers document classification features using NER, along with a faceted search engine built on these classifications [14]. The necessary ML-based NER models are integrated into KM-EP and should ideally be trained by medical experts. The medical field relies on a wide array of domain-specific terms and abbreviations, which are often ambiguous or have variations in spelling [15]. Therefore, involving medical experts who consider specific terms, abbreviations, and potential spelling errors is crucial for training new models and achieving optimal results [16]. However, the dynamic nature of NER research presents several challenges for experts. First, they must compare various tools to identify NLP features such as **Named Entities (NEs)** and entity relations and select the most suitable solution for their tasks [6]. This comparison is demanding, as NLP practitioners often find it difficult to clearly determine which software performs best and which tools efficiently extract, analyze, and visualize NLP features [5]. Second, choosing the right tool is critical, as it significantly impacts the accuracy of analytical tasks [17]. Third, users often lack the necessary computational and storage resources to train high-quality NER models within their domain [10]. While cloud computing could potentially address this issue, experts frequently lack the requisite knowledge

and experience to effectively utilize this technology [10]. The primary research objective of this work is to develop a flexible NER system that aids medical professionals in efficiently analyzing domain-specific texts while addressing challenges such as selecting and comparing new NER frameworks. To achieve this goal, the following research questions have been defined and will be addressed in this study: *(RQ1) What are effective methods for domain experts to experiment with and compare NER frameworks? (RQ2) How can an integrated and distributed information system be developed that enables domain experts to apply ML-based NER methods and supports the continuous development of NER frameworks?*

We employ the well-established Nunamaker methodology [18] to systematically answer our research questions. This approach guides the development of information systems through multiple **Research Objectives (ROs)** encompassing observation, theory building, implementation, and evaluation phases. Section 2 address the observation objectives by reviewing the current state of the art. Section 3 focuses on theory building objectives, developing models to aid medical experts in training ML-based NER models. Section 4 sets out the system development objectives, which include describing a prototype system for training ML-based NER models. Section 5 describes the experimentation objectives and describes the execution and analysis of expert tests using the prototype. Finally, Section 6 summarizes the results of the study.

2. STATE OF THE ART IN SCIENCE AND TECHNOLOGY

This chapter explores the observation phase, providing the background of this work and relevant research activities. It aims to review the state of the art and identify potential **Remaining Challenges (RCs)** in the addressed domains. First, NER and the challenges of dealing with various NER tools are examined. Second, a short overview of cloud technologies is provided. Finally, a survey is presented to identify the criteria ML experts use to evaluate NER tools and frameworks, highlighting the key challenges they face in selecting the most suitable solutions.

NER is an NLP technique that aims to extract NEs from unstructured text documents [19]. A NE is a word or phrase that refers to a specific entity such as a person, place, or organization. NER is a crucial technique used in various applications, including IR [20], question answering systems [21], machine translation [22], and social media analysis [23]. In the medical domain, NER plays a pivotal role in **Clinical Decision Support Systems (CDSSs)** and enables clinical information mining from **Electronic Health Records (EHRs)** [24]. In recent years, NER has seen significant progress due to the development of new techniques and models, including deep learning [25]. These advancements have led to substantial improvements in the performance of NER systems, making NER one of the most extensively researched NLP tasks today [25]. In NER, there are different techniques available, including traditional, ML-based, and hybrid approaches [26]. Traditional NER approaches rely on methods that use manually created rules or are dictionary-based. Although these systems are often efficient and accurate, they are also limited by fixed rules or dictionaries and do not generalize well across different domains and languages [26]. ML-based approaches to NER have gained popularity in recent years, mainly due to the availability of large annotated datasets and advancements in deep learning techniques [25]. These approaches are capable of efficiently processing unstructured and large datasets and achieve superior results. Instead of relying on fixed rules or dictionaries, ML-based NER uses statistical models that learn to detect NEs from annotated data through a process of training and testing. ML techniques are divided into supervised, unsupervised, and semi-supervised learning [27]. Supervised learning [27] relies on manually annotated data to train a model, where the model learns to predict the labels of unseen data. Unsupervised learning [27], on the other hand, relies only on statistical algorithms to detect patterns from unlabeled data. Semi-supervised learning [27] combines these two approaches by training a model with a small set of annotated data and using it to label a larger set of unlabeled data, thus improving the accuracy of the model. In recent years, pre-

training large language models such as BERT [28], GPT-2 [29], and RoBERTA [30] on large corpora have shown remarkable improvements in NER performance. These models can achieve state-of-the-art performance on NER tasks and can efficiently finetune on smaller datasets for domain-specific tasks. Although further improvements can be made, AI advancements have already made significant progress in addressing complex NER challenges. The research field of NER continues to evolve rapidly, with new and innovative tools being developed to address different challenges and use cases. Therefore, it is crucial for ML experts to compare and evaluate the performance of available NER tools and to select the one that best fits their specific task, such as training and fine-tuning ML models on custom datasets.

Amazon Web Service (AWS) launched in the early 2000s, pioneering the concept of cloud computing by offering scalable computing resources on demand as a service [31]. This groundbreaking technology has since evolved into a widely available solution that offers vast amounts of computing resources at any given time. The availability of scalable and costeffective cloud computing has revolutionized the field of AI by providing a scalable and costeffective platform for creating, training, and deploying AI models. ML-based NER is one of the many AI applications that have benefited from the cloud's capabilities [10]. The unprecedented growth of data has made it challenging to manage and analyze large amounts of information using local compute resources [25]. To tackle this issue, leading providers such as AWS, Microsoft Azure, and Google Cloud Platform offer cloud-based platforms at various levels of abstraction, including **Infrastructure as a Service (IaaS)**, **Platform as a Service (PaaS)**, and **Software as a Service (SaaS)** [31]. These platforms provide the necessary computing resources and tools to store, process, and analyze massive amounts of data efficiently and cost-effectively. Cloud-based ML platforms not only provide computing power (IaaS) but also offer a comprehensive suite of tools and services for data processing, model training, and deployment (PaaS). These platforms make it easy to scale performance up or down as needed, even for demanding applications with real-time requirements. Cloud providers offer not only cloudbased ML platforms but also NLP and NER services. These services include pre-built models and **Application Programming Interfaces (APIs)** that enable users to easily incorporate AI functionality into their applications without requiring extensive expertise in the AI domain [10]. To leverage cloud-based resources effectively, ML experts must carefully evaluate which level of abstraction and which cloud-based services from which provider to use. Furthermore, utilizing cloud-based resources requires familiarity with the relevant technologies, including understanding their strengths, limitations, and best practices [32]. Although cloud technology offers many benefits, including scalability and cost effectiveness, there are legitimate concerns about privacy, security, and ethical implications, particularly in the medical field [33]. As a result, ML experts must carefully consider these factors and evaluate whether cloud-based resources can be used while still meeting regulatory and ethical standards. In summary, cloud technology has rapidly evolved into a powerful platform for creating, training, and deploying AI models. However, ML experts face the challenge of determining whether and which cloudbased resources to deploy, requiring careful evaluation of factors such as scalability, cost, security, privacy, and ethical implications.In recent years, several scientific papers have compared and evaluated NER tools for various application domains, such as formal and social media texts [34], software documentation [35], historical texts [36], news sources [17], [37], and specific languages [38]. Pinto et al. [34] conducted a study to compare and analyze the performance of multiple NLP tools, including their effectiveness on formal and social media texts in four commonly used NLP tasks, which include NER. Their findings suggest that it is a challenge *"to select which one to use, out of the range of available tools"*, and *"this choice may depend on several aspects, including the kind and source of text"* [34]. Al Omran et al. [35] emphasized the importance of choosing the right NLP library for tasks like tokenization and POS tagging, as these significantly impact NER performance. Won et al. [36] demonstrated that NER performance is corpus-dependent and can be enhanced by combining different tools, a finding also supported by

Aldumaykhi et al. [38] for Arabic texts. Weiying et al. [17] conducted benchmarking for enterprise applications, while Schmitt et al. [37] noted the challenges in NER tool comparability and called for structured comparison methodologies. Jehangir et al. [19] analyzed various NER approaches, recommending a combination of deep learning and rulebased methods for optimal results. Overall, their findings suggest the need for contextdependent selection and comparison of NER tools, considering corpus-specific performance and the potential for tool combination.

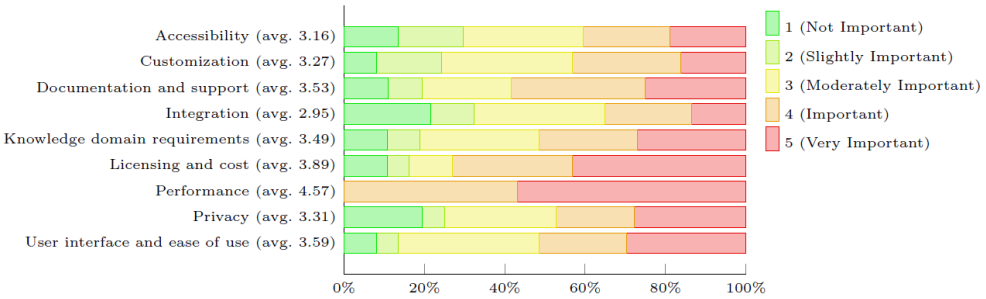

Figure 1 Priority Distribution per Selection Criteria

However, these studies did not address the challenges ML experts face when comparing NER tools to find the most suitable solution for their projects. A new study was conducted in the fall of 2024 [5] to fill this research gap. The Kasunic's survey research methodology [39] was employed to gather insights from nearly 90 NLP experts regarding their experiences in comparing and selecting NER frameworks and tools. Responses were received from 23 experts, most of whom had advanced degrees, such as a PhD or Master's. While most of the experts were from the field of computer science, there were also participants with backgrounds in history or economics. Figure 1 details the evaluations of participants of various selection criteria across all NER tools and frameworks, revealing performance (average 4.57) as the most critical factor. The consistently high prioritization of performance underscores the need for NER tools to deliver accurate and reliable results in different operational contexts. Questions were also posed regarding cloud-based tools, such as Microsoft Azure Cognitive Services. Especially, for these cloud-based tools, licensing and cost (average 3.89), as well as user interface and ease of use (average 3.59), also rank highly. This reflects the growing importance of affordability and accessibility in encouraging adoption among diverse user groups, such as newbies and ML experts. In contrast, factors such as integration (average 2.95) and customization (average 3.27) were moderately important, suggesting that participants found these areas less critical when evaluating NER tools. Privacy (average 3.31) received mixed scores, signaling varying levels of concern depending on the application and deployment model. In general, it can be concluded that all the criteria were considered relevant. This suggests that the importance of criteria is highly dependent on the specific project and that there are no criteria that can be universally deemed unimportant. This is also illustrated by the fact that there are different specific challenges for NER, depending on which knowledge domain NER is to be applied [4]. Numerous studies highlight the importance of performance in NER tools and frameworks by comparing their effectiveness [17], [37], [40]. Therefore, a supportive system should be designed with flexibility to accommodate diverse evaluation criteria, ensuring adaptability to project-specific needs while maintaining high model training performance (RC1). When analyzing the results of cloud-based and locally installable NER tools and frameworks, and calculating the average performance values, notable insights emerge. Table 1 illustrates the average results for both types of tools, highlighting the differences (Delta) between them. For locally installable systems, documentation and support play a critical role, as evidenced by a Delta of -1.14. This observation aligns with findings from related studies, where documentation is frequently emphasized as a key factor in evaluating NER tools. For instance, Schmitt et al. argue that criteria such as documentation

should be carefully assessed before selecting and deploying an NER solution [37]. However, for cloud-based systems, the user interface and ease of use are particularly relevant (Delta 1.09). Tamla et al. highlighted that managing cloud-based resources poses challenges for both beginners and ML experts [10], emphasizing the importance of user interface design and usability. Similarly, Kurdi et al. recognized that an intuitive interface significantly enhances the usability of cloud-based services [41]. Therefore, users of cloud-based NER services would greatly benefit from systems designed to simplify their use, making user-friendliness a crucial requirement (RC2).

Table 1 Comparison of Average Priority per Selection Criteria

Criteria	**Cloud**	**Local**	**Delta**
Accessibility	3.57	3.0	0.57
Customization	3.43	3.31	0.12
Documentation and support	2.57	3.71	-1.14
Integration	2.57	3.1	-0.53
Knowledge Domain Requirements	3.43	3.45	-0.02
Licensing and cost	3.71	3.9	-0.19
Performance	4.57	4.55	0.02
Privacy	3.14	3.29	-0.15
User interface and ease of use	4.43	3.34	1.09

The responses to the question *"How hindering have the following challenges or limitations with < selectedTool > been in the past?"* for all NER tools and frameworks are presented in **Fehler! Verweisquelle konnte nicht gefunden werden.**. The responses are surprising in that very few challenges were classified by participants as very hindering. The most frequently cited issue was "Time and effort to learn the new framework" (average 2.84). This was followed by "Performance Issues" (average 2.57), which is consistent with the findings in Figure 1. "Challenges with integration into existing applications" were mentioned the least as a hindrance (average 2.05). Other challenges, such as "Cost" (average 2.36), "Lack of support" (average 2.32), and "Lack of documentation" (average 2.30), were ranked mid-range but low. Each challenge was mentioned at least once as hindering or very hindering, supporting the assertion that the requirements for NER tools and frameworks are project specific. In general, it can be concluded that reducing the time and effort required to learn new frameworks is essential. Here, also, the results of this question were grouped with respect to cloud-based and locally installable NER tools and frameworks and the average value was calculated. As shown in Table 2, the time and effort required to learn the new framework are particularly restrictive for locally installable systems (Delta -0.57). There, a system designed to support the comparison and selection of NER tools and frameworks must also enable users to work efficiently across different platforms. For locally installed solutions, reducing the adoption effort through intuitive software and seamless integration is crucial (RC3). In contrast, for cloud-based services, cost remains a major barrier to adoption (Delta 1.79). Therefore, effective cost management and optimization strategies, such as pay-as-you-go models and resource monitoring, are essential to ensure cost efficiency [42] (RC4). The survey emphasized that performance is a key criterion, though expert opinions varied significantly. Notably, every specified selection criterion was deemed important or very important at least once, suggesting that their relevance depends on the specific project. Therefore, a supportive system must be flexible enough to accommodate diverse criteria while ensuring strong model training performance (RC1). A distinction is made between cloud-based

services and locally installed tools. For cloud-based NER services, cost efficiency and userfriendliness are particularly crucial, making them essential requirements for any system utilizing cloud-based resources (RC2, RC4). In contrast, for locally installed solutions, minimizing the adoption effort is critical, which can be achieved through well-designed software integration (RC3). After reviewing the state of the art in science and technology and identifying key research challenges (RCs) in NER and cloud computing, the next chapter presents the modelling and design of a system tailored to address these challenges.

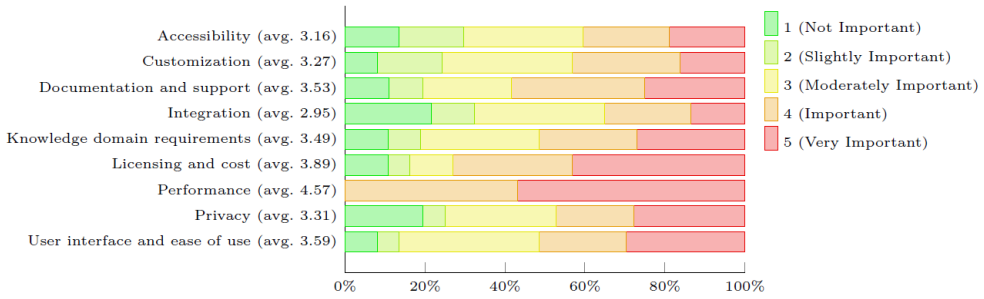

Figure 2 Priority Distribution per Hindrance Criteria

Table 2 Comparison of Average Priority per Hindrance Criteria

Hindrance	**Cloud**	**Local**	**Delta**
Challenges with integration into existing applications	2.29	1.90	0.39
Cost	3.71	1.93	1.79
Lack of documentation	2.43	2.31	0.12
Lack of support	3.00	2.21	0.79
Performance issues	2.29	2.62	-0.33
Time and effort to learn the new framework	2.43	3.00	-0.57

3. MODELLING

This section focuses on the theory-building phase, aiming to develop essential models for a system that supports medical experts in creating ML-based NER models. This is achieved by integrating the Research Challenges (RCs) outlined in Chapter 2, grounded in the latest scientific and technological advancements. **User-Centered System Design (UCSD)** [43] is employed for design and conceptual modeling, while the **Unified Modeling Language (UML)** [44] serves as the specification language. Within the FIT4NER project, Use Cases (UCs) were developed for the roles of Model Definition User, Model End User, and Administrator, alongside a generic architecture [6]. These components enable domain experts to experiment with various ML-based NER frameworks, compare their results, and select the most suitable framework for their application needs. As highlighted in Chapter 2, a survey of ML experts identified key aspects FIT4NER should support for effective framework comparison. A Master's thesis based on FIT4NER further refined the existing models to align with these requirements [45]. The subsystem developed in this thesis, termed the **Framework-Independent Layer for Training and Applying Named Entity Recognition (FILTANER)**, enhances the flexibility and adaptability of ML-based NER applications.

3.1. Use Cases

The FIT4NER Use Cases (UCs) are first analyzed and further refined, with enhancements introduced by FILTANER highlighted in magenta. As shown in Figure 3, *UC1 "Extract Data"* facilitates the retrieval of necessary data for training. *UC2 "Select NER Framework"* helps identify the most suitable framework for training NER models, ensuring a solid foundation for the training process. Since each framework comes with distinct functionalities and requirements, this selection is crucial. Building on this, *UC2.1 "Support Model Training"* assists the Model Definition User in training NER models, regardless of the chosen framework, ensuring flexibility and broad applicability.

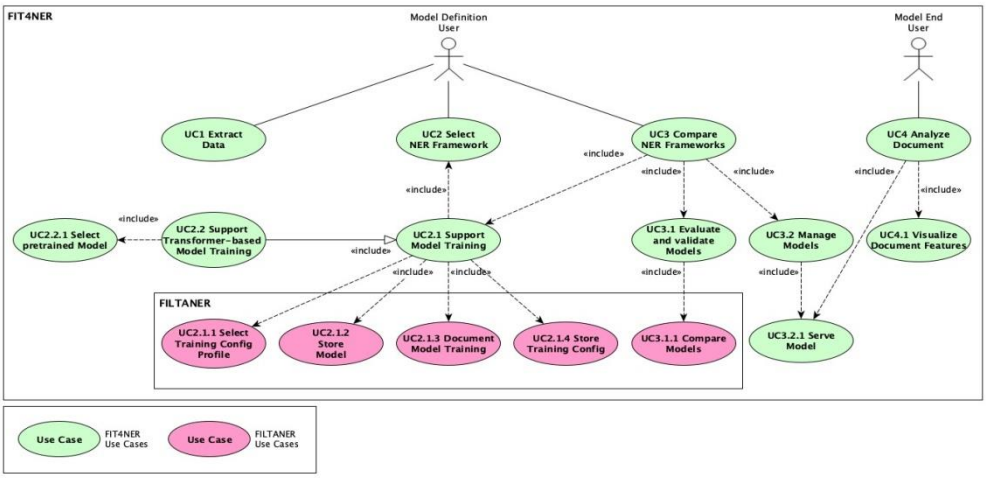

Figure 3 Use Cases for Model Definition User and Model End User

This is achieved through a **generic approach**that enables the consistent use of various frameworks without requiring specific implementation details. A key emphasis is placed on **transformer-based models**, given their central role in current research and practice. To address their unique requirements, "*UC2.2 Support Transformer-Based Model Training"* provides dedicated support for training these models. To **standardize while maintaining flexibility** in the training process, *"UC2.1.1 Select Training Config Profile"* allows users to choose from predefined configuration profiles. These profiles offer a structured way to explore different training strategies without the need for manual parameter adjustments in every training run. After a model is successfully trained, it is essential to ensure **sustainable storage and versioning**. *"UC2.1.2 "Store Model"* guarantees that trained NER models are securely stored for application, evaluation, or retraining, with a strong focus on **version control and traceability**. Additionally, documenting the entire training process is crucial for maintaining **transparency and reproducibility**. *"UC2.1.3 Document Model Training"* ensures that all relevant information is recorded, facilitating the reconstruction of complex training pipelines across different frameworks. To **preserve training configurations** for future reuse, *"UC2.1.4 Store Training Config"* saves the applied settings, enabling rapid model adjustments and optimizations without restarting from scratch. For **comparing different NER frameworks**, *"UC3 Compare NER Framework"* allows the **Model Definition User** to systematically evaluate and contrast framework performance based on the criteria outlined in **Chapter 2**. The sub-use case *"UC3.1 Evaluate and Validate Models"* ensures that trained models undergo **consistent and comparable assessments** across frameworks. Expanding FIT4NER's capabilities, *"UC3.1.1 Compare Models"* (a **FILTANER** UC) enhances model comparison by enabling direct evaluations across multiple frameworks. Using a **standardized evaluation format**, this feature allows detailed

performance assessments between models trained on different frameworks. For **efficient model organization and management**, *"UC3.2 Manage Models"* provides functionalities for saving, loading, and reusing models for further training, re-evaluation, or deployment in various applications. A key component, *"UC3.2.1 Serve Models"*, supports the **deployment of trained models** for real-world production use. Finally, *"UC4 Analyze Document"* utilizes the trained models to analyze **documents from KM-EP**, with the results visually presented in *"UC4.1 Visualize Document Features"*.

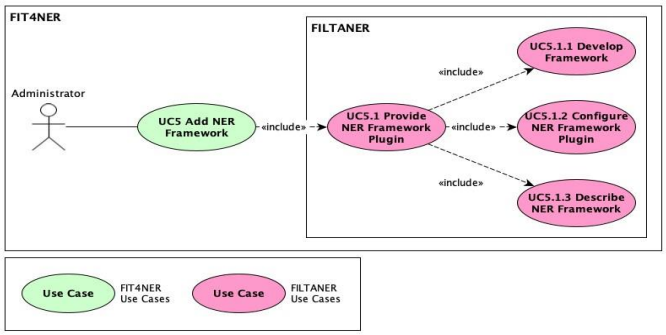

Figure 4 Use Cases for Administrator

The *"UC5 Add NER Framework"* in Figure 4 aims to integrate new frameworks. Within the scope of FILTANER, this use case is expanded with four additional UCs. *"UC5.1 Provide NER Framework Service"* which focuses on offering NER frameworks as a service, while providing functionalities such as training, evaluation, and application of NER models as independent services. The initial step, *"UC5.1.1 Develop Framework"*, involves developing the foundational structure for utilizing NER frameworks, emphasizing the definition of standards that ensure consistent integration and lay the groundwork for delivering NER framework services. *"UC5.1.2 Configure NER Framework Service"* allows for the customization and fine-tuning of the provided services, enabling administrators to make framework-specific adjustments, such as disabling transformer support. Finally, *"UC5.1.3 Describe NER Framework"* permits a detailed description of the features and functionalities of each framework. These descriptions are displayed to service consumers, providing a well-informed basis for selecting the appropriate framework.

3.2. Components

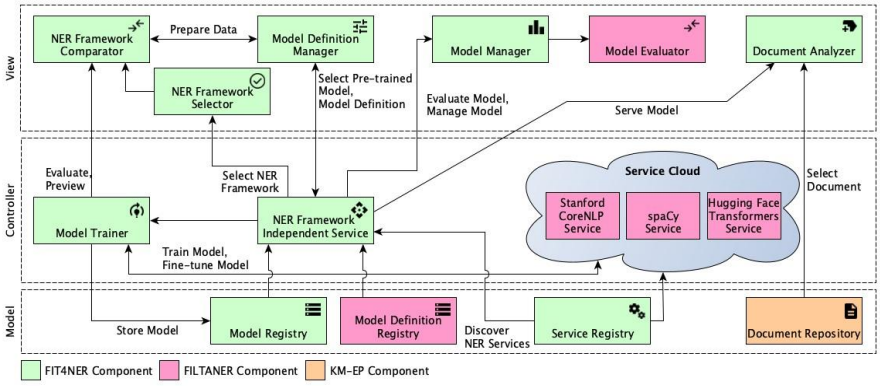

Figure 5 FIT4NER Revised General Architecture

Following the description of the UCs, this section provides a detailed overview of the FIT4NER General Architecture [6]. The components expanded within the FILTANER project are highlighted in magenta. The updated diagram (Figure 5) introduces a new component, the Model Definition Registry. This central repository allows for the storage and retrieval of specific configuration files for the NER training process. The configurations facilitate the parameterization of the training process and can be tailored to meet specific requirements. Each configuration may include metadata or tags that provide information on expected training duration and accuracy. This feature enables domain experts with limited experience in NLP to select appropriate predefined configurations for the training process. The Model Evaluator View is another new component designed to assess NER models using test datasets and key performance indicators, such as Precision, Recall, and F1. Additionally, the FILTANER project defines three specific NER framework services that can be utilized in the service cloud for training ML-based NER models. For this purpose, three NER frameworks were selected: Stanford CoreNLP [46], a well-established Java-based NER framework previously used in various projects; and spaCy [47] and Hugging Face Transformers [48], which were chosen due to their prevalent use among experts, as indicated by the prior survey [5].

Next, the core components, including their interfaces, operations, and interactions with external systems are specified. The goal is to create a modular and scalable structure that standardizes the management and deployment of trained models and configurations, enabling dynamic use of various NER frameworks. Figure 6 illustrates the detailed structures of the components, including the interfaces and operations of the services and controllers that govern communication between the components and the KM-EP. On the left side are the *ModelRegistry* and the *ModelDefinitionRegistry*, which are used for storing and providing trained NER models as well as reusable configuration files for model training. The *ModelRegistry* allows for the storage of new models via the Upload Model operation and the retrieval of stored models using Download Model and Get Models by Tag. Models can also be filtered by tags or searched by associated frameworks. The *ModelDefinitionRegistry* manages configuration files required for the training process and provides operations such as Upload Config, Download Config, Get Configs by Tag, and Get Framework Configs. The NER Framework Service provides an abstract representation of how various services for individual NER frameworks, such as Stanford CoreNLP, spaCy, and Hugging Face Transformers, are internally structured.

This service includes three key components: *ApplyNERService*, *EvaluateNERService*, and *TrainNERModelService*, which are responsible for applying trained models, evaluating models with validation data, and training or fine-tuning models, respectively. Additionally, the *RegistrationController* ensures the registration of the NER Framework Service with the NER Framework Independent Service. On the right side is the NER Framework Independent Service, which provides controllers for applying, training, and evaluating NER models centrally, thereby functioning as a middleware unit within the overall system. External systems like the KM-EP can access this service via REST interfaces to analyze documents, train models, or perform evaluations. The *UsageController* in the NER Framework Independent Service consolidates access to all NER Framework Services and provides a central interface for the KM-EP. It coordinates requests such as Apply Model for document analysis using a selected model, Train Model to initiate the training process, or Evaluate Model for model evaluation using validation data. Additionally, the status of an ongoing process can be queried through Get Job Status. The *ServiceRegistry* manages the registration and discoverability of available NER services and offers operations such as Register NER Service, Get NER Service, and Delete NER Service. An *NERService* entity stored in the *ServiceRegistry* describes a registered service through attributes like *framework_name*, *endpoint_url*, description, and *has_base_model_support*. Training, evaluation, and application processes are represented by the *TrainJob*, *EvaluationJob*, and *ApplyJob* entities. A *TrainJob* instance contains information such as *framework_name*,

config_path, *model_name*, optional *base_model_path*, and references to *train_data* and *eval_data*. The *EvaluationJob* entity stores *framework_name*, *model_path*, and *eval_data*, while the *ApplyJob* includes attributes like *framework_name*, *model_path*, and *text_file*. The status of an ongoing or completed process is indicated by the *JobStatus* entity, which includes *job_id*, *framework_name*, status, and result. This architecture facilitates a modular and scalable solution, wherein NER Framework Services are centrally accessible via the NER Framework Independent Service. The KM-EP can control all processes through well-defined interfaces. The *ModelRegistry* and *ModelDefinitionRegistry* ensure structured management of trained models and associated configurations, allowing for a dynamic application landscape and flexible extensibility to accommodate additional NER frameworks.

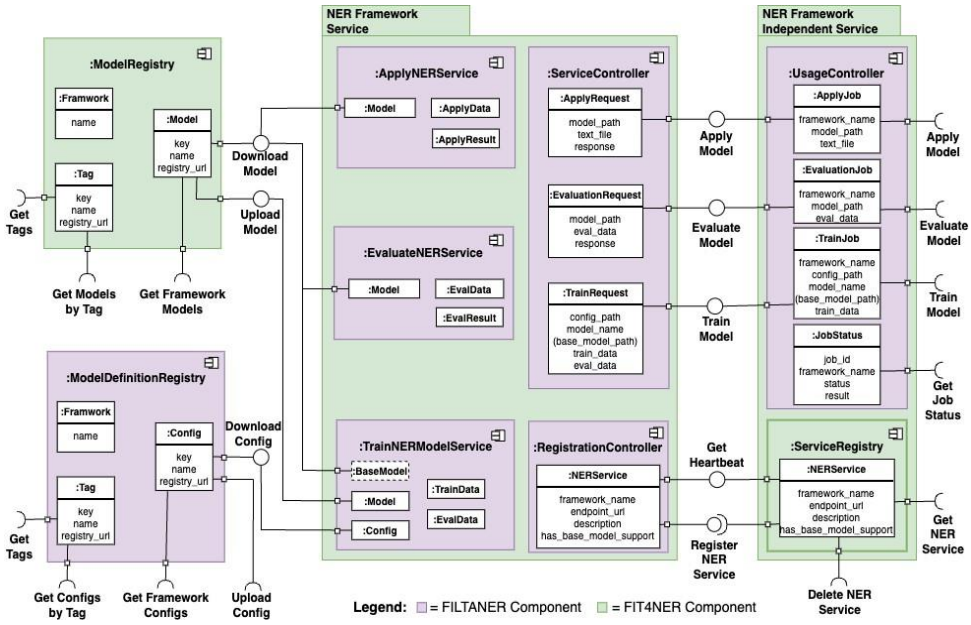

Figure 6 FIT4NER Component Diagram

4. IMPLEMENTATION

This section covers the system development phase, focusing on the research objective of building a prototype that enables medical experts to train ML-based NER models. The implementation of FIT4NER is ongoing, and this chapter details the planned user interfaces, and the current progress of system components as modelled in the previous chapter.

4.1. Graphical User Interfaces

The proposed user interfaces are designed to help domain experts make informed choices among NER frameworks. The **NER Framework Comparator View** (Figure 7) provides a structured and intuitive presentation of key frameworks, including **SpaCy, Stanford CoreNLP, and Hugging Face Transformers**, each accompanied by concise descriptions highlighting their core features and benefits. To streamline interaction, the interface features three primary action buttons: **"Train"** initiates the training of new NER models, **"Evaluate"** assesses the performance of existing models, and **"Use"** applies a trained model directly. A comprehensive tabular view presents details of previously trained models, displaying critical performance metrics such as **Precision, Recall, and F1-Score** for each model within its respective framework. Interactive buttons enable domain experts to seamlessly engage with specific models

and navigate through the system with ease. To enhance usability and navigation, the user interface employs a **consistent color-coding scheme**, where green is used for training, blue for applying models, and gray for evaluation. This uniform design approach ensures an intuitive user experience, allowing medical experts to efficiently compare, train, and deploy NER models within the FIT4NER system.

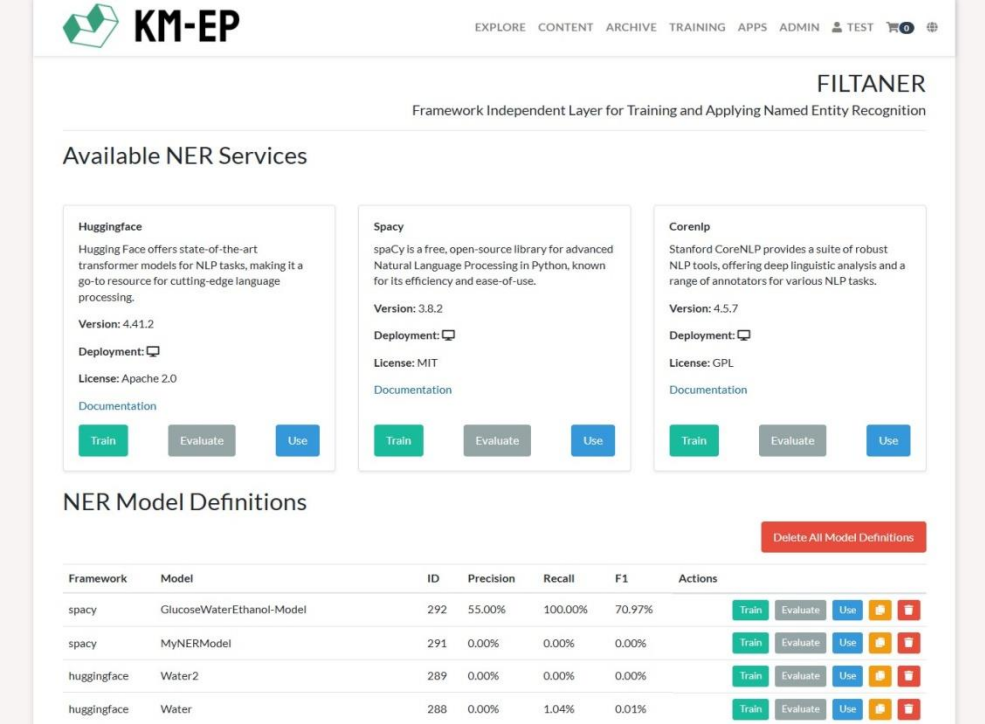

Figure 7 KM-EP NER Framework Comparator View

Designing efficient user interfaces for complex processes like NER model training requires reducing perceived complexity and ensuring intuitive usability. Blechschmitt [49] emphasizes the need to decompose complex user interfaces into smaller, manageable components to enhance user-friendliness. This approach, known as "tailoring," aims to guide domain experts step-by-step through interactions. Adaptive user interfaces, such as the use of sliders, have proven particularly effective in reducing complexity [50]. Building on this approach, model configuration is simplified through prototypical training profiles that represent various parameter combinations. Domain experts can use these profiles as starting points. The profiles are organized into dimensions such as "Fast - Medium - Accurate," "Learning Rate (low, medium, high)", or "Evaluation Strategy (none, medium, detailed)", as exemplified in the Model Definition Manager View in Figure 8 (*spacy/1-basic-config.cfg – spacy2-advanced-config.cfg*). These dimensions can be adjusted using sliders, allowing for gradual customization. Domain experts can select a base model from a dropdown menu, upload separate files for training and evaluation data, and adjust training parameters via sliders. The parameters are dynamically displayed and can be manually edited if necessary. A collapsible configuration editor ensures a tidy interface, and a prominent start button facilitates the initiation of the training process. The Model Evaluator View (Figure 9) is designed to make the assessment process clear and straightforward for domain experts. A dropdown menu allows experts to select the specific model to be evaluated, and the evaluation file can be adjusted via an edit icon. The evaluation results focus on performance metrics such as Precision, Recall, and F1-Score, as the survey described in Chapter 2 [5] identified these as the most important criteria for selecting NER frameworks. Below this, a

detailed table breaks down the results for each entity type, providing targeted insights into the model's strengths and weaknesses.

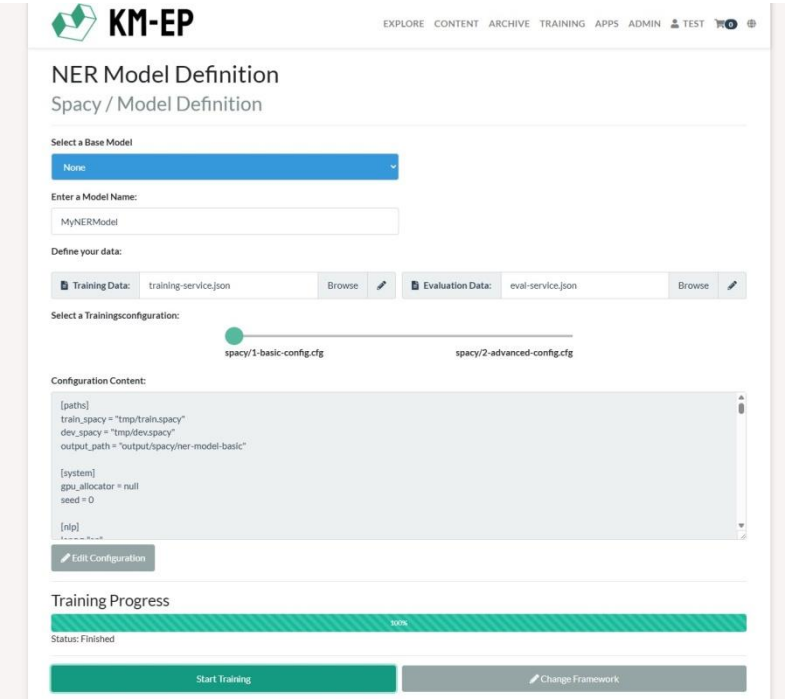

Figure 8 KM-EP Model Definition Manager View

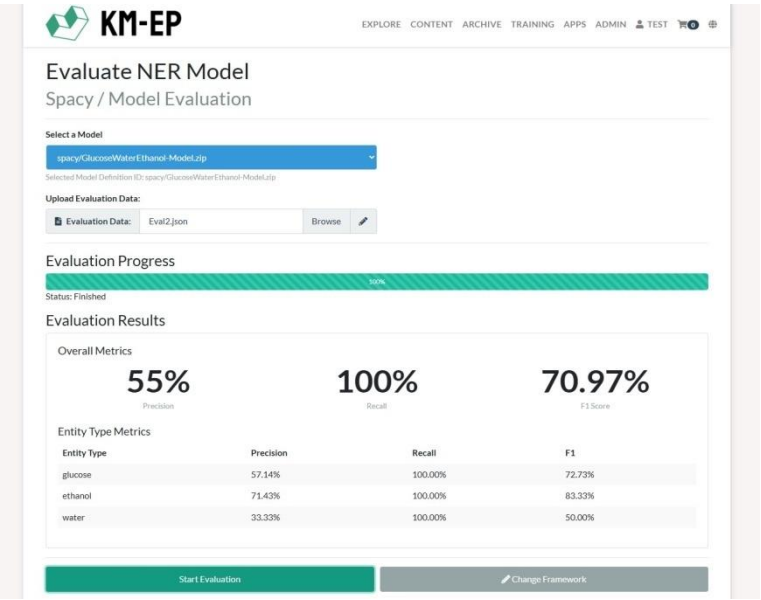

Figure 9 KM-EP Model Evaluator View

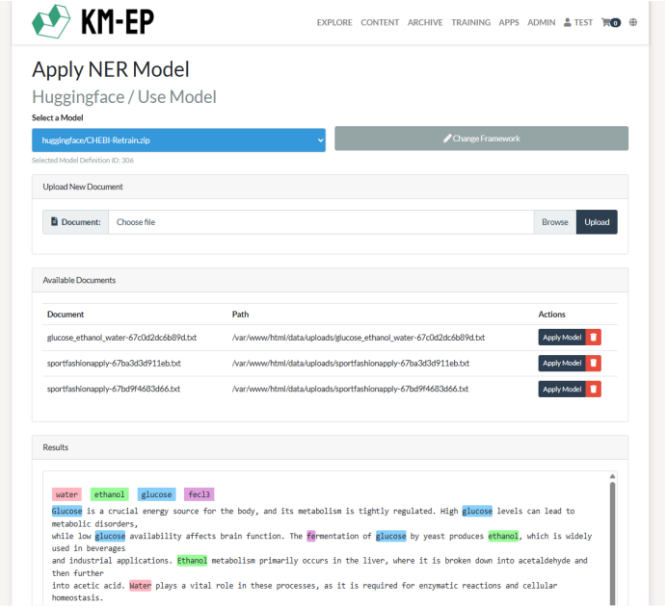

Figure 10 KM-EP Document Analyzer View

Consistent color coding is maintained: the green "Start Evaluation" button indicates an actionable step, while the "Change Framework" button allows switching to a different environment. This clear color scheme ensures intuitive navigation, helping domain experts quickly orient themselves within the interface. The Document Analyzer View intuitively allows trained models to be applied to text documents (Figure 10). In the upper section, domain experts can select the desired model, such as "spacy/GlucoseWaterEthanol-Model.zip", from a dropdown menu. Additionally, a "Change Framework" button permits switching to a different framework if needed. The central part of the interface contains a text field where domain experts can select the documents to be analyzed. With a click on the prominently visible "Apply Model" button, the model is applied to the entered text, and the results are displayed in the lower section. Recognized NEs are highlighted with colors. Each entity category has a consistent color-coding, described by legend buttons such as "Entity 1", "Entity 2", and "Entity 3". The results are directly embedded in the analyzed text, visually highlighting the recognized entities. This design allows users to quickly identify which parts of the text have been classified as specific entities. It facilitates rapid interpretation of the model results and allows for adjustments to the text if necessary.

4.2. Implementation Plan

Following the presentation of the user interfaces, the implementation plan is now detailed. Various technologies have been selected for development. Integration into KM-EP will be achieved using PHP, HTML, and JavaScript, ensuring seamless interaction with existing components. The user interfaces will communicate with the NER Framework Independent Service and the corresponding NER Framework Services via a REST API. These backend services will be implemented in Python, as it is widely supported by NER frameworks. Even Java-based frameworks like Stanford CoreNLP can be accessed via Python, ensuring broad compatibility. FastAPI will be used to generate the required REST APIs, providing high performance communication between components. A key aspect of the implementation is defining the data models, which must accommodate both the requirements of the user interfaces and the specifications of the NER services. JSON schemas will be employed to structure the data and ensure validation, facilitating consistency across the system. To enable efficient metadata

management and cloud-based storage within the Model Registry and Model Definition Registry components, an S3-compatible storage solution such as MinIO will be utilized. The implementation will also incorporate model versioning, allowing previous versions to be restored when necessary. All services will be containerized using Docker to ensure modularity and portability. During development, Docker Compose will be used for orchestration, supporting flexible local deployment and simplifying the transition to cloud-based environments. For cloud-based container management, Kubernetes will be implemented to provide scalability and fault tolerance. Comprehensive testing will be conducted throughout the development process, including unit and integration tests to validate the functionality of individual components. Automated testing will be integrated into **Continuous Integration/Continuous Deployment (CI/CD)** pipelines, which will be managed using GitLab to ensure smooth deployment workflows. As part of the system implementation, interfaces and data formats for internal communication will be clearly defined. This will include the creation of detailed API documentation to facilitate developer integration, leveraging OpenAPI specifications to provide clear and structured documentation of the API endpoints.

5. EVALUATION

The previous chapter detailed the user interfaces, and the implementation plan of a prototype aimed at assisting medical experts in training ML-based NER models. This chapter focuses on the experimentation phase and addresses the research objective of conducting and describing expert tests based on the developed prototype. It first introduces the evaluation methodologies used, followed by an assessment of the prototype's effectiveness in supporting medical experts in training ML-based NER models across various NER frameworks through expert evaluations.

Table 3 Tasks for the Cognitive Walkthrough

Task	**Description**	**Stereotype**	**Objective**
Task 1	Comparison of NER Services: Analyze available NER services and select a local (non-cloud-based) service.	Model Definition User	Identify a suitable local service and initiate NER model training.
Task 2	Selection of an NER Framework: Start training using an Advanced Configuration training profile.	Model Definition User	Evaluate system flexibility and the capability to implement specific training configurations.
Task 3	Evaluation of an NER Model: Select and assess an existing NER model from the Model Registry.	Model Definition User	Assess the performance of the selected NER model and identify potential optimization opportunities.
Task 4	Adjustment of the Training Configuration: Initiate training with a customized model definition.	Model Definition User	Evaluate the integration of individual training configurations and their impact.
Task 5	Document Analysis: Choose an NER model for document analysis.	Model End User	Conduct and evaluate the document analysis.

A **Cognitive Walkthrough (CW)**was selected as the qualitative evaluation approach, a method for assessing user interfaces as described by Polsen et al. . A CW involves simulating and analyzing the cognitive processes of a user interacting with the interface. During the preparation

phase, tasks with specific objectives and required action sequences are defined. Subsequently, a panel of experts evaluates the anticipated user interaction with the interface, considering criteria such as action availability and labeling, the likelihood of correct action selection, and action complexity. For the initial CW evaluation of the user interfaces, a meeting was conducted with one PhD, three doctoral candidates, and one master's student, all of whom are researching in the fields of NLP and NER. Future CW sessions are planned with medical professionals to assess usability outside the realm of computer science. For this meeting, five tasks were prepared (Table 3), covering UC2, UC3, and UC4 (Figure 3), including their respective sub-use cases. The tasks were carried out in a development environment that provided the user interfaces, although not all functionalities were fully implemented. The evaluation findings are presented in Table 4. In Task 1, the NER services available in the GUI (Figure 7) were initially compared. The locally provisioned service spaCy was selected because no cloud-based services are integrated at the current stage of development. The Model Definition Manager View (Figure 8) then opened for training an NER model. Experts provided feedback that the interface texts and descriptions should be expanded and improved to facilitate use by domain experts (Finding 1 and 3). Additionally, implementing version control for trained models would be beneficial, allowing models trained with the same data but different parameters to be compared (Finding 2). Following the training, the model was evaluated in Task 2 within the Model Evaluator View (Figure 9). It was noted that users were unclear about which data and formats should be used when selecting a test dataset. A more detailed presentation of the data formats and datasets could improve this (Finding 4). During the retraining of an NER model with a modified model definition (Task 4), experts observed several points. It would be advantageous to upload custom pre-trained models to the Model Registry, as currently only system-trained models are available (Finding 5). The model training configuration should be automatically saved to document the configuration used (Finding 6). Finally, it should be possible to reuse the configuration of previous trainings as a basis for new ones (Finding 7). There were no additional comments during the final document analysis (Task 5).

Table 4 Evaluation Results of the Cognitive Walkthrough

Finding	Task	Description
Finding 1	Task 2	Enhance information delivery by providing explanatory texts for terms such as Model Name and Base Model, as well as explanations of the functions of respective GUI elements.
Finding 2	Task 2	Implement version control in the model registry to prevent overwriting models and to archive previous training states.
Finding 3	Task 2	Expand the configuration GUI with meaningful descriptions of slider step values to make the impact of parameter changes through training profiles more transparent.
Finding 4	Task 3	Provide a more detailed representation of the data formats and datasets used in the evaluation of NER models.
Finding 5	Task 4	Implement a manual upload option for models into the model registry.
Finding 6	Task 4	Automatically save the model definition used in the Model Definition Registry to document the training process.
Finding 7	Task 4	Integrate a copy button for reusing relevant parameters/model definition and base model for model training.

This chapter explained how the developed user interfaces of FIT4NER were qualitatively evaluated by NLP experts. Seven findings were identified, which will be addressed in further research. The prototype was considered a suitable foundation during the session to support

domain experts in selecting and comparing NER frameworks and will therefore be further developed in future work. Additionally, further quantitative evaluation experiments are planned, along with an additional qualitative evaluation of the system with medical experts.

6. CONCLUSION

This article presented a survey that investigates how ML experts compare and select NER frameworks. Based on the survey results, a system was designed and evaluated using the structured Nunamaker methodology for developing information systems [18], [51]. This system aims to assist medical experts in comparing and selecting NER frameworks, as well as in training ML-based NER models.

Chapter 1 provides an overview of the topic and situates the research within its relevant context, laying the groundwork for the subsequent analysis. Chapter 2 conducted a comprehensive review of the current state of the art by planning and executing an expert survey. The survey highlighted that performance is a particularly important criterion. Furthermore, expert opinions varied significantly. All specified selection criteria were regarded important or very important at least once. This indicates that the relevance of the criteria may vary depending on the project. Therefore, a supportive system should be flexible enough to accommodate various criteria along with performance. A distinction is made between cloud-based services and locally installed tools and frameworks. For cloud-based services, cost and user-friendliness are particularly significant. Both aspects represent important requirements for a system that uses cloud-based services for NER. In the case of locally installed systems, the effort required for users to adopt a new system should be minimized, which can be facilitated by an appropriate software solution. These remaining challenges are addressed in Chapter 3, which lays the foundation for the design of an information system aimed at assisting medical experts in the comparison and selection of ML-based NER frameworks, as well as in training NER models using these frameworks. Chapter 3 presents the modeling of this system, named FIT4NER. In Chapter 4, the graphical user interfaces of FIT4NER are developed and showcased, thereby achieving the objectives of the system development. Chapter 5 discusses the experimental goals and evaluates, through expert-supported evaluation experiments, the extent to which the developed user interfaces are suitable for supporting domain experts. During these experiments, areas for future work were identified, including addressing the findings from the expert evaluation of the FIT4NER interfaces. The prototype will be further developed to better assist domain experts in selecting NER frameworks. Additionally, further quantitative and qualitative evaluations are planned, including those with medical experts.

In summary, this study successfully addressed the defined research questions and resolved the challenges identified in Chapter 2. The results highlight the potential of the prototype to support NER model training for medical professionals. Future experiments will involve additional sessions with medical experts to assess the effectiveness of FIT4NER in developing ML-based NER models. The contributions of this work provide a solid foundation for advancements in the field of NER within medical informatics.

REFERENCES

[1] F. Freund, P. Tamla, and M. Hemmje, “Towards improving clinical practice guidelines through named entity recognition: Model development and evaluation,” in *2023 31st irish conference on artificial intelligence and cognitive science (AICS)*, 2023, pp. 1–8. doi: 10.1109/AICS60730.2023.10470480.

[2] E. Steinberg, S. Greenfield, D. M. Wolman, M. Mancher, R. Graham, and others, *Clinical practice guidelines we can trust*. national academies press, 2011.

[3] T. S. Valika, S. E. Maurrasse, and L. Reichert, "A Second Pandemic? Perspective on Information Overload in the COVID-19 Era," *Otolaryngol. Neck Surg.*, vol. 163, no. 5, pp. 931–933, Nov. 2020, doi: 10.1177/0194599820935850.

[4] K. Pakhale, "Comprehensive Overview of Named Entity Recognition: Models, Domain-Specific Applications and Challenges," Sep. 2023, *arXiv*. Accessed: Nov. 14, 2024. [Online]. Available: http://arxiv.org/abs/2309.14084

[5] F. Freund, P. Tamla, and M. Hemmje, "Survey: Understand the challenges of Machine Learning Experts using Named Entity Recognition Tools," in *Computer Science*, Coppenhagen: AIRCC, Jan. 2024, pp. 115–134. doi: 10.5121/csit.2024.150208.

[6] F. Freund, P. Tamla, T. Reis, M. Hemmje, and P. M. Kevitt, "FIT4NER - Towards a FrameworkIndependent Toolkit for Named Entity Recognition," in *Proceedings CERC 2023*, Barcelona: Hochschule Darmstadt, 2023. doi: https://doi.org/10.48444/h_docs-pub-518.

[7] Bielefeld University, "RATIO: Rationalizing Recommendations (RecomRatio)." Accessed: Aug. 06, 2024. [Online]. Available: https://spp-ratio.de/projects/recomratio/

[8] C. Nawroth, "Supporting Information Retrieval of Emerging Knowledge and Argumentation," phd, FernUniversität in Hagen, Hagen, 2020.

[9] FTK, "Artificial Intelligence for Hospitals, Healthcare & Humanity (AI4H3)," FTK e.V. Research Institute for Telecommunications and Cooperation, Dortmund, Germany, R&D White Paper, Apr. 2020.

[10] P. Tamla, B. Hartmann, N. Nguyen, C. Kramer, F. Freund, and M. Hemmje, "CIE: a cloud-based information extraction system for named entity recognition in AWS, azure, and medical domain," in *Knowledge discovery, knowledge engineering and knowledge management*, F. Coenen, A. Fred, D. Aveiro, J. Dietz, J. Bernardino, E. Masciari, and J. Filipe, Eds., Cham: Springer Nature Switzerland, 2023, pp. 127–148.

[11] B. Vu *et al.*, "A metagenomic content and knowledge management ecosystem platform," in *2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, IEEE, 2019, pp. 1–8.

[12] M. Hemmje, "Chair of Multimedia and Internet Applications." [Online]. Available: http://www.lgmmia.fernuni-hagen.de/en.html

[13] FTK, "FTK e.V. Research Institute for Telecommunications and Cooperation." Accessed: Feb. 25, 2023. [Online]. Available: https://www.ftk.de/en

[14] P. Tamla, "Supporting Access to Textual Resources Using Named Entity Recognition and Document Classification," 2022.

[15] C. Wen, T. Chen, X. Jia, and J. Zhu, "Medical Named Entity Recognition from Un-labelled Medical Records based on Pre-trained Language Models and Domain Dictionary," *Data Intell.*, vol. 3, no. 3, pp. 402–417, Sep. 2021, doi: 10.1162/dint_a_00105.

[16] F. Giachelle, O. Irrera, and G. Silvello, "MedTAG: a portable and customizable annotation tool for biomedical documents," *BMC Med. Inform. Decis. Mak.*, vol. 21, no. 1, p. 352, Dec. 2021, doi: 10.1186/s12911-021-01706-4.

[17] K. Weiying, D. N. Pham, Y. Eftekharypour, and A. J. Pheng, "Benchmarking NLP Toolkits for Enterprise Application," in *PRICAI 2019: Trends in Artificial Intelligence*, A. C. Nayak and A. Sharma, Eds., in Lecture Notes in Computer Science. Cham: Springer International Publishing, 2019, pp. 289–294. doi: 10.1007/978-3-030-29894-4_24.

[18] J. F. Nunamaker Jr, M. Chen, and T. D. M. Purdin, "Systems Development in Information Systems Research," *J. Manag. Inf. Syst.*, vol. 7, no. 3, pp. 89–106, Dec. 1990, doi: 10.1080/07421222.1990.11517898.

[19] B. Jehangir, S. Radhakrishnan, and R. Agarwal, "A survey on Named Entity Recognition — datasets, tools, and methodologies," *Nat. Lang. Process. J.*, vol. 3, p. 100017, Jun. 2023, doi: 10.1016/j.nlp.2023.100017.

[20] D. Petkova and W. B. Croft, "Proximity-based document representation for named entity retrieval," in *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, Lisbon Portugal: ACM, Nov. 2007, pp. 731–740. doi: 10.1145/1321440.1321542.

[21] R. Barskar, G. F. Ahmed, and N. Barskar, "An Approach for Extracting Exact Answers to Question Answering (QA) System for English Sentences," *Procedia Eng.*, vol. 30, pp. 1187–1194, 2012, doi: 10.1016/j.proeng.2012.01.979.

[22] Y. Marton and I. Zitouni, "Transliteration normalization for information extraction and machine translation," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 26, no. 4, pp. 379–387, 2014.

[23] J. Kim, Y. Kim, and S. Kang, "Weakly labeled data augmentation for social media named entity recognition," *Expert Syst. Appl.*, vol. 209, p. 118217, Dec. 2022, doi: 10.1016/j.eswa.2022.118217.

[24] N. S. Pagad and N. Pradeep, "Clinical named entity recognition methods: an overview," in *International conference on innovative computing and communications*, 2022, pp. 151–165.

[25] J. Li, A. Sun, J. Han, and C. Li, "A Survey on Deep Learning for Named Entity Recognition," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 1, pp. 50–70, Jan. 2022, doi: 10.1109/TKDE.2020.2981314.

[26] I. M. Konkol, "Named entity recognition," PhD Thesis, PhD thesis, University of West Bohemia, 2015.

[27] Ariruna Dasgupta and Asoke Nath, "Classification of Machine Learning Algorithms," *Figshare*, 2016, doi: 10.6084/m9.figshare.3504194.v1.

[28] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *ArXiv Prepr. ArXiv181004805*, 2018.

[29] A. Radford *et al.*, "Language models are unsupervised multitask learners," *OpenAI Blog*, vol. 1, no. 8, p. 9, 2019.

[30] Y. Liu *et al.*, "Roberta: A robustly optimized bert pretraining approach," *ArXiv Prepr. ArXiv190711692*, 2019.

[31] S. Achar, "Cloud-based system design," *Int. J. Res. Educ. Sci. Methods IJARESM*, vol. 7, no. 8, pp. 23–30, 2019.

[32] W. Kim, "Cloud Computing: Today and Tomorrow," *CLOUD Comput.*, vol. 8, no. 1, p. 8, 2009.

[33] M. Blohm, C. Dukino, M. Kintz, M. Kochanowski, F. Koetter, and T. Renner, "Towards a Privacy Compliant Cloud Architecture for Natural Language Processing Platforms:," in *Proceedings of the 21st International Conference on Enterprise Information Systems*, Heraklion, Crete, Greece: SCITEPRESS - Science and Technology Publications, 2019, pp. 454–461. doi: 10.5220/0007746204540461.

[34] A. Pinto, H. G. Oliveira, and A. O. Alves, "Comparing the performance of different NLP toolkits in formal and social media text," in *5th symposium on languages, applications and technologies (SLATE'16)*, M. Mernik, J. P. Leal, and H. G. Oliveira, Eds., in OpenAccess series in informatics (OASIcs), vol. 51. Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2016, p. 3:1-3:16. doi: 10.4230/OASIcs.SLATE.2016.3.

[35] F. N. A. Al Omran and C. Treude, "Choosing an NLP Library for Analyzing Software Documentation: A Systematic Literature Review and a Series of Experiments," in *2017 IEEE/ACM 14th International Conference on Mining Software Repositories (MSR)*, Buenos Aires, Argentina: IEEE, May 2017, pp. 187–197. doi: 10.1109/MSR.2017.42.

[36] M. Won, P. Murrieta-Flores, and B. Martins, "Ensemble Named Entity Recognition (NER): Evaluating NER Tools in the Identification of Place Names in Historical Corpora," *Front. Digit. Humanit.*, vol. 5, p. 2, Mar. 2018, doi: 10.3389/fdigh.2018.00002.

[37] X. Schmitt, S. Kubler, J. Robert, M. Papadakis, and Y. LeTraon, "A Replicable Comparison Study of NER Software: StanfordNLP, NLTK, OpenNLP, SpaCy, Gate," in *2019 Sixth International Conference on Social Networks Analysis, Management and Security (SNAMS)*, Granada, Spain: IEEE, Oct. 2019, pp. 338–343. doi: 10.1109/SNAMS.2019.8931850.

[38] A. Aldumaykhi, S. Otai, and A. Alsudais, "Comparing open arabic named entity recognition tools," 2022, [Online]. Available: https://arxiv.org/abs/2205.05857

[39] M. Kasunic, *Designing an Effective Survey*. Citeseer, 2005.

[40] A. Casey *et al.*, "A systematic review of natural language processing applied to radiology reports," *BMC Med. Inform. Decis. Mak.*, vol. 21, no. 1, p. 179, Dec. 2021, doi: 10.1186/s12911-021-01533-7.

[41] H. A. Kurdi, S. Hamad, and A. Khalifa, "Towards a Friendly User Interface on the Cloud," in *Design, User Experience, and Usability. User Experience Design for Diverse Interaction Platforms and Environments*, vol. 8518, D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, A. Kobsa, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, D. Terzopoulos, D. Tygar, G. Weikum, and A. Marcus, Eds., Cham: Springer International Publishing, 2014, pp. 148–157. doi: 10.1007/978-3-319-07626-3_14.

[42] R. Chard, K. Chard, K. Bubendorfer, L. Lacinski, R. Madduri, and I. Foster, "Cost-aware cloud provisioning," in *2015 IEEE 11th international conference on e-Science*, 2015, pp. 136–144.

[43] D. A. Norman and S. W. Draper, *User Centered System Design; New Perspectives on HumanComputer Interaction*. USA: L. Erlbaum Associates Inc., 1986.

[44] L. Jacobson and J. R. G. Booch, *The unified modeling language reference manual*. 2021.
[45] F. Wilde, "Entwicklung einer Microservice-basierten Abstraktionsebene für das Frameworkunabhängige Training von Named Entity Recognition in einem Wissensmanagement-System für den medizinischen Bereich," Masterthesis, FernUniversität in Hagen, Hagen, 2025.
[46] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky, "The Stanford CoreNLP Natural Language Processing Toolkit," in *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, Baltimore, Maryland: Association for Computational Linguistics, Jun. 2014, pp. 55–60. doi: 10.3115/v1/P14-5010.
[47] M. Honnibal, I. Montani, S. Van Landeghem, and A. Boyd, "spaCy: Industrial-strength natural language processing in python," *Zenodo*, 2020.
[48] S. M. Jain, *Introduction to Transformers for NLP: With the Hugging Face Library and Models to Solve Problems*. Berkeley, CA: Apress, 2022. doi: 10.1007/978-1-4842-8844-3.
[49] E. Blechschmitt, "Adaptive mensch-maschine interaktion für mobile agenten," phd, Technische Universität Darmstadt, 2005.
[50] J. J. Dudley and P. O. Kristensson, "A review of user interface design for interactive machine learning," *ACM Trans. Interact. Intell. Syst. TiiS*, vol. 8, no. 2, pp. 1–37, 2018.
[51] P. G. Polson, C. Lewis, J. Rieman, and C. Wharton, "Cognitive walkthroughs: a method for theorybased evaluation of user interfaces," *Int. J. Man-Mach. Stud.*, vol. 36, no. 5, pp. 741–773, May 1992, doi: 10.1016/0020-7373(92)90039-N.